

CRANFIELD UNIVERSITY

ASHUTOSH TIWARI

***Evolutionary Computing Techniques for Handling
Variable Interaction in Engineering Design Optimisation***

SCHOOL OF INDUSTRIAL AND MANUFACTURING SCIENCE

PhD

ProQuest Number: 10820895

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 10820895

Published by ProQuest LLC (2019). Copyright of the Dissertation is held by Cranfield University.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

CRANFIELD UNIVERSITY

SCHOOL OF INDUSTRIAL AND MANUFACTURING SCIENCE

DEPARTMENT OF ENTERPRISE INTEGRATION

PHD

Academic Year 2001-2002

ASHUTOSH TIWARI

***Evolutionary Computing Techniques for Handling
Variable Interaction in Engineering Design Optimisation***

Supervisors: Dr. Rajkumar Roy and Mr. Graham Jared

November 2001

This thesis is submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

© Cranfield University, 2001. All rights reserved. No part of this publication may
be reproduced without the written permission of the copyright owner.

Abstract

The ever-increasing market demands to produce better products, with reduced costs and lead times, has prompted the industry to look for rigorous ways of optimising its designs. However, the lack of flexibility and adequacy of existing optimisation techniques in dealing with the challenges of engineering design optimisation, has prevented the industry from using optimisation algorithms. The aim of this research is to explore the field of evolutionary computation for developing techniques that are capable of dealing with three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction.

An industry survey grounds the research within the industrial context. A literature survey of EC techniques for handling multiple objectives, constraints and variable interaction highlights a lack of techniques to handle variable interaction. This research, therefore, focuses on the development of techniques for handling variable interaction in the presence of multiple objectives and constraints. It attempts to fill this gap in research by formally defining and classifying variable interaction as inseparable function interaction and variable dependence. The research then proposes two new algorithms, GRGA and GAVD, that are respectively capable of handling these types of variable interaction.

Since it is difficult to find a variety of real-life cases with required complexities, this research develops two test beds (RETB and RETB-II) that have the required features (multiple objectives, constraints and variable interaction), and enable controlled testing of optimisation algorithms. The performance of GRGA and GAVD is analysed and compared to the current state-of-the-art optimisation algorithm (NSGA-II) using RETB, RETB-II and other ‘popular’ test problems.

Finally, a set of real-life optimisation problems from literature are analysed from the point of variable interaction. The performance of GRGA and GAVD is finally validated using three appropriately chosen problems from this set. In this way, this research proposes a fully tested and validated methodology for dealing with engineering design optimisation problems with variable interaction.

Acknowledgements

I would like to take this opportunity to thank my principal supervisor, Dr. Rajkumar Roy, for all his dedicated support and guidance throughout this research. His guidance and inspiration are the key factors that enabled successful completion of this research. Thanks also to my second supervisor, Mr. Graham Jared, for all his help and valuable guidance in this research.

I would also like to thank the funding body of this research: Engineering and Physical Sciences Research Council (EPSRC), and its industrial collaborators: Nissan Technical Centre – Europe (NTC-E) and Structural Dynamics Research Corporation (SDRC).

I am also grateful to my friend Mr. Christopher Rush for providing valuable suggestions and all the help I needed. I would like to thank all the members of the Decision Engineering Group (DEG) for the useful discussions we had throughout the last year. Thanks also to all the staff of the Department of Enterprise Integration (DEI) for their support.

Finally, I would like to thank my parents and family for their love and support.

List of Publications

- Rogero, J.M., Tiwari, A., Munaux, O., Rubini, P.A., Roy, R. and Jared, G. (2000). Applications of evolutionary algorithms for solving real-life design optimisation problems. In: *6th International Conference on Parallel Problem Solving from Nature (PPSN-VI) Workshop on Real-life Evolutionary Design Optimisation*, Paris (France), 16 September.
- Roy, R., Tiwari, A. and Braneby, A. (2001). Making evolutionary design optimisation popular in industry: Issues and techniques. In: *6th Online World Conference on Soft Computing in Industrial Applications (WSC-6)*, Cyberspace, 10-28 September.
- Roy, R., Tiwari, A., Munaux, O. and Jared, G. (2000). Real-life engineering design optimisation: Features and techniques. In: Martikainen, J. and Tanskanen, J. (eds.). *CDROM Proceedings of the 5th Online World Conference on Soft Computing in Industrial Applications (WSC-5) - ISBN 951-22-5205-8*, IEEE, Finland.
- Tiwari, A. and Roy, R. (2002). Variable dependence interaction and multi-objective optimisation. Accepted for publication: *Genetic and Evolutionary Computation Conference (GECCO-2002)*, New York (USA), 9-13 July.
- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001a). Interaction and multi-objective optimisation. In: Spector, L., Goodman, E., Wu, A., Langdon, W.B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. and Burke, E. (eds.). *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp. 671-678, San Francisco (USA), 7-11 July.
- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001b). Challenges in real-life engineering design optimisation: An analysis. In: *Genetic and Evolutionary Computation Conference (GECCO-2001) Workshop on Real-life Evolutionary Design Optimisation*, pp. 289-294, San Francisco (USA), 7 July.
- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001c). Evolutionary-based techniques for real-life optimisation: Development and testing. Accepted for publication: *Applied Soft Computing (ASC) Journal*, Elsevier Science (Netherlands).

Contents

1	INTRODUCTION	1
1.1	DEFINITION AND CLASSIFICATION OF ENGINEERING DESIGN OPTIMISATION PROBLEMS.....	1
1.1.1	<i>Based on Number of Variables.....</i>	2
1.1.2	<i>Based on Existence of Constraints</i>	2
1.1.3	<i>Based on Number of Objective Functions</i>	3
1.1.4	<i>Based on Nature of Objective Functions</i>	3
1.1.5	<i>Based on Separability of Functions.....</i>	3
1.1.6	<i>Based on Dependence among Variables</i>	4
1.1.7	<i>Based on Nature of Search Space.....</i>	5
1.1.8	<i>Miscellaneous Classifications</i>	5
1.2	INTRODUCTION TO EC TECHNIQUES AS EFFICIENT OPTIMISERS	7
1.3	INTRODUCTION TO GAS	8
1.4	'FLEXO': PARENT PROJECT OF THIS RESEARCH.....	9
1.4.1	<i>Previous Work</i>	10
1.4.2	<i>Brief Description of 'FLEXO'</i>	10
1.4.3	<i>'FLEXO' Methodology.....</i>	12
1.5	PROBLEM STATEMENT AND MOTIVATION	12
1.6	THESIS LAYOUT.....	14
2	A REVIEW OF LITERATURE	17
2.1	ENGINEERING DESIGN OPTIMISATION APPROACHES	17
2.1.1	<i>Manual versus Algorithmic Approaches to Optimisation.....</i>	18
2.1.2	<i>Algorithmic Approaches to Optimisation</i>	18
2.1.3	<i>Optimisation Approaches to Handle Uncertainty</i>	20
2.1.4	<i>Optimisation Approaches involving FEA/CFD.....</i>	22
2.1.5	<i>Sensitivity Analysis.....</i>	22
2.1.6	<i>Engineering Design Support System</i>	24
2.2	EVOLUTIONARY-BASED TECHNIQUES FOR MULTI-OBJECTIVE OPTIMISATION	25
2.2.1	<i>Problem Statement.....</i>	25
2.2.2	<i>Classical versus Evolutionary Approaches</i>	27
2.2.3	<i>Classification of EMOTs</i>	27
2.2.3.1	<i>Function-based Classification</i>	28
2.2.3.2	<i>User-based Classification</i>	29
2.2.3.3	<i>Hybrid Classification.....</i>	29
2.2.4	<i>Plain Aggregating A Priori Approaches</i>	31

2.2.5	<i>Population-based Non-Pareto A Posteriori Approaches</i>	32
2.2.6	<i>Pareto-based A Posteriori Approaches</i>	32
2.2.7	<i>Examples of EMOTs</i>	33
2.2.7.1	Thermo-Dynamical GA (TDGA)	33
2.2.7.2	Strength Pareto Evolutionary Algorithm (SPEA)	34
2.2.7.3	Fast Elitist Non-dominated Sorting GA (NSGA-II)	35
2.2.8	<i>Summary</i>	35
2.3	EVOLUTIONARY-BASED CONSTRAINED OPTIMISATION TECHNIQUES	36
2.3.1	<i>Problem Statement</i>	36
2.3.2	<i>Classical versus Evolutionary Approaches</i>	37
2.3.3	<i>Classification of ECOTs</i>	37
2.3.3.1	Methods based on Preserving Feasibility of Solutions	38
2.3.3.2	Methods based on Penalty Functions	39
2.3.3.3	Methods based on Feasible over Infeasible Solutions	40
2.3.3.4	Methods based on Decoders	41
2.3.3.5	Hybrid Methods	41
2.3.4	<i>Jimenez-Verdegay-Gomez-Skarmeta's Method</i>	42
2.3.5	<i>Constrained Tournament Method</i>	43
2.3.6	<i>Ray-Tai-Seow's Method</i>	44
2.3.7	<i>Summary</i>	44
2.4	EVOLUTIONARY-BASED TECHNIQUES FOR HANDLING INSEPARABLE FUNCTION INTERACTION	45
2.4.1	<i>Problem Statement</i>	46
2.4.2	<i>Classical versus Evolutionary Approaches</i>	48
2.4.3	<i>Classification of ETIFIs</i>	48
2.4.4	<i>Managing Race between Linkage Evolution and Allele Selection</i>	50
2.4.4.1	Methods that Manipulate Representation of Solutions	50
2.4.4.2	Methods that Use Specialised Reproduction Operators	51
2.4.4.3	Methods that Avoid Race between Linkage Evolution and Allele Selection	51
2.4.5	<i>Modelling Promising Solutions</i>	52
2.4.5.1	No Interaction	53
2.4.5.2	Pairwise Interaction	53
2.4.5.3	Multiple Interaction	54
2.4.6	<i>Examples of ETIFIs</i>	55
2.4.6.1	Continuous-domain EDA using a Flexible Probability Density Estimator	56
2.4.6.2	Multi-objective Mixture-based Iterated Density Estimation Evolutionary Algorithm (MIDEA)	56
2.4.7	<i>Summary</i>	57
2.5	EVOLUTIONARY-BASED TECHNIQUES FOR HANDLING VARIABLE DEPENDENCE	57

2.5.1	<i>Problem Statement</i>	57
2.5.2	<i>Classical versus Evolutionary Approaches</i>	59
2.5.3	<i>Classification of ETVD</i>	60
2.5.4	<i>Techniques for Step 1: Identification of Dependency Relationships</i>	61
2.5.4.1	Regression Analysis (RA).....	62
2.5.4.2	Neural Networks (NNs).....	63
2.5.4.3	Probabilistic Modelling (PM).....	65
2.5.5	<i>Techniques for Step 2: Classification of Variables</i>	66
2.5.5.1	Tree Diagrams (TDs).....	67
2.5.5.2	Direct Analysis (DA).....	67
2.5.6	<i>Summary</i>	68
2.6	TEST PROBLEMS FOR SIMULATING MULTI-OBJECTIVE OPTIMISATION.....	69
2.6.1	<i>Non-tuneable Test Problems</i>	70
2.6.2	<i>Tuneable Test Problems</i>	71
2.6.3	<i>Summary</i>	75
2.7	TEST PROBLEMS FOR SIMULATING CONSTRAINED MULTI-OBJECTIVE OPTIMISATION	75
2.7.1	<i>Non-tuneable Test Problems</i>	76
2.7.2	<i>Tuneable Test problems</i>	76
2.7.3	<i>Summary</i>	79
2.8	TEST PROBLEMS FOR VARIABLE INTERACTION.....	80
2.8.1	<i>Test Problems for Inseparable Function Interaction</i>	80
2.8.2	<i>Test Problems for Variable Dependence</i>	81
2.8.3	<i>Summary</i>	81
2.9	SUMMARY.....	81
3	RESEARCH AIM, OBJECTIVES AND METHODOLOGY	83
3.1	RESEARCH AIM.....	83
3.2	RESEARCH OBJECTIVES	83
3.3	RESEARCH SCOPE	84
3.4	RESEARCH METHODOLOGY	85
3.4.1	<i>Problem Identification</i>	85
3.4.2	<i>Literature Survey</i>	85
3.4.3	<i>Identification of Research Aim, Objectives and Focus</i>	86
3.4.4	<i>Industry Survey</i>	86
3.4.5	<i>Development of EC Techniques</i>	86
3.4.6	<i>Development of Test Bed</i>	87
3.4.7	<i>Performance Analysis Using the Proposed Test Bed</i>	87
3.4.8	<i>Validation Using Real-life Case Studies</i>	87

3.4.9	<i>Identification of Limitations and Future Research Directions</i>	88
3.5	SUMMARY	88
4	INDUSTRIAL CONTEXT AND FOCUS	89
4.1	INDUSTRIAL SURVEY	89
4.1.1	<i>Aim and Objectives</i>	90
4.1.2	<i>Methodology</i>	90
4.1.2.1	Method of Approach to Respondents	90
4.1.2.2	Determination of Question Sequence	91
4.1.2.3	Types of Questions	91
4.1.2.4	Analysis of Responses	92
4.2	DESIGN IMPROVEMENT IN INDUSTRY	92
4.2.1	<i>Industrial Applications of Optimisation Algorithms</i>	92
4.2.2	<i>Optimisation Algorithms in Applied Research</i>	93
4.3	FEATURES OF REAL-LIFE ENGINEERING DESIGN OPTIMISATION PROBLEMS	94
4.3.1	<i>Features</i>	94
4.3.2	<i>An Example of a Real-life Design Optimisation Problem</i>	95
4.4	INDUSTRIAL CONTEXT OF THE RESEARCH	96
4.4.1	<i>Inhibitors to Industrial Applications of Optimisation Algorithms</i>	96
4.4.2	<i>Industrial Context of the Research</i>	98
4.5	RESEARCH FOCUS FOR DEVELOPMENT OF EC TECHNIQUES	98
4.5.1	<i>Multi-objective Optimisation</i>	98
4.5.2	<i>Constrained Multi-objective Optimisation</i>	99
4.5.3	<i>Variable Interaction</i>	99
4.5.3.1	Inseparable Function Interaction	100
4.5.3.2	Variable Dependence	100
4.5.4	<i>Research Focus</i>	101
4.6	RESEARCH FOCUS FOR DEVELOPMENT OF OPTIMISATION TEST BEDS	101
4.6.1	<i>Multi-objective Optimisation</i>	101
4.6.2	<i>Constrained Multi-objective Optimisation</i>	102
4.6.3	<i>Variable Interaction</i>	102
4.6.3.1	Inseparable Function Interaction	102
4.6.3.2	Variable Dependence	102
4.6.4	<i>Research Focus</i>	102
4.7	SUMMARY	103
5	DEVELOPING AN EC TECHNIQUE TO HANDLE INSEPARABLE FUNCTION INTERACTION	105
5.1	CHALLENGES FOR MULTI-OBJECTIVE OPTIMISATION ALGORITHMS	106

5.1.1	<i>Convergence to Global Pareto Front</i>	106
5.1.2	<i>Maintenance of Diverse Pareto-optimal Solutions</i>	107
5.2	PROPOSED SOLUTION STRATEGY	108
5.3	PROPOSED GENERALISED REGRESSION GA (GRGA)	110
5.4	DISTRIBUTION ALGORITHMS	112
5.4.1	<i>Linear Distribution Algorithm (LDA)</i>	113
5.4.2	<i>Random Distribution Algorithm (RDA)</i>	114
5.4.3	<i>Hybrid Distribution Algorithm (HDA)</i>	117
5.5	COMPUTATIONAL EXPENSE OF GRGA.....	118
5.6	PERFORMANCE ANALYSIS OF GRGA.....	119
5.6.1	<i>Experimental Results</i>	119
5.6.2	<i>Discussion of Results</i>	121
5.6.2.1	ROT.....	125
5.6.2.2	ZDT4.....	126
5.6.2.3	ZDT6.....	126
5.6.3	<i>Summary of Results</i>	127
5.7	SUMMARY.....	128
6	DEVELOPING AN EC TECHNIQUE TO HANDLE VARIABLE DEPENDENCE	130
6.1	CHALLENGES FOR MULTI-OBJECTIVE OPTIMISATION ALGORITHMS	131
6.2	ALTERNATIVE STRATEGIES FOR HANDLING VARIABLE DEPENDENCE.....	133
6.2.1	<i>Solving Optimisation Problems Having Known Dependency Equations</i>	133
6.2.1.1	Step 1: Identification of Dependency Relationships.....	133
6.2.1.2	Step 2: Classification of Variables	133
6.2.2	<i>Solving Optimisation Problems Having Unknown Dependency Equations</i>	134
6.2.2.1	Regression Analysis (RA).....	134
6.2.2.2	Neural Networks (NNs).....	136
6.2.2.3	Probabilistic Modelling (PM).....	137
6.3	PROPOSED SOLUTION STRATEGY	138
6.3.1	<i>Analysis of Alternative Solution Strategies</i>	138
6.3.1.1	Solving Optimisation Problems Having Known Dependency Equations.....	139
6.3.1.2	Solving Optimisation Problems Having Unknown Dependency Equations	139
6.3.2	<i>Proposed Solution Strategy</i>	140
6.4	PROPOSED GENETIC ALGORITHM FOR VARIABLE DEPENDENCE (GAVD).....	141
6.4.1	<i>Step 1: Identification of Dependency Relationships</i>	142
6.4.2	<i>Step 2: Classification of Variables</i>	143
6.4.3	<i>Computational Expense</i>	145
6.5	WORKED EXAMPLES.....	145

6.5.1	<i>Worked Example 1</i>	145
6.5.2	<i>Worked Example 2</i>	147
6.5.3	<i>Worked Example 3</i>	148
6.6	PERFORMANCE ANALYSIS OF GAVD	148
6.6.1	<i>Experimental Results</i>	149
6.6.2	<i>Discussion of Results</i>	154
6.6.2.1	ROT.....	155
6.6.2.2	ZDT4.....	156
6.6.2.3	ZDT6.....	157
6.6.3	<i>Summary of Results</i>	157
6.7	SUMMARY.....	158
7	DEVELOPING A TEST BED FOR ENGINEERING DESIGN OPTIMISATION	160
7.1	METHODOLOGY FOR TEST BED DEVELOPMENT.....	161
7.2	IDENTIFICATION OF TEST BED PARAMETERS	164
7.2.1	<i>Convergence to Pareto Front</i>	164
7.2.2	<i>Maintaining Diversity across Pareto Front</i>	165
7.3	PROPOSED TEST BED	167
7.3.1	<i>Strategy for Development of Proposed Test Bed</i>	167
7.3.2	<i>Proposed Reverse Engineered Test Bed (RETB)</i>	169
7.4	FUNCTION PROTOTYPES FOR RETB.....	170
7.4.1	<i>Diversity Function (D)</i>	171
7.4.2	<i>Shape Function (S)</i>	174
7.4.3	<i>Interaction Function (I)</i>	177
7.4.4	<i>Constraint Function (C)</i>	179
7.4.4.1	Pareto Blocking Constraints.....	180
7.4.4.2	Pareto Intersecting Constraints.....	182
7.4.4.3	Composite Constraints	184
7.5	GUIDELINES FOR USE OF RETB.....	186
7.6	RETB CASE STUDIES	188
7.7	RETB-II: EXTENSION OF RETB TO INCORPORATE VARIABLE DEPENDENCE.....	192
7.7.1	<i>Number of Dependency Relationships</i>	193
7.7.2	<i>Nature of Dependency Relationships</i>	193
7.7.3	<i>Nature of Available Information</i>	197
7.8	GUIDELINES FOR USE OF RETB-II.....	198
7.9	RETB-II CASE STUDIES	199
7.9.1	<i>Example 1: Biased Problem</i>	201
7.9.1.1	Step 1	201

7.9.1.2	Step 2	201
7.9.1.3	Step 3	204
7.9.2	<i>Example 2: Multi-front Problem</i>	205
7.9.2.1	Step 1	205
7.9.2.2	Step 2	205
7.9.2.3	Step 3	206
7.9.3	<i>Example 3: Deceptive Problem</i>	207
7.9.3.1	Step 1	207
7.9.3.2	Step 2	208
7.9.3.3	Step 3	208
7.9.4	<i>Example 4: Discontinuous Search Space Problem</i>	209
7.9.4.1	Step 1	209
7.9.4.2	Step 2	210
7.9.4.3	Step 3	210
7.10	RETB/RETB-II VERSUS EXISTING TEST BEDS	212
7.11	SUMMARY.....	214
8	PERFORMANCE ANALYSIS OF GRGA/GAVD USING RETB/RETB-II.....	216
8.1	CASE STUDY DEVELOPMENT	216
8.1.1	<i>Case-1</i>	217
8.1.1.1	Dependency Scenario 1.1	219
8.1.1.2	Dependency Scenario 1.2.....	220
8.1.2	<i>Case-2</i>	221
8.1.2.1	Dependency Scenario 2.1	223
8.1.2.2	Dependency Scenario 2.2.....	224
8.1.3	<i>Case-3</i>	225
8.1.3.1	Dependency Scenario 3.1	227
8.1.3.2	Dependency Scenario 3.2.....	229
8.2	EXPERIMENTAL RESULTS.....	230
8.2.1	<i>Case-1</i>	231
8.2.1.1	Dependency Scenario 1.1	231
8.2.1.2	Dependency Scenario 1.2.....	232
8.2.2	<i>Case-2</i>	233
8.2.2.1	Dependency Scenario 2.1	233
8.2.2.2	Dependency Scenario 2.2.....	234
8.2.3	<i>Case-3</i>	235
8.2.3.1	Dependency Scenario 3.1	236
8.2.3.2	Dependency Scenario 3.2.....	239
8.3	DISCUSSION OF RESULTS	241
8.3.1	<i>Case-1</i>	241

8.3.1.1	Dependency Scenario 1.1	241
8.3.1.2	Dependency Scenario 1.2	242
8.3.2	<i>Case-2</i>	243
8.3.2.1	Dependency Scenario 2.1	243
8.3.2.2	Dependency Scenario 2.2	244
8.3.3	<i>Case-3</i>	245
8.3.3.1	Dependency Scenario 3.1	245
8.3.3.2	Dependency Scenario 3.2	246
8.4	KEY RESULTS	247
8.5	SUMMARY	249
9	REAL-LIFE CASE STUDIES	251
9.1	CASE STUDIES FROM REAL-LIFE ENGINEERING DESIGN OPTIMISATION	251
9.2	SELECTION OF CASE STUDIES	258
9.3	DESIGN OF A WELDED BEAM	263
9.3.1	<i>Experimental Results</i>	264
9.3.2	<i>Discussion of Results</i>	266
9.4	DESIGN OF A MACHINE TOOL SPINDLE	267
9.4.1	<i>Experimental Results</i>	267
9.4.2	<i>Discussion of Results</i>	268
9.5	DESIGN OF A TURBINE BLADE COOLING SYSTEM	271
9.5.1	<i>Experimental Results</i>	274
9.5.2	<i>Discussion of Results</i>	279
9.6	VALIDATION OF RESULTS	282
9.6.1	<i>Design of a Welded Beam</i>	282
9.6.2	<i>Design of a Machine Tool Spindle</i>	282
9.6.3	<i>Design of a Turbine Blade Cooling System</i>	283
9.7	SUMMARY	284
10	DISCUSSION AND CONCLUSIONS	285
10.1	DISCUSSION	285
10.1.1	<i>Key Observations of this Research</i>	285
10.1.1.1	Literature Survey	286
10.1.1.2	Industrial Context and Focus	287
10.1.1.3	Gap Analysis: EC versus Multiple Objectives, Constraints and Variable Interaction	288
10.1.1.4	Development of GRGA	288
10.1.1.5	Development of GAVD	289
10.1.1.6	Development of RETB and RETB-II	290
10.1.1.7	Performance Analysis of GRGA and GAVD	290

10.1.1.8	Validation Using Real-life Problems	291
10.1.2	<i>Main Contributions</i>	292
10.1.3	<i>Generality of Research</i>	293
10.1.3.1	Limitations of Research Methodology	293
10.1.3.2	Limitations of GRGA.....	294
10.1.3.3	Limitations of GAVD	295
10.1.3.4	Limitations of Proposed Test Beds	296
10.2	FUTURE RESEARCH.....	296
10.3	CONCLUSIONS	298
11	REFERENCES.....	301
A.	APPENDIX: ‘FLEXO’ QUESTIONNAIRE	324
B.	APPENDIX: DESCRIPTION OF NSGA-II.....	332
C.	APPENDIX: PERFORMANCE METRICS FOR MULTI-OBJECTIVE OPTIMISATION	337

List of Figures

FIGURE 1.1: RECTANGULAR CANTILEVER BEAM (LENGTH = L, BREADTH = B AND HEIGHT = H).....	2
FIGURE 1.2: SCHEMATIC DESCRIPTION OF GAS (SOURCE: JARED ET AL., 1998).....	9
FIGURE 1.3: FLEXIBLE OPTIMISATION WHEEL (SOURCE: ROY ET AL., 2000).....	11
FIGURE 1.4: PERSPECTIVES OF ‘FLEXO’ – (A) AREA PERSPECTIVE (B) DATA FLOW PERSPECTIVE (SOURCE: ROY ET. AL., 2000A)	13
FIGURE 1.5: FLEXIBLE OPTIMISATION ‘TOOL BOX’ – (A) LOCATION (B) DETAILS	14
FIGURE 1.6: THESIS LAYOUT	15
FIGURE 2.1: CLASSICAL OPTIMISATION ALGORITHMS (SOURCE: DEB, 2001).....	19
FIGURE 2.2: EXAMPLES OF INTERACTION – (A) NO INTERACTION (B) SYNERGISTIC INTERACTION (C) ANTI-SYNERGISTIC INTERACTION (PHADKE, 1989)	46
FIGURE 2.3: ESTIMATION OF DISTRIBUTION ALGORITHM (EDA) APPROACH TO OPTIMISATION (SOURCE: LARRANAGA ET AL., 1999)	52
FIGURE 2.4: ILLUSTRATION OF REGRESSION MODEL (A) REGRESSION LINE THROUGH POPULATION MEANS (B) ERRORS ASSOCIATED WITH INDIVIDUAL OBSERVATIONS (SOURCE: EVANS AND OLSON, 2000)	63
FIGURE 2.5: NN WITH ONE HIDDEN LAYER (SOURCE: GERSHENFELD, 1999).....	64
FIGURE 2.6: EXAMPLE OF PROBABILISTIC MODELLING (PM) (ASSUMING THAT $p(\mathbf{X} = \mathbf{x})$ IS JGPDF)	65
FIGURE 2.7: AN EXAMPLE OF A TREE DIAGRAM (TD).....	67
FIGURE 2.8: NON-TUNEABLE TEST PROBLEMS FOR MULTI-OBJECTIVE OPTIMISATION – (A) SCH1 (B) SCH2 (C) KUR (D) VNT (SOURCE: DEB, 2001).....	71
FIGURE 2.9: ZITZLER-DEB-THIELE (ZDT) TEST PROBLEMS – (A) ZDT1 (B) ZDT2 (C) ZDT3 (D) ZDT4 (E) ZDT5 (F) ZDT6 (SOURCE: DEB, 2001).....	74
FIGURE 2.10: NON-TUNEABLE TEST PROBLEMS FOR CONSTRAINED MULTI-OBJECTIVE OPTIMISATION - (A) OSY (B) SRN (C) TNK (SOURCE: DEB, 2001)	77
FIGURE 2.11: TUNEABLE TEST PROBLEMS FOR CONSTRAINED MULTI-OBJECTIVE OPTIMISATION – (A) CTP1 (B) CTP5 (C) CTP6 (D) CTP7 (SOURCE: DEB, 2001)	79
FIGURE 5.1: AN EXAMPLE OF INSEPARABLE FUNCTION INTERACTION	108
FIGURE 5.2: PROPOSED SOLUTION STRATEGY	109
FIGURE 5.3: GENERALISED REGRESSION GA (GRGA)	111
FIGURE 5.4: LINEAR DISTRIBUTION ALGORITHM (LDA)	113
FIGURE 5.5: RANDOM DISTRIBUTION ALGORITHM (RDA).....	115
FIGURE 5.6: IDENTIFICATION OF UNIQUE POINTS (ASSUMING TWO OBJECTIVE FUNCTIONS).....	116
FIGURE 5.7: ALGORITHM FOR IDENTIFICATION OF UNIQUE POINTS	116
FIGURE 5.8: HYBRID DISTRIBUTION ALGORITHM (HDA)	118

FIGURE 5.9: OBJECTIVE FUNCTIONS OF ROT (ASSUMING TWO VARIABLES)	120
FIGURE 5.10: OBJECTIVE FUNCTIONS OF ZDT4 (ASSUMING TWO VARIABLES).....	120
FIGURE 5.11: OBJECTIVE FUNCTIONS OF ZDT6 (ASSUMING TWO VARIABLES).....	120
FIGURE 5.12: GRGA PERFORMANCE ANALYSIS USING ROT PROBLEM – (A) FULL SEARCH SPACE (B) MAGNIFIED SEARCH SPACE.....	122
FIGURE 5.13: GRGA PERFORMANCE ANALYSIS USING ZDT4 PROBLEM – (A) FULL SEARCH SPACE (B) MAGNIFIED SEARCH SPACE.....	123
FIGURE 5.14: GRGA PERFORMANCE ANALYSIS USING ZDT6 PROBLEM – (A) FULL SEARCH SPACE (B) MAGNIFIED SEARCH SPACE.....	124
FIGURE 6.1: RELATIONSHIP BETWEEN STRESS(S) AND TEMPERATURE(T) (FRIV: FEASIBLE REGION WITH INDEPENDENT VARIABLES AND FRDV: FEASIBLE REGION WITH DEPENDENT VARIABLES).....	132
FIGURE 6.2: SOLVING OPTIMISATION PROBLEMS HAVING KNOWN DEPENDENCY EQUATIONS	134
FIGURE 6.3: APPLICATION OF REGRESSION ANALYSIS (RA) FOR SOLVING DEPENDENT-VARIABLE OPTIMISATION PROBLEMS	135
FIGURE 6.4: APPLICATION OF NEURAL NETWORKS (NNs) FOR SOLVING DEPENDENT-VARIABLE OPTIMISATION PROBLEMS	136
FIGURE 6.5: APPLICATION OF PROBABILISTIC MODELLING (PM) FOR SOLVING DEPENDENT- VARIABLE OPTIMISATION PROBLEMS	137
FIGURE 6.6: GENETIC ALGORITHM FOR VARIABLE DEPENDENCE (GAVD).....	142
FIGURE 6.7: AN EXAMPLE OF A DEPENDENCY TREE (DT) (F: OBJECTIVE FUNCTION AND A, B, C, D, E, G, H, I: DECISION VARIABLES).....	144
FIGURE 6.8: DEPENDENCY TREE (DT): WORKED EXAMPLE 1	147
FIGURE 6.9: DEPENDENCY TREE (DT): WORKED EXAMPLE 2	147
FIGURE 6.10: DEPENDENCY TREE (DT): WORKED EXAMPLE 3	148
FIGURE 6.11: DEPENDENCY RELATIONSHIP IN ROT	150
FIGURE 6.12: DEPENDENCY RELATIONSHIP IN ZDT4.....	150
FIGURE 6.13: DEPENDENCY RELATIONSHIP IN ZDT6.....	150
FIGURE 6.14: GAVD PERFORMANCE ANALYSIS USING ROT PROBLEM – (A) NSGA-II AND GRGA (B) GAVD (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	151
FIGURE 6.15: GAVD PERFORMANCE ANALYSIS USING ZDT4 PROBLEM – (A) NSGA-II AND GRGA (B) GAVD (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	152
FIGURE 6.16: GAVD PERFORMANCE ANALYSIS USING ZDT6 PROBLEM – (A) NSGA-II AND GRGA (B) GAVD (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	153

FIGURE 7.1: METHODOLOGY FOR TEST BED DEVELOPMENT	163
FIGURE 7.2: AN EXAMPLE OF DIVERSITY FUNCTION D ($K=2$; $M_i=2,2$; $T=4$; $A_{ij}=0.1,0.1,0.1,0.2$; $B_{ij}=1,5,1,3$; $R_i=0,0.2,0.5,0.7$).....	173
FIGURE 7.3: AN EXAMPLE OF SHAPE FUNCTION S ($M=3$; $A_i=2,0.4$; $C_{1j}=1,1$; $B_i=1,4$; $C_i=4,7$; $D_i=2,4$; $E=8$)	176
FIGURE 7.4: AN EXAMPLE OF INTERACTION FUNCTION I ($N=2$; $K=1$; $B_i=1$; $C_i=4$; $M_i=3$; $D_i=2$; $\epsilon=0.004$; $E=4.38$).....	179
FIGURE 7.5: EXAMPLES OF CONSTRAINT FUNCTIONS C - (A) PARETO BLOCKING CONSTRAINTS (C_1) ($J=4$; $E_{1j}=-0.1,0.4,1.5,2.9$; $E_{2j}=0.1,0.9,2.1,3.7$; $M=2$; $A_i=0.6$; $C_{1i}=0$; $E=1$) (B) PARETO INTERSECTING CONSTRAINTS (C_2) ($J=5$; $M_i=1$; $E_{1j}=-0.1,0.5,1.5,2.9,4.7$; $E_{2j}=0.1,0.9,2.1,3.7,5.7$; $M=2$; $A_i=0.6$; $C_{1i}=0$; $E=1$) (C) COMPOSITE CONSTRAINTS (C_3) ($J=2$; $E_j=1,2$; $A_{ij}=0.9,0.2$; $P_{ij}=2.5,4$; $M=2$; $A_i=0.6$; $C_{1i}=0$; $E=1$).....	181
FIGURE 7.6: STEPS FOR USING REVERSE ENGINEERED TEST BED (RETB)	186
FIGURE 7.7: TREE DIAGRAM FOR CONSTRUCTING RETB OPTIMISATION PROBLEMS (PF: PARETO FRONT, HIGHLIGHTED NUMBERS: RE-USEABLE SUB-TREES)	188
FIGURE 7.8: SEARCH SPACE OF RETB CASE-1	191
FIGURE 7.9: SEARCH SPACE OF RETB CASE-2	191
FIGURE 7.10: SEARCH SPACE OF RETB CASE-3	192
FIGURE 7.11: STEPS FOR USING REVERSE ENGINEERED TEST BED – II (RETB-II).....	199
FIGURE 7.12: ORIGINAL SEARCH SPACE FOR RETB-II EXAMPLES 1, 2, 3 & 4 (ASSUMING INDEPENDENT VARIABLES)	200
FIGURE 7.13: DEPENDENCY RELATIONSHIP IN RETB-II EXAMPLE 1: BIASED PROBLEM.....	204
FIGURE 7.14: EXHAUSTIVE SEARCH FOR RETB-II EXAMPLE 1: BIASED PROBLEM	204
FIGURE 7.15: DEPENDENCY RELATIONSHIP IN RETB-II EXAMPLE 2: MULTI-FRONT PROBLEM	205
FIGURE 7.16: EXHAUSTIVE SEARCH FOR RETB-II EXAMPLE 2: MULTI-FRONT PROBLEM.....	206
FIGURE 7.17: DEPENDENCY RELATIONSHIP IN RETB-II EXAMPLE 3: DECEPTIVE PROBLEM.....	207
FIGURE 7.18: EXHAUSTIVE SEARCH FOR RETB-II EXAMPLE 3: DECEPTIVE PROBLEM	208
FIGURE 7.19: DEPENDENCY RELATIONSHIP IN RETB-II EXAMPLE 4: DISCONTINUOUS PROBLEM	209
FIGURE 7.20: EXHAUSTIVE SEARCH FOR RETB-II EXAMPLE 4: DISCONTINUOUS PROBLEM.....	211
FIGURE 7.21: DEPENDENCY TREE (DT) – (A) EXAMPLES 1 & 2 (B) EXAMPLES 3 & 4.....	212
FIGURE 8.1: CASE-1	219
FIGURE 8.2 DEPENDENCY RELATIONSHIP IN CASE-1 (DEPENDENCY SCENARIO 1.1)	221
FIGURE 8.3: DEPENDENCY RELATIONSHIP IN CASE-1 (DEPENDENCY SCENARIO 1.2) (ORIGINAL: ACTUAL RELATIONSHIP, APPROXIMATED: RELATIONSHIP ESTIMATED BY RA).....	221
FIGURE 8.4: CASE-2	222
FIGURE 8.5: DEPENDENCY RELATIONSHIP IN CASE-2 (DEPENDENCY SCENARIO 2.1)	223

FIGURE 8.6: DEPENDENCY RELATIONSHIP IN CASE-2 (DEPENDENCY SCENARIO 2.2) (ORIGINAL: ACTUAL RELATIONSHIP, APPROXIMATED: RELATIONSHIP ESTIMATED BY RA).....	225
FIGURE 8.7: DEPENDENCY RELATIONSHIP IN CASE-3 (DEPENDENCY SCENARIO 3.1).....	229
FIGURE 8.8: DEPENDENCY RELATIONSHIP IN CASE-3 (DEPENDENCY SCENARIO 3.2).....	229
FIGURE 8.9: GAVD PERFORMANCE IN RETB-II CASE-1 (DEPENDENCY SCENARIO 1.1) (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	232
FIGURE 8.10: GAVD PERFORMANCE IN RETB-II CASE-1 (DEPENDENCY SCENARIO 1.2) (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	233
FIGURE 8.11: GAVD PERFORMANCE IN RETB-II CASE-2 (DEPENDENCY SCENARIO 2.1) (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	234
FIGURE 8.12: GAVD PERFORMANCE IN RETB-II CASE-2 (DEPENDENCY SCENARIO 2.2) (PARETO FRONT FOR INDEPENDENT VARIABLES: PFIV, PARETO FRONT FOR DEPENDENT VARIABLES: PFDV, ESTIMATED PARETO FRONT: EPF).....	235
FIGURE 8.13: NSGA-II AND GRGA PERFORMANCE IN RETB-II CASE-3 (DEPENDENCY SCENARIO 3.1) – (A) F1-F2 GRAPH (B) F1-F3 GRAPH (C) F1-F4 GRAPH (D) F2-F3 GRAPH (E) F2-F4 GRAPH (F) F3-F4 GRAPH.....	237
FIGURE 8.14: GAVD PERFORMANCE IN RETB-II CASE-3 (DEPENDENCY SCENARIO 3.1) – (A) F2-F1 GRAPH (B) F3-F1 GRAPH (C) F3-F2 GRAPH (D) F4-F1 GRAPH (E) F4-F2 GRAPH (F) F4-F3 GRAPH.....	238
FIGURE 8.15: GAVD PERFORMANCE IN RETB-II CASE-3 (DEPENDENCY SCENARIO 3.2) – (A) F2-F1 GRAPH (B) F3-F1 GRAPH (C) F3-F2 GRAPH (D) F4-F1 GRAPH (E) F4-F2 GRAPH (F) F4-F3 GRAPH.....	240
FIGURE 9.1: WELDED BEAM DESIGN (SOURCE: DEB, PRATAP AND MOITRA, 2000).....	253
FIGURE 9.2: DESIGN OF A MACHINE TOOL SPINDLE (SOURCE: COELLO, 1997).....	254
FIGURE 9.3: GENERAL ARRANGEMENT OF FIVE-PASS COOLING OF TURBINE ROTOR BLADE (SOURCE: ROY, 1997).....	257
FIGURE 9.4: SCHEMATIC DIAGRAM SHOWING GENERAL ARRANGEMENT OF COOLANT FLOW THROUGH TURBINE BLADE WITH FILM COOLING MECHANISM (1: COOLANT AIR INLET, 2: FILM COOLING PASSAGE INLET, 3: COOLING AIR EXIT AND 3': FILM COOLING HOLE EXIT) (SOURCE: ROY, 1997).....	258
FIGURE 9.5: RESULTS FROM NSGA-II ON WELDED BEAM DESIGN (UNITS: DEFLECTION IN INCH, COST IN COST UNITS).....	265
FIGURE 9.6: RESULTS FROM GRGA (WITHOUT FINAL REDISTRIBUTION) ON WELDED BEAM DESIGN (UNITS: DEFLECTION IN INCH, COST IN COST UNITS).....	265

FIGURE 9.7: RESULTS FROM GRGA (WITH FINAL REDISTRIBUTION) ON WELDED BEAM DESIGN (UNITS: DEFLECTION IN INCH, COST IN COST UNITS)	266
FIGURE 9.8: RESULTS FROM NSGA-II ON DESIGN OF MACHINE TOOL SPINDLE (UNITS: DISPLACEMENT IN MM, VOLUME IN MM ³)	269
FIGURE 9.9: RESULTS FROM GRGA (WITH FINAL REDISTRIBUTION) ON DESIGN OF MACHINE TOOL SPINDLE (UNITS: DISPLACEMENT IN MM, VOLUME IN MM ³)	269
FIGURE 9.10: RESULTS FROM GA VD (WITH FINAL REDISTRIBUTION) ON DESIGN OF MACHINE TOOL SPINDLE (UNITS: DISPLACEMENT IN MM, VOLUME IN MM ³)	270
FIGURE 9.11: RESULTS FROM NSGA-II ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING TWO OBJECTIVES) (UNITS: WCR IN KG/S, TWG IN K)	275
FIGURE 9.12: RESULTS FROM GRGA (WITHOUT FINAL REDISTRIBUTION) ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING TWO OBJECTIVES) (UNITS: WCR IN KG/S, TWG IN K)	275
FIGURE 9.13: RESULTS FROM GRGA (WITH FINAL REDISTRIBUTION) ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING TWO OBJECTIVES) (UNITS: WCR IN KG/S, TWG IN K)	276
FIGURE 9.14: RESULTS FROM GRGA ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING FOUR OBJECTIVES) (UNITS: WCR IN KG/S, WCF IN KG/S, TWG IN K, TWF IN K) – (A) WCR-TWG GRAPH (B) WCR-WCF GRAPH (C) WCR-TWF GRAPH (D) TWG-WCF GRAPH (E) TWG-TWF GRAPH (F) WCF-TWF GRAPH	277
FIGURE 9.15: RESULTS FROM NSGA-II ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING FOUR OBJECTIVES) (UNITS: WCR IN KG/S, WCF IN KG/S, TWG IN K, TWF IN K) – (A) TWG-WCR GRAPH (B) WCF-WCR GRAPH (C) WCF-TWG GRAPH (D) TWF-WCR GRAPH (E) TWF-TWG GRAPH (F) TWF-WCF GRAPH	278
FIGURE B.1: NON-DOMINATED SORTING GA –II (NSGA-II)	333
FIGURE C.1: ILLUSTRATION OF DISTANCE METRIC γ	337
FIGURE C.2: ILLUSTRATION OF DIVERSITY METRIC Δ	338

List of Tables

TABLE 1.1: CLASSIFICATION SCHEMES FOR ENGINEERING DESIGN OPTIMISATION PROBLEMS.....	6
TABLE 2.1: HYBRID CLASSIFICATION OF EMOTs.....	30
TABLE 2.2: CLASSIFICATION OF ECOTs.....	38
TABLE 2.3: CLASSIFICATION OF ETIFIs.....	49
TABLE 2.4: CLASSIFICATION OF TECHNIQUES FOR HANDLING VARIABLE DEPENDENCE.....	60
TABLE 2.5: CLASSIFICATION OF TEST PROBLEMS FOR OPTIMISATION ALGORITHMS.....	69
TABLE 2.6: NON-TUNEABLE TEST PROBLEMS FOR MULTI-OBJECTIVE OPTIMISATION.....	70
TABLE 2.7: ZITZLER-DEB-THIELE (ZDT) TEST PROBLEMS.....	73
TABLE 2.8: NON-TUNEABLE TEST PROBLEMS FOR CONSTRAINED MULTI-OBJECTIVE OPTIMISATION.....	76
TABLE 2.9: TUNEABLE TEST PROBLEMS FOR CONSTRAINED MULTI-OBJECTIVE OPTIMISATION.....	78
TABLE 4.1: EVOLUTIONARY-BASED OPTIMISATION ALGORITHMS IN APPLIED RESEARCH.....	93
TABLE 4.2: FEATURES OF REAL-LIFE ENGINEERING DESIGN OPTIMISATION PROBLEMS.....	95
TABLE 5.1: TEST PROBLEMS FOR PERFORMANCE ANALYSIS OF GRGA (PF: PARETO FRONT).....	119
TABLE 5.2: PERFORMANCE METRICS IN ROT, ZDT4 AND ZDT6.....	125
TABLE 6.1: COMPARISON OF DATA MODELLING TECHNIQUES.....	140
TABLE 6.2: DEPENDENCY CHART (DC): WORKED EXAMPLE 1.....	146
TABLE 6.3: DEPENDENCY CHART (DC): WORKED EXAMPLE 2.....	147
TABLE 6.4: DEPENDENCY CHART (DC): WORKED EXAMPLE 3.....	148
TABLE 6.5: TEST PROBLEMS FOR PERFORMANCE ANALYSIS OF GAVD.....	149
TABLE 6.6: PERFORMANCE METRICS IN ROT, ZDT4 AND ZDT6.....	155
TABLE 7.1: SUMMARY OF PARAMETERS IN S, D AND I FUNCTIONS OF RETB.....	162
TABLE 7.2: SUMMARY OF PARAMETERS IN C ₁ , C ₂ AND C ₃ FUNCTIONS OF RETB.....	162
TABLE 7.3: SUMMARY OF PARAMETERS IN RETB-II.....	163
TABLE 7.4: IDENTIFICATION OF TEST BED PARAMETERS.....	165
TABLE 7.5: ADDITIONAL TEST BED PARAMETERS TO INCORPORATE VARIABLE DEPENDENCE.....	166
TABLE 7.6: FEATURES OF RETB TEST CASES.....	189
TABLE 7.7: RETB PARAMETERS VALUES FOR TEST CASES – (A) CASE-1 (B) CASE-2 (C) CASE-3.....	189
TABLE 7.8: PARAMETER VALUES FOR RETB-II EXAMPLES – (A) EXAMPLE 1: BIASED PROBLEM AND EXAMPLE 2: MULTI-FRONT PROBLEM (B) EXAMPLE 3: DECEPTIVE PROBLEM AND EXAMPLE 4: DISCONTINUOUS PROBLEM.....	202
TABLE 7.9: DEPENDENCY CHART (DC) – (A) EXAMPLES 1 & 2 (B) EXAMPLES 3 & 4.....	212
TABLE 8.1: TEST CASES.....	217
TABLE 8.2: RETB PARAMETER VALUES FOR CASE-1.....	218

TABLE 8.3: RETB-II PARAMETER VALUES FOR CASE-1	220
TABLE 8.4: RETB PARAMETERS FOR CASE-2	223
TABLE 8.5: RETB-II PARAMETER VALUES FOR CASE-2	224
TABLE 8.6: RETB PARAMETERS FOR CASE-3	226
TABLE 8.7: CASE-3 (DEPENDENCY SCENARIO 3.1) (UPPER DIAGONAL GRAPHS: SEARCH SPACE WITH VARIABLE DEPENDENCY, LOWER DIAGONAL GRAPHS: SEARCH SPACE WITHOUT VARIABLE DEPENDENCY).....	227
TABLE 8.8: RETB-II PARAMETER VALUES FOR CASE-3	228
TABLE 8.9: CASE-3 (DEPENDENCY SCENARIO 3.2) (UPPER DIAGONAL GRAPHS: SEARCH SPACE WITH VARIABLE DEPENDENCY, LOWER DIAGONAL GRAPHS: SEARCH SPACE WITHOUT VARIABLE DEPENDENCY).....	230
TABLE 8.10: PERFORMANCE METRICS IN CASE-1 (DEPENDENCY SCENARIO 1.1).....	231
TABLE 8.11: PERFORMANCE METRICS IN CASE-1 (DEPENDENCY SCENARIO 1.2).....	232
TABLE 8.12: PERFORMANCE METRICS IN CASE-2 (DEPENDENCY SCENARIO 2.1).....	234
TABLE 8.13: PERFORMANCE METRICS IN CASE-2 (DEPENDENCY SCENARIO 2.2).....	235
TABLE 8.14: GAVD PERFORMANCE IN RETB-II CASE-3 (DEPENDENCY SCENARIO 3.1) (UPPER DIAGONAL GRAPHS: PARETO FRONT WITH NSGA-II AND GRGA SOLUTIONS, LOWER DIAGONAL GRAPHS: PARETO FRONT WITH GAVD SOLUTIONS).....	236
TABLE 8.15: PERFORMANCE METRICS IN CASE-3 (DEPENDENCY SCENARIO 3.1).....	236
TABLE 8.16: GAVD PERFORMANCE IN RETB-II CASE-3 (DEPENDENCY SCENARIO 3.2) (UPPER DIAGONAL GRAPHS: SEARCH SPACE WITH VARIABLE DEPENDENCY, LOWER DIAGONAL GRAPHS: PARETO FRONT WITH NSGA-II, GRGA AND GAVD SOLUTIONS)	239
TABLE 8.17: PERFORMANCE METRICS IN CASE-3 (DEPENDENCY SCENARIO 3.2).....	239
TABLE 9.1: CASE STUDIES FROM REAL-LIFE ENGINEERING DESIGN OPTIMISATION	260
TABLE 9.2: VARIABLE VALUES CORRESPONDING TO IDENTIFIED PARETO FRONT.....	267
TABLE 9.3: VARIABLE VALUES CORRESPONDING TO IDENTIFIED PARETO FRONT.....	271
TABLE 9.4: COOLING SYSTEM DESIGN PROCEDURE USED IN TBCOM (SOURCE: ROY, 1997).....	273
TABLE 9.5: RESULTS FROM NSGA-II, GRGA (WITHOUT FINAL REDISTRIBUTION) AND GRGA (WITH FINAL REDISTRIBUTION) ON DESIGN OF TURBINE BLADE COOLING SYSTEM (ASSUMING FOUR OBJECTIVES) (UPPER DIAGONAL GRAPHS: GRGA RESULTS, LOWER DIAGONAL GRAPHS: NSGA-II RESULTS) (UNITS: WCR IN KG/S, WCF IN KG/S, TWG IN K, TWF IN K).....	276
TABLE 9.6: VARIABLE VALUES CORRESPONDING TO IDENTIFIED PARETO FRONT.....	281

Commonly Used Abbreviations

DA	Direct Analysis
DC	Dependency Chart
DT	Dependency Tree
EC	Evolutionary Computing
ECOTs	Evolutionary-based Constrained Optimisation Techniques
EDAs	Estimation of Distribution Algorithms
EMOTs	Evolutionary-based Multi-objective Optimisation Techniques
EP	Evolutionary Programming
ESs	Evolution Strategies
ETIFIs	Evolutionary-based Techniques for handling Inseparable Function Interaction
ETVD	Evolutionary-based Techniques for handling Variable Dependence
FLEXO	Flexible Optimisation within CAD/CAM Environment
GAs	Genetic Algorithms
GAVD	Genetic Algorithm for Variable Dependence
GP	Genetic Programming
GRGA	Generalised Regression Genetic Algorithm
HDA	Hybrid Distribution Algorithm
LDA	Linear Distribution Algorithm
NNs	Neural Networks
NSGA-II	Fast Elitist Non-dominated Sorting Genetic Algorithm
PM	Probabilistic Modelling
RA	Regression Analysis
RDA	Random Distribution Algorithm
RETB	Reverse Engineered Test Bed
RETB-II	Reverse Engineered Test Bed – II
TDs	Tree Diagrams

1 INTRODUCTION

The ever-increasing market demands to produce better products, with reduced costs and lead times, has prompted industry to look for rigorous ways of optimising its designs. Traditional trial-and-error method of design optimisation is not capable of meeting the industrial demands. Industries are, therefore, looking for automating the optimisation process using algorithms and computational techniques. However, the complexity of engineering design optimisation problems, coupled with limited flexibility and adequacy of existing optimisation techniques in dealing with the challenges of these problems, has prevented industry from adopting the optimisation algorithms. This research focuses on developing optimisation algorithms based on the Evolutionary Computing (EC) approach to address the complexities of engineering design optimisation problems. It has the broad aim of making the optimisation algorithms more popular in industry. This research is carried out as part of a project called 'FLEXO', titled 'Flexible Optimisation within CAD/CAM Environment' (Engineering and Physical Sciences Research Council (EPSRC) Grant No. GR/M 71473). This chapter attempts to address the following.

- ◆ *To define and classify the engineering design optimisation problems.*
- ◆ *To analyse the suitability of EC techniques for solving these problems.*
- ◆ *To present a brief introduction to the GAs.*
- ◆ *To introduce 'FLEXO': the parent project of this research.*
- ◆ *To give the problem statement and motivation of this research.*
- ◆ *To present the thesis layout.*

1.1 Definition and Classification of Engineering

Design Optimisation Problems

Optimisation can be defined as the process of selecting a superior design, based on some pre-defined criteria, from a set of feasible alternative designs. The engineering

design optimisation problems are the real-life problems that, as opposed to the theoretical problems (test cases), are encountered in industry. Some examples of these problems are the design of aerospace structures for better aerodynamics, the surface design of automobiles for improved aesthetics, the design of mechanical components for maximum performance, and the design of pumps, turbines and heat transfer equipment for maximum efficiency (Rao, 1996). There are a number of ways in which these optimisation problems can be classified. Some of the commonly used classification schemes are as follows.

1.1.1 Based on Number of Variables

The engineering design optimisation problems can be classified as single and multi-dimensional based on the number of variables involved in the problem. Consider the problem of optimising the design of the rectangular cantilever beam, shown in Figure 1.1, for given material and loading conditions. An example of single-dimensional optimisation problem is the design of this beam when the cross-section is fixed leaving the length as the only variable. The same problem is classified as multi-dimensional if more than one dimension of the beam can be varied.

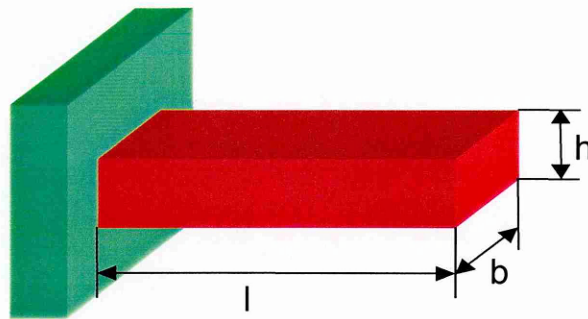


Figure 1.1: Rectangular Cantilever Beam (Length = l , Breadth = b and Height = h)

1.1.2 Based on Existence of Constraints

An engineering design optimisation problem can be classified as constrained or unconstrained depending on whether constraints exist in the problem [Rao, 1996; Schwefel, 1995]. The optimisation of rectangular cantilever beam shown in Figure 1.1 is classified as unconstrained if there are no bounds and pre-defined relationships

involving its variables (beam dimensions). The same problem is called constrained if it has bounds and/or relationships concerning beam dimensions.

1.1.3 Based on Number of Objective Functions

Depending upon the number of objective functions in the engineering design optimisation problem, it can be classified as single-objective and multi-objective (Rao, 1996). The optimisation problem of Figure 1.1 is classified as single-objective if it has only one objective, say, the minimisation of end deflection under the given constraints. On the other hand, it is classified as multi-objective if it involves a number of conflicting objectives that need to be simultaneously optimised. An example could be the simultaneous minimisation of end deflection, maximum stress along the beam length and cost involved.

1.1.4 Based on Nature of Objective Functions

The objective functions involved in an engineering design optimisation problem may be either quantitative or qualitative in nature. Some examples of quantitative objective functions for the beam design problem of Figure 1.1 are those involving the end deflection, maximum stress along the beam length and cost. On the other hand, the qualitative objective functions involve issues like manufacturability and designers' special preferences (Rogero *et al.*, 2000). Based on the nature of objective functions, the optimisation problems can be classified as quantitative, qualitative or hybrid. These categories of optimisation problems involve quantitative, qualitative and a combination of quantitative and qualitative objective functions respectively.

1.1.5 Based on Separability of Functions

A function is said to be separable if it can be expressed as the sum of single-variable functions. An alternative definition of separability relaxes the above definition to include decomposition into functions that involve groups of variables rather than just a single variable. Inseparability manifests itself as cross-product terms, and makes the effect of a variable on the function dependent on the values of other variables in

the function. The engineering design optimisation problems can be classified as separable and inseparable based on the separability of objective functions. Inseparability causes difficulties for an optimisation algorithm by requiring it to update all decision variables in a unique way in order to converge to an optimum solution. Suppose the problem of Figure 1.1 requires the minimisation of the end deflection (δ) of the cantilever beam. This becomes an inseparable optimisation problem since the equation for δ involves cross-product terms among its variables (beam dimensions) (Equation 1.1).

$$\delta = 4\pi Pl^3 / 3Eb^2h^2, \quad \text{Equation 1.1}$$

$\pi, P, E \Rightarrow \text{Constants.}$

On the other hand, the problem is classified as separable if it requires the minimisation of the sum of all edge lengths (S) of the beam (Equation 1.2).

$$S = 4(l + b + h). \quad \text{Equation 1.2}$$

1.1.6 Based on Dependence among Variables

Variable dependence occurs when the variables are functions of each other, and hence cannot be varied independently. Here, the change in one variable has an impact on the value of the other. This causes additional problems for an optimisation algorithm due to the requirement that all dependency relationships need to be satisfied while searching for an optimum solution. This has an effect of constraining the search space. The optimisation of the rectangular cantilever beam, shown in Figure 1.1, is classified as a dependent-variable optimisation problem if it involves relationship(s) among its variables (beam dimensions) that need to be satisfied. These relationships may arise due to some physical/practical requirements or due to designers' special preferences. An example of these relationships is shown in Equation 1.3 in which the designer prefers those designs that have the cross-section aspect ratio as defined by h/b .

$$\text{Cross-section_Aspect_Ratio} \Rightarrow h/b = 0.7. \quad \text{Equation 1.3}$$

The same problem is called an independent-variable problem if none of these relationships exist, thereby allowing the variables to vary independently of each other.

1.1.7 Based on Nature of Search Space

The nature of search space also defines an important classification of engineering design optimisation problems. Based on this, the two categories that are identified are known search space and unknown search space optimisation problems. The real-life optimisation problems in which the designers lack prior knowledge about the shape of search space, and about the location and performance of optimal points are classified as unknown search space optimisation problems (Rogerio *et al.*, 2000). As opposed to this, most theoretical problems (test cases), being lab-designed, are classified as known search space optimisation problems. The problem of Figure 1.1 can be classified as a known search space problem. Chapter 9 discusses the design of a turbine blade cooling system, which is an example of an unknown search space optimisation problem.

The nature of search space also classifies the engineering design optimisation problems as uni-modal and multi-modal based on the number of optimal solutions that the problem has. The problem of Figure 1.1 for minimising the end deflection under the given constraints is a uni-modal problem.

1.1.8 Miscellaneous Classifications

There are several other classifications of engineering design optimisation problems besides the ones mentioned above. Some of these miscellaneous classifications are outlined below (Rao, 1996).

- ◆ **Based on Nature of Equations Involved:** This classification is based on the nature of expressions that represent the objective functions in the optimisation problem. According to this classification, the engineering design optimisation problems can be classified as linear, non-linear, geometric and quadratic. Based on this criterion, the engineering design optimisation problems can also be classified as

continuous and discontinuous depending on whether the equations involved in the problem have any discontinuities.

- ◆ **Based on Nature of Design Variables:** Based on the nature of design variables, the engineering design optimisation problems can be classified as static and dynamic. In parameter or static optimisation problems, the design variables are independent of each other whereas in trajectory or dynamic optimisation problems, the design variables are all continuous functions of some other variable(s). Another perspective of this classification is provided by Schwefel (1995), based on time-dependence of the optimisation problems.
- ◆ **Based on Permissible Values of Design Variables:** Depending on the values permitted for design variables, the engineering design optimisation problems can be classified as integer-valued, real-valued and hybrid (that involve both integer and real variables).

Table 1.1: Classification Schemes for Engineering Design Optimisation problems

Classification Schemes	Categories
Based on Number of Parameters	<ul style="list-style-type: none"> • Single-dimensional • Multi-dimensional
Based on Existence of Constraints	<ul style="list-style-type: none"> • Constrained • Unconstrained
Based on Number of Objective Functions	<ul style="list-style-type: none"> • Single-objective • Multi-objective
Based on Nature of Objective Functions	<ul style="list-style-type: none"> • Quantitative • Qualitative • Hybrid
Based on Separability of Functions (for Quantitative and Hybrid Problems)	<ul style="list-style-type: none"> • Separable • Inseparable
Based on Dependence among Variables	<ul style="list-style-type: none"> • Independent-variable • Dependent-variable
Based on Nature of Search Space	<ul style="list-style-type: none"> • Known Search Space • Unknown Search Space
	<ul style="list-style-type: none"> • Uni-modal • Multi-modal
Based on Nature of Equations Involved (for quantitative and hybrid problems)	<ul style="list-style-type: none"> • Linear • Non-linear • Geometric • Quadratic
	<ul style="list-style-type: none"> • Continuous • Discontinuous
Based on Nature of Design Variables	<ul style="list-style-type: none"> • Parameter or Static • Trajectory or Dynamic
Based on Permissible Values of Design Variables	<ul style="list-style-type: none"> • Integer-valued • Real-valued • Hybrid

The summary of classification schemes, described in this section is given in Table 1.1. These classifications enable the categorisation of problems based on their

prominent features. This facilitates the choice of a suitable algorithm for a given engineering design optimisation problem.

1.2 Introduction to EC Techniques as Efficient Optimisers

In the natural world, evolution has created an unimaginably diverse range of designs, having much greater complexity than mankind could ever hope to achieve. Inspired by this, researchers have started using the EC techniques that use the principles of evolution to guide the optimisation process. There are a number of benefits of evolutionary-based optimisation that justify the effort invested in this area. The most significant advantage lies in the gain of flexibility and adaptability to the task in hand, in combination with robust performance and global search characteristics (Back *et al.*, 1997). The evolutionary-based optimisation techniques use a population of solutions in each iteration, instead of a single solution. This enables them, in principle, to identify multiple optimal solutions in their final population.

These characteristics of the EC techniques also make them a suitable candidate for handling a combination of multiple features of engineering design optimisation problems in a single run. These features include the presence of multiple objectives, constraints and interaction among decision variables. As a consequence, the EC techniques are better suited to deal with engineering design optimisation problems as compared to their classical counterparts. This research, therefore, focuses on EC techniques for the development of optimisation algorithms for engineering design.

The majority of current implementations of evolutionary algorithms descend from four strongly related but independently developed approaches: Genetic Algorithms (GAs), Evolutionary Programming (EP), Evolution Strategies (ESs) and Genetic Programming (GP) (Back *et al.*, 1997). These approaches are defined below.

- ◆ The GAs are a type of search and optimisation algorithms that are based on the mechanics of genetics and natural selection.

- ◆ The EP, which was originally offered as an attempt to create artificial intelligence, relies on transformations depending upon a finite set of states and state transition rules.
- ◆ The ESs, which were initially designed for solving complex optimisation problems, involve the modification of behavioural traits of solutions.
- ◆ Finally, the GP is an automated method for creating a working computer program from a high-level problem statement of a problem. The GP does this by genetically breeding a population of computer programs using the principles of Darwinian natural selection and biologically inspired operations. (Deb, 2001).

Over the last decade, the GAs have been extensively used as search and optimisation tools in various problem domains, including engineering design. The primary reasons for their success over other EC techniques are their broad applicability, ease of use and global perspective (Goldberg, 1989). This is the main reason that has led to the choice of GAs as the principal EC technique for this research.

1.3 Introduction to GAs

The GAs are robust search and optimisation techniques that mimic natural evolution. They were introduced by Holland, and subsequently studied by De Jong, Goldberg, and others such as Davis, Eshelman, Forrest, Grefenstette, Koza, Michalewicz, Mitchell, Riolo, and Schaffer, to name only a few (Back *et al.*, 1997). Because of their robustness, the GAs have attracted a vast interest among the researchers all over the world (Goldberg, 1989).

The GAs work on a population of individuals, which represent alternative solutions to the given optimisation problem. For each of these individuals, a score called fitness value is allocated based on the objective function defined in the problem. The population for the next generation is created by applying selection, crossover and mutation operators (genetic operators) to the current population. The high performing individuals are selected for the mating pool, where they reproduce with other individuals to produce offspring. Through this process, the members of new generation attain a higher proportion of the characteristics possessed by the ‘good’ members of previous generation. As a result, each new generation stands a relatively

higher chance of finding the optimal value of the objective function (Goldberg, 1989).

When a GA is used, the first task is to encode the solutions in a way that is easy to store and manage. The representation of a solution is termed as chromosome. Chromosomes are often developed as binary strings or a list of real numbers. Based on this, the GAs are classified as binary-coded and real-parameter, the latter being ideally suited for handling problems with continuous search spaces. The genetic operators work on chromosomes to produce a new set. The crossover operation is performed by exchanging genetic information between two randomly selected chromosomes. After crossover, the mutation operator is individually applied to each chromosome. It randomly alters a small part of the chromosome to produce a new individual. Mutation can help the optimisation process by introducing new search areas that crossover alone might not reach (Goldberg, 1989). Figure 1.2 gives a schematic description of the GAs.

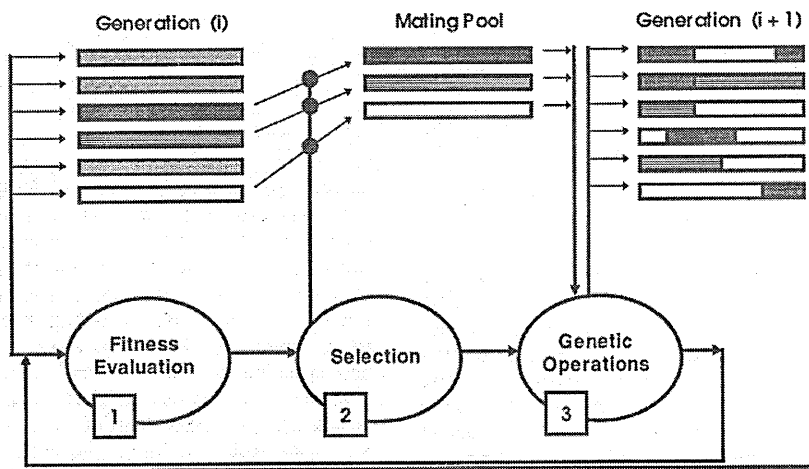


Figure 1.2: Schematic Description of GAs (Source: Jared et al., 1998)

1.4 'FLEXO': Parent Project of this Research

Optimisation algorithms do not find popular use in industry. There are a number of factors that are responsible for this situation. There is a general lack of robust optimisers that are capable of handling the complexity of engineering design

optimisation problems. Further, most of the currently available optimisation packages are not integrated within CAD/CAM systems, making their use cumbersome. To further augment this situation, all optimisation algorithms work on mathematical models of real-life designs, which attract little confidence from designers in the industry (Roy *et al.*, 2001). 'FLEXO' targets these inhibitors that prevent the use of optimisation algorithms in industry (Roy *et al.*, 2000a).

1.4.1 Previous Work

In the recent past, some work has been carried out in the field of flexible optimisation. Roy *et al.* (1998), Jared *et al.* (1998) and Mussa *et al.* (1998) specifically addressed the issue of enhancing the optimisation capabilities of existing CAD/CAM systems. Roy (1997), and Bentley and Wakefield (1998) also demonstrated the feasibility of developing a generic evolutionary design system. Keane (1996), Parmee (1996) and Greene (1998) have further attempted to develop a compact 'tool box' of robust optimisation techniques. However, the above-mentioned work has limited scope in terms of the robustness of optimisation techniques employed, their integration with CAD/CAM systems, and enhancement of designers' confidence. In contrast to the previous work that has only attempted to solve a specific optimisation problem in hand, 'FLEXO' adopts a holistic view of flexible optimisation within CAD/CAM environment (Rogero *et al.*, 2000).

1.4.2 Brief Description of 'FLEXO'

'FLEXO' is funded by EPSRC, which is the largest of the seven UK Research Councils. It funds research and postgraduate training in universities and other organisations throughout the UK (EPSRC, 2001). The industrial sponsors of 'FLEXO' are Nissan Technical Centre – Europe (NTC-E), UK and Structural Dynamics Research Corporation (SDRC), UK. NTC-E (UK) is one of the main technology centres of Nissan in Europe. It is responsible for carrying out research and development activities in the area of vehicle development to improve design, performance and costs of Nissan automobiles (Nissan, 2001). SDRC, which includes Nissan as one of its customers, is a major provider of CAD/CAM/CAE software. It

constitutes a part of the Product Lifecycle Management (PLM) Solutions of EDS, and includes I-DEAS and IMAGEWARE as its main products, with both having a vast client base (SDRC, 2001). ‘FLEXO’ involves the author and a fellow researcher, Mr. Olivier Munaux. The research detailed in this thesis is carried out by the author, and is a part of his contribution to the project.

‘FLEXO’ aims to develop a framework for flexible optimisation within Computer Aided Designing (CAD) / Computer Aided Manufacturing (CAM) environment using EC techniques. This framework would enable a CAD/CAM environment to select appropriate techniques and parameters for an optimisation task. The flexible optimisation wheel, shown in Figure 1.3, depicts the different combinations that are possible within this framework. This would provide a platform for dealing with various settings of tools and evaluation techniques, geometric modellers and optimisation algorithms (Roy *et al.*, 2000a).

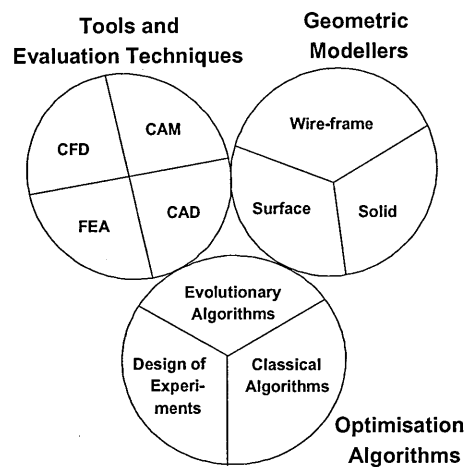


Figure 1.3: Flexible Optimisation Wheel (Source: Roy *et al.*, 2000)

The research objectives of ‘FLEXO’ can be summarised as follows (Roy *et al.*, 2000a).

- ◆ To identify the key industrial requirements on a flexible optimisation environment.
- ◆ To develop a ‘tool box’ of flexible optimisation techniques.
- ◆ To establish the core functionality required of a CAD system’s Application Programming Interface (API) to support such an environment.

- ◆ To implement the findings of the research through a prototype decision support system.

1.4.3 'FLEXO' Methodology

The 'FLEXO' framework is developed on the basis of the industrial requirements for flexible optimisation. The project involves the development of a 'tool box' containing flexible optimisation algorithms, capable of solving a variety of engineering design optimisation problems. The decision of developing the 'tool box' was guided by the No Free Lunch (NFL) theorem, which states that it is not possible to develop a single optimisation technique that is capable of simultaneously handling multiple features of optimisation problems in an efficient manner (Wolpert and Macready, 1997). Furthermore, the robustness of EC techniques has led to their use in this project for constructing the 'tool box'. The performance of this 'tool box' is also validated using a representative set of case studies reported in the literature. In order to provide the optimisation capability online, this 'tool box' is integrated with the CAD/CAM environment. This involves the development of generic Application Programming Interface (API) and a wrapper software for integrating the API with CAD system IMAGEWARE. Finally, the findings of this project are implemented through a prototype decision support system using an industrial case study on surface development. This case study provides the surface designers with an interactive tool for the attainment of aesthetic geometry (Rogerero *et al.*, 2000; Roy *et al.*, 2000a).

The key industrial deliverables of 'FLEXO' are as follows (Roy *et al.*, 2000a).

- ◆ Compilation of industrial requirements for optimisation.
- ◆ A 'tool box' of optimisation techniques that is suitable for the framework.
- ◆ An API specification to support optimisation.
- ◆ A prototype decision support system using a standard CAD/CAM environment.

1.5 Problem Statement and Motivation

The problem statement of this research is as follows.

Development of tools and techniques that are capable of dealing with the challenges of engineering design optimisation problems.

The development of robust optimisers enables the handling of the features of engineering design optimisation problems, such as multiple measures of performance (objectives), constraints and interaction among decision variables. This enhances the effectiveness of optimisation algorithms by giving them the capability of dealing with a wide variety of problems. In this way, one of the main inhibitors to their industrial use is addressed, and their popularity and relevance for industry could be enhanced. This is the main motivation for this research. Furthermore, the development of evolutionary-based optimisation techniques for engineering design optimisation problems makes a direct contribution to 'FLEXO', as discussed below.

'FLEXO' is an inter-disciplinary project involving design, geometric modelling and optimisation (Roy *et al.*, 2000a). The area and data flow perspectives of 'FLEXO', shown in Figure 1.4, depict the interactions between its three areas. This research deals with the optimisation aspect of 'FLEXO', shown as shaded area in Figure 1.4.

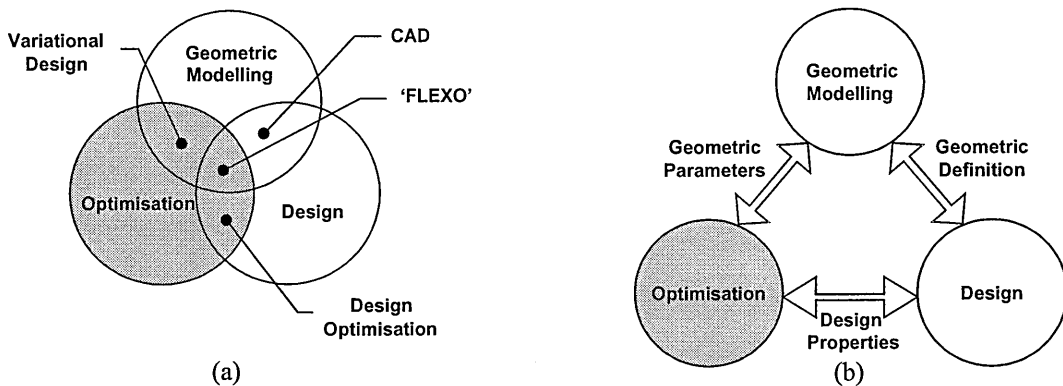


Figure 1.4: Perspectives of 'FLEXO' – (a) Area Perspective (b) Data Flow Perspective (Source: Roy et. al., 2000a)

The flexible optimisation 'tool box', which defines the shaded areas in Figure 1.4, optimises the model parameters based on prior problem knowledge, constraints and evaluation functions provided by the designers. Figure 1.5 presents the relative

location and the details of this ‘tool box’. This research aids the creation of this ‘tool box’ through the development of evolutionary-based optimisation techniques for engineering design optimisation. In this way, the research directly contributes to the objectives and industrial deliverables of ‘FLEXO’.

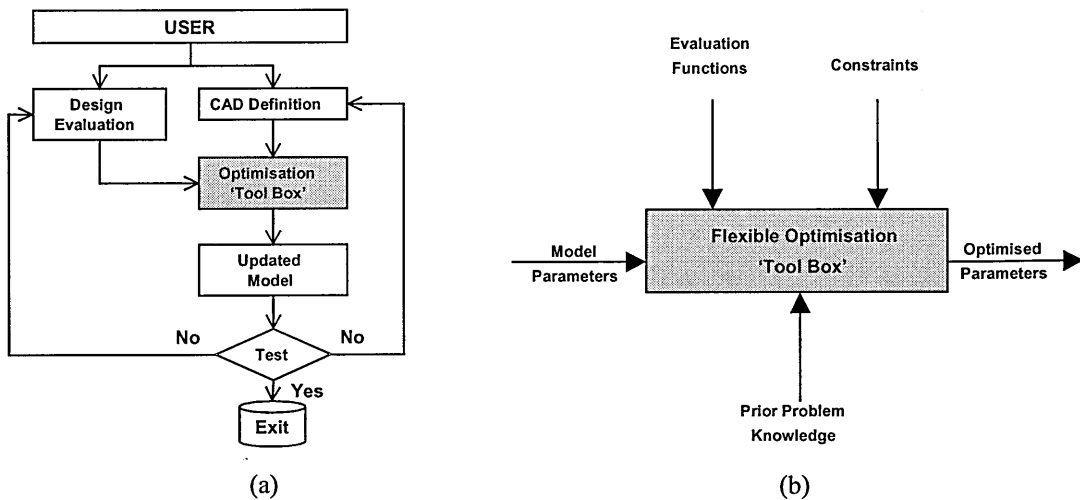


Figure 1.5: Flexible Optimisation ‘Tool Box’ – (a) Location (b) Details

1.6 Thesis Layout

The layout of this thesis is developed based on the story of this research. This story, which is pictorially depicted in Figure 1.6, aids the identification of individual chapters. A brief description of these chapters is given below.

Chapter 1 discusses the background of this research, briefly explaining the aim and objectives of its parent project, ‘FLEXO’. It presents the problem statement and motivation for this research, and describes its contribution to ‘FLEXO’.

Chapter 2 provides a survey of literature in the area of engineering design optimisation. It presents a critical analysis of the state-of-the-art evolutionary-based optimisation techniques and of the test problems used for the evaluation of these techniques.

Chapter 3 gives a brief description of this research, outlining its aim, objectives and scope. It also discusses the methodology that is adopted for ensuring that the aim and objectives of this research are attained.

Chapter 4 grounds the research within the industrial context based on the results of an industry survey.

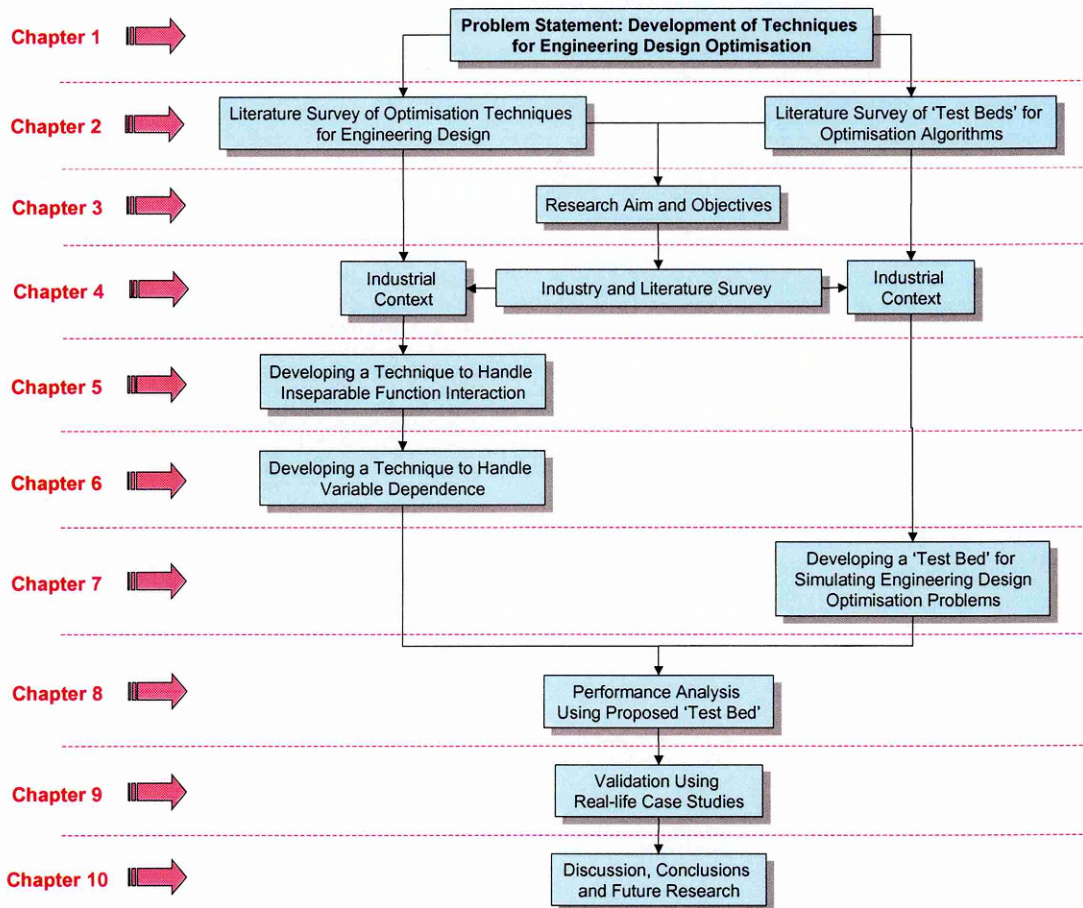


Figure 1.6: Thesis Layout

Chapter 5 develops a technique to handle a form of variable interaction, called inseparable function interaction, in multi-objective optimisation problems. It also compares the performance of the proposed algorithm with other high performing novel optimisation algorithms.

Chapter 6 proposes a technique to handle the second form of variable interaction, called variable dependence, in multi-objective optimisation problems. It also

evaluates the proposed algorithm with respect to other state-of-the-art optimisation algorithms.

Chapter 7 proposes a test bed that is capable of controlled and systematic simulation of the characteristics of engineering design optimisation problems, especially with respect to variable interaction. It compares the proposed test bed with the existing ones, and validates its behaviour using case studies.

Chapter 8 analyses the performance of the proposed optimisation algorithms (Chapters 5 and 6) using the test bed that is developed in Chapter 7. It also presents a discussion of the results that are obtained from these tests.

Chapter 9 validates the research using three industrial case studies: designs of a welded beam, a machine tool spindle and a turbine blade cooling system. The proposed optimisation algorithms are applied on these problems, and the results thus obtained are analysed, compared and discussed.

Chapter 10 concludes this thesis with a discussion on the generality of this research, contribution to knowledge, and limitations of the research methodology, proposed algorithms and test bed. It finally discusses the future research directions that could follow from this research.

2 A REVIEW OF LITERATURE

Optimisation refers to the process of finding one or more feasible solutions that correspond to the extreme values of one or more objectives. The need for finding such optimal solutions in a problem comes mostly from an extreme purpose, such as designing a solution for minimum possible cost of fabrication or for maximum possible reliability or others. Because of such extreme properties of optimal solutions, the optimisation algorithms are of great importance in practice, particularly in engineering design. The aim of this chapter is to give an overview of techniques for solving engineering design optimisation problems, and the test beds for evaluating these techniques. It attempts to achieve the following.

- ◆ *To provide an overview of engineering design optimisation approaches.*
- ◆ *To classify and describe the main EC techniques for handling three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction.*
- ◆ *To present a survey of test functions that can be used for evaluating these optimisation techniques.*

2.1 Engineering Design Optimisation Approaches

Design can be considered to represent a process that begins with recognition of the need and the conception of an idea to meet this need. Thus, in design decision making the main aim of the designer is to find a design solution that meets or closely meets the performance requirements of the design, while satisfying all the constraints. That defines a concept of ‘optimum design’ as a design that is feasible and also superior to all other feasible alternative designs. As mentioned in Chapter 1, optimisation is the process of selecting this ‘optimum design’, based on some pre-defined criteria, from a set of feasible alternative designs.

2.1.1 Manual versus Algorithmic Approaches to Optimisation

There are two ways to obtain an optimum design: through a manual process or by using an algorithmic approach (Roy, 1997). The manual process improves a design by repeated modifications. The design variables are changed one at a time. Designers often use their previous experience to decide changes in the design variables. They may easily improve a design involving few variables. If the design involves many variables this can pose a great challenge to the human designer, especially if he or she needs to consider variable interaction. If the designer does not have prior knowledge about the design the manual process can simply become a trial-and-error exercise. Thus, the manual approach can be very time-consuming and tedious.

On the other hand, the second approach (i.e. use of algorithms for optimisation) can simultaneously determine all the design variables so as to satisfy a set of constraints and optimise a set of objectives. To solve an optimisation problem, a computable design model is required. Many aspects of a design process can be represented by a formal model and are thus computable. However, some of the required designer's knowledge can be abstract and complex, and thus difficult to formalise. A design can therefore involve computable or quantitative formal knowledge as well as qualitative or abstract knowledge. In the absence of a formal model of the design process or at least a partial model, the manual approach may often become the only choice.

2.1.2 Algorithmic Approaches to Optimisation

Literature suggests a number of optimisation techniques for solving modelled optimisation problems. These techniques can be classified into two broad categories: classical and evolutionary.

Most classical algorithms use a point-by-point deterministic procedure for approaching the optimum solution. Such algorithms start from a random guess solution. Thereafter, based on a pre-specified transition rule, the algorithm suggests a search direction, which is often arrived at by considering local information. A uni-directional search is then performed along the search direction to find the best solution. This best solution becomes the new solution and the above procedure is

continued for a number of times. Figure 2.1 illustrates this procedure. Algorithms vary mostly in the way the search directions are defined at each intermediate solution (Deb, 1995).

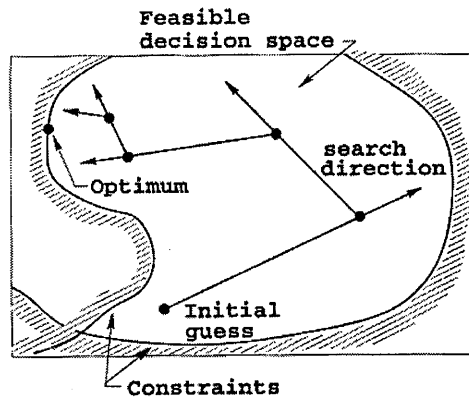


Figure 2.1: Classical Optimisation Algorithms (Source: Deb, 2001)

Classical optimisation methods can be classified into two distinct groups: direct methods and gradient-based methods. In direct search methods, only the objective functions and the constraint values are used to guide the search strategy. Some examples of these methods are the Simplex Search Method (Spendley *et al.*, 1962), Hooke-Jeeves Pattern Search Method (Hooke and Jeeves, 1961) and Powell's Conjugate Direction Method (Powell, 1964). On the other hand, the gradient-based methods use the first- and/or second-order derivatives of the objective functions and/or constraints to guide the search process. Some examples of these methods are Cauchy's (Steepest Descent) Method (Cauchy, 1847), Marquardt's Method (Marquardt, 1963) and Conjugate Gradient Method (Fletcher and Reeves, 1964). Since derivative information is not used, the direct search methods are usually slow, requiring many function evaluations for convergence. For the same reason, they can also be applied to many problems without a major change in the algorithm. On the other hand, gradient-based methods quickly converge near an optimal solution, but are not efficient in non-differentiable or discontinuous problems. In addition, there are some common difficulties with most classical direct and gradient-based techniques, as mentioned below (Deb, 1995).

- ◆ The convergence to an optimal solution depends on the chosen initial solution.

- ◆ Most algorithms tend to get stuck to a sub-optimal solution.
- ◆ An algorithm that is efficient in solving one optimisation problem may not be efficient in solving a different optimisation problem.
- ◆ Algorithms are not efficient in handling problems having a discrete search space.
- ◆ Algorithms cannot be efficiently used on a parallel machine.

The above-mentioned drawbacks of classical optimisation techniques have led to the growth of research in the field of evolutionary computation. The EC techniques can handle most of the drawbacks of classical algorithms, and as discussed in the previous chapter, the characteristics of the EC techniques, especially their robustness, make them suitable for dealing with the features of engineering design optimisation problems. This research, therefore, focuses on EC for the development of optimisation algorithms.

2.1.3 Optimisation Approaches to Handle Uncertainty

Designers typically require a lot of information for design decision making. Information is collected from the laws of physics, previous experiences, available literature, logical deductions and designers' intuition. Some of the information may be imprecise and ambiguous, whereas some may be precise and structured. The designer often faces a challenge to manipulate this combination of precise and imprecise information in order to reach a decision. To achieve good decisions, the designers must be able to take an overview of the possible alternative actions at any point in the design process. The designers can then predict the results of more than one selected course of action. The predictions can be heavily influenced by various other industrial factors and also the market environment. For example, predictions about a design action can be affected by the impact of that decision on the manufacturing organisation responsible for implementing the decision and on the end user (that is the customer). The impact of the decision on the overall market (that is the market environment within which the industry operates) also needs to be assessed. With the dynamic nature of the industrial and market environment in many cases it becomes almost impossible to predict the outcome of a decision very

precisely. Design decisions that use precise information from historical data, scientific evidence, etc. can be said to be virtually certain. The decisions that involve designers' knowledge, intuition and judgement involve a certain degree of uncertainty. Uncertainty can also be caused due to the complex dynamic interactions within the industry, between the industry and the market environment, imprecision involved in the designers' knowledge and vagueness involved in the designers' language (Roy, 1997).

All the optimisation techniques discussed previously consider the design variables as deterministic which means that the parameters used in the optimisation problem have precise values. These approaches therefore do not take into account any uncertainty related to design variables. In order to address uncertainty, probability theory has been widely accepted and applied in engineering design. In this theory, some statistical knowledge of random variables such as their mean values and standard deviations is used to address uncertainty. The first application of random variables in optimisation problems was studied by Charnes and Cooper (1959), in which a stochastic optimisation problem is converted into an equivalent deterministic one by using the chance constraint programming technique. Although there are many successful applications of this technique in the literature, it is not sufficient for problems that have highly non-linear performance functions. A number of methods are proposed in the recent past to overcome the weaknesses of this technique (Nikolaidis and Burdisso, 1988; Reddy *et al.*, 1994; Wang and Grandhi, 1996; Kaymaz *et al.*, 1998). The method proposed by Kaymaz *et al.* (1998) has a strong potential for dealing with real-life optimisation problems. This algorithm combines the response surface method with Monte Carlo simulation. The polynomial function (the response function) reduces the number of repetitions of an expensive analysis method such as Finite Element Analysis (FEA) (McMahon and Meng, 1996), and direct Monte Carlo simulation carries out the reliability calculation within the optimisation process.

2.1.4 Optimisation Approaches involving FEA/CFD

It is very common in engineering design optimisation problems to have the involvement of computationally expensive analysis techniques such as the FEA and Computational Fluid Dynamics (CFD). A number of researchers have discussed the integration of these techniques with the computer-based design systems (Kaymaz *et al.*, 1998; Tilley and Burrows, 1995). Literature reports some techniques for reducing the number of repetitions of these analyses during the optimisation process. The Monte Carlo method is one of the most widely used simulation approximations. The approach involves generating a sufficiently large set of observations to reproduce the statistical characteristics of the underlying population that the observations are taken from. The demerit of the approach is its computational cost, since the accuracy of the result largely depends on the number of samples used. Presently, there are efforts to improve the technique in two respects. One is to find approaches to reduce the size of sample as far as possible, such as the importance sampling technique in reliability analysis (Melchers, 1989). The second is to simplify the model or function that will be called many times in the sampling process. The influence surface method is one such approach to reduce the computational effort spent on the calculation of the function. An influence surface (or response surface) replaces a real analysis, and is often a surface in hyperspace identified by pre-analysis or experiment before implementing multiple analyses (e.g., in a simulation) (Meng and McMahon, 1998).

2.1.5 Sensitivity Analysis

Information concerning the sensitivity of engineering designs can be essential for engineering decision making. Sensitivity analysis provides the information on the performance of a design when there is some minor change in the values of the design variables. Sensitivities of a design can be defined in terms of the following (Emch and Parkinson, 1993; Sundaresan *et al.*, 1993).

- ◆ Design Solution Sensitivity: This refers to the sensitivity of a design solution performance within a defined neighbourhood.

- ◆ Design Variable Sensitivity: This is the effect of each design variable on the design solution performance within a defined neighbourhood.
- ◆ Constraint Sensitivity: Violations of constraints within the neighbourhood of a design define the constraint sensitivity of the design.

The study of the effect of varying the independent variables on a dependent variable requires the relationship between the dependent and independent variables to be known. An empirical method, known as design of experiments, is sometimes used to establish such relationship. For an empirical study all possible combinations of the values of the independent variables (also known as factors) are required to define the relationship using a statistical technique. This method of exhaustive trials is known as full factorial experiments.

In many cases, it is too expensive to run a full factorial experiment, for example a multi-dimensional real-life problem. In this situation, a fractional factorial experiment can be performed where a fraction of the full factorial experiments is considered. The price of running a fractional factorial experiment is the loss of some information regarding the independent variables and their relation to the dependent variable. Taguchi (1987a, 1987b) advocates a systematic approach and has developed several standard orthogonal metrics to define the fractional factorial experiments (Phadke, 1989). The use of the orthogonal metrics involves the least amount of information loss, especially if the variables do not interact with each other.

Taguchi's methodology (1987a, 1987b) assumes no interaction among variables. Thus, the analysis can be very reliable provided there is no or very little interaction among the design variables in the neighbourhoods of the design variables. One way of checking for the presence of interaction is to validate the additivity principle in the region. The additivity principle assumes that the result of each experiment is the superposition of the single factor effects plus the error due to this assumption and any repetition of the tests.

2.1.6 Engineering Design Support System

Chandrasekaran (1990) describes a design problem as a search problem in a large space for objects that satisfy multiple constraints. An object in the design space is equivalent to an acceptable value of a design variable. Only a very small number of objects in this space constitute satisfying, not to mention optimal, solutions. In order to make design decisions, practical strategies that radically shrink the search space are needed. A good design decision support tool can assist a designer in the search space reduction. The first step towards the search space reduction is to separate the information required for a design into two categories: formal and non-formal. The information obtained from the laws of physics, design catalogues, and design archives is structured and probably computable. Thus the information can be considered as contributing towards formal knowledge. The designer's experience, intuition and judgement can be abstract, unstructured and incomplete, thus they constitute the non-formal knowledge.

The optimisation approaches discussed above can reduce the search space by providing the designer with multiple preferred solutions. However, most of these approaches deal with formal knowledge only. During the last decade, some researchers have developed frameworks that can deal with both formal and non-formal knowledge. Yang and Sen (1994) describe an interactive multiple objective decision making procedure. The process describes a multi-objective preliminary design problem as a non-linear vector maximisation problem. The technique defines the design model using some computable functions. The methodology is a learning-oriented interactive technique that supports the designer in easily searching for preferred solutions following an adaptive approach. The technique allows designer's preferences to be progressively articulated with the generation of efficient design solutions. Through designer interaction the technique also makes sure that no unacceptable solutions is selected as a preferred design. Roy *et al.* (1996) propose an Adaptive Search Manager (ASM) that integrates a GA with knowledge-based software. The developed approach allows utilising both quantitative and qualitative information in engineering design decision making.

Most engineering design optimisation problems involve multiple objectives, constraints and interaction among decision variables. This chapter, therefore, provides a detailed description of the state-of-the-art evolutionary-based optimisation techniques that are reported in literature for dealing with these features of engineering design optimisation problems.

2.2 Evolutionary-based Techniques for Multi-objective Optimisation

Most engineering design optimisation problems are multi-objective in nature since they normally have several conflicting objectives, say, cost and performance, that must be satisfied at the same time. This has encouraged the growth of research in the field of multi-objective optimisation using evolutionary algorithms. The considerably large number of open questions in this area has provided a further impetus to the field (Coello, 2001). This section gives an overview of Evolutionary-based Multi-objective Optimisation Techniques (EMOTs) in terms of their types and features.

2.2.1 Problem Statement

Multi-objective optimisation, also known as multi-criteria, multi-performance or vector optimisation, can be defined as the problem of finding “a vector of decision variables, which satisfies constraints and optimises a vector function whose elements represent the objective functions.” (Osyczka, 1985). This problem can be formally stated as follows (Coello, 2001).

Find a vector $\vec{x}^* = [x_1^*, x_2^*, \dots, x_n^*]^T$, which satisfies the J inequality constraints,

$$g_j(\vec{x}) \geq 0, j = 1, 2, \dots, J; \quad \text{Equation 2.1}$$

the K equality constraints,

$$h_k(\vec{x}) = 0, k = 1, 2, \dots, K; \quad \text{Equation 2.2}$$

and optimises the vector function,

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \dots, f_M(\vec{x})]^T. \quad \text{Equation 2.3}$$

where $\vec{x} = [x_1, x_2, \dots, x_n]^T$ is the vector of decision variables, and each variable has upper and lower limits such that $x_i^{(L)} \leq x_i \leq x_i^{(U)}$; $i = 1, 2, \dots, n$.

As can be seen from this definition, the principles of multi-objective optimisation differ widely from those of single objective optimisation. The main goal in a single objective optimisation problem is to find the best solution (Deb, 1995). On the contrary, in a multi-objective optimisation problem, there are many objective functions, each of which may have a different individual optimal solution (Steuer, 1986). The interaction among different objectives gives rise to a set of compromised solutions, none of which can be considered to be better than the others without any further consideration (Deb, 1999a). These optimal solutions are called non-inferior, non-dominated or Pareto-optimal solutions. The boundary of the feasible region on which these solutions are located is called the Pareto front. A related concept that is used by many multi-objective optimisation algorithms for comparing alternative solutions is called the concept of domination. This concept is defined as follows (Coello, 1999).

A solution $\vec{x}^{(1)}$ is said to dominate the other solution $\vec{x}^{(2)}$, if both the following conditions are true.

- ◆ The solution $\vec{x}^{(1)}$ is strictly better than $\vec{x}^{(2)}$ in at least one objective.
- ◆ The solution $\vec{x}^{(1)}$ is no worse than $\vec{x}^{(2)}$ in all objectives.

In general, it is mathematically difficult to find an analytical expression of the Pareto front. So, iterative computation techniques are normally used for solving multi-objective optimisation problems (Coello, 1998). The two primary goals that these techniques must achieve are as follows (Deb, 1999a).

- ◆ To guide the search towards global Pareto-optimal region.
- ◆ To maintain population diversity in the Pareto front.

2.2.2 Classical versus Evolutionary Approaches

The classical ways of tackling multi-objective optimisation problems have primarily ignored the second goal mentioned above. Most methods, such as Weighted Sum Approach, ε -perturbation Method, Tchybeshev Method, Min-max Method and Goal Programming Method, convert multiple objectives into one objective using different heuristics (Miettinen, 1999; Sen and Yang, 1998; Steuer, 1986). Since multiple objectives are converted into one objective, the resulting solution to the single-objective optimisation problem usually depends on the parameter settings, for example, weights chosen for each objective in the Weighted Sum Approach. Since most classical methods use point-by-point approach, it is expected that one unique solution (hopefully a Pareto-optimal solution) will be found in each run of the optimisation algorithm. Thus, in order to find multiple Pareto-optimal solutions, the chosen optimisation algorithm needs to be used a number of times. Furthermore, the ability of a classical optimisation method to find a different Pareto-optimal solution in each simulation run is found to be dependent on the convexity and continuity of the Pareto-optimal region. Finally, the classical methods are not reliable in problems that involve uncertainties or stochasticities (Deb, 1999a).

Evolutionary approaches can successfully handle most of the above-mentioned drawbacks of classical algorithms. Since they use a population-based approach, it is intuitive that a number of Pareto-optimal solutions can, in principle, be obtained in a single simulation run. Therefore, these techniques have the potential of dealing with a variety of multi-objective optimisation problems (Coello, 2001).

2.2.3 Classification of EMOTs

Since the pioneering work of Rosenberg (1967), the research in the field of evolutionary multi-objective optimisation has grown considerably, especially in the last decade. In the recent past, a number of researchers have attempted to summarise the studies in this field. Notable among these are Fonseca and Fleming (1995), Tamaki *et al.* (1996), Horn (1997), Coello (1998), Fonseca and Fleming (1998a), Veldhuizen and Lamont (1998), Coello (1999), Deb (1999a), Zitzler and Thiele

(1999), and Deb (2001). The two main classifications of EMOTs that are commonly used by researchers are outlined below.

2.2.3.1 Function-based Classification

The function-based classification, which is the most commonly used, divides EMOTs based on the way the search is guided. Most of the researchers, who use this classification, also check for the presence of mechanisms for maintaining population diversity in the Pareto front. Some of the main diversity-preserving mechanisms that are reported in literature are Diversity through Mutation, Preselection, Crowding Model and Sharing Function Model, among many others (Deb, 2001). Fonseca and Fleming (1995), Tamaki *et al.* (1996), Coello (1998), and Fonseca and Fleming (1998a) have used this classification in their work. Another form of Function-based Classification is presented by Deb (2001), who classifies EMOTs based on the presence/absence of elitism in each of the techniques. As the name suggests, elitism favours the elite of a population by giving them an opportunity to be directly carried over to the next generation. In this way, a good solution is never lost unless a better solution is discovered, thereby ensuring that the fitness of the population-best solution does not deteriorate.

The function-based classification divides the EMOTs into three broad categories that are discussed below (The presence of a diversity-preserving mechanism and an elite-preserving operator is checked in each of the EMOTs.).

- ◆ **Plain Aggregating Approaches:** In these approaches, the multiple objectives are artificially combined, or aggregated, into a scalar function according to some understanding of the problem, and then the evolutionary algorithm is applied.
- ◆ **Population-based Non-Pareto Approaches:** In these approaches, the selection/reproduction, while applying the evolutionary algorithm, is performed by treating the objective functions separately. Therefore, these approaches do not use the concept of Pareto domination in carrying out the optimisation process.
- ◆ **Pareto-based Approaches:** In these approaches, the selection/reproduction, while applying the evolutionary algorithm, is performed by referring not only to the objective values themselves but also to their Pareto dominance property.

2.2.3.2 User-based Classification

User-based classification has been used by Horn (1997), Fonseca and Fleming (1998a), and Veldhuizen and Lamont (1998) for classifying EMOTs. Since these techniques provide the user with a set of optimal solutions, the final choice has to be made by him/her based on some understanding of the problem. This classification divides the EMOTs into three categories based on the stages in the optimisation process when the user makes the final decision. These three categories are outlined below.

- ◆ *A Priori* Preference Articulation (First Decide then Search): In this case, the user gives his/her preferences for the objectives prior to the optimisation process.
- ◆ Progressive Preference Articulation (Search and Decide Together): Here, the decision making and the optimisation processes are intertwined. Partial preference information is provided by the user during the optimisation process.
- ◆ *A Posteriori* Preference Articulation (First Search then Decide): In this case, the user makes a choice from a set of efficient candidate solutions provided to him/her after the optimisation process.

2.2.3.3 Hybrid Classification

This research proposes a hybrid classification, which combines the two schemes discussed above. The aim of this classification is to provide both the researcher and the user with a clear understanding of various EMOTs in terms of their function and use. In this classification, each of the three categories of user-based classification is sub-divided based on the way the search is guided. Finally, the presence of a diversity-preserving mechanism and an elite-preserving operator in the technique is checked. Table 2.1 divides some of the important EMOTs using hybrid classification. **D** and **E** in this table respectively represent the presence of a diversity-preserving mechanism and an elite-preserving operator in the technique. The computational complexities of some of the main techniques listed in this table are: VEGA - $O(N/M)$, HLGA - $O(MN^2)$, MOGA - $O(MN^2)$, NSGA – larger of $O(MN^2)/O(nN^2)$, NPGA - $O(MN^2)$, REMEA - $O(MN^2)$, NSGA-II - $O(MN^2)$, DPGA - $O(Mn^2)$, SPEA - $O(MN^2)$, TDGA - $O(N^3)$ and PAES - $O(MN^2d)$; where M is the

number of objectives, N is the population size, n is the number of variables, η is the current size of the elite set, and d is a user-defined parameter in PAES.

Table 2.1: Hybrid Classification of EMOTs

EMOTs	A Priori (Before)	Progressive (During)	A Posteriori (After)
Plain Aggregating Approaches	<ul style="list-style-type: none"> Evolutionary Weighted Sum Approach (Syswerda and Palmucci, 1991) Evolutionary Goal Programming (Wienke <i>et al.</i>, 1992) 		<ul style="list-style-type: none"> Hajela's and Lin's GA-HLGA (D)(Hajela and Lin, 1992) Random Weighted GA-RWGA (D)(Murata and Ishibuchi, 1995)
Population-based Non-Pareto Approaches	<ul style="list-style-type: none"> Lexicographic Selection (Fourman, 1985) 		<ul style="list-style-type: none"> Vector Evaluated GA-VEGA (D)(Schaffer, 1985) Non-generational GA (D) (Valenzuela-Rendon and Uresti-Charre, 1997) Predator-Prey ES-PPES (Laumanns <i>et al.</i>, 1998)
Pareto-based Approaches		<ul style="list-style-type: none"> Tchebycheff Method (Steuer, 1986) Multi-Objective GA-MOGA (D)(Fonseca and Fleming, 1993) Evolutionary Co-design-EvoC (Hu, 1996) 	<ul style="list-style-type: none"> Vector-Optimised ES-VOES (D)(Kursawe, 1990) Niched Pareto GA-NPGA (D)(Horn and Nafpliotis, 1993) Non-dominated Sorting GA-NSGA (D)(Srinivas and Deb, 1994) Distance-based Pareto GA-DPGA (D,E)(Osyczka and Kundu, 1995) Thermo-Dynamical GA-TDGA (D,E)(Kita <i>et al.</i>, 1996) Strength Pareto Evolutionary Algorithm-SPEA (D,E)(Zitzler and Thiele, 1998a) Multi-Objective Messy GA-MOMGA (D,E)(Veldhuizen, 1999) Fast Elitist Non-dominated Sorting GA-NSGA-II (D,E)(Deb <i>et al.</i>, 2000) Pareto-Archived ES-PAES (D,E)(Knowles and Corne, 2000) Rudolph's Elitist Multi-objective Evolutionary Algorithm-REMEA (E)(Rudolph, 2001)

The hybrid classification gives a holistic view of the EMOTs, making the classification relevant for both the researcher and the user. It also aids in identification of a suitable EMOT for a given problem by matching the features of each category with the problem characteristics. Further, the classification draws

attention to the research trends and to the most popular group of techniques. In the following discussion, the features of three most commonly used categories of EMOTs (shown as shaded regions in Table 2.1) are discussed in order to identify their relative strengths and weaknesses in dealing with multi-objective optimisation problems.

2.2.4 Plain Aggregating A Priori Approaches

The main EMOTs that are classified as Plain Aggregating A Priori Approaches are Evolutionary Weighted Sum Approach and Evolutionary Goal Programming. These techniques, which have strong roots in classical methods, require aggregation of objective functions prior to the application of evolutionary algorithms. If the combination of objectives is possible, these are one of the most computationally efficient techniques.

The main strength of these approaches is that they are easy to understand and implement. These approaches are computationally efficient and guarantee that the results would be at least sub-optimal in most cases. They, however, require a priori problem knowledge for combining the objectives. This information is difficult to obtain in most real-life cases. They also require accurate scalar information on the range of objectives, which is computationally very expensive to obtain. Since these approaches aggregate the objectives, they produce only one unique solution (perhaps a Pareto-optimal solution) in a single run. Therefore, these algorithms need to be run a number of times to obtain multiple Pareto-optimal solutions. This, however, does not guarantee the diversity of solutions.

Some applications of these approaches that are reported in literature include solution of task planning and transportation problems, optimisation of the designs of plane trusses and planar mechanisms, generation of hyper-structures from a set of chemical structures, and solution of pollution containment problems (Coello, 1998).

2.2.5 Population-based Non-Pareto A Posteriori Approaches

The main techniques that fall in the category of Population-based Non-Pareto A Posteriori Approaches are VEGA, Non-generational GA and PPES. These techniques, which were initially developed to overcome the weaknesses of plain aggregating approaches, involve alternative population strategies ideally suited for solving those problems in which approximate optimal solutions are required in simple search spaces.

Since these approaches are easy to implement and are computationally less expensive, a number of their successful applications are reported in the literature. They also seem to overcome most of the weaknesses of Plain Aggregating Approaches. However, their main drawback is that the solutions they produce are mostly sub-optimal and are sensitive to the values of the control parameters. Most of these algorithms also have a bias towards individuals that excel in different aspects of performance, a problem that in genetics is known as 'speciation'. Another weakness of some of these techniques, like VEGA, is their inability to produce Pareto-optimal solutions in the presence of non-convex search spaces (Coello, 1998). Therefore, prior knowledge regarding the shape of search space is required for the application of these algorithms.

The main applications of these techniques reported in the literature are the solution of groundwater pollution containment problem, optimisation of gas supply networks and development of preliminary airframe designs (Coello, 1998).

2.2.6 Pareto-based A Posteriori Approaches

The main techniques that can be classified as Pareto-based A Posteriori Approaches are VOES, NPGA, NSGA, DPGA, TDGA, SPEA, MOMGA, NSGA-II, PAES and REMEA. The techniques in this category, being capable of tackling a vast variety of problems, form the centre-stage of research in the area of evolutionary-based multi-objective optimisation.

The main strength of these techniques is that in most cases they satisfy the two principal requirements of multi-objective optimisation: convergence to the Pareto-optimal front and maintenance of population diversity across the front. Another strength of these techniques is that they can handle multiple variables involved in optimisation problems. A number of researchers have confirmed that, in solving multi-objective optimisation problems, these techniques perform better than most others (Zitzler *et al.*, 1999). However, their main weakness is that the algorithms they use for checking non-dominance in a set of feasible solutions are computationally very expensive, and exhibit serious degradation of performance as the population size and the number of objectives is increased. In a number of cases, the performance of these approaches is also dependent on the values of control parameters. Finally, in most cases the application of diversity-preserving mechanisms to these techniques makes it necessary to evaluate the sharing factor, which is generally difficult to compute (Coello, 1998).

Of the vast variety of applications of these approaches reported in the literature, some are as follows: minimisation of the back scattering of aerodynamic reflectors, design of an electromagnetic system, optimisation of the investment portfolio and design of laminated ceramic composites (Coello, 1998).

2.2.7 Examples of EMOTs

A brief description of three state-of-the-art EMOTs is presented here. All these are Pareto-based A Posteriori Approaches, and involve the use of both diversity-preserving mechanisms and elite-preserving operators.

2.2.7.1 Thermo-Dynamical GA (TDGA)

The TDGA minimises Gibb's free energy term, constructed by using a mean energy term representing a fitness function and an entropy term representing the diversity term needed in a multi-objective optimisation problem. The fitness function is used as the non-domination rank of the solution obtained from the objective function values. By carefully choosing solutions from a combined parent and offspring

population, the algorithm attempts to achieve both the goals of an ideal multi-objective optimisation algorithm (Kita *et al.*, 1996).

The overall complexity of one generation of the TDGA is $O(N^3)$, where N is the population size. This complexity is higher than that of a number of other EMOTs. Further, the performance of this algorithm is strongly dependent on the setting of control parameters (Kita *et al.*, 1996).

2.2.7.2 Strength Pareto Evolutionary Algorithm (SPEA)

The SPEA also maintains a separate elite population, which contains the fixed number of non-dominated solutions found till the current generation. The elite population participates in the genetic operations and influences the fitness assignment procedure, which in this case is easy to calculate. A clustering technique is used to control the size of the elite set, thereby indirectly maintaining diversity among the elite population members. This clustering algorithm is parameter-less, which makes it attractive to use. In the absence of this clustering algorithm, the SPEA exhibits a convergence proof (Zitzler and Thiele, 1998a).

The overall complexity needed in each generation of the SPEA is $O(MN^2)$, where M is the number of objectives and N is the size of the population. The SPEA introduces an extra parameter, the size of the external population, which influences its performance. In this algorithm, since non-dominated sorting of the whole population is not used for assigning fitness, the fitness values do not favour all non-dominated solutions of the same rank equally. Moreover, in the SPEA fitness assignment, an external solution, which dominates more solutions, gets a worse fitness. This is justified when all dominated solutions are concentrated near the dominating solution. Since in most cases this is not true, the crowding effect should come only from the clustering procedure to prevent a wrong solution pressure for the non-dominated solutions (Zitzler and Thiele, 1998a).

2.2.7.3 Fast Elitist Non-dominated Sorting GA (NSGA-II)

The NSGA-II carries out a non-dominated sorting of a combined parent and offspring population. Thereafter, starting from the best non-dominated solutions, each front is accepted until all population slots are filled. This makes the algorithm an elitist type. For the solutions of the last allowed front, a parameter-less crowded distance-based niching strategy is used to resolve which solutions are carried over to the new population. This algorithm also has a crowded distance metric, which makes it fast and scalable to more than two objectives. The convergence of NSGA-II (without crowded comparison operator) can be proved. (Deb *et al.*, 2000).

The overall complexity of the NSGA-II is at most $O(MN^2)$, where M is the number of objectives and N is the population size. This algorithm loses its convergence property when the crowded comparison is used to restrict the population size. Further, the non-dominated sorting needs to be performed on a population of size $2N$, instead of a population of size N , as required in most algorithms (Deb *et al.*, 2000).

2.2.8 Summary

This section can be summarised with the following remarks.

- ◆ The two main goals of multi-objective optimisation are convergence to the Pareto front and maintenance of diversity across the front (Deb, 1999a).
- ◆ The concept of Pareto domination is a powerful way of encouraging convergence of solutions to the Pareto front.
- ◆ The spread of solutions can be encouraged by using diversity-preserving mechanisms with EMOTs.
- ◆ The use of elitism ensures that the ‘good’ solutions are not lost after being located once.
- ◆ In general, elitist EMOTs, such as NSGA-II, that use Pareto domination and diversity-preserving operators perform better than other multi-objective optimisation techniques.

2.3 Evolutionary-based Constrained Optimisation Techniques

Constraints are common in most engineering design optimisation problems. They arise due to practical/physical requirements or due to designers' special preferences. Whereas evolutionary computation techniques assume the existence of evaluation functions for feasible individuals, there is no uniform methodology for handling infeasible ones (Michalewicz, 1995). This section analyses the different types of Evolutionary-based Constrained Optimisation Techniques (ECOTs), together with their strengths and weaknesses.

2.3.1 Problem Statement

Constrained optimisation can be defined as the problem of locating optimal solutions in the presence of constraints in the search space. Typically, a constrained optimisation problem can be written as follows (Equation 2.4) (Deb, 2001).

$$\begin{aligned}
 & \text{Minimise / Maximise} \Rightarrow f_m(\vec{x}), m = 1, 2, \dots, M; & \text{Equation 2.4} \\
 & \text{Subject to} \Rightarrow g_j(\vec{x}) \geq 0, j = 1, 2, \dots, J; \\
 & h_k(\vec{x}) = 0, k = 1, 2, \dots, K; \\
 & x_i^{(L)} \leq x_i \leq x_i^{(U)}, i = 1, 2, \dots, n.
 \end{aligned}$$

Any evolutionary computation technique, applied to a particular constrained optimisation problem, should address the issue of handling infeasible individuals. In general, a search space S consists of two disjoint subsets of feasible and infeasible subspaces, F and U , respectively. A constrained optimisation technique should be able to locate the feasible optima without making any assumptions regarding the shape and connectivity of the subspaces. Since in constrained optimisation process, both feasible and infeasible individuals are encountered, the algorithm must be able to handle the following issues (Michalewicz, 1995).

- ◆ Comparison of two feasible individuals.
- ◆ Comparison of two infeasible individuals.

- ◆ Comparison of a feasible with an infeasible individual.
- ◆ Treatment of infeasible individuals: eliminate, repair or penalise.

2.3.2 Classical versus Evolutionary Approaches

In addition to the drawbacks mentioned Section 2.2.2, the classical approaches (e.g. Complex Method (Box, 1965), Sequential Linear Programming (Cheney and Goldstein, 1959) etc.) suffer from the following limitations in dealing with constraints in optimisation problems (Deb, 1995).

- ◆ These algorithms suffer from serious limitations in dealing with constraints in the presence of multiple objectives.
- ◆ Most of these algorithms demand some knowledge about the shape, size and nature of F and U subspaces. Further, the performance of a number of these algorithms is sensitive to the shape of constraint boundaries.
- ◆ These algorithms are not suitable for solving the problems that have a discontinuous feasible subspace (F).
- ◆ Finally, they are not reliable in the presence of uncertainties or stochasticities in constraints.

2.3.3 Classification of ECOTs

Richardson *et al.* (1989) claimed, “Attempts to apply GAs with constrained optimisation problems follow two different paradigms (1) modification of the genetic operators; and (2) penalising strings which fail to satisfy all the constraints.” This is no longer the case as a number of methods, which use different methodologies for handling constraints, have been proposed. A survey of constraint handling techniques in evolutionary computation is provided by Michalewicz (1995) for single objective optimisation, and by Deb (2001) for multi-objective optimisation.

Based on the algorithm for constraint handling, most ECOTs that are reported in literature can be classified into five main categories for both single- and multi-objective optimisation (Table 2.2) (Michalewicz, 1995; Michalewicz and Schoenauer, 1996; Michalewicz *et al.*, 2000; Roy *et al.*, 2000b; Deb, 2001). Each of these categories is briefly discussed in this section.

Table 2.2: Classification of ECOTs

ECOTs	Single-objective Optimisation	Multi-objective Optimisation
Methods based on Preserving Feasibility of Solutions	<ul style="list-style-type: none"> Baldwinian Method (Liu <i>et al.</i>, 2000) Lamarckian Method (Deb and Goel, 2001) 	<ul style="list-style-type: none"> Baldwinian Method (Liu <i>et al.</i>, 2000) Lamarckian Method (Deb and Goel, 2001)
Methods based on Penalty Functions	<ul style="list-style-type: none"> Static Penalty Method (Homaifar <i>et al.</i>, 1994) Dynamic Penalty Method (Joines and Houck, 1994) Dynamic Penalty Method (Michalewicz and Attia, 1994) 	<ul style="list-style-type: none"> Static Penalty Method (Srinivas and Deb, 1994) Static Penalty Method (Deb, 1999a)
Methods based on Feasible over Infeasible Solutions	<ul style="list-style-type: none"> Behavioural Memory Method (Schoenauer and Xanthakis, 1993) Powell and Skolnick's Method (1993) Death Penalty Method (Michalewicz, 1995) Deb's Method (2000) 	<ul style="list-style-type: none"> Jimenez-Verdagay-Gomez-Skarmeta's Method (Jimenez <i>et al.</i>, 1999) Death Penalty Method (Coello and Christiansen, 1999) Constrained Tournament Method (Deb, 2000) Ray-Tai-Seow's Method (Ray <i>et al.</i>, 2001)
Methods based on Decoders	<ul style="list-style-type: none"> Koziel and Michalewicz's Method (1998) 	<ul style="list-style-type: none"> Koziel and Michalewicz's Method (1998)
Hybrid Methods	<ul style="list-style-type: none"> Method of Waagen <i>et al.</i> (1992) Co-evolutionary Approach (Paredis, 1994) Method based on Multi-objective Optimisation Techniques (Michalewicz, 1995) Barbosa's Co-evolutionary Approach (Barbosa, 1996) 	<ul style="list-style-type: none"> Method based on Multi-objective Optimisation Techniques (Michalewicz, 1995)

2.3.3.1 Methods based on Preserving Feasibility of Solutions

In this approach, two feasible solutions, after recombination and mutation operations, create two feasible offspring. Here, one decision variable is manipulated using an equality constraint, either implicitly or explicitly. In the explicit case, the value of one of the decision variable is calculated using the constraint function, thereby allowing the GA to use one variable less. In the implicit case, the GA uses all the variables, but estimates the value of one using the constraint function. In both single- and multi-objective optimisation, there are two ways of handling this new value. In the Lamarckian Method, the new value is substituted in the individual vector, while in the Baldwinian Method, it is not. However, both the methods use the new values to compute objective functions and other constraints. Although the Lamarckian Method of repairing an infeasible solution seems desirable (Deb and Goel, 2001), the Baldwinian Method is found to be more efficient in some problems (Liu *et al.*, 2000).

These methods that are based on preserving feasibility of solutions, though computationally inexpensive and easy to implement, have limited practical significance due to their inability in handling inequality constraints.

2.3.3.2 Methods based on Penalty Functions

In many algorithms, an exterior penalty term (Deb, 1995; Reklaitis *et al.*, 1983), which penalises infeasible solutions, is used. In most of these cases, the penalties are based on the degree of constraint violations (Richardson *et al.*, 1989). In general, the penalty functions can be classified as static or dynamic based on their relationship with the generation number. In addition, there are adaptive penalties that can be incorporated in the chromosome structures (Michalewicz, 1995). In this way, these penalties are treated as variables of the problem. They adaptively take suitable values as the population grows.

Static penalties are independent of the generation number (Michalewicz, 1995). However, there are two main difficulties associated with them. Firstly, the users require extensive experimentation to find the values of the penalty parameters that would steer the search towards the feasible region. Secondly, the inclusion of the penalty term distorts the objective function space, making it difficult for the GA to locate the optimum (Deb, 1995). An example of an ECOT that uses static penalties for solving single-objective optimisation problems is that proposed by Homaifar *et al.* (1994). Most of the studies in multi-objective evolutionary optimisation use carefully chosen static penalties (Srinivas and Deb, 1994; Deb, 1999a).

In the case of dynamic penalties, which were introduced to tackle the drawbacks of static penalties, the penalty parameter is changed with the generation number (Michalewicz, 1995). Some methods that fall in this category are those of Joines and Houck (1994), and Michalewicz and Attia (1994). The use of these penalties is not trivial and only partial analysis of their performance is available. Further, all these methods require exogenous parameters, which must be tuned for each problem (Michalewicz, 1995).

2.3.3.3 Methods based on Feasible over Infeasible Solutions

In these methods, a clear distinction is made between the feasible and infeasible solutions in the search space. In the simplest of these methods, called the Death Penalty Method (Michalewicz, 1995; Coello and Christiansen, 1999), the infeasible solutions are completely rejected. In both single- and multi-objective optimisation, this method works only when F has a simple shape, and constitutes a reasonable part of S (Michalewicz, 1995). In the Behavioural Memory Method (Schoenauer and Xanthakis, 1993), which deal with constraints in single-objective optimisation, all the constraints are considered in a sequence. In another approach (Powell and Skolnick, 1993), which is also a constrained single-objective optimisation technique, a heuristic rule (suggested earlier by Richardson *et al.*, 1989) that states, “every feasible solution is better than all infeasible solutions”, is used to process infeasible solutions. A recent study (Deb, 2000) suggested an approach that does not need any penalty parameter. One fundamental difference between this approach and the approach of Powell and Skolnick (1993) is that here the objective function value is not computed for any infeasible solution. This method uses a niched binary tournament selection operator, where two solutions are compared in a tournament only if their Euclidean distance is within a pre-specified limit. Niching is used here to maintain diversity among the feasible solutions. In this tournament selection, the following scenarios are always assured (Deb, 2001).

- ◆ Any feasible solution is preferred to any infeasible solution.
- ◆ Among two feasible solutions, the one having a better objective function is preferred.
- ◆ Among two infeasible solutions, the one having a smaller constraint violation is preferred.

This Constrained Tournament Method (Deb, 2000) can also be extended to deal with multi-objective optimisation problems. Jimenez-Verdegay-Gomez-Skarmeta’s Method (Jimenez *et al.*, 1999) and Ray-Tai-Seow’s Method (Ray *et al.*, 2001) are two other constrained multi-objective optimisation algorithms that fall in the

category of 'Methods based on Feasible over Infeasible Solutions'. The description of all these three methods is provided later in this section.

2.3.3.4 Methods based on Decoders

The infeasible individuals could be repaired for either evaluation or replacement (Orvosh and Davis, 1993). In this strategy, a chromosome stores information about how to fix an infeasible solution. For example, the chromosome may keep information about the ordering of decision variables, which may be altered to make a solution feasible. In some instances, a decoder imposes a mapping between a feasible solution and a decoded solution. One such example (for both single- and multi-objective optimisation) is the use of a homomorphous mapping between an n -dimensional cube and a feasible search space (Koziel and Michalewicz, 1998). These algorithms are particularly interesting for combinatorial optimisation problems due to the relative ease of repairing an infeasible solution in these problems.

2.3.3.5 Hybrid Methods

There are two types of hybrid methods. In the first type, a classical constrained optimisation method is combined with the existing operators of an evolutionary algorithm. Three such techniques that are reported in literature for single-objective optimisation are discussed here. In the method suggested by Waagen *et al.* (1992), the Hooke-Jeeves Pattern Search Method is combined with evolutionary algorithms. Paredis (1994) suggested another approach, called the Co-evolutionary Approach. Here, a population of decision variable vectors is evolved along with a population of constraints, with fitter solutions satisfying more constraints and fitter constraints being violated by fewer solutions. A different twist to this approach suggested by Barbosa (1996), uses a population of solutions and a population of Lagrange multipliers.

In the second type, the given objective function(s) and constraint violation measures are both treated as objectives of a multi-objective optimisation problem. One of the EMOTs (Section 2.2) can then be used to locate the optimal solutions (Michalewicz,

1995). This Method based on Multi-objective Optimisation Techniques can be used to solve both single- and multi-objective optimisation problems.

In the following discussion, three main algorithms that specialise in handling constraints in multi-objective optimisation problems are described.

2.3.4 Jimenez-Verdegay-Gomez-Skarmeta's Method

Jimenez *et al.* (1999) suggested a systematic constraint handling procedure that lies in the category of 'Methods based on Feasible over Infeasible Solutions'. Here, the feasible and infeasible solutions are carefully evaluated by ensuring that no infeasible solution gets a better fitness than any feasible solution. This algorithm uses the binary tournament selection in its core. In this case, as long as the decisions can be made with the help of feasibility and dominance of solutions, they are followed. However, when both solutions enter a tie with respect to feasibility and dominance considerations, the algorithm attempts to satisfy the second task of multi-objective optimisation by using a niching concept to encourage a less crowded solution.

This algorithm has $O(N^2)$ complexity, where N is the population size. This is comparable to that of other algorithms. Another advantage is that it uses the tournament selection operator, which has better convergence properties than proportionate selection operator.

This algorithm could be improved by restricting the niching computations to the members of the chosen comparison set, instead of the entire population. This would reduce its computational complexity. Further, by explicitly preserving diversity among infeasible solutions, this algorithm sacrifices the progress towards the feasible region. It also has a couple of additional parameters that a user must set right. In order to make the non-domination check less stochastic, a large comparison set is also needed here. Furthermore, this algorithm does not explicitly check the domination of participating solutions in a tournament.

2.3.5 Constrained Tournament Method

The Constrained Tournament Method, which also falls in the category of ‘Methods based on Feasible over Infeasible Solutions’, was proposed by Deb (2000). The original version of this method for dealing with single-objective optimisation problems is given in Section 2.3.3.3. Here, the modified version of this method for multi-objective optimisation is discussed.

In this algorithm, the definition of domination is modified. Before comparing two solutions for domination, they are checked for their feasibility. If one solution is feasible and the other is not, the feasible solution dominates the other. If the two solutions are infeasible, the solution with the smaller normalised constraint violation dominates the other. On the other hand, if both the solutions are feasible, the usual domination principle is applied. This Constrain-domination Principle can be applied with any EMOT. Its formal definition is given below.

A solution $\bar{x}^{(i)}$ is said to ‘constrain-dominate’ a solution $\bar{x}^{(j)}$, if any of the following conditions is true.

- ◆ Solution $\bar{x}^{(i)}$ is feasible and solution $\bar{x}^{(j)}$ is not.
- ◆ Solutions $\bar{x}^{(i)}$ and $\bar{x}^{(j)}$ are both infeasible, but solution $\bar{x}^{(i)}$ has a smaller constraint violation.
- ◆ Solutions $\bar{x}^{(i)}$ and $\bar{x}^{(j)}$ are both feasible, and solution $\bar{x}^{(i)}$ dominates solution $\bar{x}^{(j)}$ in the usual sense of Pareto domination, as defined in Section 2.2.1.

This is a penalty-parameter-less constraint handling approach. It does not require any extra computational burden, other than the constraint violation computations. Further, this strategy is generic and can be used with any EMOT. Since it forces an infeasible solution to be always dominated by a feasible one, no other constraint handling strategy is required here.

In solving simple problems, this algorithm exhibits better performance as compared to other constraint handling techniques. Moreover, since this approach requires domination checks to be performed with the constraint violation values, it has a slightly higher computational expense as compared to other techniques.

2.3.6 Ray-Tai-Seow's Method

Ray *et al.* (2001) suggested a more elaborate constraint handling technique, which is also a 'Method based on Feasible over Infeasible Solutions'. Here, the constraint violations of all constraints are not simply added together; instead, a non-domination check of the constraint violation is made. In this case, three different non-dominated sorting procedures are used. In addition to a non-dominated sorting of the objective functions, a couple of non-dominated sortings using the constraint violation values and a combined set of objective function and constraint violation values are needed to construct the new population.

The overall complexity of this algorithm is $O((M+J)N^2)$, where J is the number of constraints, M is the number of objective functions and N is the population size. This method emphasises the solutions that violate different constraints. In this way, the diversity is maintained in the population.

In a later generation, when all population members are feasible and belong to a sub-optimal non-dominated front, the algorithm stagnates. The performance of this algorithm is also dependent on the choice of appropriate values for its parameters. Further, the algorithm has a tendency of losing the diversity in its population. Finally, three non-dominated ranking and head-count computations make the algorithm more computationally expensive than the other algorithms discussed so far.

2.3.7 Summary

The following points summarise this section.

- ◆ There are only a few specialised techniques for handling constraints in multi-objective optimisation problems.
- ◆ The ease of understanding and implementation of penalty function approaches make them the most popular techniques for handling constraints in optimisation problems.
- ◆ Incorporation of constraint violations in the definition of Pareto domination is a powerful way of handling constraints in multi-objective optimisation problems.

- ◆ Niching can also be used with the above strategy to encourage diversity among solutions.
- ◆ Techniques, such as Constrained Tournament Method, that use the above concepts achieve better convergence and diversity of solutions as compared to other approaches.

Interaction among decision variables is inherent to a number of engineering design optimisation problems. In spite of its immense potential for real-life problems, lack of systematic research has plagued this field for a long time. The main reason for this was that no sophisticated technique was available in the past for effectively dealing with variable interaction. Inadequate hardware and software technologies also hampered the growth of research in this computationally complex field. However, in the last two decades, with the growth of these technologies, some research has been carried out in this area, especially in the fields of probability (Scott, 1992) and statistics (Draper and Smith, 1998). This has been further augmented in the recent past with the growth of computational intelligence techniques like EC, Neural Networks (NN) and Fuzzy Logic (FL) (Pedrycz, 1998).

In an ideal situation, desired results could be obtained by varying the decision variables of a given problem in a random fashion independently of each other. However, due to interaction this is not possible in a number of cases, implying that if the value of a given variable changes, the values of others should be changed in a unique way to get the desired results. The interaction among decision variables can be classified into two broad levels: inseparable function interaction and variable dependence. The EC techniques for handling both these types of variable interaction are discussed in this chapter.

2.4 Evolutionary-based Techniques for Handling Inseparable Function Interaction

This section presents a literature survey of the Evolutionary-based Techniques for handling Inseparable Function Interaction (ETIFIs) in optimisation problems.

2.4.1 Problem Statement

Inseparable function interaction is a form of variable interaction. It occurs when the effect that a variable has on the objective function depends on the values of other variables in the function (Taguchi, 1987a; Taguchi, 1987b). This concept of interaction can be understood from Figure 2.2. Figure 2.2(a) shows the case of no interaction between two variables A and B . Here, the lines representing the effect of variable A for the settings B_1 and B_2 of variable B are parallel to each other. Figure 2.2(b) and Figure 2.2(c) show two examples of the presence of interaction. The type of interaction in Figure 2.2(b) is sometimes called synergistic interaction and the one in Figure 2.2(c) is called anti-synergistic interaction (Phadke, 1989).

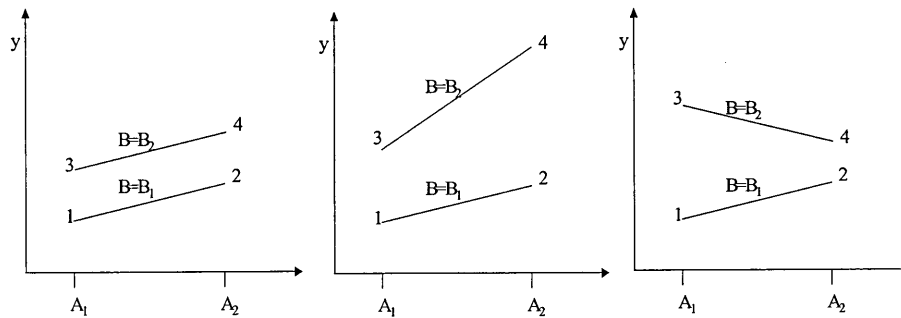


Figure 2.2: Examples of Interaction – (a) No Interaction (b) Synergistic Interaction (c) Anti-synergistic Interaction (Phadke, 1989)

The above discussion reveals that this interaction depends on the definition of objective functions, and manifests itself as cross-product terms. As an example, assume y in Figure 2.2 stands for A^2+B^2 , having no cross-product terms. Here, y_3-y_1 is equal to y_4-y_2 (Equation 2.5), making the two lines parallel and implying that there is no interaction between A and B in the given function.

$$y_3 - y_1 = (A_1^2 + B_2^2) - (A_1^2 + B_1^2) = (B_2^2 - B_1^2), \quad 5$$

$$y_4 - y_2 = (A_2^2 + B_2^2) - (A_2^2 + B_1^2) = (B_2^2 - B_1^2).$$

Let us take the other case in which y in Figure 2.2 stands for A^2+B^2+AB , having a

Let us take the other case in which y in Figure 2.2 stands for A^2+B^2+AB , having a cross-product term AB . Here, y_3-y_1 is not equal to y_4-y_2 (Equation 2.6). This makes

the two lines non-parallel implying interaction between variables in the given function.

$$\begin{aligned}
 y &= A^2 + B^2 + AB, && \text{Equation 2.6} \\
 y_3 - y_1 &= (A_1^2 + B_2^2 + A_1B_2) - (A_1^2 + B_1^2 + A_1B_1) \\
 &= (B_2^2 - B_1^2 + A_1B_2 - A_1B_1), \\
 y_4 - y_2 &= (A_2^2 + B_2^2 + A_2B_2) - (A_2^2 + B_1^2 + A_2B_1) \\
 &= (B_2^2 - B_1^2 + A_2B_2 - A_2B_1).
 \end{aligned}$$

In GA literature, the inseparable function interaction, as defined above, is termed as epistasis. The GA community defines epistasis as the interaction between different genes in a chromosome (Beasley *et al.*, 1993). In other words, it determines the extent to which the contribution to fitness of one gene depends on the values of other genes. Beasley *et al.* (1993) put forward the following three levels of epistasis.

- ◆ Level 0 indicates no epistasis.
- ◆ Level 1 indicates synergistic epistasis, where a particular change in one gene always produces a change in fitness of the same sign.
- ◆ Level 2 indicates anti-synergistic epistasis in which a change in one gene causes a change in fitness that varies in sign and magnitude depending on the values of other genes.

An alternative definition of epistasis is given by Reeves and Wright (1995a), who define it in terms of alleles. In this sense, the term epistasis is used to denote the effect of a combination of alleles on the chromosome fitness that is not merely a linear function of the effects of individual alleles. In general, epistasis can be thought of as expressing the degree of cross-terms in the fitness function.

Davidor (1991), and Reeves and Wright (1995a and 1995b) argued that understanding the distribution and level of epistasis is often an indicator of the difficulty of an optimisation problem. Davidor (1991) introduced the epistatic variance as a tool for the evaluation of interdependencies between genes, thus possibly giving clues about the difficulty of optimising functions with a GA (Fonlupt *et al.*, 1998). In recent years, Reeves and Wright (1999) have also attempted to put Davidor's methodology (Davidor, 1990; Davidor, 1991) on a firmer footing by

drawing on existing work in the field of Experimental Design (ED), which can be used to give insights into epistatic effects.

A number of real-life examples can be found in literature that involve this level of interaction. For example, the temperature (T) of an ideal gas varies with its pressure (P) and volume (V) as $T=kPV$, where k is the constant of proportionality (Equation 2.7). This equation has cross-product term PV clearly demonstrating the interaction between P and V in the definition of T (Tiwari *et al.*, 2001a).

$$T = kPV.$$

Equation 2.7

2.4.2 Classical versus Evolutionary Approaches

Section 2.2.2 listed a number of drawbacks that classical approaches face in dealing with multi-objective optimisation problems. The discussion in the previous section revealed that the presence of complex inseparable function interaction further enhances the challenges for multi-objective optimisation algorithms (Deb, 1999b). Since the classical approaches suffer from inherent drawbacks in handling complex interaction among decision variables, it becomes even more important to explore the field of EC for solving these optimisation problems. This is also supported by the fact that the EC can handle most of the drawbacks of classical algorithms, and has the potential of handling inseparable function interaction in optimisation problems.

2.4.3 Classification of ETIFIs

The success of a GA depends on its capability to grow ‘good’ building blocks (Thierens, 1995). In the presence of complex inseparable function interaction, it becomes difficult for a GA to meet this requirement. In these cases, it is essential to provide the simple GA with some additional features that can enable it to support the growth of ‘good’ building blocks. A number of techniques are reported in the literature that attempt to achieve this by preventing the disruption of important partial solutions.

Table 2.3: Classification of ETIFIs

ETIFIs	Classification of ETIFIs		
	Methods that Manipulate Representation of Solutions	Methods that Use Specialised Reproduction Operators	Methods that Avoid Race between Linkage Evolution and Allele Selection
Managing Race between Linkage Evolution and Allele Selection	<ul style="list-style-type: none"> • Method of Bagley (1967) • Method of Rosenberg (1967) • Method of Frantz (1972) • Method of Holland (1975) • Method of Goldberg and Lingle (1985) • Method of Schaffer and Morishima (1987) • Method of Goldberg and Bridges (1990) • Method of Levenick (1995) • Method of Paredis (1995) • Method of Smith and Fogarty (1996) • Linkage Learning GA (LLGA) (Harik, 1997) 	<ul style="list-style-type: none"> • Partially Mapped Crossover (PMX) (Reported by Harik, 1997) • Edge Recombination (Reported by Harik, 1997) • Enhanced Edge Recombination (Reported by Harik, 1997) • Order Crossover (Reported by Harik, 1997) • Molecular GA Crossover (Reported by Harik, 1997) 	<ul style="list-style-type: none"> • messy GA (mGA) (Goldberg <i>et al.</i>, 1989) • Gene Expression Messy GA (GEMGA) (Kargupta, 1998)
	Modelling Promising Solutions (Estimation of Distribution Algorithms (EDA)) (Muhlenbein and Paab, 1996)	No Interaction	Pairwise Interaction
<ul style="list-style-type: none"> • Population Based Incremental Learning (PBIL) algorithm (Baluja, 1994) • Univariate Marginal Distribution Algorithm (UMDA) (Muhlenbein and Paab, 1996) • Stochastic Hill Climbing with Learning by Vectors of Normal Distributions (SHCLVND) (Rudolf and Koppen, 1996) • compact GA (cGA) (Harik <i>et al.</i>, 1997) • Technique of Servet <i>et al.</i> (1997) • Extended PBIL for continuous domain (PBIL_c) (Sebag and Ducoulombier, 1998) 		<ul style="list-style-type: none"> • Method of Baluja and Davies, 1997 • MIMIC (De Bonet <i>et al.</i>, 1997) • Bivariate Marginal Distribution Algorithm (BMDA) (Pelikan and Muhlenbein, 1999) 	<ul style="list-style-type: none"> • Factorised Distribution Algorithm (FDA) (Muhlenbein and Mahnig, 1999a; Muhlenbein and Mahnig, 1999b) • Bayesian Optimisation Algorithm (BOA) (Pelikan <i>et al.</i>, 1999) • Learning FDA (LFDA) (Muhlenbein and Mahnig, 1999a; Muhlenbein and Mahnig, 1999b) • Extended Compact GA (ECGA) (Harik, 1999) • Polytree Approximation of Distribution Algorithms (PADA) (Soto <i>et al.</i>, 1999) • Continuous-domain EDA using a Flexible Probability Density Estimator (Gallagher <i>et al.</i>, 1999) • Multi-objective Mixture-based Iterated Density Estimation Evolutionary Algorithm (MIDEA) (Thierens and Bosman, 2001)

As shown in Table 2.3, ETIFIs can be classified into two broad categories based on the approach that are used for the prevention of building block disruption. The first class of these techniques manages the race between the building block growth and mixing (Harik, 1997). The second class attempts to model promising solutions for extracting some information from them in order to generate new solutions (Pelikan *et al.*, 1998). This section provides a brief overview of the techniques belonging to both of these categories.

2.4.4 Managing Race between Linkage Evolution and Allele Selection

Linkage is defined as the logical grouping of building block components to facilitate their growth and mixing. This strategy handles epistasis on the basis of the observation that the force that causes the evolution of linkage is, in effect, in a race against the force of allele selection. Therefore, a GA may fail if it is unable to control this struggle between the building block growth and mixing. In order to make the GA successful, this strategy proposes ways of managing this race (Harik, 1997). Most of the techniques that fall in this category work on single-objective optimisation problems in binary domains. These techniques can be classified as follows.

2.4.4.1 Methods that Manipulate Representation of Solutions

Various studies have shown that the struggle between linkage evolution and allele selection can be easily overcome when a problem's building blocks are tightly linked (Goldberg *et al.*, 1992; Goldberg *et al.*, 1993). The methods in this category attain this by manipulating the representation of solutions in the algorithm, in order to make the interacting components of partial solutions less likely to be broken by recombination operators. These methods evolve a problem's ordering alongside its solution, thereby enabling the GA to smooth the path of building block mixing. Some of the important studies in this area of dynamic adjustment of building block linkages were undertaken by Bagley (1967), Rosenberg (1967), Frantz (1972), Holland (1975), Goldberg and Lingle (1985), Schaffer and Morishima (1987), Goldberg and Bridges (1990), Levenick (1995), Paredis (1995), and Smith and Fogarty (1996). One of the latest algorithms in this category is Linkage Learning GA (LLGA). It was developed by Harik (1997). In this algorithm, the decision variables are mapped onto a circle. Their mutual distances evolve during optimisation, grouping together those with strong interaction so that recombination is less likely to disrupt them.

Many of the above-mentioned studies were undertaken before the logistics of building block mixing were well understood. They were inspired by a more holistic view of the GAs operation that considered the exploitation of tight building blocks

the Holy Grail to GA optimisation. The main drawback of these techniques is that the reordering operators that they use are often too slow and lose the race against selection, resulting in premature convergence to low quality solutions. Reordering is not sufficiently powerful in order to ensure a proper mixing of partial solutions before these are lost. This line of research has resulted in algorithms that evolve the representation of a problem among individual solutions (Harik, 1997).

2.4.4.2 Methods that Use Specialised Reproduction Operators

Some binary permutation operators, such as Partially Mapped Crossover (PMX), Edge Recombination, Enhanced Edge Recombination, Order Crossover and Molecular GA Crossover, have the potential of speeding-up the rate at which linkage evolution occurs (Harik, 1997). These crossover operators are directly useful for those optimisation problems that are naturally encoded as permutation, such as combinatorial optimisation problems. However, since the ordering problem can be considered as a combinatorial optimisation problem that the GA must tackle, these operators are potentially useful even within the confines of traditional GA optimisation.

2.4.4.3 Methods that Avoid Race between Linkage Evolution and Allele Selection

There are a number of techniques that entirely avoid the race between linkage evolution and allele selection. In the messy GA (mGA) (Goldberg *et al.*, 1989), the steps of building block identification and mixing are separate. In the first phase, the important building blocks are identified. This is done by simply applying the selection operator to them. The remaining solution components are substituted from a special solution called the template. The template is updated every few generations. In the second phase, the identified building blocks are mixed using selection and crossover operators. In the Gene Expression Messy GA (GEMGA) (Kargupta, 1998), the interaction in a problem is identified by manipulating individual solutions. These are used in order to improve the effects of recombination.

2.4.5 Modelling Promising Solutions

A different way to cope with the disruption of partial solutions is to change the basic principle of recombination. In this approach, instead of implicit reproduction of important building blocks and their mixing by selection and two-parent recombination operators, new solutions are generated by using the information extracted from the entire set of promising solutions. Global information about the set of promising solutions can be used to estimate their distribution, and new solutions can be generated according to this estimate. A general scheme of the algorithms based on this principle is called the Estimation of Distribution Algorithm (EDA) (Muhlenbein and Paab, 1996). A typical EDA approach to optimisation is illustrated in Figure 2.3.

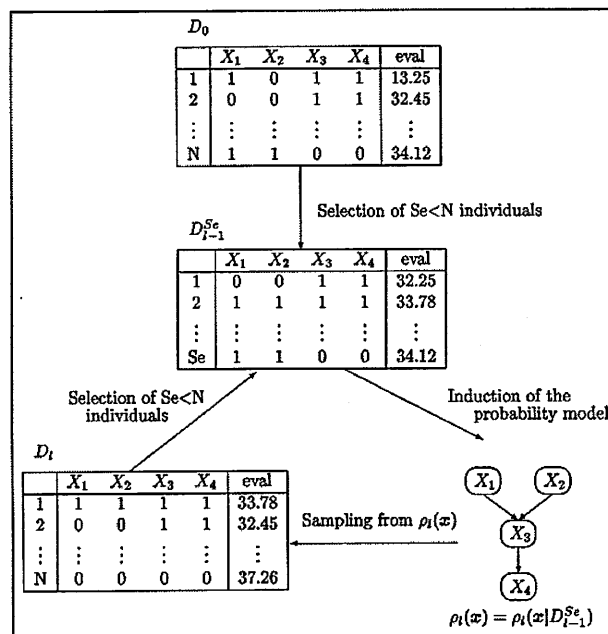


Figure 2.3: Estimation of Distribution Algorithm (EDA) Approach to Optimisation
(Source: Larranaga et al., 1999)

In EDA, the better solutions are selected from an initially randomly generated population of solutions like in the simple GA. The distribution of the selected set of solutions is estimated. New solutions are generated according to this estimate. The new solutions are then added into the original population, replacing some of the old

ones. The process is repeated until the termination criteria are met. However, estimating the distribution is not an easy task. There is a trade-off between the accuracy of the estimation and its computational cost (Larranaga *et al.*, 1999). The techniques that fall in this category can be classified as follows.

2.4.5.1 No Interaction

The simplest way to estimate the distribution of ‘good’ solutions is to consider each variable in a problem independently, and generate new solutions by only preserving the proportions of the values of all variables independently of the remaining solutions. This is the basic principle of the Population Based Incremental Learning (PBIL) algorithm (Baluja, 1994), the compact GA (cGA) (Harik *et al.*, 1997) and the Univariate Marginal Distribution Algorithm (UMDA) (Muhlenbein and Paab, 1996). There is theoretical evidence that the UMDA approximates the behaviour of the simple GA with uniform crossover (Muhlenbein, 1997). It reproduces and mixes the building blocks of order one very efficiently. The theory of UMDA, based on the techniques of quantitative genetics, can be found in Muhlenbein (1997). Some analysis of PBIL can be found in Kvasnicka *et al.* (1996). The PBIL, cGA and UMDA algorithms work well for problems with no significant interaction among variables (Muhlenbein, 1997; Harik *et al.*, 1997; Pelikan and Muhlenbein, 1999). However, partial solutions of order more than one are disrupted, and therefore these algorithms experience a great difficulty to solve problems with interaction among the variables.

2.4.5.2 Pairwise Interaction

First attempts to solve the problems that have interaction among variables were based on covering some pairwise interaction, for example, the incremental algorithm using the so-called dependency trees as a distribution estimate (Baluja and Davies, 1997), the population-based MIMIC algorithm using simple chain distributions (De Bonet *et al.*, 1997), or the Bivariate Marginal Distribution Algorithm (BMDA) (Pelikan and Muhlenbein, 1999). In the algorithms based on covering pairwise interaction, the reproduction of building blocks of order one is generated. Moreover,

the disruption of some important building blocks of order two is prevented. Important building blocks of order two are identified using various statistical methods. Mixing of building blocks of order one and two is guaranteed, assuming the independence of the remaining group of variables. However, covering pairwise interaction does not preserve higher order partial solutions. Moreover, interaction of higher order does not necessarily imply pairwise interaction that can be detected at the level of partial solutions of order two. Therefore, covering only pairwise interaction has been shown to be insufficient to efficiently solve problems with interaction of higher order (Pelikan and Muhlenbein, 1999).

2.4.5.3 Multiple Interaction

The Factorised Distribution Algorithm (FDA) (Muhlenbein and Mahnig, 1999a; Muhlenbein and Mahnig, 1999b) is capable of covering the interaction of higher order, and combining important partial solutions effectively. Here, a factorisation of the distribution is used for generating new solutions. The distribution factorisation is a conditional distribution constructed by analysing the problem decomposition. The FDA works very well on additively decomposable problems. The theory of UMDA can be used in order to estimate the time to convergence of the FDA. However, the FDA requires prior information about the problem in the form of problem decomposition and its factorisation. As input, this algorithm gets a complete or approximate information about the structure of a problem. Unfortunately, the exact distribution factorisation is often not available without computationally expensive problem analysis. Moreover, the use of an approximate distribution according to the current state of information represented by the set of promising solutions can be very effective even if it is not a valid distribution factorisation. However, by providing sufficient conditions for the distribution estimate that ensure a fast and reliable convergence on decomposable problems, the FDA is of great theoretical value. Moreover, for problems in which the factorisation of the distribution is known, the FDA is a very powerful optimisation tool.

The Bayesian Optimisation Algorithm (BOA), proposed by Pelikan *et al.* (1999), is also capable of covering higher order interaction. It uses techniques from the field of

modelling data by Bayesian networks in order to estimate the joint distribution of promising solutions. The class of distributions that are considered is identical to the class of conditional distributions used in the FDA. Therefore, the theory of the FDA can be used in order to demonstrate the power of this algorithm to solve decomposable problems. However, unlike the FDA, the BOA does not require any prior information about the problem. It discovers the structure of a problem, and identifies, reproduces and mixes building blocks up to a specified order very efficiently. Some other optimisation algorithms that have been proposed in the recent past to handle higher order interaction, while at the same time addressing the drawbacks of the FDA, are the Learning Factorised Distribution Algorithm (LFDA) (Muhlenbein and Mahnig, 1999a; Muhlenbein and Mahnig, 1999b), Extended Compact GA (ECGA) (Harik, 1999) and Polytrees Approximation of Distribution Algorithms (PADA) (Soto *et al.*, 1999).

It should be noted here that most of the EDAs mentioned here are proposed for combinatorial optimisation problems in binary domains. Literature also reports a few EDAs for continuous-domain problems. These include the Stochastic Hill Climbing with Learning by Vectors of Normal Distributions (SHCLVND) (Rudolf and Koppen, 1996), extended PBIL for continuous domain (PBIL_c) (Sebag and Ducoulombier, 1998) and the technique of Servet *et al.* (1997). A major drawback of these algorithms is that they fail in problems that have any significant interaction among their decision variables.

2.4.6 Examples of ETIFIs

It is evident that most of the ETIFIs discussed so far in this section work on single-objective optimisation problems in binary domains. However, for these techniques to be of any practical significance, they should be able to work in continuous domains in the presence of multiple objectives. Some attempts have been made by the researchers in the recent past to address these issues. In the following discussion, two new ETIFIs are analysed. Both work on continuous domains, but the first is a single-

objective optimisation technique and the second is a multi-objective optimisation technique.

2.4.6.1 Continuous-domain EDA using a Flexible Probability Density Estimator

Gallagher *et al.* (1999) extended the PBIL technique to real-valued search spaces. They proposed a powerful and general algorithmic framework that enables the use of arbitrary probability estimation techniques in evolutionary optimisation. To illustrate the usefulness of this framework, they also developed and implemented an evolutionary algorithm that uses a Finite Adaptive Gaussian Mixture Model Density Estimator. This method offers considerable power and flexibility in the forms of the density that can be effectively modelled. However, it is not suitable for those problems that have complex inseparable function interaction. It also cannot deal with multi-objective optimisation problems.

2.4.6.2 Multi-objective Mixture-based Iterated Density Estimation Evolutionary Algorithm (MIDEA)

Thierens and Bosman (2001) proposed a Multi-objective optimisation algorithm using a Mixture-based Iterated Density Estimation Evolutionary Algorithm (MIDEA). The MIDEA algorithm is a probabilistic model building evolutionary algorithm that constructs at each generation a mixture of factorised probability distributions. The use of a mixture distribution gives a powerful, yet computationally tractable, representation of complicated interaction. In addition, it results in an elegant procedure to preserve the diversity in the population, which is necessary in order to be able to cover the Pareto front. The algorithm searches for the Pareto front by computing the Pareto dominance among all solutions. As specific instantiations of the proposed algorithm, Thierens and Bosman (2001) have successfully implemented a mixture of universal factorisations and a mixture of tree factorisations for discrete multi-objective optimisation, and a mixture of continuous univariate factorisations and a mixture of conditional Gaussian factorisations for continuous optimisation problems. However, similar to other EDAs, this algorithm also specialises in binary

domains. The application of this algorithm to continuous domains requires it to be customised for each problem, especially with respect to the clustering algorithm. This customisation is mostly done based on trial-and-error. As Thierens and Bosman (2001) admit, this algorithm is still at an early stage of development, and is currently being developed for generalising it to enable its wider testing on a variety of problems.

2.4.7 Summary

The discussion in this section can be summarised as follows.

- ◆ A number of research questions need to be answered regarding the theory of epistasis, its measurement and its relationship with the difficulty of an optimisation problem.
- ◆ Most of the current research in the field of inseparable function interaction (epistasis) deals with single-objective optimisation in discrete domain.
- ◆ The few ETIFs that are available for dealing with continuous search spaces have limited capability in handling any significant inseparable function interaction.
- ◆ The development of ETIFs for dealing with real-valued, multi-objective optimisation problems is an important area of research. It needs to be addressed in order to develop techniques that can handle the challenges of engineering design optimisation problems.

2.5 Evolutionary-based Techniques for Handling Variable Dependence

This section presents a survey of Evolutionary-based Techniques for handling Variable Dependence (ETVD).

2.5.1 Problem Statement

Variable dependence, which is a form of variable interaction, occurs when the variables are functions of each other, and hence cannot be varied independently. Here, change in one variable has an impact on the value of the other. Unlike

inseparable function interaction that depends on the nature of objective functions, variable dependence depends on the nature of variables. Equation 2.8 depicts a dependent-variable optimisation problem, having multiple objectives and constraints.

$$\begin{aligned}
 & \text{Minimise / Maximise} \Rightarrow f_m(\bar{x}), m = 1, 2, \dots, M; & \text{Equation 2.8} \\
 & x_i = d_i(\bar{x}_{ind}), i = 1, 2, \dots, N_d; \\
 & \bigcup_{i=1}^{N_d} x_i = \bar{x}_{dep}; \\
 & \bar{x} - \bar{x}_{dep} = \bar{x}_{ind}; \\
 & \bar{x} = \bar{x}_{dep} \cup \bar{x}_{ind}; \\
 & \phi = \bar{x}_{dep} \cap \bar{x}_{ind}; \\
 & \text{Subject_to} \Rightarrow g_j(\bar{x}) \geq 0, j = 1, 2, \dots, J; \\
 & h_k(\bar{x}) = 0, k = 1, 2, \dots, K; \\
 & x_i^{(L)} \leq x_i \leq x_i^{(U)}, i = 1, 2, \dots, n; \\
 & N_d = \text{Number_of_dependent_variables}; \\
 & \bar{x}_{dep} = \text{Set_of_depedent_variables}; \\
 & \bar{x}_{ind} = \text{Set_of_independent_variables}.
 \end{aligned}$$

A typical example of this type of interaction is the case when the function y is defined as A^2+B^2 , where A is $\text{Random}(a,b)$ and B is $f(A)+\text{Random}(c,d)$ (Equation 2.9) (Tiwari *et al.*, 2001b).

$$\begin{aligned}
 & y = A^2 + B^2, & \text{Equation 2.9} \\
 & A = \text{Random}(a,b), \\
 & B = f(A) + \text{Random}(c,d).
 \end{aligned}$$

As can be seen, variable A is fully independent and can take any random value between a and b . On the other hand, variable B is not fully independent and has two components. The first component that is a function of variable A takes values depending on the values of A . The second component is a random number lying between c and d . The origin of this random component lies in one of the following two reasons or in a combination of both.

- ◆ This component may arise due to the deficiency of the mathematical model in accurately representing the real-life problem.

- ◆ Alternatively, it may represent noise in the data, arising due to the measurement/rounding/calculation errors, disturbances from the environment or other inaccuracies in the set-up.

It should be noted that in case of no dependence among decision variables, the function f does not exist. Therefore, a and b define the range of A , and c and d define the range of B .

The above example reveals that the presence of dependence among decision variables has the following effects on the search process.

- ◆ Both variables A and B cannot simultaneously take random values in their respective ranges. If variable A takes a value A_I , variable B can take only those random values that lie between $[f(A_I)+c]$ and $[f(A_I)+d]$. With the change in value of A , the range of random values that B can take also changes. So, the variables cannot be varied independently of each other.
- ◆ The above discussion implies that the presence of dependence among decision variables modifies the shape and location of variable search space. In case of no dependence among decision variables, both variables A and B can independently take random values in their respective ranges, making the A - B search space rectangular in nature. However, the presence of dependence makes the search space take the shape and location based on the nature of function $f(A)$.

2.5.2 Classical versus Evolutionary Approaches

Classical optimisation techniques suffer from serious limitations in handling the complexity of multi-objective optimisation problems (Section 2.2.2; Section 2.3.2; Section 2.4.2). As mentioned in the above discussion, the presence of variable dependence may introduce some additional features, such as bias (non-linearity), multi-modality, deception and discontinuity, in the optimisation problem. This makes it even more difficult for the classical optimisation algorithms to give satisfactory results. However, in recent years the growth of research in the fields of probability, statistics, EC, NN and FL has improved the situation. Literature reveals the potential of EC in removing most of the drawbacks of classical techniques (Deb, 2001). This makes the EC more suitable for dealing with dependent variable multi-objective optimisation problems than its classical counterparts.

2.5.3 Classification of ETVD

As discussed below, the evolutionary-based techniques need to follow the given two steps for solving optimisation problems that have dependence among their decision variables.

- ◆ Identification of Dependency Relationships (Step 1): In this step, the relationships that determine the dependency among decision variables are determined. The user may either explicitly know these relationships in the form of equations or need to infer them based on the data provided regarding decision variables. In both these cases, the user needs to ensure that the dependency relationships do not involve any cyclic dependencies.
- ◆ Classification of Variables (Step 2): The next step in solving these problems is to analyse the dependency equations for classifying the variables as independent and dependent. This allows the GA to operate on the independent variables, varying them independently of each other. For each alternative solution generated by the GA, the dependency equations are used to calculate the values of the dependent variables. In this way, the whole set of decision variables is determined, which is then used for evaluating the objective function(s).

Table 2.4: Classification of Techniques for Handling Variable Dependence

Techniques for Step 1: Identification of Dependency Relationships	<ul style="list-style-type: none"> • Regression Analysis (RA) (Frees, 1996; Draper and Smith, 1998; Evans and Olson, 2000) • Neural Networks (NNs) (Kolmogorov, 1957; Cybenko, 1989; Hertz <i>et al.</i>, 1991; Bishop, 1996; Richards, 1998; Gershenfeld, 1999) • Probabilistic Modelling (PM) (Pelikan <i>et al.</i>, 1998; Larranaga <i>et al.</i>, 1999; Gallagher <i>et al.</i>, 1999; Pelikan <i>et al.</i>, 1999; Evans and Olson, 2000)
Techniques for Step 2: Classification of Variables	<ul style="list-style-type: none"> • Tree Diagrams (TDs) (Banzhaf <i>et al.</i>, 1998; Richards, 1998; Larranaga <i>et al.</i>, 1999) • Direct Analysis (DA) (Gershenfeld, 1999)

Due to the lack of systematic research in the area of variable dependence, the literature in the field of optimisation does not report dedicated techniques that can deal with these problems. However, the survey of literature in related areas of research reveals some techniques that could form part of the above-mentioned two-step procedure for solving these problems. This section presents a critical analysis of the techniques for each of these steps. A summary of these techniques is provided in Table 2.4.

2.5.4 Techniques for Step 1: Identification of Dependency Relationships

The optimisation problems that involve variable dependence can be classified into two broad categories. In the first category of these problems, the user explicitly knows the equations that define the dependence among decision variables. However, the user still needs to ensure that the equations that are provided to him/her are free of cyclic dependencies. The next sub-section analyses the techniques that can be used for identifying the independent variables and removing any cyclic dependencies. In the second category, the dependency equations are unknown, but multiple sets of variable values are provided to the user from which the dependency relationships can be inferred. Literature reveals a number of sophisticated data modelling techniques that can be used for deriving the dependency relationships from the given data (Gershenfeld, 1999). Here, the use of three most popular data modelling techniques is analysed. As mentioned in Section 2.4.5, some of these techniques, especially the Probabilistic Modelling (PM), have also been applied in literature to deal with epistasis in optimisation problems (Pelikan *et al.*, 1999). Here, it is worth noting that epistasis, referred to as inseparable function interaction, and variable dependence are the two categories of variable interaction. As mentioned in the discussion that follows, the second step of the solution procedure is carried out based on the choice of the data modelling technique.

It should be noted that along with these two categories, there is another category of dependent variable optimisation problems, in which neither the dependency equations nor the data are available to the user. In these problems, only the compound fitness/evaluation model is available to the user. In this compound model, the independent input variables define the dependent variables; both of which define the objective functions. Since in these problems the relationships among dependent and independent variables are known, it is evident that they fall in the first category in which the dependency equations are provided to the user.

2.5.4.1 Regression Analysis (RA)

Regression Analysis (RA) is a tool for building statistical models that characterise relationships between a dependent variable and one or more independent variables, all of which are numerical (Evans and Olson, 2000). Figure 2.4 depicts a regression line through population means, and errors associated with individual observations. The non-linear multivariable RA, which is the most generic form of RA, attempts to fit a non-linear equation (of pre-defined degree) to the data, having one dependent variable and multiple independent variables. In this method, the coefficients of the non-linear equation are obtained in such a way that some function of the errors between the given values and the predicted values of the dependent variable is minimised. The most common approach for doing this is called least-squares regression, which minimises the sum of squares of the errors. The non-linear multivariable equation, which is attained from this analysis, is used for predicting the dependent variable in terms of the independent variables (Frees, 1996).

Advantages

RA is easy to understand and implement in a computer language. It has lesser computational expense than other sophisticated non-linear modelling techniques like the Neural Networks. Since it derives explicit dependency equations, it also gives better insight to the user regarding the relationships among decision variables (Draper and Smith, 1998). Further, due to its reasonable computational expense, the RA can be repeatedly applied on the given set of data, to determine the dependency equation, if any, for each variable. This makes it possible for the RA to identify multiple relationships among decision variables. This also lends it the capability of classifying the variables as dependent and independent, thereby eliminating the need for prior information regarding the nature of variables.

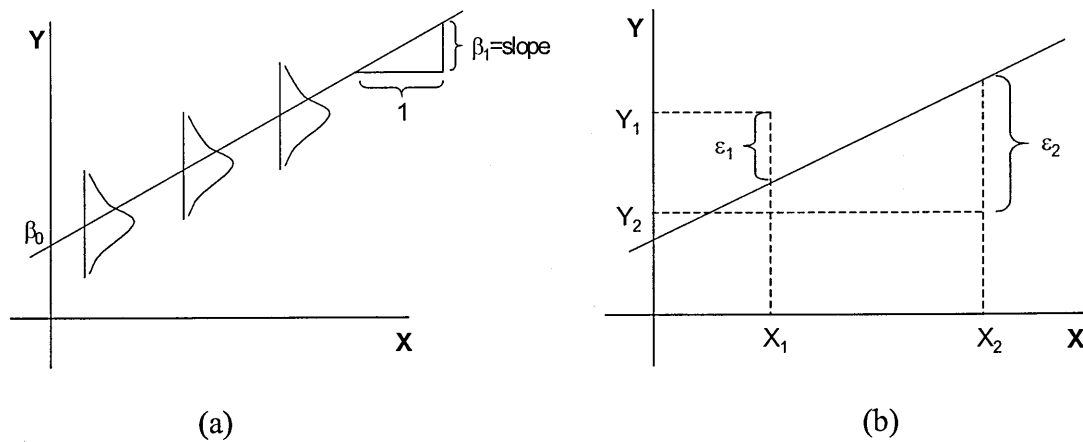


Figure 2.4: Illustration of Regression Model (a) Regression Line through Population Means (b) Errors Associated with Individual Observations (Source: Evans and Olson, 2000)

Disadvantages

The accuracy of RA depends on the degree of non-linear equation that is being used for modelling the given data. Therefore, this method suffers from limitations in modelling data that involve complex relationships (Frees, 1996). Also, each time a new set of data is added, the whole RA needs to be repeated with the updated data set (including both old and new data). Further, it is also not suitable for dealing with excessively noisy information (Evans and Olson, 2000).

2.5.4.2 Neural Networks (NNs)

The study of NNs started as an attempt to build mathematical models that worked in the same way that brains do. While biology is so complex that such explicit connections have been hard to make outside of specialised areas (Richards, 1988), the effort to do so has led to a powerful language for using large flexible non-linear models. The spirit of NN or connectionist modelling is to use fully non-linear functions (to handle the curse of dimensionality), and use a large number of terms (so that model mismatch errors are not a concern). Instead of matching the architecture of the model to a problem, a generic model is used, and careful training of the model is used to constrain it to describe the data (Gershenfeld, 1999). A typical NN with one hidden layer is depicted in Figure 2.5.

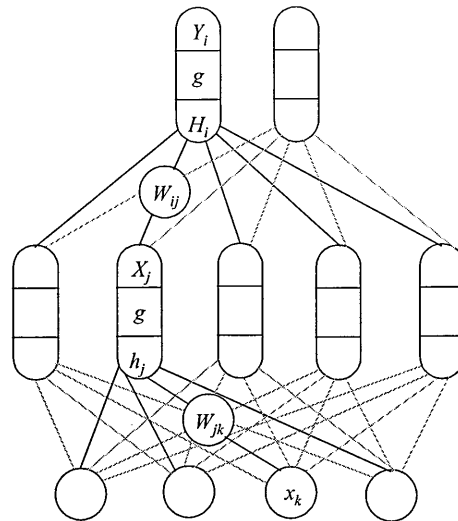


Figure 2.5: NN with One Hidden layer (Source: Gershenfeld, 1999)

Advantages

Since the NNs can handle the curse of dimensionality and the model mismatch errors, they are robust for simulating the dependency among decision variables (Gershenfeld, 1999). It has been shown that with one hidden layer a NN can describe any continuous function (if there are enough hidden units), and that with two hidden layers it can describe most functions (Kolmogorov, 1957 and Cybenko, 1989). Some types of NNs can also incorporate new sets of data without needing to recreate the whole model. This feature, together with the capability of dealing with noisy data, give them the potential of modelling noisy environments, where more information is added with time. A single NN also has the capability of modelling multiple relationships among decision variables (Bishop, 1996).

Disadvantages

The major drawback of NNs is that due to their huge computational expense, they cannot be repeatedly applied, using different sets of dependent and independent variables. Therefore, they require prior problem knowledge for classification of variables as independent and dependent, making them unsuitable for real-life optimisation problems, in which there is lack of prior information about the nature of variables (Hertz *et al.*, 1991). In terms of computer implementation, the NNs are more difficult than the RA, and have higher computational expense. Since they do

not provide dependency equations, they are also not as explicit as the RA (Bishop, 1996).

2.5.4.3 Probabilistic Modelling (PM)

In Probabilistic Modelling (PM), a probability distribution is constructed from the data provided by the user. This probability distribution, which acts as the model for representing the data, is expressed using a function $\rho(X = \mathbf{x})$, called Joint Generalised Probability Density Function (JGPDF). In this way, the model provides a characterisation of the possible values that its variables may assume, along with the probabilities of assuming these values. A typical probabilistic model is shown in Figure 2.6 (assuming that $\rho(X = \mathbf{x})$ is the JGPDF). This model can be used to create new sets of data that have the same relationship among their variables as in the original data provided by the user. Evans and Olson (2000) suggest a number of probability distributions for PM. These include Bernoulli, Binomial, Poisson and Bayesian distributions for discrete variable models, and Uniform, Normal, Triangular, Exponential, Lognormal, Gamma, Weibull, Beta, Geometric, Negative Binomial, hypergeometric, Logistic, Pareto, Extreme Value and Gaussian distributions for continuous variable models.

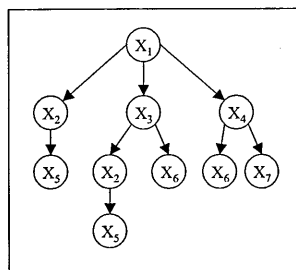


Figure 2.6: Example of Probabilistic Modelling (PM) (Assuming that $\rho(X = \mathbf{x})$ is JGPDF)

Advantages

PM is a very powerful data modelling technique since it does not require a priori classification of variables as dependent and independent (Pelikan *et al.*, 1999). Since PM provides an implicit model of the data, it is capable of dealing with complex

relationships among decision variables (Larranaga *et al.*, 1999). PM also has lower computational expense than the NNs but higher than that of the RA. Further, when a new set of data is added, it is not required here to re-create the model from scratch. PM makes it possible to update the model using just the new data. Therefore, it is simple and computationally inexpensive to update the model with the addition of more data. This makes the PM suitable for working in those noisy environments in which information slowly evolves with time (Pelikan *et al.*, 1998).

Disadvantages

The intricacy of concepts involved in PM makes it a difficult technique to understand and implement in a computer language. By not providing explicit dependency equations, it also does not give an insight into the dependency among decision variables (Evans and Olson, 2000). Finally, the field of multivariate PM for continuous variables is a subject of ongoing development, and a number of issues (such as setting of parameters, robustness, etc.) remain unanswered regarding its implementation. Some of the methods that fall in this category are Kernel, Mixture Model and Nearest Neighbour methods. The Adaptive Mixture algorithm, a type of Mixture Model, has the potential of modelling multivariate data from real-life problems. However, a number of research questions (such as scalability with number of variables and complexity of relationships, robustness, etc.) need to be answered before this algorithm could be used for any real-life application (Gallagher *et al.*, 1999).

2.5.5 Techniques for Step 2: Classification of Variables

After obtaining the dependency relationships, the user needs to carry out the next step, which is to identify the independent variables that form part of the GA chromosome. A number of tools are suggested in literature for analysing the dependency equations to classify the variables (as dependent and independent) and remove any cyclic dependencies. Some of these tools are discussed below.

2.5.5.1 Tree Diagrams (TDs)

The dependence among decision variables can be graphically represented using TDs, in which each node represents a variable in the problem. An example of a TD is shown in Figure 2.7. In this example, variable A is dependent on B, C and D, variable B on E, variable C on B and F, and variable D on F and G. Stepwise construction of TDs from the dependency equations can also be used to identify the cyclic dependencies. They are then resolved either by using some additional information from the source of equations or by eliminating the weaker leg of the cyclic dependency by comparing the coefficients of the corresponding terms in the dependency equations. TDs or their adaptations are used for visual representation of relationship among variables in a number of areas of research including GP (Banzhaf *et al.*, 1998), NNs (Richards, 1988) and probability (Larranaga *et al.*, 1999). The main motivation for the use of TDs is their ease of use and visualisation capabilities. However, TDs in their pure forms are difficult to be encoded in a computer language.

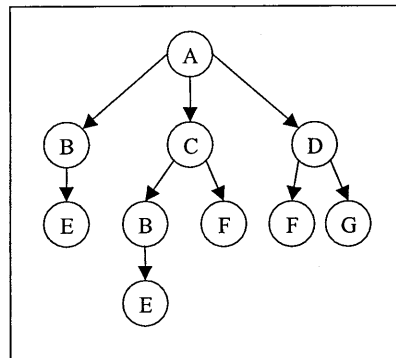


Figure 2.7: An Example of a Tree Diagram (TD)

2.5.5.2 Direct Analysis (DA)

The method of DA involving dependency equations could also be used for the classification of variables (Gershenfeld, 1999). It identifies independent variables as those that do not have any explicit equations for their definition. Each of the equations is then decomposed into independent variables. In doing so, all pairs of variables that play the roles of both independent and dependent variables for each other are identified. As in the case of TDs, these cyclic dependencies are then

resolved. This method is easy to be encoded in a computer language but is difficult to visualise.

2.5.6 Summary

The discussion in this section can be summarised with the following comments.

- ◆ Since variable dependence is defined by the nature of variables, rather than by that of the objective functions or constraints, the procedure for its handling is independent of the number of objectives and constraints in the problem.
- ◆ The solution of dependent-variable optimisation problems requires a two-step procedure to be appended to the EC: identification of dependency relationships (Step 1) and classification of variables (Step 2).
- ◆ The literature does not report any dedicated technique for handling variable dependence. However, some techniques, extracted from related areas of research, could form part of the overall two-step procedure for handling variable dependence.
- ◆ There is a need to develop a complete EC framework for dealing with dependent-variable optimisation problems.

The development of optimisation algorithms requires systematic and controlled testing. Therefore, it is required to have test functions that simulate the features of optimisation problems. The next three sections provide a survey of the optimisation test functions that are reported in the literature. Since the presence of multiple objectives, constraints and dependence among decision variables are common features of engineering design optimisation problems, these sections present a survey of existing test functions that are reported in literature for each of these categories. A categorised list of these test problems is provided in Table 2.5.

Table 2.5: Classification of Test Problems for Optimisation Algorithms

Test Problems	Classification of Optimisation Test Problems	
	Non-tuneable Test Problems	Tuneable Test Problems
Test Problems for Multi-objective Optimisation	<ul style="list-style-type: none"> • SCH1 (Schaffer, 1984) • SCH2 (Schaffer, 1984) • KUR (Kursawe, 1990) • FON (Fonseca and Fleming, 1995) • VNT (Viennet, 1996) • Poloni <i>et al.</i> (2000) 	<ul style="list-style-type: none"> • Tuneable Test Bed (Deb, 1999b) • ZTD1 to ZTD6 (Zitzlet <i>et al.</i>, 2000)
Test Problems for Constrained Multi-objective Optimisation	<ul style="list-style-type: none"> • BNH (Binh and Korn, 1997) • OSY (Osyczka and Kundu, 1995) • SRN (Srinivas and Deb, 1994) • TNK (Tanaka, 1995) 	<ul style="list-style-type: none"> • Tuneable Test Bed (Deb <i>et al.</i>, 2001)
Test Problems for Variable Interaction	Test Problems for Inseparable Function Interaction	Test Problems for Variable Dependence
	<ul style="list-style-type: none"> • Tuneable Test Bed (Deb, 1999b) (Partial fulfilment of requirements) 	<ul style="list-style-type: none"> • No test problem was observed in this category

2.6 Test Problems for Simulating Multi-objective Optimisation

The literature reports a number of test functions for simulating single-objective optimisation problems. Beale (1958), Rosenbrock (1960), Fletcher and Powell (1963), Smith and Rudd (1964), Box (1966), De Jong (1975), Schwefel (1995) and Gershenfeld (1999) proposed non-tuneable test problems that fall in this category. More recently, Michalewicz *et al.* (2000) proposed a parametric test bed for controlled simulation of the features of single-objective optimisation problems. All these test functions have contributed to the development of test beds for multi-objective optimisation, which is the principal focus of this section.

In multi-objective evolutionary computation, researchers have used many different test problems with known sets of Pareto-optimal solutions. Veldhuizen (1999) in his doctoral thesis outlined many such problems. A number of such popular test problems are presented here.

2.6.1 Non-tuneable Test Problems

Most of the test problems in the area of multi-objective optimisation fall in this category. Some of the most commonly used of these test problems are listed in Table 2.6.

Table 2.6: Non-tuneable Test Problems for Multi-objective Optimisation

Test Problems	Objective Functions (Minimisation)	Search Spaces
SCH1 (Schaffer, 1984)	$f_1(x) = x^2,$ $f_2(x) = (x-2)^2,$ $-A \leq x \leq A.$	Figure 2.8(a)
SCH2 (Schaffer, 1984)	$f_1(x) = -x, \text{ if } : x \leq 1,$ $= x - 2, \text{ if } : 1 < x \leq 3,$ $= 4 - x, \text{ if } : 3 < x \leq 4,$ $= x - 4, \text{ if } : x > 4,$ $f_2(x) = (x-5)^2, \dots$ $-5 \leq x \leq 10.$	Figure 2.8(b)
KUR (Kursawe, 1990)	$f_1(\vec{x}) = \sum_{i=1}^2 \left[-10 \exp\left(-0.2 \sqrt{x_i^2 + x_{i+1}^2}\right) \right]$ $f_2(\vec{x}) = \sum_{i=1}^3 \left[x_i ^{0.8} + 5 \sin(x_i^3) \right]$ $-5 \leq x_i \leq 5, i = 1, 2, 3.$	Figure 2.8(c)
VNT (Viennet, 1996)	$f_1(\vec{x}) = 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2),$ $f_2(\vec{x}) = (3x_1 - 2x_2 + 4)^2 / 8 + (x_1 - x_2 + 1)^2 / 27 + 15,$ $f_3(\vec{x}) = 1 / (x_1^2 + x_2^2 + 1) - 1.1 \exp[-(x_1^2 + x_2^2)],$ $-3 \leq (x_1, x_2) \leq 3.$	Figure 2.8(d)

In addition to the above test problems, Fonseca and Fleming (1995) proposed a two-objective optimisation problem, having n variables. This problem gives a concave Pareto front. Poloni *et al.* (2000) also proposed a two-variable, two-objective problem that gives a non-convex and disconnected Pareto-optimal set. Although researchers have used a number of other test problems, the fundamental problem with all of these is that the difficulty caused by such problems cannot be controlled. In most problems, neither the dimensionality/objectivity can be changes, nor the associated complexity (such as non-convexity, the extent of discreteness of the pareto-optimal region, etc.) can be changed in a simple manner. Furthermore, in most of these problems, it is difficult to establish what feature of an algorithm has been

tested. All these drawbacks of test problems led to the development of a tuneable test bed by Deb (1999b), as discussed below.

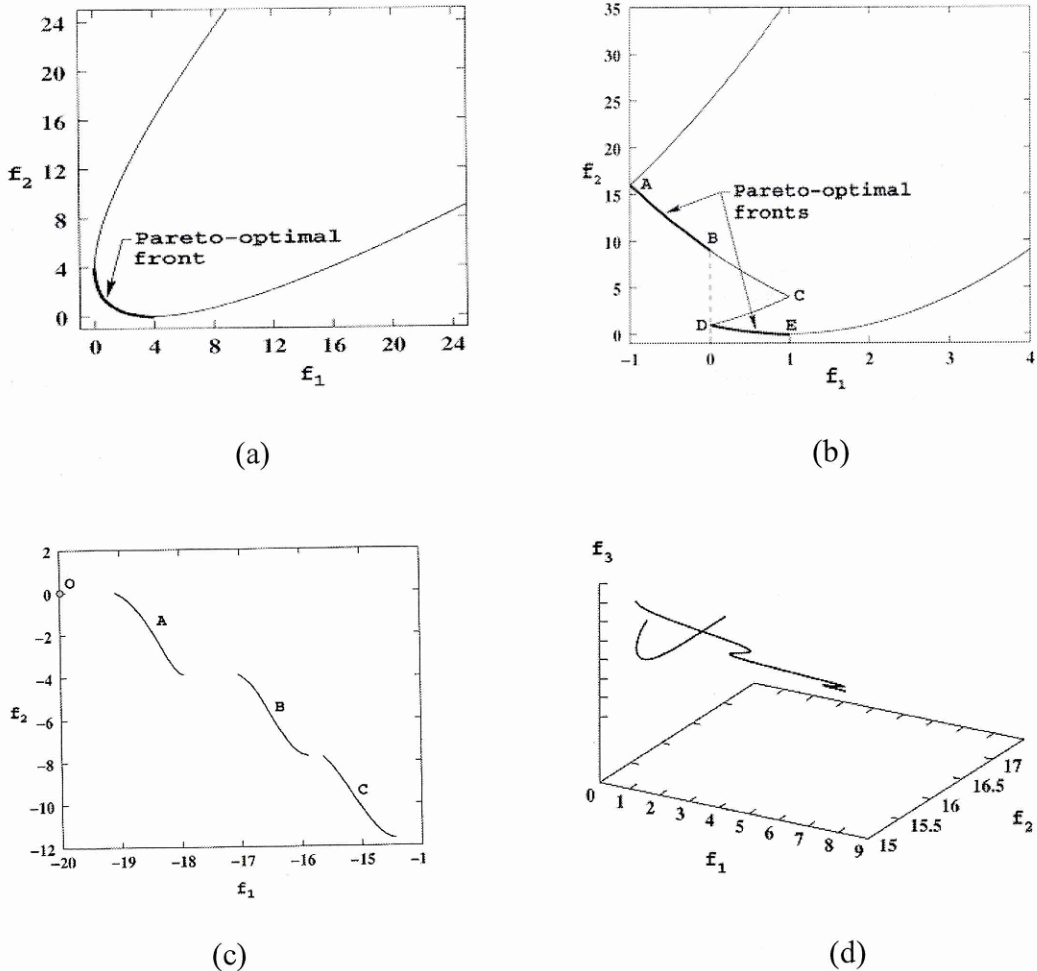


Figure 2.8: Non-tuneable Test problems for Multi-objective Optimisation – (a) SCH1 (b) SCH2 (c) KUR (d) VNT (Source: Deb, 2001)

2.6.2 Tuneable Test Problems

Deb (1999b) suggested a tuneable test bed based on the two tasks that a multi-objective optimisation algorithm must do well: convergence to Pareto front and maintenance of diversity across the front. Keeping in mind these two tasks, Deb (1999b) designed a problem where the difficulty involved in each of the above tasks can be controlled. He listed the following problem features that create difficulties in

converging to the Pareto-optimal front and in maintaining diverse Pareto-optimal solutions. Deb's test bed gives the control of tuning these features in a problem.

- ◆ Difficulties in Converging to Pareto-optimal Front
 - *Multi-modality.*
 - *Deception.*
 - *Isolated optimum.*
 - *Collateral noise.*
- ◆ Difficulties in Maintaining Diverse Pareto-optimal Solutions
 - *Convexity or non-convexity in the Pareto-optimal front.*
 - *Discontinuity in the Pareto-optimal front.*
 - *Non-uniform distribution of solutions in the search space and in the pareto-optimal front.*

In this scheme, an n -variable, two-objective optimisation problem is defined in terms of three functions (g , f_l , and h), as shown in Equation 2.10.

$$\begin{aligned} \text{Minimise} &\Rightarrow f_1(\vec{x}) = f_1(x_1, x_2, \dots, x_m), & \text{Equation 2.10} \\ \text{Minimise} &\Rightarrow f_2(\vec{x}) = g(x_{m+1}, \dots, x_n) \times h(f_1, g). \end{aligned}$$

Complications are avoided in this test suite by choosing f_l and g functions such that they take only positive values in the search space. By choosing appropriate functions for f_l , g and h , multi-objective problems with pre-defined features can be constructed, as shown below.

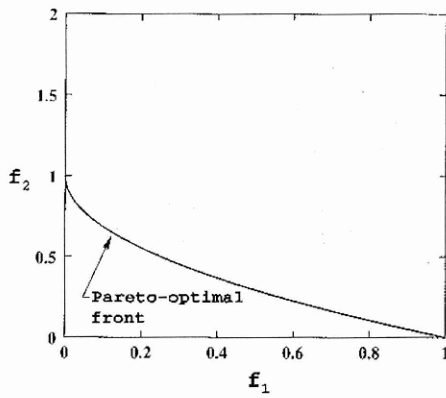
- ◆ Convexity or discontinuity in the Pareto front can be affected by choosing an appropriate h function.
- ◆ Convergence to the true Pareto front can be influenced by using a difficult g function (multi-modal, deceptive or others).
- ◆ Diversity in the Pareto front can be controlled by choosing an appropriate (non-linear or multi-dimensional) f_l function.

This scheme could also be extended to include more than two objectives, as shown below with M objectives (Equation 2.11).

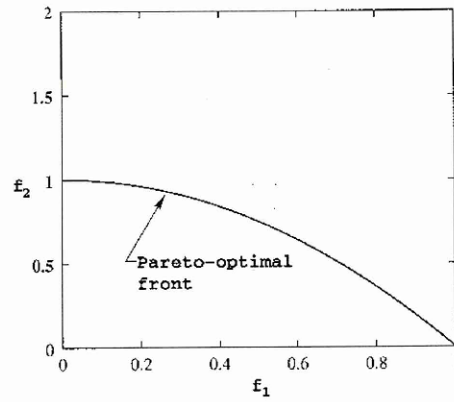
$$\begin{aligned}
 & \text{Minimise} \Rightarrow f_1(\bar{x}_1), && \text{Equation 2.11} \\
 & \text{Minimise} \Rightarrow f_2(\bar{x}_2), \dots, \\
 & \text{Minimise} \Rightarrow f_{M-1}(\bar{x}_{M-1}), \\
 & \text{Minimise} \Rightarrow f_M(\bar{x}) = g(\bar{x}_M)h(f_1(\bar{x}_1), f_2(\bar{x}_2), \dots, f_{M-1}(\bar{x}_{M-1}), g(\bar{x}_M)), \\
 & \text{Subject_to} \Rightarrow \bar{x}_i \in \mathfrak{R}^{|\bar{x}_i|}, i = 1, 2, \dots, M.
 \end{aligned}$$

Table 2.7: Zitzler-Deb-Thiele (ZDT) Test Problems

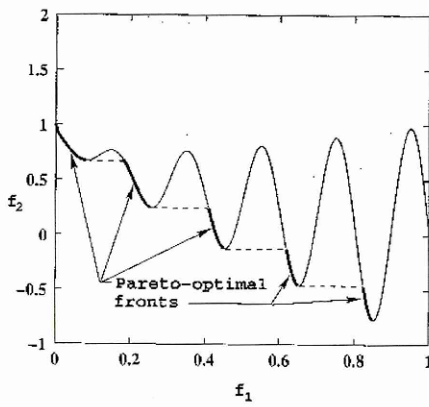
Tests	Vars.	Bounds	Objective Functions (Minimisation)	Search Spaces
ZDT1	30	[0,1]	$f_1(\bar{x}) = x_1,$ $f_2(\bar{x}) = g(\bar{x})[1 - \sqrt{x_1 / g(\bar{x})}],$ $g(\bar{x}) = 1 + 9(\sum_{i=2}^n x_i) / (n - 1).$	Figure 2.9(a)
ZDT2	30	[0,1]	$f_1(\bar{x}) = x_1,$ $f_2(\bar{x}) = g(\bar{x})[1 - (x_1 / g(\bar{x}))^2],$ $g(\bar{x}) = 1 + 9(\sum_{i=2}^n x_i) / (n - 1).$	Figure 2.9(b)
ZDT3	30	[0,1]	$f_1(\bar{x}) = x_1,$ $f_2(\bar{x}) = g(\bar{x})[1 - \sqrt{x_1 / g(\bar{x})} - (x_1 / g(\bar{x})) \sin(10\pi x_1)],$ $g(\bar{x}) = 1 + 9(\sum_{i=2}^n x_i) / (n - 1).$	Figure 2.9(c)
ZDT4	10	$x_1 \in [0,1]$ $x_i \in [-5,5]$ $i=2, \dots, 10$	$f_1(\bar{x}) = x_1,$ $f_2(\bar{x}) = g(\bar{x})[1 - \sqrt{x_1 / g(\bar{x})}],$ $g(\bar{x}) = 1 + 10(n - 1) + \sum_{i=2}^n [x_i^2 - 10 \cos(4\pi x_i)].$	Figure 2.9(d)
ZDT5	11	$x_1 \in 30 \text{ bits}$ $x_i \in 5 \text{ bits}$ $i=2, \dots, 11$	$f_1(\bar{x}) = 1 + u(x_1),$ $g(\bar{x}) = \sum_{i=2}^n v[u(x_i)],$ $v[u(x_i)] = 2 + u(x_i), \text{ if } : u(x_i) < 5,$ $= 1, \text{ if } : u(x_i) = 5,$ $h(f_1, g) = 1 / f_1(\bar{x}).$	Figure 2.9(e)
ZDT6	10	[0,1]	$f_1(\bar{x}) = 1 - \exp(-4x_1) \sin^6(4\pi x_1),$ $f_2(\bar{x}) = g(x)[1 - (f_1(\bar{x}) / g(\bar{x}))^2],$ $g(\bar{x}) = 1 + 9[(\sum_{i=2}^n x_i) / (n - 1)]^{0.25}.$	Figure 2.9(f)



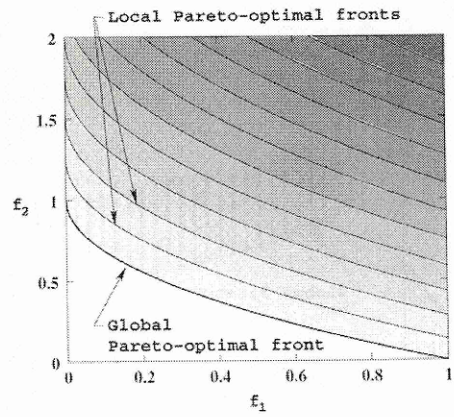
(a)



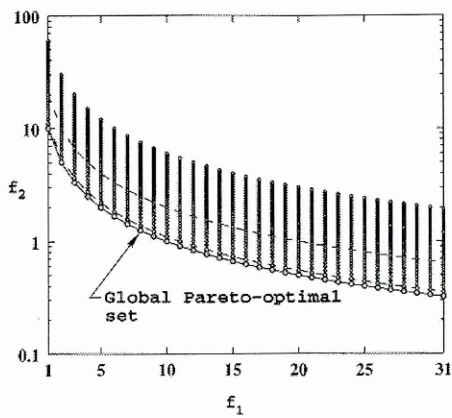
(b)



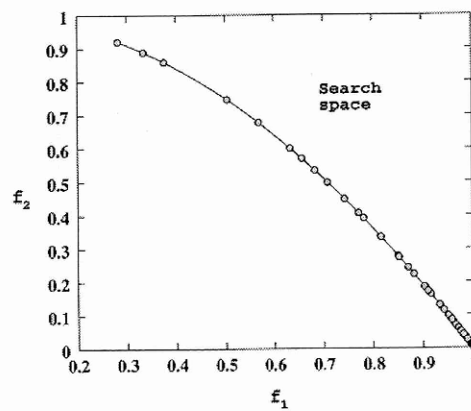
(c)



(d)



(e)



(f)

Figure 2.9: Zitzler-Deb-Thiele (ZDT) Test Problems – (a) ZDT1 (b) ZDT2 (c) ZDT3 (d) ZDT4 (e) ZDT5 (f) ZDT6 (Source: Deb, 2001)

In this equation, the decision variable vector \bar{x} is partitioned into M non-overlapping blocks as follows: $\bar{x} \equiv (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_{M-1}, \bar{x}_M)^T$.

Zitzler *et al.* (2000) framed six problems (ZDT1 to ZDT6) based on the above construction process. These Zitzler-Deb-Thiele (ZDT) test problems are summarised in Table 2.7.

Deb's tuneable test bed that is discussed here provides a generic framework for explicitly simulating the features of multi-objective optimisation problems in a controlled manner. It also exhibits scalability with the number of dimensions and objectives. However, it is incapable of handling constraints. Further, the degree of control that is provided by this test suite is limited since it does not provide parametric function prototypes for g, f_i and h .

2.6.3 Summary

This section can be concluded with the following comments.

- ◆ Most of the test problems in the area of multi-objective optimisation are not tuneable in nature.
- ◆ Deb (1999b) proposed a tuneable strategy, but it also provides only a limited control due to the lack of generic, parametric prototypes for the functions in its definition.

2.7 Test Problems for Simulating Constrained Multi-objective Optimisation

The presence of 'hard' constraints in a multi-objective optimisation problem may cause further hurdles. Veldhuizen (1999) has cited a number of constrained test problems used by several researchers. This section provides a description of a number of test problems commonly used in literature.

2.7.1 Non-tuneable Test Problems

Most of the constrained multi-objective optimisation test problems are not tuneable in nature. In most of these, there are only two to three variables, and the constraints are not sufficiently non-linear. A summary of some of these problems is provided in Table 2.8.

Table 2.8: Non-tuneable Test Problems for Constrained Multi-objective Optimisation

Test Problems	Objective Functions (Minimisation)	Constraints	Search Spaces
OSY (Osyczka and Kundu, 1995)	$f_1(\bar{x}) = -[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 + (x_4 - 4)^2 + (x_5 - 1)^2],$ $f_2(\bar{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2.$	$x_1 + x_2 - 2 \geq 0,$ $6 - x_1 - x_2 \geq 0,$ $2 + x_1 - x_2 \geq 0,$ $2 - x_1 + 3x_2 \geq 0,$ $4 - (x_3 - 3)^2 - x_4 \geq 0,$ $(x_3 - 3)^2 + x_6 - 4 \geq 0,$ $0 \leq x_1, x_2, x_6 \leq 10,$ $1 \leq x_3, x_5 \leq 5,$ $0 \leq x_4 \leq 6.$	Figure 2.10(a)
SRN (Srinivas and Deb, 1994)	$f_1(\bar{x}) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2,$ $f_2(\bar{x}) = 9x_1 - (x_2 - 1)^2.$	$x_1^2 + x_2^2 \leq 225;$ $x_1 - 3x_2 + 10 \leq 10,$ $-20 \leq x_1 \leq 20,$ $-20 \leq x_2 \leq 20.$	Figure 2.10(b)
TNK (Tanaka, 1995)	$f_1(\bar{x}) = x_1,$ $f_2(\bar{x}) = x_2.$	$x_1^2 + x_2^2 - 1 -$ $0.1 \cos(16 \arctan(x_1 / x_2)) \geq 0,$ $(x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5,$ $0 \leq x_1 \leq \pi,$ $0 \leq x_2 \leq \pi.$	Figure 2.10(c)

2.7.2 Tuneable Test problems

Recently, Deb *et al.* (2001) presented a tuneable test bed for constrained multi-objective optimisation problems. This test bed is based on the challenges that the constraints pose for multi-objective optimisation problems. It has two sets of generic test functions. In the first set, the constraints are designed in such a way that some portion of the unconstrained Pareto front becomes infeasible. In this way, the final Pareto front is a composite of the unconstrained Pareto front and the constraint boundaries. Equation 2.12 defines this test problem. The parameters (a_j , b_j) control

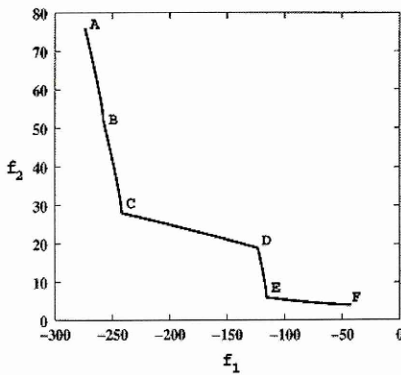
the nature of the final Pareto front. Deb *et al.* (2001) suggest a procedure for calculating these parameters.

$$\text{Minimise} \Rightarrow f_1(\bar{x}_I), \tag{Equation 2.12}$$

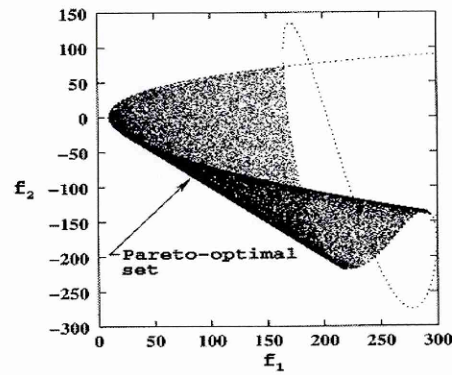
$$\text{Minimise} \Rightarrow f_2(\bar{x}) = g(\bar{x}_{II}) \times \exp(-f_1(\bar{x}_I)/g(\bar{x}_{II})),$$

$$\text{Subject_to} \Rightarrow c_j(\bar{x}) \equiv f_2(\bar{x}) - a_j \exp(-b_j f_1(\bar{x}_I)) \geq 0, j = 1, 2, \dots, J;$$

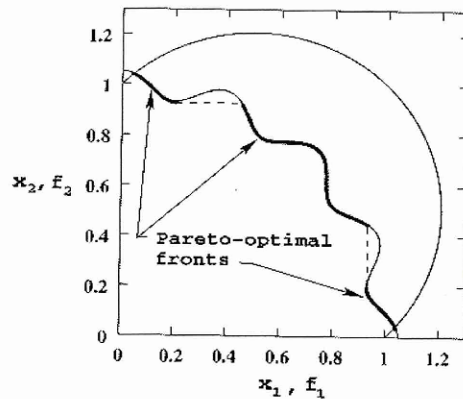
$$\bar{x} = (\bar{x}_I, \bar{x}_{II})^T$$



(a)



(b)



(c)

Figure 2.10: Non-tunable Test Problems for Constrained Multi-objective Optimisation - (a) OSY (b) SRN (c) TNK (Source: Deb, 2001)

In the second set, the whole of unconstrained Pareto front is made infeasible, making the constraints define the resulting Pareto front. This test problem is defined in Equation 2.13. The constraint function has six parameters (θ, a, e, b, c and d), which

respectively control the slope of Pareto front, the transition from continuous to discontinuous feasible regions, and the location, number, distribution and size of disconnected Pareto regions. Here, the decision variable x_1 is restricted to $[0,1]$, and the bounds on other variables depend on the chosen g function.

$$\begin{aligned}
 \text{Minimise} &\Rightarrow f_1(\bar{x}) = x_1, & \text{Equation 2.13} \\
 \text{Minimise} &\Rightarrow f_2(\bar{x}) = g(\bar{x}) \times (1 - (f_1(\bar{x}) / g(\bar{x}))), \\
 \text{Subject_to} &\Rightarrow c(\bar{x}) \equiv \cos(\theta)(f_2(\bar{x}) - e) - \sin(\theta)f_1(\bar{x}) \geq \\
 &a \left| \sin(b\pi(\sin(\theta)(f_2(\bar{x}) - e) + \cos(\theta)f_1(\bar{x}))^c) \right|^d.
 \end{aligned}$$

The two sets of equations, suggested by Deb *et al.* (2001), can together model the difficulties for constrained multi-objective optimisation algorithms both near the Pareto-optimal front and in the entire search space. Deb *et al.* (2001) illustrated this through the development of eight problems (CTP1 to CTP8), constructed from these sets of equations. Table 2.9 lists some of these problems with their parameter values.

This scheme provides a tuneable framework for test bed development. As suggested by Deb *et al.* (2001), it can also be extended to include more than two objectives. However, since it concentrates on the challenges posed by constraints, it does not directly control the complexity of test problems in terms of their objective functions. Therefore, this scheme lacks a unified approach to multi-objective test bed development, and hence suffers from limitations in performing controlled simulation of the features of engineering design optimisation problems.

Table 2.9: Tuneable Test Problems for Constrained Multi-objective Optimisation

Test Problems		Parameter Values	Search Spaces
Difficulty in Vicinity of Pareto-optimal Front	CTP1	$a_1=0.858, b_1=0.541, a_2=0.728, b_2=0.295$	Figure 2.11(a)
	CTP5	$\theta=-0.2\pi, a=0.1, b=10, c=2, d=0.5, e=1$	Figure 2.11(b)
Difficulty in Entire Search Space	CTP6	$\theta=0.1\pi, a=40, b=0.5, c=1, d=2, e=-2$	Figure 2.11(c)
	CTP7	$\theta=-0.05\pi, a=40, b=5, c=1, d=6, e=0$	Figure 2.11(d)

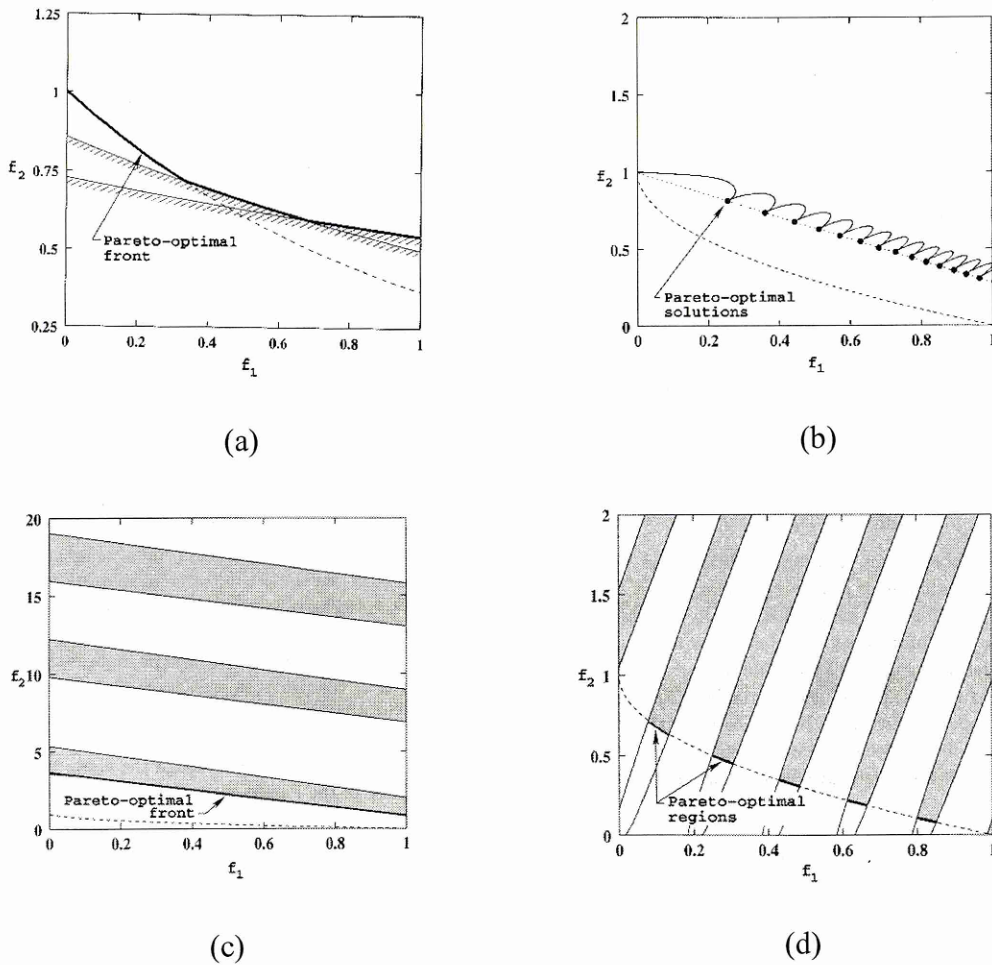


Figure 2.11: Tuneable Test Problems for Constrained Multi-objective Optimisation –
 (a) CTP1 (b) CTP5 (c) CTP6 (d) CTP7 (Source: Deb, 2001)

2.7.3 Summary

Following are the concluding remarks for this section.

- ◆ Most of the test problems for simulating constrained multi-objective optimisation problems are not tuneable in nature.
- ◆ Deb *et al.* (2001) proposed a tuneable strategy, but it focuses only on constraints, without addressing its interactions with the complexity introduced by the objective functions. Therefore, this scheme also lacks a complete approach to multi-objective test bed development.

2.8 Test Problems for Variable Interaction

As mentioned earlier, there are two categories of interaction among decision variables: inseparable function interaction and variable dependence. This section presents a survey of test problems in each of these categories.

2.8.1 Test Problems for Inseparable Function Interaction

Literature reveals a general lack of test problems in which the epistasis/inseparable function interaction can be controlled. This applies to both single- and multi-objective optimisation problems, working in all domains. However, in recent years Deb (1999b) has attempted to introduce control over variable interaction in his test suite for multi-objective optimisation.

Deb (1999b) observed that variable interaction may create difficulty for a GA in converging to the true Pareto front. In Equation 2.10, the Pareto-optimal set corresponds to all solutions of different f_I values. Since the purpose in a multi-objective optimisation is to find as many Pareto-optimal solutions as possible, and since in Equation 2.10 the variables defining $f_I(x_I)$ are different from those defining $g(x_{II})$, a multi-objective optimisation algorithm may work in two stages. In stage one, all variables x_I may be found and in the other stage optimal x_{II} values may be obtained. This rather simple mode of working of a multi-objective optimisation algorithm in two stages can face difficulty if the above variables are mapped to another set of variables. If M is a random orthonormal matrix of size $n \times n$, the true variables \mathbf{y} can first be mapped to derive the variables \mathbf{x} by using the following equation (Equation 2.14).

$$\mathbf{x} = M\mathbf{y} \quad \text{Equation 2.14}$$

Thereafter, the objective functions defined in Equation 2.10 can be computed by using the variable vector \mathbf{x} . Since a multi-objective optimisation algorithm will be operating on the variable vector \mathbf{y} and the function values depend on the interaction among the variables of \mathbf{y} , any change in one variable must be accompanied by related

changes in other variables in order to remain on the Pareto front. This makes the mapped version of the problem difficult to solve.

The major drawback of this strategy is that it takes a narrow perspective of variable interaction. It exploits the structure of Deb's (1999b) test bed to artificially introduce an external variable interaction in the problem. However, it completely ignores the interaction that is already present in the problem in the definition of its objective functions. Therefore, this strategy lacks a holistic view to variable interaction in multi-objective optimisation.

2.8.2 Test Problems for Variable Dependence

Literature reports a complete lack of test problems for simulating variable dependence in multi-objective optimisation problems. However, the vast pool of test functions in optimisation, and the popular equations in other areas of research could be used to derive these problems.

2.8.3 Summary

This section can be concluded with the following remarks.

- ◆ The development of test beds for simulating variable interaction has not been adequately addressed by previous research in the area of optimisation.
- ◆ There is only one test bed (Deb, 1999b) that attempts to address inseparable function interaction. Even this test bed lacks a holistic approach to this concept, and hence does not provide full control over inseparable function interaction in a multi-objective optimisation problem.
- ◆ Literature reports a complete lack of test problems for simulating variable dependence in multi-objective optimisation problems.

2.9 Summary

This chapter has achieved the following.

- ◆ This chapter has presented an overview of the main engineering design optimisation approaches.

- ◆ It has then presented a literature review of the EC techniques in four areas that are based on three features of engineering design optimisation problems.
 - *Evolutionary-based Multi-objective Optimisation Techniques.*
 - *Evolutionary-based Constrained Optimisation Techniques.*
 - *Evolutionary-based Techniques for Handling Inseparable Function Interaction.*
 - *Evolutionary-based Techniques for Handling Variable Dependence.*
- ◆ It has finally presented a review of optimisation test functions in the same four areas, as mentioned above.

As mentioned in Chapter 1, this research attempts to develop EC techniques for dealing with the challenges of engineering design optimisation problems. The current chapter has given an overview of EC techniques for handling three features of these problems: multiple objectives, constraints and variable interaction. This survey of literature enables the identification of the research aim and objectives in the next chapter.

3 RESEARCH AIM, OBJECTIVES AND METHODOLOGY

The research aims to develop EC techniques for handling the complexities of engineering design optimisation problems. This chapter identifies the objectives of this research. As shown below, it attempts to discuss the following.

- ◆ *Research aim.*
- ◆ *Objectives.*
- ◆ *Scope.*
- ◆ *Methodology.*

3.1 Research Aim

The aim of this research is to explore the field of EC for developing techniques that are capable of dealing with the challenges posed by three features of engineering design optimisation problems: multiple objectives, constraints and interaction among decision variables. This would enhance the industrial usefulness of optimisation algorithms by giving them the capability of dealing with a wide variety of real-life problems.

3.2 Research Objectives

There are a number of research issues involved in the fulfilment of the aim of this research. The research objectives, which address these issues, are as follows.

- ◆ To carry out a literature survey for classification and critical analysis of EC techniques for handling three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction.
- ◆ To carry out a literature survey of existing test functions for evaluating their capabilities of performing systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems.

- ◆ To identify the industrial context of the research.
- ◆ To develop EC techniques that can fill the gap between the capabilities of existing optimisation algorithms and the challenges posed by multiple objectives, constraints and variable interaction.
- ◆ To develop test beds that can address the drawbacks of the existing optimisation test functions in mimicking multiple objectives, constraints and variable interaction, in a systematic and controlled manner.
- ◆ To compare the performance of the proposed algorithms with the current state-of-the-art optimisation algorithm (NSGA-II), using the proposed test bed and other popular test functions from literature.
- ◆ To analyse a set of case studies in real-life engineering design optimisation, and to validate the performance of the proposed algorithms using three appropriately chosen case studies from this set.

3.3 Research Scope

Based on the objectives mentioned above, the scope of this research can be summarised as follows.

- ◆ **Domain:** This research focuses only on engineering design optimisation.
- ◆ **Optimisation Techniques:** As mentioned in the previous chapter, this research concentrates on EC techniques due to their flexibility, adaptability, robustness and global search characteristics. Further, within the EC techniques, this research mainly focuses on the GAs because of their broad applicability.
- ◆ **Literature Survey:** The literature survey in this research concentrates on EC techniques that attempt to handle three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction. It should be noted here that objectives and constraints are interchangeable in a number of optimisation problems.
- ◆ **Industry Survey:** Although the industry survey involved a wide range of companies, the focus was on engineering design optimisation problems.
- ◆ **Areas of Development of Optimisation Techniques:** In this research, the EC techniques are developed for handling three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction.

- ◆ Areas of Development of Test Bed: This research focuses on the development of test beds for performing systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems.
- ◆ Validation: In this research, the validation is performed using newly developed test beds and case studies borrowed from literature in the area of real-life engineering design optimisation. These case studies are analysed, and three of them are selected in such a way that a broad spectrum of features is attained.

3.4 Research Methodology

This section discusses the methodology that has guided the main activities of this research. A pictorial representation of this methodology is given in Figure 1.6, which also forms the basis for the layout of this thesis.

3.4.1 Problem Identification

As mentioned in Chapter 1, this research forms a part of the project ‘FLEXO’ (Roy, *et al.*, 2000a). The problem statement for this research is, therefore, derived based on the objectives of ‘FLEXO’. Hence, it shares the vision of ‘FLEXO’, which is to make optimisation algorithms more popular in industry through the removal of hurdles in their industrial use.

3.4.2 Literature Survey

An extensive literature survey is carried out as part of this research in order to analyse and classify the state-of-the-art evolutionary-based optimisation techniques, and the test beds for evaluating these techniques. Since the focus in this research is engineering design optimisation, the literature survey is carried out with respect to three features, as identified by literature, of engineering design optimisation problems: multiple objectives, constraints and variable interaction. This enables to attain a broad understanding of the existing work in terms of its strengths and weaknesses.

3.4.3 Identification of Research Aim, Objectives and Focus

The survey of literature highlights the main research issues that need to be addressed for handling the problem statement of this research. This enables the precise identification of the research aim and objectives that can address these issues. The literature survey also enables the identification of the drawbacks of the EC techniques in handling three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction. A similar procedure is adopted to identify the limitations of existing optimisation test functions in performing systematic and controlled modelling of multiple objectives, constraints and variable interaction. These limitations in existing EC techniques and test functions define the focus of this research.

3.4.4 Industry Survey

The aim of this industry survey is to support the literature survey for grounding the research within the industrial context (Roy *et al.*, 2000c; Roy *et al.*, 2000d). In order to attain a broad perspective of design optimisation in industry, companies belonging to a number of industry sectors are surveyed. However, only the engineering design optimisation activities in these companies are observed. This survey is carried out through industry visits, and uses semi-structured questionnaires for collecting information from the designers. The detailed survey methodology is discussed in the next chapter.

3.4.5 Development of EC Techniques

Two new EC techniques are developed in this research to address the drawbacks of existing ones in handling multiple objectives, constraints and variable interaction. This development is carried out in a systematic, step-by-step fashion, adding a single new feature at a time. These features are those that are not adequately addressed by the existing techniques. Here, the current state-of-the-art technique is used as the starting point of development. In this way, all the strengths of current research are inherited, while addressing its weaknesses. At the end of this development, the

performance of the proposed algorithms is compared, using some popular test functions, with the state-of-the-art optimisation technique in its original form.

3.4.6 Development of Test Bed

In this research, a test bed is developed to enable systematic and controlled simulation of three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction. Similar to the previous case, the development of this test bed is also guided, in a step-by-step fashion, by the drawbacks of the existing ones in mimicking the above-mentioned features of engineering design optimisation problems. Also, the philosophy of the existing ‘tuneable’ test beds is used here to develop the proposed test bed. Furthermore, the performance of the proposed test bed is validated by applying it to construct multiple test functions, modelling a number of features of engineering design optimisation problems with varying degrees of complexity.

3.4.7 Performance Analysis Using the Proposed Test Bed

Here, the performance of the proposed algorithms is compared with the state-of-the-art optimisation technique. This comparison is carried out using a wide spectrum of test problems created from the proposed test bed. These test problems are developed such that they evaluate the performance of the optimisation algorithms in the presence of a number of features that are commonly present in engineering design optimisation problems.

3.4.8 Validation Using Real-life Case Studies

Here, a set of real-life engineering design optimisation problems, reported in literature, are analysed from the point of view of the challenges that they pose for optimisation algorithms. The performance of the proposed optimisation algorithms is validated using three appropriately chosen case studies from this set. In this way, this research proposes a fully tested and validated methodology for dealing with engineering design optimisation problems.

3.4.9 Identification of Limitations and Future Research Directions

Finally, the limitations of the research methodology, and proposed optimisation algorithms and test bed are identified. Based on these limitations, the generality of the research and its contribution to knowledge are established, and the corresponding future research directions are proposed.

3.5 Summary

This chapter has discussed the following.

- ◆ It has stated the research aim.
- ◆ It has outlined the objectives that address the aim of this research.
- ◆ It has summarised the scope of this research based on its objectives.
- ◆ This chapter has finally discussed the methodology that has guided this research. This methodology has seven main parts, as given below.
 - *Problem identification.*
 - *Literature survey.*
 - *Identification of research aim and objectives.*
 - *Identification of industrial context and focus of the research.*
 - *Development.*
 - *Testing and Validation.*
 - *Identification of limitations and future research directions.*

As stated in this chapter, the aim of this research is to develop EC techniques that are capable of dealing with the challenges posed by three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction. The next chapter reports the findings of an industry survey to enable the grounding of this research within the industrial context. It also determines the focus of this research by analysing the gap between the capabilities of existing EC techniques and the challenges posed by multiple objectives, constraints and variable interaction.

4 INDUSTRIAL CONTEXT AND FOCUS

Chapter 2 analysed the EC techniques for handling multiple objectives, constraints and variable interaction. It also presented a survey of test beds for controlled simulation of these features of engineering design optimisation problems. This chapter grounds the research within the industrial context based on a survey of companies, coupled with a study of existing literature in the area of real-life optimisation. This chapter also analyses the observations made in Chapter 2 to determine the focus of this research. This analysis guides the course of action that is followed in this research. The objectives of this chapter can be summarised as follows.

- ◆ *To explain the methodology of industrial survey.*
- ◆ *To identify the inhibitors to industrial applications of optimisation algorithms.*
- ◆ *To ground the research within the industrial context.*
- ◆ *To identify the gap between the capabilities of existing EC techniques and test beds with respect to the challenges posed by multiple objectives, constraints and variable interaction in engineering design optimisation problems.*
- ◆ *To determine the focus of the research.*

4.1 Industrial Survey

An industry survey is carried out for grounding the research within the industrial context. This is complemented by a survey of literature in the area of real-life optimisation, which compiles those real-world applications of evolutionary-based optimisation techniques that are reported in literature. As mentioned in Chapter 3, a representative set of these problems is chosen in this research to validate the performance of the proposed optimisation algorithms. This section concentrates on the methodology that is adopted here for carrying out the industry survey.

4.1.1 Aim and Objectives

The main aim of the industry survey is to ground the research within the industrial context. The survey objectives that lead to this aim can be summarised as follows (Roy *et al.*, 2000c).

- ◆ To assess the current status of engineering design optimisation in industry.
- ◆ To identify the features of real-life engineering design optimisation problems.
- ◆ To identify the factors that inhibit the industrial applications of optimisation algorithms.

4.1.2 Methodology

In the industry survey, the designers belonging to various industry sectors were interviewed. However, only the engineering design optimisation activities in these companies were observed. This survey was carried out through industry visits, and used a semi-structured questionnaire as a tool for collecting information. Appendix A presents a copy of this 'FLEXO' questionnaire. The following companies were visited by the 'FLEXO' researchers (Roy *et al.*, 2000).

- ◆ Nissan Technical Centre – Europe (NTC-E).
- ◆ Corus – British Steel.
- ◆ Ikeda Hoover Ltd. (IHL).
- ◆ TRW Automotive.
- ◆ Trelleborg Automotive.
- ◆ Xerox Limited Technical Centre (XETC).

The questionnaire was developed based on the recommendations of Oppenheim (1992). The following factors were considered.

4.1.2.1 Method of Approach to Respondents

Prior to the visit, some information regarding the research is sent to the main contact person in the company. Further, the interviews are preceded by a presentation, which introduces the research and explains the purpose of the visit. The objectives of the

questionnaire and of each of its modules are also provided in the questionnaire. After guaranteeing the confidentiality, the designers are individually interviewed by one or two researchers. The researchers write and tape (subject to the approval of the respondents) the responses given during the interview. The information thus collected from the interview is used for preparing the final transcript. The questions are set to fit the time frame available in the industrial environment. The development of the questionnaire also addresses the issue of biases by broadening the aspects covered by the questions.

4.1.2.2 Determination of Question Sequence

The questionnaire is structured in four modules covering general design issues, industrial requirements for optimisation algorithms, general remarks and finally the self-assessment of the designer (Appendix A). The first module of the questionnaire is aimed at understanding the general design optimisation practice in industry and the degree of involvement of designers in those activities. The second module targets at capturing the industrial requirements for optimisation algorithms. This is achieved by identifying the features of real-life optimisation problems (Section 4.3.1) and the inhibitors to the industrial applications of optimisation algorithms (Section 4.4.1) that highlight the corresponding limitations of the existing design systems. The third and fourth modules respectively deal with the general comments and the self-assessment of the interviewed designers (Roy *et al.*, 2000d). These modules, which are kept optional, aim at gathering some information about the respondents so that their comments can be evaluated in the right perspective. Each module starts with some broad questions, which are gradually made specific. The process is called funnelling, which is a standard practice in similar applications (Roy, 1997). The broad questions at the beginning of a module prepare the ground for subsequent questions.

4.1.2.3 Types of Questions

The questionnaire uses both ‘closed’ or pre-coded answer and ‘open’ or free-response types of questions. A ‘closed’ question is one in which the respondents are offered a choice of alternative replies. Although this type of questions allow less

freedom of expression, but they are easy to answer and analyse. On the other hand, ‘open’ or free-response type questions are not followed by any kind of choice, and the answers have to be written in full. These questions allow freedom of expression and are easy to ask, but are difficult to answer and analyse. They also involve personal emotions and biases (Roy, 1997). The above discussion justifies the use of a combination of both ‘closed’ and ‘open’ types of questions in the ‘FLEXO’ questionnaire.

4.1.2.4 Analysis of Responses

For each of the questions in the questionnaire, the responses given by the respondents are compiled into two categories. The first category, known as ‘common’, identifies the observations made by a majority of respondents. On the other hand, the second category, known as ‘special’, identifies the observations made only by a few respondents. Unless stated otherwise, all the observations listed in the subsequent sections of this chapter are derived from the ‘common’ category of the analysis of responses.

4.2 Design Improvement in Industry

After investigating the design processes in different companies using the industry survey, it is observed that they exhibit a number of similarities. It is observed that the design optimisation in industry is an iterative process of creating a model, using a design system, carrying out some analyses which give indications of how to improve the design, and then modifying the model and repeating the process. This manual process contributes significantly to lengthening the design cycle and depends critically for its success on the skill of the designer. It is observed that trial-and-error finds widespread use in industry for improving designs.

4.2.1 Industrial Applications of Optimisation Algorithms

Despite the immense potential of optimisation algorithms (Table 4.1), it is observed that no company surveyed uses any of these algorithms actively as a day-to-day

design tool. However, the design centres of some multi-national corporations use optimisation packages, like OPTISTRUCT from Altair Engineering and DOT from Vanderplaats Research and Development, Inc. Companies using these software systems report benefits in terms of reduction of design lead times and attainment of better designs through the use of these packages.

Table 4.1: Evolutionary-based Optimisation Algorithms in Applied Research

Product Families	Applications
Aircraft Body Shells	<ul style="list-style-type: none"> • Aerodynamic shape design (Poloni, 1997) • Aircraft wing platform design (Obayashi <i>et al.</i>, 1998) • Airframe design (Cvetkovic and Parmee, 1998) • Airfoil design (Poloni and Pediroda, 1997 and Quagliarella and Vicini, 1997)
Civil Engineering Structures	<ul style="list-style-type: none"> • Design of frames, foundations, bridges, towers, chimneys and dams (Rao, 1996) • Structure design for random loading (earthquake, wind) (Rao, 1996) • Design of water resource systems (Rao, 1996) • Design of plane trusses (Liu <i>et al.</i>, 1998 and Hajela and Lin, 1992) • Design of I-beams (Coello, 1997; Coello and Christiansen, 1999) • Design of two-bar truss (Deb, Pratap and Moitra, 2000)
Mechanical Components	<ul style="list-style-type: none"> • Design of linkages, cams, gears and machine tools (Rao, 1996) • Extruder screw design (Cunha <i>et al.</i>, 1997; Cunha, 2000) • Design of planar mechanisms (Sandgren, 1994) • Design of robot arm (Coello, 1997; Coello <i>et al.</i>, 1998) • Design of machines (Osyczka, 1984) • Design of machine tool spindle (Coello, 1997) • Design of compound gear train (Deb, Pratap and Moitra, 2000) • Welded beam design (Deb, Pratap and Moitra, 2000)
Networks	<ul style="list-style-type: none"> • Design of electrical networks (Rao, 1996) • Design of pipeline networks (Rao, 1996)
High Performance Materials	<ul style="list-style-type: none"> • Microwave absorber design (Weile <i>et al.</i>, 1996) • Laminated ceramic composites (Belegundu <i>et al.</i>, 1994)
Energy Conversion Machines	<ul style="list-style-type: none"> • Pumps, turbines and heat transfer equipment (Fonseca and Fleming, 1998b; Chipperfield and Fleming, 1995) • Design of a turbine blade cooling system (Roy, 1997) • Motors, generators and transformers (Rao, 1996) • Reactor design (Mitra <i>et al.</i>, 1998) • Compressor design (Obayashi, 1997)
Electronic Components	<ul style="list-style-type: none"> • Microprocessor chip design (Stanley and Mudge, 1995) • Synthesis of multiprocessor system (Zitzler and Thiele, 1998b) • Design of control systems (Tan and Li, 1997) • Multiplierless filters (Wilson and Macleod, 1993)
Miscellaneous Applications	<ul style="list-style-type: none"> • Design of conveyors, trucks and cranes (Rao, 1996) • Determination of optimal machining parameters (Coello, 1997) • Design of helical compression spring (Deb, Pratap and Moitra, 2000)

4.2.2 Optimisation Algorithms in Applied Research

Interestingly, the popularity of optimisation algorithms in applied research is much more than that in industry. The reasons for this difference could be understood from Section 4.4.1 that discusses the inhibitors to the industrial applications of optimisation algorithms. Literature reveals a number of real-life applications of optimisation algorithms, especially in the area of evolutionary computing. Table 4.1

lists some of these applications encountered across various industry sectors. Some of the applications listed in this table are analysed in detail in Chapter 9.

4.3 Features of Real-life Engineering Design Optimisation Problems

The real-life optimisation problems, as opposed to the theoretical problems (test cases), are those that are encountered in industry. This section compiles the features of real-life engineering design optimisation problems using the results from the industry and literature survey. It also gives a simple example of these problems.

4.3.1 Features

The main features of real-life engineering design optimisation problems are listed below.

- ◆ The principal feature of most real-life problems is the presence of multiple measures of performance, or objectives, which should be improved simultaneously.
- ◆ Almost all these problems require some constraints to be satisfied.
- ◆ The complexity of a number of these problems is further increased by the presence of multiple interacting decision variables, involving both inseparable function interaction and variable dependence. In many cases, the variables also take both integer and real values.
- ◆ Most real-life problems also involve qualitative issues (such as manufacturability and designers' special preferences), and elements of incompleteness, inaccuracy and uncertainty in their models.
- ◆ The models of a number of real-life engineering design optimisation problems are computationally expensive in nature. Examples include those models that involve the use of FEA or CFD.
- ◆ A number of real-life engineering design optimisation problems also use multiple models for different physical aspects.

- ◆ The computational expense for solving these problems and their difficulty are also augmented by the presence of several optimal solutions, as defined in Chapter 2.
- ◆ Lack of prior knowledge regarding the shape of search space is also commonly observed in these problems. There is also no prior information about the performance, and the location of optimal and sub-optimal points in the search space.
- ◆ Finally, the model development for the solution of real-life engineering design optimisation problems is a very complex task. It involves determining the variables in the problem, and the objective functions and constraints in such a way that the model thus obtained closely matches the features of the real-life problem.

Table 4.2: Features of Real-life Engineering Design Optimisation Problems

Classification Schemes	Features
Based on Number of Parameters	Multi-dimensional
Based on Existence of Constraints	Constrained
Based on Number of Objective Functions	Multi-objective
Based on Nature of Objective Functions	Hybrid
Based on Separability of Functions (for Quantitative and Hybrid Problems)	Inseparable
Based on Dependence among Variables	Independent- and Dependent-variable
Based on Nature of Search Space	Unknown Search Space
	Multi-modal
Based on Nature of Equations Involved (for quantitative and hybrid problems)	Linear, Non-linear, Geometric and Quadratic
Based on Nature of Design Variables	Static and Dynamic
Based on Permissible Values of Design Variables	Hybrid

Chapter 2 discussed the different classification schemes used for optimisation problems. These classification schemes are used here to summarise the features of real-life engineering design optimisation problems, as shown in Table 4.2.

4.3.2 An Example of a Real-life Design Optimisation Problem

Figure 1.1, which depicts the model constructed from the real-life scenario, can be considered as an example of a real-life engineering design optimisation problem. This problem involves the optimisation of the design of a rectangular cantilever beam, for given material and loading conditions. In a typical real-life case, this problem may possess the following features.

- ◆ Multiple variable, such as l , b and h .

- ◆ Constraints, such as variable bounds and relationships involving beam dimensions.
- ◆ Multiple objectives, such as minimisation of the end deflection, maximum stress along the beam length and cost involved.
- ◆ Qualitative issues, such as manufacturability and designers' special preferences.
- ◆ Inseparable function interaction leading to an inseparable optimisation problem, such as minimisation of the end deflection of the beam (Equation 1.1).
- ◆ Dependence among decision variables, such as the designer preference to have a fixed cross-section aspect ratio (Equation 1.3).
- ◆ Lack of prior knowledge about the problem in terms of its search space.
- ◆ Non-linear objective function(s), such as minimisation of the end deflection of the beam (Equation 1.1).
- ◆ Dynamic variables.
- ◆ Combination of integer and real variables, such as l (real variable) and material type (discrete/integer variable).
- ◆ Complexity of model development in terms of the cross-section, support and loading.

4.4 Industrial Context of the Research

The industry and literature surveys also enable the identification of the factors that inhibit the applications of optimisation algorithms in industry. These inhibitors are presented here and compared against the research objectives with an aim of grounding the research within the industrial context.

4.4.1 Inhibitors to Industrial Applications of Optimisation Algorithms

The main inhibitors to the industrial applications of optimisation algorithms are outlined below.

- ◆ The features of real-life engineering design optimisation problems, such as the presence of multiple objectives, constraints and interaction among decision

variables, create challenges for optimisation algorithms that are currently in use in industry. This discourages the industry from adopting these algorithms. This is particularly true for those industries that deal with a wide range of complex designs.

- ◆ All optimisation algorithms work on mathematical models of real-life designs. It is observed that since designers prefer maintaining full control on the design improvement process, they have little faith in the models that are provided to them. This makes them sceptical about the results obtained from the optimisation algorithms. This situation is further worsened by the fact that there is a lack of model development skills among designers in industry. There is also a lack of commercial tools required for carrying out the task of model development.
- ◆ Most of the currently available optimisation packages are not integrated within CAD/CAM systems, making their use cumbersome. The designers need to extract the parameters from the CAD/CAM models, feed them to the optimisation packages and bring the optimised parameters back to the CAD system. There are a number of difficulties, associated with this off-line optimisation, which prevent the designers from using the optimisation algorithms. The data transfer often leads to loss of quality and information, which makes the optimisation process inaccurate. This off-line scenario of optimisation also makes designers lose control over the design process. Finally, the inflexible nature of this scenario makes the process iterative and time consuming.
- ◆ Another inhibitor to the use of optimisation algorithms in industry is the important role of designers' skills and experience in the design improvement process. This makes the optimisation task extremely difficult to be modelled and encoded in an algorithmic form. Further, the lack of knowledge of designers in using these algorithms also presents an additional obstacle to their use in industry.
- ◆ Each company has its own design improvement process. This process gradually evolves in the company, and hence its people resist the implementation of any new optimisation system and the associated organisational changes. Further, the costs associated with creation, installation and maintenance of optimisation algorithms discourage their use in industry.

4.4.2 Industrial Context of the Research

As mentioned in Chapter 1, the project ‘FLEXO’ targets the first three of the above-mentioned inhibitors. This research focuses on the first of these inhibitors, which is the lack of robust optimisers in industry. It attempts to develop optimisation techniques that can handle, within a single framework, the following three features of real-life engineering design optimisation problems: presence of multiple objectives, constraints and variable interaction. Since it is difficult to find a variety of real-life cases with required complexities, this research also develops test beds that are capable of performing systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems.

The existing EC techniques and test beds in these three areas were critically analysed in Chapter 2. The next two sections utilise the main findings of Chapter 2 in order to identify the research gap that forms the focus of this research.

4.5 Research Focus for Development of EC Techniques

This section compares the capabilities of the existing EC techniques against the challenges posed by multiple objectives, constraints and variable interaction in engineering design optimisation problems. In more specific terms, it checks whether the existing EC techniques can handle the above-mentioned features of real-life engineering design optimisation problems.

4.5.1 Multi-objective Optimisation

Chapter 2 reveals that the area of multi-objective optimisation is well addressed within the EC community. This has led to the development of techniques that can effectively handle this feature of real-life problems. It is now known that a combination of elitism, Pareto domination and diversity preservation lead to EC techniques that can, in principle, meet both the goals of multi-objective optimisation: convergence to the Pareto front and maintenance of diversity across the front. In the

recent past, some techniques, such as DPGA (Osyczka and Kundu, 1995), TDGA (Kita *et al.*, 1996), SPEA (Zitzler and Thiele, 1998a), MOMGA (Veldhuizen, 1999), NSGA-II (Deb *et al.*, 2000) and PAES (Knowles and Corne, 2000), have been developed that incorporate the above-mentioned concepts in a single algorithm. Deb (2001) demonstrated the superiority of these techniques over others. It has also been shown that these techniques can deal with a variety of multi-objective optimisation problems, but fail to give satisfactory results in the presence of complex inseparable function interaction among decision variables (Deb *et al.*, 2000).

4.5.2 Constrained Multi-objective Optimisation

Similarly, it is evident from Chapter 2 that the area of constrained optimisation is well researched. However, most of this research is limited to single-objective optimisation, and the field of constrained multi-objective optimisation has grown only recently. In spite of this, a powerful strategy for handling constraints in multi-objective optimisation problems has now been developed. This strategy incorporates constraint violations in the definition of Pareto domination, and uses niching to encourage diversity among solutions. Techniques, such as the Constrained Domination Method (Deb, 2000), that use this strategy achieve better convergence and diversity of solutions as compared to other approaches. These techniques have been shown to satisfactorily handle constraints in a variety of multi-objective optimisation problems (Deb, 2001).

4.5.3 Variable Interaction

Interaction among decision variables is inherent to a number of real-life engineering design optimisation problems. In spite of its immense potential for real-life problems, there is a lack of systematic research in both halves of this field: inseparable function interaction and variable dependence. The gaps created by this lack of research are identified below for the two types of variable interaction.

4.5.3.1 Inseparable Function Interaction

A number of research questions remain unanswered regarding the theory of inseparable function interaction (epistasis), which is a type of variable interaction. However, from the practical point of view some applications have been developed that can deal with this type of interaction in single-objective optimisation problems, defined in discrete domains. But there are only a few of these techniques that can work in real search spaces (Gallagher *et al.*, 1999). Furthermore, inseparable function interaction in multi-objective optimisation problems is a research area that is at its very early stage of development (Thierens and Bosman, 2001). Therefore, given the importance of handling inseparable variable interaction for solving real-life engineering design optimisation problems, a systematic research effort is urgently required to address this interaction in hybrid-valued (with integer and real variables), constrained, multi-objective optimisation problems.

4.5.3.2 Variable Dependence

Most real-life engineering design optimisation problems that have interaction among decision variables do not have known dependency equations. Therefore, the dependency relationships in these problems need to be inferred from the multiple sets of measured variable values that are available in most real-life cases. This implies that any strategy for solving these optimisation problems should provide a framework that is capable of satisfying the following two objectives.

- ◆ Determination of the relationships among decision variables.
- ◆ Incorporation of these relationships in the optimisation engine.

The lack of systematic research in the area of variable dependence has led to a scarcity of dedicated frameworks that can deal with the above-mentioned objectives for solving dependent-variable optimisation problems. This highlights the need to develop a complete EC framework for dealing with variable dependence in constrained multi-objective optimisation problems. However, as mentioned in Chapter 2, some techniques (RA, NNs, PM, TDs and DA), extracted from related areas of research, can aid the development of this framework.

4.5.4 Research Focus

The above discussion reveals that there are effective techniques available in the literature for handling multiple objectives and constraints. However, there is a research gap in EC techniques for handling variable interaction.

- ◆ There is a research vacuum in the area of inseparable function interaction, especially for multi-objective optimisation problems in hybrid search spaces (with integer and real variables).
- ◆ There is also a need to develop dedicated optimisation techniques that can handle variable dependence in multi-objective optimisation problems.

This gap defines the main focus of this research, which is to develop EC techniques that can effectively handle the two types of variable interaction in constrained multi-objective optimisation problems, defined in hybrid search spaces (with integer and real variables).

4.6 Research Focus for Development of Optimisation Test Beds

This section compares the capabilities of the existing test beds for controlled simulation of the following three features of real-life engineering design optimisation problems: multiple objective, constraints and variable interaction.

4.6.1 Multi-objective Optimisation

Most of the multi-objective optimisation test problems are not tuneable in nature. However, Deb (1999b) provides a tuneable strategy that does not incorporate constraints. Furthermore, due to a lack of generic, parametric prototypes for the functions in its definition, this strategy provides only a limited control over the complexity of the test problems.

4.6.2 Constrained Multi-objective Optimisation

Similar to the previous case, most of the test problems in the area of constrained multi-objective optimisation are not tuneable in nature. Deb *et al.* (2001) recently proposed a tuneable strategy that incorporates parametric constraints in the test bed of Deb (1999b). However, this strategy also lacks a complete approach to multi-objective ‘test bed’ development since it focuses only on constraints, without addressing its interactions with the complexity introduced by the objective functions.

4.6.3 Variable Interaction

As mentioned below, the lack of systematic research in the area of variable interaction has also led to a deficiency of test problems that can mimic the two types of variable interaction in a controlled fashion.

4.6.3.1 Inseparable Function Interaction

Most of the current multi-objective optimisation test beds do not explicitly address the complexity introduced in the problem by the inseparable function interaction. Deb (1999b) attempts to introduce variable interaction in his test bed through re-definition of the original variables. However, by completely ignoring the interaction that is already present in the problem definition of its objective functions, it does not provide full control over inseparable function interaction in a multi-objective optimisation problem.

4.6.3.2 Variable Dependence

There is a complete lack of test problems in literature that can simulate dependent-variable multi-objective optimisation problems. However, popular equations from other areas of research could be potentially used to construct these test problems.

4.6.4 Research Focus

Similar to the case of EC techniques, the areas of multi-objective and constrained optimisation test bed development are well addressed in literature as almost separate

streams. However, as revealed in the above discussion, there are a number of gaps in this area that this research attempts to address.

- ◆ There is a need to develop tuneable test beds that can simulate the complexity introduced by both the objective functions and constraints in a single framework.
- ◆ In order to provide full control, it is also required to develop parametric prototypes for the functions in this test bed.
- ◆ Furthermore, it is required to introduce functions/parameters in the test bed that can explicitly control the complexity introduced by the two types of variable interaction.

4.7 Summary

This chapter has achieved the following.

- ◆ It has explained the methodology of the industry survey that is carried out for grounding the research within the industrial context.
- ◆ It has described the current status of engineering design optimisation in industry. This discussion has highlighted that although a number of real-life applications of optimisation algorithms are reported in literature, they attract only a little interest in industry.
- ◆ It has compiled the features of real-life engineering design optimisation problems.
- ◆ It has identified the inhibitors to the industrial applications of optimisation algorithms, and has compared them against the research objectives with an aim of grounding the research within the industrial context. It has stated the industrial context of this research, which is to develop robust optimisers for handling the following three features of real-life engineering design optimisation problems: multiple objectives, constraints and variable interaction.
- ◆ It has analysed the gap between the capabilities of existing EC techniques and the challenges posed by multiple objectives, constraints and variable interaction. This has identified variable interaction as the main area of focus in this research for the development of optimisation algorithms.
- ◆ Finally, it has analysed the capabilities of existing test beds with respect to the following three features of real-life optimisation problems: multiple objectives,

constraints and variable interaction. Here also, variable interaction constitutes as the main focus of test bed development.

This chapter has identified the interaction among decision variables as the main focus of this research. The next chapter develops an algorithm for handling the first type of variable interaction, viz. inseparable function interaction, in complex multi-objective optimisation problems.

5 DEVELOPING AN EC TECHNIQUE TO HANDLE INSEPARABLE FUNCTION INTERACTION

As explained in Chapter 2, inseparable function interaction is a type of variable interaction, and occurs when the effect that a variable has on the objective function depends on the values of other variables in the function. This type of variable interaction is commonly evident in real-life optimisation problems. Chapter 4 identified the gap in the literature to deal with these problems. The aim of this chapter is to develop a generic solution strategy and to propose an algorithm capable of handling complex multi-objective optimisation problems having high degrees of inseparable function interaction. This chapter attempts to achieve the following.

- ◆ *To identify the challenges that inseparable function interaction poses for multi-objective optimisation algorithms.*
- ◆ *To devise a generic solution strategy for handling this interaction in multi-objective optimisation problems.*
- ◆ *To propose a multi-objective optimisation algorithm based on the solution strategy.*
- ◆ *To analyse the performance of the proposed algorithm using existing test problems.*

In addition, the proposed optimisation algorithm is expected to possess the following features.

- ◆ *Generic, so that it is not specific to a problem.*
- ◆ *Modular, so that it can be used to enhance the interaction handling capability of any multi-objective optimisation algorithm.*
- ◆ *Better performance than the current state-of-the-art techniques on a wide variety of multi-objective optimisation problems.*
 - *Better convergence to Pareto front.*
 - *Better distribution of Pareto-optimal solutions.*

5.1 Challenges for Multi-objective Optimisation Algorithms

Complex inseparable function interaction poses a number of challenges for multi-objective optimisation algorithms. A GA operates on the building blocks, growing them and mixing them with each other in an attempt to solve the search problem at hand. Epistasis, termed here as inseparable function interaction, causes problems for a GA by making it more difficult for it to build these building blocks (Harik, 1997). Further, in its presence, a multi-objective optimisation problem cannot be decomposed into simpler parts. Hence, a GA requires updating all decision variables in a unique way in order to maintain a spread of solutions over the Pareto-optimal region or even converge to any particular solution. With a generic search operator, this becomes a difficult task for the GA. Furthermore, even if a set of Pareto-optimal solutions are obtained, it is difficult to maintain them since any change in one variable must be accompanied by related changes in others in order to remain on the Pareto front. The difficulties that inseparable function interaction may create for a GA are summarised below, with respect to the two goals of multi-objective optimisation (Deb, 1999a; Deb, 1999b).

5.1.1 Convergence to Global Pareto Front

Inseparable function interaction in objective functions may lead to one or more of the following features that obstruct convergence to the true (or global) Pareto front.

- ◆ **Multi-modality:** In this case, a GA, like many other search and optimisation methods, may converge to a local Pareto front.
- ◆ **Deception:** Deception is a kind of multi-modality in which almost the entire search space favours the deceptive (non-global) optimum. If present in a problem, deception misleads a GA towards deceptive attractors (Goldberg *et al.*, 1989).
- ◆ **Collateral Noise:** Complex inseparable function interaction in objective functions may lead to problems that are ‘rugged’ with relatively large variations in the

function landscape. This collateral noise may create convergence problems for a GA.

- ◆ **Isolated Optimum:** In some problems, the optimum may be surrounded by a fairly flat search space. Since there is no useful information provided by most of the search space, a GA faces difficulty in solving such problems with isolated optima.

5.1.2 Maintenance of Diverse Pareto-optimal Solutions

Maintenance of diversity in Pareto-optimal solutions may become difficult for a GA due to one or more of the following features that may be introduced in the problem by inseparable function interaction.

- ◆ **Discontinuity in Pareto Front:** Here the Pareto front is a collection of discretely spaced continuous sub-regions (Schaffer, 1984). In such problems, although solutions within each sub-region may be found, competition among them may lead to extinction of some sub-regions.
- ◆ **Non-uniform Distribution over Pareto Front:** In this case, feasible solutions have a non-uniform density across the Pareto front. This leads to a natural tendency for a GA to find a biased distribution in the Pareto-optimal region.
- ◆ **Shape Complexity of Pareto Front:** Inseparable function interaction also influences the shape of Pareto front. In some cases, the shape complexity of the front may be so high that it becomes difficult for a GA to find uniformly distributed solutions across it.

Further, in many real-life multi-objective optimisation problems, inseparable function interaction leads to Pareto fronts that correspond to complex relationships among decision variables. All such cases become difficult for a GA to handle since it is required to update the decision variables in a unique way in order to attain the desired results (Tiwari *et al.*, 2001a). An example of a complex multi-objective optimisation problem that has high degrees of inseparable function interaction is given in Figure 5.1. This problem is given in Equation 5.1. This problem has a number of features discussed above including multi-modality and biased search space.

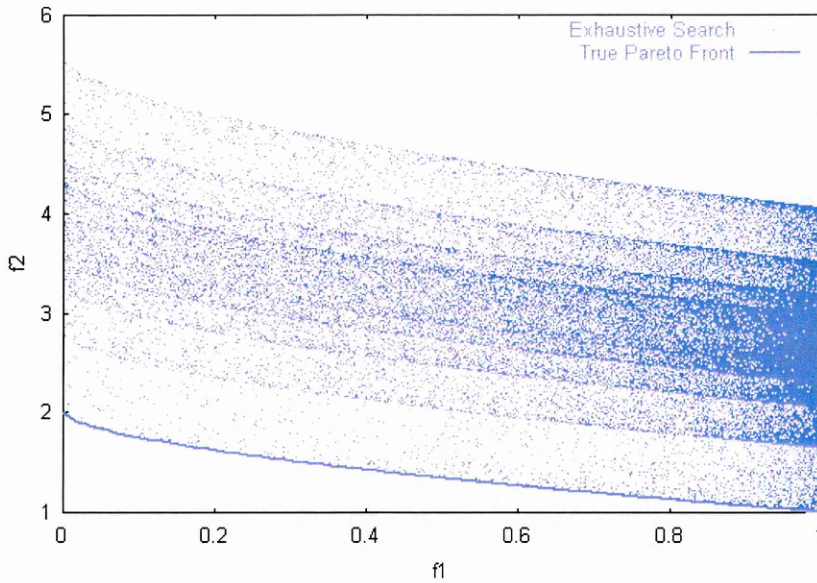


Figure 5.1: An Example of Inseparable Function Interaction

$$f_1(x_1, x_2) = \frac{1}{(1 - \exp(-4))} [1 - \exp(-4x_1)], \forall 0 \leq x_1, x_2 \leq 1, \quad \text{Equation 5.1}$$

$$f_2(x_1, x_2) = (2 - (f_1 / I)^{0.6}) \times (I), \forall 0 \leq x_1, x_2 \leq 1,$$

$$I(x_1, x_2) = 2 - \exp(-2x_2) \cos(8\pi x_2), \forall 0 \leq x_1, x_2 \leq 1.$$

5.2 Proposed Solution Strategy

For any continuous portion of the Pareto front, there is a unique relationship involving objective functions. This relationship is difficult to obtain analytically, and even if it is found, it has limited usefulness since mapping from function space to variable space is very complex. However, the existence of a relationship among objective functions of Pareto solutions necessarily implies that corresponding relationship(s) exist among the decision variables of these solutions.

A simple multi-objective optimisation problem is used below for explaining the above concept (Figure 5.2). Consider a two-objective optimisation problem having f_1 and f_2 as the two objective functions. For any continuous portion of the Pareto front, there exists a Function F involving f_1 and f_2 (Equation 5.2).

$$F(f_1, f_2) = 0$$

Equation 5.2

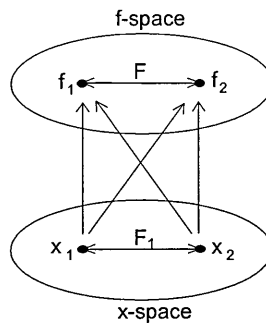


Figure 5.2: Proposed Solution Strategy

Suppose the problem has two decision variables x_1 and x_2 that define the functions f_1 and f_2 i.e. f_1 and f_2 can be expressed as $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$. Substituting the expressions for f_1 and f_2 in the above equation yields the function F_1 in decision variables (Equation 5.3).

$$F(f_1(x_1, x_2), f_2(x_1, x_2)) = 0$$

$$\Rightarrow F_1(x_1, x_2) = 0$$

Equation 5.3

This proves the statement made earlier that there is existence of relationship(s) among the decision variables of the solutions belonging to any continuous portion of the Pareto front. The proposed algorithm aims to explore this relationship using non-linear, multi-variable regression analysis (Draper and Smith, 1998). It uses the relationship thus obtained for the following purposes.

- ◆ To perform periodic re-distribution of solutions for aiding their spread over the current front.
- ◆ To use history of change of regression coefficients for guiding the search towards the global Pareto front.
- ◆ To use rate of change of regression coefficients for determining the termination condition of the algorithm.
- ◆ To re-distribute the final solutions for obtaining the whole range of well-distributed Pareto-optimal solutions.

- ◆ To identify the relationship(s) among the decision variables of the Pareto-optimal solutions, in order to enable the designers to create and choose the Pareto-optimal solutions based on their preferences.

5.3 Proposed Generalised Regression GA (GRGA)

The solution strategy, discussed in the previous section, is encoded in C++ using a new algorithm called ‘Generalised Regression GA (GRGA)’. This algorithm is illustrated in Figure 5.3. It should be noted that being a state-of-the-art optimisation algorithm, NSGA-II (described in Appendix B) has been chosen as the optimisation engine for GRGA (Deb *et al.*, 2000). However, since GRGA is completely modular it can also be used with any other multi-objective optimisation algorithm for enhancing the algorithm performance in handling problems with complex inseparable function interaction. The steps involved in GRGA are explained below.

1. Run the optimisation cycle until all individuals have rank 0. This ensures that a front containing only non-dominated solutions is achieved. This intermediate front can be assumed to have as many clusters as the number of clusters in the global Pareto front.
2. Identify all the clusters in variable space using tree-clustering analysis.
3. Perform regression analysis individually on the decision variables belonging to each cluster. This gives the correlation coefficients (that show how accurately the regression model represents relationship among variables) and the regression coefficients (that determine the exact nature of relationship) for all the clusters.
4. If the correlation coefficient of at least one cluster is greater than the empirically determined value of 0.7 (obtained by trial-and-error experiments using various values), proceed to Step 5 else continue running optimisation cycle, and performing clustering and regression analysis until the correlation coefficient of at least one cluster becomes greater than 0.7. This ensures that regression analysis is used in subsequent steps only for those clusters in which the correlation coefficient has a value greater than 0.7. This removes the possibility of misleading the search through the use of a regression model that does not accurately represent relationship among variables.
5. If the generation number is a multiple of 10, proceed to Step 6 else go to Step 8.

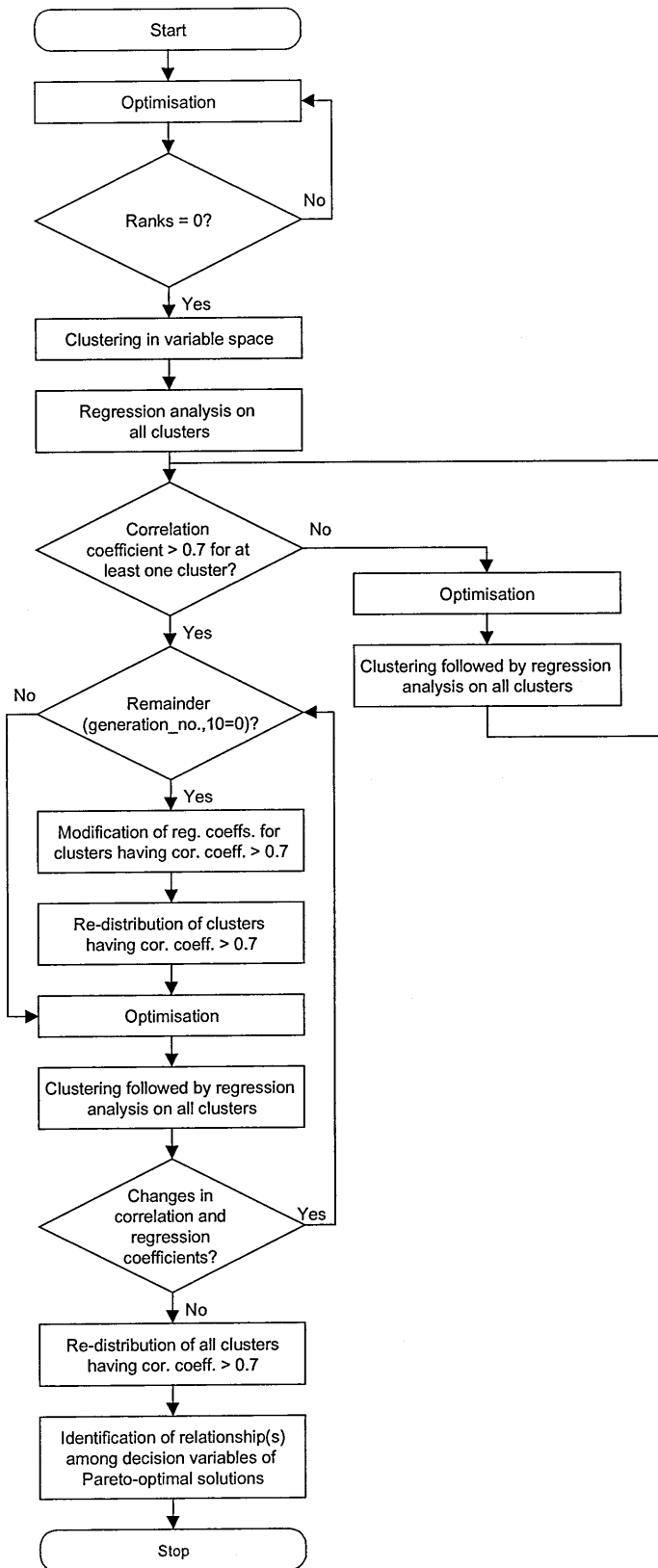


Figure 5.3: Generalised Regression GA (GRGA)

6. After every 10 generations, artificially modify the regression coefficients (for those clusters in which the correlation coefficient has a value greater than 0.7) to guide the search towards the global Pareto front. This is done using the history of change of regression coefficients observed in previous generations. This guides the search towards global Pareto front by preventing it from getting trapped in local fronts.
7. After every 10 generations, re-distribute the solutions, using modified regression coefficients, in those clusters in which the correlation coefficient has a value greater than 0.7. The aim of this step is to encourage diversity among solutions. The algorithms that can be used for re-distribution of solutions are discussed in the next section.
8. Proceed to the next generation by running the optimisation process.
9. Perform clustering, and regression analysis on the decision variables belonging to each cluster.
10. If there are any changes in the values of correlation and regression coefficients in the last two generations, go to Step 5 else proceed to Step 11. No changes in the values of these coefficients imply that the Pareto front has been reached and that the algorithm should now be terminated.
11. Re-distribute the final solutions, using regression coefficients, in all those clusters in which the correlation coefficient has a value greater than 0.7. This creates solutions that are well distributed across the Pareto front.
12. Using the regression coefficients corresponding to each cluster, identify the relationship(s) among the decision variables of the Pareto-optimal solutions.

It should be noted that infinite looping is avoided in this algorithm by restricting the maximum number of generations to a pre-determined value. For the sake of simplicity, this feature is not depicted in Figure 5.3.

5.4 Distribution Algorithms

Distribution algorithms are used here for periodically spreading out solutions over their current front. The aim is to encourage diversity among solutions. The distribution algorithm should be able to deal with complex objective functions

without significantly adding to the computational expense of the optimisation algorithm. This section proposes and analyses three different distribution algorithms.

5.4.1 Linear Distribution Algorithm (LDA)

LDA re-distributes the solutions using equally spaced decision variables in their respective ranges. This means that in a problem that has two decision variables x_1 : [0:1] and x_2 : [0:1], the algorithm chooses equally spaced x_1 values in [0:1] such that the number of points chosen is equal to the population size. It then uses results from regression analysis to find the x_2 values corresponding to these x_1 values. The algorithm uses this set of decision variables to form the new individuals and proceeds forward. This algorithm, applied to a problem with two decision variables x_1 and x_2 , is as follows (Figure 5.4).

1. Choose equally spaced values for x_1 in its range. Number of values chosen should be equal to the population size.
2. Use results from regression analysis to get corresponding x_2 values.
3. Map the above set of x_1 - x_2 values back to the function space. The solutions thus attained form the re-distributed set of solutions.

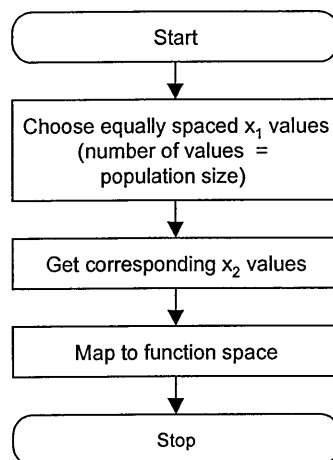


Figure 5.4: Linear Distribution Algorithm (LDA)

The computational complexity of this algorithm is $O(MN)$, where M is the number of objectives and N is the population size. This algorithm is simple to implement but it works on the assumption that well-distributed points in parameter space will give rise

to well-distributed points in function space. This works well for uni-modal objective functions that do not have any steep changes. However, for complex functions this assumption is not valid causing the algorithm to fail.

5.4.2 Random Distribution Algorithm (RDA)

The failure of LDA in handling complex objective functions was the motivation for the development of RDA. This algorithm first generates a set of random values for x_1 in its range such that the number of points generated is equal to the population size. It then uses results from regression analysis to find the corresponding x_2 values and maps the obtained set of x_1 - x_2 values back to the function space. The algorithm then determines unique points from this set and repeats the above process until the number of unique points becomes equal to the population size. This algorithm is described below for a problem that has two variables (x_1 and x_2) (Figure 5.5).

1. Generate random values for x_1 in its range (number of points generated should be equal to population size). The reason for using random values is that it aids the exploration of the entire search space with equal probability.
2. Use regression analysis to get corresponding x_2 values.
3. Map the above set of x_1 - x_2 values back to the function space.
4. Mix the points obtained in Step 3 with pre-determined unique points.
5. Determine the unique points from the combined set obtained in Step 4. This ensures that only those solutions are selected that have the required separation in the function space.
6. Check if the number of unique points obtained is greater than the population size. If yes go to Step 7 else go to Step 1.
7. From the set of unique points obtained in Step 5, randomly delete points until their number becomes equal to the population size.

The computational complexity of this algorithm is $O(MN^2)$, where M is the number of objectives and N is the population size. It is evident that this algorithm ensures equal distribution of solutions in the function space using the concept of unique point. Hence, an appropriate definition of unique point becomes critical to the success of this algorithm. Here, a unique point is defined using the concept of

diversity metric (Δ) given by Deb *et al.* (2000). This concept is described in detail in Appendix C.

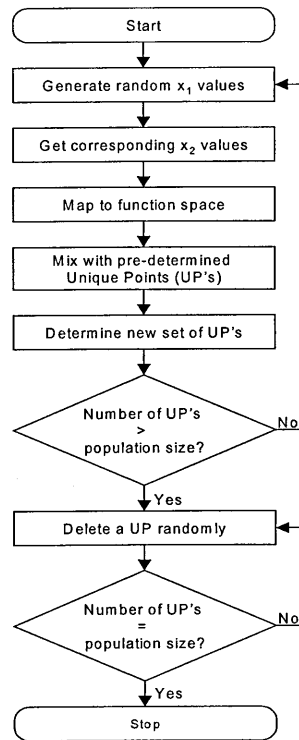


Figure 5.5: Random Distribution Algorithm (RDA)

To demonstrate the concept of unique point, a problem with two objective functions f_1 and f_2 is considered here. Figure 5.6 shows a hypothetical circle, with radius equal to the average Euclidean distance, drawn around the first point. This point is now marked as unique and all other points lying in its circle are deleted. This process is repeated until all the given points have been analysed. This algorithm is depicted in Figure 5.7 for a two-objective problem, and is briefly described below.

1. Sort the given population based on function values. Since all the solutions analysed at this stage are non-dominated (Figure 5.3), sorting for one function would inevitably sort the solutions for the other function. Sorting is carried out to ensure that the algorithm proceeds sequentially from one end to the other of the Pareto front.
2. Find the Euclidean distances between consecutive members of the given population.
3. Find the average of these Euclidean distances.

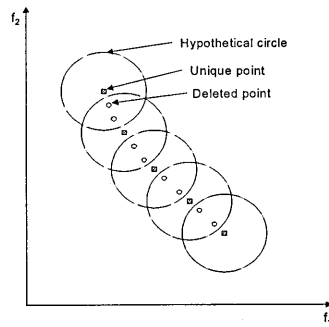


Figure 5.6: Identification of Unique Points (Assuming two objective functions)

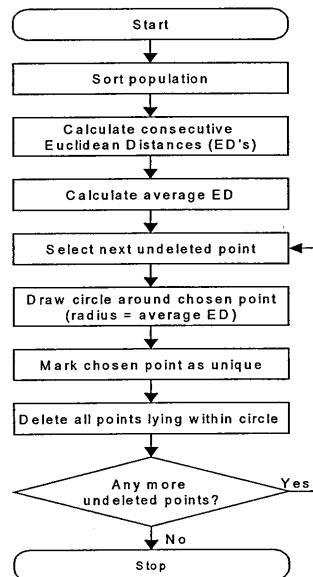


Figure 5.7: Algorithm for Identification of Unique Points

4. Select the next undeleted point based on the sequence in which the population members have been sorted.
5. Draw a hypothetical circle around the selected point with radius equal to the average Euclidean distance.
6. Mark the selected point as a unique point.
7. Delete all other points lying in the hypothetical circle around the unique point (Figure 5.6). This removes any point whose Euclidean distance from the unique point is less than the average Euclidean distance. This ensures that the Euclidean distance between each pair of consecutive unique points is more than or equal to the average Euclidean distance.

8. Check if there is any undeleted/unmarked point left in the population. If yes go to Step 4 else stop the process.

The above algorithm ensures that only well-distributed points are defined as unique so that only these points are passed on to the future iterations of RDA (Figure 5.5).

5.4.3 Hybrid Distribution Algorithm (HDA)

Although RDA has satisfactory performance, it is difficult to implement and has high computational expense. This has led to the development of HDA in which a part of the population is generated by linear distribution and the rest is generated on a point-by-point basis using differences in objective function values between consecutive points in the objective-variable space. The computational complexity of this algorithm is $O(N^2)$, where N is the population size. Being a good compromise between performance and computational expense, HDA is chosen for use with GRGA. This algorithm is described below for a two-variable (x_1 - x_2) problem (Figure 5.8).

1. Generate equally distributed values for x_1 in its range. Number of values generated should be equal to a pre-determined proportion (say 10%) of the population size.
2. Use results from regression analysis to get corresponding x_2 values. Map the above set of x_1 - x_2 values back to the function space.
3. Sort these points based on x_1 values. In each objective-variable space, find the gap between consecutive points in terms of the objective function values. Express the gap as a percentage of the sum of total differences between consecutive points.
4. Generate a new x_1 value as mid-point of two consecutive x_1 values that correspond to the maximum percentage gap, considering all objective-variable spaces.
5. Use results from regression analysis to get the x_2 value corresponding to this new x_1 value. Map the new x_1 - x_2 pair back to the function space.
6. Find the percentage gaps in each objective-variable space between this new point and its immediate neighbours.

7. Check if the total number of generated points is equal to the population size. If yes stop the process else go to Step 4.

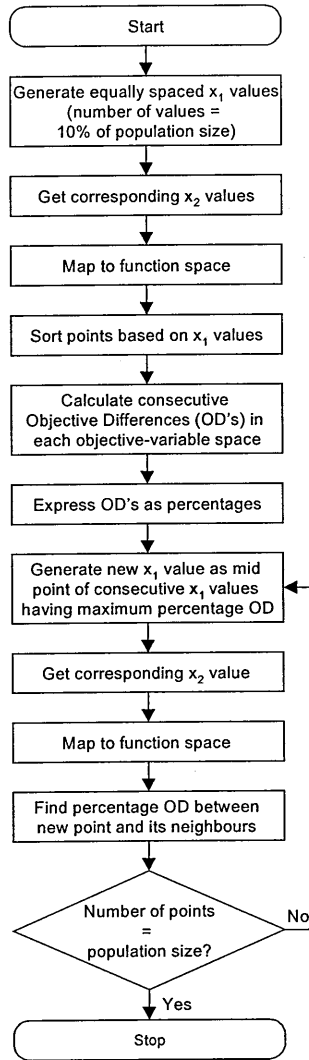


Figure 5.8: Hybrid Distribution Algorithm (HDA)

5.5 Computational Expense of GRGA

GRGA uses NSGA-II as its optimisation engine, whose computational complexity is $O(MN^2)$ (where M is the number of objectives and N is the population size) (Deb *et al.*, 2000). In addition, it uses distribution algorithms for spreading out solutions over their current front in order to encourage diversity. GRGA uses HDA whose overall computational complexity is $O(N^2)$. Since NSGA-II, whose complexity is $O(MN^2)$, is a part of GRGA, the overall complexity of GRGA becomes $O(MN^2)$. The conclusion

of this analysis is that both NSGA-II and GRGA have the same order of complexity ($O(MN^2)$). However, due to the presence of the distribution algorithm, the actual complexity of GRGA is slightly higher than that of NSGA-II.

5.6 Performance Analysis of GRGA

This section compares the performance of GRGA with a state-of-the-art multi-objective optimisation algorithm, NSGA-II, using three test problems.

Table 5.1: Test Problems for Performance Analysis of GRGA (PF: Pareto Front)

Problem	n	Variable Bounds	Objective Functions (Minimisation)	Main Features
ROT (Deb <i>et al.</i> , 2000)	5	[-0.3,0.3]	$f_1(y) = y_1$ $f_2(y) = g(y) \exp(-y_1 / g(y))$ $g(y) = 1 + 10(n-1) + \sum_{i=2}^n [y_i^2 - 10 \cos(4\pi y_i)]$ $y = Rx$ $R = \text{Rotation_Matrix}$	<ul style="list-style-type: none"> • PF: $y_1 \in [\text{Based on } R]$, $y_i = 0, i=2, \dots, n$ • Convex distribution • Linearly related decision variables • Multiple local fronts • Collateral noise
ZDT4 (Zitzler <i>et al.</i> , 2000)	10	$x_1 \in [0, 1]$ $x_i \in [-5, 5]$ $i=2, \dots, n$	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{x_1 / g(x)}]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^n [x_i^2 - 10 \cos(4\pi x_i)]$	<ul style="list-style-type: none"> • PF: $x_1 \in [0, 1]$, $x_i = 0, i=2, \dots, n$ • Convex distribution • Multiple local fronts (21^9 or 7.94×10^{11}) • Collateral noise
ZDT6 (Zitzler <i>et al.</i> , 2000)	10	[0, 1]	$f_1(x) = 1 - \exp(-4x_1) \sin^6(4\pi x_1)$ $f_2(x) = g(x)[1 - (f_1(x) / g(x))^2]$ $g(x) = 1 + 9[(\sum_{i=2}^n x_i) / (n-1)]^{0.25}$	<ul style="list-style-type: none"> • PF: $x_1 \in [0, 1]$, $x_i = 0, i=2, \dots, n$ • Non-convex distribution • Non-uniformly spaced

5.6.1 Experimental Results

GRGA was tested using three problems namely ROT, ZDT4 and ZDT6 listed in Table 5.1. The objective functions of these problems are plotted in Figure 5.9, Figure 5.10 and Figure 5.11 respectively. As can be seen from Table 5.1, these problems form a representative set since they together possess a number of features that create difficulties for optimisation algorithms. Further, a number of existing multi-objective optimisation algorithms have exhibited limitations in solving these problems. This section compares the performance of GRGA with that of NSGA-II, which demonstrates better performance than most other contemporary algorithms in solving these optimisation problems (Deb *et al.*, 2000).

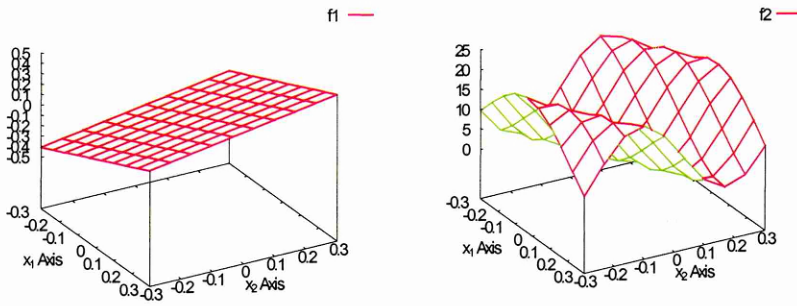


Figure 5.9: Objective Functions of ROT (Assuming Two Variables)

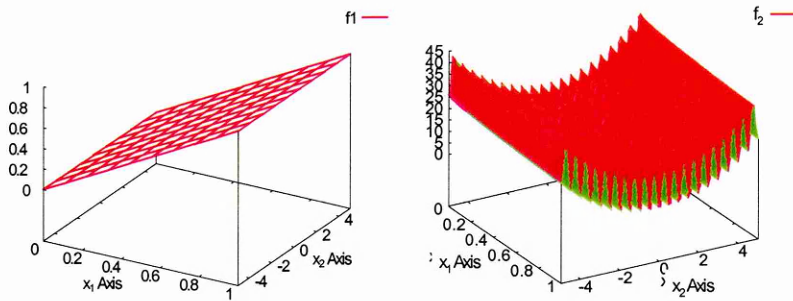


Figure 5.10: Objective Functions of ZDT4 (Assuming Two Variables)

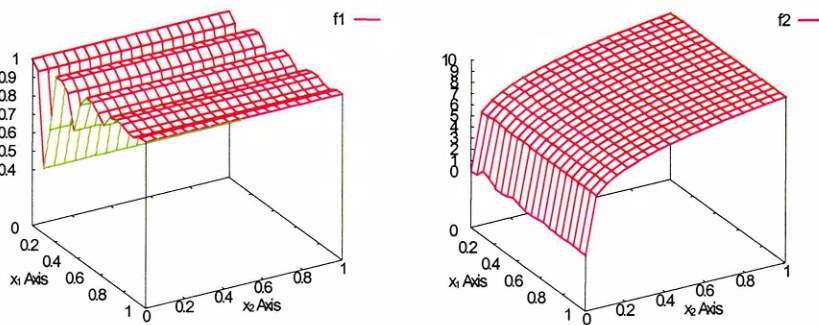


Figure 5.11: Objective Functions of ZDT6 (Assuming Two Variables)

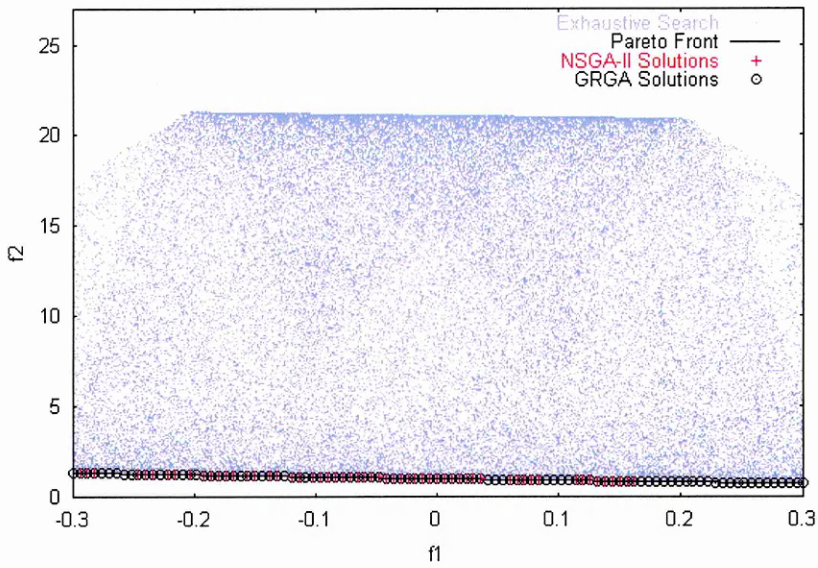
The parameters for carrying out the tests reported in this section are chosen based on their typical values that are used in literature for these test problems. These values are as follows.

- ◆ ROT: 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index.
- ◆ ZDT4: 100 population size, 250 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index.
- ◆ ZDT6: 100 population size, 250 generations, 0.9 crossover probability, 0.1 mutation probability, and simulated binary crossover with 20 crossover distribution index and 20 mutation distribution index.

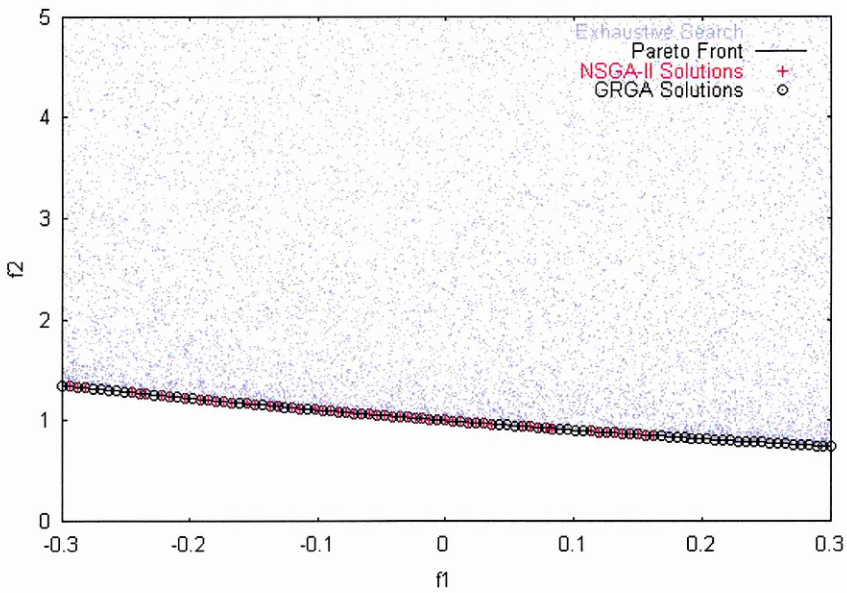
The results obtained from these tests are shown in Figure 5.12 for ROT, Figure 5.13 for ZDT4 and Figure 5.14 for ZDT6. These results form the typical set obtained from 10 runs with different seed values for the random number generator. No major variation was observed in the results with the change in the seed values. To enable fair comparison, the termination condition and re-distribution of final solutions are not applied here for reporting the GRGA results. Also, unless otherwise stated, HDA is used with GRGA in all the tests.

5.6.2 Discussion of Results

Here, the performances of GRGA and NSGA-II are measured, with respect to the goals of multi-objective optimisation (convergence to the Pareto front and diversity across it), using the two metrics proposed by Deb *et al.* (2000). The metrics used here are the convergence metric (γ) and diversity metric (Δ), which are explained in detail in Appendix C. The lower the values of these metrics, the better is the performance of the given optimisation algorithm. The γ and Δ values for the results reported here are shown in Table 5.2. The results obtained from each of the three test problems are discussed below.

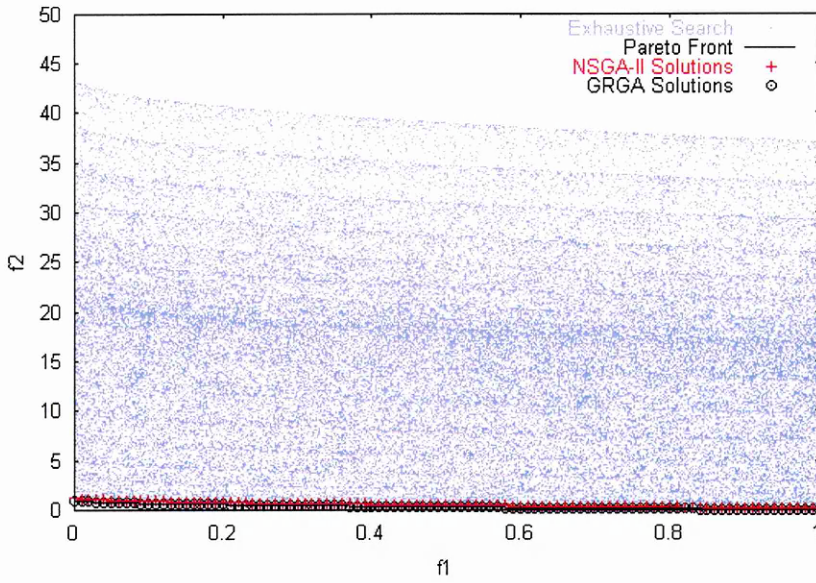


(a)

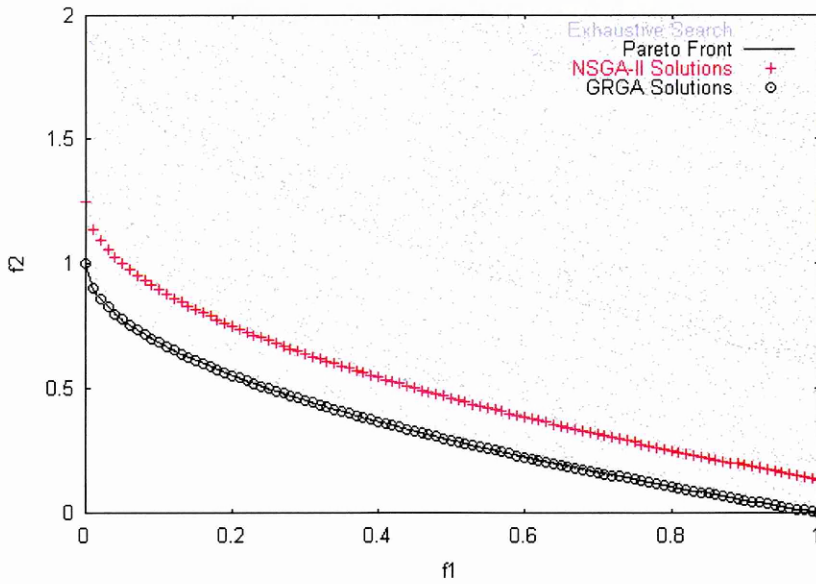


(b)

Figure 5.12: GRGA Performance Analysis Using ROT Problem – (a) Full Search Space (b) Magnified Search Space

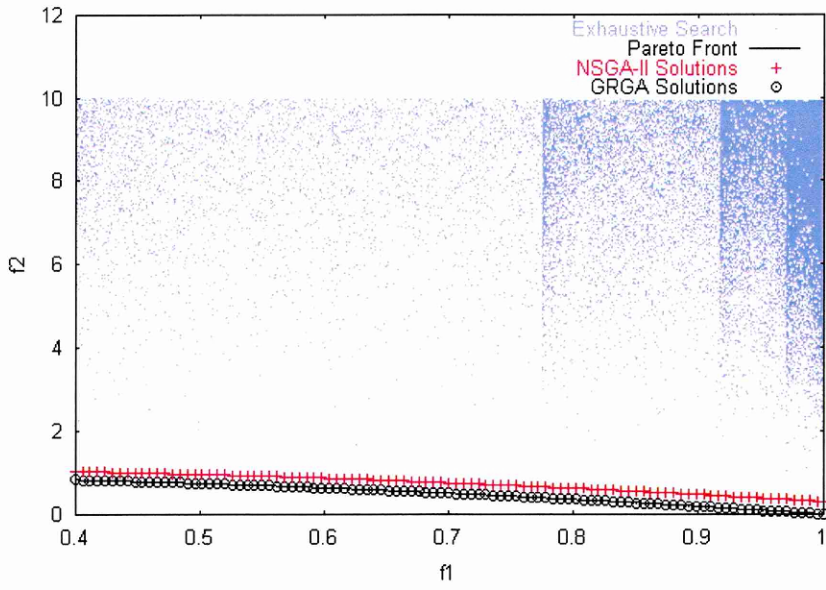


(a)

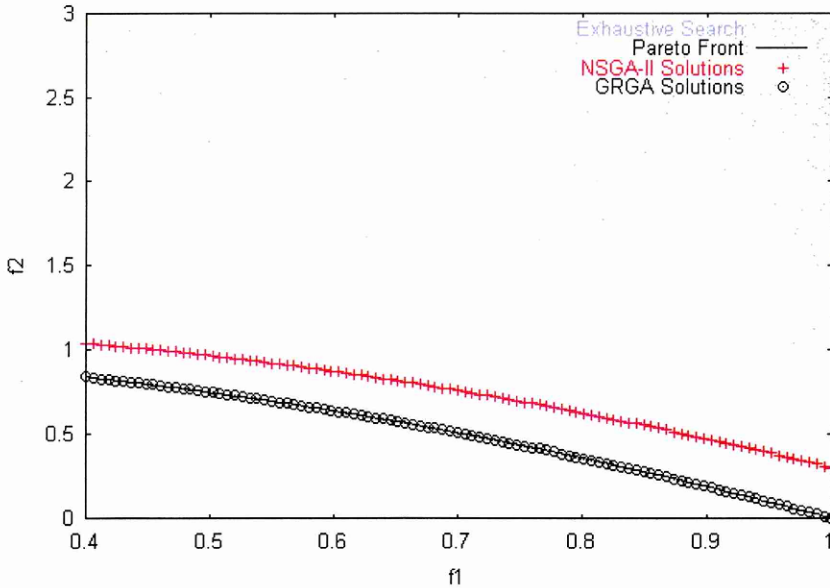


(b)

Figure 5.13: GRGA Performance Analysis Using ZDT4 Problem – (a) Full Search Space (b) Magnified Search Space



(a)



(b)

Figure 5.14: GRGA Performance Analysis Using ZDT6 Problem – (a) Full Search Space (b) Magnified Search Space

Table 5.2: Performance Metrics in ROT, ZDT4 and ZDT6

ROT, ZDT4 and ZDT6		Performance Metrics	
		γ	Δ
ROT	NSGA-II	0.012694	1.192163
	GRGA	0.009162	0.341217
ZDT4	NSGA-II	0.253053	0.702612
	GRGA	0.018982	0.745652
ZDT6	NSGA-II	0.296564	0.668025
	GRGA	0.032871	0.441495

5.6.2.1 ROT

ROT has an externally introduced inseparable function interaction due to the rotation matrix R . This causes ROT to have a linear relationship among decision variables corresponding to the Pareto-optimal solutions. Therefore, the NSGA-II and GRGA require updating all decision variables in a unique way in order to maintain a spread of solutions over the Pareto-optimal front and to converge to the Pareto front. Further, even if a set of Pareto-optimal solutions are obtained, it is difficult to maintain them since any change in one variable must be accompanied by related changes in others in order to remain on the Pareto front. Here, NSGA-II gives inferior distribution of solutions as compared to GRGA. This is illustrated in Figure 5.12. Table 5.2 also supports this by showing a much lower value of Δ for GRGA as compared to that for NSGA-II. The reason for this is that the Crowded Comparison Operator used in NSGA-II attempts to attain solution diversity using external means, without addressing the inherent features that lead to diversity problems. On the other hand, GRGA addresses the core issue of this problem by determining the relationships among the decision variables of the solutions, and using them to redistribute the solutions for aiding their spread over the current front.

It is interesting to note that both NSGA-II and GRGA exhibit satisfactory convergence to the Pareto front (Figure 5.12), in spite of the presence of multiple local fronts and inseparable function interaction in this problem. This is facilitated in

NSGA-II by the use of Pareto-domination and elitism, and in GRGA by the use of NSGA-II supported by artificial modification of regression coefficients. Table 5.2 exhibits nearly the same values for γ , thereby supporting the above observation.

5.6.2.2 ZDT4

ZDT4 is characterised by the presence of multiple local fronts. As shown in Figure 5.13, GRGA exhibits better convergence as compared to NSGA-II. Table 5.2 also shows that the γ value of GRGA is an order less as compared to that of NSGA-II. This is because the Pareto-domination/elitism strategy used by NSGA-II ceases to produce the driving force towards the global Pareto front once most of the solutions of the population share the same non-domination level. Therefore, in this problem, the NSGA-II solutions get trapped in one of the local fronts. GRGA addresses this drawback of NSGA-II by artificially modifying the regression coefficients after every ten generations using their history of change observed in previous generations. This guides the search towards the global Pareto front by preventing it from getting trapped in local fronts.

Furthermore, this problem does not exhibit any difficulty with respect to the maintenance of diversity of solutions across the Pareto front. Therefore, both NSGA-II and GRGA exhibit satisfactory diversity in this case (Figure 5.13). This is also supported by Table 5.2, which shows small values of Δ for both NSGA-II and GRGA that are nearly same in magnitude.

5.6.2.3 ZDT6

This problem is characterised by a biased search space. In this case as well, GRGA exhibits better convergence as compared to NSGA-II (Figure 5.14). Also, both GRGA and NSGA-II demonstrate satisfactory diversity in this case (Figure 5.14). The γ and Δ values in Table 5.2 support the above observations. The reasons for this behaviour of NSGA-II and GRGA are on the same lines as those discussed for ZDT4, with the difference that in this case the bias in the search space, rather than

multi-modality, creates obstacles in convergence to the Pareto front. This bias also makes it more difficult to maintain diversity across the Pareto front.

5.6.3 Summary of Results

In addition to the above, GRGA identifies the following relationships for the decision variables corresponding to the Pareto-optimal solutions. In solving real-life problems, this information is very useful for designers since it provides them with an easy way of generating a Pareto-optimal solution based on their preferences. Comparison with Table 5.1 reveals that the GRGA has been able to accurately locate the Pareto front in each case.

- ◆ ROT: Pareto front corresponds to $y_i = 0$; $i = 2, \dots, 5$; with y_1 taking values in its range.
- ◆ ZDT4: Pareto front corresponds to $x_i = 0$; $i = 2, \dots, 10$; with x_1 taking values in its range.
- ◆ ZDT6: Pareto front corresponds to $x_i = 0$; $i = 2, \dots, 10$; with x_1 taking values in its range.

It is observed that the proposed algorithm enhances the interaction handling capabilities of NSGA-II. The tests reported in this section lead to the following general conclusions regarding the performance of GRGA.

- ◆ The periodic modification of regression coefficients using history of search is successful in guiding the algorithm towards global Pareto front by preventing it from getting trapped in local fronts.
- ◆ The periodic re-distribution of solutions using regression analysis ensures that better distribution of solutions is attained across the Pareto front.
- ◆ The use of regression analysis for the termination of the optimisation cycle ensures that the process is terminated when no further improvements are possible in terms of convergence to the Pareto front. This reduces unnecessary repetitions of optimisation loop, making the process faster. It is worth noting that the termination condition is solely based on convergence and does not take distribution into account. This is because once the solutions converge to the Pareto front, they are re-distributed by the algorithm using regression analysis.

- ◆ The re-distribution of final solutions performed in this algorithm provides the designers with the whole range of well-distributed Pareto-optimal solutions.
- ◆ The identification of the relationship(s) among the decision variables of the Pareto-optimal solutions enables the designers to create and choose the Pareto-optimal solutions based on their preferences.

5.7 Summary

This chapter has proposed a novel algorithm capable of handling inseparable function interaction in multi-objective optimisation problems. As illustrated below, GRGA meets all the objectives of its development mentioned at the beginning of this chapter.

- ◆ As revealed in Chapter 2, most of the optimisation problems have varying degrees of inseparable function interaction. Since GRGA is capable of handling this interaction and the logic behind it is generic in nature, it is expected that the algorithm would perform better than the existing ones in dealing with a wide variety of optimisation problems. This also includes those multi-objective optimisation problems that involve constraints.
- ◆ GRGA is completely modular in nature. So, it can be used to enhance the interaction handling capability of any optimisation algorithm.
- ◆ GRGA exhibits better performance than the high-performing NSGA-II on a variety of multi-objective optimisation problems.
 - *Better convergence to Pareto front.*
 - *Better distribution of Pareto-optimal solutions.*

This chapter has achieved the following.

- ◆ It has identified the challenges that inseparable function interaction poses for multi-objective optimisation algorithms.
- ◆ It has devised a generic solution strategy for handling this interaction in multi-objective optimisation problems.
- ◆ It has proposed a new multi-objective optimisation algorithm, called Generalised Regression GA (GRGA), based on the solution strategy.

- ◆ It has analysed the performance of the proposed algorithm using existing test problems.

This chapter has proposed an algorithm for handling inseparable function interaction in multi-objective optimisation problems. The next chapter deals with the second category of variable interaction: variable dependence. It proposes a dedicated algorithm for handling variable dependence in multi-objective optimisation problems.

6 DEVELOPING AN EC TECHNIQUE TO HANDLE VARIABLE DEPENDENCE

As discussed in Chapter 2, variable dependence occurs when the variables are functions of each other, and hence cannot be varied independently. This dependence among decision variables is frequently observed in real-life problems. Chapter 4 identified the research gap in the area of variable dependence, and highlighted the need to develop dedicated optimisation techniques for handling dependence among decision variables in multi-objective optimisation problems. The aim of this chapter is to develop a generic solution strategy and to implement it for proposing this dedicated optimisation algorithm for variable dependence. This chapter attempts to achieve the following.

- ◆ *To identify the challenges that variable dependence poses for multi-objective optimisation algorithms.*
- ◆ *To devise a generic solution strategy for handling this interaction in multi-objective optimisation problems.*
- ◆ *To propose a multi-objective optimisation algorithm based on the solution strategy.*
- ◆ *To analyse the performance of the proposed algorithm using existing test problems.*

In addition, the proposed optimisation algorithm is expected to possess the following features.

- ◆ *Generic, so that it is not specific to a problem.*
- ◆ *Modular, so that it can be used to enhance the capability of handling variable dependence of any multi-objective optimisation algorithm.*
- ◆ *Capability to handle a wide variety of dependent-variable multi-objective optimisation problems in terms of satisfying its two goals.*
 - *Convergence to Pareto front.*
 - *Satisfactory distribution of Pareto-optimal solutions.*

6.1 Challenges for Multi-objective Optimisation Algorithms

Complex variable dependence poses a number of challenges for multi-objective optimisation algorithms. In the presence of variable dependence, the decision variables cannot be varied independently of each other. Also, the search space gets modified creating a new feasible region based on the dependence among decision variables. This is shown in Figure 6.1. Depending upon the nature of variable dependency, additional features (such as bias (non-linearity), multi-modality, deception and discontinuity) may also be introduced in the problem. A generic GA independently varies the decision variables and works in the feasible region that does not take variable dependence into account. So, it creates solutions that have limited practical significance since they do not lie in the actual feasible region of the search space.

The dependence among decision variables is frequently observed in real-life problems. Here, the effects of dependence among decision variables are illustrated graphically using a real-life example in which the Resistance (R) of a wire is defined in terms of two parameters, namely Temperature (T) and Stress (S) (Kreyszig, 1993). Here, T is $Random(T_1, T_2)$ and S is $f(T) + Random(S_1, S_2)$ (Equation 6.1).

$$\begin{aligned} R &= F(S, T), & \text{Equation 6.1} \\ T &= Random(T_1, T_2), \\ S &= f(T) + Random(S_1, S_2). \end{aligned}$$

This real-life problem is analogous to the example discussed earlier. As can be seen, Temperature T can take any random value between T_1 and T_2 . On the other hand, Stress S has two components. The first component is a function of Temperature T and the second is a random number lying between S_1 and S_2 . It should be noted here that Temperatures T_1 and T_2 are defined by the range of the heating device, and Stresses S_1 and S_2 are defined by the range of the loading instrument. The effects of dependence among decision variables are analysed in the following discussion.

- ◆ As expected, both variables T and S cannot simultaneously take random values in their respective ranges. If Temperature T takes a value T_1 , Stress S can take only those random values that lie between $[f(T_1)+S_1]$ and $[f(T_1)+S_2]$. With the change in value of T , the range of random values that S can take also changes. So, the variables cannot be varied independently of each other.

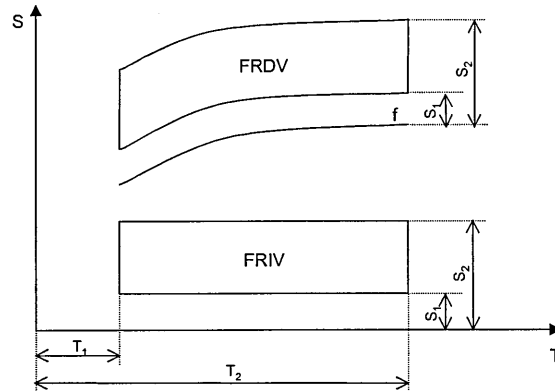


Figure 6.1: Relationship between Stress(S) and Temperature(T) (FRIV: Feasible Region with Independent Variables and FRDV: Feasible Region with Dependent Variables)

- ◆ As shown in Figure 6.1, which gives a graphical representation of this problem, the presence of dependence among decision variables modifies the variable search space. Figure 6.1 represents the search space for both the cases: (i) without dependence among decision variables and (ii) with dependence among decision variables. These two cases are analysed below.
 - *Without Dependence:* If there is no dependence among decision variables, both variables T and S can independently take random values in their respective ranges. This gives a rectangular shape to the T - S search space, shown as FRIV (Feasible Region with Independent Variables) in Figure 6.1.
 - *With Dependence:* The presence of dependence among decision variables modifies the shape and location of the search space. It makes the T - S search space take the shape and location based on the nature of function $f(T)$. The modified search space is shown as FRDV (Feasible Region with Dependent Variables) in Figure 6.1.

6.2 Alternative Strategies for Handling Variable Dependence

Chapter 2 provided a survey of techniques for dealing with the two steps involved in solving the dependent-variable optimisation problems: identification of dependency relationships and classification of variables. This section presents alternative solution strategies that can utilise these techniques for handling the two categories of dependent-variable optimisation problems: those with and those without dependency equations, defined in Chapter 2.

6.2.1 Solving Optimisation Problems Having Known Dependency Equations

The two steps involved in solving the optimisation problems that have known dependency equations are as follows.

6.2.1.1 Step 1: Identification of Dependency Relationships

In these problems, the dependency equations are given to the user, but he/she still needs to make sure that these relationships are free of cyclic dependencies. Step 2 deals with the removal of cyclic dependencies and the identification of independent variables.

6.2.1.2 Step 2: Classification of Variables

The techniques discussed in Chapter 2, TDs and DA, can be used for the classification of variables into dependent and independent, and for the removal of cyclic dependencies. The independent variables define the GA chromosome. The dependent variables are then calculated using the dependency equations, and their bounds are treated as constraints. Finally, the objective functions are calculated using the complete set of variables. This solution procedure is pictorially presented in Figure 6.2.

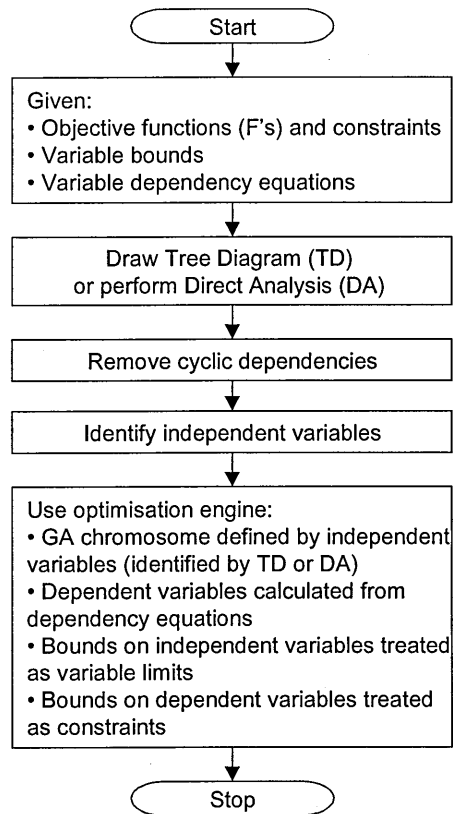


Figure 6.2: Solving Optimisation Problems Having Known Dependency Equations

6.2.2 Solving Optimisation Problems Having Unknown Dependency Equations

In these problems, the dependency equations are unknown, but multiple sets of variable values are provided to the user from which the dependency relationships can be inferred using one of the data modelling techniques, viz. RA, NNs and PM, discussed in Chapter 2. The application of these techniques for solving dependent-variable optimisation problems is presented in this section. The two steps involved in solving these optimisation problems are discussed below for each of the three data modelling techniques.

6.2.2.1 Regression Analysis (RA)

When the RA is used, the two steps involved in the solution procedure of a dependent-variable optimisation problem are as follows.

Step 1: Identification of Dependency Relationships

For solving an optimisation problem that has interaction among decision variables, the RA can be applied to the given sets of variable values. In this way, the equations that define the interaction among decision variables can be obtained. RA may be designed in such a way that it creates equations that do not have any cyclic dependencies. Alternatively, the techniques TDs and DA can be used to identify and remove any cyclic dependencies.

Step 2: Classification of Variables

The techniques, TDs and DA, can also be used to carry out the second step of the solution procedure, which is to identify the independent variables that form part of the GA chromosome. Figure 6.3 summarises the application of RA for solving optimisation problems that have dependence among decision variables.

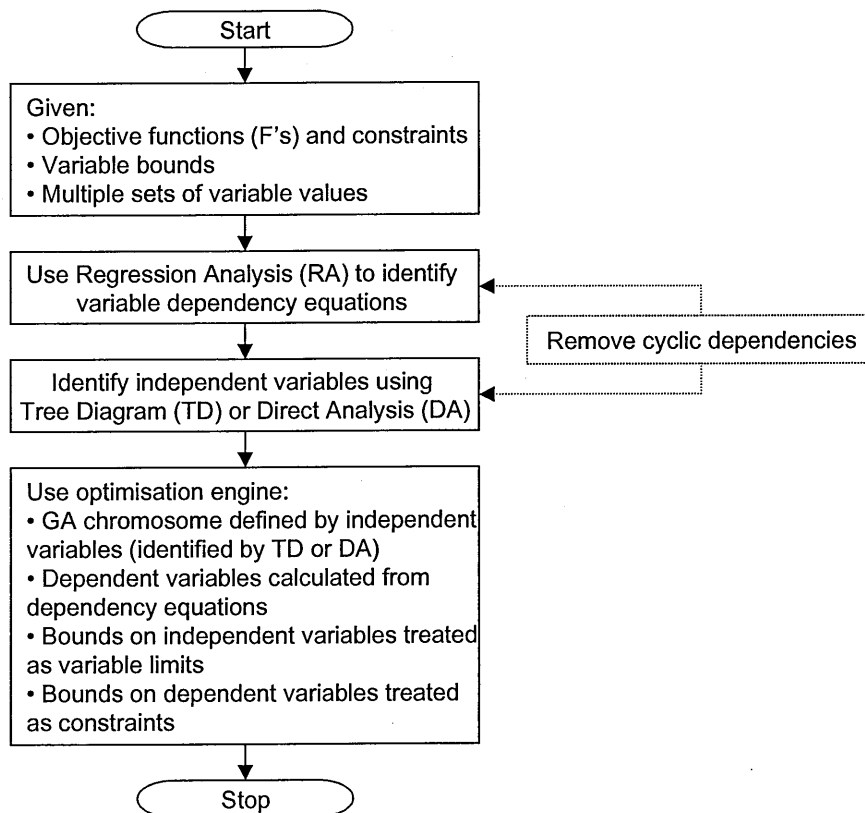


Figure 6.3: Application of Regression Analysis (RA) for Solving Dependent-variable Optimisation Problems

6.2.2.2 Neural Networks (NNs)

The steps involved in solving the dependent-variable optimisation problems using NNs are as follows.

Step 1: Identification of Dependency Relationships

Similar to the RA, the NNs can be used for deriving the dependency relationships from the data provided. However, unlike the RA, in which explicit dependency equations are attained, a NN model that is trained in the given data can be used to predict the values of the dependent variables in terms of the independent ones, without using any explicit equations. Therefore, the application of NNs requires a priori classification of variables as dependent and independent. Since this classification is required to be mutually exclusive, the possibility of having cyclic dependencies does not exist, and the independent variables are assumed to be known.

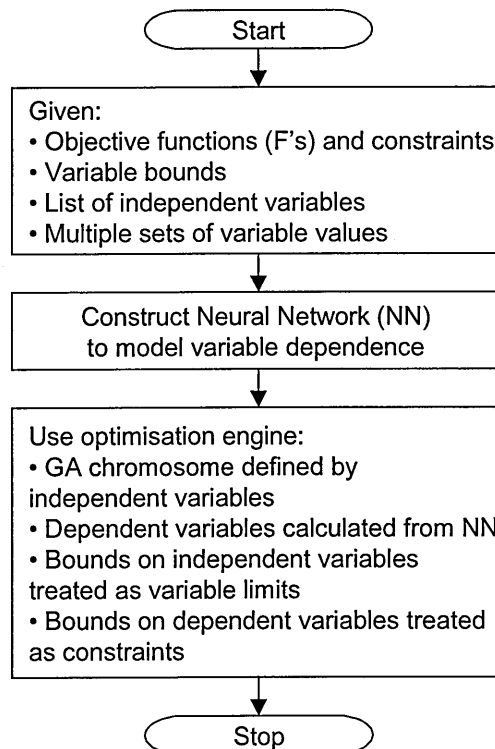


Figure 6.4: Application of Neural Networks (NNs) for Solving Dependent-variable Optimisation Problems

Step 2: Classification of Variables

In this case, the classification of variables into dependent and independent is assumed to be known, making it possible for the GA to use the independent variables in its chromosome. For each solution generated by the GA, the dependent variables are predicted using the NN model. In this way, all the variables in the problem are determined, which can now be used for calculating the objective functions. The above-mentioned solution procedure is summarised in Figure 6.4.

6.2.2.3 Probabilistic Modelling (PM)

If the PM is applied to solve dependent-variable optimisation problems, the steps involved in the solution procedure are as follows.

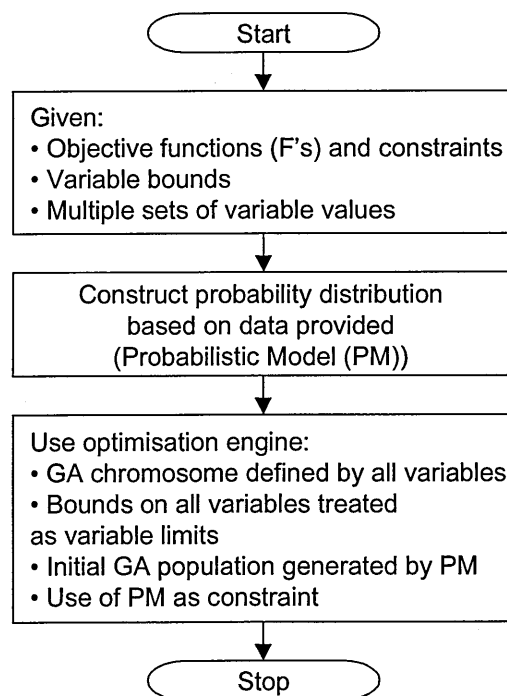


Figure 6.5: Application of Probabilistic Modelling (PM) for Solving Dependent-variable Optimisation Problems

Step 1: Identification of Dependency Relationships

PM creates a probability distribution based on the data provided by the user. The distribution thus created models the relationship among decision variables, and can

be used for creating additional data that have the same relationship among decision variables. Therefore, the PM implicitly models the relationship among decision variables. This is unlike the RA and NNs that attempt to classify the variables (as dependent and independent), and estimate the dependent variables from the independent ones. Since the PM does not classify the variables as dependent and independent, the issue of cyclic dependencies does not arise.

Step 2: Classification of Variables

Since the PM does not classify the variables, the GA chromosome is made up of all the decision variables in the problem, and the initial population is generated by the PM. In successive generations, each individual is analysed to determine its closeness to the PM. To ensure that the solutions created are always in a pre-defined vicinity of the PM, a constraint is introduced that requires the closeness measure (defined by the user) to have a value greater than the pre-defined limit. In this way, the optimisation problem is handled. This solution procedure is depicted in Figure 6.5.

6.3 Proposed Solution Strategy

The previous section analysed some solution strategies for handling optimisation problems that have dependence among decision variables. This discussion is revisited here to select the most appropriate solution strategy for solving real-life dependent-variable optimisation problems.

6.3.1 Analysis of Alternative Solution Strategies

It was stated in Chapter 2 that the lack of systematic research in the area of variable dependence has led to a scarcity of dedicated frameworks that can deal with dependent-variable real-life optimisation problems. However, the previous section presented some solution strategies that attempt to solve these problems using some techniques from related areas of research. This section presents a critical analysis of these strategies in order to select one for this research. This analysis is presented here for both the categories of dependent-variable optimisation problems: those with and those without dependency equations.

6.3.1.1 Solving Optimisation Problems Having Known Dependency Equations

The solution strategy presented in Figure 6.2 performs the two steps, as mentioned in Chapter 2, involved in solving the dependent-variable optimisation problems. However, the choice here is between using a TD or DA for identifying the dependent variables and removing any cyclic dependencies. It is evident that a TD has better visualisation capability but is difficult to be encoded in a computer language, whereas the opposite is true for the DA.

6.3.1.2 Solving Optimisation Problems Having Unknown Dependency Equations

Similar to the previous case, the solution strategies that are presented in Figure 6.3, Figure 6.4 and Figure 6.5 are capable of performing the two steps, as mentioned in Chapter 2, involved in solving the dependent-variable optimisation problems. However, in this case as well, the choice is among the three data modelling techniques (RA, NN and PM) that form part of these solution strategies. Table 6.1 gives a summary of the features of these techniques. These features are analysed below to guide the selection of an appropriate data modelling technique for dealing with the dependent-variable real-life optimisation problems.

- ◆ NNs: As can be seen from Table 6.1, the NNs require a priori knowledge regarding the classification of variables as dependent and independent. Since this information is rarely available in real-life problems, the choice of the NNs is ruled out in spite of their other attractive features.
- ◆ PM: The PM is also a very powerful technique, requiring little information regarding the nature of variables. As shown in Table 6.1, it also has a number of other features that are required for dealing with real-life problems. However, Figure 6.5 shows that the application of PM requires its use as a constraint, which is difficult to implement. Furthermore, the application of PM to model multiple interacting decision variables is a relatively new area of research, and as mentioned in Chapter 2, a number of research issues need to be addressed before it could be chosen for handling real-life problems having multiple real variables.

- ◆ RA: Table 6.1 reveals that the multiple explicit equations that are identified by the RA give good insight to the designer regarding the relationships among decision variables. RA is also easy to implement and maintain. Further, it addresses most of the above-mentioned limitations of NNs and PM. This leads to the choice of RA in the proposed solution strategy.

Table 6.1: Comparison of Data Modelling Techniques

Data Modelling		Data Modelling Techniques		
		Regression Analysis (RA)	Neural Networks (NNs)	Probabilistic Modelling (PM)
Features	Difficulty of Implementation	Medium	High	Very high (due to many open issues)
	Accuracy	Dependent on degree of RA equation	Dependent on number of hidden units	Dependent on choice of modelling method
	Computational Expense	Low	High	Medium
	Nature of Dependency Relationships	Explicit	Explicit (for given dependent variables)	Purely implicit
	Identification of Multiple Dependency Relationships	Multiple RA equations	Built-in multiple relationships (based on choice of NN structure)	Built-in multiple relationships
	Identification of Independent Variables	Through multiple repetitions of RA	Not possible	Not required
	Difficulty of Data Addition	Medium (repetition required)	Medium (repetition required by most NNs)	Low (updating required)

6.3.2 Proposed Solution Strategy

The above analysis leads to the choice of RA for the solution of those dependent-variable problems in which the dependency equations are not known. Here, RA is applied in such a way that it resolves all cyclic dependencies in the model. The application of RA reduces the given optimisation problem to the one in which the dependency equations are known. It is now required to identify the independent variables that form part of the GA chromosome. Chapter 2 discussed two tools for analysing the dependency equations in order to identify the independent variables and remove any cyclic dependencies. Based on this analysis, the TDs are chosen here for visual representation of relationships among decision variables and for removal

of cyclic dependencies. Since the TDs are difficult to be encoded in a computer language, the method of DA is used to automate the process of identification of independent variables and removal of cyclic dependencies. This strategy uses the strengths of both TDs and DA, while avoiding their weaknesses. It should be noted that the removal of cyclic dependencies is not required when RA is used, since all the dependency equations provided by it are free of these dependencies. The main features of this novel solution strategy can be summarised as follows.

- ◆ This strategy takes a holistic view of variable dependence in order to propose a complete framework for dealing with dependent-variable optimisation problems.
- ◆ It can deal with both the categories of dependent-variable optimisation problems: those with and without dependency equations.
- ◆ It performs both the steps, as mentioned in Chapter 2, involved in solving these optimisation problems.
- ◆ It uses existing tools and techniques for proposing this novel framework.
- ◆ It provides insight into the design model provided by the user, thereby giving him/her the opportunity of removing any inconsistencies that arise in the form of cyclic dependencies.

6.4 Proposed Genetic Algorithm for Variable Dependence (GAVD)

In this section, the solution strategy that is introduced in the previous section is implemented to propose a novel algorithm, called Genetic Algorithm for Variable Dependence (GAVD), which is capable of solving real-life optimisation problems that have dependence among their decision variables. Figure 6.6 gives a flowchart of GAVD. As can be seen from this figure, the GAVD comprises of the following two parts, corresponding to the two steps involved in solving the dependent-variable optimisation problems.

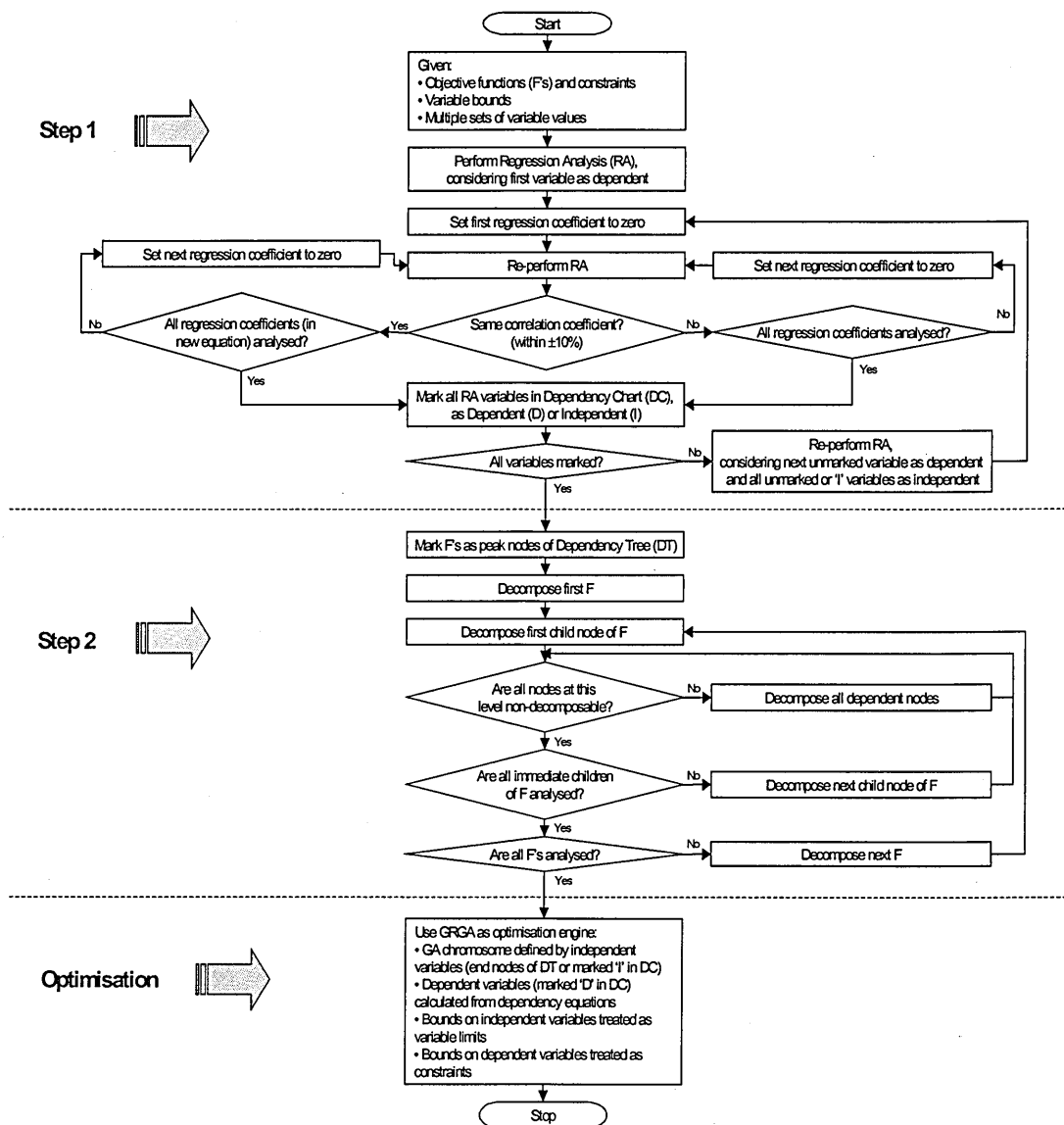


Figure 6.6: Genetic Algorithm for Variable Dependence (GAVD)

6.4.1 Step 1: Identification of Dependency Relationships

This step is omitted in the case when the dependency equations are given to the user. In the other case, this step analyses the given data for identifying multiple dependency equations, while keeping the computational expense as low as possible. GAVD applies RA in such a way that it not only identifies all non-decomposable relationships among decision variables but also removes any cyclic dependency in

those relationships. To attain this, a strategy that ensures better ‘book keeping’ is adopted. The salient features of this strategy are discussed below.

- ◆ The RA that is used in GAVD breaks down a regression equation until it becomes non-decomposable. In this way, all the underlying relationships among decision variables are identified.
- ◆ A Dependency Chart (DC), which is a tool for DA, is maintained to keep track of the variables that are identified as dependent (D) and independent (I) in the regression process. In this way, unnecessary repetitions of RA are avoided for the variables that have already been identified as D or I. This also ensures that the regression equations do not involve any cyclic dependency.
- ◆ When determining the regression equation for a given variable, only those variables that are marked as ‘I’ or are unmarked in DC are considered as independent. This guarantees that the variables that are identified as ‘D’ are not considered as independent in subsequent stages of the RA, thereby ensuring that the regression equations obtained are as non-decomposable as possible. This also reduces the number of variables that are considered at each stage of the RA.

6.4.2 Step 2: Classification of Variables

In both the categories of dependent variable optimisation problems (with and without dependency equations), TDs are used for visual representation of relationships among decision variables. Using the given dependency equations or the regression equations determined in the previous step (as the case may be), a Dependency Tree (DT) is constructed. This tree, which is a form of TD, gives a visual representation of dependency relationships. The end nodes of this tree are the independent variables. The DT also aids in the identification of cyclic dependencies that may be present in the given dependency equations. A typical DT is shown in Figure 6.7. This DT represents Equation 6.2.

$$F = F(A, B, C, D),$$

Equation 6.2

$$A = f_1(B, D, E),$$

$$B = f_2(I),$$

$$C = f_3(A),$$

$$D = f_4(A, B, C, G) = f_4(B, G),$$

$$E = f_5(G, H).$$

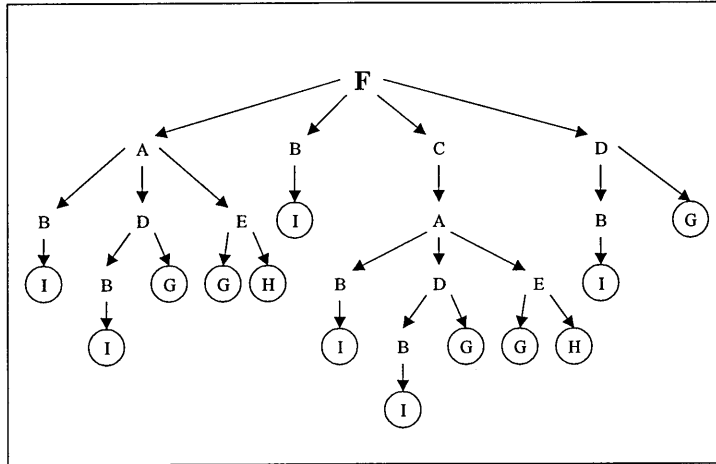


Figure 6.7: An Example of a Dependency Tree (DT) (*F*: Objective Function and *A*, *B*, *C*, *D*, *E*, *G*, *H*, *I*: Decision Variables)

Since TDs are difficult to be encoded in a computer language, the method of DA is used in both the categories of dependent variable optimisation problems, to automate the process of identification of independent variables and removal of cyclic dependencies. Here, the DC is used to identify the independent variables as those that are marked as 'I'. The construction of this chart also aids the identification and removal of cyclic dependencies from the given dependency equations.

Finally, GAVD makes use of GRGA as the optimisation engine. Here, the independent variables, identified in the previous step, define the GA chromosome. For each alternative solution generated by the GA, the dependency equations are used to calculate the values of the dependent variables. It should be noted here that the bounds on independent variables are treated as variable limits and those on dependent variables are treated as constraints.

6.4.3 Computational Expense

Since GAVD uses GRGA as its optimisation engine, the basic operations of GRGA also form part of GAVD. In addition, it uses the RA to model the relationship among decision variables. The RA used in GAVD has an overall complexity of $O(n^2)$ for the determination of all dependency equations (where n is the number of variables in the problem). Since GRGA, whose complexity is $O(MN^2)$, is a part of GAVD, the overall complexity of GAVD becomes $O(MN^2+n^2)$ (where M is the number of objectives and N is the population size). Since in most cases the value of N is much greater than that of n , the conclusion of this analysis is that both GAVD and GRGA have nearly the same order of complexity ($O(MN^2)$). However, due to the presence of RA, the actual complexity of GAVD exceeds that of GRGA by an additional amount of $O(n^2)$.

6.5 Worked Examples

To demonstrate the application of GAVD, the following optimisation problems, which have dependence among decision variables, are considered here.

6.5.1 Worked Example 1

Consider the following optimisation problem (Equation 6.3).

$$\begin{aligned} \text{Objective_Function} &\Rightarrow F = F(x_1, x_2, x_3, x_4, x_5), & \text{Equation 6.3} \\ \forall x_i^{(L)} &\leq x_i \leq x_i^{(U)}, i = 1 \dots 5, \\ \text{Given} &\Rightarrow \text{Multiple_Sets_of_Variable_Values.} \end{aligned}$$

Suppose that the underlying relationships among decision variables, which the user is trying to identify, are as follows (Equation 6.4).

$$\begin{aligned} x_1 &= f_1(x_2, x_3), & \text{Equation 6.4} \\ x_3 &= f_2(x_2, x_4, x_5). \end{aligned}$$

The steps involved in solving this problem are as follows.

- ◆ Determine the following equation for x_1 (Equation 6.5).

$$x_1 = v_1(x_2, x_3, x_4, x_5). \quad \text{Equation 6.5}$$

- ◆ No change is observed in correlation coefficient, when the regression coefficient of x_2 is set to zero and the RA is performed. The new equation is as follows (Equation 6.6).

$$x_1 = v_1'(x_3, x_4, x_5). \quad \text{Equation 6.6}$$

- ◆ Reduction is observed in correlation coefficient, when the regression coefficients of x_3, x_4 and x_5 are set to zero in steps and the RA is performed in each step.
- ◆ Mark x_1 as 'D' and x_3, x_4 and x_5 as 'I' in the DC (Table 6.2).
- ◆ Determine the following equation for x_2 (Equation 6.7).

$$x_2 = v_2(x_3, x_4, x_5). \quad \text{Equation 6.7}$$

- ◆ Repetition of the above process for x_2 yields the same equation, as given above.
- ◆ Mark x_2 as 'D' in the DC (Table 6.2). The variables that are marked as 'I' in the DC are treated as independent variables.
- ◆ Draw the DT for the problem (Figure 6.8). The nodes that are encircled in this figure represent the independent variables.
- ◆ Use GRGA as the optimisation engine considering the following.
 - x_3, x_4 and x_5 constitute the GA chromosome.
 - Regression equations determine x_1 and x_2 .
 - Bounds on x_3, x_4 and x_5 are treated as variable limits.
 - Bounds on x_1 and x_2 are treated as constraints.

Table 6.2: Dependency Chart (DC): Worked Example 1

Dependency Chart (DC)		Variables				
		X_1	X_2	X_3	X_4	X_5
Regression Equations	X_1	D		I	I	I
	X_2		D	I	I	I
	X_3					
	X_4					
	X_5					

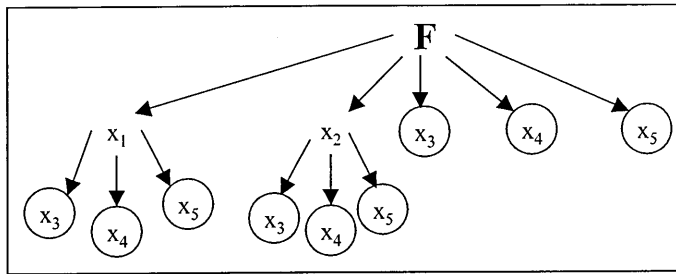


Figure 6.8: Dependency Tree (DT): Worked Example 1

6.5.2 Worked Example 2

Suppose the example optimisation problem mentioned above has the following relationships among decision variables (Equation 6.8).

$$\begin{aligned}
 x_1 &= f_1(x_2, x_3), \\
 x_3 &= f_2(x_4, x_5).
 \end{aligned}
 \tag{Equation 6.8}$$

The steps for solving this problem can be derived from Figure 6.6, and are similar to the ones listed for the previous problem. Table 6.3 and Figure 6.9 respectively present the DC and DT for this example.

Table 6.3: Dependency Chart (DC): Worked Example 2

Dependency Chart (DC)		Variables				
		X_1	X_2	X_3	X_4	X_5
Regression Equations	X_1	D	I		I	I
	X_2					
	X_3			D	I	I
	X_4					
	X_5					

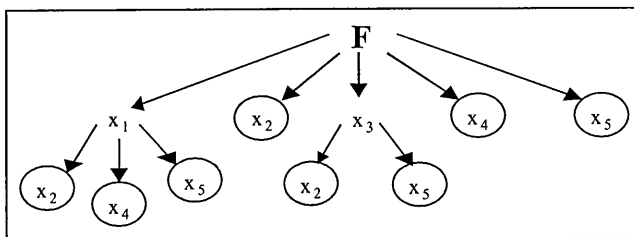


Figure 6.9: Dependency Tree (DT): Worked Example 2

6.5.3 Worked Example 3

Suppose the relationships among decision variables are as follows (Equation 6.9).

$$\begin{aligned} x_1 &= f_1(x_2, x_3), \\ x_4 &= f_2(x_5). \end{aligned} \quad \text{Equation 6.9}$$

The steps for solving this problem are similar to the ones mentioned for the previous two problems. Its DC and DT are given in Table 6.4 and Figure 6.10 respectively.

Table 6.4: Dependency Chart (DC): Worked Example 3

Dependency Chart (DC)		Variables				
		X_1	X_2	X_3	X_4	X_5
Regression Equations	X_1	D	I	I		
	X_2					
	X_3					
	X_4				D	I
	X_5					

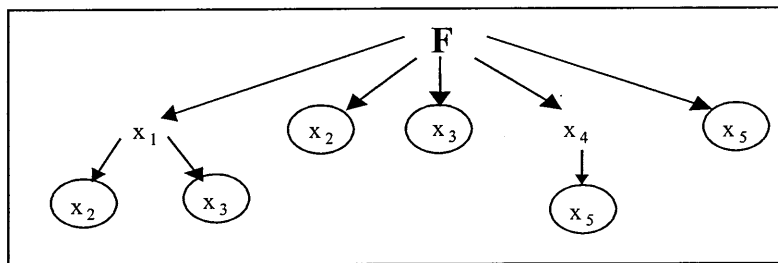


Figure 6.10: Dependency Tree (DT): Worked Example 3

6.6 Performance Analysis of GAVD

Chapter 2 reported a complete lack of test problems in literature for simulating variable dependence in multi-objective optimisation problems. However, in this section, three test problems are created by introducing variable dependence in existing multi-objective optimisation problems. These test problems are modified versions of the test problems used in the previous chapter for the performance analysis of GRGA. The three dependency equations that are introduced here are

linear, cubic polynomial and cyclic. The first can be accurately modelled by the quadratic RA used in GAVD, whereas the last two can only be approximated by it.

6.6.1 Experimental Results

GAVD is tested here using three test problems, as listed in Table 6.5. The objective functions of these problems are plotted in Figure 5.9, Figure 5.10 and Figure 5.11 respectively, and their function search spaces are shown in Figure 5.12, Figure 5.13 and Figure 5.14 respectively. The features of these test problems make them particularly difficult for multi-objective optimisation algorithms. In the absence of any dedicated technique for handling variable dependence, this section compares the performance of GAVD with two high-performing multi-objective optimisation algorithms: NSGA-II and GRGA.

Table 6.5: Test Problems for Performance Analysis of GAVD

Problem	n	Variable Bounds	Objective Functions (Minimisation)	Additional Dependency Equations
ROT (Deb <i>et al.</i> , 2000)	4	[-0.3,0.3]	$f_1(y) = y_1$ $f_2(y) = g(y) \exp(-y_1 / g(y))$ $g(y) = 1 + 10 + [y_2^2 - 10 \cos(4\pi y_2)]$ $y = Rx$ $R = \text{Rotation_Matrix}$	$x_2 = 0.2x_3 + 0.8x_4$ (Figure 6.11)
ZDT4 (Zitzler <i>et al.</i> , 2000)	3	$x_1 \in [0,1]$ $x_i \in [-5,5]$ $i=2, \dots, n$	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{x_1 / g(x)}]$ $g(x) = 1 + 10 + [x_2^2 - 10 \cos(4\pi x_2)]$	$x_2 = 1 - 0.2x_3' - 0.6x_3'^2 - 0.1x_3'^3$, $x_3' = (x_3 + 5) / 10$ (Figure 6.12)
ZDT6 (Zitzler <i>et al.</i> , 2000)	4	[0,1]	$f_1(x) = 1 - \exp(-4x_1) \sin^6(4\pi x_1)$ $f_2(x) = g(x)[1 - (f_1(x) / g(x))^2]$ $g(x) = 1 + 9x_2^{0.25}$	$x_2 = 1 - 0.1x_3 - 0.3x_3$ $\cos^2(2\pi x_3) - 0.2x_4 - 0.2x_4^2$ (Figure 6.13)

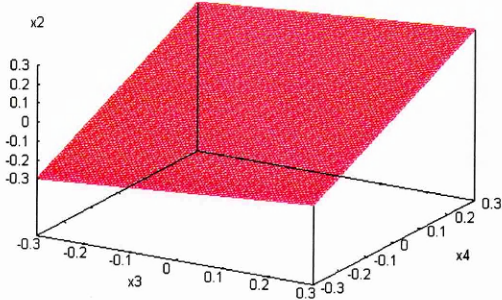


Figure 6.11: Dependency Relationship in ROT

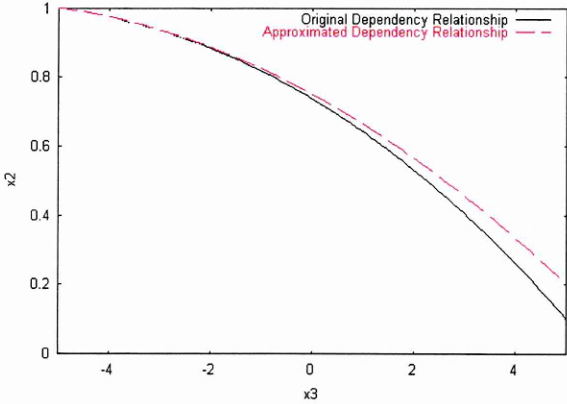


Figure 6.12: Dependency Relationship in ZDT4

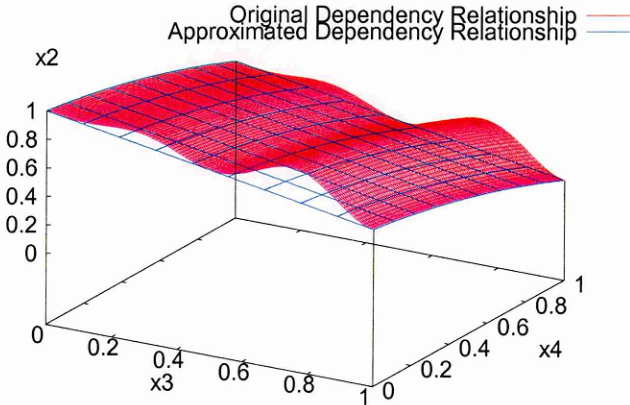
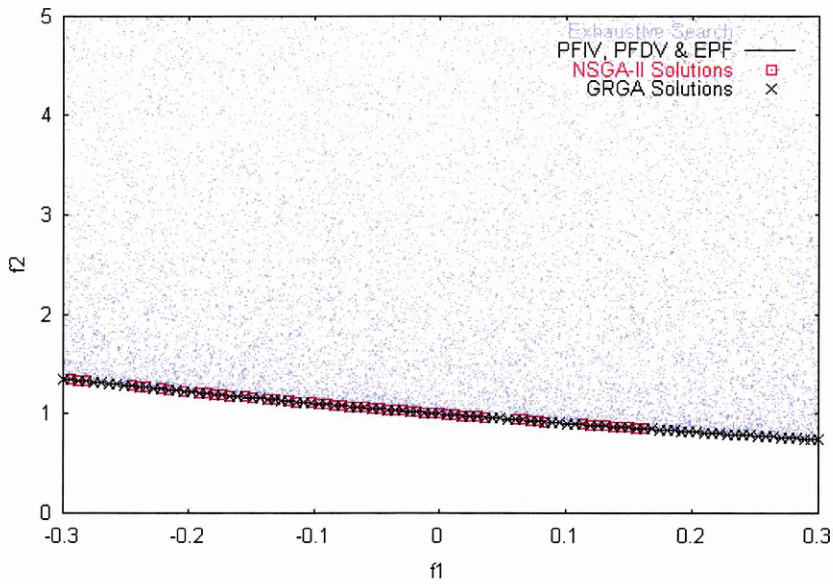
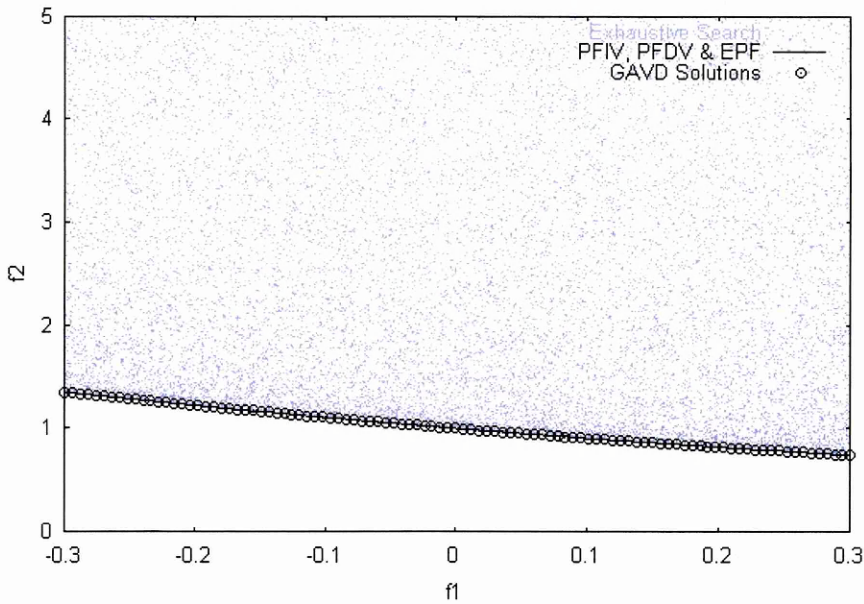


Figure 6.13: Dependency Relationship in ZDT6

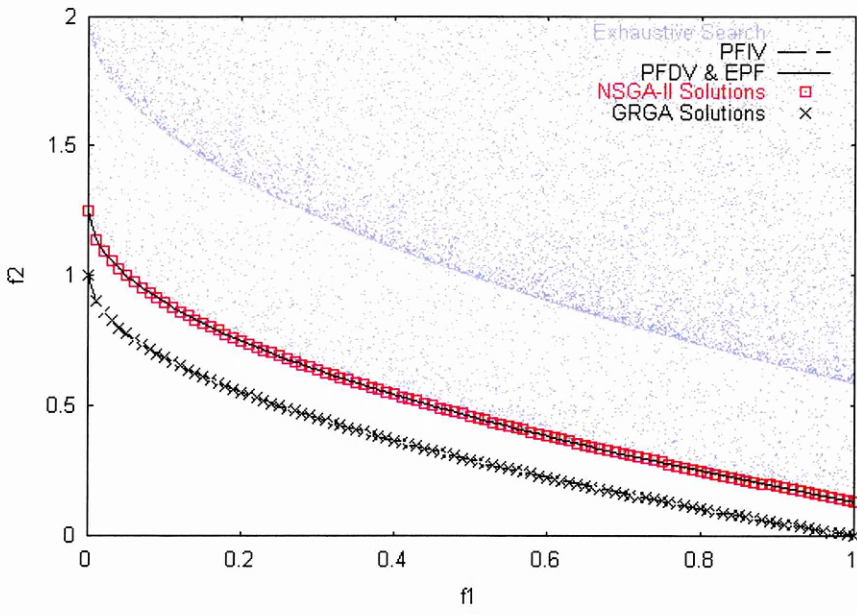


(a)

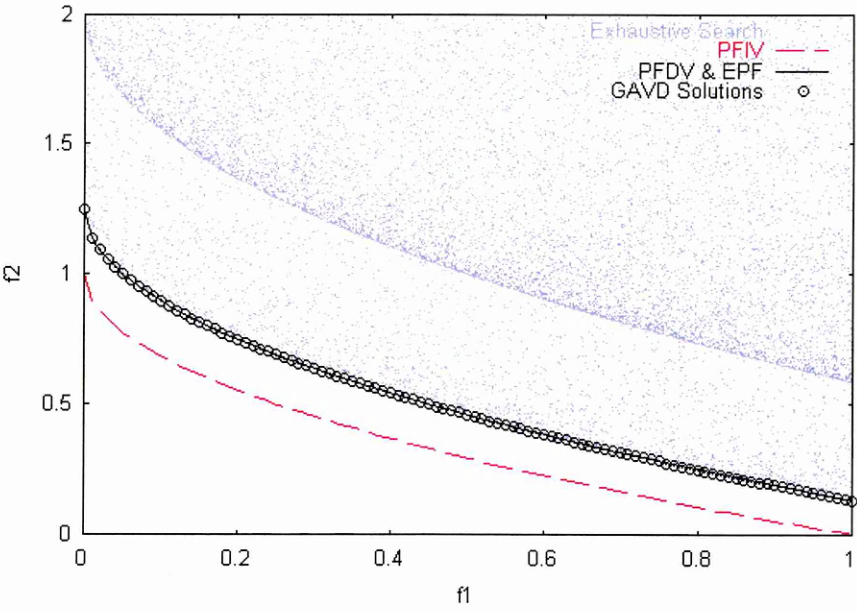


(b)

Figure 6.14: GAVD Performance Analysis Using ROT Problem – (a) NSGA-II and GRGA (b) GAVD (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)

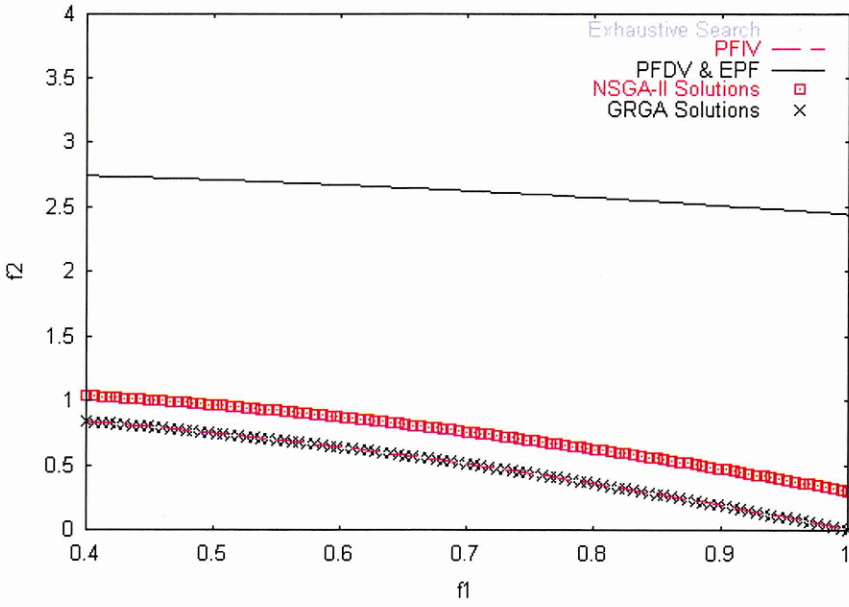


(a)

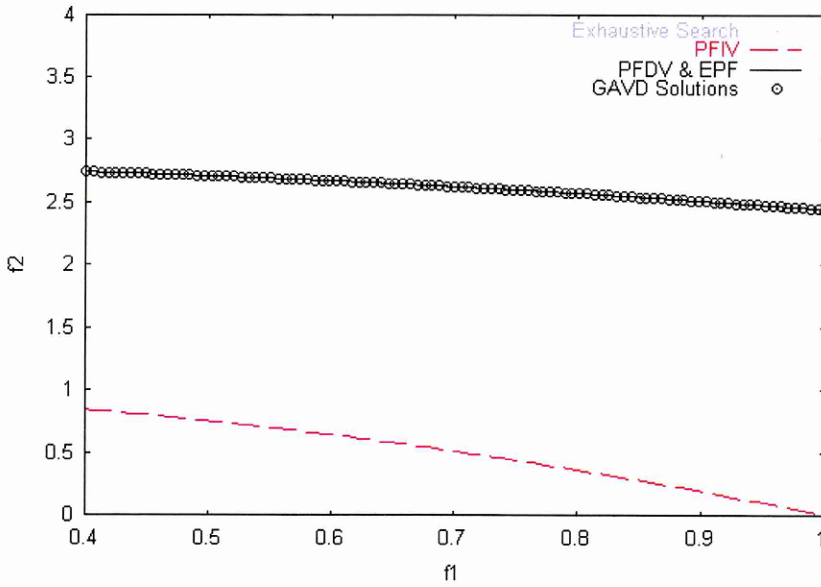


(b)

Figure 6.15: GAVD Performance Analysis Using ZDT4 Problem – (a) NSGA-II and GRGA (b) GAVD (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)



(a)



(b)

Figure 6.16: GAVD Performance Analysis Using ZDT6 Problem – (a) NSGA-II and GRGA (b) GAVD (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)

The parameters for carrying out the tests reported in this section are chosen based on their typical values that are used in literature for these test problems. These values are as follows.

- ◆ ROT: 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index.
- ◆ ZDT4: 100 population size, 250 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index.
- ◆ ZDT6: 100 population size, 250 generations, 0.9 crossover probability, 0.1 mutation probability, and simulated binary crossover with 20 crossover distribution index and 20 mutation distribution index.

The results obtained from these tests are shown in Figure 6.14 for ROT, Figure 6.15 for ZDT4 and Figure 6.16 for ZDT6. These results form the typical set obtained from 10 runs with different seed values. No major variation was observed in the results with the change in seed values. To enable fair comparison, the termination condition and re-distribution of final solutions are not applied here for reporting the GRGA results. Also, unless otherwise stated, HDA is used with GRGA in all the tests.

6.6.2 Discussion of Results

Here, the performances of GAVD, GRGA and NSGA-II are measured, with respect to the goals of multi-objective optimisation (convergence to the Pareto front and diversity across it), using the convergence metric (γ) and diversity metric (Deb *et al.*, 2000) (Appendix C). The lower the values of these metrics, the better is the performance of the given optimisation algorithm. The γ and Δ values for the results reported here are shown in Table 6.6. The results obtained from each of the three test problems are discussed below.

Table 6.6: Performance Metrics in ROT, ZDT4 and ZDT6

ROT, ZDT4 and ZDT6		Performance Metrics	
		γ	Δ
ROT	NSGA-II	0.002095	1.093412
	GRGA	0.000345	0.341113
	GAVD	0.008573	0.347539
ZDT4	NSGA-II	0.042951	0.703519
	GRGA	0.249721	0.725295
	GAVD	0.025735	0.698345
ZDT6	NSGA-II	1.853340	0.556320
	GRGA	2.294010	0.428945
	GAVD	0.029410	0.402756

6.6.2.1 ROT

Along with the challenges introduced by the objective functions, this problem ROT also has a linear dependency relationship among its decision variables (Table 6.5). However, this dependency relationship does not change the range of variables, thereby maintaining the boundaries of the search space. Hence, in this case, the Pareto front does not change with the introduction of variable dependence.

Since the GAVD uses a quadratic RA, it is able to accurately predict the dependency relationship in this case. Therefore, the GAVD converges to the true Pareto front. As shown in the previous chapter, the NSGA-II and GRGA converge to the original Pareto front, which in this case coincides with the Pareto front in the presence of variable dependence. Therefore, in this problem, all the three algorithms have very small γ values as shown in Table 6.6.

Here the GRGA exhibits good distribution of solutions in this problem. Since the GAVD uses GRGA as the optimisation engine, it also gives a satisfactory distribution of Pareto optimal solutions. However, NSGA-II lacks good distribution in this case. These observations are supported by the values of Δ depicted in Table

6.6. As shown in this table, the Δ values of GRGA and GAVD are similar, and are about one-third of the Δ value for NSGA-II.

6.6.2.2 ZDT4

ZDT4 is characterised by the presence of multiple local fronts. Here, a cubic dependency relationship is also introduced in this problem. The introduction of this dependency restricts the values of x_2 in the $[0.1,1]$ range, which is only a fraction of its original range $[-5,5]$. This constraint on the range of x_2 changes the boundaries of the search space, thereby modifying the Pareto front.

In this case, the cubic nature of the dependency relationship prevents its accurate prediction by the quadratic RA used by the GAVD. However, since the equation that it predicts (Equation 6.10) contains that value of x_2 ($=0.5$) in its range that defines the modified Pareto front, the GAVD does not introduce an error in its estimation. Hence in this case as well, the GAVD is able to converge to the modified Pareto front, thereby producing very small values of γ . Here, the GRGA converges to the original Pareto front that does not take into account the dependence among decision variables. Since the introduction of variable dependence modifies the Pareto front, the results produced by the GRGA lie in an infeasible belt. Therefore, the GRGA gives high values of γ in this problem (Table 6.6). Similar to the results shown in Chapter 2, NSGA-II gets trapped in a local front in this problem. However, incidentally in this problem, this particular local front has assumed the role of the global front due to the modification of the search space by variable dependence. This means that here NSGA-II has converged to the modified Pareto front, which gives it a low value of γ , of the same order as that obtained from GAVD.

$$x_2 = 0.7375 - 0.0875x_3 - 0.0075x_3^2, \forall 0 \leq x_3 \leq 1. \quad \text{Equation 6.10}$$

Furthermore, since this problem does not exhibit any difficulty with respect to the diversity of solutions across the Pareto front, all the three algorithms (NSGA-II, GRGA and GAVD) provide satisfactory distribution of solutions. Therefore, in this case, all the three algorithms have nearly the same values of Δ .

6.6.2.3 ZDT6

This problem has a biased search space, and a cyclic dependency equation among its decision variables. The introduction of this dependency restricts the values of x_2 in the $[0.2,1]$ range, which is only a sub-set of its original range $[0,1]$. Therefore, similar to the previous problem, the introduction of variable dependence modifies the search space and the Pareto front in this case.

Since the given dependency equation is cyclic in nature, the GAVD is not able to estimate it accurately. The use of quadratic RA in the GAVD makes it see this cyclic dependency equation as shown in Equation 6.11. However, since the range of x_2 defined by this equation is also $[0.2,1]$, the approximation that is introduced by the quadratic RA does not artificially modify the search space and the Pareto front. Therefore, in this problem, the GAVD converges to the modified Pareto front, giving very small values for γ . However, both GRGA and NSGA-II cannot see the dependency relationships, making them converge to the infeasible regions. It is worth noting here that the GRGA exhibits better convergence to the original Pareto front as compared to NSGA-II. However, it gives a higher value of γ than NSGA-II since the front located by it has a higher average distance from the modified Pareto front as compared to that located by NSGA-II.

$$x_2 = 1 - 0.4x_3 - 0.2x_4 - 0.2x_4^2, \forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1. \quad \text{Equation 6.11}$$

Similar to the previous case, all the three algorithms (NSGA-II, GRGA and GAVD) perform satisfactorily in this case with respect to the distribution of solutions. Therefore, the Δ values of all these algorithms are similar, with NSGA-II having slightly inferior value as compared to the other two algorithms.

6.6.3 Summary of Results

In addition to the above, the GAVD identifies the following relationships among the decision variables corresponding to the Pareto-optimal solutions. Comparison with Table 6.6 reveals that the GAVD has been able to accurately locate the Pareto front in each case.

- ◆ ROT: Estimated Pareto front corresponds to $y_2 = 0$; with y_1 taking values in its range.
- ◆ ZDT4: Estimated Pareto front corresponds to $x_2 = 0.5$; with x_1 taking values in its range.
- ◆ ZDT6: Estimated Pareto front corresponds to $x_2 = 0.2$; with x_1 taking values in its range.

It is observed that the proposed algorithm enhances the variable dependence handling capabilities of GRGA. The tests reported in this section lead to the following general conclusions regarding the performance of GAVD.

- ◆ Since GAVD uses GRGA as the optimisation engine, it inherits all its features, listed in Chapter 5, for effectively dealing with inseparable function interaction in multi-objective optimisation problems. Therefore, GAVD is able to satisfy the two goals of multi-objective optimisation: convergence to the Pareto front and maintenance of diversity across the front, in complex problems.
- ◆ GAVD attaches an additional module to GRGA for dealing with variable dependence in optimisation problems.
- ◆ The capability of GAVD to estimate the dependence among decision variables is limited by the degree of the RA that it uses. Here, the RA that is used is quadratic in nature.

6.7 Summary

This chapter has proposed a new algorithm for dealing with variable dependence in multi-objective optimisation problems. As can be seen, GAVD meets all the objectives of its development set at the beginning of this chapter.

- ◆ GAVD uses a generic methodology for dealing with variable dependence. This methodology can be applied to a vast spectrum of optimisation problems, limited only by the degree of the RA that is being used.
- ◆ GAVD is completely modular in nature. So, it can be used to enhance the capability of handling variable dependence of any multi-objective optimisation algorithm.

- ◆ GAVD exhibits the capability to handle a wide variety of dependent-variable multi-objective optimisation problems in terms of satisfying its two goals.
 - *Convergence to Pareto front.*
 - *Satisfactory distribution of Pareto-optimal solutions.*

This chapter has achieved the following.

- ◆ It has identified the challenges that variable dependence poses for multi-objective optimisation algorithms.
- ◆ It has devised a generic solution strategy for handling this interaction in multi-objective optimisation problems.
- ◆ It has proposed a new multi-objective optimisation algorithm, called Genetic Algorithm for Variable Dependence (GAVD), based on the solution strategy.
- ◆ It has analysed the performance of the proposed algorithm using existing test problems.

The previous chapter and this chapter have respectively proposed two optimisation algorithms (GRGA and GAVD) for handling the two categories of variable interaction: inseparable function interaction and variable dependence. However, there is a need to develop test beds that can be used for controlled testing of the proposed optimisation algorithms on a variety of cases that are difficult to obtain from real-life. The next chapter aims to develop this generic, parametric test bed that can handle objective functions, constraints and variable interaction in a single framework.

7 DEVELOPING A TEST BED FOR ENGINEERING DESIGN OPTIMISATION

The development of optimisation algorithms requires systematic and controlled testing. However, since it is difficult to find a wide variety of real-life cases to support this, it is important to develop test beds that have the required features and enable controlled testing of algorithms. This research concentrates on the development of techniques for handling multiple objectives, constraints and interaction among decision variables in engineering design optimisation problems. It was identified in Chapter 4 that there is a need to develop a generic, parametric test bed that can simulate the complexity introduced by both the objective functions and constraints in a single framework. This test bed should also be able incorporate the two types of variable interaction: inseparable function interaction and variable dependence. The aim of this chapter is to propose a test bed that meets the requirements set above. This chapter is organised in two main parts. In the first part, it proposes a generic, parametric test bed for controlled simulation of multi-objective optimisation problems, having constraints and inseparable function interaction. In the second part, this test bed is extended to incorporate dependence among decision variables. In short, this chapter attempts to achieve the following.

- ◆ *To identify the factors that need to be controlled for simulating multiple objectives, constraints and variable interaction in engineering design optimisation problems.*
- ◆ *To devise a generic strategy for test bed development.*
- ◆ *To apply this strategy for proposing an optimisation test bed for simulating multiple objectives, constraints and variable interaction.*
- ◆ *To develop parametric function prototypes for the proposed test bed.*
- ◆ *To present guidelines and case studies for demonstrating the use of the proposed test bed.*
- ◆ *To compare the proposed test bed with the existing ones.*

In addition, the proposed test bed is expected to possess the following features.

- ◆ *Generic, so that it is can generate a wide variety of optimisation problems.*
- ◆ *Unified framework, to deal with multiple objective, constraints and variable interaction.*
- ◆ *Parametric, to have better control on the complexity of optimisation problems.*
- ◆ *Capability to control the complexity introduced by the two types of variable interaction.*
 - *Inseparable function interaction.*
 - *Variable dependence.*

This chapter proposes two test beds, Reverse Engineered Test Bed (RETB) and RETB-II. The parameters that are provided in the function prototypes of these test beds are summarised in Table 7.1, Table 7.2 and Table 7.3.

7.1 Methodology for Test Bed Development

This chapter proposes a test bed for systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems. The methodology that is adopted here for developing this test bed is given in Figure 7.1, and is explained below.

The first step in the development of a test bed for simulating multiple objectives, constraints and variable interaction is to identify the challenges, as discussed in Chapter 2, Chapter 5 and Chapter 6, which these features of engineering design optimisation problems pose for optimisation algorithms. These challenges are then translated into functions and their parameters that need to be controlled for attaining a 'truly' tuneable test bed. A strategy for the development of a tuneable test bed is evolved and is applied to develop the proposed test bed. The first set of parametric function prototypes are then developed with an aim of giving full control to the user over the complexity of proposed test problems. These prototypes are modified

iteratively until they provide the functions and parameters required in the proposed test bed for systematic and controlled simulation of multiple objectives, constraints and variable interaction.

Table 7.1: Summary of Parameters in S, D and I Functions of RETB

RETB	S		D		I	
	Pars.	Meaning	Pars.	Meaning	Pars.	Meaning
	M	Number of objective functions	k	Number of variables defining D	n	Number of decision variables
	a_i	General shape of S / Pareto front	M_i	Number of parts into which i^{th} variable is sub-divided	k	Number of variables defining D functions
	c_{1i}	Flag to control discontinuity	T	Number of parts into which variable space is sub-divided	b_i	Height of cosine waves
	b_i	Height of cosine waves	a_{ij}	Location of exponential function	c_i	Number of cosine waves
	c_i	Number of cosine waves	b_{ij}	Shape of exponential function	M_i	Number of deceptive fronts
	d_i	Spacing between consecutive cosine waves	R_i	Added to sum of exponential functions	d_i	Height of deceptive exponential function
	e	General location of Pareto front	a_k	Location of D for purely unbiased Pareto fronts	ϵ	Small positive constant number
			M	Number of objective functions	e	Range of values taken by I function

Table 7.2: Summary of Parameters in C_1 , C_2 and C_3 Functions of RETB

RETB	C_1		C_2		C_3	
	Pars.	Meaning	Pars.	Meaning	Pars.	Meaning
	J	Number of feasible belts	J	Number of feasible belts	J	Number of feasible holes
	E_{1j}	Size, location and distribution of feasible regions	m_i	Slope of constraint boundaries	E_j	Location of constraint boundary
	E_{2j}	Size, location and distribution of feasible regions	E_{ij}	Location of constraint boundaries	a_{ij}	Location and shape of constraint boundary
	M	Number of objective functions	E_{2j}	Location of constraint boundaries	P_{ij}	Location and shape of constraint boundary
	a_i	Same values as in definition of S	M	Number of objective functions	M	Number of objective functions
	c_{1i}		a_i	S function parameter	a_i	S function parameter
	b_i		c_{1i}		c_{1i}	Cyclical nature of constraint boundary
	c_i		b_i		b_i	
	d_i		c_i		c_i	
	e		d_i		d_i	
			e		e	General location

Table 7.3: Summary of Parameters in RETB-II

RETB-II	Dependency Function	
	Parameters	Meaning
	N_d	Number of dependent variables
	$N_{ind,i}$	Number of independent variables defining i^{th} dependent variable
	N_i	Total number of independent variables
	n	Number of decision variables
	Q_i	Number of parts into which variable space is sub-divided
	P_{ik}	Polynomial
	c_{1ijk}	Flag to control multi-modality
	b_{ijk}	Height of cosine waves
	c_{ijk}	Number of cosine waves
	d_{ijk}	Spacing between consecutive cosine waves
	e_{ik}	General location
	c_{2ijk}	Flag to control deception
	M_{ijk}	Number of deceptive optima
	g_{ijk}	Height of deceptive exponential function
	p_{ijk}	Value of the decision variable corresponding to deceptive optimum.
	ϵ	small positive constant number
	μ	Mean of normal noise distribution in dependency relationship
	σ^2	Variance of normal noise distribution in dependency relationship (σ is standard deviation).

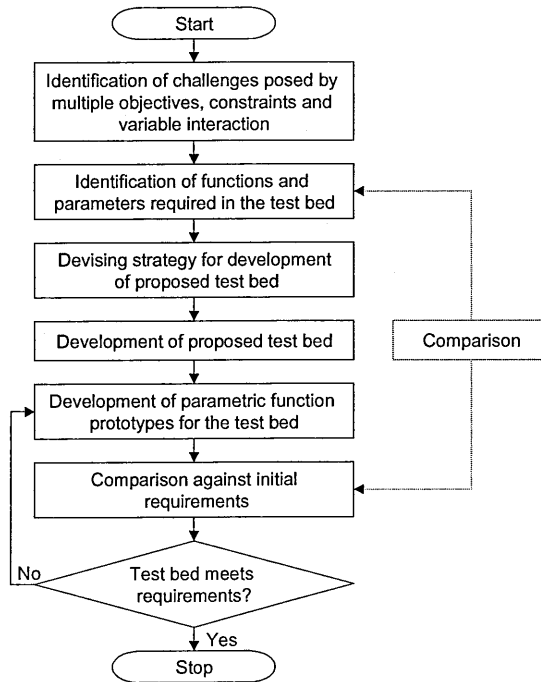


Figure 7.1: Methodology for Test Bed Development

7.2 Identification of Test Bed Parameters

This section identifies the parameters that are required to control the complexity introduced in an optimisation problem by multiple objectives, constraints and interaction among decision variables. Here, the identification of test bed parameters is carried out based on the challenges, as discussed in Chapter 2 and Chapter 5, which the above-mentioned features of real-life optimisation problems pose for optimisation algorithms. The parameters that are spotted in this section form the basis for the construction of a tuneable test bed in the next section.

There are primarily two goals that a multi-objective optimisation algorithm must achieve: convergence to the Pareto-optimal front and maintenance of population diversity across the front. In view of these two goals, the test bed parameters can be classified into two broad categories. The following discussion identifies the parameters in each of these categories that need to be controlled for attaining a ‘truly’ tuneable test bed. Table 7.4 provides a summary of the discussion in this section, including a detailed list of these parameters.

7.2.1 Convergence to Pareto Front

A multi-objective optimisation problem may have features that obstruct convergence to the Pareto front. In general, these obstacles may lie in the entire search space, and as discussed below, they may be generated both by constraints and objective functions.

- ◆ **Direct Hindrance to Convergence:** The presence of constraints in a multi-objective optimisation problem may create infeasible regions, causing direct obstacles to convergence. In order to have control on the complexity introduced by this feature, it is essential for the test bed to provide suitable parameters in the constraint function (C) for governing the features of these infeasible regions.
- ◆ **Indirect Hindrance to Convergence:** The inherent features of some objective functions create indirect hindrance to convergence. In general, the relationship(s) among the decision variables of Pareto-optimal solutions creates obstacles in convergence by introducing multiple local fronts that compete with the global

front. In order to exercise control over this feature, the test bed should have a dedicated interaction function (I), with parameters to control the characteristics of these local fronts.

Table 7.4: Identification of Test Bed Parameters

Convergence to Pareto Front <i>(Challenges in Entire Search Space)</i>	Maintaining Diversity across Pareto Front <i>(Challenges Close to Pareto Front)</i>
<p>Direct Hindrance to Convergence</p> <ul style="list-style-type: none"> • Caused by constraints. • Number, size, location, distribution and shape of infeasible regions. <p><i>Controlled by Constraint Functions (C)</i></p> <p>Indirect Hindrance to Convergence</p> <ul style="list-style-type: none"> • Caused by multi-modality of objective functions. • Number, location and distribution of multiple fronts. • Number, location and distribution of deceptive fronts. <p><i>Controlled by Interaction Functions (I)</i></p>	<p>Direct Hindrance to Diversity</p> <p>(a) General Features of Pareto Front</p> <ul style="list-style-type: none"> • Caused by objective functions. • Shape and location of Pareto front. <p><i>Controlled by Shape Functions (S)</i></p> <p>(b1) Discontinuity of Pareto Front</p> <ul style="list-style-type: none"> • Caused by objective functions. • Number, size and distribution of disconnected Pareto regions. • Size of transition from continuous to discontinuous feasible regions. <p><i>Controlled by Shape Functions (S)</i></p> <p>(b2) Discontinuity of Pareto Front</p> <ul style="list-style-type: none"> • Caused by constraints. • Number, size, location, distribution and shape of infeasible regions. <p><i>Controlled by Constraint Functions (C)</i></p> <p>(c) Composition of Pareto front</p> <ul style="list-style-type: none"> • Caused by constraints. • Number, size, location, distribution and shape of constraint boundaries. • Proportion of original front vis-à-vis constraint boundaries. <p><i>Controlled by Constraint Functions (C)</i></p> <p>Indirect Hindrance to Diversity</p> <ul style="list-style-type: none"> • Caused by objective functions due to bias towards certain regions of Pareto front. • Number, location and extent of bias. <p><i>Controlled by Diversity Functions (D)</i></p>

7.2.2 Maintaining Diversity across Pareto Front

There may be features in multi-objective optimisation problems that create obstacles in maintaining diversity across the Pareto front. In most cases, these obstacles lie close to the Pareto front, and as described below, they may be created both by constraints and objective functions.

- ◆ **Direct Hindrance to Diversity:** The shape of the Pareto front has a direct impact on the difficulty of maintaining diversity of Pareto-optimal solutions. This relates

to the general features of the front, including its shape and location, and to the degree of discontinuity in the front. Further, the maintenance of diversity is also obstructed in cases where the Pareto front is composed of a combination of original front and constraint boundaries. The shape and location of the front are governed by the objective functions, the front composition by the constraints and the front discontinuity by both objective functions and constraints. The test bed should have a separate shape function (S) to control the general characteristics of the Pareto front, including the features of its disconnected regions. Further, the constraint function C in the test bed should have parameters for controlling the composition and discontinuity of the Pareto front.

- ◆ **Indirect Hindrance to Diversity:** Some objective functions have an inherent bias towards particular regions of the Pareto front. In order to control this indirect hindrance to diversity, the test bed should provide a dedicated diversity function (D) to control the nature of bias in the Pareto front.

In order to construct problems that have dependence among their decision variables, additional parameters are required that represent the challenges that variable dependence poses for optimisation algorithms. These challenges, along with the required parameters, are outlined in Table 7.5 with respect to the two objectives of multi-objective optimisation (convergence to Pareto front and maintaining diversity across it).

Table 7.5: Additional Test Bed Parameters to Incorporate Variable Dependence

Variable Dependence	
Independent manipulation of variables is not possible, leading to modification of search space	
Convergence to Pareto Front	Maintaining Diversity on Pareto Front
<p>Modification of search space</p> <ul style="list-style-type: none"> • Number, location and distribution of multiple fronts. • Number, location and distribution of deceptive fronts. 	<p>Modification of search space</p> <ul style="list-style-type: none"> • Number, location, size and distribution of disconnected regions. • Number, location and extent of bias (non-linearity).

The next section proposes a test bed that creates optimisation problems that do not have any dependence among decision variables. In later parts of this chapter, the test bed proposed in the next section is extended to incorporate variable dependence.

7.3 Proposed Test Bed

This section proposes a parametric test bed based on the factors identified in Table 7.4. This test bed is designed to simulate multi-objective optimisation problems, having constraints and inseparable function interaction (Tiwari *et al.*, 2001c).

7.3.1 Strategy for Development of Proposed Test Bed

Table 7.4 reveals that convergence to the Pareto front is obstructed by constraints and by local fronts in the search space that are generated due to the relationship(s) among the decision variables of the Pareto-optimal solutions. Similarly, the shape of the front, constraints and inherent bias in the search space influence the difficulty of maintaining diversity of solutions across the Pareto front. In short, the design of a multi-objective optimisation test problem requires controlling the shape of Pareto front, relationship(s) among decision variables of the Pareto-optimal solutions, inherent bias across Pareto front and nature of constraints. Therefore, the ideal scenario for test bed development would be to have separate functions for controlling each of these features. These functions and their intended roles are summarised below (Table 7.4).

- ◆ Shape Function (S): This function should be able to directly specify the shape of Pareto front.
- ◆ Diversity Function (D): This function should have the capability of controlling the inherent bias in the problem towards certain regions of the search space and towards certain parts of the Pareto front.
- ◆ Interaction Function (I): This function should be able to specify the interaction among decision variables of the Pareto-optimal solutions, thereby controlling the traps created by multiple local fronts in the search space.
- ◆ Constraint Function (C): This function has two purposes. It should be able to provide controlled hindrance to convergence of solutions towards the Pareto front and to diversity across the front.

In order to attain maximum control over the complexity of test problems, one of the best alternatives would be to begin with an equation for the Pareto front, and then

derive the objective functions based on this equation. This would also enhance the visibility of the problems thus constructed. The discussion that follows analyses the feasibility of this strategy.

In principle, there exist an infinite number of multi-objective optimisation problems that correspond to a given Pareto front. The aim here is to identify a particular problem set whose complexity can be varied in a controlled fashion using functions C , I , S and D . A simple multi-objective optimisation problem, having f_1 and f_2 as the two objective functions, is used here as an example. Here, the given Pareto front is defined in terms of the shape function S (Equation 7.1).

$$f_2 = S(f_1). \quad \text{Equation 7.1}$$

Suppose an optimisation problem is generated using an arbitrary function D that defines f_1 (Equation 7.2). Substituting it in Equation 7.1 gives corresponding f_2 (Equation 7.2). Since f_1 is defined as a function D of decision variables, any bias in function D influences the diversity of solutions across the Pareto front. Therefore, D plays the part of diversity function in this case. The major drawback of this strategy is that its search space is limited only to the Pareto front, since any arbitrary combination of decision variables always gives a point lying on the front.

$$\begin{aligned} f_1 &= D(\vec{x}), \\ f_2 &= S(f_1) = S(D(\vec{x})). \end{aligned} \quad \text{Equation 7.2}$$

An effective way of handling this limitation is to re-define the problem such that f_2 is expressed as a product of two functions, S and I , where the function I is defined in terms of those decision variables that are not included in f_1 (Equation 7.3). In this way, the simultaneous minimisation of f_1 and f_2 requires the function I to be minimised, which drives the search towards the Pareto front. Therefore, the Pareto front of this problem (Equation 7.3) corresponds to the minimum of function I (I_{min}). Hence, the selection of a multi-modal function I would create traps (local fronts) in the search space, thereby obstructing convergence to the global Pareto front. Further, in order to remain on the Pareto front the decision variables require to satisfy the relationship $I = I_{min}$. Therefore, I plays the role of interaction function in this case.

$$\begin{aligned}
\text{Minimise} &\Rightarrow f_1 = D(x_1, \dots, x_m), \\
\text{Minimise} &\Rightarrow f_2 = S(f_1) \times I(x_{m+1}, \dots, x_n), \\
\text{Pareto_front} &\Rightarrow f_2 = I_{\min} \times S(f_1).
\end{aligned}$$

Equation 7.3

To ensure the generality of the proposed test bed, the shape function S should be expressed as a function of both f_1 and I . This puts a restraint on the test bed by requiring S to be a monotonically non-decreasing function of I for a fixed f_1 . This ensures that the global Pareto front occurs for the global minimum of I (I_{\min}). In this way, a generic multi-objective optimisation test bed can be achieved in which the shape of Pareto front, the relationship(s) among variables that defines this front and the hindrance to diversity across the front are explicitly controlled by functions S , I and D respectively. This test bed and its Pareto front are given in Equation 7.4. The formal definition of this test bed is presented in the discussion that follows.

$$\begin{aligned}
\text{Minimise} &\Rightarrow f_1 = D(x_1, \dots, x_m), \\
\text{Minimise} &\Rightarrow f_2 = S(f_1, I) \times I(x_{m+1}, \dots, x_n), \\
\text{Pareto_front} &\Rightarrow f_2 = I_{\min} \times S(f_1, I_{\min}).
\end{aligned}$$

Equation 7.4

7.3.2 Proposed Reverse Engineered Test Bed (RETB)

The strategy for test bed development described above explains how a multi-objective optimisation problem can be reverse engineered to correspond to a given Pareto front, relationship(s) among variables corresponding to the Pareto-optimal solutions and hindrance to diversity. Therefore, the proposed test bed is termed here as Reverse Engineered Test Bed (RETB). RETB and its equation for Pareto front are formally stated in Equation 7.5. For the sake of simplicity, the three RETB functions D , S and I are chosen such that they take only positive values in the search space. Further, to ensure that the Pareto front corresponds to I_{\min} , S is chosen to be a monotonically non-decreasing function of I for a fixed f_1 . Also, the chosen S should monotonically decrease in f_1 for a fixed I , if a continuous Pareto front is desired. The roles of functions D , S , I and C in simulating the complexity of multi-objective optimisation problems are discussed in Section 7.3.1, and summarised in Table 7.4.

$$\begin{aligned}
\text{Minimise} &\Rightarrow f_1 = D(\bar{x}'), && \text{Equation 7.5} \\
\text{Minimise} &\Rightarrow f_2 = S(f_1, I) \times I(\bar{x}'), \\
\text{Constraints} &\Rightarrow C \equiv c_j(\bar{x}) \geq 0, j = 1, 2, \dots, J, \\
\bar{x}' \cup \bar{x}'' &= \bar{x} = \text{Vector_of_decision_variables}, \\
\bar{x}' \cap \bar{x}'' &= \phi, \\
\text{Pareto_front} &\Rightarrow f_2 = I_{\min} \times S(f_1, I_{\min}).
\end{aligned}$$

$$\begin{aligned}
\text{Minimise} &\Rightarrow f_1 = D_1(\bar{x}'), && \text{Equation 7.6} \\
\text{Minimise} &\Rightarrow f_2 = D_2(\bar{x}''), \\
\text{Minimise} &\Rightarrow f_3 = S(f_1, f_2, I) \times I(\bar{x}'''), \\
\text{Constraints} &\Rightarrow C \equiv c_j(\bar{x}) \geq 0, j = 1, 2, \dots, J, \\
\bar{x}' \cup \bar{x}'' \cup \bar{x}''' &= \bar{x} = \text{Vector_of_decision_variables}, \\
\bar{x}' \cap \bar{x}''' &= \phi, \\
\bar{x}'' \cap \bar{x}''' &= \phi, \\
\text{Pareto_front} &\Rightarrow f_3 = I_{\min} \times S(f_1, f_2, I_{\min}).
\end{aligned}$$

RET B can also be extended to more than two objectives. The RET B scheme and its Pareto front for a three-objective optimisation problem are stated in Equation 7.6. Here, the functions D_1 , D_2 , S , I and C carry similar interpretations as in the two-objective case.

7.4 Function Prototypes for RET B

This section develops parametric function prototypes for RET B, which give full control to the user over the complexity of proposed test problems. The basic forms of these prototypes are derived from existing test functions in optimisation, and from known equations in other areas of research. These basic forms are customised here to incorporate the parameters that are required for controlling the complexity introduced in the optimisation problems by multiple objectives, constraints and variable interaction.

7.4.1 Diversity Function (D)

Equation 7.7 defines the function prototype for diversity function D . The main features of this function prototype are listed below.

- ◆ The main purpose of diversity function D is to introduce bias towards certain regions of the search space, and towards certain parts of the Pareto front. Therefore, the exponential function, due to its inherent non-linearity, is chosen here for defining the function D . This introduces an inherent bias in the problem towards certain regions of the search space. This leads to test problems with biased Pareto fronts.
- ◆ In order to provide scalability with respect to number of biased Pareto regions, the test bed sub-divides the variable space and defines different exponential functions for each sub-space. This explains the use of different parameters in the D function for each of the sub-spaces. In this way, Equation 7.7 is able to generate multiple biased Pareto regions, having pre-defined locations and strengths.
- ◆ The parameter of the exponential function is the product of a constant term and a function of x_i . The constant part controls the strength of bias introduced by the D function whereas the variable part is designed in such a way that the exponential function sees it ranging from 0 to 1 in each of the sub-spaces.
- ◆ The exponential part of the D function is multiplied by a parameter and the result is added to another parameter. Both these parameters control the location of bias in the search space. The choice of these parameters also ensures that the overall D function has a maximum value of 1. As will be seen later (Equation 7.10), this simplifies the equation of the RETB Pareto front.

The parameters used in Equation 7.7 and their influences on the complexity of RETB are discussed below.

- ◆ M : It is the total number of objective functions in the problem.
- ◆ k : It is equal to the number of variables defining D . Higher values of k contribute to the biased nature of the search space.
- ◆ M_i : It is the number of equal parts into which the span of i^{th} variable is sub-divided. M_i influences the total number of biased regions in the Pareto front.

- ◆ T : It is the total number of equal parts into which the whole variable space is subdivided. Therefore, T is equal to the total number of biased regions in the Pareto front.
- ◆ a_{ij} : This parameter influences the location of exponential function defined for the j^{th} part of the i^{th} variable. In this way, a_{ij} directly controls the location of bias in the Pareto front.
- ◆ b_{ij} : This influences the shape of exponential function defined for the j^{th} part of the i^{th} variable. A higher value of b_{ij} implies a stronger contribution to the bias by the corresponding a_{ij} .
- ◆ $SVSS_i$: This indicates the i^{th} Sub-set of Variable Search Space ($SVSS$) of the problem. All $SVSS_i$'s are mutually exclusive and their union gives the whole Variable Search Space (VSS).
- ◆ R_i : This is a constant value that is added to the sum of exponential functions corresponding to each $SVSS_i$. The role of R_i 's is to ensure that the biased regions on the Pareto front occur at required locations. Further, R_i values may also be chosen such that the maximum value of the corresponding objective function (f_{kmax}) is 1. As will be seen later (Equation 7.10), this simplifies the equation of the RETB Pareto front.

Equation 7.7

$$D_l(\vec{x}') = \sum_{i=1}^k \sum_{j=1}^{M_i} D_{ij}(\vec{x}') + R(\vec{x}'), \forall l = 1, \dots, M-1, \forall 0 \leq x_1, \dots, x_k \leq 1,$$

$$D_{ij}(\vec{x}') = \frac{a_{ij}}{(1 - \exp(-b_{ij}))} \left[1 - \exp \left(\frac{-b_{ij} \left(x_i - \frac{(j-1)}{M_i} \right)}{\frac{1}{M_i}} \right) \right], \forall \frac{(j-1)}{M_i} < x_i < \frac{j}{M_i},$$

$$D_{ij}(\vec{x}') = 0, \forall x_i \leq \frac{(j-1)}{M_i},$$

$$D_{ij}(\vec{x}') = 0, \forall x_i \geq \frac{j}{M_i},$$

$$R(\vec{x}') = R_i, \forall \vec{x}' \in SVSS_i,$$

$$VSS = \bigcup_{i=1}^T SVSS_i,$$

$$\phi = \bigcap_{i=1}^T SVSS_i,$$

$$T = \prod_{j=1}^k M_j.$$

Equation 7.7 can also be simplified to Equation 7.8, in case of problems having purely unbiased Pareto fronts.

$$D_l(\vec{x}') = a_l [1 - x_l] \quad \forall l = 1, \dots, M-1, \quad \forall 0 \leq x_l \leq 1. \quad \text{Equation 7.8}$$

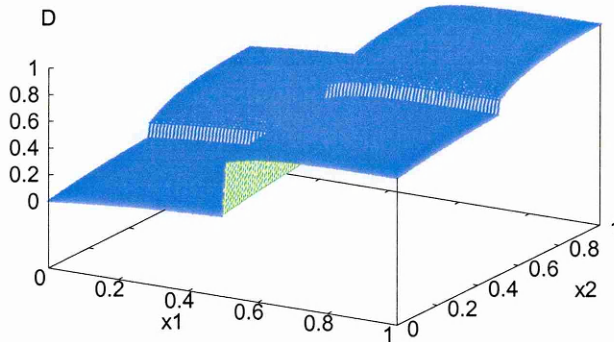


Figure 7.2: An Example of Diversity Function D ($k=2$; $M_i=2,2$; $T=4$;
 $a_{ij}=0.1,0.1,0.1,0.2$; $b_{ij}=1,5,1,3$; $R_i=0,0.2,0.5,0.7$)

Suppose it is required to construct a D function that ranges from 0 to 1, and exhibits four biases at the values of 0.2, 0.5, 0.7 and 1.0 in increasing orders of their strengths. Figure 7.2 depicts this diversity function D . The main features of this function are as follows.

- ◆ Since only one D function is defined here, the problem has only two objective functions ($M = 2$).
- ◆ The given function D is defined by two variables ($k = 2$), and the span of each of these variables is divided into two equal parts ($M_i = 2,2$). This implies that the whole search space is divided into 4 equal parts ($T = M_1 \times M_2$). The given function D takes different parameters and hence different forms in each of these sub-spaces ($SVSS_i$'s).
- ◆ The values of a_{ij} ($= 0.1, 0.1, 0.1$ and 0.2) and R_i ($= 0, 0.2, 0.5$ and 0.7) are chosen such that the biased regions occur at D values of 0.2, 0.5, 0.7 and 1.0. The choice of these parameters also ensures that the overall D function has a minimum value of 0 and a maximum value of 1. As mentioned earlier, the choice of D_{max} (or f_{lmax}) equal to 1 simplifies the equation of the RETB Pareto front.
- ◆ The values of b_{ij} ($= 1, 5, 1$ and 3) ensure that the maximum bias occurs at the D value of 1, followed by the biases that correspond to the D values of 0.7, 0.5 and 0.2 in that order.

7.4.2 Shape Function (S)

The function prototype for shape function S is given in Equation 7.9. The main features of this function prototype are listed below.

- ◆ As evident from this equation, the function S is monotonically non-decreasing with respect to I for fixed f_i 's. This ensures that the Pareto front corresponds to I_{min} , thereby preserving the basic philosophy of RETB.
- ◆ The S function should monotonically decrease in f_i 's for a fixed I , if a continuous Pareto front is desired. In order to attain a discontinuous Pareto front, the function prototype of S relaxes this condition by using a cosine function that gives it a cyclical shape. This creates a discontinuous Pareto front since S is no longer monotonically decreasing in f_i 's for a fixed I . However, the function prototype of S provides a flag variable that is multiplied to the cosine function. Hence, the discontinuity in S could be switched off by setting this flag variable to 0.
- ◆ The parameter of the cosine function is the product of a constant term and a function of f_i . The constant term controls the number of discontinuous fronts and the function of f_i controls the spacing between the discontinuous fronts.
- ◆ The cosine function is multiplied by a function of f_i that controls the variations in the sizes of the discontinuous fronts.
- ◆ Added to the cosine function and its multiplier is another function of f_i that influences the general shape of the Pareto front (concave/convex/linear).
- ◆ Finally, the S function has a constant term that influences the general location of the Pareto front.

Following is the list of parameters used in Equation 7.9, along with their influences on the complexity of RETB.

- ◆ M : This is equal to the total number of objective functions.
- ◆ a_i : This parameter directly influences the general shape (with respect to f_i) of S and that of the corresponding Pareto front. A value of a_i greater than 1 gives a concave front and a value less than 1 gives a convex front.

- ◆ b_i : It determines the height of cosine waves with respect to the i^{th} objective function. A higher b_i implies more variations in the sizes of the disconnected Pareto regions.
- ◆ c_i : It is equal to the number of cosine waves in the range of f_i . This gives the number of disconnected Pareto regions in the optimisation problem.
- ◆ c_{li} : It is a flag to indicate whether the optimisation problem is discontinuous with respect to f_i . c_{li} takes a value of 0 for continuous fronts and a value of 1 for discontinuous fronts.
- ◆ d_i : This parameter influences the spacing between consecutive cosine waves in the f_N - f_i space. Higher values of d_i lead to more closely packed regions.
- ◆ e : It decides the general location of Pareto front. Higher values of e push the Pareto front towards higher values of f_N .
- ◆ $b_i \times d_i$: This product gives an indication of the shape of transition from continuous to discontinuous feasible regions in the f_N - f_i space. Higher values of this product indicate the presence of a long narrow feasible tunnel leading to the Pareto front.

$$S(f_1, \dots, f_{M-1}, I) = 2(M-1)e - \sum_{i=1}^{M-1} \left[\left(\frac{f_i}{If_{i\max}} \right)^{a_i} + c_{li} \left(\frac{f_i}{If_{i\max}} \right)^{b_i} \cos \left(2\pi c_i \left(\frac{f_i}{f_{i\max}} \right)^{d_i} \right) \right], \quad \text{Equation 7.9}$$

$$\forall 0 \leq f_i \leq f_{i\max}, \forall i = 1, \dots, M-1.$$

The equation of Pareto front, for all $f_{i\max}$ and I_{\min} equal to 1, is given in Equation 7.10.

$$f_M = 2(M-1)e - \sum_{i=1}^{M-1} \left[(f_i)^{a_i} + c_{li} (f_i)^{b_i} \cos(2\pi c_i (f_i)^{d_i}) \right], \quad \text{Equation 7.10}$$

$$\forall 0 \leq f_i \leq 1, \forall i = 1, \dots, M-1.$$

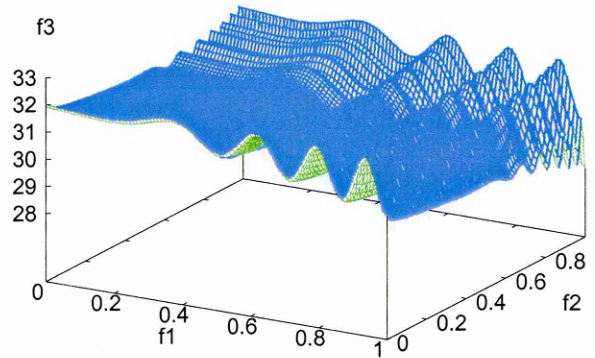


Figure 7.3: An Example of Shape Function S ($M=3$; $a_i=2,0.4$; $c_{1j}=1,1$; $b_i=1,4$; $c_i=4,7$; $d_i=2,4$; $e=8$)

Suppose it is required to construct a shape function S for a problem that has 3 objective functions. This function S is required to be concave with respect to the first objective function and convex with respect to the second. It is required to have 40 discontinuous Pareto surfaces arising from 5 and 8 discontinuous fronts corresponding to the first and second objective functions respectively. Also, the Pareto surfaces with respect to f_2 are required to be closer and with larger variations in their sizes as compared to those with respect to f_1 . This shape function S is plotted in Figure 7.3. The main features of this function are as follows.

- ◆ Since this problem has three objective functions, M takes a value of 3.
- ◆ In order to have concave and convex shapes for S with respect to the first and second objective functions respectively, a_1 takes a value of 2 (greater than 1) and a_2 takes a value of 0.4 (less than 1).
- ◆ Since in this case the Pareto front is discontinuous with respect to both the first and second objective functions, the values of c_{11} and c_{12} are both chosen to be 1. In order to attain 5 and 8 discontinuous fronts corresponding to the first and second objective functions respectively, c_1 takes a value of 4 ($= 5 - 1$) and c_2 takes a value of 7 ($= 8 - 1$). This gives a total of 40 ($= 5 \times 8$) discontinuous Pareto surfaces.
- ◆ A higher value of b_2 ($= 4$) than b_1 ($= 1$) produces discontinuous Pareto surfaces that have more variations in their sizes with respect to f_2 than f_1 . Similarly, a

higher value of d_2 ($= 4$) than d_1 ($= 2$) leads to discontinuous Pareto surfaces that are more closely packed with respect to f_2 than f_1 .

- ◆ e is arbitrarily given a value of 8.

7.4.3 Interaction Function (I)

Equation 7.11 gives the function prototype for the interaction function I . The main features of this function prototype are listed below.

- ◆ Since this function needs to be minimised to obtain the Pareto front, a cosine function is used in its definition to obtain multiple fronts in the function search space. The parameter of this cosine function controls the number of local Pareto fronts in the search space.
- ◆ To ensure that these fronts are created at different locations, an exponential function is multiplied to the cosine function. The parameter of this exponential function controls the concentration of local Pareto fronts in the search space.
- ◆ Similarly, deception is introduced in the function search space using an exponential function that exhibits sudden drop at a given value of decision variable, but remains zero otherwise.
- ◆ The above-mentioned function is multiplied by another exponential function that prevents the deceptive fronts from coinciding with each other. The parameter of this exponential function controls the concentration of deceptive Pareto fronts in the search space. Equation 7.11 also provides parameters to control the number of deceptive fronts.
- ◆ Finally, the I function provides a parameter that allows I_{min} to be 1. As evident from Equation 7.10, this simplifies the equation of Pareto front.

The various parameters used in Equation 7.11, together with their significance, are discussed below.

- ◆ n : This is equal to the total number of decision variables in the problem.
- ◆ k : This is equal to the number of variables that define the D functions (D_1, \dots, D_{M-1} ; where M is the number of objective functions in the problem). Therefore, these k variables form the X' -space and the remaining $(n-k)$ variables form the X'' -space that defines the I function.

- ◆ b_i : This parameter controls the height of cosine waves with respect to the i^{th} decision variable. A higher value of b_i implies greater concentration of fronts towards the centre of function search space.
- ◆ c_i : It is equal to the number of cosine waves in the range of x_i . This gives the number of local Pareto fronts in the function search space.
- ◆ M_i : This is equal to the number of deceptive fronts corresponding to the variable x_i .
- ◆ d_i : This parameter controls the height of the deceptive exponential function corresponding to the i^{th} decision variable. A higher value of d_i implies that the deceptive fronts are farther away from each other.
- ◆ e : This controls the range of values taken by the I function. e should be chosen such that I_{\min} is equal to 1. As evident from Equation 7.10, this simplifies the equation of Pareto front.
- ◆ ε : This is a small positive constant number used with the deceptive exponential function.

$$I(\vec{x}'') = 2e - \sum_{i=k+1}^n [\exp(-b_i x_i) \cos(\pi c_i x_i)] - \sum_{i=k+1}^n \sum_{j=1}^{M_i} \left[\exp(d_i x_i) \exp\left(-\left(\frac{x_i - \frac{j}{M_i}}{\varepsilon}\right)^2\right) \right], \forall 0 \leq x_i \leq 1, \forall i = k+1, \dots, n. \quad \text{Equation 7.11}$$

Equation 7.11 could be simplified to Equation 7.12 to represent problems having a single Pareto front, with no biased region parallel to the Pareto front.

$$I(\vec{x}'') = 2e + x_2, \forall 0 \leq x_2 \leq 1. \quad \text{Equation 7.12}$$

Suppose an interaction function I is required to be constructed for a problem that has 2 variables, with one variable defining the I function. The function I is required to have 5 local Pareto fronts that have low concentration, and 3 deceptive fronts that are far apart from each other. Figure 7.4 depicts this I function. The main features of this function are as follows.

- ◆ Being a 2-variable problem, the parameter n takes a value of 2 in this case. Since one variable defines I , the value of k is 1 ($= 2 - 1$), which is the number of variables that define the D functions.
- ◆ The parameter c_l is given a value of 4 to attain 5 ($= 4 + 1$) local Pareto fronts. The value of b_l is chosen to be 1 to have low concentration of these fronts.
- ◆ The parameter M_l is given a value of 3 to attain 3 deceptive Pareto fronts. The value of d_l equal to 2 makes the deceptive fronts far apart from each other.
- ◆ The value of e is chosen to be 4.38 so that I_{min} takes a value of 1. As mentioned earlier, this simplifies the equation of the Pareto front.
- ◆ Finally, the choice of ε equal to 0.004 is arbitrary.

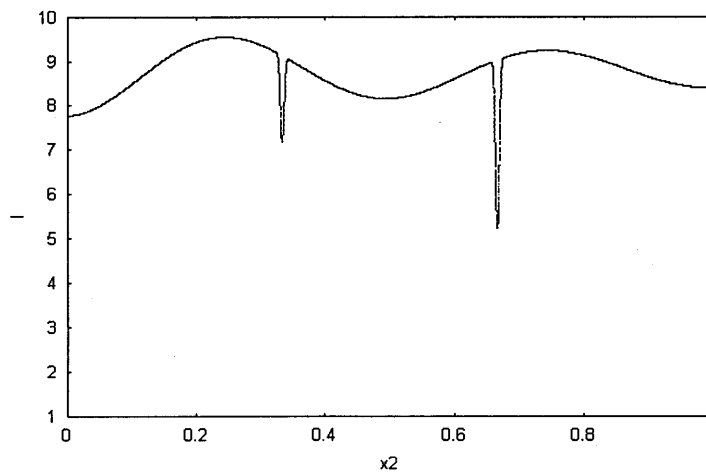


Figure 7.4: An Example of Interaction Function I ($n=2$; $k=1$; $b_l=1$; $c_l=4$; $M_l=3$; $d_l=2$; $\varepsilon=0.004$; $e=4.38$)

7.4.4 Constraint Function (C)

As evident from Table 7.4, constraints can create three types of difficulties for multi-objective optimisation algorithms. The function prototypes for the three types of constraints that individually specialise in simulating each of these difficulties are discussed below.

7.4.4.1 Pareto Blocking Constraints

Constraints can create direct hindrance to convergence by completely blocking the Pareto front. They may also render the Pareto front infeasible, thereby making one of their boundaries as the feasible front for the given optimisation problem. The function prototype for this type of constraints is given in Equation 7.13. The main features of this equation are given below.

- ◆ These constraints constitute bands of infeasible regions that are parallel to the Pareto front. In order to ensure that the boundaries of these regions are parallel to the Pareto front, constant terms are added to the S function to define the boundaries. These constant terms could be used to control the size, distribution and location of infeasible belts with respect to the Pareto front.
- ◆ The number of infeasible belts could also be varied using the parameters in Equation 7.13.

$$C_1 \equiv \left[\begin{array}{l} E_{11} + S(f_1, \dots, f_{M-1}, I_{\min}) \leq f_M \leq \\ E_{21} + S(f_1, \dots, f_{M-1}, I_{\min}) \end{array} \right] \vee \dots \quad \text{Equation 7.13}$$

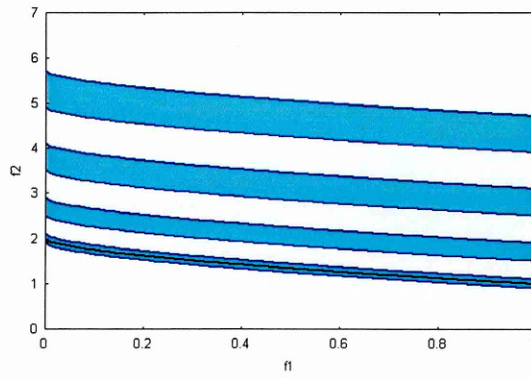
$$\dots \vee \left[\begin{array}{l} E_{1j} + S(f_1, \dots, f_{M-1}, I_{\min}) \leq f_M \leq \\ E_{2j} + S(f_1, \dots, f_{M-1}, I_{\min}) \end{array} \right] \vee \dots$$

$$\dots \vee \left[\begin{array}{l} E_{1J} + S(f_1, \dots, f_{M-1}, I_{\min}) \leq f_M \leq \\ E_{2J} + S(f_1, \dots, f_{M-1}, I_{\min}) \end{array} \right],$$

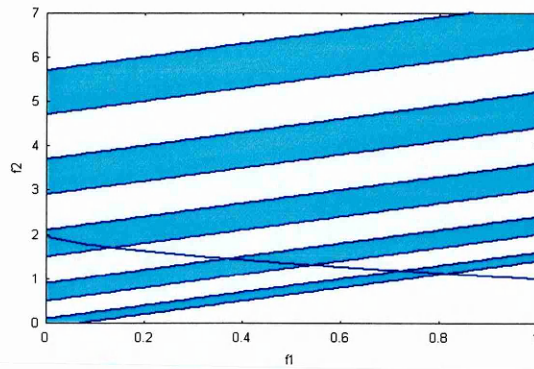
$$\forall 0 \leq f_i \leq f_{i_{\max}}, \forall i = 1, \dots, M-1. -1.$$

The various parameters used in this equation and their significance are discussed below.

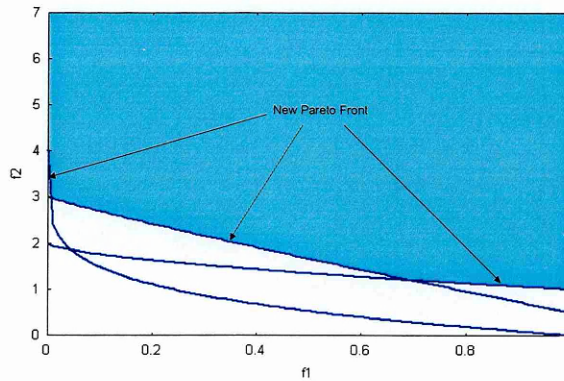
- ◆ M : This is equal to the number of objective functions in the optimisation problem.
- ◆ S Function Parameters: In this C_I equation, all S function parameters are given the same values as used in the problem definition of S . Since the unchanged parameters control the shape of constraint boundaries, this ensures that these boundaries are parallel to the Pareto front. In this way, constraints that completely block the Pareto front are achieved.



(a)



(b)



(c)

Figure 7.5: Examples of Constraint Functions C - (a) Pareto Blocking Constraints (C_1) ($J=4$; $E_{1j}=-0.1, 0.4, 1.5, 2.9$; $E_{2j}= 0.1, 0.9, 2.1, 3.7$; $M=2$; $a_i=0.6$; $c_{1i}=0$; $e=1$) (b) Pareto Intersecting Constraints (C_2) ($J=5$; $m_i=1$; $E_{1j}= -0.1, 0.5, 1.5, 2.9, 4.7$; $E_{2j}=0.1, 0.9, 2.1, 3.7, 5.7$; $M=2$; $a_i=0.6$; $c_{1i}=0$; $e=1$) (c) Composite Constraints (C_3) ($J=2$; $E_j=1, 2$; $a_{ij}=0.9, 0.2$; $P_{ij}=2.5, 4$; $M=2$; $a_i=0.6$; $c_{1i}=0$; $e=1$)

- ◆ E_{1j}, E_{2j} : The choice of appropriate E_{1j} and E_{2j} values enables the design of feasible regions having required size, location and distribution. Higher values of E_{1j} move the constraints away from the original Pareto front, towards higher values of f_N . Further, the more the difference between E_{1j} and E_{2j} , the bigger is the size of the corresponding feasible belt. The difference between E_{2j} and E_{1j+1} also determines the separation between feasible belts. E_{1j} and E_{2j} also control the feasibility under constraints of the original Pareto front.
- ◆ J : The total number of feasible belts in the search space is determined by the parameter J .

Suppose a Pareto blocking constraint that has four feasible belts needs to be constructed. It is required that the Pareto front remains feasible but is blocked by the infeasible belts. Figure 7.5(a) depicts this constraint function. The main features of this function are as follows.

- ◆ Being a 2-objective problem, the value of M is 2 in this case. Since there are 4 feasible belts in this problem, the value of J is 4.
- ◆ In order to ensure that the Pareto front remains feasible, the values of E_{11} and E_{21} are chosen to be -0.1 and 0.1 respectively. The values of other E_{1j} 's (= 0.4, 1.5, 2.9) and E_{2j} 's (= 0.9, 2.1, 3.7) are chosen to make the feasible belts and their separation gradually increase in size as their distance from the Pareto front increases.
- ◆ All S function parameters are given the same values as used in the problem definition of S ($a_1 = 0.6$, $c_{11} = 0$, $e = 1$) to ensure that the constraint boundaries are parallel to the Pareto front.

7.4.4.2 Pareto Intersecting Constraints

The second category of constraints introduces direct hindrance to diversity by making some parts of Pareto front infeasible. This leads to discontinuous Pareto fronts that create diversity problems for the optimisation algorithms. Equation 7.14 gives the function prototype for this type of constraints. The main features of this function prototype are discussed below.

- ◆ The boundaries of the infeasible belts introduced by these constraints are defined using a linear equation that cut away parts of Pareto front using infeasible belts

passing through the front. The coefficients of this equation control the orientation of constraint boundaries with respect to the Pareto front, and its intercept controls the size, location and distribution of infeasible regions.

- ◆ The number of infeasible belts could also be varied using the parameters in Equation 7.14.

$$\begin{aligned}
 C_2 \equiv & \left[E_{11} + \sum_{i=1}^{M-1} m_i f_i \leq f_M \leq E_{21} + \sum_{i=1}^{M-1} m_i f_i \right] \vee \dots & \text{Equation 7.14} \\
 & \dots \vee \left[E_{1j} + \sum_{i=1}^{M-1} m_i f_i \leq f_M \leq E_{2j} + \sum_{i=1}^{M-1} m_i f_i \right] \vee \dots \\
 & \dots \vee \left[E_{1J} + \sum_{i=1}^{M-1} m_i f_i \leq f_M \leq E_{2J} + \sum_{i=1}^{M-1} m_i f_i \right], \\
 & \forall 0 \leq f_i \leq f_{i\max}, \forall i = 1, \dots, M - 1.
 \end{aligned}$$

The parameters used in this equation have the following significance.

- ◆ M : This is equal to the number of objective functions in the optimisation problem.
- ◆ m_i : This is equal to the slope of constraint boundaries with respect to f_i . m_i 's should be chosen in such a way that the constraint boundaries have required orientation with respect to the Pareto front. An easy way of ensuring this is to determine the ends of the Pareto front using its equation (Equation 7.10). This information can now be used to determine m_i 's such that the constraint boundaries have required orientation.
- ◆ J : This is equal to the total number of feasible belts in the function search space.
- ◆ E_{1j}, E_{2j} : These parameters control the location of constraint boundaries corresponding to the j^{th} feasible belt. Higher values of E_{1j} move the constraints towards higher values of f_N . Further, the more the difference between E_{1j} and E_{2j} , the bigger is the size of the corresponding feasible belt. The difference between E_{2j} and E_{1j+1} also determines the separation between feasible belts. In this way, E_{1j} and E_{2j} control the size, location and distribution of feasible regions.

Suppose it is needed to construct a Pareto interesting constraint that introduces 5 feasible belts. It is required to create discontinuities in the unconstrained Pareto front using the separation between these feasible belts. Figure 7.5(b) depicts this constraint function. The main features of this function are as follows.

- ◆ Being a 2-objective problem, the value of M is 2 in this case. Since there are 5 feasible belts in this problem, the value of J is 5.
- ◆ The values of E_{1j} 's (= -0.1, 0.5, 1.5, 2.9, 4.7), E_{2j} 's (= 0.1, 0.9, 2.1, 3.7, 5.7) and m_j (= 1) are chosen such that the constraint boundaries intersect the Pareto front, thereby introducing the discontinuities. The choice of these parameters also ensures that the feasible belts and their separation gradually increase in size as their distance from the Pareto front increases.

7.4.4.3 Composite Constraints

There are also cases when the constraint boundaries intersect the original Pareto front in such a way that the new front becomes a combination of original front and constraint boundaries. Due to this composite nature of the constraints, maintenance of diversity becomes a problem. Equation 7.15 gives the function prototype for this type of constraints. The main features of this function prototype are listed below.

- ◆ This C_3 function takes a form that is similar to the shape function S , thereby enabling better control over the intersection of the constraint boundaries with the unconstrained Pareto front. This function also provides parameters to control the shape, cyclical nature and location (with respect to the original Pareto front) of the constraint boundaries.
- ◆ The number of constraints could also be varied using the parameters in Equation 7.15.

The parameters used in this equation provide the following control.

- ◆ M : This is equal to the total number of objective functions in the optimisation problem.
- ◆ E_j : This parameter influences the location of the j^{th} constraint boundary with respect to the original Pareto front. Higher values of E_j move the constraint boundary away from the original Pareto front, towards higher values of f_N .
- ◆ J : This is equal to the total number of feasible holes in the function search space.
- ◆ P_{ij} , a_{ij} : These parameters control the location and shape of the j^{th} constraint boundary with respect to the i^{th} objective function. Decreasing the value of a_{ij} and increasing the value of P_{ij} both have the influence of pulling the constraint boundary away from the original Pareto front, towards lower values of f_N .

Increasing P_{ij} also makes the constraint boundary steeper. Finally, the difference between a_{ij} and a_i (a parameter in the equation of original Pareto front – Equation 7.10) explains the dissimilarities in general shapes of Pareto front and the constraint boundary.

- ◆ c_{li}, b_i, c_i, d_i, e : The first four of these parameters control the cyclical nature of the constraint boundary and the fifth controls its general location. By giving these parameters the same values as those used in the problem for the corresponding parameters of shape function S , it can be ensured that the constraint boundary has the same cyclical nature and general location as the Pareto front.

$$\begin{aligned}
 C_3 \equiv & \left[\begin{array}{l} f_M \geq E_1 + 2(M-1)e - \\ \sum_{i=1}^{M-1} \left[P_{i1} \left(\frac{f_i}{f_{i\max}} \right)^{a_{i1}} + c_{li} \left(\frac{f_i}{f_{i\max}} \right)^{b_i} \right] \\ \cos \left(2\pi c_i \left(\frac{f_i}{f_{i\max}} \right)^{d_i} \right) \end{array} \right] \wedge \dots & \text{Equation 7.15} \\
 \dots \wedge & \left[\begin{array}{l} f_M \geq E_j + 2(M-1)e - \\ \sum_{i=1}^{M-1} \left[P_{ij} \left(\frac{f_i}{f_{i\max}} \right)^{a_{ij}} + c_{li} \left(\frac{f_i}{f_{i\max}} \right)^{b_i} \right] \\ \cos \left(2\pi c_i \left(\frac{f_i}{f_{i\max}} \right)^{d_i} \right) \end{array} \right] \wedge \dots \\
 \dots \wedge & \left[\begin{array}{l} f_M \geq E_j + 2(M-1)e - \\ \sum_{i=1}^{M-1} \left[P_{ij} \left(\frac{f_i}{f_{i\max}} \right)^{a_{ij}} + c_{li} \left(\frac{f_i}{f_{i\max}} \right)^{b_i} \right] \\ \cos \left(2\pi c_i \left(\frac{f_i}{f_{i\max}} \right)^{d_i} \right) \end{array} \right], \\
 & \forall 0 \leq f_i \leq f_{i\max}, \forall i = 1, \dots, M-1.
 \end{aligned}$$

Suppose it is required to construct a composite constraint that is composed of 2 constraints leading to a new Pareto front, which is made up of a combination of the original Pareto front and the two constraint boundaries. Figure 7.5(c) depicts this constraint function. The main features of this function are as follows.

- ◆ Being a 2-objective problem, the value of M is 2 in this case. Since there are 2 constraints in this problem, the value of J is 2.
- ◆ The values of e ($= 1$), P_{ij} ($= 2.5, 4$), a_{ij} ($= 0.9, 0.2$) and E_j ($= 1, 2$) are chosen such that the constraint boundaries intersect the Pareto front, thereby leading to a new Pareto front that is a combination of the original Pareto front and the two constraint boundaries. The other parameters in Equation 7.15 take the same values as those in the problem definition of S ($a_I = 0.6, c_{II} = 0$).

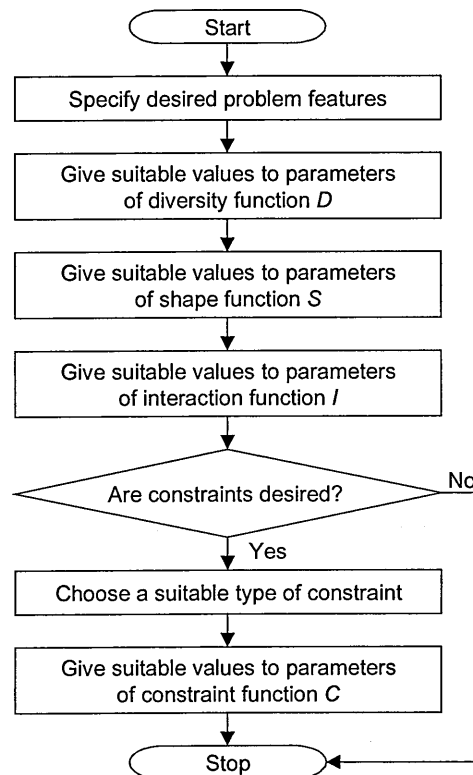


Figure 7.6: Steps for Using Reverse Engineered Test Bed (RETB)

7.5 Guidelines for Use of RETB

The following steps can be used as guides for using RETB to construct a test problem with desired features. Figure 7.6 presents a flow chart that can be used to design test problems using RETB.

- ◆ Step 1: Specify the problem features in terms of the desired complexity of Pareto front, relationship(s) among decision variables of the Pareto-optimal solutions, hindrance to diversity and nature of constraints.
- ◆ Step 2: Give suitable values to the parameters of the diversity function D (Equation 7.7, Equation 7.8), based on the problem features defined in Step 1.
- ◆ Step 3: Repeat the above process for the shape function S (Equation 7.9).
- ◆ Step 4: Also repeat the above process for the interaction function I (Equation 7.11, Equation 7.12).
- ◆ Step 5: If a constrained optimisation problem is desired, choose a suitable type of constraint function C (Pareto blocking - Equation 7.13, Pareto intersecting - Equation 7.14 or composite - Equation 7.15), and give appropriate values to its parameters based on the nature of constraints desired in Step 1.

To further facilitate the development of test problems using RETB, a pictorial representation of the proposed test bed is given in Figure 7.7. This figure presents a tree diagram that facilitates the stepwise selection of RETB functions and parameters, based on pre-defined problem features. It should be noted that Equation 7.16, mentioned in this figure, refers to the function prototypes for single objective optimisation problems. This equation is developed as a special case of the prototypes proposed in the previous section.

$$f(x_1, \dots, x_n) = 2ne - \sum_{i=1}^n \left[(x_i)^{a_i} + c_{i_i} (x_i)^{b_i} \cos(2\pi c_i (x_i)^{d_i}) \right] \quad \text{Equation 7.16}$$

$$- \sum_{i=1}^n \sum_{j=1}^{M_i} \left[\exp(g_i x_i) \exp\left(- \left(\frac{x_i - p_{ij}}{\varepsilon} \right)^2 \right) \right], \forall 0 \leq x_i \leq 1, \forall i = 1, \dots, n,$$

$$C_j(x_1, \dots, x_n) \geq 0, \forall j = 1, \dots, J.$$

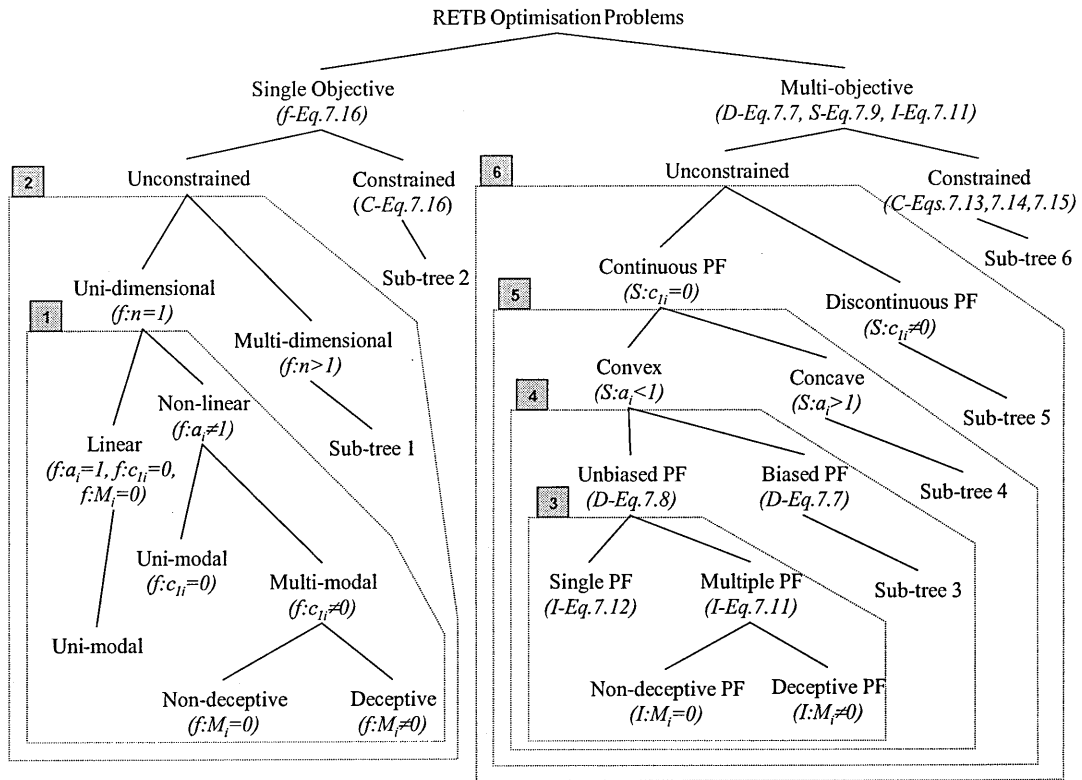


Figure 7.7: Tree Diagram for Constructing RETB Optimisation Problems (PF: Pareto Front, Highlighted Numbers: Re-useable Sub-trees)

7.6 RETB Case Studies

This section illustrates the use of the above-mentioned guidelines to select RETB parameters for constructing three multi-objective optimisation problems, having pre-defined characteristics. The three test cases are chosen such that they form a representative set of the characteristics of multi-objective optimisation problems. Table 7.6 lists the desired features of these test cases. As can be seen from this table, the test cases thus created incorporate a variety of features of multi-objective optimisation problems: discontinuous Pareto front, inherent bias on Pareto front, interaction among variables corresponding to the Pareto-optimal solutions (leading to multiple local fronts), and constraints.

The parameters in S , D , I and C functions of these test cases are determined based on their pre-defined features, as listed in Table 7.6. Table 7.7 gives the RETB parameter

values for the three test cases. The reasons behind the selection of these parameter values are also stated in this table. Further, Figure 7.8, Figure 7.9 and Figure 7.10 present pictorial representations of these test cases by depicting the results of exhaustive search applied on them. These figures aid in visual validation of the test functions by confirming that their characteristics conform to the pre-defined problem features (Table 7.6), which guided their development. To support the process of validation, Figure 7.8, Figure 7.9 and Figure 7.10 also show the global Pareto fronts.

Table 7.6: Features of RETB Test Cases

RETB Case Studies	Case-1	Case-2	Case-3
Problem Features	<ul style="list-style-type: none"> • Two objectives • Convex and continuous S (Pareto front) • Biased D (biased region on Pareto front) • Multi-front (multiple local Pareto fronts) • No constraints 	<ul style="list-style-type: none"> • Two objectives • Convex and discontinuous S (Pareto front) • Biased D (biased region on Pareto front) • Multi-front (multiple local Pareto fronts) • No constraints 	<ul style="list-style-type: none"> • Two objectives • Convex and continuous S (Pareto front) • Biased D (biased region on Pareto front) • Multi-front (multiple local Pareto fronts) • Pareto blocking constraints with infeasible Pareto front

Table 7.7: RETB Parameters Values for Test Cases – (a) Case-1 (b) Case-2 (c) Case-3

(a)

Case-1	S			D			I		
	Parameter Values		Reasoning	Parameter Values		Reasoning	Parameter Values		Reasoning
	M	2	2 objectives	k	1	1 variable defining D	n	2	2 variable-problem
a_i	0.6	< 1 for convex Pareto front	M_i	1	1 biased region on Pareto front	k	1	1 variable defining D	
c_{1i}	0	To attain continuous Pareto front	T	1	Product of M_i 's	b_i	2	Arbitrary	
b_i	NA	-	a_{ij}	1	Arbitrary	c_i	8	(8+1)=9 local Pareto fronts	
c_i	NA	-	b_{ij}	4	Arbitrary	M_i	0	No deceptive front	
d_i	NA	-	R_i	0	So that f_{1max} is 1	d_i	NA	-	
e	1	For simplicity	a_k	NA	-	ε	NA	-	
						e	1	So that l_{min} is 1	

Table 7.7: RETB Parameters Values for Test Cases – (a) Case-1 (b) Case-2 (c) Case-3 (contd.)

(b)

Case-2	S			D			I		
	Parameter Values		Reasoning	Parameter Values		Reasoning	Parameter Values		Reasoning
	M	2	2 objectives	k	1	1 variable defining D	n	3	3 variable-problem
a_i	0.4	< 1 for convex Pareto front	M_i	1	1 biased region on Pareto front	k	1	1 variable defining D	
c_{ii}	1	To attain discontinuous Pareto front	T	1	Product of M_i 's	b_i	1,1	Arbitrary	
b_i	1	Arbitrary	a_{ij}	1	Arbitrary	c_i	2,4	(2+1)x(4+1)=15 local Pareto fronts	
c_i	4	(4+1)=5 disconnected Pareto regions	b_{ij}	3	Arbitrary	M_i	0	No deceptive front	
d_i	2	Arbitrary	R_i	0	So that f_{1max} is 1	d_i	NA	-	
e	1	For simplicity	a_k	NA	-	ϵ	NA	-	
						e	1.5	So that I_{min} is 1	

(c)

Case 3	S			D			I			C		
	Par. Vals.		Reasoning	Par. Vals.		Reasoning	Par. Vals.		Reasoning	Par. Vals.		Reasoning
	M	2	2 objectives	k	1	1 variable defining D	n	2	2 variable-problem	J	4	4 feasible belts
a_i	0.6	< 1 for convex Pareto front	M_i	1	1 biased region on Pareto front	k	1	1 variable defining D	E_{1j}	0.3, 0.9, 1.9, 3.3	To make Pareto front infeasible, and attain required size and separation of constraint boundaries	
c_{ii}	0	To attain continuous Pareto front	T	1	Product of M_i 's	b_i	2	Arbitrary	E_{2j}	0.5, 1.3, 2.5, 4.1		
b_i	NA	-	a_{ij}	1	Arbitrary	c_i	8	(8+1)=9 local Pareto fronts	M	2	2 objectives	
c_i	NA	-	b_{ij}	4	Arbitrary	M_i	0	No deceptive front	a_i	0.6	Same values as in S (to have constraint boundaries parallel to Pareto front)	
d_i	NA	-	R_i	0	So that f_{1max} is 1	d_i	NA	-	c_{ii}	0		
e	1	For simplicity	a_k	NA	-	ϵ	NA	-	b_i	NA		
						e	1	So that I_{min} is 1	c_i	NA		
									d_i	NA		
									e	1		

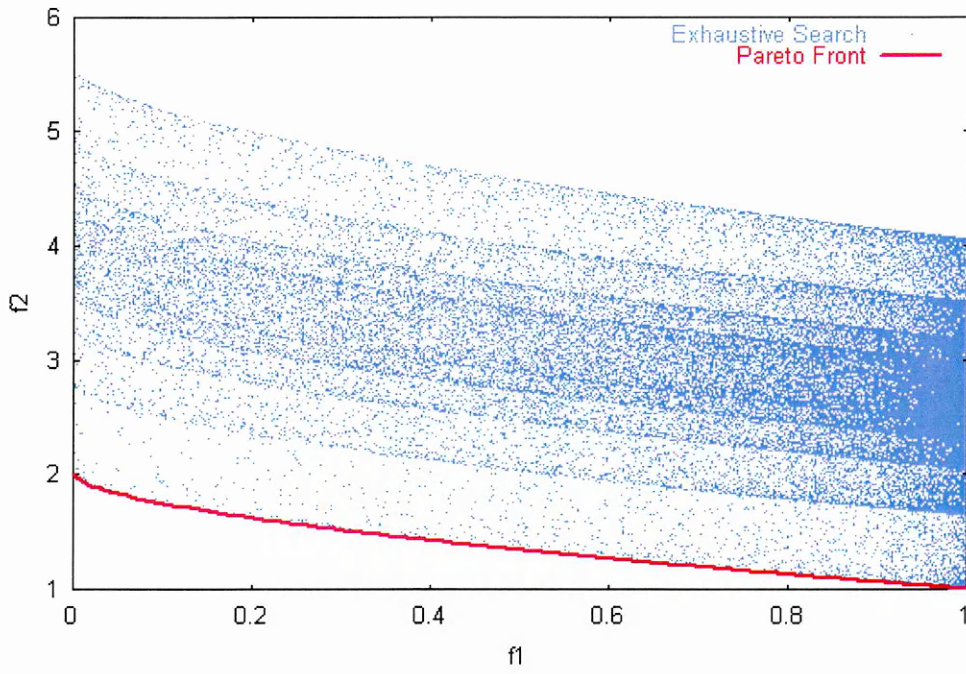


Figure 7.8: Search Space of RETB Case-1

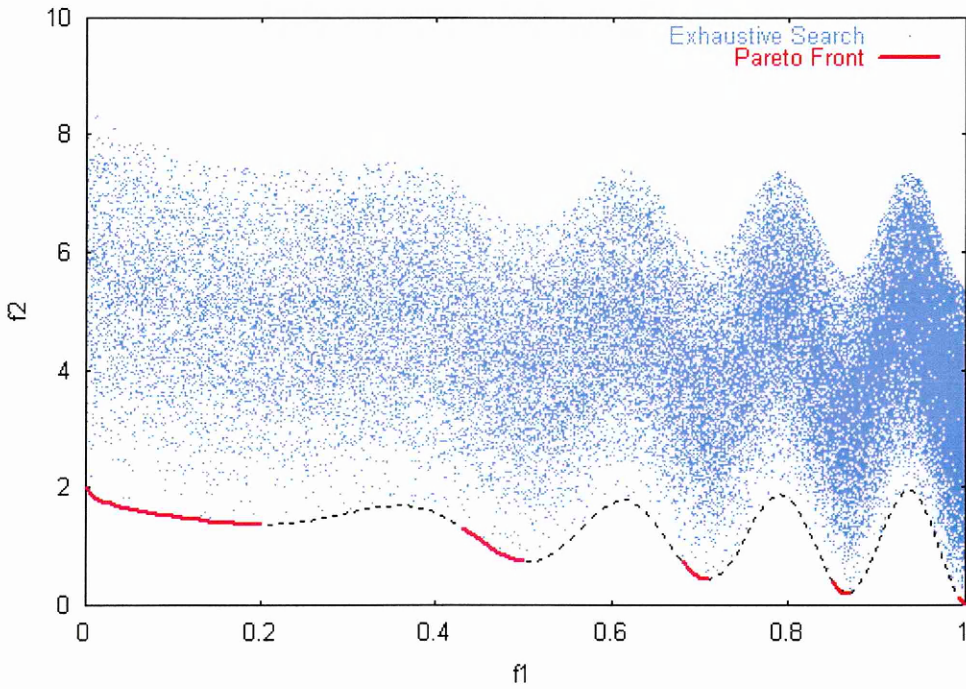


Figure 7.9: Search Space of RETB Case-2

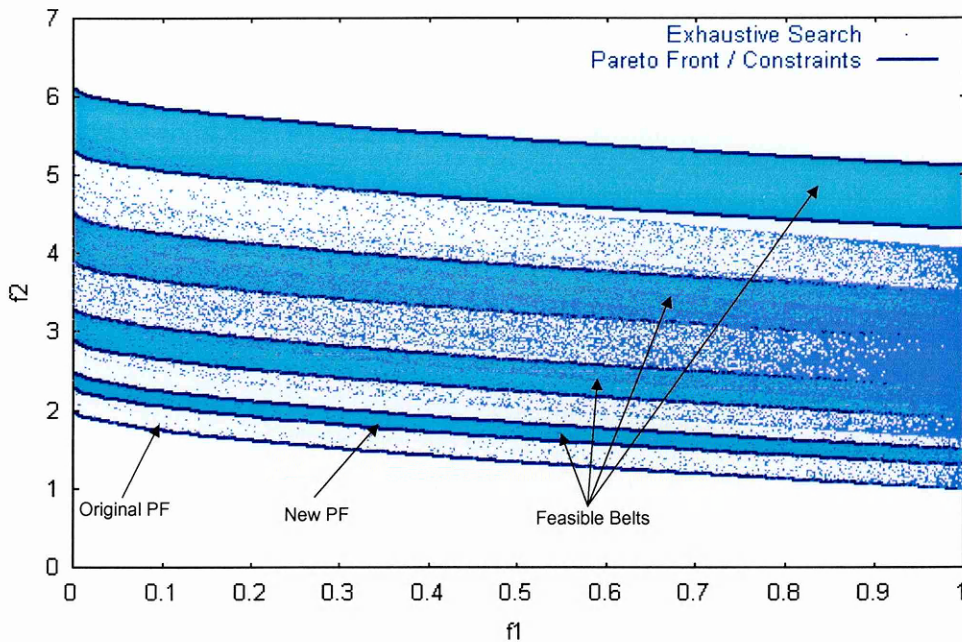


Figure 7.10: Search Space of RETB Case-3

7.7 RETB-II: Extension of RETB to Incorporate Variable Dependence

As mentioned in Chapter 4, there is a complete lack of test problems in literature that can simulate dependent-variable multi-objective optimisation problems. This section attempts to extend RETB to enable it to carry out controlled testing of the performance of optimisation algorithms in the presence of variable dependence. Hence, the test bed that is proposed in this section is given the name Reverse Engineered Test Bed – II (RETB-II). Since variable dependence is defined by the nature of variables rather than by that of the objective functions and constraints, it can be treated independently of them. Therefore, RETB simulates the complexities that are introduced by objective functions and constraints, whereas RETB-II extends RETB to include the complexities that are introduced by the nature of variables. In this way, the parametric equations defined for RETB are also valid for RETB-II, with the addition of equations that are presented in this section for modelling variable dependence.

In order to perform controlled simulation, RETB-II should have parameters that represent the challenges that variable dependence poses for optimisation algorithms. These challenges, which are outlined in Table 7.5, form the basis for the development of RETB-II, as discussed in this section.

7.7.1 Number of Dependency Relationships

Since RETB-II utilises explicit equations to model dependence among decision variables, the number of dependency relationships is equal to the number of dependent variables desired in the problem (N_d). Since the bounds on dependent variables are treated as constraints, increasing the value of N_d has an effect of making the search space more constrained. This makes it more difficult for an optimisation algorithm to work in the feasible region, as defined by variable dependence, and locate optimal solutions. To avoid cyclic dependencies and to maintain the consistency of dependent and independent variables, the DC is used while defining the dependency equations. The DT can also be used to visualise the relationship among variables.

7.7.2 Nature of Dependency Relationships

In this test bed, the nature of dependency relationships is controlled by the dependency equations. The following parametric equation (Equation 7.17) can be used, in conjunction with the DC/DT, to define the dependency relationships. The basic form of these prototypes is derived from existing test functions in optimisation, and from known equations in other areas of research. These basic forms are customised here based on the specific requirements of this research. The main features of Equation 7.17 are given below.

- ◆ This function is provided with a polynomial function that directly controls its degree of non-linearity. In this way, the bias in the search space is controlled.
- ◆ This function also has a cosine function that introduces multiple fronts in the search space and produces multiple basins of attraction across the Pareto front.

- ◆ The parameter of the cosine function has two parts. The first part is a constant term that controls the number of multiple fronts and basins of attraction in the search space. The second part that is a function of x_{ij} controls the distribution of the multiple fronts and basins of attraction.
- ◆ The cosine function is multiplied by another function of x_{ij} that also has an effect on the distribution of the multiple fronts and basins of attraction.
- ◆ The cosine function has a multiplier whose value can be set to 0 in order to switch-off its effects.
- ◆ Similarly, deception is introduced in the search space using an exponential function that exhibits sudden drop at a given value of decision variable, but remains zero otherwise.
- ◆ The above-mentioned function is multiplied by another exponential function that prevents the deceptive regions from coinciding with each other. The parameter of this exponential function controls the concentration of deceptive regions in the search space. Equation 7.17 also provides parameters to control the number of deceptive fronts.
- ◆ This exponential function has a multiplier whose value can be set to 0 in order to switch-off its effects.
- ◆ Equation 7.17 also provides the facility of sub-dividing the search space that allows different dependency equations to be defined for each of the sub-spaces. In this way, discontinuity can be introduced in the optimisation problem, and its nature can be controlled using the parameters provided in Equation 7.17.
- ◆ Equation 7.17 also provides the parameters for controlling the general location of the dependency relationship in the search space.

Here, the DC/DT should be used such that each x_{di} and x_{ij} correspond to a given decision variable in the problem (x_1, \dots, x_n) , where n is the total number of decision variables. In this case, x_{di} 's always correspond to different dependent variables, but x_{ij} 's may correspond to the same independent variable. It should be noted that the sequence of independent variables has an influence on those GAs whose performance is dependent on variable sequence. Hence, sequencing of variables may be controlled to influence the difficulty for some optimisation algorithms. With the increase in separation between the dependencies of a variable in the GA

chromosome, the difficulty of a problem increases due to the enhanced probability of the ‘good’ building blocks being broken by the recombination operators (Pelikan *et al.*, 1999).

$$x_{di} = \text{Function}(x_{ij}), \forall 0 \leq x_{ij} \leq 1 \quad \text{Equation 7.17}$$

$$x_{dik} = 2N_{ind,i}e_{ik} + P_{ik}(x_{ij}) - \sum_{j=1}^{N_{ind,i}} \left[c_{1ijk} (x_{ij})^{b_{jk}} \cos^2 \left(2\pi c_{ijk} (x_{ij})^{d_{jk}} \right) \right] - \sum_{j=1}^{N_{ind,i}} c_{2ijk} \sum_{l=1}^{M_{ijk}} \left[\exp(g_{ijk} x_{ij}) \exp \left(- \left(\frac{x_{ij} - P_{ijlk}}{\varepsilon} \right)^2 \right) \right], \forall x_{ij} \in SVSS_{ik},$$

$$\forall i = 1, \dots, N_d, \forall j = 1, \dots, N_{ind,i}, \forall k = 1, \dots, Q_i,$$

$$VSS_i = \bigcup_{k=1}^{Q_i} SVSS_{ik},$$

$$\phi = \bigcap_{k=1}^{Q_i} SVSS_{ik},$$

$$x_{di} = \text{Dependent_Variables}(i = 1, \dots, N_d),$$

$$x_{dik} = \text{Function_Definition_of_}i^{\text{th}}\text{_Dependent_Variable_in_}k^{\text{th}}\text{_}SVSS_i,$$

$$x_{ij} = \text{Independent_Variables_Defining_}i^{\text{th}}\text{_Dependent_Variable}(i = 1, \dots, N_d; j = 1, \dots, N_{ind,i}).$$

The parameters used in the above-mentioned equation and their significance are discussed below.

- ◆ n : This is equal to the total number of decision variables in the problem.
- ◆ N_i : This is equal to the total number of independent variables in the problem. Higher N_i implies greater complexity in the problem.
- ◆ $N_{ind,i}$: This is equal to the number of independent variables that define the i^{th} dependent variable. The complexity of a dependency equation increases with the increase in the value of this parameter.
- ◆ N_d : This parameter is equal to the number of dependent variables in the problem. Higher N_d implies greater complexity in the problem.
- ◆ Q_i : It is the total number of parts into which the variable space, corresponding to the i^{th} dependent variable, is sub-divided. Since the function parameters may take different values for each part of the variable space, Q_i influences the total number

of disconnected regions in the Pareto front. Higher values of this parameter indicate higher numbers of disconnected regions in the function search space, and hence more complexity.

- ◆ $SVSS_{ik}$: This indicates a Sub-set of Variable Search Space (SVSS) of the problem for the i^{th} dependent variable. The subscript k is an index for numbering the sub-sets. Since the problem has Q_i sub-sets for the i^{th} dependent variable, the value of k ranges from 1 to Q_i . Further, all $SVSS_{ik}$'s are mutually exclusive. This makes their intersection a null set. Also, the union of all $SVSS_{ik}$'s gives the whole Variable Search Space (VSS) for the i^{th} dependent variable. The way in which $SVSS_{ik}$'s are designed controls the location, size and distribution of disconnected regions in the function search space.
- ◆ VSS_i : It indicates the Variable Search Space for the i^{th} dependent variable. Its relevance to the test bed development is described in the discussion for $SVSS_{ik}$.
- ◆ P_{ik} : This is a polynomial that is defined in terms of x_{ij} 's. It directly influences the general shape of a dependency relationship (in $SVSS_{ik}$) with respect to x_{ij} 's. The degree of P_{ik} influences the complexity of this relationship. The parameters P_{ik} 's control the number, location and extent of bias in the function search space.
- ◆ b_{ijk} : It determines the height of cosine waves (in $SVSS_{ik}$) with respect to the decision variable x_{ij} . In this way, it controls the distribution of optimal values in the dependency relationship. A higher b_{ijk} implies more variations in these values.
- ◆ c_{ijk} : It is equal to the number of cosine waves (in $SVSS_{ik}$) in the range of x_{ij} . This gives the number of optima in the dependency relationship. Higher values of this parameter increase the complexity of the dependency relationship.
- ◆ c_{1ijk} : It is a flag to indicate whether the dependency relationship is multi-modal (in $SVSS_{ik}$) with respect to x_{ij} . c_{1ijk} takes a value of 0 for uni-modal equations and a value of 1 for multi-modal equations.
- ◆ c_{2ijk} : Similarly, c_{2ijk} is a flag to indicate whether the dependency relationship (in $SVSS_{ik}$) is deceptive with respect to x_{ij} . c_{2ijk} takes a value of 0 for non-deceptive equations and a value of 1 for deceptive equations.
- ◆ d_{ijk} : This parameter influences the spacing between consecutive cosine waves in the dependency relationship. Therefore, it directly controls the distribution of points that correspond to the optimal values of the dependent variables (in $SVSS_{ik}$), with higher values of d_{ijk} leading to closer points.

- ◆ e_{ik} : It decides the general location of the dependency equation. Higher values of e_{ik} push the equation towards higher values of the dependent variable (in $SVSS_{ik}$). e_{ik} should be chosen in such a way that the dependency equation takes only positive values.
- ◆ M_{ijk} : This is equal to the number of deceptive optima (in $SVSS_{ik}$) corresponding to the variable x_{ij} . Higher values of this parameter lead to higher complexity.
- ◆ g_{ijk} : This parameter controls the height of deceptive exponential function (in $SVSS_{ik}$) corresponding to the decision variable x_{ij} . Since deceptive optima correspond to the maximum values of these functions, the parameter g_{ijk} controls the values of the dependent variable corresponding to the deceptive optima. A higher value of g_{ijk} implies that the deceptive optimal values are farther away from each other.
- ◆ p_{ijkl} : This parameter gives the value of the decision variable x_{ij} (in $SVSS_{ik}$) corresponding to its l^{th} deceptive optimum. Using these parameters, the location of the points that lead to deceptive optima can be controlled.
- ◆ ε : This is a small positive number used with the deceptive exponential function to ensure that it exhibits a sudden drop at a given value of the decision variable.

7.7.3 Nature of Available Information

As mentioned in Chapter 2, there are two categories of dependent variable optimisation problems: with and without dependency equations. The above discussion proposed a test bed that simulates those dependent variable optimisation problems in which the dependency equations are known. However, in many real-life problems, these equations are not known explicitly. In these problems, multiple sets of measured variable values are known from which the optimisation algorithms need to infer the dependency relationships. This adds another dimension to the challenges posed by these problems. In order to simulate this feature, Equation 7.17 is modified to get the following equation (Equation 7.18).

$$x_{di}' = x_{di} + Normal(\mu, \sigma^2), \forall i = 1, \dots, N_d. \quad \text{Equation 7.18}$$

In this case, the data for the independent variables are generated in a random fashion whereas those for the dependent ones are obtained using the above equation. In this

equation, the second term is added to introduce an error in the data generated by the previously mentioned equation for x_{di} . Since x_{di}' simulates the data that is measured/calculated in real-life, the introduction of this term makes the data more realistic. The parameters used in the above equation and their significance are discussed below.

- ◆ N_d : This is the number of dependent variables in the problem. Higher N_d implies greater complexity in the problem.
- ◆ x_{di}' : This gives the data corresponding to the i^{th} dependent variable.
- ◆ x_{di} : This defines the nature of the dependency relationship for the i^{th} dependent variable. The value of this term is obtained using the equation that was proposed earlier for x_{di} . The parameters in this equation are chosen based on the desired behaviour of the dependency relationships.
- ◆ $Normal(\mu, \sigma^2)$: This term is used to introduce noise in the dependency relationships. This noise is normally distributed with mean μ and variance σ^2 (σ is the standard deviation). This term could also be made to follow another continuous variable distribution, such as Uniform, Triangular, Exponential, Lognormal, Gamma, Weibull, Beta, Geometric, Negative Binomial, Hypergeometric, Logistic, Pareto, Extreme Value and Gaussian. A value of μ equal to 0 keeps the data centred at the relationship defined by the equation for x_{di} . Any change in μ from 0 moves the data away from this equation. Similarly, the parameter σ influences the distribution of data around the equation for x_{di} . Higher values of σ imply wider distribution around this equation. An increase in the absolute values of μ and σ make it more difficult for the optimisation algorithm to predict the real underlying relationship among the decision variables. This increases the complexity of the optimisation problem.

7.8 Guidelines for Use of RETB-II

The following simple steps are guides for using RETB-II to construct test problems with desired features. Figure 7.11 presents a flow chart that can be used to design test problems using RETB-II.

- ◆ Step I: Fix the number of dependent (N_d) and independent (N_i) variables in the problem. The sum of these two parameters gives the total number of variables in

the problem (n). Decide the relative sequence of these variables based on the desired complexity.

- ◆ Step II: Identify the characteristics that are desired in the dependency relationships. Use these in conjunction with DC/DT to choose the parameters in the equations for x_{di} 's.
- ◆ Step III: To design a problem in which multiple sets of variable values define the dependency relationships, use the equation for x_{di} ' to generate the data. The mean and distribution of noise around the relationship defined by x_{di} are controlled using the appropriate parameters provided in the equation for x_{di} '.

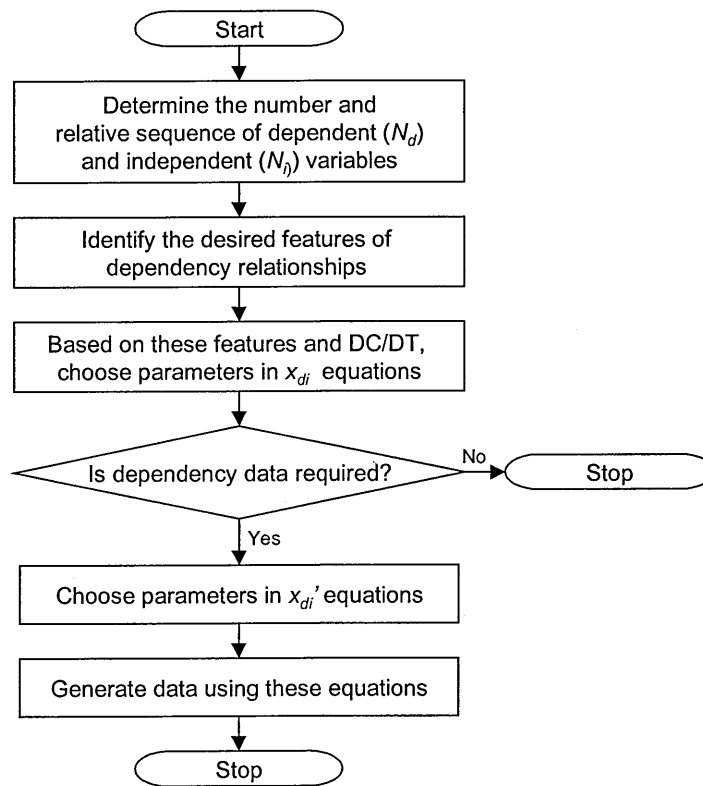


Figure 7.11: Steps for Using Reverse Engineered Test Bed – II (RETB-II)

7.9 RETB-II Case Studies

In this section, four RETB-II case studies are presented; each of which specialises in a particular challenge that variable dependence poses for optimisation algorithms. In this way, these examples illustrate all the challenges that are listed in Table 7.5. The characteristics of variable dependence associated with these examples are as follows.

- ◆ Example 1: Biased problem.
- ◆ Example 2: Multi-front problem.
- ◆ Example 3: Deceptive problem.
- ◆ Example 4: Problem with discontinuous function search space.

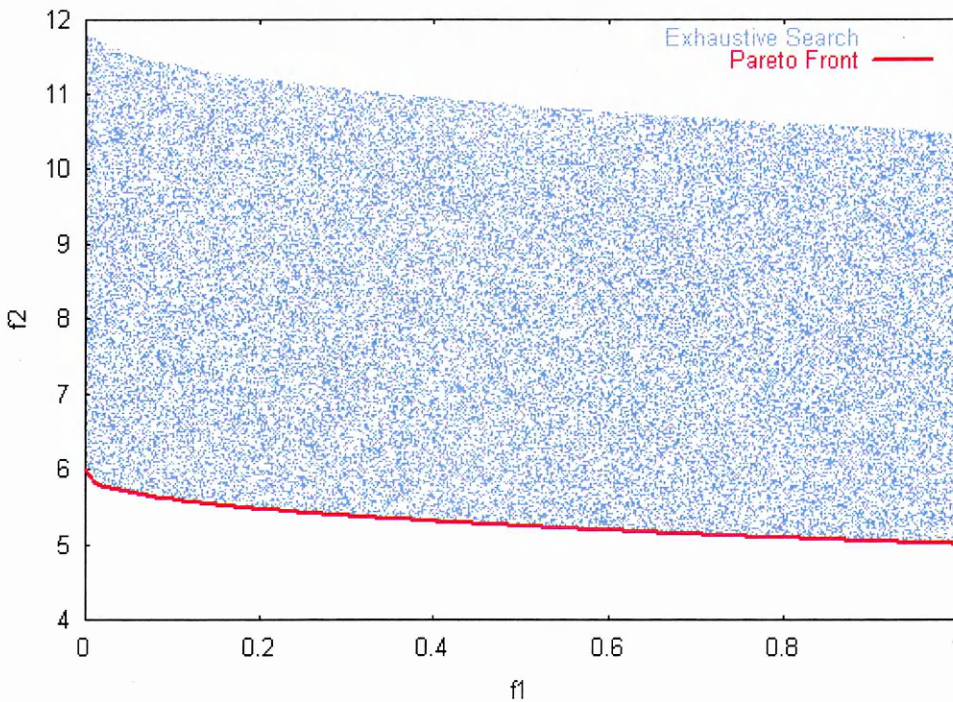


Figure 7.12: Original Search Space for RETB-II Examples 1, 2, 3 & 4 (Assuming Independent Variables)

For these examples, the objective functions are defined using RETB. In order to concentrate on the challenges introduced by variable dependence, the RETB parameters are chosen in such a way that the objective functions (without variable dependence) give a simple problem that has no bias and multi-modality (Equation 7.19). The main features of this independent-variable problem and the corresponding objective functions are as follows. The search space for this problem is also given in Figure 7.12.

- ◆ Two objectives.
- ◆ Convex and continuous S /Pareto front.
- ◆ Unbiased D (uniform Pareto front).

- ◆ Uni-modal I (single Pareto front).
- ◆ No constraints.

$$\begin{aligned}
 D(\bar{x}') &= 1 - x_1, \forall 0 \leq x_1 \leq 1, \\
 I(\bar{x}'') &= 2 - x_2, \forall 0 \leq x_2 \leq 1, \\
 s(f_1, I) &= 6 - (f_1 / I)^{0.4}, \\
 f_1 &= D(\bar{x}'), \\
 f_2 &= s(f_1, I) \times I(\bar{x}''), \\
 \text{Pareto_front} &\Rightarrow f_2 = 6 - (f_1)^{0.4}, \forall 0 \leq f_1 \leq 1.
 \end{aligned}
 \tag{Equation 7.19}$$

7.9.1 Example 1: Biased Problem

A non-linear variable dependence is introduced in the above problem to attain a biased search space. The steps that are given below for the development of this test problem are based on the guidelines presented in the previous section.

7.9.1.1 Step 1

Here, the variable x_2 is assumed to be dependent on two independent variables (x_3 and x_4) that are introduced in this problem. The variable x_1 is also assumed to be independent. This gives a value of 1 to $N_d(x_2)$, a value of 3 to $N_i(x_1, x_3, x_4)$ and hence a value of 4 to $n(N_d + N_i)$. To concentrate on the effects of the dependency equation, the two dependencies of $x_2(x_3$ and $x_4)$ are arranged close to each other in the GA chromosome. This is attained by placing the variables in the order of their indices.

7.9.1.2 Step 2

In order to attain a biased search space, a non-linear dependency equation needs to be developed for x_2 in terms of x_3 and x_4 . This guides the selection of parameters for the required dependency equation, as shown in Table 7.8. This equation is given below (Equation 7.20) and is also graphically represented in Figure 7.13.

$$\begin{aligned}
 x_2 &= 1 - 0.1x_3 - 0.2x_3^2 - 0.3x_4 - 0.1x_4^2 - 0.3x_3x_4, \\
 \forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1.
 \end{aligned}
 \tag{Equation 7.20}$$

Table 7.8: Parameter Values for RETB-II Examples – (a) Example 1: Biased Problem and Example 2: Multi-front Problem (b) Example 3: Deceptive Problem and Example 4: Discontinuous Problem

(a)

		Example 1: Biased Problem		Example 2: Multi-front Problem		
		Parameter Values	Reasoning	Parameter Values	Reasoning	
Eg. 1 & Eg. 2	N_d	1	1 dependent variable	N_d	1	1 dependent variable
	$N_{ind,i}$	2	2 variables defining dependent variable	$N_{ind,i}$	2	2 variables defining dependent variable
	N_i	3	3 independent variables	N_i	3	3 independent variables
	n	4	4-variable problem	n	4	4-variable problem
	Q_i	1	To attain continuous function search space	Q_i	1	To attain a continuous function search space
	P_{ik}	Degree 2	To get a non-linear dependency equation; coefficients are chosen to attain minimum value of 0	P_{ik}	Degree 2	To get a non-linear dependency equation; coefficients are chosen to attain minimum value of 0.2
	c_{1ijk}	0	No cyclic terms	c_{1ijk}	0.3	Not equal to 0 (to get a cyclic term)
	b_{ijk}	NA	-	b_{ijk}	1	Arbitrary
	c_{ijk}	NA	-	c_{ijk}	1	Arbitrary
	d_{ijk}	NA	-	d_{ijk}	1	Arbitrary
	e_{ik}	0.25	To attain maximum value of dependent variable equal to 1	e_{ik}	0.25	To attain maximum value of dependent variable equal to 1
	c_{2ijk}	0	No deceptive terms	c_{2ijk}	0	No deceptive terms
	M_{ijk}	NA	-	M_{ijk}	NA	-
	g_{ijk}	NA	-	g_{ijk}	NA	-
	p_{ijk}	NA	-	p_{ijk}	NA	-
	ε	NA	-	ε	NA	-
μ	NA	-	μ	0	To have the data centred on the equation for x_2	
σ^2	NA	-	σ^2	0.05	To attain a distribution of about 10%	

Table 7.8: Parameter Values for RETB-II Examples – (a) Example 1: Biased Problem and Example 2: Multi-front Problem (b) Example 3: Deceptive Problem and Example 4: Discontinuous Problem (contd.)

(b)

Example 3: Deceptive Problem			Example 4: Discontinuous Problem		
Parameter Values		Reasoning	Parameter Values		Reasoning
N_d	1	1 dependent variable	N_d	1	1 dependent variable
$N_{ind,l}$	1	1 variable defining dependent variable	$N_{ind,l}$	1	1 variable defining dependent variable
N_i	2	2 independent variables	N_i	2	2 independent variables
n	3	3-variable problem	n	3	3-variable problem
Q_l	1	To attain continuous function search space	Q_l	2	To attain a discontinuity in the function search space
P_{ik}	Degree 2	To get a non-linear dependency equation; coefficients are chosen to attain minimum value of 0 and maximum value of $\cong 1$	P_{ik}	Degree 2,2	To get non-linear dependency equations for the two discontinuous parts; coefficients are chosen to attain minimum value of 0 and maximum value of 0.9
c_{1ijk}	0	No cyclic terms	c_{1ijk}	0,0	No cyclic terms in the two discontinuous parts
b_{ijk}	NA	-	b_{ijk}	NA	-
c_{ijk}	NA	-	c_{ijk}	NA	-
d_{ijk}	NA	-	d_{ijk}	NA	-
e_{ik}	0.1	To attain minimum value of 0 and maximum value of $\cong 1$	e_{ik}	0,0.05	To attain minimum value of 0 and maximum value of 0.9
c_{2ijk}	1	Not equal to 0 (to get a deceptive term)	c_{2ijk}	0,0	No deceptive terms
M_{ijk}	1	1 deceptive optimum	M_{ijk}	NA	-
g_{ijk}	0.0957	To attain minimum value of 0 and maximum value of $\cong 1$	g_{ijk}	NA	-
p_{ijk}	0.5	To attain deception at variable value of 0.5	p_{ijk}	NA	-
ε	0.004	Arbitrary, small, positive	ε	NA	-
μ	NA	-	μ	0	To have the data centred on the equation for x_2
σ^2	NA	-	σ^2	0.05	To attain a distribution of about 10%

Eg. 3
&
Eg. 4

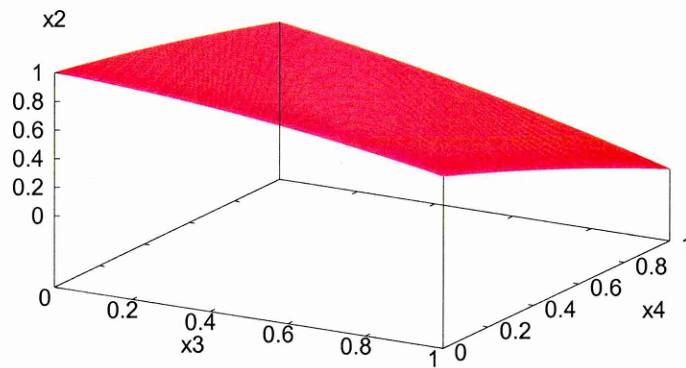


Figure 7.13: Dependency Relationship in RETB-II Example 1: Biased Problem

7.9.1.3 Step 3

Since in this problem, the dependency equation is assumed to be known, it is not required to generate data for representing the dependency relationship. Hence, in this case, Step 3 is omitted. For the sake of illustration, the exhaustive search corresponding to this problem is given in Figure 7.14. The non-linearity of the search space is evident from this figure.

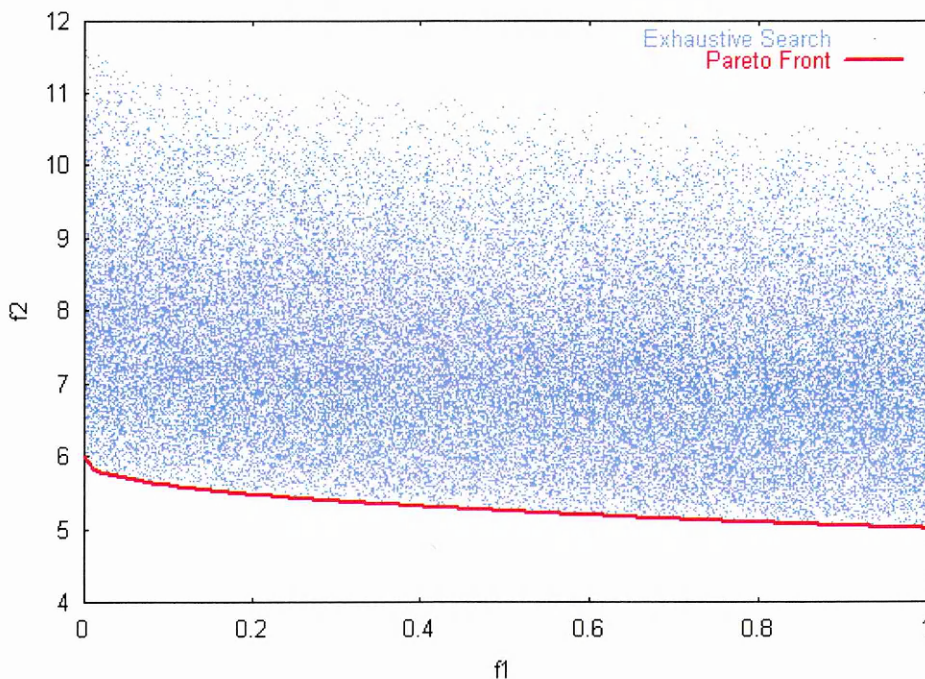


Figure 7.14: Exhaustive Search for RETB-II Example 1: Biased Problem

7.9.2 Example 2: Multi-front Problem

In this problem, the dependency relationship introduces multiple local fronts in the search space. The steps involved in creating this problem are as follows.

7.9.2.1 Step 1

As in the previous case, the variable x_2 is assumed to be dependent on two independent variables (x_3 and x_4) that are introduced in this problem. The variable x_1 is also assumed to be independent. This gives a value of 1 to $N_d(x_2)$, a value of 3 to $N_i(x_1, x_3, x_4)$ and hence a value of 4 to $n(N_d + N_i)$. As in Example 1, the independent variables are arranged in the order of their indices in the GA chromosome.

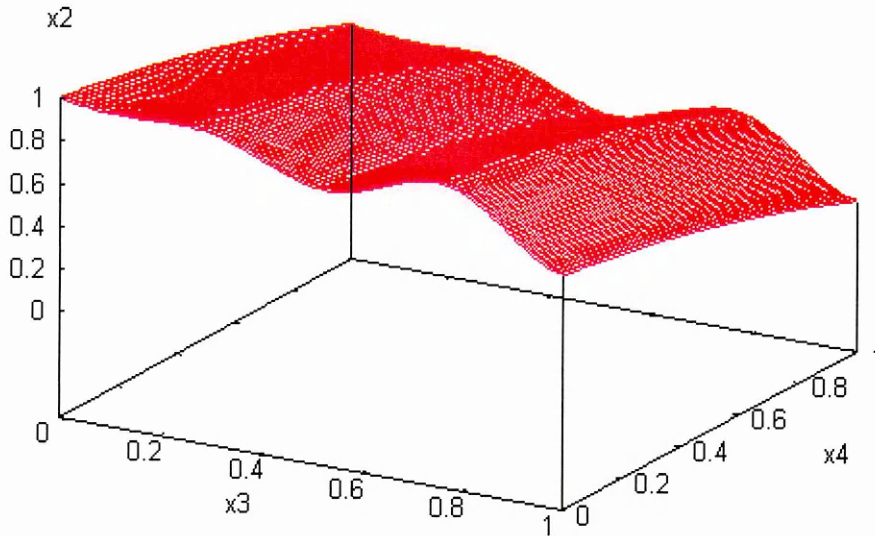


Figure 7.15: Dependency Relationship in RETB-II Example 2: Multi-front Problem

7.9.2.2 Step 2

In order to attain a search space with multiple local fronts, a cyclic dependency equation needs to be developed for x_2 in terms of x_3 and x_4 . This guides the selection of parameters for the required dependency equation (Equation 7.21), as shown in Table 7.8. This equation is given below and is also graphically represented in Figure 7.15.

$$x_2 = 1 - 0.1x_3 - 0.3x_3 \cos^2(2\pi x_3) - 0.2x_4 - 0.2x_4^2, \quad \text{Equation 7.21}$$

$$\forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1.$$

7.9.2.3 Step 3

In this problem, the dependency equation is assumed to be unknown. Therefore, multiple sets of variable values are created using the above-mentioned equation for x_2 in the equation for x_{di} . In this case, the noise values are obtained with mean (μ) of 0 (to have the data centred on the equation for x_2) and variance (σ^2) of 0.05 (to give a distribution of about 10%). The equation that is used for generating variable values is given below.

$$x_2' = x_2 + \text{Normal}(0,0.05) \quad \text{Equation 7.22}$$

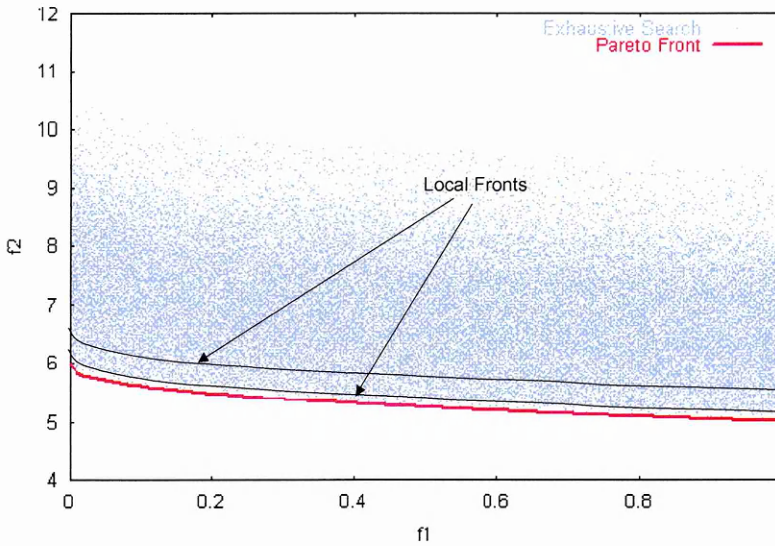


Figure 7.16: Exhaustive Search for RETB-II Example 2: Multi-front Problem

For the sake of illustration, the exhaustive search corresponding to this problem is given in Figure 7.16. Here, the global Pareto front corresponds to the global minimum of the I function, which occurs at the global maximum of the x_2 function. The two local fronts in the figure correspond to the two minima of \cos^2 function (that create the two local maxima of the x_2 function), and the global Pareto front corresponds to the global maximum of the x_2 function. Further, since x_2 now varies

from 0.2 to 1, as opposed to 0 to 1 in the case of independent variables, the upper part of the search space gets truncated.

7.9.3 Example 3: Deceptive Problem

Here, a test problem is created that has deceptiveness in its search space due to dependence among its decision variables. The steps involved in constructing this problem are as follows.

7.9.3.1 Step 1

In this case, the variable x_2 is assumed to be dependent on a newly introduced independent variable (x_3). The variable x_1 is also assumed to be independent. This gives a value of 1 to $N_d(x_2)$, a value of 2 to $N_i(x_1, x_3)$ and hence a value of 3 to $n(N_d + N_i)$. As in previous examples, the independent variables are arranged in the order of their indices in the GA chromosome.

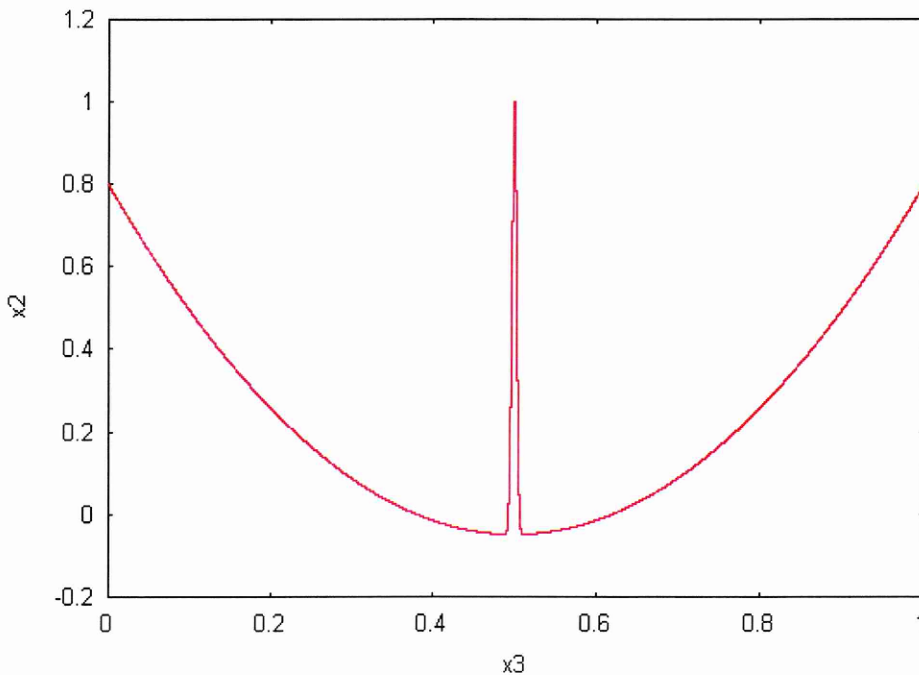


Figure 7.17: Dependency Relationship in RETB-II Example 3: Deceptive Problem

7.9.3.2 Step 2

In order to attain a deceptive search space, a dependency equation with a deceptive optimum is developed for x_2 in terms of x_3 . This guides the selection of parameters for the required dependency equation, as shown in Table 7.8. This equation is given below (Equation 7.23) and is also graphically represented in Figure 7.17.

$$x_2 = 1 - 0.2 - 3.396x_3 + 3.396x_3^2 + \exp(0.0957x_3) \exp\left(-\left(\frac{x_3 - 0.5}{0.004}\right)^2\right), \quad \text{Equation 7.23}$$

$$\forall 0 \leq x_3 \leq 1.$$

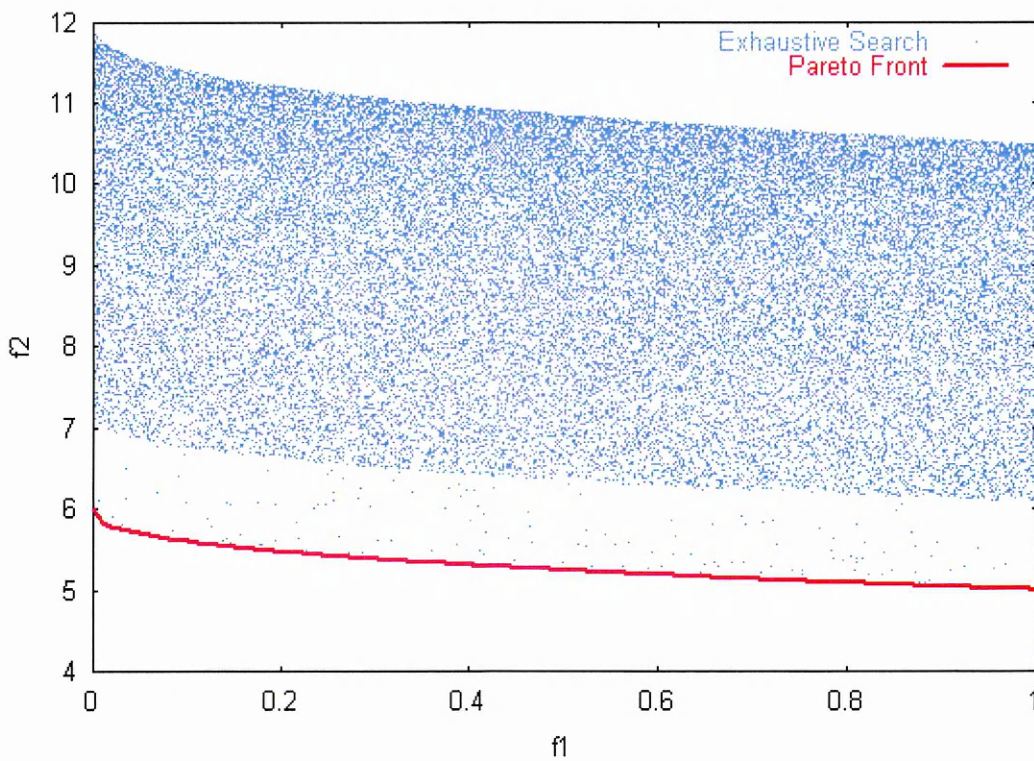


Figure 7.18: Exhaustive Search for RETB-II Example 3: Deceptive Problem

7.9.3.3 Step 3

Since in this problem, the dependency equation is assumed to be known, it is not required to generate data for representing the dependency relationship. Hence, in this case, Step 3 is omitted. The exhaustive search corresponding to this problem is given in Figure 7.18. It can be seen from this figure that there is a deception in the search

space, which corresponds to the deceptiveness in the relationship of x_2 in terms of x_3 . This situation arises because in this problem the global Pareto front corresponds to the global maximum of x_2 , which is an isolated optimum.

7.9.4 Example 4: Discontinuous Search Space Problem

Here, the aim is to construct a problem with a discontinuous search space. The RETB-II steps involved in constructing this problem are as follows.

7.9.4.1 Step 1

In this case, the variable x_2 is assumed to be dependent on a newly introduced independent variable (x_3). The variable x_1 is also assumed to be independent. This gives a value of 1 to $N_d(x_2)$, a value of 2 to $N_i(x_1, x_3)$ and hence a value of 3 to $n(N_d + N_i)$. As in previous examples, the independent variables are arranged in the order of their indices in the GA chromosome.

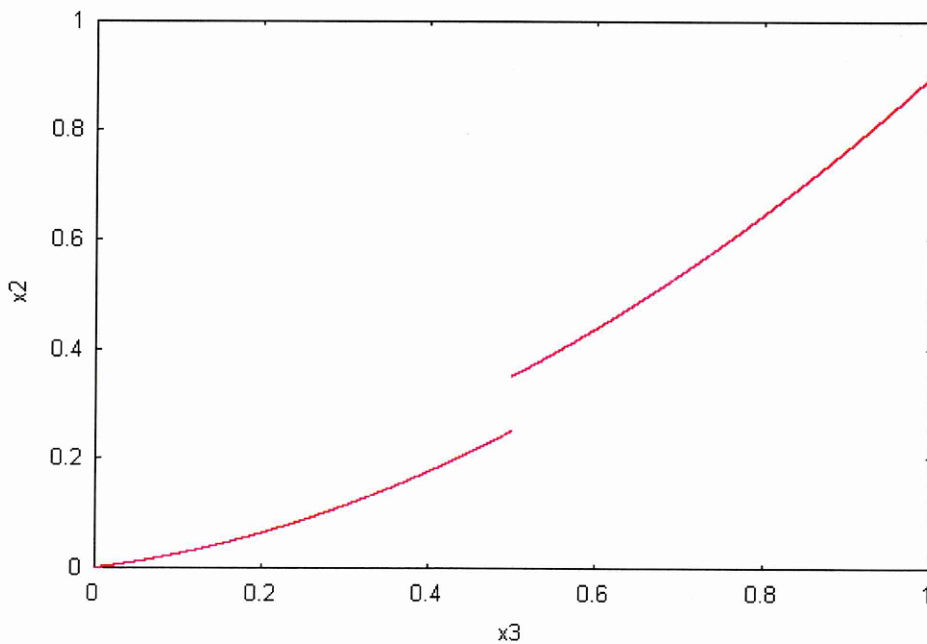


Figure 7.19: Dependency Relationship in RETB-II Example 4: Discontinuous Problem

7.9.4.2 Step 2

In order to attain a discontinuous search space, a dependency equation that has a discontinuity is developed for x_2 in terms of x_3 . This guides the selection of parameters for the required dependency equation, as shown in Table 7.8. This equation is given below (Equation 7.24) and is also graphically represented in Figure 7.19.

$$\begin{aligned} x_2 &= 0.2x_3 + 0.6x_3^2, \forall 0 \leq x_3 \leq 0.5, \\ x_2 &= 0.1 + 0.2x_3 + 0.6x_3^2, \forall 0.5 < x_3 \leq 1. \end{aligned} \quad \text{Equation 7.24}$$

7.9.4.3 Step 3

In this problem, the dependency equation is assumed to be unknown. Therefore, multiple sets of variable values are created using the above-mentioned equation for x_2 in the equation for x_{di} . In this case, the noise values are obtained with mean (μ) of 0 (to have the data centred on the equation for x_2) and variance (σ^2) of 0.05 (to give a distribution of about 10%). The equation that is used for generating variable values is given below (Equation 7.25).

$$x_2' = x_2 + \text{Normal}(0, 0.05). \quad \text{Equation 7.25}$$

The exhaustive search corresponding to this problem is given in Figure 7.20. Here, the global Pareto front corresponds to the global minimum of the I function, which occurs at the global maximum of the x_2 function. Since the introduction of dependence changes the global maximum of x_2 from 1 to 0.9, the part of the search space that corresponds to the values of x_2 between 0.9 and 1 becomes infeasible. This gives a new Pareto front to the search space. As revealed in Figure 7.19, the dependency equation also exhibits a discontinuity that makes a certain band of the values of x_2 infeasible. This leads to discontinuity in the search space corresponding to these values of x_2 . It should be noted that in this problem the variable search space would also be discontinuous, and would take the shape of two disjoint rectangles to reflect the discontinuity gap in the values of x_2 . Unlike this problem, all the previous test problems reported here have continuous, rectangular variable search spaces.

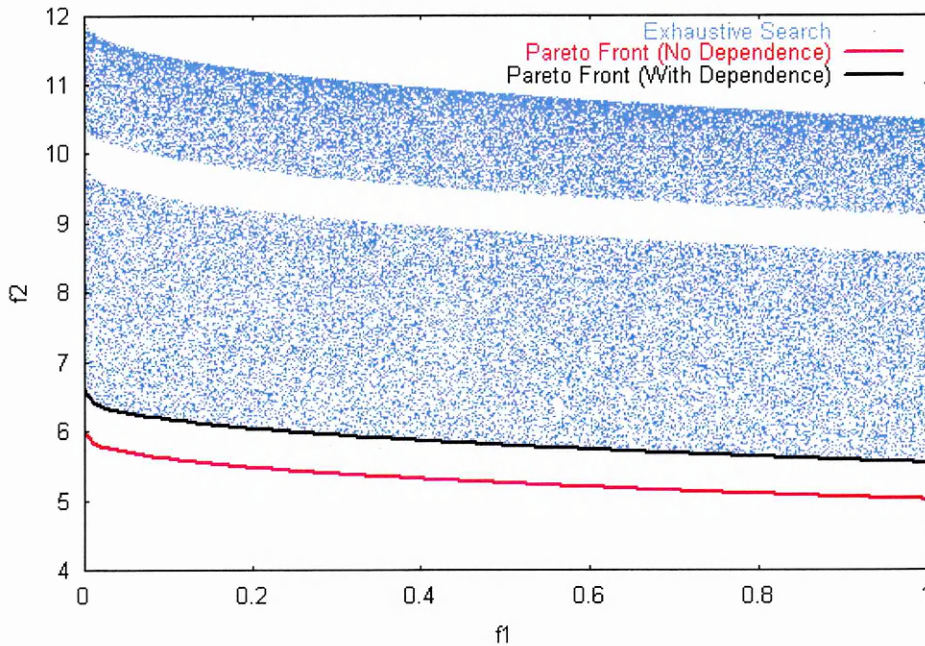


Figure 7.20: Exhaustive Search for RETB-II Example 4: Discontinuous Problem

A closer look at RETB reveals that there are two functions D and I that are defined in terms of the decision variables. In all the above problems, the variable x_2 was treated as dependent. Since this variable defines the I function, the introduction of variable dependence creates features that obstruct only convergence to the Pareto front. These features, which are introduced parallel to the Pareto front, include bias, multiple fronts, deception and discontinuity. Let us consider the other case, when the variable x_1 is treated as dependent. This has an effect of varying the function D that controls diversity across the Pareto front. Hence, introduction of dependency equations, similar to the ones used in the above examples, would have the following influences on the search space.

- ◆ Non-linear/Multi-dimensional Function: Bias towards certain parts of the Pareto front.
- ◆ Cyclic Function: Multiple basins of attraction across the Pareto front.
- ◆ Deceptive Function: Deception on the Pareto front.
- ◆ Discontinuous Function: Part of the Pareto front is rendered infeasible.

Table 7.9: Dependency Chart (DC) – (a) Examples 1 & 2 (b) Examples 3 & 4

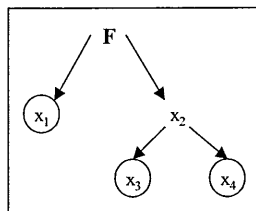
(a)

Dependency Chart (DC)		Variables			
		x_1	x_2	x_3	x_4
Regression Equations	x_1	I			
	x_2		D	I	I
	x_3				
	x_4				

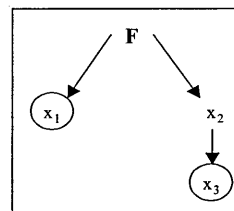
(b)

Dependency Chart (DC)		Variables		
		x_1	x_2	x_3
Regression Equations	x_1	I		
	x_2		D	I
	x_3			

Finally, it is worth mentioning that in all the above examples, the DC and DT were used to guide the development of dependency equations. The DC and DT corresponding to all the above examples are given in Table 7.9 and Figure 7.21 respectively.



(a)



(b)

Figure 7.21: Dependency Tree (DT) – (a) Examples 1 & 2 (b) Examples 3 & 4

7.10 RETB|RETB-II versus Existing Test Beds

RETB presents a generic tuneable framework for the development of test functions. It is capable of controlled simulation of three features of real-life optimisation problems: multiple objectives having inseparable function interaction and

constraints. Since RETB provides handles for controlled variation of its complexity, it is able to determine the extent to which a given optimisation technique is capable of handling a specified feature of real-life problems.

Due to its tuneable nature, RETB has a clear advantage for real-life simulation over the non-tuneable test problems, mentioned in Chapter 2. These non-tuneable test beds are static in nature, i.e., they do not provide parameters for varying their complexity levels. This prevents them from being used as controlled testing beds for optimisation algorithms. Furthermore, as opposed to RETB, these test beds also do not provide the facility of explicitly analysing the performance of optimisation algorithms with respect to each aspect of multi-objective optimisation.

Deb (1999b) suggested a basic tuneable framework that is capable of constructing multi-objective optimisation problems having varying degrees of complexity. RETB, on the other hand, uses a ‘reverse engineering’ strategy that constructs multi-objective optimisation problems to correspond to a given Pareto front, relationship(s)/interaction among variables of Pareto-optimal solutions, hindrance to diversity and nature of constraints. Another drawback of Deb’s (1999b) work is that he does not propose generic, parametric function prototypes for his test bed. This prevents systematic and controlled analysis of the optimisation algorithms. RETB, on the other hand, proposes parametric prototypes for each of its four test bed functions: *S*, *D*, *I* and *C*. This allows stepwise increments in difficulty of the test bed, with respect to each aspect of complexity. In this way, RETB is able to systematically analyse optimisation algorithms, thereby successfully overcoming the drawback of Deb’s (1999b) framework. Recently, Deb *et al.* (2001) presented a parametric function prototype for constraints that enables controlled variation of the challenges that they pose for multi-objective optimisation problems. However, this work also has drawbacks since the systematic analysis of multi-objective optimisation algorithms that is facilitated by this approach focuses only on constraints, without addressing its interactions with the complexity introduced by the objective functions. As opposed to this, RETB takes a complete picture of constrained multi-objective

optimisation through its four function prototypes (*S*, *D*, *I* and *C*), which individually represent the four critical aspects of real-life optimisation.

The effectiveness of RETB is further enhanced due to its extension to RETB-II that is capable of constructing optimisation test problems that have dependence among their decision variables. Since there are no test problems reported in literature for variable dependence, RETB-II is unique in its capability of testing optimisation algorithms in the presence of dependence among decision variables.

This makes RETB/RETB-II more effective than the existing test beds. This is because unlike other test function development schemes, they simulate the core issues associated with multiple objectives, constraints and variable interaction, rather than their symptoms. The ‘reverse engineering’ strategy adopted here also enables a more intensive interpretation of each term of the proposed test bed, in terms of its relevance to the complexity of optimisation problems thus created. In this way, RETB/RETB-II provide controlled testing of optimisation algorithms with respect to multiple objectives, constraints and variable interaction in engineering design optimisation problems.

7.11 Summary

This chapter has proposed two test beds, RETB and RETB-II, for controlled simulation of multiple objectives, constraints and variable interaction in engineering design optimisation problems. As shown below, RETB and RETB-II together meet all the objectives for their development set at the beginning of this chapter.

- ◆ RETB and RETB-II provide a generic methodology for the development of test problems. So, they can generate a wide variety of optimisation problems, having varying degrees of complexity levels.
- ◆ They provide a unified framework, for controlled testing of optimisation algorithms with respect to three features of real-life engineering design optimisation: presence of multiple objectives, constraints and variable interaction.

- ◆ They provide generic, parametric prototypes for each of the functions in their definition. This provides better control on the complexity of optimisation problems thus generated, thereby enabling systematic and controlled analysis of the optimisation problems.
- ◆ They provide explicit functions/parameters to control the complexity introduced by the two types of variable interaction.
 - *Inseparable function interaction.*
 - *Variable dependence.*

This chapter has achieved the following.

- ◆ It has identified the factors that need to be controlled for simulating multiple objectives, constraints and variable interaction in engineering design optimisation problems.
- ◆ It has devised a generic strategy for test bed development.
- ◆ It has applied this strategy for proposing two test beds, Reverse Engineered Test Bed (RETB) and RETB-II, for simulating multiple objective, constraints and variable interaction.
- ◆ It has developed parametric function prototypes for the proposed test beds.
- ◆ It has presented guidelines and case studies that demonstrate the use of the proposed test beds.
- ◆ It has finally compared the proposed test bed with the existing ones.

The last three chapters (Chapter 5, Chapter 6 and Chapter 7) have focused on interaction among decision variables. Chapter 5 and Chapter 6 have respectively proposed two EC techniques, GRGA and GAVD, for handling inseparable function interaction and variable dependence in multi-objective optimisation problems. This chapter has proposed two generic, parametric test beds, RETB and RETB-II, that can handle multiple objective functions, constraints and variable interaction in a single framework. The next chapter applies these test beds for analysing the performance of GRGA and GAVD.

8 PERFORMANCE ANALYSIS OF GRGA/GAVD USING RETB/RETB-II

This chapter analyses the performance of GRGA and GAVD using the two test beds, RETB and RETB-II, developed in the previous chapter. Here, a high-performing, novel multi-objective optimisation algorithm, NSGA-II, is also included in the analysis to enable the comparison of GRGA and GAVD with the state-of-the-art reported in literature. This chapter attempts to achieve the following.

- ◆ *To develop a set of RETB/RETB-II case studies such that it represents three features of real-life engineering design optimisation problems: presence of multiple objectives, constraints and variable interaction.*
- ◆ *To report the experimental results produced by GAVD, GRGA and NSGA-II in each of the cases.*
- ◆ *To analyse the experimental results.*
- ◆ *To draw conclusions regarding the performance of GRGA and GAVD based on this analysis.*

8.1 Case Study Development

This section constructs three multi-objective optimisation problems, having pre-defined characteristics, and uses them for comparing the performances of GRGA, GAVD and NSGA-II. The three test cases are constructed such that they together represent the presence of multiple objectives, constraints and interaction among decision variables in engineering design optimisation problems. However, explicit constraints are not included in these test cases since the presence of variable dependence in these problems has the effect of constraining the search space. This is because the bounds on dependent variables are treated as constraints. Hence, to focus on the complexity introduced by variable dependence, explicit constraints are avoided in the test cases. Here, RETB is first used to define the objective functions

based on the desired features of these test cases (as listed in Table 8.1). As can be seen from Table 8.1, the test cases thus created incorporate a variety of features of multi-objective optimisation problems: discontinuous Pareto front, inherent bias on Pareto front and multiple local fronts. RETB-II is then applied to each of these test cases to introduce the desired nature of dependency among decision variables. Since most real-life optimisation problems do not have explicit dependency equations, RETB-II introduces variable dependency in these three test cases by providing multiple sets of variable values. From this information, the GAVD needs to infer the dependency relationships. These data are created using the equation for x_{di}' (Equation 7.18), and involve noise terms with mean (μ) of 0 (to have the data centred on the equation for x_{di}) and variance (σ^2) of 0.05 (to give a distribution of about 10%). The equation that creates this data from x_{di} is given below (Equation 8.1).

$$x_{di}' = x_{di} + Normal(0,0.05) \tag{Equation 8.1}$$

The three test cases are presented in the discussion that follows.

Table 8.1: Test Cases

Case Studies	Case-1	Case-2	Case-3
Problem Features	<ul style="list-style-type: none"> • Two objectives • Two variables • Convex and continuous S (Pareto front) • Biased D (biased region on Pareto front) • Multi-front (multiple local Pareto fronts) • No constraints 	<ul style="list-style-type: none"> • Two objectives • Two variables • Convex and discontinuous S (Pareto front) • Biased D (biased region on Pareto front) • Multi-front (multiple local Pareto fronts) • No constraints 	<ul style="list-style-type: none"> • Four objectives • Four variables • Convex and continuous S (Pareto front) with respect to f_1 and f_2; concave and continuous S (Pareto front) with respect to f_3 • Biased D_1 and D_2 (two biased regions on Pareto front); unbiased D_3 • Single front (Dependency scenario 3.1) • Multi-front (multiple local Pareto fronts) (Dependency scenario 3.2) • No constraints

8.1.1 Case-1

As shown in Table 8.1, this problem has multiple local fronts and a biased region on the Pareto front. Based on these features, RETB parameters are given appropriate values, as shown in Table 8.2. Equation 8.2 and Equation 8.3 show the equations

corresponding to this problem. This problem is also pictorially depicted in Figure 8.1.

$$D(\bar{x}') = \frac{1}{(1 - \exp(-4))} [1 - \exp(-4x_1)], \forall 0 \leq x_1 \leq 1, \tag{Equation 8.2}$$

$$I(\bar{x}'') = 2 - \exp(-2x_2) \cos(8\pi x_2), \forall 0 \leq x_2 \leq 1,$$

$$s(f_1, I) = 2 - (f_1 / I)^{0.6},$$

$$f_1 = D(\bar{x}'),$$

$$f_2 = s(f_1, I) \times I(\bar{x}'').$$

$$\text{Pareto_front} \Rightarrow f_2 = 2 - (f_1)^{0.6}. \tag{Equation 8.3}$$

Here, two variable dependency scenarios are introduced in this problem. Since GAVD uses RA to model the dependence relationships, the first scenario is created such that it can be exactly modelled by GAVD, whereas the second needs to be approximated by it. Both the scenarios are non-linear, with one having cyclic features.

Table 8.2: RETB Parameter Values for Case-1

	S			D			I		
	Parameter Values	Reasoning	Parameter Values	Reasoning	Parameter Values	Reasoning			
Case-1	M	2	2 objectives	k	1	1 variable defining D	n	2	2 variable-problem
	a_i	0.6	< 1 for convex Pareto front	M_i	1	1 biased region on Pareto front	k	1	1 variable defining D
	c_{ii}	0	To attain continuous Pareto front	T	1	Product of M_i 's	b_i	2	Arbitrary
	b_i	NA	-	a_{ij}	1	Arbitrary	c_i	8	(8+1)=9 local Pareto fronts
	c_i	NA	-	b_{ij}	4	Arbitrary	M_i	0	No deceptive front
	d_i	NA	-	R_i	0	So that $f_{i\max}$ is 1	d_i	NA	-
	e	1	For simplicity	a_k	NA	-	ε	NA	-
							e	1	So that l_{\min} is 1

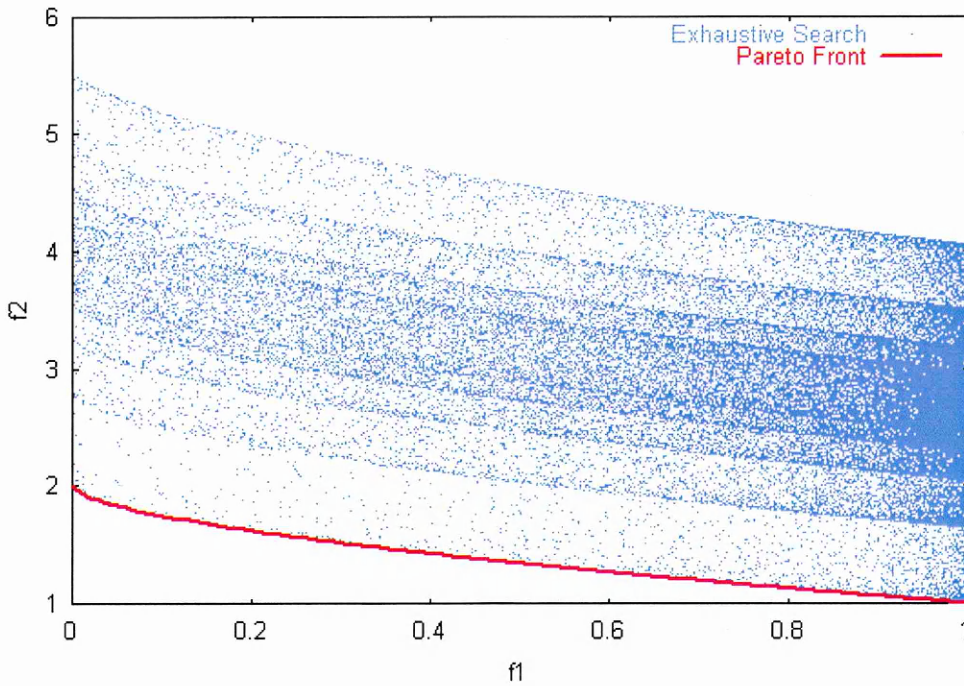


Figure 8.1: Case-1

8.1.1.1 Dependency Scenario 1.1

GAVD uses non-linear (quadratic) multi-variable RA to model the dependence among variables. In order to allow GAVD to exactly model the dependence, the equation that is constructed in this scenario is of degree two, with no cyclic and deceptive terms. This equation is defined in terms of two independent variables (x_3 and x_4). The RETB-II parameters that make this equation are given in Table 8.3, and the corresponding equation is given below (Equation 8.4). This dependency equation is pictorially depicted in Figure 8.2.

$$x_2 = 1 - 0.1x_3 - 0.2x_3^2 - 0.3x_4 - 0.1x_4^2 - 0.3x_3x_4, \quad \text{Equation 8.4}$$

$$\forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1.$$

Table 8.3: RETB-II Parameter Values for Case-1

	Dependency Scenario 1.1			Dependency Scenario 1.2		
	Parameter Values		Reasoning	Parameter Values		Reasoning
Case-1	N_d	1	1 dependent variable	N_d	1	1 dependent variable
	$N_{ind,i}$	2	2 variables defining dependent variable	$N_{ind,i}$	2	2 variables defining dependent variable
	N_i	3	3 independent variables	N_i	3	3 independent variables
	n	4	4-variable problem	n	4	4-variable problem
	Q_i	1	To attain continuous function search space	Q_i	1	To attain a continuous function search space
	P_{ik}	Degree 2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain minimum value of 0	P_{ik}	Degree 2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain minimum value of 0.2
	c_{1ijk}	0	No cyclic terms	c_{1ijk}	0.3	Not equal to 0 (to get a cyclic term)
	b_{ijk}	NA	-	b_{ijk}	1	Arbitrary
	c_{ijk}	NA	-	c_{ijk}	1	Arbitrary
	d_{ijk}	NA	-	d_{ijk}	1	Arbitrary
	e_{ik}	0.25	To attain maximum value of dependent variable equal to 1	e_{ik}	0.25	To attain maximum value of dependent variable equal to 1
	c_{2ijk}	0	No deceptive terms	c_{2ijk}	0	No deceptive terms
	M_{ijk}	NA	-	M_{ijk}	NA	-
	g_{ijk}	NA	-	g_{ijk}	NA	-
	p_{ijk}	NA	-	p_{ijk}	NA	-
	ε	NA	-	ε	NA	-
	μ	0	To have the data centred on the equation for x_2	μ	0	To have the data centred on the equation for x_2
σ^2	0.05	To attain a distribution of about 10%	σ^2	0.05	To attain a distribution of about 10%	

8.1.1.2 Dependency Scenario 1.2

In this case, a two-variable cyclic equation is used to define the dependency equation. This relationship is approximated by GAVD. The RETB-II parameters that make this equation are given in Table 8.3, and the corresponding equation is given below (Equation 8.5). This dependency equation is shown in Figure 8.3.

$$x_2 = 1 - 0.1x_3 - 0.3x_3 \cos^2(2\pi x_3) - 0.2x_4 - 0.2x_4^2,$$

$$\forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1.$$

Equation 8.5

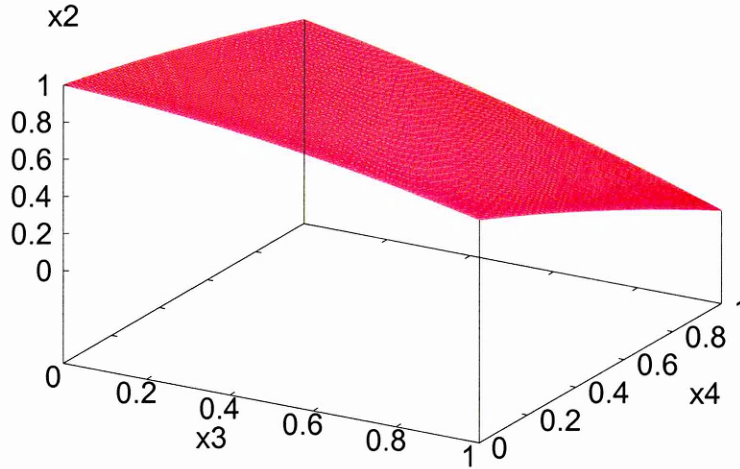


Figure 8.2 Dependency Relationship in Case-1 (Dependency Scenario 1.1)

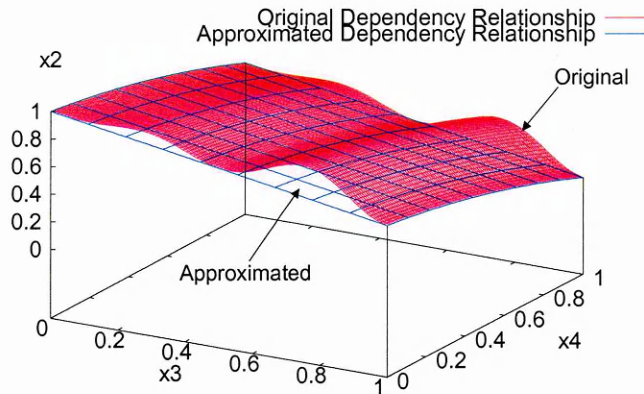


Figure 8.3: Dependency Relationship in Case-1 (Dependency Scenario 1.2)
(Original: Actual Relationship, Approximated: Relationship Estimated by RA)

8.1.2 Case-2

The features of this problem are similar to those of Case-1, with the difference that here the Pareto front is discontinuous in nature (Table 8.1). Based on these features, RETB parameters are given appropriate values to get the following problem (Equation 8.6, Equation 8.7 and Figure 8.4). The rationale behind the choice of these parameter values is given in Table 8.4.

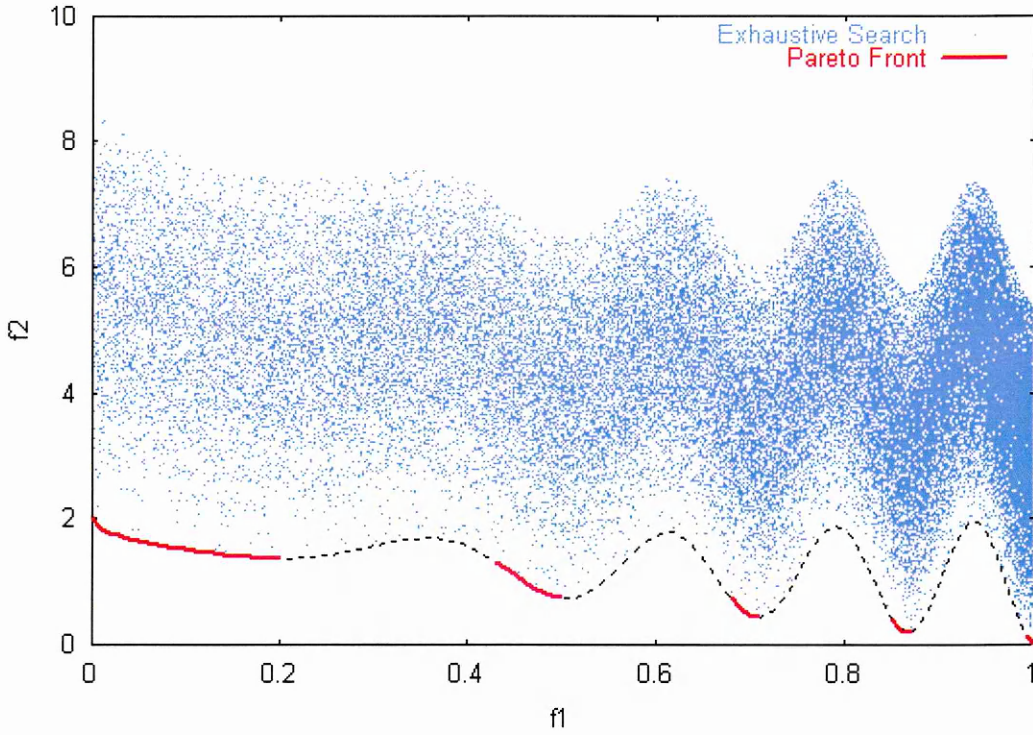


Figure 8.4: Case-2

$$D(\bar{x}') = \frac{1}{(1 - \exp(-3))} [1 - \exp(-3x_1)], \forall 0 \leq x_1 \leq 1, \quad \text{Equation 8.6}$$

$$I(\bar{x}'') = 3 - \exp(-x_2) \cos(2\pi x_2) - \exp(-x_3) \cos(4\pi x_3), \forall 0 \leq x_2, x_3 \leq 1,$$

$$s(f_1, I) = 2 - (f_1 / I)^{0.4} - (f_1 / I) \cos(8\pi f_1^2),$$

$$f_1 = D(\bar{x}'),$$

$$f_2 = s(f_1, I) \times I(\bar{x}'').$$

$$\text{Pareto_front} \Rightarrow f_2 = 2 - (f_1)^{0.4} - (f_1) \cos(8\pi f_1^2). \quad \text{Equation 8.7}$$

Similar to Case-1, the performance of GAVD is analysed here using two variable dependency scenarios: one that can be exactly modelled by GAVD and the other that can only be approximated. Both the scenarios are non-linear, with one having deceptive features.

Table 8.4: RETB Parameters for Case-2

Case-2	S			D			I		
	Parameter Values		Reasoning	Parameter Values		Reasoning	Parameter Values		Reasoning
	M	2	2 objectives	k	1	1 variable defining D	n	3	3 variable-problem
a_i	0.4	< 1 for convex Pareto front	M_i	1	1 biased region on Pareto front	k	1	1 variable defining D	
c_{1i}	1	To attain discontinuous Pareto front	T	1	Product of M_i 's	b_i	1,1	Arbitrary	
b_i	1	Arbitrary	a_{ij}	1	Arbitrary	c_i	2,4	$(2+1) \times (4+1) = 15$ local Pareto fronts	
c_i	4	$(4+1) = 5$ disconnected Pareto regions	b_{ij}	3	Arbitrary	M_i	0	No deceptive front	
d_i	2	Arbitrary	R_i	0	So that f_{1max} is 1	d_i	NA	-	
e	1	For simplicity	a_k	NA	-	ε	NA	-	
						e	1.5	So that l_{min} is 1	

8.1.2.1 Dependency Scenario 2.1

In this case, a quadratic equation is used to define the dependency relationship. This relationship can be exactly modelled by GAVD. The RETB-II parameters that make this equation are given in Table 8.5, and the corresponding equation is given below (Equation 8.8). This dependency equation is shown in Figure 8.5.

$$x_2 = 0.2 + 0.2x_3 + 0.6x_3^2, \forall 0 \leq x_3 \leq 1. \tag{Equation 8.8}$$

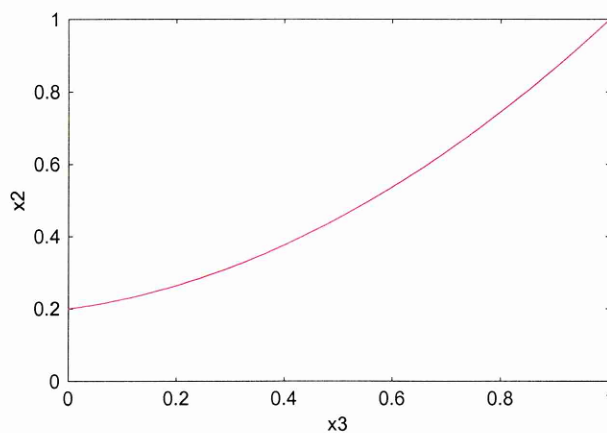


Figure 8.5: Dependency Relationship in Case-2 (Dependency Scenario 2.1)

Table 8.5: RETB-II Parameter Values for Case-2

		Dependency Scenario 2.1		Dependency Scenario 2.2		
		Parameter Values	Reasoning	Parameter Values	Reasoning	
Case-2	N_d	1	1 dependent variable	N_d	1	1 dependent variable
	$N_{ind,i}$	1	1 variable defining dependent variable	$N_{ind,i}$	1	1 variable defining dependent variable
	N_i	2	2 independent variables	N_i	2	2 independent variables
	n	3	3-variable problem	n	3	3-variable problem
	Q_i	1	To attain continuous function search space	Q_i	1	To attain continuous function search space
	P_{ik}	Degree 2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain maximum value of 1	P_{ik}	Degree 2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain minimum value of 0.2 and maximum value of $\cong 1$
	c_{1ijk}	0	No cyclic terms	c_{1ijk}	0	No cyclic terms
	b_{ijk}	NA	-	b_{ijk}	NA	-
	c_{ijk}	NA	-	c_{ijk}	NA	-
	d_{ijk}	NA	-	d_{ijk}	NA	-
	e_{ik}	0.25	To attain minimum value of dependent variable equal to 0.2	e_{ik}	0.1	To attain minimum value of 0.2 and maximum value of $\cong 1$
	c_{2ijk}	0	No deceptive terms	c_{2ijk}	1	Not equal to 0 (to get a deceptive term)
	M_{ijk}	NA	-	M_{ijk}	1	1 deceptive optimum
	g_{ijk}	NA	-	g_{ijk}	0.0957	To attain minimum value of 0.2 and maximum value of $\cong 1$
	p_{ijk}	NA	-	p_{ijk}	0.5	To attain deception at variable value of 0.5
	ε	NA	-	ε	0.004	Arbitrary, small, positive
	μ	0	To have the data centred on the equation for x_2	μ	0	To have the data centred on the equation for x_2
σ^2	0.05	To attain a distribution of about 10%	σ^2	0.05	To attain a distribution of about 10%	

8.1.2.2 Dependency Scenario 2.2

In this case, a two-variable deceptive equation is used to define the dependency relationship. This relationship is approximated by GAVD. The RETB-II parameters that make this equation are given in Table 8.5, and the corresponding equation is given below (Equation 8.9). This dependency equation is shown in Figure 8.6.

$$x_2 = 0.2 + 3.396x_3 - 3.396x_3^2 - \exp(0.0957x_3) \exp\left(-\left(\frac{x_3 - 0.5}{0.004}\right)^2\right), \quad \text{Equation 8.9}$$

$$\forall 0 \leq x_3 \leq 1.$$

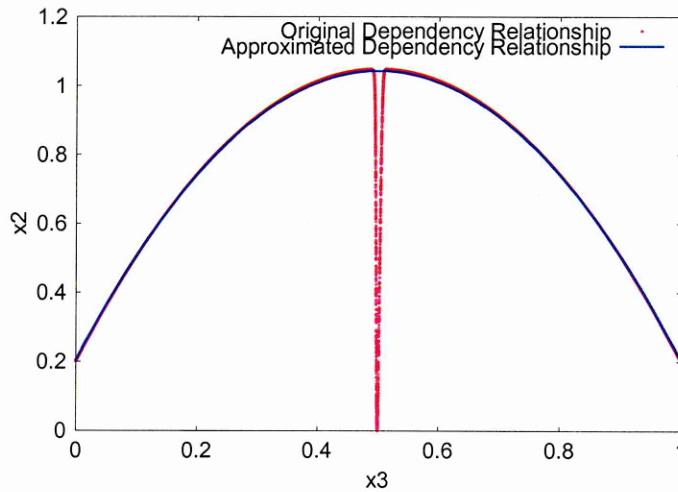


Figure 8.6: Dependency Relationship in Case-2 (Dependency Scenario 2.2)
(Original: Actual Relationship, Approximated: Relationship Estimated by RA)

8.1.3 Case-3

This problem is characterised by the presence of four objective functions. It has biased search space with respect to two of its objectives, and possesses single Pareto front (in Dependency Scenario 3.1) and multiple local Pareto fronts (in Dependency Scenario 3.2). This problem also has a multi-dimensional Pareto front that is convex with respect to two objectives, and concave with respect to one objective. Based on these features, RETB parameters are given appropriate values to get the following problem (Equation 8.10 and Equation 8.11). The rationale behind the choice of these parameters is given in Table 8.6.

$$D_1(\bar{x}') = \frac{1}{(1 - \exp(-5))} [1 - \exp(-5x_1)], \forall 0 \leq x_1 \leq 1, \tag{Equation 8.10}$$

$$D_2(\bar{x}'') = \frac{1}{(1 - \exp(-2))} [1 - \exp(-2x_2)], \forall 0 \leq x_2 \leq 1,$$

$$D_3(\bar{x}''') = 1 - x_3, \forall 0 \leq x_3 \leq 1,$$

$$I(\bar{x}''''') = 1 + x_4, \forall 0 \leq x_4 \leq 1 \dots \dots \dots \text{Dependency_Scenario} - 3.1,$$

$$I(\bar{x}''''') = 2 - \exp(-2x_4) \cos(4\pi x_4), \forall 0 \leq x_4 \leq 1 \dots \dots$$

....Dependency_Scenario - 3.2,

$$s(f_1, f_2, f_3, I) = 4 - (f_1 / I)^{0.6} - (f_2 / I)^{0.4} - (f_3 / I)^2,$$

$$f_1 = D_1(\bar{x}'),$$

$$f_2 = D_2(\bar{x}''),$$

$$f_3 = D_3(\bar{x}'''),$$

$$f_4 = s(f_1, f_2, f_3, I) \times I(\bar{x}''''').$$

$$\text{Pareto_front} \Rightarrow f_4 = 4 - (f_1)^{0.6} - (f_2)^{0.4} - (f_3)^2. \tag{Equation 8.11}$$

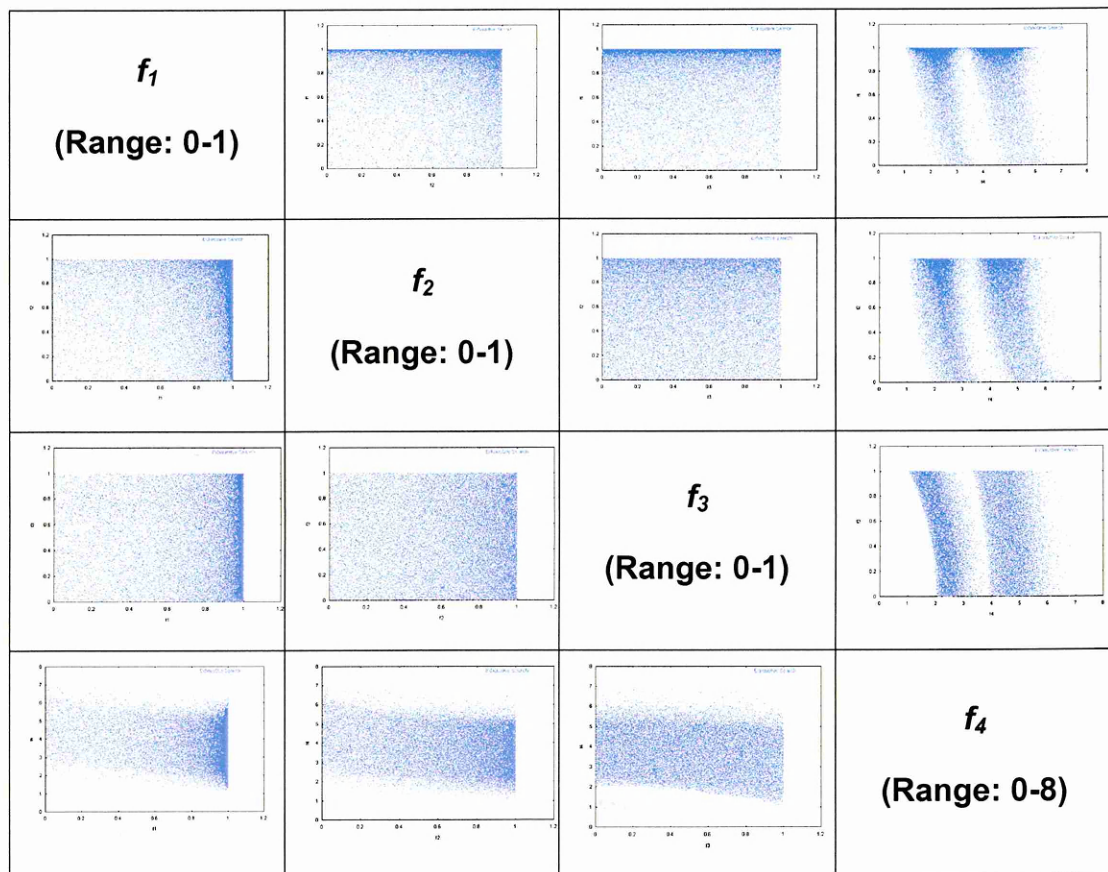
Table 8.6: RETB Parameters for Case-3

Case-3	S			D			I(Scenario 3.1); I(Scenario 3.2)		
	Parameter Values		Reasoning	Parameter Values		Reasoning	Parameter Values		Reasoning
	<i>M</i>	4	4 objectives	<i>k</i>	1,1,1	1 variable defining each <i>D</i>	<i>n</i>	4;4	4 variable-problem
<i>a_i</i>	0.6, 0.4, 2	< 1 for convex Pareto front and > 1 for non-convex Pareto front	<i>M_i</i>	1,1,0	Number of biased regions on Pareto front corresponding to each <i>D</i>	<i>k</i>	3;3	3 variables defining <i>D</i>	
<i>c_{1i}</i>	0	To attain continuous Pareto front	<i>T</i>	1,1,NA	Product of <i>M_i</i> 's in case of biased <i>D</i> 's	<i>b_i</i>	2; NA	Arbitrary for biased <i>I</i>	
<i>b_i</i>	NA	-	<i>a_{ij}</i>	1,1,NA	Arbitrary for biased <i>D</i> 's	<i>c_i</i>	4; NA	(4+1)=5 local Pareto fronts for biased <i>I</i>	
<i>c_i</i>	NA	-	<i>b_{ij}</i>	5,2,NA	Arbitrary for biased <i>D</i> 's	<i>M_i</i>	0; NA	No deceptive front for biased <i>I</i>	
<i>d_i</i>	NA	-	<i>R_i</i>	0,0,NA	So that <i>f_{imax}</i> is 1 for biased <i>D</i> 's	<i>d_i</i>	NA; NA	-	
<i>e</i>	NA	For simplicity	<i>a_k</i>	NA, NA, 1	So that <i>f_{imax}</i> is 1 for unbiased <i>D</i>	<i>ε</i>	NA; NA	-	
						<i>e</i>	1.5; 0.5	So that <i>I_{min}</i> is 1	

Here, the performance of GAVD is analysed using two variable dependency scenarios. The dependency equations in both the scenarios are non-linear and

discontinuous, with one having higher discontinuity than the other has. Further, the first scenario has single Pareto front, whereas the second one has multiple local Pareto fronts.

Table 8.7: Case-3 (Dependency Scenario 3.1) (Upper Diagonal Graphs: Search Space with Variable Dependency, Lower Diagonal Graphs: Search Space without Variable Dependency)



8.1.3.1 Dependency Scenario 3.1

The search space corresponding to this problem is depicted in Table 8.7. In this case, a highly discontinuous quadratic equation is used to define the dependency relationship. This relationship can be exactly modelled by GAVD (using piece-wise RA). The RETB-II parameters that make this equation are given in Table 8.8, and the corresponding equation is given below (Equation 8.12). This dependency equation is shown in Figure 8.7.

$$x_4 = 0.2x_5 + 0.4x_5^2, \forall 0 \leq x_5 \leq 0.5$$

Equation 8.12

$$x_4 = 0.4 + 0.2x_5 + 0.4x_5^2, \forall 0.5 < x_5 \leq 1$$

Table 8.8: RETB-II Parameter Values for Case-3

	Scenario 3.1			Scenario 3.2		
	Parameter Values		Reasoning	Parameter Values		Reasoning
	Case-3	N_d	1	1 dependent variable	N_d	1
$N_{ind,i}$		1	1 variable defining dependent variable	$N_{ind,i}$	1	1 variable defining dependent variable
N_i		4	4 independent variables	N_i	4	4 independent variables
n		5	5-variable problem	n	5	5-variable problem
Q_i		2	To attain a discontinuity in the function search space	Q_i	2	To attain a discontinuity in the function search space
P_{ik}		Degree 2,2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain minimum value of 0 and maximum value of 1.0	P_{ik}	Degree 2,2	Equal to degree of RA in GAVD; Non-linear; coefficients are chosen to attain minimum value of 0 and maximum value of 0.9
c_{1ijk}		0,0	No cyclic terms	c_{1ijk}	0,0	No cyclic terms
b_{ijk}		NA	-	b_{ijk}	NA	-
c_{ijk}		NA	-	c_{ijk}	NA	-
d_{ijk}		NA	-	d_{ijk}	NA	-
e_{ik}		0,0.05	To attain minimum value of 0 and maximum value of 1.0	e_{ik}	0,0.05	To attain minimum value of 0 and maximum value of 0.9
c_{2ijk}		0,0	No deceptive terms	c_{2ijk}	0,0	No deceptive terms
M_{ijk}		NA	-	M_{ijk}	NA	-
g_{ijk}		NA	-	g_{ijk}	NA	-
p_{ijk}		NA	-	p_{ijk}	NA	-
ε		NA	-	ε	NA	-
μ	0	To have the data centred on the equation for x_2	μ	0	To have the data centred on the equation for x_2	
σ^2	0.05	To attain a distribution of about 10%	σ^2	0.05	To attain a distribution of about 10%	

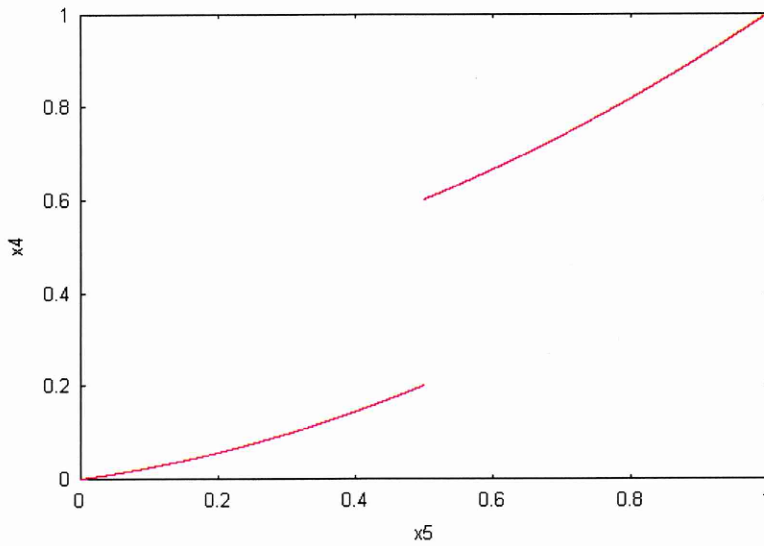


Figure 8.7: Dependency Relationship in Case-3 (Dependency Scenario 3.1)

8.1.3.2 Dependency Scenario 3.2

The search space corresponding to this problem is depicted in Table 8.9. In this case, a mildly discontinuous quadratic equation is used to define the dependency relationship. This relationship can be exactly modelled by GAVD (using piece-wise RA). The RETB-II parameters that make this equation are given in Table 8.8, and the corresponding equation is given below (Equation 8.13). This dependency equation is shown in Figure 8.8.

$$x_4 = 0.2x_5 + 0.6x_5^2, \forall 0 \leq x_5 \leq 0.5$$

Equation 8.13

$$x_4 = 0.1 + 0.2x_5 + 0.6x_5^2, \forall 0.5 < x_5 \leq 1$$

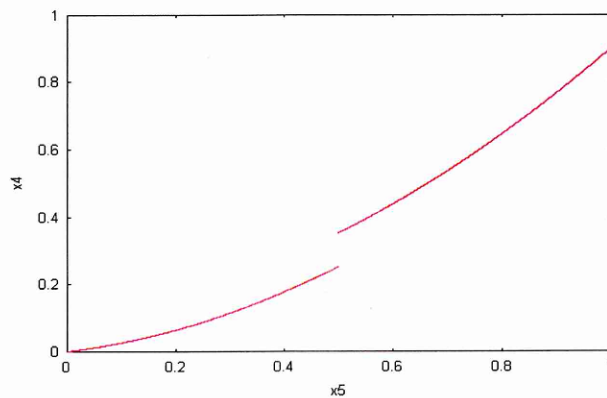
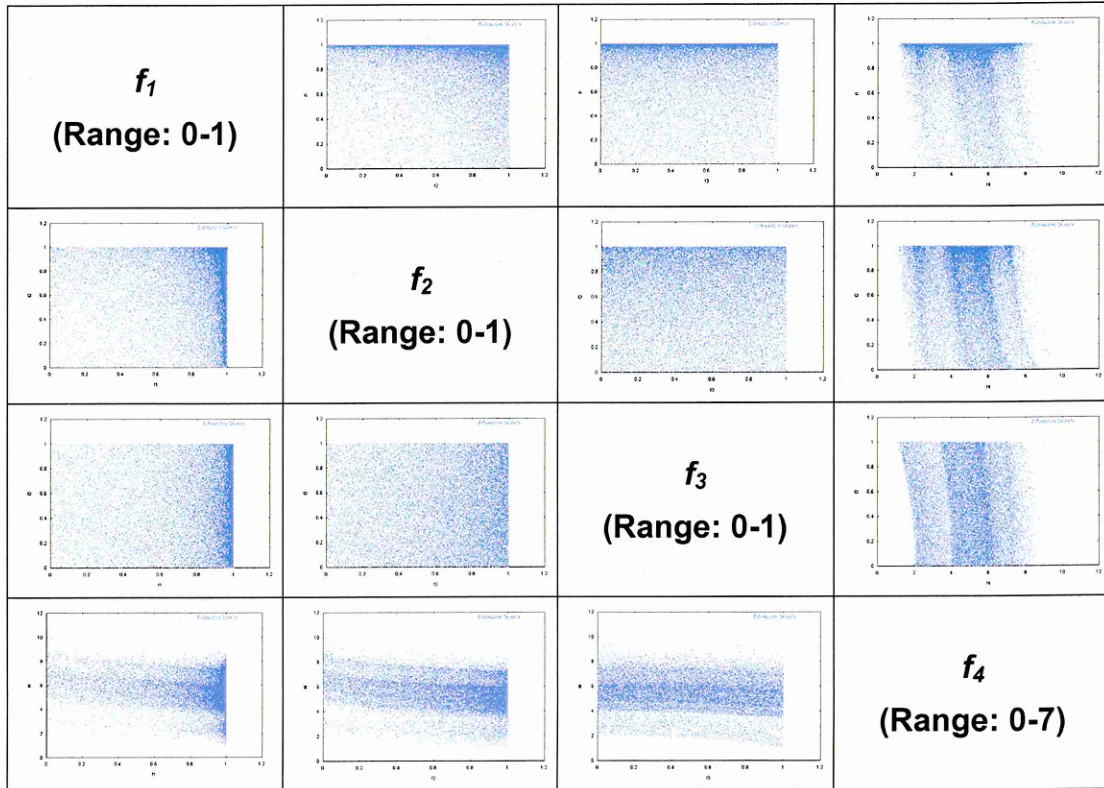


Figure 8.8: Dependency Relationship in Case-3 (Dependency Scenario 3.2)

Table 8.9: Case-3 (Dependency Scenario 3.2) (Upper Diagonal Graphs: Search Space with Variable Dependency, Lower Diagonal Graphs: Search Space without Variable Dependency)



8.2 Experimental Results

All the tests reported in this section correspond to 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability and simulated binary crossover, with 10 crossover distribution index and 50 mutation distribution index. The results reported here form the typical set obtained from 10 runs with different seed values for the random number generator. No major variation was observed in the results with the change in seed values.

Both GRGA and NSGA-II assume independence of variables, and hence do not take variable dependency into account. In all the above-mentioned case studies, these two algorithms work in search spaces that do not have any dependence among their decision variables. Since the two algorithms work on the same search space, the performance of GRGA is compared with that of NSGA-II in this chapter. However,

in the absence of dedicated techniques for handling variable dependence, the performance of GAVD cannot be compared to an algorithm that accounts for variable dependence. Therefore, in this chapter, GAVD is compared against two novel state-of-the-art optimisation algorithms (NSGA-II and GRGA). The idea is to demonstrate that in the presence of variable dependence, even the most effective of optimisation algorithms fail to produce good results if they do not have in-built mechanisms to address variable dependency. Here, the performances of GAVD, GRGA and NSGA-II are measured, with respect to the goals of multi-objective optimisation (convergence to the Pareto front and diversity across it), using the convergence metric (γ) and diversity metric (Deb *et al.*, 2000) (Appendix C). The lower the values of these metrics, the better is the performance of the given optimisation algorithm.

8.2.1 Case-1

The performances of GAVD, GRGA and NSGA-II are compared on Case-1, considering the two variable dependency scenarios (1.1 and 1.2).

8.2.1.1 Dependency Scenario 1.1

In this case, NSGA-II and GRGA work on x_1 and x_2 as the independent variables, while ignoring the dependency data provided. On the other hand, GAVD performs RA on the given data to obtain the dependency equation. It considers x_1 , x_3 and x_4 as independent variables, and determines x_2 using the estimated dependency equation. The results obtained from NSGA-II, GRGA and GAVD are depicted in Figure 8.9. The γ and Δ values that are obtained in this case are listed in Table 8.10.

Table 8.10: Performance Metrics in Case-1 (Dependency Scenario 1.1)

Case-1		Performance Metrics	
Dependency Scenario 1.1		γ	Δ
Optimisation Algorithms	NSGA-II	1.209567	0.090002
	GRGA	0.009143	0.080121
	GAVD	0.008221	0.081124

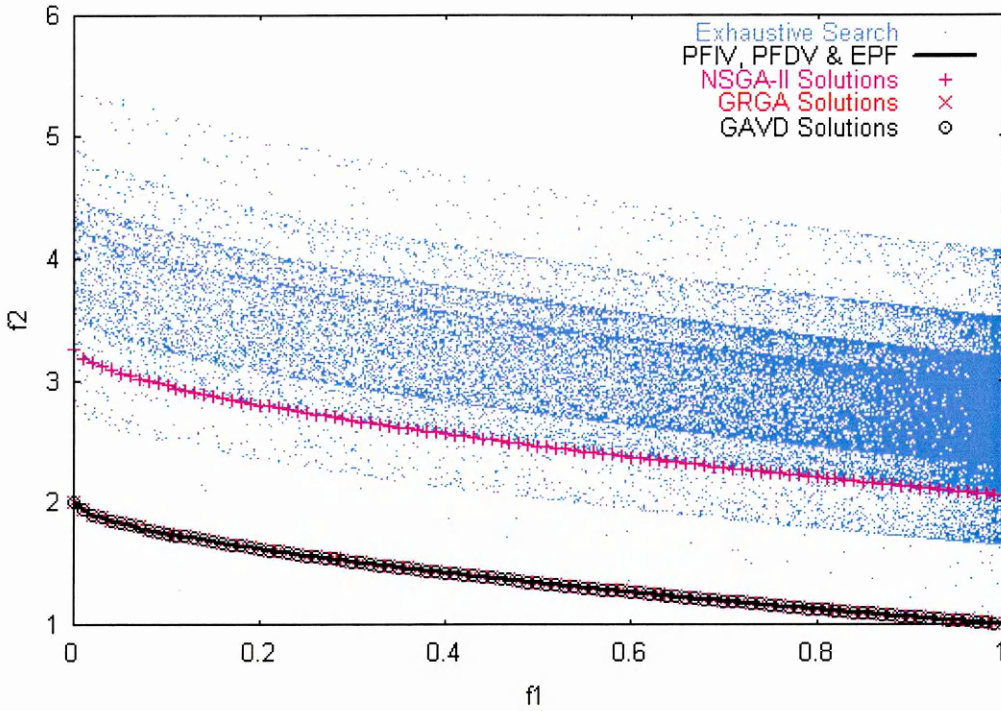


Figure 8.9: GAVD Performance in RETB-II Case-1 (Dependency Scenario 1.1)
 (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)

8.2.1.2 Dependency Scenario 1.2

The classification (dependent/independent) of variables used in the application of NSGA-II, GRGA and GAVD is similar to the one used in the Dependency Scenario 1.1. The results obtained from NSGA-II, GRGA and GAVD are depicted in Figure 8.10. The γ and Δ values that are obtained in this case are listed in Table 8.11.

Table 8.11: Performance Metrics in Case-1 (Dependency Scenario 1.2)

Case-1		Performance Metrics	
Dependency Scenario 1.2		γ	Δ
Optimisation Algorithms	NSGA-II	0.436589	0.090002
	GRGA	0.856745	0.080121
	GAVD	0.000356	0.078659

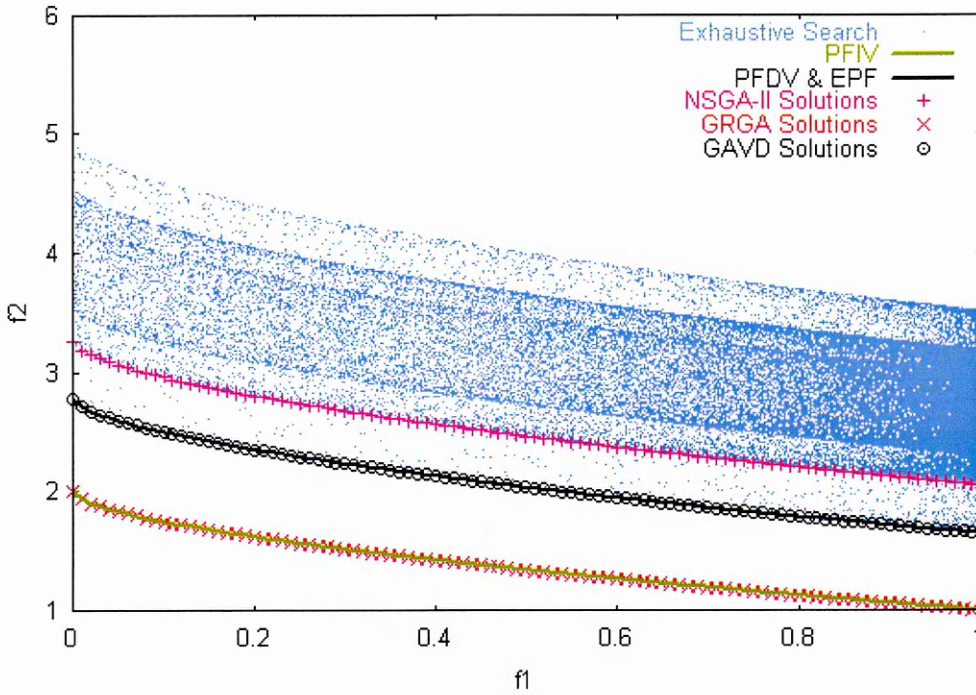


Figure 8.10: GAVD Performance in RETB-II Case-1 (Dependency Scenario 1.2)
(Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent
Variables: PFDV, Estimated Pareto Front: EPF)

8.2.2 Case-2

The results obtained from GAVD, GRGA and NSGA-II on Case-2 are illustrated here.

8.2.2.1 Dependency Scenario 2.1

In this case, NSGA-II and GRGA are applied with x_1 , x_2 and x_3 as the independent variables, while ignoring the dependency data provided. On the other hand, GAVD performs RA on the given data to obtain the dependency equation. It considers x_1 and x_3 as independent variables, and determines x_2 using the estimated dependency equation. The results obtained from NSGA-II, GRGA and GAVD are depicted in Figure 8.11, and the corresponding γ and Δ values are given in Table 8.12.

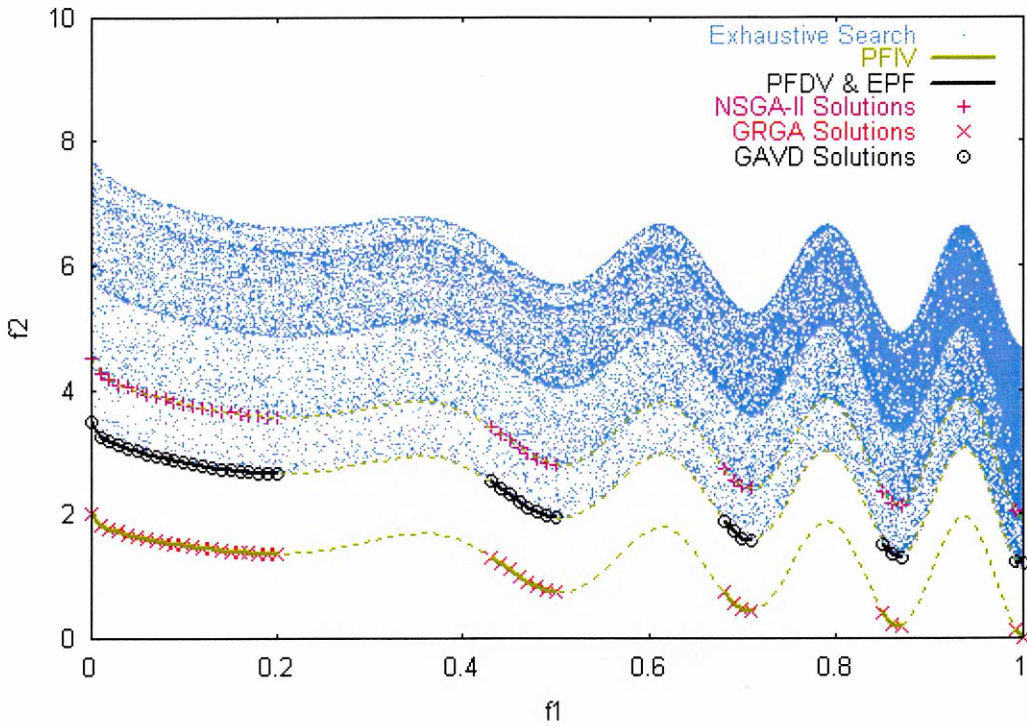


Figure 8.11: GAVD Performance in RETB-II Case-2 (Dependency Scenario 2.1)
 (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)

Table 8.12: Performance Metrics in Case-2 (Dependency Scenario 2.1)

Case-2		Performance Metrics	
Dependency Scenario 2.1		γ	Δ
Optimisation Algorithms	NSGA-II	0.986345	0.083956
	GRGA	1.654703	0.045431
	GAVD	0.001373	0.014564

8.2.2.2 Dependency Scenario 2.2

The classification (dependent/independent) of variables used in the application of NSGA-II, GRGA and GAVD is similar to the one used in Scenario 2.1. The results obtained from NSGA-II, GRGA and GAVD are depicted in Figure 8.12, and the corresponding γ and Δ values are shown in Table 8.13.

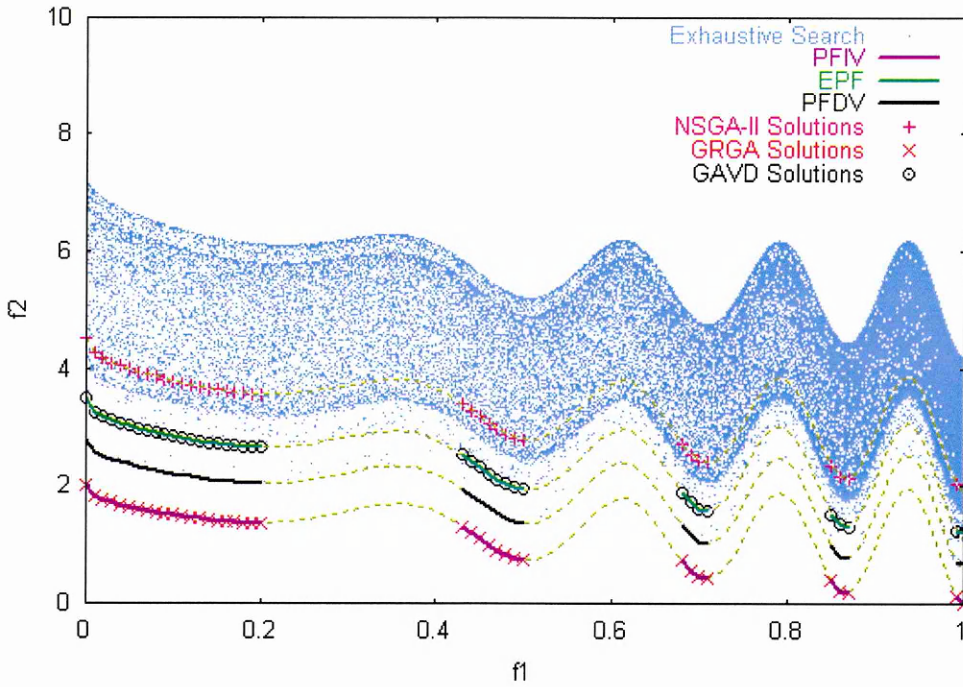


Figure 8.12: GAVD Performance in RETB-II Case-2 (Dependency Scenario 2.2)
 (Pareto Front for Independent Variables: PFIV, Pareto Front for Dependent Variables: PFDV, Estimated Pareto Front: EPF)

Table 8.13: Performance Metrics in Case-2 (Dependency Scenario 2.2)

Case-2 Dependency Scenario 2.2		Performance Metrics	
		γ	Δ
Optimisation Algorithms	NSGA-II	1.394502	0.093956
	GRGA	0.759007	0.045432
	GAVD	0.742356	0.045831

8.2.3 Case-3

Here, the performances of GAVD, GRGA and NSGA-II on Case-3 are depicted.

Table 8.14: GAVD Performance in RETB-II Case-3 (Dependency Scenario 3.1)
(Upper Diagonal Graphs: Pareto Front with NSGA-II and GRGA Solutions, Lower Diagonal Graphs: Pareto Front with GAVD Solutions)

f_1 (Range: 0-1)	Figure 8.13(a)	Figure 8.13(b)	Figure 8.13(c)
Figure 8.14(a)	f_2 (Range: 0-1)	Figure 8.13(d)	Figure 8.13(e)
Figure 8.14(b)	Figure 8.14(c)	f_3 (Range: 0-1)	Figure 8.13(f)
Figure 8.14(d)	Figure 8.14(e)	Figure 8.14(f)	f_4 (Range: 0-5)

8.2.3.1 Dependency Scenario 3.1

In this case, NSGA-II and GRGA are applied with x_1, x_2, x_3, x_4 and x_5 as the independent variables, while ignoring the dependency data provided. On the other hand, GAVD performs RA on the given data to obtain the dependency equation. It considers $x_1, x_2, x_3,$ and x_5 as independent variables, and determines x_4 using the estimated dependency equation. The results obtained from NSGA-II, GRGA and GAVD are depicted in Table 8.14, and the corresponding γ and Δ values are shown in Table 8.15.

Table 8.15: Performance Metrics in Case-3 (Dependency Scenario 3.1)

Case-3 Dependency Scenario 3.1		Performance Metrics	
		γ	Δ
Optimisation Algorithms	NSGA-II	0.220678	0.129059
	GRGA	0.198456	0.110856
	GAVD	0.124536	0.100985

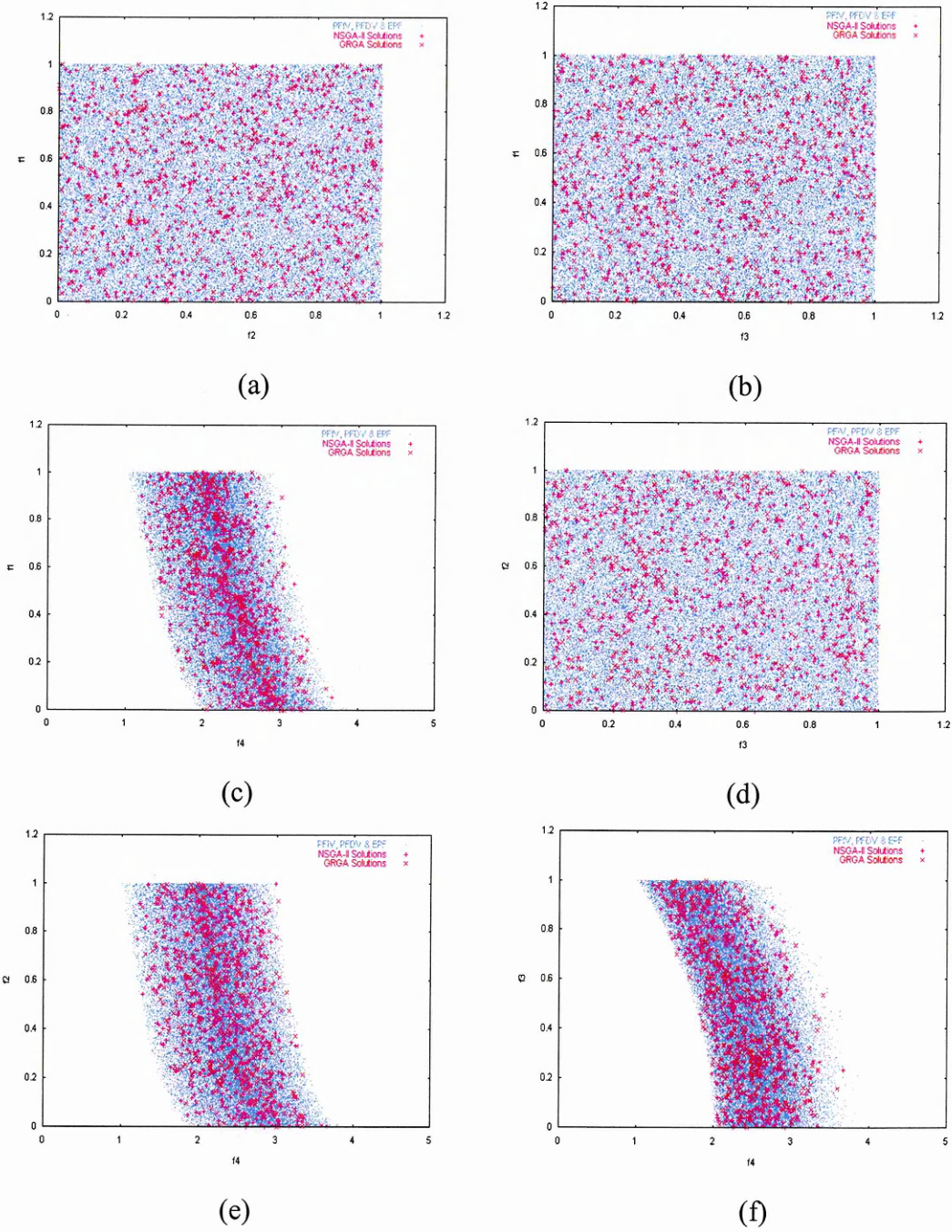


Figure 8.13: NSGA-II and GRGA Performance in RETB-II Case-3 (Dependency Scenario 3.1) – (a) f_1 - f_2 Graph (b) f_1 - f_3 Graph (c) f_1 - f_4 Graph (d) f_2 - f_3 Graph (e) f_2 - f_4 Graph (f) f_3 - f_4 Graph

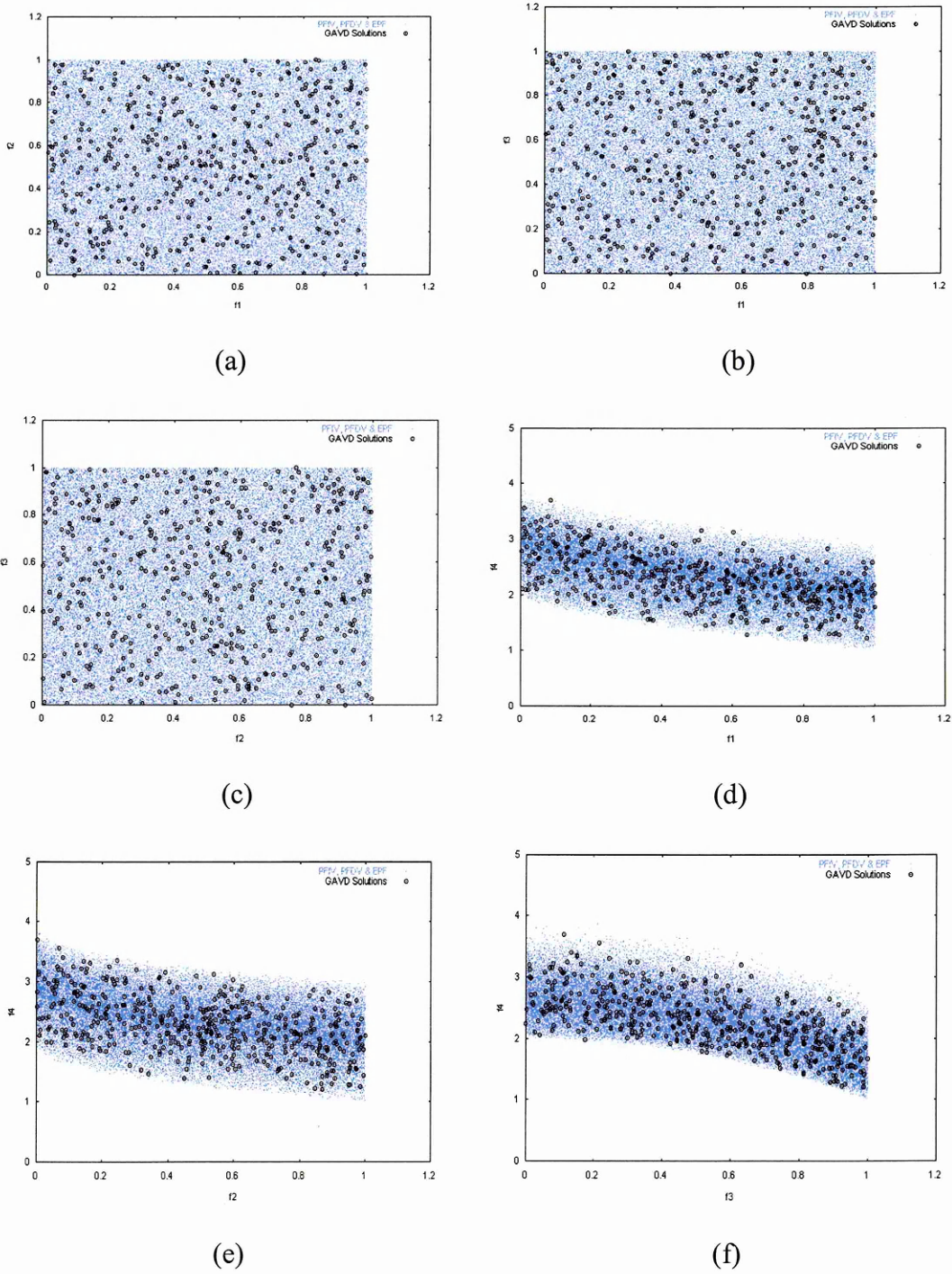
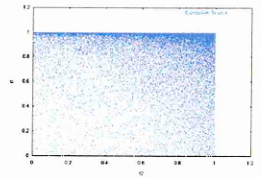
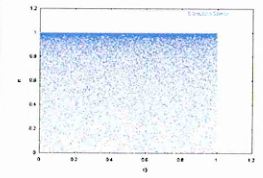
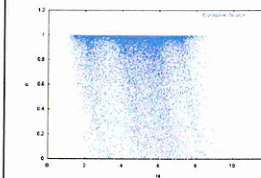
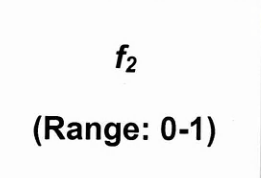
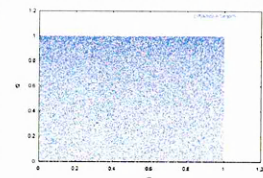
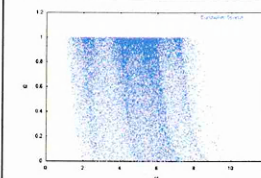


Figure 8.14: GAVD Performance in RETB-II Case-3 (Dependency Scenario 3.1) –
 (a) f_2 - f_1 Graph (b) f_3 - f_1 Graph (c) f_3 - f_2 Graph (d) f_4 - f_1 Graph (e) f_4 - f_2 Graph (f)
 f_4 - f_3 Graph

Table 8.16: GAVD Performance in RETB-II Case-3 (Dependency Scenario 3.2)
 (Upper Diagonal Graphs: Search Space with Variable Dependency, Lower Diagonal
 Graphs: Pareto Front with NSGA-II, GRGA and GAVD Solutions)

f_1 (Range: 0-1)			
Figure 8.15(a)	f_2 (Range: 0-1)		
Figure 8.15(b)	Figure 8.15(c)	f_3 (Range: 0-1)	
Figure 8.15(d)	Figure 8.15(e)	Figure 8.15(f)	f_4 (Range: 0-7)

8.2.3.2 Dependency Scenario 3.2

The classification (dependent/independent) of variables used in the application of NSGA-II, GRGA and GAVD is similar to the one used in Scenario 3.1. The results obtained from NSGA-II, GRGA and GAVD are depicted in Table 8.16, and corresponding γ and Δ values are given in Table 8.17.

Table 8.17: Performance Metrics in Case-3 (Dependency Scenario 3.2)

Case-3 Dependency Scenario 3.2		Performance Metrics	
		γ	Δ
Optimi- sation Algori- thms	NSGA-II	1.270974	0.198407
	GRGA	0.284569	0.153057
	GAVD	0.200846	0.145223

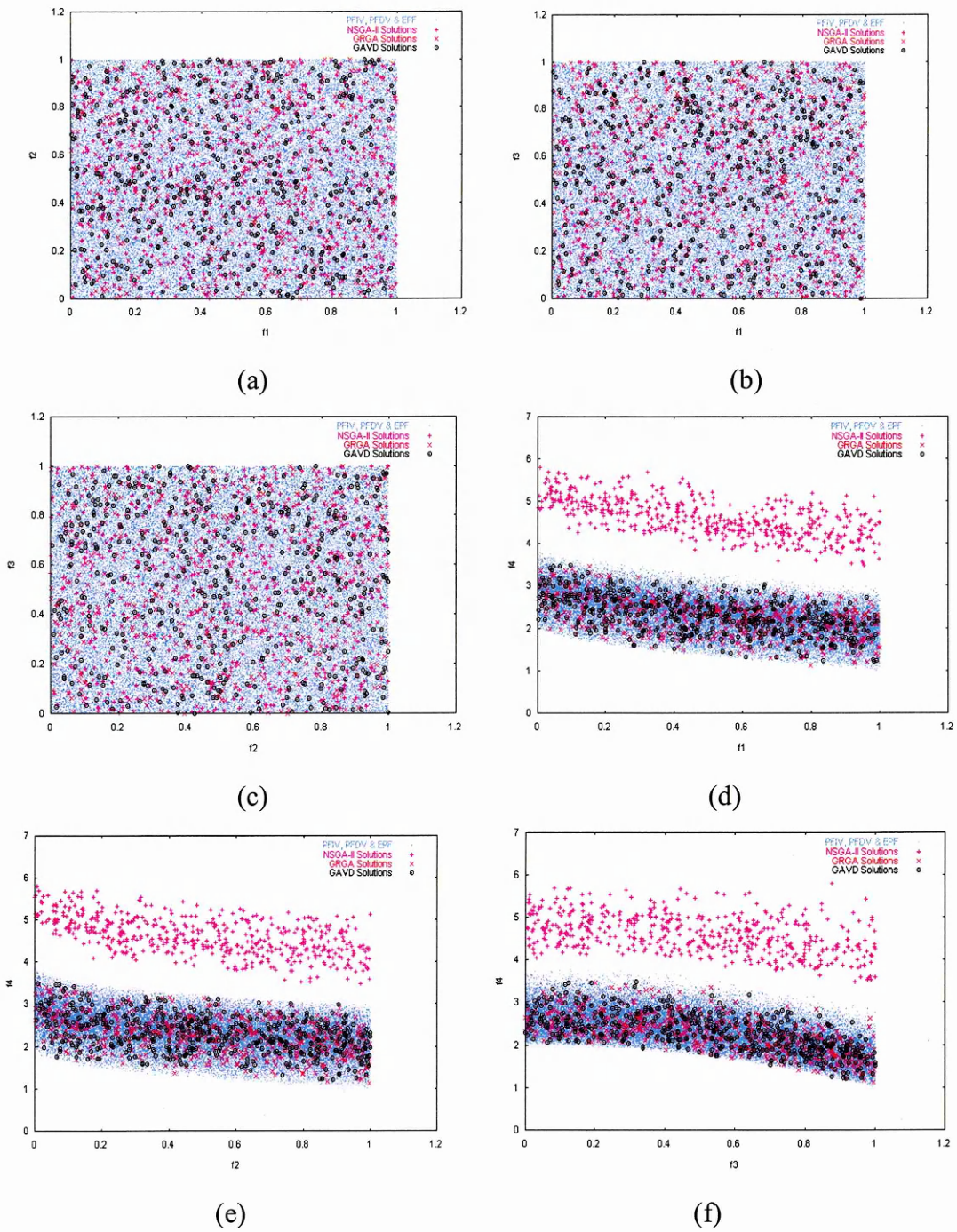


Figure 8.15: GAVD Performance in RETB-II Case-3 (Dependency Scenario 3.2) – (a) f_2 - f_1 Graph (b) f_3 - f_1 Graph (c) f_3 - f_2 Graph (d) f_4 - f_1 Graph (e) f_4 - f_2 Graph (f) f_4 - f_3 Graph

8.3 Discussion of Results

The results obtained from each of the three RETB/RETB-II case studies are discussed below.

8.3.1 Case-1

Here, GAVD, GRGA and NSGA-II are tested using the two scenarios of variable dependence (1.1 and 1.2), discussed in Section 8.1.

8.3.1.1 Dependency Scenario 1.1

Based on the γ and Δ values, the following observations can be made regarding this problem and the performances of GAVD, GRGA and NSGA-II. The results discussed here are pictorially depicted in Figure 8.9.

- ◆ Since the dependency relationship (Equation 8.4) covers the full range of x_2 , it does not alter the Pareto front. Therefore, the Pareto fronts for the original problem (with no dependence) and the dependent-variable problem coincide with each other.
- ◆ GRGA and NSGA-II do not incorporate variable dependence in their solution strategies. However, since the original and the new Pareto fronts are coincident in this case, the GRGA is able to locate the Pareto front ($\gamma = 0.009143$, $\Delta = 0.080121$). However, NSGA-II gets trapped in one of the local fronts, and so is not able to locate the global front ($\gamma = 1.209567$, $\Delta = 0.090002$), giving a much higher value of γ as compared to that given by GRGA. This is because the convergence strategy (Pareto domination cum elitism) used by NSGA-II ceases to produce the driving force towards the global front once most of the solutions of the population share the same non-domination level. Therefore, in this case, the NSGA-II has prematurely converged to a local front. This situation is avoided in GRGA through the artificial modification of regression coefficients at regular intervals using their history of search observed in previous generations. This guides the search towards the global Pareto front by preventing it from getting trapped in local fronts.

- ◆ The equation that is constructed in this scenario is of degree two, with no cyclic and deceptive terms (Equation 8.4), making it possible for the GAVD (that uses quadratic RA) to exactly model the dependence. Hence, the Pareto front that the GAVD sees coincides with the true Pareto front. Furthermore, since GAVD uses GRGA as its optimisation engine, it is able to converge to the Pareto front and distribute the solutions uniformly across the front ($\gamma = 0.008221$, $\Delta = 0.081124$). This is reflected by the small values of γ and Δ attained from GAVD. These values are also similar to those attained from GRGA.

8.3.1.2 Dependency Scenario 1.2

Based on the γ and Δ values, the following observations can be made regarding this problem and the performances of GAVD, GRGA and NSGA-II. The results discussed here are pictorially depicted in Figure 8.10.

- ◆ In this case, since the minimum value of the given x_2 is 0.2 (Equation 8.5), it does not cover its original range (in which the minimum value was 0) when there is no dependence among the decision variables. This truncates a part of the original search space. Furthermore, the Pareto front gets modified since the original one corresponded to x_2 equal to 0.
- ◆ Here, GRGA converges to the global Pareto front of the original problem (with no dependence among its variables). However, these results are infeasible since the original Pareto front has become infeasible with the introduction of variable dependence. Since the new Pareto front does not coincide with the original one, GRGA exhibits poor convergence (high γ) in this case ($\gamma = 0.856745$, $\Delta = 0.080121$). Similar to the results shown in the previous scenario, NSGA-II gets trapped in a local front in this problem. However, incidentally in this problem, this particular local front lies in the feasible region of the search space modified by variable dependence. Therefore, the results from NSGA-II are feasible but are sub-optimal with respect to the new Pareto front ($\gamma = 0.436589$, $\Delta = 0.090002$). Furthermore, the average distance between the new Pareto front and the original Pareto front at which GRGA converges is more than that between the new Pareto front and the local front at which NSGA-II converges. Therefore, the performance metric γ shows better convergence for NSGA-II as compared to GRGA.

- ◆ Due to the use of quadratic RA in GAVD, it estimates the given dependency relationship (Equation 8.5) as the following equation (Equation 8.14). This is pictorially depicted in Figure 8.3. Since this equation has the same range as the given dependency equation, the estimated search space has identical boundaries to the actual search space under variable dependency. Therefore, the estimated Pareto front coincides with the actual one. Hence, the use of GRGA in GAVD is able to ensure that the solutions converge to the new Pareto front and distribute evenly across it. This gives very small values for γ and Δ ($\gamma = 0.000356$, $\Delta = 0.078659$) in this case.

$$x_2 = 1 - 0.4x_3 - 0.2x_4 - 0.2x_4^2, \forall 0 \leq x_3 \leq 1, \forall 0 \leq x_4 \leq 1. \quad \text{Equation 8.14}$$

8.3.2 Case-2

Here, GAVD, GRGA and NSGA-II are tested using the two scenarios of variable dependence (2.1 and 2.2), discussed in Section 8.1.

8.3.2.1 Dependency Scenario 2.1

The features of this problem together with the performances of GAVD, GRGA and NSGA-II are discussed here. These results are illustrated in Figure 8.11.

- ◆ In this problem, the original Pareto front occurs when both x_2 and x_3 are equal to 0. Due to the given dependency among these variables, this is no longer possible. This causes modifications in the search space and the Pareto front based on the nature of the given dependency equation (Equation 8.8 and Figure 8.11).
- ◆ Here as well, GRGA converges to the global Pareto front of the original problem (with no dependence among its decision variables). However, since the new Pareto front does not coincide with the original one, the results from GRGA are not feasible in this case ($\gamma = 1.654703$, $\Delta = 0.045431$). Similar to the previous case, NSGA-II gets trapped on a local front, which incidentally lies in the new search space. However, its results are sub-optimal with respect to the new Pareto front ($\gamma = 0.986345$, $\Delta = 0.083956$). Furthermore, since the average distance from the new Pareto front is more to the original Pareto front (at which GRGA converges) than to the local front at which NSGA-II converges, the performance metric γ shows better convergence for NSGA-II as compared to GRGA.

- ◆ Also, since GAVD uses quadratic RA, it is able to exactly determine the dependency equation in this case (Equation 8.8). Hence, the Pareto front seen by GAVD is the same as that of the actual dependent-variable problem. Therefore, GRGA used in GAVD converges to the Pareto front and distributes the solutions uniformly across the front ($\gamma = 0.001373$, $\Delta = 0.014564$). Hence, very small values for γ and Δ are attained.

8.3.2.2 Dependency Scenario 2.2

The following observations can be made on the basis of the results obtained by applying GAVD, GRGA and NSGA-II on this problem. These results are pictorially represented in Figure 8.12.

- ◆ In this problem, when there is no dependence among the decision variables, the Pareto front occurs when both x_2 and x_3 are equal to 0. Due to the given dependency among these variables (Equation 8.9), this is no longer possible. This leads to modifications in the search space and the Pareto front.
- ◆ Here also, GRGA converges to the global Pareto front of the original independent-variable problem. However, since the Pareto front with independent variables does not coincide with the one in the presence of variable dependence, the results attained by GRGA become infeasible in this problem ($\gamma = 0.759007$, $\Delta = 0.045432$). Similar to the previous case, NSGA-II gets trapped in a local front that lies in the new search space (obtained by introduction of variable dependence) ($\gamma = 1.394502$, $\Delta = 0.093956$). In this case, GRGA exhibits better convergence (smaller γ) than NSGA-II since here the average distance from the new Pareto front to NSGA-II solutions is about twice that from the new Pareto front to GRGA solutions.
- ◆ Due to the use of two-degree RA in GAVD, it estimates the given dependency relationship (Equation 8.9) as the following equation (Equation 8.15). This approximation is pictorially depicted in Figure 8.6. Since this estimated equation does not model the spike in the relationship, it only provides an approximation. Also, in this problem, the Pareto front occurs for the variable values corresponding to the bottom of the spike. Therefore, the search space and the Pareto front that GAVD sees are different from those in the given dependent-variable problem. Hence, GAVD is not able to locate the new Pareto front due to the limitations posed by the degree of RA that it uses ($\gamma = 0.742356$, $\Delta =$

0.045831). However, since GRGA and GAVD solutions lie at almost the same average distance from the new Pareto front, the γ values corresponding to them are almost equal to each other.

$$x_2 = 0.204 + 3.351x_3 - 3.351x_3^2, \forall 0 \leq x_3 \leq 1. \quad \text{Equation 8.15}$$

8.3.3 Case-3

Here, two scenarios of variable dependence (3.1 and 3.2), as discussed in Section 8.1, are used for analysing the performances of GAVD, GRGA and NSGA-II.

8.3.3.1 Dependency Scenario 3.1

The pictorial representation of this test is shown in Table 8.14. The following comments can be made based on this information.

- ◆ In this case, since the dependency relationship (Equation 8.12) covers the full range of x_2 , it does not alter the Pareto front. Therefore, the Pareto fronts for the original problem (with no dependence) and the dependent-variable problem coincide with each other. It should be noted that due to the multi-dimensional nature of the function search space in this problem, the Pareto front cannot be depicted by the two-dimensional plots of Table 8.14.
- ◆ This problem does not have multiple local fronts. Furthermore, since the original and the new Pareto fronts are coincident, GRGA and NSGA-II are also able to locate the front (GRGA: $\gamma = 0.198456$, $\Delta = 0.110856$; NSGA-II: $\gamma = 0.220678$, $\Delta = 0.129059$). Therefore, both GRGA and NSGA-II give small values of γ and Δ , such that the values given by the two algorithms are nearly equal to each other.
- ◆ Furthermore, the dependency equation in this scenario is of degree two (Equation 8.12), with no cyclic and deceptive terms, making it possible for the GAVD to exactly model the dependence (using piece-wise quadratic RA). Hence, the Pareto front that the GAVD sees coincides with the true Pareto front. Therefore, the optimisation engine of GAVD (GRGA) converges to the Pareto front and distributes the solutions uniformly across the front ($\gamma = 0.124536$, $\Delta = 0.100985$). This gives small values for γ and Δ . In this problem, although all the three algorithms exhibit good convergence to the Pareto front, it is not possible to

visually see the results due to the multi-dimensional nature of the function search space.

8.3.3.2 Dependency Scenario 3.2

The features of this problem together with the performances of GAVD, GRGA and NSGA-II are discussed here. These results are illustrated in Table 8.17.

- ◆ The search space shown in Table 8.9 does not clearly show the discontinuity in this case. This is because of its cyclical nature (due to the choice of a cyclic I function in RETB) that causes overlap of points in the two-dimensional plot, thereby hindering the depiction of discontinuity in the plot. Further, the mildness of the discontinuity also makes the discontinuity less prominent in this case.
- ◆ With the introduction of variable dependence (Equation 8.13), the maximum value of the given x_4 becomes 0.9, and so it does not cover its original range (in which the maximum value was 1) when there was no dependence among the decision variables. This truncates a part of the original search space. However, since the Pareto front corresponds to x_4 equal to 0, which still is a part of the search space, there is no modification in it with the introduction of variable dependency. Therefore, the Pareto fronts for the original problem (with no dependence) and the dependent variable problem coincide with each other.
- ◆ Since the original and the new Pareto fronts are coincident, GRGA is able to locate the Pareto front in this case ($\gamma = 0.284569$, $\Delta = 0.153057$). However, due to the presence of multiple local fronts, NSGA-II again exhibits pre-mature convergence to a local front, thereby giving values of γ that are much higher than those given by GRGA ($\gamma = 1.270974$, $\Delta = 0.198407$).
- ◆ Furthermore, the dependency equation in this scenario is of degree two (Equation 8.13)), with no cyclic and deceptive terms, making it possible for the GAVD to exactly model the dependence (using piece-wise quadratic RA). Hence, the Pareto front that the GAVD sees coincides with the true Pareto front. Therefore, GAVD converges to the Pareto front and distributes the solutions uniformly across the front ($\gamma = 0.200846$, $\Delta = 0.145223$). It should be noted here that in general the results obtained for this scenario are inferior as compared to those obtained from the previous one. This can mainly be attributed to the presence of multiple local fronts in this scenario.

8.4 Key Results

A review of the results obtained from all the three cases reveals that Case-3 is particularly challenging for all optimisation algorithms. This is because of the presence of four objectives in this problem that lends multi-dimensional nature to its Pareto front. The complexity of this problem is further enhanced due to the presence of bias in the search space (in both Dependency Scenarios 3.1 and 3.2) and multiple local fronts (only in Dependency Scenario 3.2). Furthermore, the presence of variable dependency introduces discontinuity in the search spaces corresponding to both the dependency scenarios. Therefore, it is observed that, although GAVD is able to locate points close to the global Pareto front, the convergence of the solutions to the Pareto front and their distribution across the front are inferior to that exhibited by it in those scenarios of Cases 1 and 2, where it is able to exactly model the dependency relationship among the decision variables. This is supported by comparatively higher values of γ and Δ exhibited by GAVD in the Dependency Scenarios 3.1 and 3.2 (Table 8.15 and Table 8.17). This observation particularly revealed that the multi-dimensional nature of the Pareto front has a strong impact on the difficulty of a problem.

In addition to the results illustrated in Section 8.2, the GAVD and GRGA also identify the following relationships among the decision variables corresponding to the Pareto-optimal solutions.

- ◆ Case-1 (Dependency Scenario 1.1): True Pareto front corresponds to $x_2 = 0$; $x_3 = 1$; $x_4 = 1$; with x_1 taking values in its range.
 - GRGA: Estimated Pareto front corresponds to $x_2 = 0$; with x_1 taking values in its range.
 - GAVD: Estimated Pareto front corresponds to $x_2 = 0$; $x_3 = 1$; $x_4 = 1$; with x_1 taking values in its range.
- ◆ Case-1 (Dependency Scenario 1.2): True Pareto front corresponds to $x_2 = 0.25$; $x_3 = 1$; $x_4 = 0.91$; with x_1 taking values in its range.
 - GRGA: Estimated Pareto front corresponds to $x_2 = 0$; with x_1 taking values in its range.

- *GAVD: Estimated Pareto front corresponds to $x_2 = 0.25$; $x_3 = 1$; $x_4=0.91$; with x_1 taking values in its range.*
- ◆ Case-2 (Dependency Scenario 2.1): True Pareto front corresponds to $x_2 = 0.2$; $x_3 = 0$; with x_1 taking values in its range.
 - *GRGA: Estimated Pareto front corresponds to $x_2 = 0$; $x_3 = 0$; with x_1 taking values in its range.*
 - *GAVD: Estimated Pareto front corresponds to $x_2 = 0.2$; $x_3 = 0$; with x_1 taking values in its range.*
- ◆ Case-2 (Dependency Scenario 2.2): True Pareto front corresponds to $x_2 = 0.2$; $x_3 = 0.5$; with x_1 taking values in its range.
 - *GRGA: Estimated Pareto front corresponds to $x_2 = 0$; $x_3 = 0$; with x_1 taking values in its range.*
 - *GAVD: Estimated Pareto front corresponds to $x_2 = 0.2$; $x_3 = 0$; with x_1 taking values in its range.*
- ◆ Case-3 (Dependency Scenario 3.1): True Pareto front corresponds to $x_4 = 0$; $x_5 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.
 - *GRGA: True Pareto front corresponds to $x_4 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.*
 - *GAVD: True Pareto front corresponds to $x_4 = 0$; $x_5 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.*
- ◆ Case-3 (Dependency Scenario 3.2): True Pareto front corresponds to $x_4 = 0$; $x_5 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.
 - *GRGA: True Pareto front corresponds to $x_4 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.*
 - *GAVD: True Pareto front corresponds to $x_4 = 0$; $x_5 = 0$; with x_1 , x_2 and x_3 taking values in their respective ranges.*

The tests reported in this section lead to the following general conclusions regarding the performances of GAVD and GRGA.

- ◆ If the introduction of variable dependency does not change the Pareto front, it is observed that GRGA and GAVD exhibit similar performance. This is because GAVD uses GRGA as its optimisation engine.
- ◆ GRGA exhibits better performance than NSGA-II in dealing with multi-objective optimisation problems that have complex inseparable function interaction,

leading to a variety of features such as multiple local fronts, bias, deception and discontinuous Pareto fronts. This is because the Pareto-domination/elitism strategy used by NSGA-II ceases to produce the driving force towards the global Pareto front once most of the solutions of the population share the same non-domination level. GRGA addresses this drawback of NSGA-II through periodic modification of regression coefficients based on their history of search observed in previous generations. Unlike NSGA-II, GRGA also addresses the core issue involved in maintaining diversity by re-distributing the solutions based on the relationships among their decision variables. However, since GRGA does not have an in-built mechanism for handling variable dependence, it fails to give satisfactory solutions in those problems that have dependence among their decision variables.

- ◆ GAVD removes the above-mentioned drawback of GRGA by providing it with a mechanism for handling variable dependence in multi-objective optimisation problems. This algorithm is capable of dealing with a variety of optimisation problems. However, the capability of GAVD is limited by the degree of the RA that it uses. Here, a quadratic RA has been used.

8.5 Summary

This chapter has analysed the performances of the two algorithms GAVD and GRGA that have been proposed in this research. The performance analysis has been carried out using the test beds RETB and RETB-II, which have also been developed in this research. This chapter has achieved the dual purpose of analysing the performances of GAVD and GRGA, while validating the behaviour of RETB and RETB-II. In short, this chapter has achieved the following.

- ◆ It has developed a set of RETB/RETB-II case studies such that it represents three features of real-life engineering design optimisation problems: presence of multiple objectives, constraints and variable interaction.
- ◆ It reported the experimental results obtained from GAVD, GRGA and NSGA-II in each of the cases.
- ◆ It has presented an analysis of the experimental results.

- ◆ Finally, based on this analysis, it has compiled key observations and drawn general conclusions regarding the performances of GRGA and GAVD.

This chapter has analysed the performance of GRGA and GAVD using RETB and RETB-II. The next chapter validates the observations made in this chapter using a representative set of real-life case studies in the area of engineering design optimisation. A brief analysis of the features of a selection of real-life problems is also presented in the next chapter.

9 REAL-LIFE CASE STUDIES

In the previous chapters, two algorithms, GRGA and GAVD, were proposed for dealing with multiple objectives, constraints and variable interaction in engineering design optimisation problems. Two test beds, RETB and RETB-II, were also proposed that have the same features as mentioned above, and enable controlled testing of optimisation algorithms. These test beds were used in the previous chapter to analyse the performance of GRGA and GAVD. The aim of this chapter is to validate the observations made in the previous chapter using real-life case studies. This chapter attempts to achieve the following.

- ◆ *To analyse a number of case studies from real-life engineering design optimisation.*
- ◆ *To frame the selection criteria for choosing a representative set of case studies for this research.*
- ◆ *To report the experimental results obtained from GRGA and GAVD.*
- ◆ *To analyse these results in order to validate the performance of GRGA and GAVD.*

9.1 Case Studies from Real-life Engineering Design Optimisation

The aim of this section is to present some case studies from real-life engineering design optimisation and to analyse the features of these case studies, especially with respect to variable interaction. These case studies are drawn from Table 4.1, which presents the list of some applications of evolutionary-based optimisation algorithms reported in the literature. This chapter analyses 10 problems chosen from Table 4.1. Since all these problems share similar features, it can be said that the successful application of GRGA and GAVD on a representative set of these 10 problems also ensures their success in solving other problems listed in Table 4.1.

The features of the chosen 10 problems are tabulated in Table 9.1. As can be seen, this table gives the general features of these problems in terms of the number and nature of variables, and the number of objectives and constraints. It further looks at the problem features that influence interaction among the decision variables. Since inseparable function interaction is caused by the functions used in the problem, this table analyses the complexity of objectives and constraints. It also reports the complexity of the Pareto front, together with the relationships involving decision variables that define the Pareto front. The table also checks for the presence and nature of variable dependence in the problems. Finally, the table presents some published results obtained from the application of optimisation algorithms on these problems. The following gives a brief description of these case studies.

- ◆ **Compound Gear Train Design (Deb, Pratap and Moitra, 2000):** This problem involves the design of a compound gear train to achieve a specific gear ratio between the driver and driven shafts. The objective of the gear train design is to find the number of teeth in each of the four gears so as to minimise: (i) the error between the obtained gear ratio and a required gear ratio and (ii) the maximum size of any of the four gears. Since the number of teeth must be integers, all four variables are strictly integers, having bounds attached to them. This problem has a non-linear, concave and discontinuous Pareto front with bias in its search space. Furthermore, due to the presence of discrete variables, this problem has multiple local fronts in its search space. However, there is no variable dependence in this problem. This problem has been reported in the literature to be successfully solved by NSGA-II.
- ◆ **Design of a Helical Compression Spring (Deb, Pratap and Moitra, 2000):** Here, a helical compression spring needs to be designed for minimum volume and for minimum stress. This problem has three variables: the number of spring coils N , the wire diameter d and the mean coil diameter D . Of these variables, N is an integer variable, d is a discrete variable having forty-two non-equidistant values and D is a real-parameter variable. This problem has eight constraints that involve limits on variable values, restrictions on stress to be within the allowable strength and the volume to be within a pre-specified limit. The decision variables in this problem are independent. Furthermore, the Pareto front here is non-linear, convex and discontinuous, and the search space is biased in nature. Similar to the

previous problem, this problem also has multiple local fronts due to the presence of discrete variables. It has been shown in the literature that NSGA-II (with constrained domination) is able to locate the Pareto front in this problem.

- ◆ **Welded Beam Design (Deb, Pratap and Moitra, 2000):** In this problem, a beam needs to be welded on another beam and must carry a certain load (Figure 9.1). The objective of the design is to minimise the cost of fabrication and minimise the end deflection. Here, the overhang portion of the beam and the applied force (F) are specified, making the cross-sectional dimensions of the beam (b , t) and the weld dimensions (h , l) as the variables. This problem has four constraints. The first constraint makes sure that the shear stress developed at the support location of the beam is smaller than the allowable shear strength of the material. The second constraint makes sure that normal stress at the support location of the beam is smaller than the allowable yield strength of the material. The third constraint ensures that the thickness of the beam is not smaller than the weld thickness from a practical standpoint. Finally, the fourth constraint makes sure that the allowable buckling load of the beam is more than the applied load. This problem has a non-linear, convex and continuous Pareto front, and has no dependence among its decision variables. NSGA-II with constrained domination approach has been shown to converge to the Pareto front and to distribute the solutions uniformly across the front.

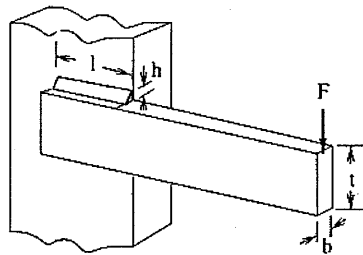


Figure 9.1: Welded Beam Design (Source: Deb, Pratap and Moitra, 2000)

- ◆ **Design of an I-beam (Coello, 1997):** The aim in this problem is to determine the dimensions of an I-beam such that the geometric and strength constraints are satisfied, and the following objectives are minimised: (i) cross-sectional area of the beam that reflects the volume for the given length (ii) static deflection of the beam for the displacement under the applied force. This problem has independent variables, a non-linear, convex and continuous Pareto front, and a biased search

space. In dealing with this problem, NSGA-II performs better than other multi-objective optimisation algorithms.

- ◆ **Determining Machining Parameters (Coello, 1997):** This problem requires to determine the values of the cutting speed, feed rate and depth of cut such that the surface roughness, scrap rate and tool wear are minimised, and the material removal rate is maximised. This problem has bounds on the cutting speed, feed rate and depth of cut to reflect the ranges over which the tests are run. It also has limits on the values of three performance measures: surface roughness, scrap rate and tool wear. Since all the objective functions in this problem are linear, it has an unbiased search space. Furthermore, the Pareto front is linear, multi-dimensional and continuous, and there is no dependence among the decision variables. In this problem as well, NSGA-II performs better than most other multi-objective optimisation algorithms.

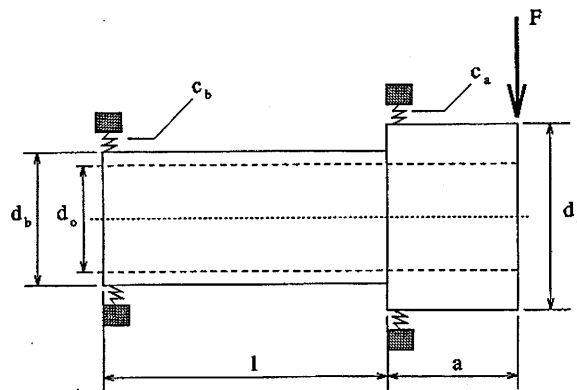


Figure 9.2: Design of a Machine Tool Spindle (Source: Coello, 1997)

- ◆ **Design of a Machine Tool Spindle (Coello, 1997):** This design is shown in Figure 9.2. The four variables in this problem are the dimensions of the spindle (l , d_o , d_a , d_b). Two of these variables are real (l , d_o) and the rest two are discrete (d_a , d_b). This problem involves the minimisation of the volume of the spindle and the static displacement under the force F . This problem has nine constraints that include variable bounds and limits on the maximum radial run-out of the spindle nose. This problem has a non-linear, convex and discontinuous Pareto front, and a biased search space. It also has multiple local fronts due to the presence of discrete variables. Another interesting aspect of this problem is that it has dependence between two of its decision variables that arises due to the designers' special preference regarding the proportion of these two variables. Due to the

presence of dependence among its decision variables, most of the optimisation algorithms fail to produce satisfactory results in this case. However, the performance of NSGA-II is found to be better than that achieved by other approaches.

- ◆ **Two-bar Truss Design (Deb, Pratap and Moitra, 2000):** Here, a truss has to be designed to carry a certain load without elastic failure. Thus, in addition to the objective of designing the truss for minimum volume, there are additional objectives of minimising the stresses in each of the two members. This two-objective optimisation problem has three variables: height of the truss and cross-sectional areas of the two members. This problem has four objectives that limit the stresses and the dimensions of the two members. It has a non-linear, convex and continuous Pareto front, and a biased search space. There is no dependence among the variables of this problem. Literature reports that in this problem NSGA-II is able to locate well distributed Pareto-optimal solutions.
- ◆ **Design of a Robot Arm (Coello, 1997):** Here, a two-member robot arm needs to be designed. This problem has four variables that include the two counterweights and their distances from the joints. The four objectives in this problem involve the minimisation of the torque and the reaction forces at the joints of the two members of the robot arm. This problem also has six constraints that restrain the movement of the arm, values of counterweights and the distances of counterweights from the joints. Here, the Pareto front is non-linear and multi-dimensional, the search space is biased in nature, and the variables are independent of each other. In this problem as well, NSGA-II performs better than other algorithms.
- ◆ **Design of a Single Screw Extruder (Cunha, 2000):** The four variables in this problem are the screw speed and the barrel temperatures in the initial, intermediate and final zones. This problem has five objectives, requiring the maximisation of output, and the minimisation of the length of screw required for melting, melt temperature, power consumption and mixing quality. It should be noted that here the objectives are not directly defined in terms of decision variables. They are defined in terms of some intermediate variables that in turn functions of the decision variables. This makes the objective functions multi-layered in nature. Furthermore, the equations that are used to calculate the values of objectives are implicit in this case. This means that the determination of their values requires an iterative procedure in many cases. Here, the Pareto front is

non-linear and multi-dimensional, the search space is biased in nature, and the variables are independent of each other. In this case, the literature does not report the results obtained from high-performing optimisation algorithms.

- ◆ Design of a Turbine Blade Cooling System (Roy, 1997): In order to maximise gas turbine engine performance and efficiency, turbine blades need to operate in an environment where the gas temperature is as high as possible. This temperature often exceeds the operational limits of the turbine blade materials. To ensure component integrity whilst operating at high gas temperatures, blade materials are cooled to safe operating temperature levels by passing relatively cool air through them and in more extreme cases, over them in the form of films. A small portion of the compressor exit airflow is utilised to cool the blades (Figure 9.3 and Figure 9.4). The temperature of this cooling air depends on the compressor pressure ratio and on the flight Mach number and temperature. The sacrifices for the blade cooling include the loss of work (and some loss of efficiency) due to the portion of the air taken from the compressor exit. Thus, this problem can be framed as multi-objective having four objectives.

- *Coolant mass flow for radial passage (W_{cr} in Kg/s).*
- *Coolant mass flow for film hole (W_{cf} in Kg/s).*
- *Metal temperature for gas side (T_{wg} in K).*
- *Metal temperature for film side (T_{wf} in K).*

Alternatively, the problem can also be framed in two objectives as follows.

- *Coolant mass flow for radial passage (W_{cr} in Kg/s).*
- *Metal temperature for gas side (T_{wg} in K).*

This problem has twelve variables as follows.

- *Type of geometry (Geom).*
- *Coefficient of discharge (radial passage) (C_{dr}).*
- *Heat transfer coefficient factor (radial passage) (F_{hc}).*
- *Inlet temperature (T_{c1}).*
- *Wall thickness (dth).*
- *Thermal conductivity of the blade material (k_w).*
- *Pressure ratio (between inlet and outlet of radial passage) ($R_p = P_{c1}/P_{c3}$).*
- *Perimeter ratio (radial passage) ($R_s = S_{gr}/S_{cr}$).*
- *Film hole diameter (df).*

- Coefficient of discharge (film hole) (C_{df}).
- Heat transfer coefficient factor (film hole) (F_f).
- Pressure ratio (film) ($R_{pf} = (P_{c1} - P_{c2}) / (P_{c1} - P_{c3})$).

Here, the first variable is discrete (plane, ribbed or pedestal) and the rest are real. Also, the values of C_{dr} and F_{hc} vary within a range according to the type of geometry. This problem also has 15 constraints that include limits on the above-mentioned variables, blade wall temperature (on the gas and film side) and flow ratio (W_{cr}/W_{cf}). Similar to the previous problem, the objective functions are implicit and multi-layered in this case. This problem has a non-linear and multi-dimensional Pareto front, and a biased search space. Literature does not report the application of any multi-objective optimisation algorithm on this problem.

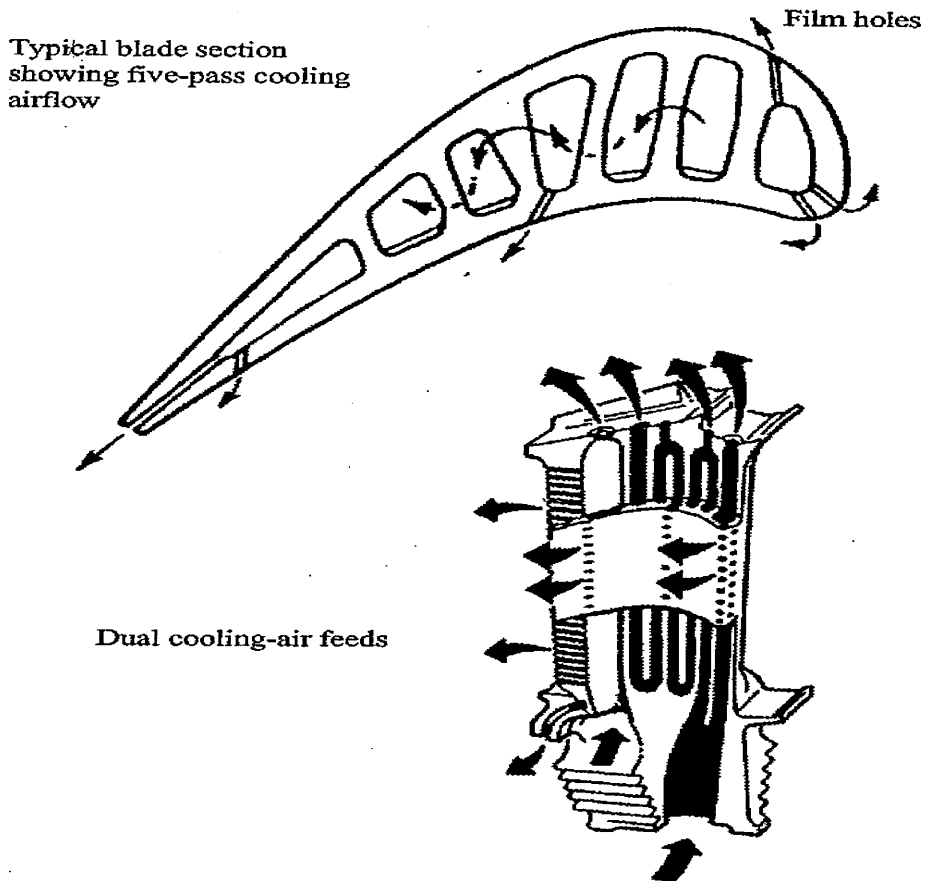


Figure 9.3: General Arrangement of Five-pass Cooling of Turbine Rotor Blade
(Source: Roy, 1997)

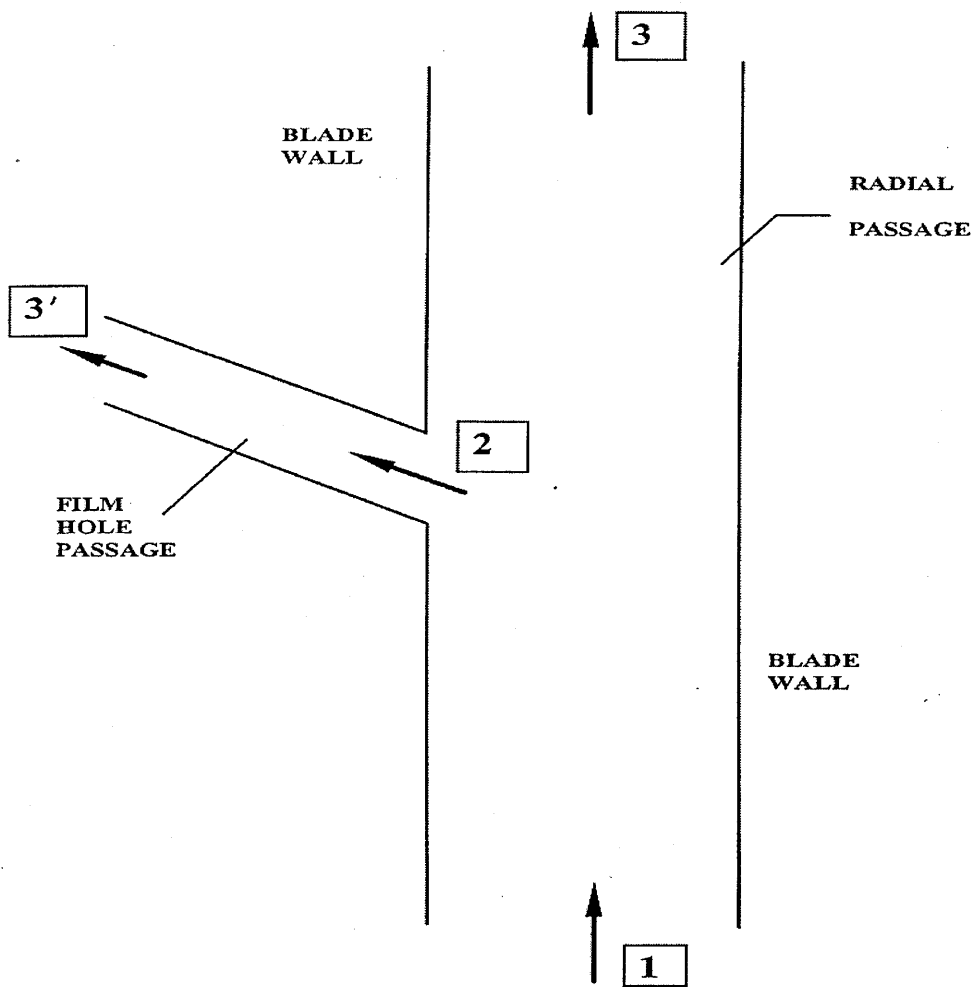


Figure 9.4: Schematic Diagram Showing General Arrangement of Coolant Flow through Turbine Blade with Film Cooling Mechanism (1: Coolant Air Inlet, 2: Film Cooling Passage Inlet, 3: Cooling Air Exit and 3': Film Cooling Hole Exit) (Source: Roy, 1997)

9.2 Selection of Case Studies

This section attempts to select a set of case studies from the ones mentioned in Table 9.1. These case studies are used in this chapter for validating the performance of GRGA and GAVD. Therefore, the aim here is to select a set of problems that together represent multiple objectives, constraints and variable interaction in engineering design optimisation problems. Based on the challenges that multiple

objectives, constraints, inseparable function interaction and variable dependence pose for optimisation algorithms, the following list can be compiled that identifies the various features that may be observed in real-life problems.

- ◆ Multiple variables.
- ◆ Integer, discrete and real variables.
- ◆ Multiple measures of performance (objectives).
- ◆ Multiple constraints.
- ◆ Polynomial, rational or complex objective functions.
- ◆ Implicit and multi-layered objective functions.
- ◆ Polynomial, rational or complex constraints.
- ◆ Implicit and multi-layered constraints.
- ◆ Unknown Pareto front
- ◆ Multi-dimensional Pareto front
- ◆ Non-linear (convex/concave) Pareto front
- ◆ Continuous or discontinuous Pareto front
- ◆ Biased search space
- ◆ Multi-front (multiple local Pareto fronts)
- ◆ Variable dependence

This list is used to select a set of problems from Table 9.1 such that all the above-mentioned features are represented. This analysis led to the choice of a set of three problems listed below (shown as shaded regions in Table 9.1).

- ◆ Design of a welded beam.
- ◆ Design of a machine tool spindle.
- ◆ Design of a turbine blade cooling system.

Table 9.1: Case Studies from Real-life Engineering Design Optimisation

Real-life Case Studies		Problem - 1	Problem - 2	Problem - 3	
		Compound Gear Train	Helical Compression Spring	Welded Beam	
General Problem Features	No. of Variables	4	3	5	
	Nature of Variables	4 integer	1 integer, 1 discrete & 1 real	5 real	
	No. of Objectives	2	2	2	
	No. of Constraints (+ Var. Bounds)	4	8	4	
Variable Interaction	Inseparable Function Interaction	Complexity of Objectives	1. Rational function (O(4)/O(4)) 2. Maximum function	1. Polynomial function (O(3)) 2. Rational function (O(2)/O(4))	1. Polynomial function (O(3)) 2. Rational function (O(0)/O(4))
		Nature of Constraints	4 linear	2 linear, 1 polynomial (O(4)) & 5 rational	1 linear, 1 polynomial (O(5)) & 2 rational
		Reported Pareto Front	Corresponds to fixed values for 2 variables	Unknown	Corresponds to fixed values of 3 variables
		Reported Complexity of Search Space	<ul style="list-style-type: none"> Non-linear, concave & discontinuous Pareto front Biased search space Multi-front (multiple local Pareto fronts) 	<ul style="list-style-type: none"> Non-linear, convex & discontinuous Pareto front Biased search space Multi-front (multiple local Pareto fronts) 	<ul style="list-style-type: none"> Non-linear, convex & continuous Pareto front Biased search space
	Variable Dependence	Independent decision variables	Independent decision variables	Independent decision variables	
Published Results from Optimisation Algorithms	Single Objective Optimisers	<ul style="list-style-type: none"> Order of performance: GeneAS-I > GeneAS-II > Augmented Lagrangian (AL) > Branch-and-Bound (BB) Sub-optimal results 	<ul style="list-style-type: none"> Order of performance: GeneAS \approx Branch-and-Bound (BB) Solutions lie on estimated Pareto front 	<ul style="list-style-type: none"> Simple GA Solution lies on estimated Pareto front 	
	Multi-objective Optimisers	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > NSGA Order of performance for distribution: NSGA-II > NSGA 	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II (with constrained domination) \approx NSGA-II (with penalty function) > NSGA Order of performance for distribution: NSGA-II (with constrained domination) > NSGA-II (with penalty function) > NSGA 	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II (with constrained domination) \approx NSGA-II (with penalty function) > NSGA Order of performance for distribution: NSGA-II (with constrained domination) > NSGA-II (with penalty function) > NSGA 	

Table 9.1: Case Studies from Real-life Engineering Design Optimisation (contd.)

Real-life Case Studies		Problem - 4	Problem - 5	Problem - 6	
		I-beam	Machining Parameters	Machine Tool Spindle	
General Problem Features	No. of Variables	4	3	4	
	Nature of Variables	4 real	3 real	2 real & 2 discrete	
	No. of Objectives	2	4	2	
	No. of Constraints (+ Var. Bounds)	5	6	9	
Variable Interaction	Inseparable Function Interaction	Complexity of Objectives	1. Polynomial function (O(2)) 2. Rational function (O(0)/O(4))	1. Linear function 2. Linear function 3. Linear function 4. Linear function	1. Polynomial function (O(3)) 2. Rational complex function (O(5)/O(8))
		Nature of Constraints	4 linear & 1 rational	6 linear	8 linear & 1 rational
		Reported Pareto Front	Unknown	Unknown	Unknown
		Reported Complexity of Search Space	<ul style="list-style-type: none"> Non-linear, convex & continuous Pareto front Biased search space 	<ul style="list-style-type: none"> Linear, multi-dimensional & continuous Pareto front Unbiased search space 	<ul style="list-style-type: none"> Non-linear, convex & discontinuous Pareto front Biased search space Multi-front (multiple local Pareto fronts)
Variable Dependence		Independent decision variables	Independent decision variables	One dependency relationship among 2 decision variables (O(1))	
Published Results from Optimisation Algorithms	Single Objective Optimisers	<ul style="list-style-type: none"> Order of performance: Lexicographic > GALC > Min-max(OS) \cong GCM (OS) \cong WMM (OS) \cong PMM (OS) \cong NMM (OS) > Monte Carlo1 \cong Monte Carlo2 Most algorithms give solutions on estimated Pareto front 	<ul style="list-style-type: none"> Order of performance: Lexicographic > GALC > Monte Carlo1 \cong Monte Carlo2 > Min-max(OS) \cong GCM (OS) \cong WMM (OS) \cong PMM (OS) \cong NMM (OS) Most algorithms give solutions on estimated Pareto front 	<ul style="list-style-type: none"> Order of performance: Lexicographic > Monte Carlo1 > GALC > Monte Carlo2 Most algorithms give solutions on estimated Pareto front 	
	Multi-objective Optimisers	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > Gaminmax 1 > NPGA \cong MOGA \cong NSGA \cong VEGA \cong Gaminmax2 \cong Hajela Order of performance for distribution: NSGA-II > NPGA > MOGA > NSGA \cong Gaminmax2 \cong VEGA > Gaminmax 1 \cong Hajela 	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > Gaminmax 1 > NPGA \cong MOGA \cong NSGA \cong VEGA \cong Gaminmax2 \cong Hajela Order of performance for distribution: NSGA-II > NPGA > MOGA > NSGA \cong Gaminmax2 \cong Gaminmax1 > Hajela \cong VEGA 	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > Gaminmax 2 \cong NSGA \cong VEGA > Hajela > Gaminmax1 > NPGA \cong MOGA Order of performance for distribution: NSGA-II > VEGA > NSGA > Gaminmax2 > NPGA > MOGA > Gaminmax 1 \cong Hajela 	

Table 9.1: Case Studies from Real-life Engineering Design Optimisation (contd.)

Real-life Case Studies		Problem - 7	Problem - 8	Problem - 9	
		Two-bar Truss	Robot Arm	Single Screw Extruder	
General Problem Features	No. of Variables	3	4	4	
	Nature of Variables	3 real	4 real	4 real	
	No. of Objectives	2	4	5	
	No. of Constraints (+ Var. Bounds)	4	6	8	
Variable Interaction	Inseparable Function Interaction	Complexity of Objectives	1. Complex function (with root function) 2. Maximum function	1. Polynomial function (O(2)) 2. Polynomial function (O(2)) 3. Complex function (with root function) 4. Complex function (with root function)	1. Implicit multi-layered relationship 2. Implicit multi-layered relationship 3. Implicit multi-layered relationship 4. Implicit multi-layered relationship
		Nature of Constraints	3 linear & 1 maximum function	6 linear	4 linear & 4 compound multi-layered relationships
		Reported Pareto Front	Unknown	Unknown	Corresponds to fixed value of 1 variable
		Reported Complexity of Search Space	<ul style="list-style-type: none"> Non-linear, convex & continuous Pareto front Biased search space 	<ul style="list-style-type: none"> Non-linear & multi-dimensional Pareto front with unknown continuity Biased search space Unknown front modality 	<ul style="list-style-type: none"> Non-linear & multi-dimensional Pareto front with unknown continuity Biased search space Unknown front modality
	Variable Dependence	Independent decision variables	Independent decision variables	Independent decision variables	
Published Results from Optimisation Algorithms	Single Objective Optimisers	<ul style="list-style-type: none"> ϵ-constraint method Solutions lie on estimated Pareto front 	<ul style="list-style-type: none"> Order of performance: Lexicographic \cong GALC \cong Monte-Carlo1 \cong Monte-Carlo2 > Min-max(OS) \cong GCM (OS) \cong WMM (OS) \cong PMM (OS) \cong NMM (OS) Most algorithms give solutions on estimated Pareto front 	<ul style="list-style-type: none"> No published results were observed in this category 	
	Multi-objective Optimisers	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > NSGA Order of performance for distribution: NSGA-II > NSGA 	<ul style="list-style-type: none"> Order of performance for convergence: NSGA-II > Gaminmax2 \cong NSGA \cong VEGA ϵ Hajela ϵ Gaminmax1 > NPGA \cong MOGA Order of performance for distribution: NSGA-II > Gaminmax2 \cong VEGA ϵ Hajela ϵ Gaminmax1 > NSGA \cong NPGA \cong MOGA 	<ul style="list-style-type: none"> Order of performance for convergence: RPSGA > NPGA Order of performance for distribution: RPSGA > NPGA 	

Table 9.1: Case Studies from Real-life Engineering Design Optimisation (contd.)

Real-life Case Studies		Problem – 10(a)	Problem – 10(b)	
		Turbine Blade Cooling System	Turbine Blade Cooling System	
General Problem Features	No. of Variables	12	12	
	Nature of Variables	1 discrete & 11 real	1 discrete & 11 real	
	No. of Objectives	2	4	
	No. of Constraints (+ Var. Bounds)	15	15	
Variable Interaction	Inseparable Function Interaction	Complexity of Objectives	1. Implicit multi-layered relationship 2. Implicit multi-layered relationship	1. Implicit multi-layered relationship 2. Implicit multi-layered relationship 3. Implicit multi-layered relationship 4. Implicit multi-layered relationship
		Nature of Constraints	12 linear & 3 compound multi-layered relationships	12 linear & 3 compound multi-layered relationships
		Reported Pareto Front	Unknown	Unknown
		Reported Complexity of Search Space	<ul style="list-style-type: none"> Non-linear & multi-dimensional Pareto front with unknown continuity Biased search space Unknown front modality 	<ul style="list-style-type: none"> Non-linear & multi-dimensional Pareto front with unknown continuity Biased search space Unknown front modality
	Variable Dependence	Independent decision variables	Independent decision variables	
Published Results from Optimisation Algorithms	Single Objective Optimisers	<ul style="list-style-type: none"> ARTS Solutions lie on estimated Pareto front 	<ul style="list-style-type: none"> ARTS Solutions lie on estimated Pareto front 	
	Multi-objective Optimisers	<ul style="list-style-type: none"> No published results were observed in this category 	<ul style="list-style-type: none"> No published results were observed in this category 	

9.3 Design of a Welded Beam

This design is shown in Figure 9.1, and is briefly described in Section 9.1. The model for this design is given in Equation 9.1. This equation assumes the following values.

- ◆ Overhang portion of the beam = 14 inch.
- ◆ $F = 6000$ lb force.
- ◆ Allowable shear strength of the material = 13600 psi.
- ◆ Allowable yield strength of the material = 30,000 psi.

$$\begin{aligned}
\text{Minimise} &\Rightarrow \text{Cost} = f_1(x) = 1.10471h^2l + 0.04811tb(14.0 + l), & \text{Equation 9.1} \\
\text{Minimise} &\Rightarrow \text{End_Deflection} = f_2(x) = \delta(x), \\
\text{Constraints} &\Rightarrow g_1(x) = 13,600 - \tau(x) \geq 0, \\
g_2(x) &= 30,000 - \sigma(x) \geq 0, \\
g_3(x) &= b - h \geq 0, \\
g_4(x) &= P_c(x) - 6,000 \geq 0. \\
\text{Deflection_Term} &= \delta(x) = 2.1952/t^3b, \\
\tau(x) &= \sqrt{\tau'^2 + \tau''^2 + (l\tau'\tau'')}/\sqrt{0.25(l^2 + (h+t)^2)}, \\
\tau' &= 6,000/\sqrt{2hl}, \\
\tau'' &= \frac{6,000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2\{0.707hl(l^2/12 + 0.25(h+t)^2)\}}, \\
\sigma(x) &= 504,000/t^2b, \\
P_c(x) &= 64,746.022(1 - 0.0282346t)tb^3.
\end{aligned}$$

9.3.1 Experimental Results

Figure 9.5, Figure 9.6 and Figure 9.7 respectively depict the results obtained by applying NSGA-II, GRGA (without final redistribution) and GRGA (with final redistribution) to the optimisation of a welded beam design. The tests reported here are carried out using 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index. These results form the typical set obtained from 10 runs with different seed values for the random number generator. No major variation was observed in the results with the change in seed values. Furthermore, HDA is used here with GRGA, and to enable fair comparison, the termination condition is not applied here for reporting the GRGA results.

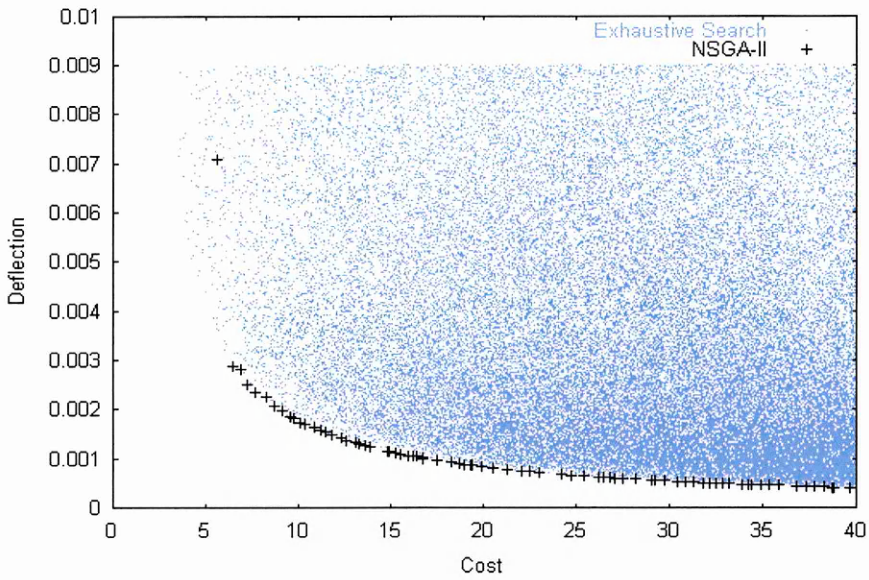


Figure 9.5: Results from NSGA-II on Welded Beam Design (Units: Deflection in inch, Cost in cost units)

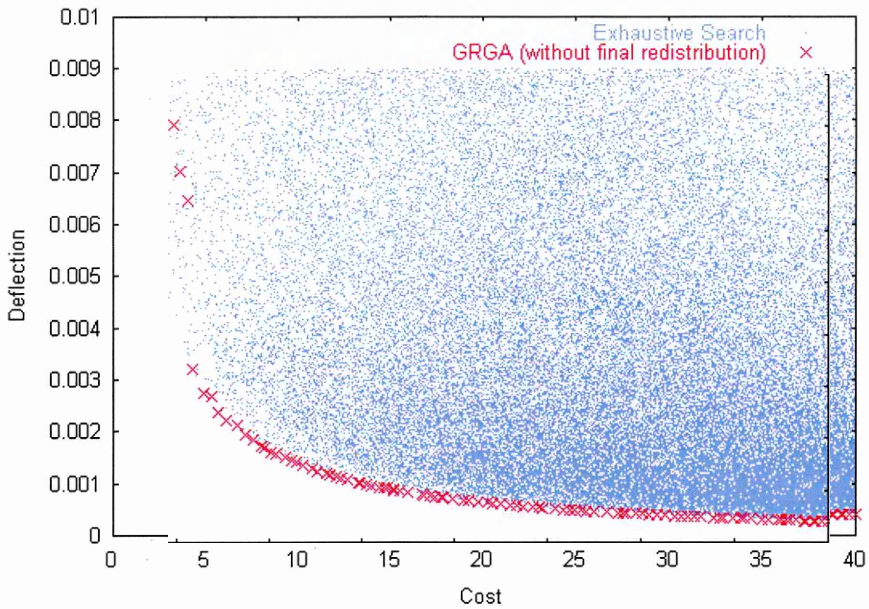


Figure 9.6: Results from GRGA (Without Final Redistribution) on Welded Beam Design (Units: Deflection in inch, Cost in cost units)

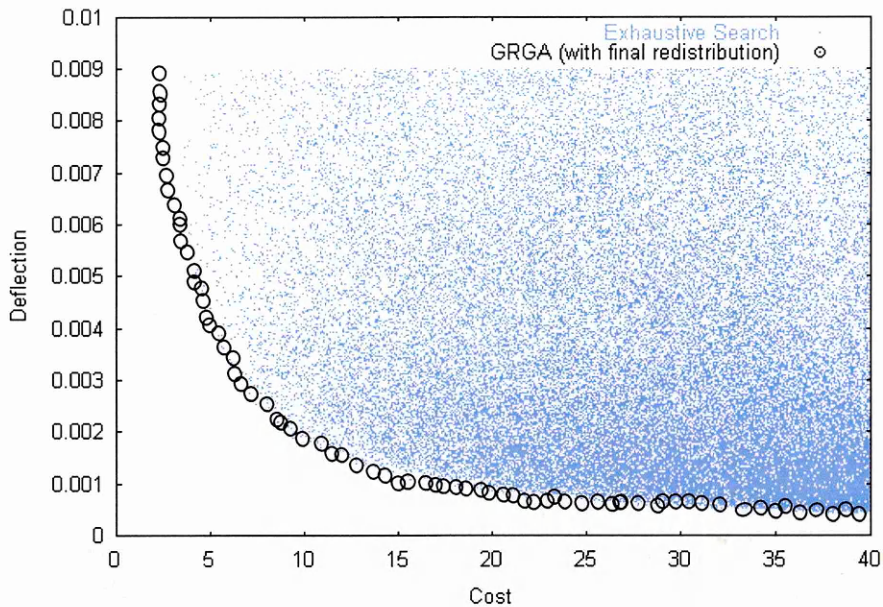


Figure 9.7: Results from GRGA (With Final Redistribution) on Welded Beam Design
(Units: Deflection in inch, Cost in cost units)

9.3.2 Discussion of Results

The salient observations from the above-mentioned results are as follows.

- ◆ The problem has a search space that is biased towards high values of cost and low values of deflection. This implies that most of the solutions of the given model lie in this region. Furthermore, the Pareto front of this problem is convex and continuous in nature.
- ◆ Both NSGA-II and GRGA converge to the Pareto front in this case. In the case of NSGA-II, this has been possible due to the absence of multiple local fronts. Since GRGA converges to the Pareto front, it has been able to determine the relationships involving decision variables that define the Pareto front. Hence, when GRGA uses these values to redistribute the final solutions, a well-defined Pareto front is attained.
- ◆ Since this problem does not have variable dependency, GAVD is not applied in this case.
- ◆ GRGA reveals that the Pareto front of this problem corresponds to $h = 0.422$, $l = 2.465$ and $t = 9.990$. Table 9.2 shows that a majority of final solutions determined by GRGA have these same values for h , l and t . Therefore, to attain any solution

on the Pareto front, the designer needs to fix h , l and t to these values, and choose a value for b based on his/her preferences.

Table 9.2: Variable Values Corresponding to Identified Pareto Front

Pareto-optimal Solutions	Variable Values	% of Final Solutions
h	0.422	83
l	2.465	87
t	9.990	97
b	NA	NA

9.4 Design of a Machine Tool Spindle

This design is shown in Figure 9.2, and is briefly described in Section 9.1. The model for this design is given in Equation 9.2. This equation assumes the following values.

- ◆ $d_{om} = 25.00$ mm, $d_{a1} = 80.00$ mm, $d_{a2} = 95.00$ mm, $d_{b1} = 75.00$ mm, $d_{b2} = 90.00$ mm, $p_1 = 1.25$, $p_2 = 1.05$, $l_k = 150.00$ mm, $l_g = 200.00$ mm, $a = 80.00$ mm, $E = 210,000.0$ N/mm², $F = 10,000$ N, $\Delta_a = 0.00540000$ mm, $\Delta_b = -0.00540000$ mm, $\Delta = 0.01000000$ mm, $\delta_{ra} = -0.00100000$ mm and $\delta_{rb} = -0.00100000$ mm.
- ◆ d_a must be chosen from the set $\{80,85,90,95\}$ and d_b from the set $\{75,80,85,90\}$.

9.4.1 Experimental Results

Figure 9.8, Figure 9.9 and Figure 9.10 respectively depict the results obtained by applying NSGA-II, GRGA (with final redistribution) and GAVD (with final redistribution) to the design optimisation of a machine tool spindle. The tests reported here are carried out using 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index. These results form the typical set obtained from 10 runs with different seed values for the random number generator. No major variation was observed in the results with the change in seed values. Furthermore, HDA is used here with GRGA, and to enable fair comparison, the termination condition is not applied here for reporting the GRGA results.

Minimise(Volume_of_Spindle) \Rightarrow

Equation 9.2

$$f_1(x) = (\pi/4)[a(d_a^2 - d_o^2) + l(d_b^2 - d_o^2)],$$

Minimise(Static_Displacement_under_F) \Rightarrow

$$f_2(x) = \frac{Fa^3}{3EI_a} \left(1 + \frac{lI_a}{aI_b}\right) + \frac{F}{c_a} \left[\left(1 + \frac{a}{l}\right)^2 + \frac{c_a a^2}{c_b l^2}\right],$$

$$\text{Constraints} \Rightarrow g_1(x) = l - l_g \leq 0,$$

$$g_2(x) = l_k - l \leq 0,$$

$$g_3(x) = d_{a1} - d_a \leq 0,$$

$$g_4(x) = d_a - d_{a2} \leq 0,$$

$$g_5(x) = d_{b1} - d_b \leq 0,$$

$$g_6(x) = d_b - d_{b2} \leq 0,$$

$$g_7(x) = d_{om} - d_o \leq 0,$$

$$g_8(x) = p_1 d_o - d_b \leq 0,$$

$$g_9(x) = \left| \Delta_a + (\Delta_a - \Delta_b) \frac{a}{l} \right| - \Delta \leq 0,$$

Dependency_Equation (Designer's_Proportion_Preference)

$$\Rightarrow p_2 d_o = d_b,$$

Δ = Maximum_Runout_of_Spindle_Nose,

Δ_a = Radial_Runout_of_Front_Bearing,

Δ_b = Radial_Runout_of_Back_Bearing,

Moment_of_Inertia = $I_a = 0.049(d_a^4 - d_o^4)$,

Moment_of_Inertia = $I_b = 0.049(d_b^4 - d_o^4)$,

Bearing_Stiffness = $c_a = 35400 |\delta_{ra}|^{1/9} d_a^{10/9}$, d_{ra} = Preload,

Bearing_Stiffness = $c_b = 35400 |\delta_{rb}|^{1/9} d_b^{10/9}$, d_{rb} = Preload.

9.4.2 Discussion of Results

The salient observations from the above-mentioned results are as follows.

- ◆ Due to two discrete variables in this problem, the search space is discontinuous. This also leads to multiple local fronts in the problem. The discontinuity in the search space also makes the global Pareto front discontinuous, composing it as a combination of parts of several local fronts. Furthermore, there is bias in the search space towards higher values of both displacement and volume. This means

that the model has a tendency to generate solutions that lie in this part of the search space.

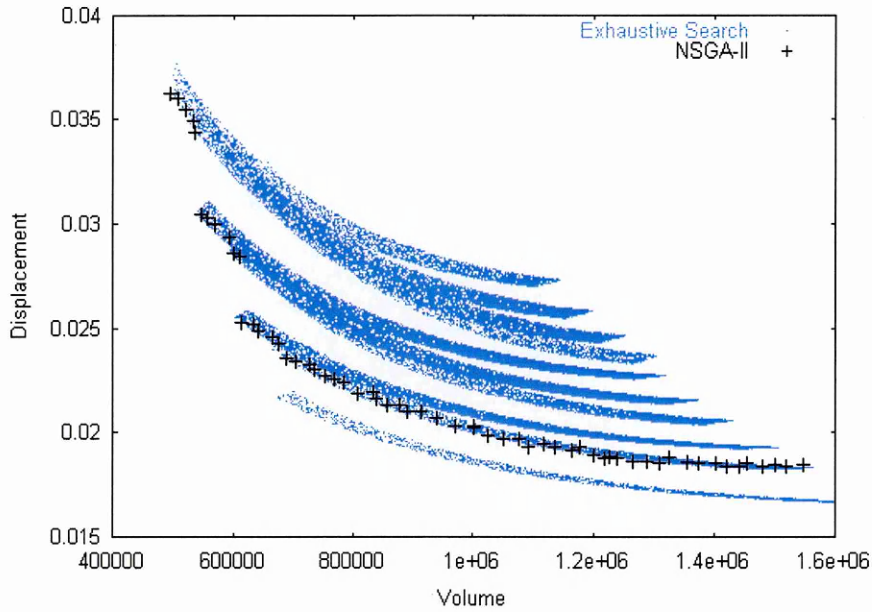


Figure 9.8: Results from NSGA-II on Design of Machine Tool Spindle (Units: Displacement in mm, Volume in mm^3)

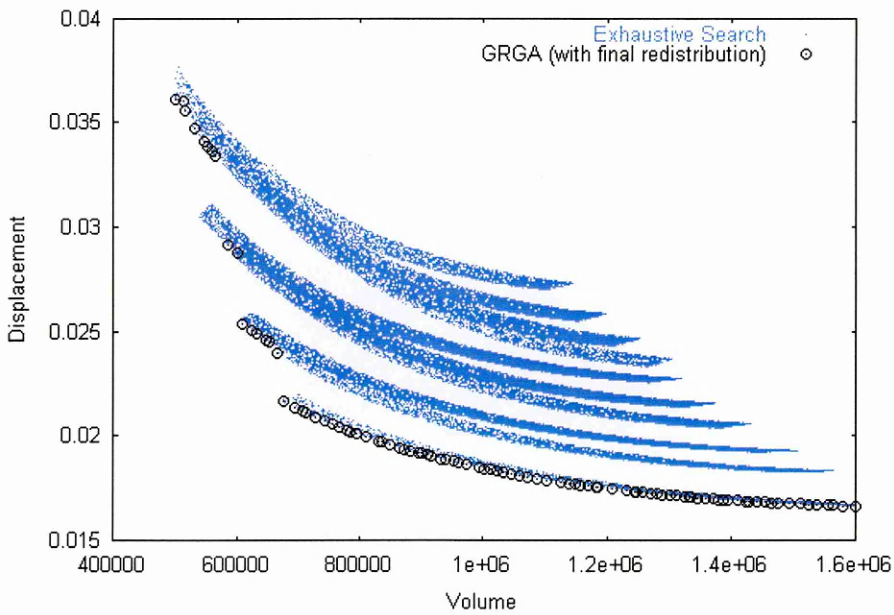


Figure 9.9: Results from GRGA (With Final Redistribution) on Design of Machine Tool Spindle (Units: Displacement in mm, Volume in mm^3)

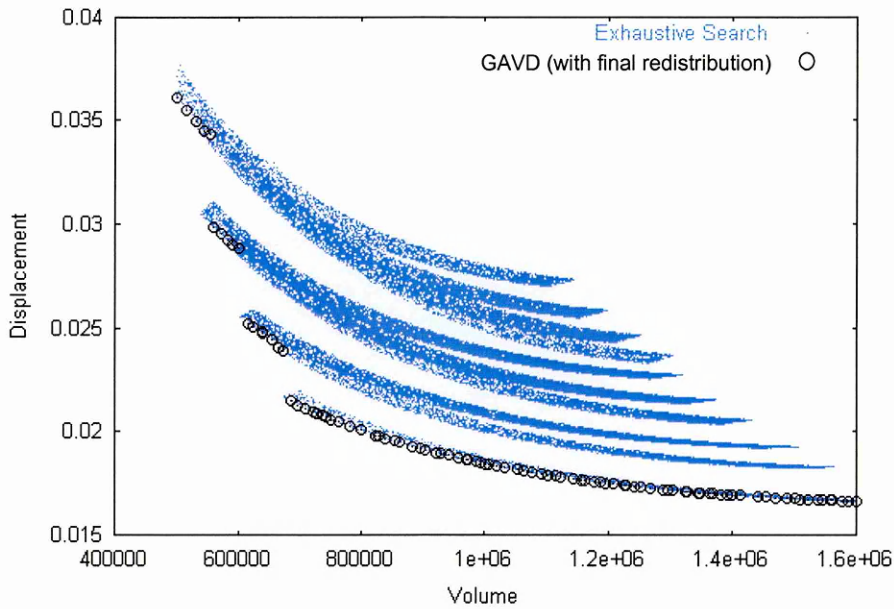


Figure 9.10: Results from GAVD (With Final Redistribution) on Design of Machine Tool Spindle (Units: Displacement in mm, Volume in mm^3)

- ◆ Here, GRGA successfully locates the global Pareto front, including all its parts. However, NSGA-II gets trapped in a local front due to the same reasons as mentioned for this behaviour in the previous chapter. Moreover, NSGA-II has also been able to locate all other individual parts of the Pareto front.
- ◆ Since GAVD has an in-built mechanism for dealing with variable dependence and since it uses GRGA as its optimisation engine, it has been able to locate all the parts of the Pareto front and distribute the solutions evenly across all these parts. In this case, the performances of GAVD and GRGA are similar since the introduction of variable dependence does not change the Pareto front. Therefore, GAVD gives better results than GRGA only in those cases in which variable dependency changes the Pareto front.
- ◆ GAVD reveals that the Pareto front of this problem corresponds to $l = 187.78$, $d_a = 95$ and $d_b = 90$. Table 9.3 shows that a majority of final solutions determined by GAVD have these same values for l , d_a and d_b . Therefore, to attain any solution on the Pareto front, the designer needs to fix l , d_a and d_b to these values, and choose a value for d_o based on his/her preferences.

Table 9.3: Variable Values Corresponding to Identified Pareto Front

Pareto-optimal Solutions	Variable Values	% of Final Solutions
l	187.78	99
d _o	NA	NA
d _a	95	92
d _b	90	96

9.5 Design of a Turbine Blade Cooling System

The preliminary design of the turbine blade cooling system was briefly described in 9.1, using Figure 9.3 and Figure 9.4. The model is developed considering one dimensional, single pass coolant flow. The model includes a cooling film mechanism, and involves twelve design variables. This Turbine Blade Cooling system Model (TBCOM) also uses several constants known as design parameters. TBCOM also includes three non-linear constraints.

The common nomenclatures used in this model are: A for cross-sectional area of passage, C_d for coefficient of discharge, C_p for specific heat at constant pressure, C_v for specific heat at constant volume, d for hydraulic diameter, d_{th} for wall thickness, h for heat transfer coefficient, H1 for parameter group for heat balance equation, H2 for parameter group for heat balance equation, H3 parameter group for heat balance equation, k for thermal conductivity, l for passage length, M for Mach number, N for number, P_c for cooling air pressure, R for gas constant, S_c for cooling side parameter, S_g for gas side effective perimeter, T_c for cooling air temperature, W for mass flow, X_f for distance from film cooling hole exit / effective slot width of film, γ for ratio of specific heats and μ for dynamic viscosity. The following subscripts are also used here: 1 for cooling air inlet, 2 for film cooling passage inlet, 3 for cooling air exit, 3' for film cooling hole exit, b for blade, c for coolant, f for film and g for gas, hpc for high pressure compressor, r for radial passage and w for wall.

Like many other design models, TBCOM involves several constants known as design parameters. The design parameter values with their respective nomenclature are as follows.

- ◆ Heat transfer coefficient (gas side), $h_g = 3000.0 \text{ W/m}^2\text{K}$; gas side temperature, $T_g = 1500.0 \text{ K}$; ratio of specific heats, $\gamma = 1.36$; mass flow (high pressure compressor), $W_{hpc} = 84.85 \text{ Kg/s}$; radial cooling hole exit pressure, $P_{c3} = 460000.0 \text{ N/m}^2$; number of blades, $N_b = 78$; wall temperature (gas side) for initial calculations, $T_{wg} = 1250.0 \text{ K}$; radial passage length, $l_r = 0.0406 \text{ m}$; specific heat at constant pressure, $C_p = 993.0$; one of two factors for heat transfer coefficient, $F = 0.01855$; gas constant, $R = 287.0$; distance from film cooling hole exit/effective slot width of film, $X_f = 10$; Mach number, $\text{Mach} = 0.6$; number of film holes, $N_f = 30$ and initial outside temperature, $T_{wgl} = 1500.0 \text{ K}$.
- ◆ Maximum radial passage area, $A_{cr} \leq 2.75\text{E-}05 \text{ m}^2$; bounds on radial coolant flow heat transfer coefficient, $100.0 \text{ W/m}^2\text{K} < h_{cr} < 4000.0 \text{ W/m}^2\text{K}$; check on metal temperature, $1000.0 \text{ K} < T_{wg} < 1500.0 \text{ K}$; for the film cooling section, heat transfer coefficients are the same for the film side and the gas side, that is $h_f = h_g$ and for the film cooling section, the perimeter ratio, $R_{sf} = 1.0$.

As stated earlier, the objective functions are implicit and multi-layered in this case. This implies that the determination of objective values requires an iterative procedure. This procedure together with the equations involved is detailed in Roy (1997). Here, the iterative design procedure used for the calculation of the values of W_{cr} and T_{wg} is shown in Table 9.4 to give an illustration of the model complexity and its equations.

$$W_{cr} = 0.003 \times W_{hpc} / N_b. \quad \text{Equation 9.3}$$

$$h_{cr} = h_g (S_{gr} / S_{cr}) \frac{(T_g - T_{wg})}{(T_{wc} - T_c)}. \quad \text{Equation 9.4}$$

$$A_{cr} = (FF \times (k / \mu^{0.8}) (W_{cr}^{0.8} / h_{cr})), \text{ where,} \quad \text{Equation 9.5}$$

$$FF = F \times Fhc,$$

$$k = \frac{2.978\text{E} - 03 \times T_c^{0.5}}{1 + (240.0 / T_c)},$$

$$\mu = \frac{1.488\text{E} - 06 \times T_c^{1.5}}{T_c + 110.4}.$$

(for initial value assume $T_c = T_{c1}$)

Table 9.4: Cooling System Design Procedure Used in TBCOM (Source: Roy, 1997)

Step Number	Task	Equation	Comment
Step 1	Estimate W_{cr}	Equation 9.3	Based on the limiting value of flow off-take from the engine compressor.
Step 2	Estimate T_{wg}	-	Based on material property limitation, suggested 1500.0 K.
Step 3	Calculate h_{cr}	Equation 9.4	-
Step 4	Calculate A_{cr}	Equation 9.5	Check the value, if within the limiting value of A_{cr} , go to Step 5. If not within the limiting value of A_{cr} , then $W_{cr} = W_{cr} * 0.99$ and go back to Step 4.
Step 5	Calculate W_{cr}	Equation 9.6	-
Step 6	Calculate h_{cr}	Equation 9.7	Compare h_{cr} value from Step 6 with Step 3, if within tolerance then proceed to check whether h_{cr} lies within the acceptable range, if yes then proceed to Step 7 otherwise reset the T_{wg} and h_{cr} values and go to Step 4. If the wall temperature calculation reaches a steady state then only accept, if not equal then go back to Step 4.
Step 7	Calculate T_{wg}	Equation 9.8	Check the value, if within the acceptable limit then accept. If not within the limit and if W_{cr} has not been changed previously, change W_{cr} as $W_{cr} = W_{cr} * 1.01$.
Step 8	Calculate T_c	Equation 9.9	-
Step 9	Recalculate k	k defined in Equation 9.5	-
Step 10	Recalculate μ	μ defined in Equation 9.5	Reset T_{wg} and h_{cr} values and go to Step 4. If the wall temperature calculation reaches a steady state then only accept.

$$W_{cr} = \frac{A_{cr} C_d P c_1}{\sqrt{T c_1}} \left(\frac{2\gamma}{R(\gamma-1)} \left(\left(\frac{P c_1}{P c_2} \right)^{-2/\gamma} - \left(\frac{P c_1}{P c_2} \right)^{-(1+\gamma)/\gamma} \right) \right)^{0.5} \quad \text{Equation 9.6}$$

$$h_{cr} = FF \times (k / \mu^{0.8}) \times (W_{cr}^{0.8} / A_{cr}^{0.9}). \quad \text{Equation 9.7}$$

$$T_{wg} = \frac{\left(1 + H2 - \frac{H1 \times H2}{H1 + H3}\right) T_g + \left(H1 - \frac{H1^2}{H1 + H3}\right) T_{c1}}{1 + H2 - \frac{H1 \times H2}{H1 + H3} + \frac{H1 \times H3}{H1 + H3}}, \text{ where,} \quad \text{Equation 9.8}$$

$$H1 = \frac{h_{cr}}{h_g \times (S_{gr} / S_{cr})},$$

$$H2 = (h_{cr} S_{cr} l_r) / (2W_{cr} C_p),$$

$$S_{cr} = 3.545 \times \sqrt{A_{cr}},$$

$$l_r = 0.0406m,$$

$$H3 = \frac{0.5k_w}{dth \times h_g} \left(1 + \frac{1}{(S_{gr} / S_{cr})}\right).$$

$$Tc = (H2 / H1)(T_g - T_{wg}) + T_{c1}, \text{ where,} \quad \text{Equation 9.9}$$

$$Tc_3 - Tc_1 = 2(Tc - Tc_1)$$

9.5.1 Experimental Results

Two sets of results are reported in this case. In the first set, only two objectives are considered for optimisation (W_{cr} and T_{wg}) whereas in the second set, all the four objectives are included (W_{cr} , W_{cf} , T_{wg} and T_{wf}). In both these cases, all the constraints in the problem are incorporated in the solution procedure. Figure 9.11, Figure 9.12 and Figure 9.13 respectively depict the results obtained by applying NSGA-II, GRGA (without final redistribution) and GRGA (with final redistribution) to the two-objective design optimisation of a turbine blade cooling system. Also, Table 9.5 depicts the results obtained by applying NSGA-II, GRGA (without final redistribution) and GRGA (with final redistribution) to the four-objective version of the same problem. All the tests reported here are carried out using 100 population size, 500 generations, 0.8 crossover probability, 0.05 mutation probability, and simulated binary crossover with 10 crossover distribution index and 50 mutation distribution index. These results form the typical set obtained from 10 runs with different seed values for the random number generator. No major variation was observed in the results with the change in seed values. Furthermore, HDA is used

here with GRGA, and to enable fair comparison, the termination condition is not applied here for reporting the GRGA results.

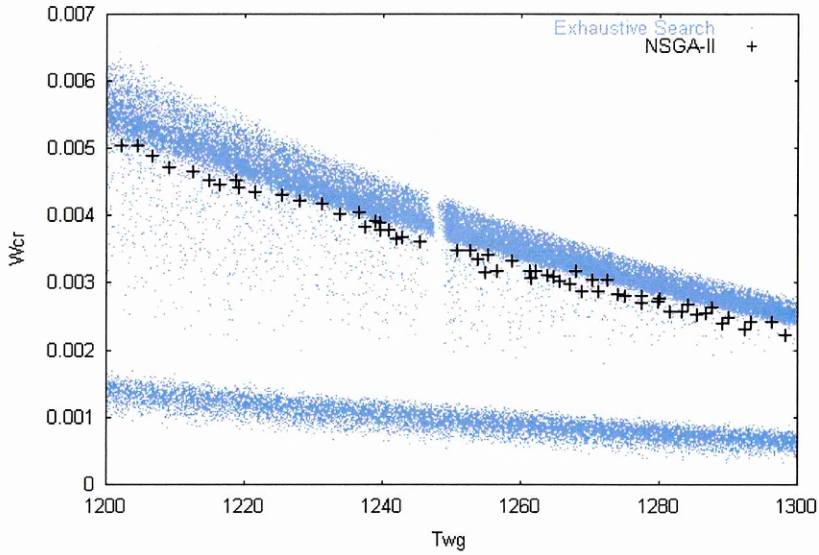


Figure 9.11: Results from NSGA-II on Design of Turbine Blade Cooling System (Assuming Two Objectives) (Units: W_{cr} in Kg/s, T_{wg} in K)

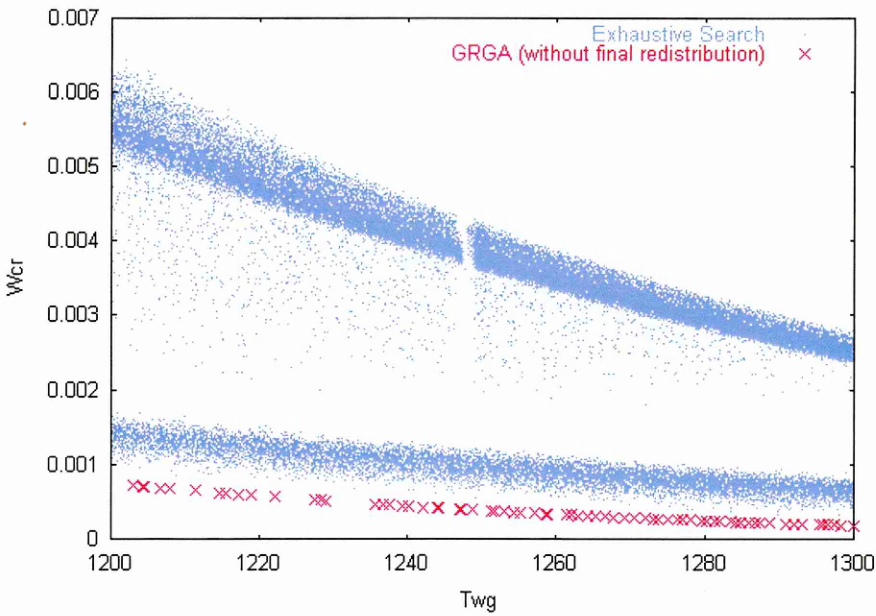


Figure 9.12: Results from GRGA (without final redistribution) on Design of Turbine Blade Cooling System (Assuming Two Objectives) (Units: W_{cr} in Kg/s, T_{wg} in K)

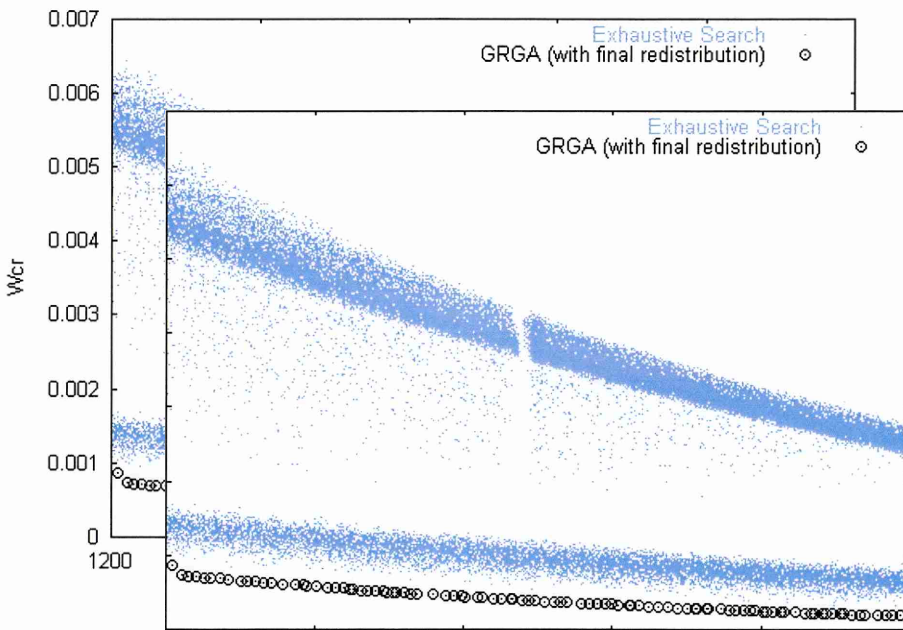
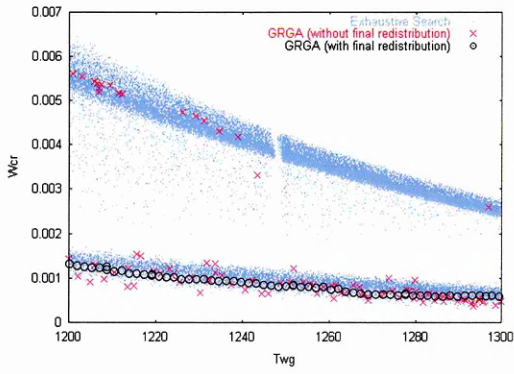


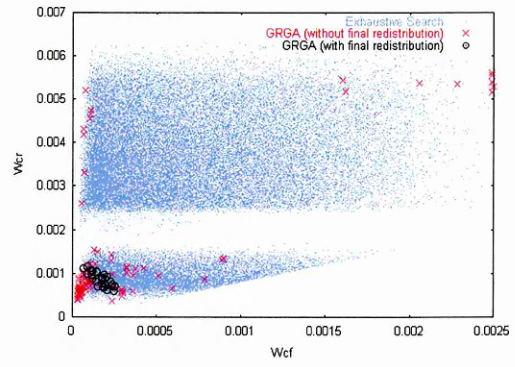
Figure 9.13: Results from GRGA (with final redistribution) on Design of Turbine Blade Cooling System (Assuming Two Objectives) (Units: W_{cr} in Kg/s, T_{wg} in K)

Table 9.5: Results from NSGA-II, GRGA (without final redistribution) and GRGA (with final redistribution) on Design of Turbine Blade Cooling System (Assuming Four Objectives) (Upper Diagonal Graphs: GRGA Results, Lower Diagonal Graphs: NSGA-II Results) (Units: W_{cr} in Kg/s, W_{cf} in Kg/s, T_{wg} in K, T_{wf} in K)

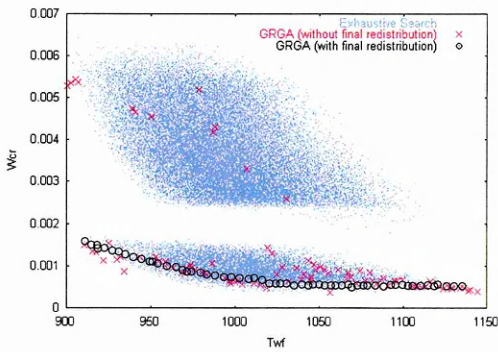
Wcr (0-0.007)	Figure 9.14(a)	Figure 9.14(b)	Figure 9.14(c)
Figure 9.15(a)	Twg (1200-1300)	Figure 9.14(d)	Figure 9.14(e)
Figure 9.15(b)	Figure 9.15(c)	Wcf (0-0.0025)	Figure 9.14(f)
Figure 9.15(d)	Figure 9.15(e)	Figure 9.15(f)	Twf (900-1150)



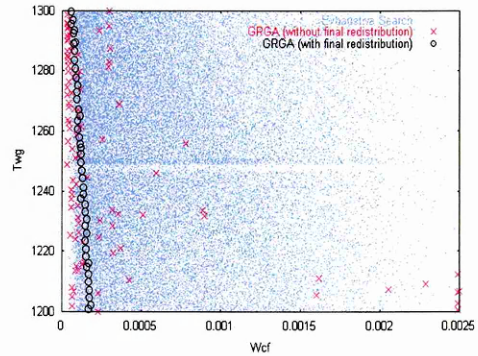
(a)



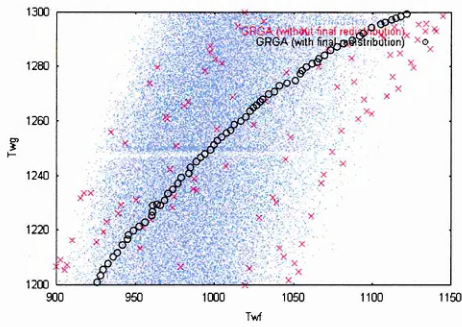
(b)



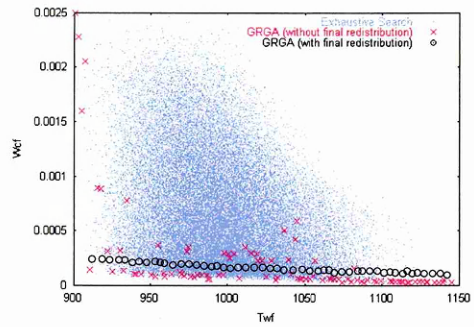
(c)



(d)

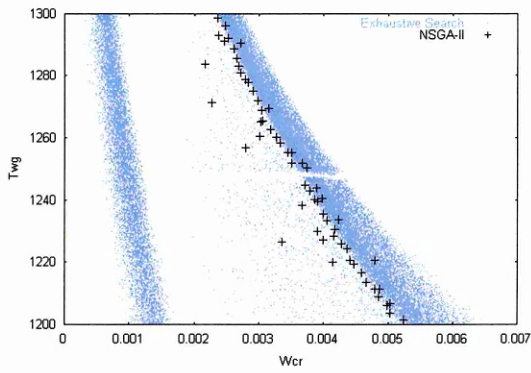


(e)

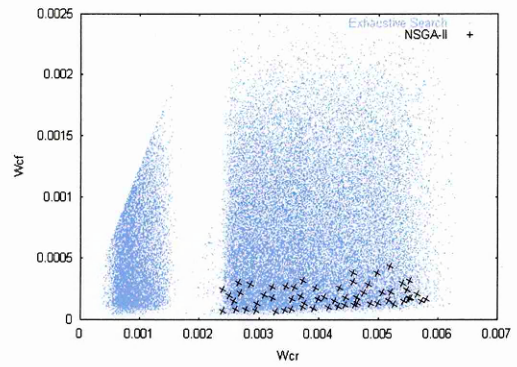


(f)

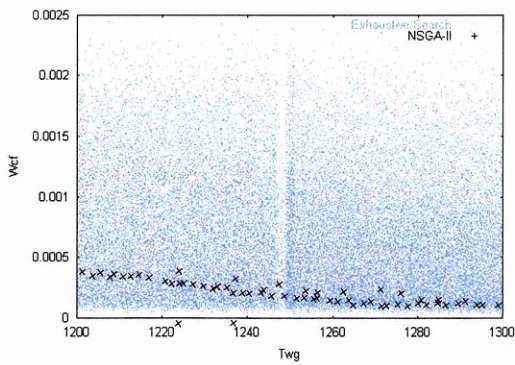
Figure 9.14: Results from GRGA on Design of Turbine Blade Cooling System (Assuming Four Objectives) (Units: Wcr in Kg/s, Wcf in Kg/s, Twg in K, Twf in K) – (a) Wcr - Twg Graph (b) Wcr - Wcf Graph (c) Wcr - Twf Graph (d) Twg - Wcf Graph (e) Twg - Twf Graph (f) Wcf - Twf Graph



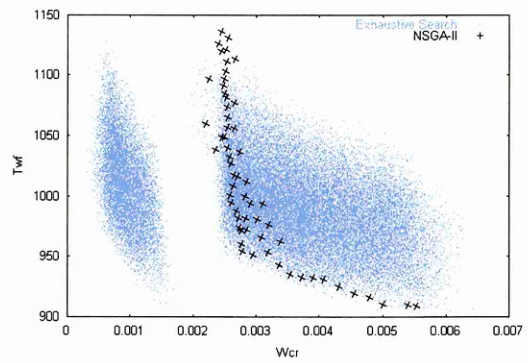
(a)



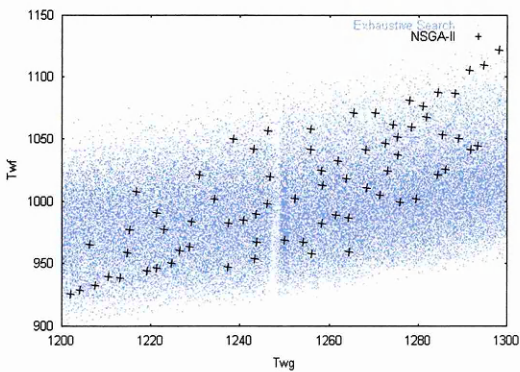
(b)



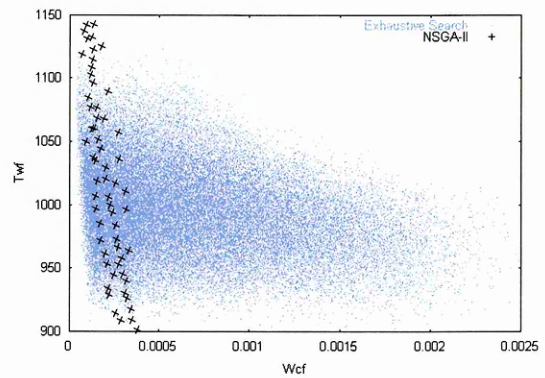
(c)



(d)



(e)



(f)

Figure 9.15: Results from NSGA-II on Design of Turbine Blade Cooling System (Assuming Four Objectives) (Units: W_{cr} in Kg/s, W_{cf} in Kg/s, T_{wg} in K, T_{wf} in K) – (a) T_{wg} - W_{cr} Graph (b) W_{cf} - W_{cr} Graph (c) W_{cf} - T_{wg} Graph (d) T_{wf} - W_{cr} Graph (e) T_{wf} - T_{wg} Graph (f) T_{wf} - W_{cf} Graph

9.5.2 Discussion of Results

The following observations can be made regarding the search space of this problem.

- ◆ It can be seen from the above figures that the Pareto fronts appear in all W-T plots, implying conflict between W's and T's. This is intuitive since any increase in coolant mass flow is expected to decrease the metal temperature and vice versa. Furthermore, as expected, the conflicting fronts do not appear in any W-W or T-T plot.
- ◆ It can also be seen from these plots that there is a bias in the search space. An example of this is the W_{cr} - T_{wg} plot (Figure 9.14(a)) that exhibits bias towards higher values of T_{wg} .
- ◆ The W_{cr} - T_{wg} plot also depicts that the given model has a local and a global Pareto front with respect to W_{cr} and T_{wg} . This multi-modality arises due to the presence of a discrete variable in the problem. This also causes very low density of population in the region between the two fronts, leading to deception in the search space.
- ◆ This model exhibits a discontinuity at the value of T_{wg} equal to 1250. At this value of T_{wg} , which is also its initial value, some of the output values are undefined, leading to discontinuity in the search space. This might be due to an error in the model.
- ◆ This model has 12 variable bounds and 3 constraints. The introduction of these 3 constraints has the following effects on the model.
- ◆ The first constraint ($1200.0 < T_{wg} < 1300.0$) defines the range of T_{wg} . This constraint, together with the relationship between T_{wg} and W_{cr} , also limits the values that can be taken by W_{cr} .
- ◆ Since, in the unconstrained model, the value of T_{wf} does not cross 1300.0, the second constraint ($T_{wf} < 1300.0$) does not have any impact on the model.
- ◆ All the solutions that lie below the line $W_{cr} = 0.8 \times W_{cf}$ in the unconstrained model become infeasible in the constrained model due to the third constraint: $W_{cr}/W_{cf} \geq 0.8$.
- ◆ As can be seen from the W_{cr} - T_{wg} plot (Figure 9.14(a)), the restrictions on T_{wg} values create a situation in which there are very few feasible points

corresponding to certain values of W_{cr} (close to a value of 0.002). This results in similar regions in all the plots that involve W_{cr} .

- ◆ In summary, the constraints focus the plot in the given range of W_{cr} and T_{wg} , introduce an infeasible region in the W_{cr} - W_{cf} plot (Figure 9.14(b)), and create sparsely populated regions that correspond to values of W_{cr} close to 0.002.
- ◆ The four-objective search space reveals well-defined Pareto front in all its two-dimensional plots. This is a clear demonstration of the fact that the Pareto front in this problem has the form of a curve, rather than a higher dimensional entity. This is because in a multi-dimensional search space only a curve can be projected as a curve in all its two-dimensional plots.
- ◆ The above observations are valid in the cases involving both two and four objectives. However, it should be noted that in the case of two objectives, the function space is only two-dimensional, having just one plot: W_{cr} - T_{wg} .

The following conclusions can be drawn from the tests performed on the four-objective version of this model.

- ◆ In this case, GRGA gives solutions that are very close to the Pareto front, but do not exactly converge to the front due to the multi-dimensional nature of the search space. However, GRGA determines the relationships involving those decision variables that define the Pareto front. Therefore, when GRGA uses these values to redistribute the final solutions, the results that are attained lie on the Pareto front and are well distributed across it.
- ◆ Here, NSGA-II gets trapped in a local front due to the same reasons as mentioned for this behaviour in the previous chapter. However, the solutions that are generated by NSGA-II exhibit a good distribution.
- ◆ Since this problem does not have variable dependency, GAVD is not applied in this case.
- ◆ GRGA reveals that the Pareto front of this problem corresponds to $Geom = pedestal$, $C_{dr} = 0.3901$, $dth = 0.0019903$, $k_w = 19.87$, $df = 0.000149$, $C_{df} = 0.6378$, $F_f = 1.101$ and $R_{pf} = 0.381$. Table 9.6 shows that a majority of final solutions determined by GRGA have these values of design variables. Therefore, to attain any solution on the Pareto front, the designer needs to fix the above-mentioned variables to these values, and choose values for other variables (F_{hc} , T_{c1} , R_p and R_s) based on his/her preferences.

Table 9.6: Variable Values Corresponding to Identified Pareto Front

Pareto-optimal Solutions	4 Objective Functions		2 Objective Functions	
	Variable Values	% of Final Solutions	Variable Values	% of Final Solutions
Geometry	3	83	3	99
Cdr	0.3901	80	0.3995	99
Fhc	NA	NA	3.199	95
Tc1	NA	NA	799.05	99
dth	0.0019903	74	0.0024381	99
kw	19.87	77	19.19	99
Rp	NA	NA	1.5933	98
Rs	NA	NA	NA	NA
df	0.000149	77	0.0001041	90
Cdf	0.6378	86	0.7491	89
Ff	1.101	78	1.497	92
Rpf	0.381	81	0.397	94

The results obtained from the two-objective case are analysed here to make the following observations.

- ◆ GRGA is able to successfully converge to the Pareto front even without the use of a local search. This is expected since the reduction in the dimensionality of the model makes it easier for the optimisation algorithms to locate the Pareto front. Due to this convergence, GRGA is also able to successfully redistribute the solutions across the Pareto front. Here as well, NSGA-II gets trapped on a local front.
- ◆ GAVD is not tested on this problem for the same reason as mentioned in the four-objective case.
- ◆ As compared to the four-objective case, the Pareto front of this problem has fixed values for a larger number of variables. This is not surprising since the reduction in the number of objectives reduces the degrees of freedom of the Pareto front, thereby increasing the number of variables that take fixed values on the Pareto front. Here, GRGA reveals that the Pareto front corresponds to $\text{Geom} = \text{pedestal}$, $C_{dr} = 0.3995$, $F_{hc} = 3.199$, $T_{c1} = 799.05$, $d_{th} = 0.0024381$, $k_w = 19.19$, $R_p = 1.5933$, $d_f = 0.0001041$, $C_{df} = 0.7491$, $F_f = 1.497$ and $R_{pf} = 0.397$. Table 9.6 shows that a majority of final solutions determined by GRGA have these values of design variables. Therefore, to attain any solution on the Pareto front, the designer needs to fix the above-mentioned variables to these values, and choose a

suitable value for R_s based on his/her preferences. It is interesting to note that the values reported here are similar to the ones observed in the four-objective case. Further, R_s is also a free variable in the previous case.

9.6 Validation of Results

The results obtained from GRGA and GAVD are validated here using exhaustive search and through comparison with the results published in literature.

9.6.1 Design of a Welded Beam

Visualisation of the GRGA results with the results of exhaustive search clearly depicts that GRGA has been able to converge to the Pareto front (Figure 9.7). It can also be seen from this plot that GRGA produces well distributed solutions across the Pareto front. These solutions also cover near the whole of Pareto front, with the extreme solutions having very high deflection and low cost on one end, and very high cost and low deflection on the other.

Literature reveals that in solving this problem NSGA-II gives better performance than all other algorithms (Deb, Pratap and Moitra, 2000). Figure 9.5 depicts the NSGA-II results on this problem. It is evident that although NSGA-II has been able to converge to the Pareto front, it does not exhibit uniform distribution of solutions. It can be seen from this graph that NSGA-II does not locate any Pareto-optimal solution in region of the search space that corresponds to low cost and high deflection. This is because the search space has a bias against these values of objective functions (Figure 9.5).

9.6.2 Design of a Machine Tool Spindle

The Pareto front in this problem is made up of four discontinuous parts that originate from the global and three local fronts in the search space. It can be seen from Figure 9.9 and Figure 9.10 that both GRGA and GAVD are able to converge to all the four parts of the Pareto front. The performances of GRGA and GAVD are similar in this case since the introduction of variable dependence does not change the Pareto front.

Coello (1997) compares the performances of a number of multi-objective optimisation algorithms in solving this problem. These include VEGA, NSGA, MOGA, NPGA, Hajela's Method and GAminmax. It is observed from these results that although these algorithms could locate points close to the Pareto front, they could not exactly converge to either the global front or any of the local fronts. Furthermore, all the results reported by Coello (1997) exhibit concentration towards a region of the search space, thereby leading to very poor distribution of solutions. The author observed that in this problem NSGA-II gives better results than the above-mentioned algorithms. Figure 9.8 shows that although NSGA-II fails to locate the global Pareto front, it converges to the front that is the closest to the Pareto front. NSGA-II is also able to locate the other two parts of the Pareto front. As compared to NSGA-II, both GRGA (Figure 9.9) and GAVD (Figure 9.10) exhibit better convergence since they are able to converge to the global Pareto front, and locate all the other three discontinuous parts originating from the local fronts. These algorithms also exhibit good distribution of solutions across all these discontinuous parts of the Pareto front.

9.6.3 Design of a Turbine Blade Cooling System

This problem also has a global and a local front. In the two-objective version of this problem, it can be seen from Figure 9.13 that GRGA is able to locate the global Pareto front. It is also evident from this figure that GRGA gives equal distribution of solutions across the Pareto front, and is able to locate solutions across the full span of the Pareto front. In the four-objective case as well, GRGA converges to the Pareto front. This can be seen from Figure 9.14. This figure also shows that GRGA is able to find well distributed Pareto-optimal solutions that span across the search space.

Roy (1997) applies Adaptive Restricted Tournament Selection (ARTS) to this problem, but considering only one objective, W_{cr} . The results obtained from ARTS lie on the global Pareto front, but as expected, are concentrated at its extreme end that corresponds to low values of W_{cr} and high values of T_{wg} . The application of NSGA-II to both the two- and four-objective versions of this problem leads to

convergence to the local front. This can be seen from Figure 9.11 for the two-objective case and from Figure 9.15 for the four-objective case. However, as revealed in these figures, NSGA-II is able to provide well distributed solutions that span across the local front.

9.7 Summary

This chapter has demonstrated the successful application of GRGA and GAVD in solving three real-life engineering design optimisation problems: design of a welded beam, a machine tool spindle and a turbine blade cooling system. Since these three problems constitute a representative set, it can be said that the successful application of GRGA and GAVD on these problems ensures their success in solving other problems listed in Table 4.1. In this way, this chapter has used real-life problems to validate the observations made previously regarding the capability of GRGA and GAVD in dealing with multiple objectives, constraints and variable dependence in engineering design optimisation problems. In particular, it has demonstrated that GRGA and GAVD can successfully handle inseparable function interaction and variable dependence in complex multi-objective optimisation problems with constraints. This chapter has also demonstrated that these two algorithms outperform a state-of-the-art optimisation algorithm, NSGA-II, on a wide variety of multi-objective optimisation problems. In short, this chapter has achieved the following.

- ◆ It has analysed a number of case studies from real-life engineering design optimisation.
- ◆ It has framed the selection criteria for choosing a representative set of case studies for this research.
- ◆ It has reported the experimental results obtained from GRGA, GAVD and NSGA-II.
- ◆ It has finally analysed these results in order to validate the performance of GRGA and GAVD.

10 DISCUSSION AND CONCLUSIONS

This chapter concludes this thesis with a discussion on the findings of this research. It also identifies the limitations of this work and the corresponding future research activities. This chapter aims to achieve the following.

- ◆ *To summarise key observations of this research.*
- ◆ *To identify the main contributions of this research.*
- ◆ *To discuss the limitations of this research.*
- ◆ *To frame future research activities based on this work.*

10.1 Discussion

This section discusses the key observations of this research. The generality of this research is also discussed in this section.

10.1.1 Key Observations of this Research

Traditional trial-and-error method of design optimisation is not capable of meeting the current industrial demands. Industries are, therefore, looking for automating the optimisation process using algorithms and computational techniques. However, the lack of flexibility and adequacy of existing optimisation techniques in dealing with the challenges of real-life engineering design optimisation problems has prevented the industry from adopting the optimisation algorithms. This research aims to explore the field of EC for developing techniques that are capable of dealing with the challenges posed by three features of real-life engineering design optimisation problems: multiple objectives, constraints and interaction among decision variables. This is a part of a broad initiative to make optimisation algorithms popular in industry. The key observations of this research are summarised here.

10.1.1.1 Literature Survey

The research has looked at the popular engineering design optimisation approaches in literature. It has surveyed the optimisation approaches for handling uncertainty, FEA/CFD analysis and sensitivity analysis. It has analysed the drawbacks of classical optimisation algorithms that have led to the growth of research in the area of EC. In order to assess the current capability of EC techniques in dealing with the challenges of engineering design optimisation, the research has carried out a detailed survey of these techniques with respect to three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction.

The two main goals of multi-objective optimisation are convergence to the Pareto front and maintenance of diversity across the front. It is observed that the elitist EMOTs that use Pareto domination and diversity-preserving operators perform better than other multi-objective optimisation techniques. However, it is observed that there are only a few EC techniques that specialise in handling constraints in multi-objective optimisation problems. The recently introduced concept of constraint-domination (Deb, 2000), which incorporates constraint violations in the definition of Pareto domination, has been shown to be successful in handling a variety of constrained multi-objective optimisation problems. This research classifies the interaction among decision variables into two broad categories: inseparable function interaction and variable dependence. Most of the EC techniques that are discussed above fail under the challenges, such as multi-modality, deception and discontinuity, posed by inseparable function interaction (epistasis). However, most of the current research in this field deals with single-objective optimisation in discrete domains. The few ETIFIs that are available for dealing with continuous search spaces have limited usefulness for real-life problems since in most cases they cannot handle multiple objectives and their performance is strongly dependent on the nature of the search space. The introduction of variable dependence introduces an additional level of complexity to the constrained, inseparable, multi-objective optimisation problems. Here, the variables are dependent on each other, implying that a two-step procedure needs to be appended to the EC: identification of dependency relationships (Step 1)

and classification of variables (Step 2). Although the literature reports some techniques that can individually deal with these two steps, there is a complete lack of dedicated EC frameworks for dealing with dependent-variable optimisation problems.

The literature survey also analyses the existing optimisation test functions with respect to their capability of simulating multiple objectives, constraints and variable interaction in engineering design optimisation problems. It is observed that most of the multi-objective optimisation test functions that are reported in literature are not tuneable in nature. Recently Deb (2001) proposed a tuneable strategy for constrained multi-objective optimisation, but it also provides only a limited control since it does not propose generic, parametric prototypes for the objective functions. Therefore, this scheme lacks a complete approach to multi-objective test bed development. Furthermore, the development of test beds for simulating variable interaction has not been adequately addressed by previous research in the area of optimisation. Although in recent years some researchers have talked about test function development for inseparable function interaction, literature does not report any test bed that can directly control the complexity introduced due to the inseparable function interaction in the objective functions of the problem. A much stronger observation was made regarding variable dependence, where it is observed that there is a complete lack of test problems for simulating variable dependence.

10.1.1.2 Industrial Context and Focus

In order to ground the research within the industrial context, an industrial survey is carried out in this research. Questionnaires are used for collection of information, and the number of companies visited is limited to six. The industry survey highlights that optimisation algorithms are not popular in industry. The survey also compiles the features of real-life engineering design optimisation problems that include multiple objectives, constraints and interaction among decision variables. It is observed in this survey that the lack of robust techniques for dealing with the features of real-life engineering design optimisation problems is one of the inhibitors to industrial applications of optimisation algorithms. This leads to the industrial

context of the research, which is to develop optimisation techniques that can handle, within a single framework, the following three features of real-life engineering design optimisation problems: presence of multiple objectives, constraints and variable interaction. Since it is difficult to find a variety of real-life cases with required complexities, this research also develops test beds that are capable of performing systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems.

10.1.1.3 Gap Analysis: EC versus Multiple Objectives, Constraints and Variable Interaction

This research compares the capabilities of the existing EC techniques against the challenges posed by multiple objectives, constraints and variable interaction in engineering design optimisation problems. This analysis reveals that there are effective techniques available in literature for handling multiple objectives and constraints. However, there is a research gap in EC techniques for handling variable interaction. This gap defines the main focus of this research, which is to develop EC techniques that can effectively handle the two types of variable interaction (inseparable function interaction and variable dependence) in constrained multi-objective optimisation problems, defined in hybrid search spaces (with integer and real variables).

Similar to the case of EC techniques, the areas of multi-objective and constrained optimisation test bed development are well addressed in literature as almost separate streams. However, there is a need to develop tuneable/parametric test beds that can simulate the complexity introduced by multiple objectives, constraints and variable interaction in a single framework. The research also attempts to address this gap.

10.1.1.4 Development of GRGA

This research identifies the challenges that inseparable function interaction poses for multi-objective optimisation algorithms. These challenges include multi-modality, deception, collateral noise and isolated optimum from the perspective of convergence to the Pareto front, and discontinuity, non-uniformity and shape complexity of the

Pareto front from the perspective of maintenance of diverse Pareto-optimal solutions. This research proposes a novel solution strategy to deal with these challenges. Furthermore, it applies this solution strategy to develop an algorithm, GRGA, which is capable of handling constrained multi-objective optimisation problems having complex inseparable function interaction. GRGA is based on the fact that there is existence of relationship(s) among the decision variables of the solutions belonging to the Pareto front. It explores this relationship using non-linear, multi-variable regression analysis. The relationship thus obtained is used for periodic and final redistribution of solutions over their respective fronts, guiding the search towards global Pareto front and determining the termination condition of the algorithm. GRGA has been shown to outperform the existing state-of-the-art multi-objective optimisation algorithm, NSGA-II. This algorithm has also been proved to be very effective in dealing with a variety of multi-objective optimisation problems. However, its performance is dependent on how accurately the relationships among the decision variables can be represented.

10.1.1.5 Development of GAVD

This research also deals with variable dependence, which is the second type of variable interaction. In the presence of variable dependence, the decision variables cannot be varied independently. Also, it has been proved here that the search space gets modified creating a new feasible region based on the dependence among the decision variables. This research observes that two additional steps need to be appended to GRGA for enabling it to deal with dependent-variable optimisation problems. These steps involve the identification of dependency relationships and the classification of variables into dependent and independent. This research proposes a novel algorithm, GAVD, which uses RA coupled with a DC to identify the dependency relationships. In this algorithm, the variables are classified as dependent and independent using a DT. GAVD makes use of GRGA as the optimisation engine. Here, the independent variables, identified by the DT, define the GA chromosome. For each alternative solution generated by the GA, the dependency equations are used to calculate the values of the dependent variables. It should be noted here that

the bounds on independent variables are treated as variable limits and those on dependent variables are treated as constraints. Hence, the presence of variable dependence has an effect of constraining the search space. GAVD exhibits successful performance on a number of dependent-variable optimisation problems. However, its capability to estimate the dependence among decision variables is limited by the degree of the RA that it uses.

10.1.1.6 Development of RETB and RETB-II

The development of optimisation algorithms requires systematic and controlled testing. However, since it is difficult to find a wide variety of real-life cases to support this, it is important to develop test beds that have the required features (multiple objectives, constraints and variable interaction), and enable controlled testing of algorithms. This research proposes two test beds, RETB and RETB-II, which provide a unified framework for controlled testing of optimisation algorithms with respect to the three features of real-life engineering design optimisation: presence of multiple objectives, constraints and variable interaction. To provide better control over the complexity of test functions, RETB and RETB-II also provide generic, parametric prototypes for each of the functions in their definition. The capability of RETB and RETB-II to handle multiple objectives, constraints and variable interaction in a single framework makes them generic in nature. Furthermore, the availability of parametric prototypes with these test beds also makes them fully tuneable.

10.1.1.7 Performance Analysis of GRGA and GAVD

This research applies RETB to compare the performance of GRGA with that of NSGA-II. The choice of NSGA-II for comparison is based on the fact that it has been shown in literature to outperform all existing techniques in dealing with multi-objective optimisation problems in real domains. The test problems that are chosen for analysis have complex inseparable function interaction leading to multi-modality in the search space, discontinuity in Pareto front and bias in the search space. It is shown that in all cases GRGA exhibits better convergence as compared to NSGA-II.

This is because the Pareto-domination/elitism strategy used by NSGA-II ceases to produce the driving force towards the global Pareto front once most of the solutions of the population share the same non-domination level. GRGA addresses this drawback of NSGA-II through periodic modification of regression coefficients based on their history of search observed in previous generations. This guides the search towards the global Pareto front by preventing it from getting trapped in local fronts. It is also observed that in most test functions GRGA provides better distribution of solutions than NSGA-II. The reason for this is that the Crowded Comparison Operator used in NSGA-II attempts to attain solution diversity using external means, without addressing the inherent features that lead to diversity problems. On the other hand, GRGA addresses the core issue of this problem by determining the relationships among the decision variables of the solutions, and using them to re-distribute the solutions for aiding their spread over the current front.

This research uses RETB-II for constructing dependent-variable test functions in order to compare the performance of GAVD against those of GRGA and NSGA-II. It is observed that in all those problems in which the introduction of variable dependence modifies the Pareto front, the GRGA and NSGA-II are not able to converge to the Pareto front. This is because of the inability of these algorithms to incorporate variable dependence. On the other hand, GAVD is able to identify variable dependency, and hence converges to the Pareto front. Furthermore, it is observed that since GAVD uses GRGA as its optimisation engine, it also inherits all its features for effectively dealing with inseparable function interaction in multi-objective optimisation problems.

10.1.1.8 Validation Using Real-life Problems

The performance of GRGA and GAVD is also tested on a set of real-life engineering design optimisation problems that together represent a variety of challenges that multiple objectives, constraints and variable interaction pose for optimisation algorithms. This representative set includes the design of a welded beam, a machine tool spindle and a turbine blade cooling system. The validation of results is performed here based on the published results and visualisation of the search space in

the presence of exhaustive search. The findings made through the testing of GRGA and GAVD with RETB and RETB-II are validated here. In particular, the application of GRGA and GAVD on these real-life problems demonstrates their capability of dealing with multiple objectives, constraints, inseparable function interaction and variable dependence in engineering design optimisation problems.

10.1.2 Main Contributions

This research has significantly contributed to understanding about the handling of variable interaction in engineering design optimisation problems. The research has mathematically defined and classified variable interaction. It has proposed two EC techniques, GRGA and GAVD, for handling variable interaction, and two test beds, RETB and RETB-II, for performing controlled testing of optimisation algorithms in the presence of multiple objectives, constraints and variable interaction.

The following points clearly identify the contribution to knowledge of this work. There was a research gap in all these areas that has subsequently been filled by this work.

- ◆ **Critical Analysis of Existing EC techniques:** The literature survey carried out in this research compares the capabilities of the existing EC techniques against the challenges posed by multiple objectives, constraints and variable interaction in engineering design optimisation problems. Based on the above analysis, this research has identified that there is a lack of techniques for handling variable interaction in engineering design optimisation problems. This has led to the identification of variable interaction as the main focus of this research for the development of optimisation algorithms and for the development of tuneable/parametric test beds.
- ◆ **Definition and Classification of Variable Interaction:** This research has defined and classified variable interaction into two broad categories: inseparable function interaction and variable dependence. It has further developed definitions for each of these categories.
- ◆ **Development of Test Beds for Engineering Design Optimisation:** It is difficult to find a wide variety of real-life design optimisation problems to support systematic and controlled testing of optimisation algorithms. Therefore, this

research has developed two parametric test beds, RETB and RETB-II that can simulate multiple objectives, constraints, inseparable function interaction and variable dependence in engineering design optimisation problems. In this way, these test beds enable controlled testing of optimisation algorithms with a large variety of problem features.

- ◆ **Development of Techniques for Handling Variable Interaction:** This research has compiled the challenges that the two categories of variable interaction pose for optimisation algorithms. Based on this, it has developed generic strategies for dealing with inseparable function interaction and variable dependence. These strategies have led to the development of two novel algorithms: GRGA for handling inseparable function interaction and GAVD for handling variable dependence. Furthermore, the research has compared the performance of GRGA, GAVD and NSGA-II based on a set of test problems generated by RETB and RETB-II. The techniques are also validated with real-life case studies.
- ◆ **Analysis of Real-life Optimisation problems:** This research has analysed a number of real-life engineering design optimisation problems, especially with respect to inseparable function interaction and variable dependence. It has selected a representative set from these problems, and used them to validate the performance of GRGA and GAVD,

10.1.3 Generality of Research

An attempt has been made in this research to keep it as general as possible. However, as with any other research, this work also has some limitations. Here, some of these limitations are identified.

10.1.3.1 Limitations of Research Methodology

The following are the main limitations of the methodology used in this research.

- ◆ In this research, semi-structured interviews with designers were used as tools for accessing the current status of design optimisation in industry. The limitations of this survey are as follows.
 - *The use of questionnaires as the method of data collection, though useful, has its limitations since the surveyors do not directly observe the optimisation process.*

- *The companies visited did not cover a full spectrum of the industry sectors.*
- *Although the questionnaire contained both 'open' and 'closed' questions, the analysis of results was mostly qualitative in nature, making it prone to the errors of subjectivity. To reduce this error, a systematic analysis was carried out to identify the factors common to most companies and those that are prevalent only in a few companies. However, the analysis provided little insight into the quantitative/statistical aspects of the information gathered.*
- ◆ The industry survey identified a number of issues that inhibit industrial applications of optimisation algorithms. However, this research has chosen only one of these issues for analysis. This issue is the lack of robust optimisers for handling the features of real-life engineering design optimisation problems. Therefore, the research methodology focuses on developing EC/GA techniques for engineering design optimisation.
- ◆ The algorithm used for comparison in this research is NSGA-II. The choice of this algorithm is justified since it outperforms most of the existing algorithms in dealing with multi-objective optimisation problems.
- ◆ The real-life case studies that are used in this research are borrowed from literature. This has provided a limited insight into the process of model development for optimisation. Furthermore, the nature of case studies could have been better understood if they were designed in consultation with the industrial designers.
- ◆ This research carried out the process of validation by performing exhaustive search on the model of the real-life problem, and analysing the location of solutions with respect to the Pareto front depicted by the exhaustive search. Although this method gives useful information regarding the performance of the optimisation algorithms, it provides only a limited insight into the usefulness of the attained results for designers in industry.

10.1.3.2 Limitations of GRGA

The limitations of GRGA are as follows.

- ◆ The performance of GRGA is dependent on how accurately the relationship among decision variables can be represented in the RA that it uses. Hence, use of

more sophisticated non-linear modelling tools have the potential of improving its performance.

- ◆ In case of very complex relationships (high order, non-linear) among the decision variables of the Pareto-optimal solutions, the performance of RA (used in GRGA) is expected to deteriorate. This is an inherent limitation of the RA technique used.
- ◆ As the number of dimensions and objectives in the problem is increased, the effectiveness of this algorithm in producing well distributed solutions exhibits a drop since it becomes more difficult for HDA (used in GRGA for distributing solutions) to evenly distribute the solutions.
- ◆ The performance of this algorithm drops as the number of relationships among the decision variables of the Pareto-optimal solutions increases. This is due to the difficulty of determining and maintaining all the relationships in the population.
- ◆ This algorithm is not capable of dealing with dependence among decision variables and qualitative issues such as manufacturability and designers' special preferences.

10.1.3.3 Limitations of GAVD

The limitations of GAVD are as follows.

- ◆ Since GAVD uses GRGA as its optimisation engine, it shares all the above-mentioned limitations of GRGA. GAVD gives better results than GRGA only when the introduction of variable dependence actually modifies the search space. In addition, it has the following limitations in its strategy of handling dependence among the decision variables.
- ◆ The performance of this algorithm in accessing the dependence among decision variables is limited by the degree of RA that it uses. Hence, in dealing with complex dependence, higher order RAs are required. This implies that the use of more sophisticated non-linear modelling tools, such as Neural Networks, have the potential of improving its performance, especially in modelling deceptive and complex non-linear functions.
- ◆ GAVD also needs to be fitted with a mechanism that can learn the dependency relationships, and update it each time a new data is added, without having to repeat the whole process.

- ◆ GAVD also needs enhancements to deal with noisy data.
- ◆ Finally, this algorithm is not capable of dealing with qualitative issues.

10.1.3.4 Limitations of Proposed Test Beds

The limitations of RETB and RETB-II are as follows.

- ◆ RETB is incapable of generating test problems that have dependence among their decision variables.
- ◆ Both RETB and RETB-II currently simulate only static environments. They are incapable of dealing with uncertain or dynamic conditions in the search space.
- ◆ As mentioned in Chapter 7, the relative sequence of dependent and independent variables has an influence on those GAs whose performance is dependent on variable sequence. However, RETB-II does not control the effects of changing the sequence of variables in the GA chromosome.
- ◆ It also needs to be further studied how the choice of a particular noise distribution effects the complexity of RETB-II test problems.

10.2 Future Research

Despite the immense potential of optimisation algorithms and the recognition of their need by industry, it was observed from the industry survey that no company surveyed uses any optimisation algorithm as a day to day tool. In order to address this grim situation, it is required to carry out a number of research activities in the area of real-life optimisation. Addressing the limitations of this research forms a part of these activities.

This research concentrates on engineering design optimisation problems. There is therefore a need to extend this research to include other areas of real-life optimisation, such as combinatorial problems (e.g. scheduling). The use of NN, FL and other areas of EC could be explored for enhancing the current capabilities of GRGA and GAVD. More tests are required to compare GRGA and GAVD against other optimisation algorithms.

The research activities for future development of GRGA can be summarised as follows.

- ◆ Further research needs to be carried out to enhance GRGA for incorporating designers' preferences for certain objectives at an intermediate stage of the optimisation process.
- ◆ The use of sophisticated non-linear modelling tools for enabling GRGA to deal with complex relationships among the decision variables of the Pareto-optimal solutions is an important area of future research.
- ◆ The distribution strategies used in GRGA can be further improved to make them more scalable with respect to number of objectives and dimensions.
- ◆ The use of sophisticated clustering techniques could be analysed for enabling GRGA to solve those problems that have multiple relationships among the decision variables of the Pareto-optimal solutions.

The future research activities for further development of GAVD are as follows.

- ◆ The performance of GAVD needs to be studied in the presence of other data modelling techniques in place of RA.
- ◆ An NN-type strategy could be provided with GAVD to enable it to learn the dependency relationships, and update them with the addition of new data.
- ◆ The enhancement of GAVD to deal with noisy data is another area of future research.

Finally, the future research activities corresponding to RETB and RETB-II are as follows.

- ◆ RETB/RETB-II could form the basis of classification of real-life engineering design optimisation problems. Further research needs to be carried out in this direction.
- ◆ Disturbance modelling techniques could be incorporated with RETB and RETB-II for enabling them to simulate uncertain or dynamic environments.
- ◆ The dependence of problem complexity on the sequence of variables in the GA chromosome needs to be modelled and incorporated within RETB and RETB-II.
- ◆ There is also a need to develop guidelines for illustrating the dependence of the noise distribution on the complexity of the RETB-II test problems.

10.3 Conclusions

This section compares the achievements of this research with the objectives stated in Chapter 3. The following discussion analyses one research objective at a time, and compares it with what is achieved in this research.

- ◆ This thesis provides a literature survey of evolutionary-based optimisation techniques. The research classifies the EC techniques, and analyses them with respect to the three features of engineering design optimisation problems: multiple objectives, constraints and variable interaction. Here, the interaction among decision variables is also defined, and classified into two broad categories: inseparable function interaction and variable dependence. This survey reveals that there are effective techniques available in literature for handling multiple objectives and constraints. However, there is a research gap in EC techniques for handling variable interaction. This gap defines the main focus of this research, which is to develop EC techniques that can effectively handle the two types of variable interaction in constrained multi-objective optimisation problems.
- ◆ The research also carries out a literature survey of existing test functions for evaluating their capabilities of performing systematic and controlled simulation of multiple objectives, constraints and variable interaction in optimisation problems. Similar to the case of EC techniques, the areas of multi-objective and constrained optimisation test bed development are well addressed in literature as almost separate streams. However, there is a need to develop tuneable/parametric test beds that can simulate the complexity introduced by multiple objectives, constraints and variable interaction in a single framework. The research also attempts to address this gap.
- ◆ An industry survey is carried out to ground the research within the industrial context. This survey highlights that optimisation algorithms are not popular in industry. It also compiles the features of real-life engineering design optimisation problems that include multiple objectives, constraints and interaction among decision variables. It is observed in this survey that the lack of robust techniques for dealing with the features of real-life engineering design optimisation problems is one of the inhibitors to industrial applications of optimisation algorithms. This leads to the industrial context of the research, which is to

develop optimisation techniques that can handle, within a single framework, the following three features of real-life engineering design optimisation problems: presence of multiple objectives, constraints and variable interaction.

- ◆ This research develops two EC techniques, GRGA and GAVD, for handling interaction among decision variables. GRGA specialises in dealing with inseparable function interaction in constrained multi-objective optimisation problems. GAVD is an extension of GRGA for dealing with dependent-variable optimisation problems. It provides a complete framework for handling multiple objectives in the presence of constraints, inseparable function interaction and variable dependence.
- ◆ This research proposes two test beds, RETB and RETB-II. These test beds together provide a unified, parametric framework for controlled testing of optimisation algorithms in the presence of the three features of real-life engineering design optimisation problems: the presence of multiple objectives, constraints and variable interaction.
- ◆ This research uses RETB and RETB-II for detailed performance analysis of GRGA and GAVD. This performance analysis establishes the superiority of GRGA in handling inseparable function interaction and of GAVD in handling variable dependence in multi-objective optimisation problems. It is observed that GRGA and GAVD outperform NSGA-II, which is currently the best-performing multi-objective optimisation technique.
- ◆ This research analyses a number of real-life engineering design optimisation problems, especially with respect to variable interaction. It selects a representative set from these problems, and analyses the performance of GRGA and GAVD on this representative set. The performance of GRGA and GAVD is also compared with NSGA-II on these problems. The validation is performed here based on the published results and through visualisation in the presence of exhaustive search.

The achievements of this research can be briefly and precisely stated as follows.

- ◆ Critical analysis of existing EC techniques.
- ◆ Definition and classification of variable interaction.

- ◆ Development of test beds for simulating multiple objectives, constraints and variable interaction in engineering design optimisation problems (RETB and RETB-II).
- ◆ Development of techniques for handling variable interaction (GRGA and GAVD).
- ◆ Analysis of real-life engineering design optimisation problems, especially from interaction point of view.

In this way, the research has proposed a fully tested and validated methodology for dealing with engineering design optimisation problems with variable interaction.

11 REFERENCES

- Back, T., Hammel, U. and Schwefel, H.P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE trans. on evolutionary computation*, 1(1), 3-17.
- Bagley, J.D. (1967). *The behavior of adaptive systems which employ genetic and correlation algorithms*, PhD. Thesis, University of Michigan, USA.
- Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning*, Technical Report No. CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, USA. (unpublished).
- Baluja, S. and Davies, S. (1997). Using optimal dependency-tree for combinatorial optimization: Learning the structure of the search space. In: *Proceedings of the 14th international conference on machine learning*, 30-38, Morgan Kaufmann, USA.
- Banzhaf, W., Nordin, P., Keller, R.E., and Francone, F.D. (1998). *Genetic programming: An introduction*. Morgan Kaufmann Publishers, Inc., San Francisco, California, USA.
- Barbosa, H.J.C. (1996). A genetic algorithm for min-max problems. In: *Proceedings of the First International Conference on Evolutionary Computation and Its Application (EvCA '96)*, 99-109.
- Beale, E.M.L. (1958). *On an iterative method for finding a local minimum of a function of more than one variable*, Technical Report No. 25, Statistical Techniques Research Group, Princeton University, Princeton, NJ, USA. (unpublished).
- Beasley, D., Bull, D. and Martin, R. (1993). An overview of genetic algorithms: Part 2, research topics. *University computing*, 15(4), 170-181.

- Belegundu, A.D., Murthy, D.V., Salagame, R.R. and Constants, E.W. (1994). Multiobjective optimization of laminated ceramic composites using genetic algorithms. In: *Proceedings of the fifth AIAA/USAF/NASA symposium on multidisciplinary analysis and optimization*, 1015-1022.
- Bentley, P.J. and Wakefield, J.P. (1998). Generic evolutionary design. In: *Soft computing in engineering design and manufacturing*, edited by Chawdhry, P.K., Roy, R. and Pant, R.K. Springer, London, UK.
- Binh, T.T. and Korn, U. (1997). MOBES: A multiobjective evolution strategy for constrained optimization problems. In: *The third international conference on genetic algorithms (Mendel'97)*, 176-182.
- Bishop, C.M. (1996). *Neural networks for pattern recognition*. Oxford University Press, Oxford, UK.
- Box, M.J. (1965). A new method of constrained optimization and a comparison with other methods. *Computer journal*, 8(1), 42-52.
- Box, M.J. (1966), A comparison of several current optimization methods and the use of transformations in constrained problems. *Computational journal*, 9, 67-77.
- Cauchy, A.L. (1847). Methode generale pour la resolution des systemes d'equations simultanees. *Comptes rendus de l'academie des sciences*, 25, 536-538.
- Chandrasekran, B. (1990). Design problem solving: A task analysis. *AI magazine*, AAAI, 59-71.
- Charnes, A. and Cooper, W.W. (1959). Chance constrained programming. *Management science*, 6, 73-79.
- Cheney, E.W. and Goldstein, A.A. (1959). Newton's method of convex programming and Tchebycheff approximation. *Numerische mathematik*, 1, 253-268.
- Chipperfield, A. and Fleming, P. (1995). Gas turbine engine controller design using multiobjective genetic algorithms. In: *Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems:*

- Innovations and Applications* (GALESIA'95), edited by Zalzala, A.M.S., 214-219, Halifax Hall, University of Sheffield, UK.
- Coello, C.A.C. (1997). *An empirical study of evolutionary techniques for multiobjective optimization in engineering design*. PhD Thesis, Department of Computer Science, Tulane University, New Orleans, LA, USA.
- Coello, C.A.C. (1998). *An updated survey of GA-based multiobjective optimization techniques*. Technical Report No. Lania-RD-98-08, Laboratorio Nacional de Informatica Avanzada (LANIA), Xalapa, Veracruz, Mexico. (unpublished).
- Coello, C.A.C. (1999). An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In: *1999 congress on evolutionary computation*, Washington DC, USA, July 1999, 3-13. IEEE, Washington DC, USA.
- Coello, C.A.C. (2001). A short tutorial on evolutionary multiobjective optimization. In: *Evolutionary multi-criterion optimization*, edited by Zitzler, E., Deb, K., Thiele, L., Coello, C.A.C. and Corne, D. Lecture notes in computer science (LNCS), Vol. 1993. Springer-Verlag, Berlin, Germany.
- Coello, C.A.C. and Christiansen, A.D. (1999). MOSES: A multiobjective optimization tool for engineering design. *Engineering optimization*, 31(3), 337-368.
- Coello C.A.C., Christiansen, A.D., and Aguirre, A.H. (1998). Using a new GA-based multiobjective optimization technique for the design of robot arms, *Robotica*, 16(4), 401-414.
- Cunha, A.G. (2000). *Modelling and optimisation of single screw extrusion*, PhD. Thesis, University of Minho, Guimarães, Portugal.
- Cunha, A.G., Oliveira, P. and Covas, J.A. (1997). Use of genetic algorithms in multicriteria optimization to solve industrial problems. In: *Proceedings of the seventh international conference on genetic algorithms*, 682-688.

} C E C

- Cvetkovic, D. and Parmee, I. (1998). Evolutionary design and multi-objective optimisation. In: *Proceedings of the Sixth European Congress on Intelligent Techniques and Soft Computing (EUFIT)*, 397-401.
- Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2, 303-314.
- Davidor, Y. (1990). Epistasis variance: Suitability of a representation to genetic algorithms. *Complex systems*, 4, 369-383.
- Davidor, Y. (1991). Epistasis variance: A viewpoint on GA-hardness. In: *Foundations of Genetic Algorithms (FOGA)*, edited by Rawlins, G.J.E., Morgan Kaufmann, San Mateo, CA, USA.
- De Bonet, J.S., Isbell, C.L. and Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. In: *Advances in neural information processing systems*, edited by Mozer, M.C., Jordan, M.I. and Petsche, T., Vol. 9, 424, The MIT Press, Cambridge, USA.
- De Jong, K. (1975). *An analysis of the behavior of a class of genetic adaptive systems*, PhD. Thesis, University of Michigan, Ann Arbor, MI, USA.
- Deb, K. (1995). *Optimization for engineering design: Algorithms and examples*. Prentice-Hall, New Delhi, India.
- Deb, K. (1999a). Evolutionary algorithms for multi-criterion optimization in engineering design. In: *Evolutionary algorithms in engineering and computer science*, edited by Miettinen, K., Makela, M.M., Neittaanmaki, P. and Periaux, J., John Wiley & Sons, Ltd., UK.
- Deb, K. (1999b). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary computation*, 7(3), 205-230.
- Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2-4), 311-338.
- Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, Ltd., New York, USA.

- Deb, K. and Goel, T. (2001). A hybrid multi-objective evolutionary approach to engineering shape design. In: *Evolutionary multi-criterion optimization*, edited by Zitzler, E., Deb, K., Thiele, L., Coello, C.A.C. and Corne, D. Lecture notes in computer science (LNCS), Vol. 1993. Springer-Verlag, Berlin, Germany.
- Deb, K., Pratap, A. and Meyarivan, T. (2001). Constrained test problems for multi-objective evolutionary optimization. In: *Evolutionary multi-criterion optimization*, edited by Zitzler, E., Deb, K., Thiele, L., Coello, C.A.C. and Corne, D. Lecture notes in computer science (LNCS), Vol. 1993. Springer-Verlag, Berlin, Germany.
- Deb, K., Pratap, A. and Moitra, S. (2000). Mechanical component design for multi-objective using elitist non-dominated sorting GA. In: *Parallel Problem Solving from Nature V (PPSN-V)*, Lecture notes in computer science, 859-868, Springer-Verlag, Berlin, Germany.
- Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2000). *A fast and elitist multi-objective genetic algorithm: NSGA-II*. KanGAL Report No. 200002, Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology (IIT), Kanpur, India. (unpublished).
- Draper, N.R. and Smith, H. (1998). *Applied regression analysis*. John Wiley and Sons, Inc., New York, USA.
- EPSRC. (2001). *Citing Internet resources* (WWW document). <http://www.epsrc.ac.uk>. (accessed 3rd November 2001).
- Emch, G. and Parkinson, A. (1993). Using engineering models to control variability: Feasibility robustness for worst-case tolerances. In: *Advances in design automation*, Vol. 1, DE-Vol. 65-1, ASME, USA.
- Evans, J.R. and Olson, D.L. (2000). *Statistics data analysis, and decision modelling*. Prentice Hall, NJ, USA.
- Fletcher, R., and Powell, M.J.D. (1963), A rapidly convergent descent method for minimization, *Computational journal*, 6, 163-168.

- Fletcher, R. and Reeves, C.M. (1964). Function minimization by conjugate gradients. *Computer journal*, 7, 149-154.
- Fonlupt, C., Robilliard, D. and Preux, P. (1998). A bit-wise epistasis measure for binary search spaces. In: *Parallel Problem Solving from Nature V (PPSN-V)*, edited by Eiben, A.E., Back, T., Schoenauer, M. and Schwefel, H.-P., 47-56, Springer, Berlin, Germany.
- Fonseca, C.M. and Fleming, P.J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: *Proceedings of the fifth international conference on genetic algorithms*. edited by Forrest, S., 416-423, Morgan Kaufman Publishers, USA.
- Fonseca, C.M. and Fleming, P.J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary computation*, 3(1), 1-16.
- Fonseca, C.M. and Fleming, P.J. (1998a). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation. *IEEE transactions on systems, man and cybernetics – Part A: Systems and humans*, 28(1), 26-37.
- Fonseca, C.M. and Fleming, P.J. (1998b). Multiobjective optimization and multiple constraint handling with evolutionary algorithms – Part II: Application example. *IEEE transactions on systems, man, and cybernetics - Part A: Systems and humans*, 28(1), 38-47.
- Fourman, M.P. (1985). Compaction of symbolic layout using genetic algorithms. In: *Genetic algorithms and their applications: Proceedings of the first international conference on genetic algorithms*, 141-153, Lawrence Erlbaum, USA.
- Frantz, D.R. (1972). *Non-linearities in genetic adaptive search*, PhD. Thesis, University of Michigan, USA.
- Frees, E.W. (1996). *Data analysis using regression models: The business perspective*. Prentice Hall, NJ, USA.

- Gallagher, M., Freaan, M., and Downs, T. (1999). Real-valued evolutionary optimization using a flexible probability density estimator. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 1999)*, Vol. 1, 840-846. Morgan Kaufmann Publishers, San Francisco, California, USA.
- Gershenfeld, N. (1999), *The nature of mathematical modelling*, Cambridge University Press, Cambridge, UK.
- Goldberg, D.E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Publishing Company, Massachusetts, USA.
- Goldberg, D.E. and Bridges, C.L. (1990). An analysis of a reordering operator on a GA-hard problem. *Biological cybernetics*, 62, 397-405.
- Goldberg, D.E. and Lingle, R.Jr. (1985). Alleles, loci, and the traveling salesman problem. In: *Proceedings of an international conference on genetic algorithms and their applications*, 154-159.
- Goldberg, D.E., Deb, K. and Clark, J.H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex systems*, 6, 333-362.
- Goldberg, D.E., Deb, K. and Thierens, D. (1993). Toward a better understanding of mixing in genetic algorithms. *Journal of the society of instrument and control engineers*, 32(1), 10-16.
- Goldberg, D.E., Korb, B. and Deb, K. (1989). Messy genetic algorithms: Motivation, analysis, and first results. *Complex systems*, 3(5), 493-530.
- Greene, J. (1998). Simulated evolution and adaptive search in engineering design – experiences at the University of Cape Town. In: *Soft computing in engineering design and manufacturing*, edited by Chawdhry, P.K., Roy, R. and Pant, R.K. Springer, London, UK.
- Hajela, P. and Lin, C.Y. (1992). Genetic search strategies in multicriterion optimal design. *Structural optimization*, 4, 99-107.

- Harik, G.R. (1997). *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. PhD. Thesis, Computer Science and Engineering, University of Michigan, USA.
- Harik, G.R. (1999). *Linkage learning via probability modeling in the ECGA*, IlliGAL Report No. 99010, University of Illinois at Urbana-Champaign, Urbana, USA. (unpublished).
- Harik, G.R., Lobo, F.G. and Goldberg, D.E. (1997). *The compact genetic algorithm*, IlliGAL Report No. 97006, University of Illinois at Urbana-Champaign, Urbana, USA. (unpublished).
- Hertz, J.A., Krogh, A.S., and Palmer, R.G. (1991). *Introduction to the theory of neural computation*. Addison-Wesley, Redwood City, CA, USA.
- Holland, J.H. (1975). *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, USA.
- Homaifar, A., Lai, S.H.Y. and Qi, X. (1994). Constrained optimization via genetic algorithms. *Simulation*, 62, 242-254.
- Hooke, R. and Jeeves, T.A. (1961). Direct search solution of numerical and statistical problems. *Journal of the ACM*, 8(2), 212-229.
- Horn, J. (1997). Multicriterion decision making. In: *Handbook of evolutionary computation*, edited by Back, T., Oxford University Press, UK.
- Horn, J. and Nafplotis, N. (1993). *Multiobjective optimization using the niched Pareto genetic algorithm*. IlliGAL Technical Report No. 93005, University of Illinois, USA (unpublished).
- Hu, X. (1996). An evolutionary approach to hardware/software partitioning. In: *Parallel problem solving from nature – PPSN-IV*. edited by Voigt, H.M. et al., 900-909, Springer, Germany.
- Jared, G., Roy, R., Grau, J. and Buchannan, T. (1998). Flexible optimisation within the CAD/CAM environment. In: *CIRP international seminar on intelligent*

- computation in manufacturing engineering*. Capri, Italy, July 1-3 1998, 503-508.
- Jimenez, F., Verdegay, J.L. and Gomez-Skarmeta, A.F. (1999). Evolutionary techniques for constrained multiobjective optimization problems. In: *Proceedings of the workshop on multi-criterion optimization using evolutionary methods held at Genetic and Evolutionary Computation Conference (GECCO-1999)*, 115-116. Morgan Kaufmann Publishers, USA.
- Joines, J.A. and Houck, C.R. (1994). On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In: *Proceedings of the evolutionary computation conference – poster sessions*, 579-584. IEEE World Congress on Computational Intelligence.
- Kargupta, H. (1998). Revisiting the GEMGA: Scalable evolutionary optimization through linkage learning. In: *Proceedings of 1998 IEEE international conference on evolutionary computation*, 603-608, IEEE, USA.
- Kaymaz, I., McMahon, C. and Meng, X. (1998). Reliability based structural optimisation using the response surface method and Monte Carlo simulation. In: *8th international machine design and production conference*, Ankara, Turkey, September 9-11, 369-378.
- Keane, A.J. (1996). A brief comparison of some evolutionary optimisation methods. In: *Modern heuristic search methods*, edited by Rayward-Smith, V., Osman, I., Reeves, C. and Smith, G.D. John Wiley, London, UK.
- Kita, H., Yabumoto, Y., Mori, N. and Nishikawa, Y. (1996). Multi-objective optimization by means of thermodynamical genetic algorithm. In: *Proceedings of Parallel Problem Solving from Nature IV (PPSN-IV)*, edited by Voigt, H.-M., Ebeling, W., Rechenberg, I. and Schwefel, H.-P., Lecture notes in computer science, 504-512, Springer-Verlag, Berlin, Germany.
- Knowles, J.D. and Corne, D.W. (2000). Approximating the non-dominated front using the Pareto archived evolution strategy. *Evolutionary computation journal*, 8(2), 149-172.

- Kolmogorov, A.N. (1957). On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition. *Doklady Akademiia Nauk SSSR*, 114, 953-956.
- Koziel, S. and Michalewicz, Z. (1998). A decoder-based evolutionary algorithm for constrained parameter optimization. In: *Parallel Problem Solving from Nature V (PPSN-V)*, Lecture notes in computer science, 231-240, Springer-Verlag, Berlin, Germany.
- Kreyszig, E. (1993). *Advanced engineering mathematics*. Wiley Eastern Limited, New Delhi, India.
- Kursawe, F. (1990). A variant of evolution strategies for vector optimization. In: *Parallel Problem Solving from Nature I (PPSN I)*, edited by Schwefel, H. P. and Männer, R. Lecture notes in computer science, Vol. 496, 193-197, Springer-Verlag, Berlin, Germany.
- Kvasnicka, V., Pelikan, M. and Pospichal, J. (1996). Hill climbing with learning (An abstraction of genetic algorithm). *Neural network world*, 6, 773-796.
- Larranaga, P., Etxeberria, R., Lozano, J.A., and Pena, J.M. (1999). *Optimization by learning and simulation of Bayesian and Gaussian networks*. Technical Report No. EHU-KZAA-IK-4/99, Intelligent Systems Group, Department of Computer Science and Artificial Intelligence, University of the Basque Country, Spain. (unpublished).
- Laumanns, M., Rudolph, G. and Schwefel, H.P. (1998). A spatial predator-prey approach to multi-objective optimisation: A preliminary study. In: *Proceedings of the Parallel Problem Solving from Nature V (PPSN-V)*, edited by Eiben, A. E., Schoenauer, M. and Schwefel, H.-P., 241-249, Springer-Verlag, Amsterdam, Holland.
- Levenick, J.R. (1995). Metabits: Generic endogenous crossover control. In: *Proceedings of the sixth international conference on genetic algorithms*, 88-95, Morgan Kaufmann Publishers, USA.

- Liu, B., Haftka, R.T., Akgin, M.A. and Todoroki, A. (2000). Permutation genetic algorithm for stacking sequence design of composite laminates. *Computer methods in applied mechanics and engineering*, 186(2-4), 357-372.
- Liu, X., Bregg, D.W. and Fishwick, R.J. (1998). Genetic approach to optimal topology/controller design of adaptive structures. *International journal for numerical methods in engineering*, 41, 815-830.
- Marquardt, D. (1963). An algorithm for least squares estimation of nonlinear parameters. *SIAM journal of applied mathematics*, 11(2), 431-441.
- McMahon, C.A. and Meng, X. (1996). A network approach to parametric design integration. *Research in engineering design*, 8, 14-32.
- Melchers, R.E. (1989). Importance sampling in structural systems. *Structural safety*, 6, 3-10.
- Meng, X. and McMahon, C.A. (1998). Use of influence surfaces in a network approach to probabilistic analysis. *Computer modeling and simulation in engineering*, 3(1), 12-19.
- Michalewicz, Z. (1995). A survey of constraint handling techniques in evolutionary computation methods. In: *Proceedings of the evolutionary programming IV conference*, 135-155, MIT Press, UK.
- Michalewicz, Z. and Attia, N. (1994). Evolutionary optimization of constrained problems. In: *Proceedings of the 3rd annual conference on evolutionary programming*, edited by Sebald, A.V. and Fogel, L.J., 98-108, World Scientific Publishing, USA.
- Michalewicz, Z. and Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary computation journal*, 4(1), 1-32.
- Michalewicz, Z., Deb, K., Schmidh, M. and Stidsen, T. (2000). Test-case generator for nonlinear continuous parameter optimization techniques. *IEEE transactions on evolutionary computation*, 4(3), 197-215.

- Miettinen, K. (1999). *Nonlinear multiobjective optimization*. Kluwer, Boston, USA.
- Mitra, K., Deb, K. and Gupta, S.K. (1998). Multiobjective dynamic optimization of an industrial nylon 6 semibatch reactor using genetic algorithms. *Journal of applied polymer science*, 69(1), 69-87.
- Muhlenbein, H. (1997). The equation for response to selection and its use for prediction. *Evolutionary computation*, 5(3), 303-346.
- Muhlenbein, H. and Mahnig, T. (1999a). The factorized distribution algorithm for additively decomposed functions. In: *Proceedings of the 1999 congress on evolutionary optimization*, 752-759.
- Muhlenbein, H. and Mahnig, T. (1999b). FDA - A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary computation*, 7(4), 353-376.
- Muhlenbein, H. and Paab, G. (1996). From recombination of genes to the estimation of distributions I. Binary parameters. In: *Parallel Problem Solving from Nature IV (PPSN-IV)*, Lecture notes in computer science, 46-55, Springer-Verlag, Berlin, Germany.
- Murata, T. and Ishibuchi, H. (1995). MOGA: Multi-objective genetic algorithms. In: *Proceedings of the second IEEE international conference on evolutionary computation*, 289-294, IEEE, Perth, Australia.
- Mussa, R., Roy, R. and Jared, G. (1998). Surface optimisation within the CAD/CAM environment using genetic algorithms. In: *Advances in soft computing*, edited by Roy, R., Furuhashi, T. and Chawdhry, P. K. Springer, London, UK.
- Nikolaidis, E. and Burdisso, R. (1988). Reliability based optimisation: A safety index approach. *Computers & structures*, 28, 781-788.
- Nissan. (2001). *Citing Internet resources* (WWW document). <http://www.nissan-global.com/EN/HOME/PROFILE/04.html>. (accessed 3rd November 2001).
- Obayashi, S. (1997). Pareto genetic algorithm for aerodynamic design using the Navier-Stokes equations. In: *Genetic algorithms and evolution strategies in*

- engineering and computer science: Recent advances and industrial applications*, edited by Quagliarella, D., Periaux, J., Poloni, C. and Winter, G. Chapter 12, 245-266. John Wiley and Sons, West Sussex, UK.
- Obayashi, S., Takahashi, S. and Takeguchi, Y. (1998). Niching and elitist models for MOGAs. In: *Parallel Problem Solving from Nature V (PPSN-V)*, Lecture notes in computer science, 260-269, Springer-Verlag, Berlin, Germany.
- Oppenheim, A.N. (1992). *Questionnaire design, interviewing and attitude measurement*. Pinter Publishers, London, UK.
- Orvosh, D. and Davis, L. (1993). Shall we repair? Genetic algorithms, combinatorial optimization, and feasibility constraints. In: *Proceedings of the fifth international conference on genetic algorithms*, 650-657, Morgan Kaufmann Publishers, USA.
- Osyczka, A. (1984). *Multicriterion optimization in engineering with FORTRAN programs*, Ellis Horwood Limited, USA.
- Osyczka, A. (1985). Multicriteria optimization for engineering design. In: *Design optimization*, edited by Gero, J.S., Academic Press, London, UK.
- Osyczka, A. and Kundu, S. (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm, *Structural optimization*, 10(2), 94-99.
- Paredis, J. (1994). Coevolutionary constraint satisfaction. In: *Parallel Problem Solving from Nature III (PPSN-III)*, Lecture notes in computer science, 46-55, Springer-Verlag, Berlin, Germany.
- Paredis, J. (1995). The symbiotic evolution of solutions and their representations. In: *Proceedings of the sixth international conference on genetic algorithms*, 359-365.
- Parmee, I.C. (1996). Towards an optimal engineering design process using appropriate adaptive search strategies. *Journal of engineering design*. 7(4), 341-362.

- Pedrycz, W. (1998). *Computational intelligence – an introduction*. CRC Press, New York, USA.
- Pelikan, M. and Muhlenbein, H. (1999). The bivariate marginal distribution algorithm. In: *Advances in soft computing – Engineering design and manufacturing*, edited by Roy, R., Furuhashi, T. and Chawdhry, P.K., 521-535, Springer-Verlag, London, UK.
- Pelikan, M., Goldberg, D.E., and Cantu-Paz, E. (1998). *Linkage problem, distribution estimation, and Bayesian networks*. IlliGAL Report No. 98013, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, USA (unpublished).
- Pelikan, M., Goldberg, D.E., and Cantu-Paz, E. (1999). BOA: The Bayesian optimization algorithm. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 1999)*, Vol. 1, 525-532, Morgan Kaufmann Publishers, San Francisco, California, USA.
- Phadke, M.S. (1989). *Quality engineering using robust design*, Prentice-Hall International Inc., London, UK.
- Poloni, C. (1997). Hybrid GA for multi-objective aerodynamic shape optimization. In: *Genetic algorithms in engineering and computer science*, edited by Winter, G., Periaux, J., Galan, M. and Cuesta, P., 397-414, Chichester, Wiley.
- Poloni, C. and Pediroda, V. (1997). GA coupled with computationally expensive simulations: Tools to improve efficiency. In: *Genetic algorithms and evolution strategies in engineering and computer science: Recent advances and industrial applications*, edited by Quagliarella, D., Periaux, J., Poloni, C. and Winter, G., Chapter 13, 267-288, John Wiley and Sons, West Sussex, UK.
- Poloni, C., Giurgevich, A., Onesti, L. and Pediroda, V. (2000). Hybridization of a multiobjective genetic algorithm, a neural network and a classical optimizer for complex design problem in fluid dynamics. *Complex methods in applied mechanics and engineering*, 186(2-4), 403-420.

- Powell, M.J.D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer journal*, 7(4), 303-307.
- Powell, D. and Skolnick, M.M. (1993). Using genetic algorithms in engineering design optimization with non-linear constraints. In: *Proceedings of the fifth international conference on genetic algorithms*. 424-430, Morgan Kaufmann Publishers, USA.
- Quagliarella, D. and Vicini, A. (1997). Coupling genetic algorithms and gradient based optimization techniques. In: *Genetic algorithms and evolution strategies in engineering and computer science: Recent advances and industrial applications* edited by Quagliarella, D., Periaux, J., Poloni, C. and Winter, G., Chapter 14, 289-309, John Wiley and Sons, West Sussex, UK.
- Rao, S.S. (1996). *Engineering optimization – theory and practice*, Wiley-Interscience, USA.
- Ray, T., Tai, K. and Seow, K.C. (2001). An evolutionary algorithm for multiobjective optimization. *Engineering optimization*, 33(3), 399-424.
- Reddy, M.V., Grandhi, R.V. and Hopkins, D.A. (1994). Reliability based structural optimisation: A simplified safety index approach. *Computers & structures*, 53(6), 21407-1418.
- Reeves, C.R. and Wright, C.C. (1995a). An experimental design perspective on genetic algorithms. In: *Foundations of Genetic Algorithms III (FOGA-III)*, edited by Whitley, D. and Vose, M., 7-22, Morgan Kaufmann, San Mateo, CA, USA.
- Reeves, C.R. and Wright, C.C. (1995b). Epistasis in genetic algorithms: An experimental design perspective. In: *Proceedings of the 6th international conference on genetic algorithms*, edited by Eshelman, L.J., 217-224, Morgan Kaufmann, San Mateo, CA, USA.

- Reeves, C.R. and Wright, C.C. (1999). Genetic algorithms and the design of experiments. In: *Evolutionary algorithms*, edited by Davis, L.D., Jong, K.D., Vose, M.D. and Whitley, L.D., Springer, New York, USA.
- Reklaitis, G.V., Ravindran, A. and Ragsdell, K.M. (1983). *Engineering optimization methods and applications*. Wiley, New York, USA.
- Richards, W. (1988). *Natural computation*. MIT Press, Cambridge, MA, USA.
- Richardson, J.T., Palmer, M.R., Liepins, G. and Hilliard, M. (1989). Some guidelines for genetic algorithms with penalty functions. In: *Proceedings of the third international conference on genetic algorithms*, 191-197, Morgan Kaufmann Publishers, USA.
- Rogero, J.M., Tiwari, A., Munaux, O., Rubini, P.A., Roy, R. and Jared, G. (2000). Applications of evolutionary algorithms for solving real-life design optimisation problems. In: *6th International Conference on Parallel Problem Solving from Nature (PPSN-VI) Workshop on Real-life Evolutionary Design Optimisation*, Paris, France, September 16 2000.
- Rosenberg, R.S. (1967). *Simulation of genetic populations with biochemical properties*. PhD. Dissertation, University of Michigan, Michigan, USA.
- Rosenbrock, H.H. (1960), An automatic method for finding the greatest or least value of a function, *Computational journal*, 3, 175-184.
- Roy, R. (1997). *Adaptive search and the preliminary design of gas turbine blade cooling system*. PhD. Thesis, Engineering Design Centre, University of Plymouth, Plymouth, UK.
- Roy, R., Jared, G., Grau, J. and Buchannan, T. (1998). Adaptive design decision support within CAD environment. In: *Design reuse*, edited by Sivalogonathan, S. and Shahin, T.M.M. Professional engineering publishing, UK.
- Roy, R., Jared, G., Tiwari, A. and Munaux, O. (2000a). *FLEXO: Project scope definition*. FLEXO Report – 1, School of Industrial and Manufacturing Science (SIMS), Cranfield University (UK). (unpublished).

- Roy, R., Jared, G., Tiwari, A. and Munaux, O. (2000b). *An overview of evolutionary-based multi-objective, multi-modal and constrained optimisation techniques*. FLEXO Report – 2, School of Industrial and Manufacturing Science (SIMS), Cranfield University (UK). (unpublished).
- Roy, R., Jared, G., Tiwari, A. and Munaux, O. (2000c). *Industrial requirements for flexible optimisation*. FLEXO Report – 4, School of Industrial and Manufacturing Science (SIMS), Cranfield University (UK). (unpublished).
- Roy, R., Jared, G., Tiwari, A. and Munaux, O. (2000d). *Design optimisation: A survey of industries*. FLEXO Report – 5, School of Industrial and Manufacturing Science (SIMS), Cranfield University (UK). (unpublished).
- Roy, R., Parmee, I.C. and Purchase, G. (1996). Adaptive search manager: An engineering design decision support approach. *Applications of artificial intelligence for technological and business processes*, 2, 46-49.
- Roy, R., Tiwari, A. and Braneby, A. (2001). Making evolutionary design optimisation popular in industry: Issues and techniques. In: *6th Online World Conference on Soft Computing in Industrial Applications (WSC-6)*, Cyberspace, September 10-28 2001.
- Roy, R., Tiwari, A., Munaux, O. and Jared, G. (2000). Real-life engineering design optimisation: Features and techniques. In: *CDROM Proceedings of the 5th Online World Conference on Soft Computing in Industrial Applications (WSC-5)*, edited by Martikainen, J. and Tanskanen, J. ISBN 951-22-5205-8, IEEE, Finland.
- Rudolph, G. (2001). Evolutionary search under partially ordered fitness sets. In: *Proceedings of the international NAISO congress on information science innovations (ISI 2001)*, 818-822, ICSC Academic Press, Millet/Sliedrecht, Germany.
- Rudolph, S. and Koppen, M. (1996). Stochastic hill climbing by vectors of normal distributions, In: *Proceedings of the first online conference on soft computing (WSC1)*, Nagoya, Japan.

- SDRC. (2001). *Citing Internet resources* (WWW document). <http://www.sdrc.com>. (accessed 3rd November 2001).
- Sandgren, E. (1994). Multicriteria design optimization by goal programming. In: *Advances in Design Optimization*, edited by Adeli, H., 225-265, Chapman & Hall, London, UK.
- Schaffer, J.D. (1984). *Some experiments in machine learning using vector evaluated genetic algorithms*. Doctoral Dissertation, Vanderbilt University, Nashville, Tennessee, USA.
- Schaffer, J.D. (1985). Multiple objective optimization with vector evaluated genetic algorithm. In: *Genetic algorithms and their applications: Proceedings of the first international conference on genetic algorithms*, 93-100, Lawrence Erlbaum, USA.
- Schaffer, J.D. and Morishima, A. (1987). An adaptive crossover distribution mechanism for genetic algorithms. In: *Proceedings of the second international conference on genetic algorithms*, 36-40.
- Schoenauer, M. and Xanthakis, S. (1993). Constrained GA optimization. In: *Proceedings of the fifth international conference on genetic algorithms*, 573-580, Morgan Kaufmann Publishers, USA.
- Schwefel, H.P. (1995). *Evolution and optimum seeking*. John Wiley & Sons, Inc., New York, USA.
- Scott, D.W. (1992). *Multivariate density estimation: Theory, practice and visualisation*, Wiley, USA.
- Sebag, M. and Ducoulombier, A. (1998). Extending population-based incremental learning to continuous search spaces. In: *Parallel Problem Solving from Nature V (PPSN-V)*, edited by Eiben, A.E., Back, T., Schoenauer, M. and Schwefel, H.P., 418-427, Springer, Berlin, Germany.
- Sen, P. and Yang, J.-B. (1998). *Multiple criteria decision support in engineering design*. Springer, London, UK.

- Servet, I., Trave-Massuyes, L. and Stern, D. (1997). Telephone network traffic overloading diagnosis and evolutionary techniques. In: *Proceedings of the third european conference on artificial evolution (AE'97)*, 137-144.
- Smith, J. and Fogarty, T.C. (1996). Recombination strategy adaptation via evolution of gene linkage. In: *Proceedings of the 1996 IEEE international conference on evolutionary computation*, 826-831, IEEE, USA.
- Smith, N.H., and Rudd, D.F. (1964). *The feasibility of directed random search*, Technical Report, Department of Chemical Engineering, University of Wisconsin, USA.
- Soto, M., Ochoa, A., Acid, S. and De Campos, L.M. (1999). Introducing the polytree approximation of distribution algorithm. In: *Second symposium on artificial intelligence and adaptive systems (CIMAFA'99)*, 360-367.
- Spendley, W., Hext, G.R. and Himsworth, F.R. (1962). Sequential application of simplex designs in optimization and evolutionary operation, *Technometrics*, 4, 441.
- Srinivas, N. and Deb, K. (1994). Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary computation*, 2(3), 221-248.
- Stanley, T.J. and Mudge, T. (1995). A parallel genetic algorithm for multiobjective microprocessor design. In: *Proceedings of the sixth international conference on genetic algorithms*, 597-604.
- Steuer, R.E. (1986). *Multiple criteria optimization: Theory, computation, and application*. Wiley, New York, USA.
- Sundaresan, S., Ishii, K. and Houser, D. (1993). A robust optimisation procedure with variations on design variables and constraints. In: *Advances in design automation*, Vol. 1, DE-Vol. 65-1, ASME, USA.
- Syswerda, G. and Palmucci, J. (1991). The application of genetic algorithms to resource scheduling. In: *Proceedings of the fourth international conference on*

- genetic algorithms*. edited by Belew, R.K. and Booker, L.B., 502-508, Morgan Kaufmann Publishers, USA.
- Taguchi, G. (1987a). *System of experimental design*, Vol. 1, UNIPUB/Kraus International Publications, New York, USA.
- Taguchi, G. (1987b). *System of experimental design*, Vol. 2, UNIPUB/Kraus International Publications, New York, USA.
- Tamaki, H., Kita, H. and Kobayashi, S. (1996). Multi-objective optimization by genetic algorithms: A review. In: *Proceedings of the 1996 international conference on evolutionary computation (ICEC96)*, edited by Fukuda, T. and Furuhashi, T., IEEE, Japan.
- Tan, K.C. and Li, Y. (1997). *Multi-objective genetic algorithm based time and frequency domain design unification of linear control systems*. Technical Report No. CSC-97007, Department of Electronics and Electrical Engineering, University of Glasgow, Scotland.
- Tanaka, M. (1995). GA-based decision support system for multi-criteria optimization. In: *Proceedings of the international conference on systems, man and cybernetics*, Vol. 2, 1556-1561.
- Thierens, D. (1995). *Analysis and design of genetic algorithms*. Research Report, Katholieke Universiteit, Leuven, Belgium. (unpublished).
- Thierens, D. and Bosman, P.A.N. (2001). Multi-objective mixture-based iterated density estimation evolutionary algorithms. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, edited by Spector, L., Goodman, E., Wu, A., Langdon, W.B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. and Burke, E. 671-678, Morgan Kaufmann Publishers, San Francisco, USA.
- Tilley, D.G. and Burrows, C.R. (1995). Development of computer-based techniques for fluid power systems design. *Design studies*, 16(4), 397-414.

- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001a). Interaction and multi-objective optimisation. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, edited by Spector, L., Goodman, E., Wu, A., Langdon, W.B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. and Burke, E. 671-678, Morgan Kaufmann Publishers, San Francisco, USA.
- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001b). Challenges in real-life engineering design optimisation: An analysis. In: *Genetic and Evolutionary Computation Conference (GECCO-2001) Workshop on Real-life Evolutionary Design Optimisation*, San Francisco, USA, July 7, 289-294.
- Tiwari, A., Roy, R., Jared, G. and Munaux, O. (2001c). Evolutionary-based techniques for real-life optimisation: Development and testing. Accepted for publication: *Applied Soft Computing (ASC) Journal*, Elsevier Science (Netherlands).
- Valenzuela-Rendon, M. and Uresti-Charre, E. (1997). A non-generational genetic algorithm for multiobjective optimization. In: *Proceedings of the seventh international conference on genetic algorithms*. edited by Back, T., 658-665, Morgan Kauffman Publishers, USA.
- Veldhuizen, D.A.V. (1999). *Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations*. PhD. Thesis, Air Force Institute of Technology, Dayton, USA.
- Veldhuizen, D.A.V. and Lamont, G.B. (1998). *Multiobjective evolutionary algorithm research: A history and analysis*. Technical Report No. TR-98-03, Air Force Institute of Technology, Wright-Patterson AFB, USA. (unpublished).
- Viennet, R. (1996). Multicriteria optimization using a genetic algorithm for determining the Pareto set. *International journal of systems science*, 27(2), 255-260.
- Waagen, D., Diercks, P. and McDonnell, J. (1992). The stochastic direction set algorithm: A hybrid technique for finding function extrema. In: *Proceedings of*

- the first annual conference on evolutionary programming*, 35-42, Evolutionary Programming Society, USA.
- Wang, L. and Grandhi, R.V. (1996). Safety index calculations using intervening variables for structural reliability analysis. *Computers & structures*, 59(6), 1139-1148.
- Weile, D.S. and Michielssen, E. and Goldberg, D.E. (1996). Genetic algorithm design of Pareto-optimal broad band microwave absorbers. *IEEE transactions on electromagnetic compatibility*, 38(3), 518-525.
- Wienke, P.B., Lucasius, C. and Kateman, G. (1992). Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta*, 265(2), 211-225.
- Wilson, P.B. and Macleod, M.D. (1993). Low implementation cost IIR digital filter design using genetic algorithms. In: *IEE/IEEE workshop on natural Algorithms in signal processing*, 4/1-4/8, Chelmsford, UK.
- Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1), 67-82.
- Yang, J.B. and Sen, P. (1994). Multiple objective design optimization by estimating local utility functions. *Advances in design automation*, 2, 135-145.
- Zitzler, E., Deb, K. and Thiele, L. (1999). *Comparison of multiobjective evolutionary algorithms: Empirical results*. Technical Report No. TIK-70, Institut für Technische Informatik and Kommunikationsnetze, ETH Zurich, Switzerland. (unpublished).
- Zitzler, E., Deb, K. and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation journal*, 8(2), 125-148.
- Zitzler, E. and Thiele, L. (1998a). Multiobjective optimization using evolutionary algorithms – A comparative case study. In: *Parallel Problem Solving from*

Nature V (PPSN-V), edited by Eiben, A.E., Back, T., Schoenauer, M. and Schwefel, H.P., 292-301, Springer, Berlin, Germany.

Zitzler, E. and Thiele, L. (1998b). *An evolutionary algorithm for multiobjective optimization: The strength Pareto approach*. Technical Report No. 43, Computer Engineering and Networks Laboratory, Zurich, Switzerland.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE transactions on evolutionary computation*, 3(4), 257-271.

A. APPENDIX: 'FLEXO' QUESTIONNAIRE

The questions that form part of the 'FLEXO' questionnaire are given below. The explanations that are provided to the designers for each of the modules are also presented here.

Module 1: General Issues

This section of the questionnaire tries to understand the general design practice involved in your company and your degree of involvement in those activities.

Q. 1: What industry sector is your company involved in?

Q. 2: In terms of the following criteria, please describe the product your company designs.

- ◆ Name
- ◆ Usage
- ◆ Material
- ◆ Size
- ◆ Complexity
- ◆ Performance/value
- ◆ Single component/assembly

Q. 3: How much time (in days) do you spend on design related activities per week?

Q. 4: Explain the stages of design cycle in your company, and your role and place in the design cycle.

Q. 5: Would it be possible to categorise the general nature of your design activities?

- ◆ Detailed design
- ◆ Preliminary design
- ◆ Creative or innovative design
- ◆ Design analysis
- ◆ Design evaluation
- ◆ Design activity management and co-ordination
- ◆ Developing tools that can be useful in design activities
- ◆ no, it cannot be categorised because:
- ◆ others, please specify:

Q. 6: Please describe the nature of data received by you.

- ◆ Origin
- ◆ Type/format/nature
- ◆ Quality
- ◆ Frequency/work load

Q. 7: Please describe the nature of data delivered by you.

- ◆ Destination
- ◆ Type/format/nature
- ◆ Quality
- ◆ Value added

Q. 8: What are the different tools you use for your day to day design activities? (Please encircle more than one answer if you wish and specify the tool used)

- ◆ Drafting board and pencil
- ◆ Pen and pencil for calculations and free hand drawings
- ◆ Digitiser
- ◆ Computer aided drafting package
- ◆ Computer aided design and analysis package
- ◆ Simple spreadsheet for calculations
- ◆ Project management software
- ◆ Others, please specify:

Q. 9: In the design process, which tool do you use most? (Please encircle one of the following)

- ◆ Surface modelling
- ◆ Solid modelling
- ◆ Drafting tool

Q. 10: Which technique do you use to improve the design? (Encircle more than one answer if you wish)

- ◆ Design for assembly
- ◆ Design for manufacture
- ◆ Design performance
- ◆ Design for quality
- ◆ Shape optimisation

Q. 11: If you are working on shape optimisation, what are the design criteria?

- ◆ Structural properties

- *Weight*
- *Stress*
- *Vibration*
- *Thermodynamics*
- ◆ **Surface properties**
 - *Aerodynamics*
 - *Fluid dynamics*
 - *Aesthetics*
 - *Acoustics*
 - *Manufacturability*

Q. 12: What are the constraints considered for the design?

Q. 13: How would you improve the design you are working on?

- ◆ Trial and error
- ◆ Many iterations of educated guesses using previous knowledge
- ◆ Using an optimisation algorithm
- ◆ Using an optimisation package available in computer integrated design tools
- ◆ Others, please specify:

Q. 14: If you are using an optimisation algorithm to improve your design, please specify the optimisation technique and the optimisation approach that you employ.

- ◆ **Optimisation approach**
 - *Single criterion*
 - *Multiple criteria*
 - *Single optimal solution*
 - *Multiple optimal solutions*
 - *Single variable*

- *Multiple variables*
- ◆ Optimisation technique
 - *Conventional optimisation algorithm*
 - *Evolutionary-based optimisation algorithm*

Q. 15: How do you ensure that your design criteria and constraints are satisfied?

Q. 16: Which optimisation scenario do you use?

- ◆ Manual
- ◆ Off-line, independent of CAD/CAM system
- ◆ On-line, fully integrated with the CAD/CAM system

Q. 17: Relative to the total design cycle, how much time do you spend refining the design?

- ◆ Below 25%
- ◆ 25 to 50%
- ◆ 50 to 75%
- ◆ Above 75%

Q. 18: Please explain your interaction with other designers in the company?

Q. 19: If you have working experience with other companies, please state the difference of working methodology.

Module 2: Industrial Requirements Capture

The aim of this module is to capture the industrial requirements on the flexible optimisation environment. This is achieved by analysing the limitations of existing CAD/CAM systems and the characteristics expected in the flexible optimisation framework. The expected improvements from the framework are also discussed. Please answer the following questions based on your design experience in CAD/CAM.

Q. 1: What are the drawbacks and limitations of the current design process?

Q. 2: Could you please describe the limitations of the CAD/CAM system you use in terms of its optimisation capability?

Q. 3: Which tasks in the design process are not essential and can be avoided?

Q. 4: What are the critical activities in terms of time and skill involved?

Q. 5: Which of the tasks are repetitive in nature?

Q. 6: Are existing designs reused?

Q. 7: How do you feel the design process can be made more efficient?

Q. 8: Which tasks in the design cycle can be 'automated'?

Q. 9: How do you convert design requirements to design parameters?

Q. 10: How do you compare the final design with the initial requirements?

Q. 11: How integrated flexible optimisation could help in the design process?

Q. 12: How can the 'FLEXO' project help you in your job?

Module 3: General Remarks

Please write any general remarks you wish to make, and mention if you have any other suggestion(s).

Module 4: Self-assessment of the Users

This section of the questionnaire is optional. The sole purpose of this module is to gather some information about you so that your comments may be evaluated in the right perspective. In case you do not feel comfortable in answering any part of this module, please ignore it. If you are happy to answer a question, please tick the appropriate box.

Q. 1: Engineering design:

	Best				Worst
Knowledge	1	2	3	4	5
Experience	1	2	3	4	5

Q. 2: Use of CAD/CAM systems:

	Best				Worst
Knowledge	1	2	3	4	5
Experience	1	2	3	4	5

Q. 3: Optimisation Algorithms:

	Best				Worst
Knowledge	1	2	3	4	5
Experience	1	2	3	4	5

B. APPENDIX: DESCRIPTION OF NSGA-II

This appendix provides a brief description of Non-dominated Sorting Genetic Algorithm – II (NSGA-II), which is a high-performing, novel multi-objective optimisation algorithm demonstrating better performance than most other contemporary algorithms (Deb *et al.*, 2000).

NSGA-II Algorithm

Multi-objective evolutionary algorithms that use non-dominated sorting and sharing have been criticised for their $O(MN^3)$ computational complexity (where M is the number of objectives and N is the population size), non-elitism approach, and the need for specifying a sharing parameter. NSGA-II is a non-dominated sorting based multi-objective evolutionary algorithm that alleviates all the above three difficulties. It uses a fast non-dominated sorting approach with $O(MN^2)$ computational complexity, an elitist approach and a parameter-less sharing approach to tackle all the above-mentioned difficulties. The steps involved in this algorithm are as follows (Figure B.1).

1. Create a random parent population of size N .
2. Sort the population based on non-domination, and to each solution assign a fitness value equal to its non-domination level.
3. Create a child population of size N using binary tournament selection, recombination and mutation operators.
4. Combine the parent and child populations to create a global population of size $2N$.
5. Sort the global population based on non-domination.
6. Create a new parent population by selecting solutions in order of their fronts until the number of selected solutions exceeds N .
7. Sort the solutions of the last accepted front using niched comparison operator.
8. Using this sorting, select solutions from the last front until the size of new parent population becomes N .

9. If the number of generations has exceeded a pre-determined value, say 100, stop the process else go to Step 3.
10. Display the final solutions.

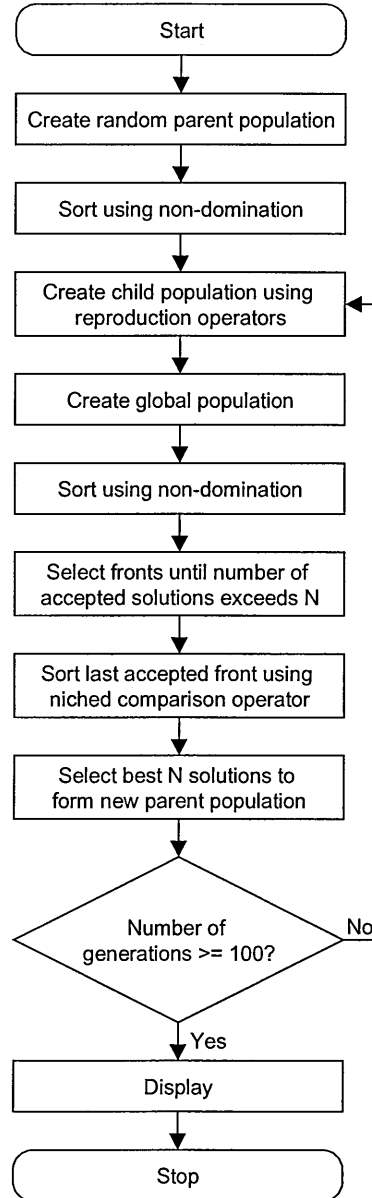


Figure B.1: Non-dominated Sorting GA –II (NSGA-II)

Analysis of NSGA-II

As stated earlier, there are primarily two goals that a multi-objective optimisation technique must achieve. These include guiding the search towards global Pareto-optimal region and maintaining population diversity in the Pareto front. This section

analyses the internal mechanisms provided in NSGA-II for dealing with each of these goals.

Convergence to Global Pareto Front

In NSGA-II, the non-domination level of a solution has a strong impact on its fitness. The fitness assignment is usually performed in such a way that the lowest fitness at a particular non-domination level has a higher value than the highest fitness at the immediate lower level. This ensures that the search is driven towards the global Pareto front. Further, NSGA-II uses an elitist approach through a selection operator that creates a mating pool by combining the parent and child populations, and selecting the best (with respect to fitness and spread) N solutions. This elitism ensures that the ‘good’ solutions of the population are not lost, thereby creating a selection pressure towards the global Pareto front.

The above-mentioned strategy used by NSGA-II works well for a number of multi-objective optimisation problems. However, complex problems having high degrees of inseparable function interaction usually possess multiple local fronts, along with deception. The fitness assignment strategy of NSGA-II ceases to produce the driving force towards the global front once most of the solutions of the population share the same non-domination level. This tendency is further augmented in NSGA-II due to the use of elitism. Therefore, NSGA-II suffers from the tendency of getting trapped in local fronts (pre-mature convergence).

Maintenance of Diverse Pareto-optimal Solutions

The diversity among non-dominated solutions is introduced in NSGA-II by using the crowded comparison operator that is used in the tournament selection and during the population reduction phase. The crowded comparison operator states that between two solutions with different non-domination ranks, the point with the lower rank is preferred. Otherwise, if both the points belong to the same front then the point that is located in a region with lesser number of points is preferred. In this way, the crowded comparison operator guides the selection process at various stages of the algorithm towards a uniformly spread-out Pareto front. Further, no extra niching parameter

(such as σ_{share} in NSGA (Deb, 1999a)) is required here, since solutions compete with their crowding distance (a measure of density of solutions in the neighbourhood). Finally, the use of elitism ensures that the diverse Pareto-optimal solutions, identified so far, are not lost.

As in the case of convergence, this strategy works well for a number of problems. However, complex inseparable function interaction in a problem may lead to one or more of the following features.

- ◆ Discontinuity in Pareto front.
- ◆ Biased Pareto front.
- ◆ Complex relationships among decision variables of Pareto-optimal solutions.

NSGA-II suffers from serious limitations in handling these problems. This is because the strategy of NSGA-II attempts to attain solution diversity using external means, without addressing the inherent problem features that lead to diversity issues.

Computational Expense of NSGA-II

The computational complexity of non-dominated sorting algorithms in use until now is $O(MN^3)$. However, NSGA-II uses a fast non-dominated sorting approach that requires at most $O(MN^2)$ computations. The two approaches are similar in principle, except that a better book-keeping strategy makes NSGA-II a faster algorithm. In this approach, every solution from the population is checked with a partially filled population for domination. The steps involved in this approach are listed below.

1. Include first member in a set P' .
2. Take one solution (p) at a time.
3. Include p in P' temporarily.
4. Compare p with other members of P' .
5. If p dominates a member of P' , delete it.
6. If p is dominated by other members of P' , do not include p in P' .

When all solutions of the population are checked, the remaining members of P' constitute the non-dominated set. To find other fronts, the members of P' are discounted and the above procedure is repeated. In this algorithm, the second

element of the population is compared with only one solution of P' , the third solution with at most two solutions of P' , and so on. This requires a maximum of $O(N^2)$ domination checks. Since each domination check requires M function value comparisons, the maximum complexity of this approach is $O(MN^2)$.

Let us now look at the complexity of one iteration of the entire algorithm, considering the worst case complexities of basic operations.

1. Non-dominated sort: $O(MN^2)$.
2. Crowding distance assignment: $O(MN\log N)$.
3. Sort using crowded comparison operator: $O(2N\log(2N))$.

As can be seen, the overall complexity of the above algorithm is $O(MN^2)$.

This appendix has provided a brief description and analysis of NSGA-II, together with its computational expense.

C. APPENDIX: PERFORMANCE METRICS FOR MULTI-OBJECTIVE OPTIMISATION

This appendix briefly describes the various performance measures that can be used for multi-objective optimisation. Unlike in single objective optimisation, there are two goals in a multi-objective optimisation – (i) convergence to the Pareto-optimal set, and (ii) maintenance of diversity in solutions of the Pareto-optimal set. Clearly, these two tasks cannot be measured with one performance metric adequately. A number of performance metrics have been suggested in the past (Fonseca and Fleming, 1995; Zitzler *et al.*, 1999). But here two performance metrics have been used (Deb *et al.*, 2000). These are more direct in evaluating each of the above two goals in a solution set obtained by a multi-objective optimisation algorithm.

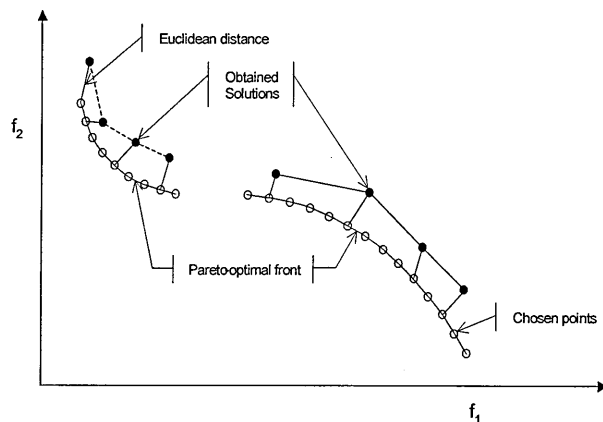


Figure C.1: Illustration of Distance Metric γ

The first metric γ measures the extent of convergence to a known set of Pareto-optimal solutions. The calculation of this metric is possible only when the Pareto-optimal set is known. Hence, this metric cannot be used for any arbitrary problem. First, a set of H equal to 500 uniformly-spaced solutions are chosen from the true Pareto front in the objective space. For each solution obtained with the algorithm, the minimum Euclidean distance from H chosen solutions on the Pareto front is computed. The average of these distances is used as the first metric γ (the convergence metric). Figure C.1 shows the calculation procedure of this metric.

Solutions with open circles are H chosen solutions on the Pareto front for the calculation of the convergence metric and solutions marked with dark circles are solutions obtained by the algorithm. It is clear that the smaller the value of this metric, the better is the convergence towards the Pareto front. When all obtained solutions lie exactly on H chosen solutions, this metric takes a value zero.

Even when all the solutions converge to the Pareto front, the above convergence metric does not have a value zero. The metric will be zero only when each obtained solution lies exactly on each of the chosen solutions. Although this metric alone can provide some information about the spread in obtained solutions, a different metric to measure the spread in solutions obtained by the algorithm is discussed below. The second metric Δ measures the extent of spread achieved among the obtained solutions. Here, the idea is to get a set of solutions that span the entire Pareto-optimal region. Firstly, the Euclidean distance d_i between consecutive solutions in the obtained non-dominated set of solutions is calculated. Then the average μ of these distances is calculated. Thereafter, from the obtained set of non-dominated solutions, the extreme solutions (in the objective space) are determined. Then, the following metric is used to calculate the non-uniformity in the distribution (Equation C.1).

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \mu|}{d_f + d_l + (N-1)\mu} \tag{Equation C.1}$$

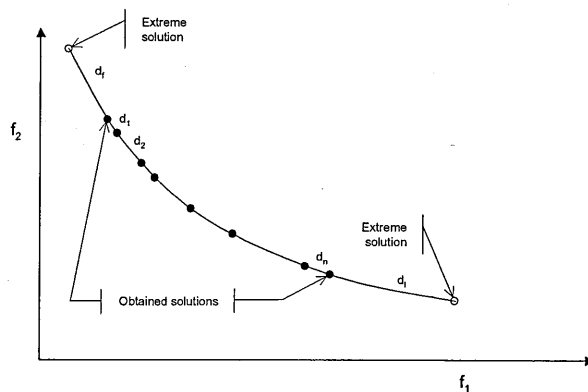


Figure C.2: Illustration of Diversity Metric Δ

Here, the parameters d_f and d_i are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set, as depicted in Figure C.2 that illustrates all distances mentioned in the above equation. The parameter μ is the average of all distances d_i , $i = 1, 2, \dots, (N-1)$, assuming that there are N solutions on the best non-dominated front. With N solutions, there are $(N-1)$ consecutive distances. The denominator is the value of the numerator for the case when all N solutions lie on one solution. It is interesting to note that this is not the worst case spread of solutions possible since a scenario with large variance of the distances may have a numerator value greater than the denominator. Thus, the maximum value of the above metric can be greater than one. But a good distribution would make all distances d_i 's equal to μ and would make $d_f = d_i = 0$ (with existence of extreme solutions in the non-dominated set). Thus, for the most widely and uniformly spread-out set of non-dominated solutions, the numerator of Δ would be zero, making the metric to take a value zero. For any other distribution, the value of the metric would be greater than zero. For two distributions having identical values of d_f and d_i , the metric Δ takes a higher value with worse distributions of solutions within the extreme solutions. Note that the above diversity metric can be used on any non-dominated set of solutions, including one which is not the Pareto-optimal set.

This appendix has provided a brief description of two performance metrics for multi-objective optimisation: γ (convergence metric) and Δ (diversity metric).