

Multi-objective Metaheuristic Approaches for Data Clustering in Engineering Application(s)

A Thesis

submitted in fulfillment of the requirement
for the award of degree of
Doctor of Philosophy

Submitted By

Gaurav Dhiman

(Regn. No.: 901503006)

Under the guidance of

Dr. Vijay Kumar

(Assistant Professor, Computer Science and Engineering Department)

Thapar Institute of Engineering & Technology, Patiala



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY
PATIALA - 147004, PUNJAB, INDIA

September, 2019



I would like to dedicate this thesis to my loving parents, sister, teachers, friends, and god, who make me whole.

“Education is the most powerful weapon which you can use to change the world.”

By Nelson Mandela

Contents

List of Figures	x
List of Tables	xiii
List of Algorithms	xix
Certificate	xx
Acknowledgement	xxi
Glossary of Abbreviations/Symbols	xxii
Abstract	xxv
1 Introduction	1
1.1 Clustering	1
1.1.1 Definitions	2
1.1.2 Clustering Procedure	3
1.1.3 Distance Measures	4
1.1.3.1 Euclidean Distance	5
1.1.3.2 Standardized Euclidean Distance	5
1.1.3.3 Cosine Distance	5
1.1.3.4 Spearman Distance	6
1.1.3.5 Chebyshev Distance	6
1.1.4 Validity Indices	6
1.1.4.1 Dunn Index (DI Index)	7
1.1.4.2 Davis-Bouldin Index (DB Index)	7

1.1.4.3	Xie and Beni Index (XB Index)	8
1.1.4.4	Pakhira-Bandyopadhyay-Maulik Index (PBM Index)	8
1.1.4.5	<i>I</i> Index	9
1.1.4.6	<i>Sym</i> Index	9
1.2	Metaheuristic Techniques	10
1.2.1	Classification of Metaheuristic Techniques	11
1.2.1.1	Single-objective Optimization	13
1.2.1.2	Multi-objective Optimization	14
1.2.2	Performance Metrics	15
1.3	Thesis Layout	17
2	Literature Review	21
2.1	Review on Multi-objective based Clustering Techniques	21
2.2	Review on Multi-objective based Feature Selection Techniques	27
2.3	Review on Multi-objective based Clustering and Feature Selection Techniques	29
2.4	Research Gaps	33
2.5	Objectives	34
2.6	Research Methodology	34
3	Bio-inspired based Metaheuristic Optimization Techniques	37
3.1	Introduction	37
3.2	Single-objective Metaheuristic Algorithms	38
3.2.1	Spotted Hyena Optimizer (SHO)	38
3.2.1.1	Mathematical Modeling	39
3.2.1.1.1	Encircling Prey	39
3.2.1.1.2	Hunting	40
3.2.1.1.3	Attacking Prey (<i>Exploitation</i>)	41
3.2.1.1.4	Search for Prey(<i>Exploration</i>)	42
3.2.1.2	Steps of SHO Algorithm	45

3.2.1.3	Computational Complexity	46
3.2.1.3.1	Time Complexity	46
3.2.1.3.2	Space Complexity	47
3.2.1.4	Experimental Results and Discussions	47
3.2.1.4.1	Benchmark Test Functions and Algorithms used for Comparisons	48
3.2.1.4.2	Experimental Setup	49
3.2.1.4.3	Performance Comparison	49
3.2.1.4.4	Convergence Analysis	50
3.2.1.4.5	Scalability Study	51
3.2.1.4.6	Statistical Testing	51
3.2.1.5	Applications of SHO on Engineering Design Problems	59
3.2.1.5.1	Constraint Handling	59
3.2.1.5.2	Welded Beam Design Problem	60
3.2.1.5.3	Tension/compression Spring Design Problem	61
3.2.1.5.4	Pressure Vessel Design Problem	63
3.2.1.5.5	Speed Reducer Design Problem	64
3.2.1.5.6	Rolling Element Bearing Design Problem .	66
3.2.1.5.7	Displacement of Loaded Structure Problem	68
3.2.2	Emperor Penguin Optimizer (EPO)	69
3.2.2.1	Mathematical Model and Optimization Algorithm .	70
3.2.2.1.1	Generate and Determine the Huddle Boundary	71
3.2.2.1.2	Temperature Profile Around the Huddle .	71
3.2.2.1.3	Distance between Emperor Penguins	72
3.2.2.1.4	Relocate the Mover	73
3.2.2.2	Steps and Flowchart of Proposed EPO Algorithm .	76
3.2.2.3	Computational Complexity	79

3.2.2.3.1	Time Complexity	79
3.2.2.3.2	Space Complexity	79
3.2.2.4	Experimental Results and Discussions	80
3.2.2.4.1	Benchmark Test Functions	80
3.2.2.4.2	Algorithms Involved for Comparison	80
3.2.2.4.3	Experimental Setup	80
3.2.2.4.4	Performance Comparison	81
3.2.2.5	Convergence Analysis	82
3.2.2.6	Sensitivity Analysis	87
3.2.2.7	Scalability Study	91
3.2.2.8	Statistical Testing	91
3.2.2.9	Applications of EPO on Engineering Design Problems	95
3.2.2.9.1	Pressure Vessel Design Problem	95
3.2.2.9.2	Speed Reducer Design Problem	96
3.2.2.9.3	Welded Beam Design Problem	97
3.2.2.9.4	Tension/compression Spring Design Problem	98
3.2.2.9.5	Rolling Element Bearing Design Problem	99
3.2.2.9.6	Displacement of Loaded Structure Problem	99
3.3	Multi-objective Metaheuristic Technique	101
3.3.1	Multi-objective Spotted Hyena Optimizer	101
3.3.1.1	Archive	101
3.3.1.1.1	The Archive Controller	102
3.3.1.1.2	The Grid	102
3.3.1.1.3	Group Selection Mechanism	103
3.3.1.2	Computational Complexity	106
3.3.1.2.1	Time Complexity	106
3.3.1.2.2	Space Complexity	107
3.3.1.3	Experimental Results and Discussions	107

3.3.1.3.1	Benchmark Test Functions	107
3.3.1.3.2	Experimental Setup	109
3.3.1.3.3	Performance Metrics	109
3.3.1.3.4	Performance Evaluation	110
3.3.1.3.5	Sensitivity Analysis	121
3.3.1.4	Applications of MOSHO on Engineering Design Problems	124
3.3.1.4.1	Welded Beam Design Problem	124
3.3.1.4.2	Multiple-disk Clutch Brake Design Problem	125
3.3.1.4.3	Pressure Vessel Design Problem	125
3.3.1.4.4	Speed Reducer Design Problem	126
3.4	Summary	127
4	Automatic Clustering using Multi-objective Spotted Hyena Optimizer	129
4.1	Introduction	129
4.2	Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO)	130
4.2.1	Inspiration	130
4.2.2	Proposed Clustering Technique	131
4.2.2.1	Cluster Encoding	131
4.2.3	Threshold Setting Computation	132
4.2.4	Threshold Cutoff Computation	133
4.2.4.1	Fitness Evaluation	133
4.2.4.2	Cluster Center Validation	133
4.2.5	Computational Complexity	134
4.2.5.1	Time Complexity	134
4.2.5.2	Space Complexity	134
4.3	Experimental Results and Discussions	134
4.3.1	Data sets and Algorithms used for Comparisons	135
4.3.2	Performance Comparison	135

4.4	Application to Image Segmentation	136
4.5	Summary	139
5	Automatic Feature Selection using Multi-objective Spotted Hyena	
	Optimizer	141
5.1	Introduction	141
5.2	Proposed Feature Selection Technique	142
5.2.1	Agent Representation and Initialization	143
5.2.2	Threshold Setting Computation	144
5.2.3	Threshold Cutoff Computation	145
5.2.4	Fitness Function Evaluation	145
5.2.5	Cluster Center Validation	146
5.3	Computational Complexity	146
5.3.1	Time Complexity	146
5.3.2	Space Complexity	147
5.4	Experimental Results and Discussions	147
5.4.1	Data sets used for Comparisons	147
5.4.2	Experimental Setup	147
5.4.3	Results and Discussions	148
5.4.4	Relevance of Selected Features	150
5.4.5	Performance Evaluation	150
5.5	Summary	152
6	Simultaneously Clustering and Feature Selection using	
	Multi-objective Spotted Hyena Optimizer	153
6.1	Introduction	153
6.2	Proposed Automatic Clustering and Feature Selection Approach	
	using Multi-objective Spotted Hyena Optimizer	154
6.2.1	Contribution	154
6.2.2	Astrophysics Concept	154

6.2.3	Astrophysics based Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)	155
6.2.4	Automatic Clustering and Feature Selection using Astrophysics based Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)	156
6.2.4.1	Agent Representation and Initialization	156
6.2.4.2	Threshold Setting Computation	158
6.2.4.3	Threshold Cutoff Computation	159
6.2.4.4	Active Cluster Center and Feature Extraction	160
6.2.4.5	Cluster Center Validation	161
6.2.4.6	Fitness Function Evaluation	162
6.2.5	Computational Complexity	162
6.2.5.1	Time Complexity	162
6.2.5.2	Space Complexity	163
6.3	Experimental Results and Discussions	163
6.3.1	Data sets used	163
6.3.2	Experimental Setup	164
6.3.3	Relevance of Selected Features	164
6.3.4	Relevance of Selected Clusters	165
6.3.5	Performance Evaluation	165
6.3.6	Statistical Testing	167
6.4	AMOSHO_CFS for Real-life Applications	168
6.4.1	Application to Image Segmentation	168
6.4.2	Microarray Data Analysis	168
6.5	Summary	173
7	Conclusions and Future Scope	175
Appendix A	Single-objective Benchmark Test Functions	179
A.1	Unimodal Benchmark Test Functions	179

A.2	Multimodal Benchmark Test Functions	179
A.3	Fixed-dimension Multimodal Benchmark Test Functions	180
A.4	Composite Benchmark Test Functions	180
A.5	IEEE CEC-2015 Benchmark Test Functions	180
Appendix B	Multi-objective Benchmark Test Functions	184
B.1	ZDT Benchmark Test Functions	184
B.1.1	ZDT1	184
B.1.2	ZDT2	184
B.1.3	ZDT3	185
B.1.4	ZDT4	185
B.1.5	ZDT6	185
B.2	DTLZ Benchmark Test Functions	186
B.2.1	DTLZ1	186
B.2.2	DTLZ2	186
B.2.3	DTLZ3	186
B.2.4	DTLZ4	186
B.2.5	DTLZ5	187
B.2.6	DTLZ6	187
B.2.7	DTLZ7	187
B.3	IEEE CEC-2009 Benchmark Test Functions	188
Appendix C	Engineering Design Problems	190
C.1	Single-objective Engineering Design Problems	190
C.1.1	Pressure Vessel Design Problem	190
C.1.2	Speed Reducer Design Problem	191
C.1.3	Welded Beam Design Problem	191
C.1.4	Tension/compression Spring Design Problem	192
C.1.5	Rolling Element Bearing Design Problem	193
C.2	Multi-objective Engineering Design Problems	194

C.2.1	Welded Beam Design Problem	194
C.2.2	Multiple-disk Clutch Brake Design Problem	195
C.2.3	Pressure Vessel Design Problem	196
C.2.4	Speed Reducer Design Problem	196
C.2.5	Gear Train Design Problem	197
References		199
Author's Publications		221
Index		223

List of Figures

1.1	Classification of Data Mining techniques.	2
1.2	Clustering procedure.	3
1.3	Classification of population based metaheuristic techniques.	11
2.1	General Methodology for Data Clustering.	36
3.1	Hunting behavior of spotted hyenas: (A) Searching and tracking; (B) Chasing; (C) Troublesome and encircling; (D) Immobile situation and attack on prey.	41
3.2	2D position vectors of spotted hyena.	42
3.3	Searching for prey ($ \vec{E} > 1$).	43
3.4	Attacking for prey ($ \vec{E} < 1$).	43
3.5	3D position vectors and possible next locations of spotted hyena.	44
3.6	The convergence curves of SHO and the existing metaheuristics on benchmark test problems.	55
3.7	Effect of scalability on the performance of SHO algorithm.	56
3.8	Schematic view of welded beam problem.	60
3.9	Schematic view of tension/compression spring problem.	62
3.10	Schematic view of pressure vessel problem.	63
3.11	Schematic view of speed reducer problem.	65
3.12	Schematic view of rolling element bearing problem.	67
3.13	Schematic view of displacement of loaded structure.	68
3.14	Huddling behavior of emperor penguins.	70
3.15	<i>L</i> -shaped polygon huddling boundary of emperor penguins.	74

3.16 Exploration and Exploitation phases of EPO on F_2 , F_9 , and F_{17} benchmark test functions.	75
3.17 Swarm behaviors of emperor penguins in two-dimensional environment.	75
3.18 Flowchart of the proposed EPO algorithm.	77
3.19 Convergence analysis between the proposed EPO and other competitive algorithms on benchmark test problems.	86
3.20 Box plot obtained from the proposed EPO and other competitor algorithms on IEEE CEC-2015 benchmark test functions.	87
3.21 Sensitivity analysis of proposed EPO algorithm, (a) Number of iterations; (b) Number of search agents; (c) Control parameter M ; (d) Control parameter f ; (e) Control parameter l	90
3.22 Effect of scalability on the performance of EPO algorithm.	92
3.23 The adaptive grid mechanism of individuals.	103
3.24 Relocate the individuals if the new element lies outside the boundary of the grid.	103
3.25 Group selection mechanism.	105
3.26 Flowchart of the proposed MOSHO algorithm.	108
3.27 Best Pareto optimal solutions obtained by the proposed MOSHO algorithm on IEEE CEC-2009 benchmark test problems.	113
3.28 Best Pareto optimal solutions obtained by multi-objective algorithms on (a) ZDT1, (b) ZDT2, (c) ZDT3, (d) ZDT4, and (e) ZDT6 test problems.	116
3.29 Box plot of the statistical results for Hypervolume, Δ_p , Spread, and Epsilon on ZDT benchmark test problems.	117
3.30 Best Pareto optimal solutions obtained by the proposed MOSHO algorithm on DTLZ benchmark test problems.	121
3.31 Convergence analysis of proposed archive on (a) Concave, (b) Convex, and (c) Disconnected Pareto fronts.	122

3.32	Convergence analysis of selection approaches on (a) ZDT1, (b) ZDT3, and (c) ZDT6 test functions.	122
3.33	Schematic view of multiple-disk clutch brake design problem. . . .	125
4.1	Agent encoding scheme.	132
4.2	Selection of thresholds and cluster centers.	132
4.3	(a) Mandrill image; (b) Segmented image by ACMOSHO; (c) Airplane (F-16) image; (d) Segmented image by ACMOSHO; (e) Peppers image; (f) Segmented image by ACMOSHO; (g) House image; (h) Segmented image by ACMOSHO.	138
5.1	Agent representation scheme.	144
5.2	Selection of features from four-dimensional data set.	144
6.1	Concept of astrophysics strategy.	155
6.2	Position vectors of spotted hyenas; (a) Spherical; (b) Elliptical. . . .	155
6.3	Agent representation scheme.	158
6.4	Proposed agent encoding scheme.	161
6.5	(a) Actual Mandrill image; (b) Clustered image by AMOSHO_CFS; (c) Actual Airplane (F-16) image; (d) Clustered image by AMOSHO_CFS; (e) Actual Peppers image; (f) Clustered image by AMOSHO_CFS; (g) Actual House image; (h) Clustered image by AMOSHO_CFS.	169
6.6	Comparison of proposed AMOSHO_CFS approach with competitor approaches in terms of: (a) Number of clusters; (b) SC values. . . .	170

List of Tables

2.1	Classification of multi-objective metaheuristic algorithms and its applications in data clustering and feature selection.	32
3.1	Parameters settings of algorithms.	52
3.2	Results of unimodal benchmark functions.	53
3.3	Results of multimodal benchmark functions.	53
3.4	Results of fixed-dimension multimodal benchmark functions.	54
3.5	Results of composite benchmark functions.	54
3.6	ANOVA test results.	57
3.7	Comparison results for welded beam design.	61
3.8	Comparison of SHO with other algorithms in terms of statistical results for welded beam design problem.	61
3.9	Comparison results for tension/compression spring design.	62
3.10	Comparison of SHO with other algorithms in terms of statistical results for tension/compression spring design.	63
3.11	Comparison results for pressure vessel design problem.	64
3.12	Comparison of SHO with other algorithms in terms of statistical results for pressure vessel design problem.	64
3.13	Comparison results for speed reducer design problem.	66
3.14	Comparison of SHO with other algorithms in terms of statistical results for speed reducer design problem.	66
3.15	Comparison results for rolling element bearing design problem.	67
3.16	Comparison of SHO with other algorithms in terms of statistical results for rolling element bearing design problem.	67

3.17 Comparison results for displacement of loaded structure.	68
3.18 Comparison of SHO statistical for displacement of loaded structure.	69
3.19 Results of unimodal benchmark test functions.	83
3.20 Results of multimodal benchmark test functions.	83
3.21 Results of fixed-dimension multimodal benchmark test functions.	84
3.22 Results of composite benchmark test functions.	84
3.23 Results of IEEE CEC-2015 benchmark test functions.	85
3.24 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions using different simulation runs (i.e., 100, 500, 800, and 1000).	89
3.25 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations is fixed as 1000. The number of search agents are varied from 30 to 100.	89
3.26 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter M is varied from [1, 2, 3, 4].	89
3.27 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter f is varied from [1, 2] to [4, 5].	89

3.28 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter f is fixed as [2, 3]. The parameter l is varied from [0.5, 1] to [2, 2.5].	90
3.29 ANOVA test results.	93
3.30 Comparison results for pressure vessel design problem.	95
3.31 Statistical results obtained from different algorithms for pressure vessel design problem.	96
3.32 Comparison results for speed reducer design problem.	96
3.33 Statistical results obtained from different algorithms for speed reducer design problem.	97
3.34 Comparison results for welded beam design problem.	97
3.35 Statistical results obtained from different algorithms for welded beam design problem.	98
3.36 Comparison results for tension/compression spring design problem.	98
3.37 Statistical results obtained from different algorithms for tension/compression spring design problem.	99
3.38 Comparison results for rolling element bearing design problem.	99
3.39 Statistical results obtained from different algorithms for rolling element bearing design problem.	100
3.40 Comparison results for displacement of loaded structure problem.	100
3.41 Statistical results obtained from different algorithms for displacement of loaded structure problem.	101
3.42 Results of multi-objective algorithms on the IEEE CEC-2009 ($UF1 - UF10$) test functions.	111
3.43 Results of multi-objective algorithms on the ZDT benchmark ($ZDT1 - ZDT6$) test functions.	115

3.44 Results of the multi-objective algorithms on the DTLZ benchmark (<i>DTLZ1 – DTLZ9</i>) test functions.	119
3.45 Archive state in different generations for ZDT1, DTLZ6, and ZDT3 test functions using MOSHO.	123
3.46 Comparison of best solution obtained from different algorithms for welded beam design problem.	124
3.47 Comparison of best solution obtained from different algorithms for multiple-disk clutch brake design problem.	126
3.48 Comparison of best solution obtained from different algorithms for pressure vessel design problem.	126
3.49 Comparison of best solution obtained from different algorithms for speed reducer design problem.	127
4.1 Description of data sets.	135
4.2 Average and standard deviation of number of partitions obtained by the proposed ACMOSHO and competitive approaches on nine data sets over 40 independent runs.	136
4.3 Average and standard deviation of intra-cluster distance obtained by the proposed ACMOSHO and competitor algorithms on nine data sets over 40 independent runs.	137
4.4 Average and standard deviation of inter-cluster distance obtained by the proposed ACMOSHO and competitor algorithms on nine real life data sets over 40 independent runs.	137
4.5 Unpaired t -test between the best and second best algorithms on nine data sets.	138
4.6 Average and standard deviation of clustering obtained by the proposed ACMOSHO and competitive algorithms on four color images over 40 independent runs.	139
4.7 Unpaired t -test between the outperforming and second outperforming algorithms on coloured images.	139

5.1	Description of data sets.	148
5.2	The average(mean) performance of proposed AFSMOSHO on low-dimensional data sets.	149
5.3	The average(mean) performance of proposed AFSMOSHO on medium-dimensional data sets.	149
5.4	The average(mean) performance of proposed AFSMOSHO on high-dimensional data sets.	149
5.5	The obtained features frequency and their corresponding Laplacian Score (LS) using AFSMOSHO algorithm.	151
5.6	The comparison results between feature selection approaches using Representational Entropy (RE).	152
5.7	The comparison results between feature selection approaches using Class Separability (CS).	152
6.1	Description of data sets.	164
6.2	The obtained features frequency and their corresponding Laplacian Score (LS) using AMOSHO_CFS algorithm.	166
6.3	The obtained clusters frequency and their corresponding Classification Accuracy (AC) using AMOSHO_CFS algorithm. . . .	167
6.4	p -values obtained form Wilcoxon's rank sum test by comparing AMOSHO_CFS with other competitor algorithms.	167
6.5	The obtained average and standard deviation of cluster quality measures using the proposed and other competitive algorithms. . .	171
6.6	Average and standard deviation of clustering results obtained from different clustering algorithms on four color images over 40 independent runs.	172
6.7	Unpaired t -test between the best outperforming and second best outperforming algorithms for color images.	172
A.1	Unimodal benchmark test functions.	179

A.2	Multimodal benchmark test functions.	180
A.3	Fixed-dimension multimodal benchmark test functions.	181
A.4	Composite benchmark test functions.	182
A.5	IEEE CEC-2015 benchmark test functions.	183
B.1	IEEE CEC-2009 benchmark test functions.	188

List of Algorithms

1	Spotted Hyena Optimizer (SHO)	46
2	Emperor Penguin Optimizer (EPO)	78
3	Multi-objective Spotted Hyena Optimizer (MOSHO)	105
4	Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO)	131
5	Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer (AFSMOSHO)	143
6	Automatic Clustering and Feature Selection using Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)	157

Certificate

I hereby certify that the work which is presented in this thesis entitled "**Multi-objective Metaheuristic Approaches for Data Clustering in Engineering Application(s)**", in fulfillment of the requirement for the award of degree of "**Doctor of Philosophy**" submitted in Computer Science and Engineering Department of Thapar Institute of Engineering & Technology, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Vijay Kumar** and refers other research works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.



(Gaurav Dhiman)

Regn. No. 901503006

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.



(Dr. Vijay Kumar)

Assistant Professor

Computer Science and Engineering Department

Thapar Institute of Engineering & Technology, Patiala 147004, Punjab, India

Acknowledgement

I would like to express sincere appreciation to my supervisor **Dr. Vijay Kumar** for being a pillar of support and encouragement throughout my research work. His experience, strength, and tenderness have taught me valuable lessons, that are going to be of immense help to me in making decisions throughout my life. My sincere thanks to **Dr. Maninder Singh**, Professor and Head, CSED, Thapar Institute of Engineering & Technology, Patiala, for providing me the necessary administrative assistance and infrastructure that helped me in the completion of my research work. I am thankful to my doctoral committee members: **Dr. R.K. Sharma**, Professor, CSED, **Dr. Rinkle Rani**, Associate Professor, CSED, and **Dr. Sunil Singla**, Associate Professor, EIED, Thapar Institute of Engineering & Technology, Patiala for their helpful suggestions and regularly ensuring the progress of my research work. I am thankful to the faculty, staff members and research scholars of CSED for their support.

I offer my deepest gratitude to my father, **Mr. Rajinder Dhiman** whose love and constant encouragement has been a major source of inspiration in turning my vision into a reality. I am also thankful to my mother **Mrs. Kamlesh Dhiman**, my sister **Dr. Pallavi** for their love, encouragement, motivation, and confidence in me. I also offer my gratitude towards gods "**Shri Shiv Shankar**", "**Shri Guga Jahar Veer**" and goddesses "**Shri Mata Kali Devi**", "**Shri Mata Vaishno Devi**" for showering their divine blessings on me.

I also acknowledge the cooperation and encouragement by my friends in this journey.



(Gaurav Dhiman)

Patiala

September, 2019

Glossary of Abbreviations/Symbols

- A
 - ABC: Artificial Bee Colony
 - ACO: Ant Colony Optimization
 - AMOSA: Archived Multi-objective Simulated Annealing
 - ANN: Artificial Neural Network
- B
 - BBO: Biogeography based Optimization
 - BH: Black Hole Algorithm
 - BA: Bat Algorithm
 - BFO: Bacterial Foraging Optimization
- C
 - CS: Cuckoo Search
 - CSS: Charged System Search
- D
 - DE: Differential Evolution
- E
 - EA: Evolutionary Algorithm
 - ES: Evolutionary Strategy
- F
 - FA: Firefly Algorithm
 - FCM: Fuzzy *C* Means

- G
 - GA: Genetic Algorithm
 - GP: Genetic Programming
 - GSA: Gravitational Search Algorithm
 - GWO: Grey Wolf Optimizer
 - GbSA: Galaxy based Search Algorithm
 - GD: Generational Distance
- H
 - HS: Harmony Search
 - HV: Hypervolume
- I
 - IGD: Inverted Generational Distance
 - IWO: Invasive Weed Optimization
- M
 - MFO: Moth-Flame Optimization
 - MOACO: Multi-objective Ant Colony Optimization
 - MOEA/D: Multi-objective Evolutionary Algorithm based on Decomposition
 - MOPSO: Multi-objective Particle Swarm Optimization
 - MVO: Multi-Verse Optimizer
- N
 - NFL: No Free Lunch
 - NSGA-II: Non-dominated Sorting Genetic Algorithm 2

- P
 - PSO: Particle Swarm Optimization
 - PESA-II: Pareto Envelope based Selection Algorithm 2
- S
 - SCA: Sine Cosine Algorithm
 - SVM: Support Vector Machine
 - SPEA2: Strength Pareto Evolutionary Algorithm 2

Abstract

Clustering is the process of combining similar data objects into a number of groups (called clusters). Clusters are formed in such a way that data objects having similar nature are kept in one cluster and are dissimilar to data objects of other clusters. This thesis focuses on the concept of clustering, i.e., automatically determining the number of features and number of clusters. Due to unknown number of cluster, clustering technique is treated as NP-hard problem. To solve this problem, metaheuristic algorithms are utilized. In this thesis, two novel bio-inspired metaheuristic optimization algorithms have been proposed namely Spotted Hyena Optimizer (SHO) and Emperor Penguin Optimizer (EPO) for solving real-life engineering design problems. The proposed approaches are assessed on standard benchmark test functions. The convergence and computational complexity have also been analyzed to ensure the applicability of proposed algorithms. The performance of the proposed algorithms are analysed and compared with different algorithms such as GWO, PSO, MFO, MVO, SCA, GSA, GA, and HS. Experimental results reveal that the proposed algorithms are able to solve constrained and unconstrained engineering design problems.

In addition to above-proposed algorithms, a multi-objective version of spotted hyena optimizer is proposed and named as Multi-objective Spotted Hyena Optimizer (MOSHO). In order to determine the better solution, the concept of Pareto dominance is utilized in the proposed approach. An external repository is used to store the Pareto optimal solutions. An adaptive grid mechanism is used to produce the distributed Pareto fronts and improve diversity. Moreover, the group selection mechanism is also employed for better convergence. The roulette wheel mechanism is used to select the effective solutions from archive to simulate the social and hunting behaviors of spotted hyenas. The proposed algorithm has been tested on multi-objective benchmark test functions and then applied on constrained engineering design problems to demonstrate its applicability on real-life problems. The experimental results reveal that the proposed algorithm

performs better than the others and produces the Pareto optimal solutions with high convergence.

An automatic data clustering technique is proposed which utilizes Multi-objective Spotted Hyena Optimizer (MOSHO) algorithm. It employs real-coded variable length representation to encode cluster centers automatically. In this technique, two concepts namely threshold setting and cutoff computation are used to refine the search process. Consequently, the proposed technique is also compared with other existing clustering techniques. This technique is then applied to image segmentation and its performance is compared with other approaches.

For high-dimensional data sets, the existing feature selection techniques may not produce optimal feature subset. To resolve this problem, a novel automatic feature selection technique is proposed, which is based on multi-objective spotted hyena optimizer. This approach finds the optimal number of features dynamically (i.e., automatically during simulation run). A novel idea of threshold setting is proposed for determining the most appropriate feature subset. Subsequently, a new objective function is also designed for the efficient search process. Experimental results suggest that the proposed approach performs superior to existing feature selection approaches.

This thesis also presents the problems associated with the collective approach of both feature selection and automatic clustering techniques. A novel approach for automatic clustering and feature selection using multi-objective spotted hyena optimizer is also proposed. This approach finds out both the features as well as the number of clusters simultaneously from the given data set. A variable length agent is proposed to encode both cluster centers and features of the different number of clusters. Two novel thresholds have been developed for finding the optimal number of clusters and features. A novel clustering criterion has also been designed to search space efficiently. The performance of the proposed technique is tested on two real-life applications and compared with the existing algorithms.

Experimental results show that the proposed approach is superior than the other competitor algorithms.

Keywords: *Clustering, Feature selection, Spotted hyena optimizer, Emperor penguin optimizer, Optimization, Metaheuristic, Constrained optimization, Unconstrained optimization, Benchmark test functions, Convergence, Diversity, Engineering design problem.*

Chapter 1

Introduction

“Although this may seem a paradox, all exact science is dominated by the concept of approximation.”

By Bertrand Russell

1.1 Clustering

Since last few years, real-life applications have been suffering from many critical challenges in data analysis due to exponential growth in multimedia tools and internet technology. Some of these challenges include information extraction from the large data set, data analysis, and data classification^[1]. In order to tackle the aforementioned challenges, data mining is one of the most preferable technique among the available techniques. Data mining is a process of discovering knowledge from a large amount of data. It can be performed on any type of data either text or image. The data mining techniques are broadly classified into two categories such as direct and indirect data mining techniques (see Fig. 1.1). The direct data mining techniques are further categorized into three categories such as classification, estimation, and prediction. Classification is a process of decomposing the given data set into different number of classes based on the knowledge of predefined classes or structure of data set itself. It is also known as supervised learning. In supervised learning, number of classes in the given data

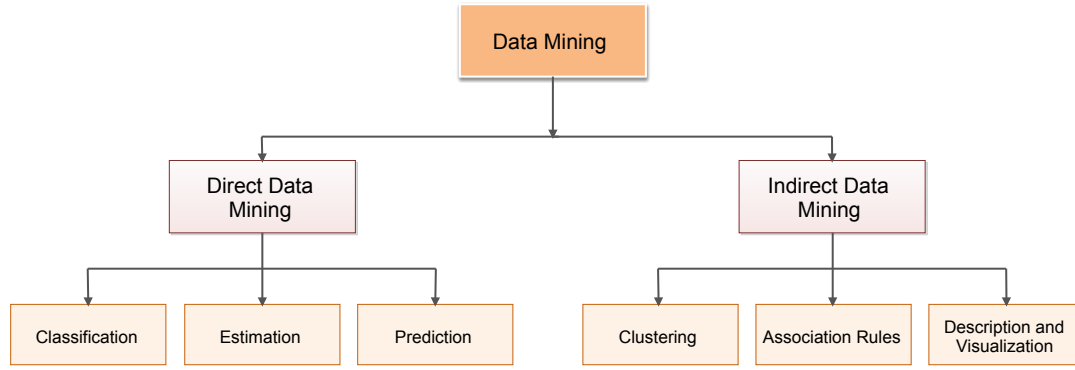


Fig. 1.1. Classification of Data Mining techniques.

set is known in advance and assign the class label of unclassified cases.

The indirect data mining techniques are further categorized into three categories such as clustering, association rule-mining, and visualization. Among them, clustering is widely used indirect data mining technique. Clustering is a group of data objects which are very much alike between data objects or dissimilar to other clusters. Clustering is an unsupervised learning approach to discover the set of relevance data objects to the similarity of characteristics without any prior knowledge. It has been applied in many scientific and engineering fields such as bio-informatics, information retrieval, forecasting, image segmentation and so forth^[2].

1.1.1 Definitions

Clustering is defined as a process for grouping the data objects into number of clusters. Everitt *et al.*^[3] provide different definitions of cluster which are given below:

Definition 1.1.1. A cluster is set of entities which are alike, and entities from different clusters are not alike.

Definition 1.1.2. A cluster is an aggregate of points in the test space such that the

distance between any two points in the cluster is less than the distance between any points in the cluster and any point not in it.

1.1.2 Clustering Procedure

There are six phases for whole clustering process using multi-objective metaheuristic technique. Fig. 1.2 shows the phases of clustering procedure which are described below:

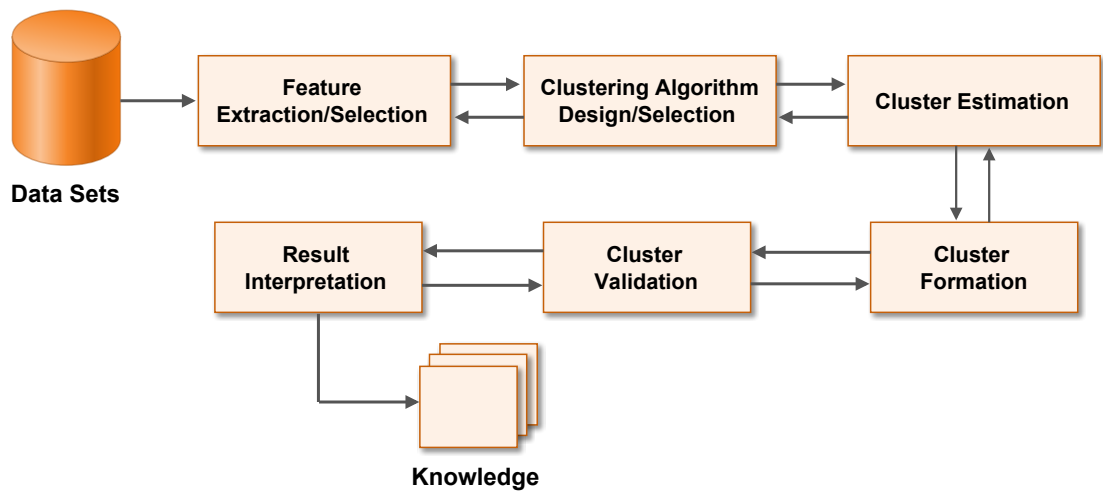


Fig. 1.2. Clustering procedure.

Phase 1: Data pre-processing techniques

The removal of noisy and irregularity from the given data set will be done in this phase. For this purpose, data pre-processing technique will be used to reduce the storage space and simplify the design process.

Phase 2: Feature selection/extraction

In this phase, the important features will be extracted/selected from the given data set. It plays an important role in determining the number of clusters in the given data set. It can also reduce time and space complexities of data clustering

technique.

Phase 3: Clustering algorithm design or selection

This step consists of determining an appropriate distance measure and creating a clustering criterion function. All clustering algorithms directly or indirectly depend upon the distance measures. After determining the distance measure, the clustering could be interpreted as an optimization problem with specific clustering criterion function.

Phase 4: Cluster estimation using metaheuristic technique

The number of cluster in a given data set depends upon the number of features and data points present in the given data set. After selecting the important features from **Phase 2**, the number of clusters will be determined using multi-objective metaheuristics during run time.

Phase 5: Cluster formation using multi-objective metaheuristic technique

In this phase, data points will be assigned to the number of clusters determined from **Phase 4**. The assignment of data points will be done through distance measures. A novel multi-objective metaheuristic based clustering technique will be used. The cluster formulation will be further refined through multi-objective metaheuristic technique.

Phase 5: Cluster validation

Cluster validity indices will be used to optimize and validate the clusters formed through multi-objective metaheuristic based clustering technique.

Phase 6: Interpretation of the results

The evaluation of the proposed approach will be done using clustering metrics such as inter-cluster and intra-cluster measures. The validation of proposed approaches will be done on real-life engineering problems.

1.1.3 Distance Measures

Distance measure is an important to distinguish the clusters embedded in the given data set. However, there is no single distance measure which is best for all

types of clustering related problems. In this subsection, five well-known distance measures are discussed in detail. These are Euclidean, Standardized Euclidean, Cosine, Spearman, and Chebyshev.

1.1.3.1 Euclidean Distance

It is most widely used distance measure. The Euclidean distance (D_e) between two points x_i and x_j is defined as^[4]:

$$D_e(x_i, x_j) = \left(\sum_{l=1}^D |x_{il} - x_{jl}|^2 \right)^{1/2} \quad (1.1)$$

where D is the dimension space, x_{il} and x_{jl} represent the l th dimension value of x_i and x_j , respectively.

1.1.3.2 Standardized Euclidean Distance

It is defined as the Euclidean distance between the points divided by their standard deviation which is described as^[4]:

$$D_{se}(x_i, x_j) = (x_i - x_j)DM^{-1}(x_i - x_j)^T \quad (1.2)$$

where DM is the diagonal matrix with diagonal elements are given by the variance of variable x_j over n data points.

1.1.3.3 Cosine Distance

It is a measure of similarity between two vectors by calculating the cosine of angle between them. It is defined as^[5]:

$$D_{cos}(x_i, x_j) = 1 - \left(\frac{x_i^T x_j}{|x_i||x_j|} \right) \quad (1.3)$$

If the values of x_i and x_j are non-negative, then it is bounded between the range of $[0, 1]$.

1.1.3.4 Spearman Distance

This distance measure is derived from the spearman correlation coefficient(s) and is defined as^[6]:

$$D_{spear}(x_i, x_j) = 1 - S(x_i, x_j) \quad (1.4)$$

$$where, \quad S(x_i, x_j) = \frac{\sum_{r=1}^D (R(x_{ir}) - \bar{R})(R(x_{jr}) - \bar{R})}{\sqrt{\sum_{r=1}^D (R(x_{ir}) - \bar{R})^2 \sum_{r=1}^D (R(x_{jr}) - \bar{R})^2}} \quad (1.5)$$

where R and $\bar{R}$ are the rank and mean rank, respectively.

1.1.3.5 Chebyshev Distance

It computes the maximum of the absolute difference between the features of a pair of data points. It is defined as^[7]:

$$D_{ch}(x_i, x_j) = \text{Max}_{1 \leq l \leq D} (|x_{il} - x_{jl}|) \quad (1.6)$$

It takes less time to calculate the distance between data sets.

1.1.4 Validity Indices

The validity indices are used for measuring the "goodness" of clustering results compared to others which are created by the same or other clustering approaches by using different parameter values. The main aspect of clustering techniques is the selection of suitable objective functions that are to be optimized. Generally, cluster validity indices are used as the objective function. Several validity indices

have been used in various clustering algorithms. In this subsection, six well-known validity indices are described in detail.

1.1.4.1 Dunn Index (DI Index)

Dunn index is based on geometrical measures of cluster compactness and separation which is defined as^[8]:

$$Dunn(K) = \min_{i=1}^K \left\{ \min_{j=i+1}^K \left(\frac{D_e(C_i, C_j)}{\max_{k=1}^K (d(C_k))} \right) \right\} \quad (1.7)$$

where $D(C_i, C_j) = \min_{x \in C_i, y \in C_j} \{D_e(x, y)\}$ and $d(C_k) = \max_{x, y \in C_k} \{D_e(x, y)\}$. The diameter of DI index should be small and distances between clusters should be large in order to achieve proper clustering.

1.1.4.2 Davis-Bouldin Index (DB Index)

Davis-Bouldin index is used for similarity measures ($SM_{i,j}$) between the clusters C_i and C_j . It is defined as the ratio of sum of within-cluster scatter (S_j) to between-cluster separation (D_{ij}).

$$SM_{i,j} = \frac{S_i + S_j}{ds_{ij}} \quad (1.8)$$

$$\text{where, } D_{ij} = D_e(m_i, m_j) \quad (1.9)$$

$$S_i = \frac{1}{|C_i|} \sum_{x \in C_i} D_e(x, m_i) \quad (1.10)$$

DB index is defined as^[9]:

$$DB(K) = \frac{1}{K} \sum_{i=1}^K R_i \quad (1.11)$$

$$\text{where, } R_i = \max_{i=1, \dots, K, i \neq j} (R_{ij}), \quad i = 1, 2, \dots, K \quad (1.12)$$

In order to achieve relevant clustering, the value of DB index should be minimized.

1.1.4.3 Xie and Beni Index (XB Index)

It is defined as the ratio of the total fuzzy cluster variance to the minimum separation of the clusters which is defined as^[10]:

$$XB(K) = \frac{\sum_{i=1}^K \sum_{j=1}^N (u_{ij})^q |x_j - m_i|^2}{N \min_{i \neq j} |m_i - m_j|^2} \quad (1.13)$$

The numerator represents the fuzzy partitions and denominator represents the separation measure between clusters.

1.1.4.4 Pakhira-Bandyopadhyay-Maulik Index (PBM Index)

PBM index is defined as^[11]:

$$PBM(K) = \left(\frac{1}{K} \times \frac{E_1}{E_K} \times D_K \right)^2 \quad (1.14)$$

$$\text{where, } E_K = \sum_{i=1}^K \sum_{j=1}^N (u_{ij}) |x_j - m_i| \quad (1.15)$$

$$E_1 = \sum_{j=1}^N |x_j - m_i|^2 \quad (1.16)$$

$$D_K = \max_{i,j=1,2,\dots,K} \{|m_i - m_j|\} \quad (1.17)$$

where N is the total number of data points in the data set, $U(X) = [u_{ij}]_{N \times K}$ is a partition matrix for the data, and m_i is the center of the i th cluster.

1.1.4.5 *I* Index

I index is defined as^[12]:

$$I(K) = \left(\frac{1}{K} \times \frac{ED_1}{ED_K} \times D_K \right)^q$$

Here,

$$ED_K = \sum_{i=1}^K \sum_{j=1}^{n_K} Dst_e(x_j^K, m_i)$$

$$D_K = \max_{i,j=1}^K Dst_e(m_i, m_j)$$
(1.18)

where K is the number of clusters for partitioning the data set. The value of q is set to 2. D_K is the maximum Euclidean distance between two center clusters. Dst_e represents the difference between m_i and m_j . $Dst_e(x_j^K, m_i)$ is calculated as:

$$Dst_e(x_j^K, m_i) = Dist_{sym}(x, p_i) \times D_e(x, m_i)$$
(1.19)

where m_i is the cluster center of particular cluster i . $D_e(x, m_i)$ is the Euclidean distance between point x and m_i .

1.1.4.6 *Sym* Index

Sym index is defined as^[13]:

$$Sym(K) = \left(\frac{1}{K} \times \frac{1}{ED_K} \times D_K \right)$$

$$ED_K = \sum_{i=1}^K E_i$$

$$E_i = \sum_{j=1}^{n_K} D_p^*(\bar{x}_j^K, \bar{m}_i)$$

$$D_K = \max_{i,j=1}^K Dst_e(m_i, m_j)$$
(1.20)

Clustering is one of the most challenging tasks as the data can not be partitioned unambiguously. A single clustering metric is not a reliable measure for confronting the performance of the clustering process. There is a need to consider multiple cluster validity indices to analyze the clustering accuracy. Several cluster validity indices are used to improve the quality of clusters produced through the clustering process. Using this fact, clustering can be considered as an optimization problem. Therefore, metaheuristic techniques are used to solve the cluster issues.

1.2 Metaheuristic Techniques

In recent years, metaheuristic algorithms are used to solve clustering problems. These algorithms are incorporated to optimize cluster validity indices. There is a lot of interest in developing metaheuristic algorithms that are computationally inexpensive, flexible, and simple by nature. Every metaheuristic algorithm needs to address the exploration and exploitation of the search space^[14,15,16,17,18,19] and maintains a good balance between them. The exploration phase investigates the different promising regions in a search space whereas exploitation is able to search the optimal solutions around the promising regions^[20,21,22,23,24]. Therefore, there is a need for fine tuning of these two phases to achieve the near optimal solutions. Despite, the number of metaheuristic algorithms is available in literature, the question is raised why we need to develop more optimization techniques. The answer lies in No Free Lunch (NFL) theorem^[25]. According to this theorem, the performance of one optimization algorithm for a specific set of problems does not guarantee to solve other optimization problems because of their different nature. NFL theorem allows researchers to propose some novel metaheuristic algorithms for solving the problems in different fields^[26]. This motivated us to develop a novel technique which is gradient free and effective optimization algorithm to solve real-life engineering problems.

1.2.1 Classification of Metaheuristic Techniques

Metaheuristics are broadly classified into two categories namely, single solution and population based algorithms. Single solution based algorithms are in which a solution is randomly generated and improved until the best result is obtained. Population based algorithms are in which a set of solutions is randomly generated in the given search space and solution, values are updated during the course of iterations until the best solution is found.

However, single solution based algorithms may trap into local optima. It reforms only one solution which is randomly generated for a given problem. On the other hand, population based algorithms are able to find the global optimum. Due to this, researchers have attracted towards population based algorithms nowadays.

The categorization of population based metaheuristic algorithms is based on the theory of evolutionary, logical behavior of physics, swarm intelligence, and biological behaviors as shown in Fig. 1.3.

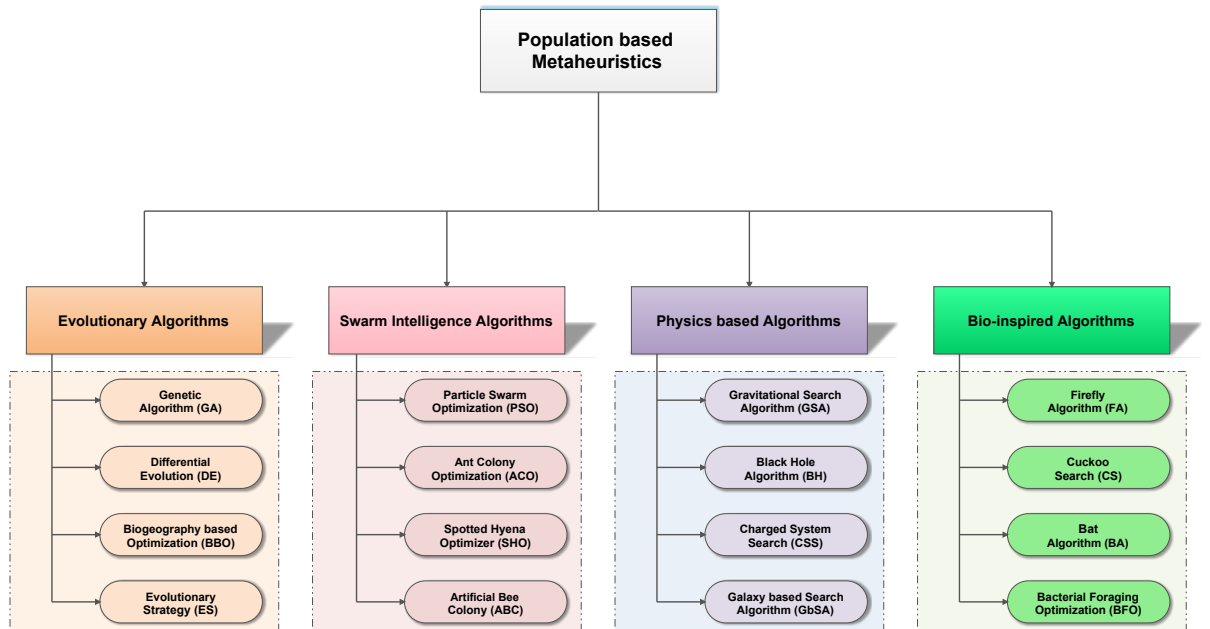


Fig. 1.3. Classification of population based metaheuristic techniques.

Evolutionary algorithms are inspired by evolutionary processes such as reproduction, mutation, recombination, and selection. These algorithms are based

on the survival fitness of candidate in a population (i.e., a set of solutions). The physics law based algorithms are inspired by physical processes according to some physics rules such as gravitational force, electromagnetic force, inertia force, heating and cooling of materials. Swarm intelligence based algorithms are inspired by the collective intelligence of swarms.

Generally, swarm intelligence based algorithms are very easy to implement than the other algorithms due to less number of parameters are required. The well-known swarm intelligence technique is Particle Swarm Optimization (PSO)^[27]. PSO is inspired by the social behavior of fish schooling or bird flocking. Each particle can move around the search space and update its current position with respect to the global best solution.

The most challenging characteristic of population based algorithms is multi-objectivity. The multi-objective optimization deals with optimization of multiple and competing for objective functions^[28]. The main challenge in multi-objective optimization is to address the multiple objectives which are frequently in conflict. There are two main approaches for multi-objective optimization such as *priori* and *posteriori*^[29,28,30,31,32,33,34].

In *priori* approach, a multi-objective problem is converted to a single-objective with a set of weights that defines the significance of each objective to solve the problem. A *posteriori* method permits to explore the performance of problem and selects one of the obtained solutions based on their obligations^[30].

In contrast to single-objective, there is no single solution in multi-objective optimization and represents various trade-offs between the objectives. Therefore, it is more time consuming to obtain the Pareto fronts because it usually requires to produce many points on Pareto front for good approximations. Even accurate solutions on Pareto front, there is still no guarantee that these solution points will distribute uniformly on the front^[35]. The higher dimensional problems can have extremely complex hyper-surface as its Pareto front^[32,36,37,38,39]. Thus, it is more

challenging to solve high-dimensional problems.

The concept of multi-objective optimization using stochastic techniques was proposed by David Schaffer^[40]. The advantages of these techniques are local optima avoidance and gradient-free mechanism that made them applicable to real world problems.

The applications of multi-objective optimization techniques can be found in various fields such as civil engineering^[41], mechanical engineering^[42], system engineering^[43,44,1], bioinformatics^[45], pattern recognition^[15], wireless sensor networks^[17,16], forecasting^[18,19,20,21], cloud computing^[24], and other fields^[46,22,37,28].

1.2.1.1 Single-objective Optimization

Many real-life problems need to achieve several objectives such as minimize risks, minimize cost, maximize reliability, etc^[31]. The main aim of single-objective optimization is to find the best optimal solution and deals with only one objective which is to be minimized or maximized. The mathematical formulation of single-objective optimization is as follows:

$$\text{Minimize/Maximize: } F(\vec{z}) = f_1(\vec{z}) \quad (1.21)$$

$$\text{Subject to:} \quad (1.22)$$

$$g_j(\vec{z}) \geq 0, j = 1, 2, \dots, p$$

$$h_j(\vec{z}) = 0, j = 1, 2, \dots, q \quad (1.23)$$

$$lb_j \leq z_j \leq ub_j, j = 1, 2, \dots, r \quad (1.24)$$

where p is the number of inequality constraints, g_j is j^{th} inequality constraints, q is the number of equality constraints, h_j is j^{th} equality constraints, r is the number of variables, lb_j and ub_j are lower and upper bounds of j^{th} variable, respectively.

1.2.1.2 Multi-objective Optimization

Multi-objective optimization refers to the optimization with more than one objective (criterion) function of a given problem. It can be formulated as^[47,33]:

$$\text{Minimize : } F(\vec{z}) = [f_1(\vec{z}), f_2(\vec{z}), \dots, f_n(\vec{z})] \quad (1.25)$$

$$\text{Subject to:} \quad (1.26)$$

$$g_i(\vec{z}) \geq 0, \quad i = 1, 2, \dots, m$$

$$h_i(\vec{z}) = 0, \quad i = 1, 2, \dots, p \quad (1.27)$$

where $\vec{z} = [z_1, z_2, \dots, z_k]^T$ is the vector of decision variables, m is the number of inequality constraints, p is the number of equality constraints, g_i is i^{th} inequality constraints, h_i is i^{th} equality constraints, and obj is the number of objective functions $f_i : \mathbb{R}^{obj} \rightarrow \mathbb{R}$.

In multi-objective, the solutions in a search space cannot be compared by relational operators due to multi-criterion comparison metrics^[34]. The comparison of two solutions was first proposed by Edgeworth^[48] and further extended by Pareto^[49]. The mathematical formulation of Pareto dominance is described as^[47]:

Definition 1.2.1. Pareto dominance

Assume there are two vectors $\vec{x} = (x_1, x_2, \dots, x_r)$ and $\vec{y} = (y_1, y_2, \dots, y_r)$. Vector $\vec{x}$ is said to dominate vector $\vec{y}$ (such as $x \prec y$) if and only if:

$$\forall i \in \{1, 2, \dots, r\} : f_i(\vec{x}) \leq f_i(\vec{y}) \wedge \exists i \in \{1, 2, \dots, r\} : f_i(\vec{x}) < f_i(\vec{y}) \quad (1.28)$$

Definition 1.2.2. Pareto optimality

A solution $\vec{x} \in X$ is called Pareto optimal if and only if:

$$\nexists \vec{y} \in X \mid \vec{y} \prec \vec{x} \quad (1.29)$$

Definition 1.2.3. Pareto optimal set

The set of all Pareto optimal solutions including all non-dominated solutions of a problem is called Pareto optimal set if and only if:

$$P_s = \{\vec{x}, \vec{y} \in X \mid \exists \vec{y} \succ \vec{x}\} \quad (1.30)$$

Definition 1.2.4. Pareto optimal front

A set containing the objective values corresponding to Pareto optimal solutions in Pareto optimal set is called Pareto optimal front. It is defined as:

$$P_f = \{f(\vec{x}) \mid \vec{x} \in P_s\} \quad (1.31)$$

1.2.2 Performance Metrics

The four well-known performance metrics are used to investigate the performance of multi-objective optimization algorithms. These are Hypervolume (HV) ^[50,51], $\Delta_p(p=1)$ ^[52,53,54], Spread ^[55,51], and Epsilon (ε) ^[56,51].

The mathematical formulation of these performance metrics are as follows:

$$\text{Hypervolume (HV)} = \text{volume} \left(\bigcup_{k=1}^{|Q|} u_k \right) \quad (1.32)$$

It computes the volume which is covered by members of Q in the objective space where objectives are to be minimized. The solution k as the diagonal corners of hypercube where for each $k \in Q$, a hypercube u_k is constructed with reference point

(R_p) . The reference point (R_p) is a vector of worst objective function values.

$$\text{Generational Distance (GD)} = \frac{\sqrt{\left(\bigcup_{k=1}^{|Q|} D_k^2\right)}}{|Q|} \quad (1.33)$$

where D_k is the Euclidean distance between the solution $k \in Q$ in the objective space such as,

$$D_k = \min_{j \in |X|} \sqrt{\sum_{y=1}^Y \left(f_y^{(z)} - f_y^{*(z)}\right)^2} \quad (1.34)$$

where $f_y^{*(z)}$ is y^{th} objective function of j^{th} member of set X .

$$\text{Inverted Generational Distance (IGD)} = \frac{\sqrt{\sum_{m=1}^{tn} D_m^2}}{tn} \quad (1.35)$$

where D_m is the Euclidean distance between m^{th} true Pareto optimal solution and approximation Pareto optimal solutions in a given set. tn describes the number of true Pareto optimal solutions.

$$\text{Spread}(\Delta) = \frac{\sum_{t=1}^T d_t^e + \sum_{l=1}^Q |D - d^*|}{\sum_{t=1}^T d_t^e + |Q|d^*} \quad (1.36)$$

where D is the Euclidean distances having mean value d^* , d_t^e is distance between the intense solutions of Q corresponding to t^{th} objective function.

$$\text{Epsilon}(\varepsilon) = \text{Sup}_{x \in [0,1]} \text{Min}_{p \in P} \text{Max}\{p - x, f(p) - f(x)\} \quad (1.37)$$

where f is the Pareto front, P is a set of size k , and p is ϵ -approximates of the x if $\text{Max}\{p - x, f(p) - f(x)\} \leq \epsilon$.

1.3 Thesis Layout

The thesis is organized as per the following chapters.

Chapter 1: Introduction

This chapter introduces the concept of metaheuristics, data clustering, and feature selection. It briefly describes the concept of clustering and cluster validation. The problem of data clustering and feature selection is stated. This chapter also discusses the challenges faced while dealing with large data sets. In a nutshell, this chapter explains the basic concepts of metaheuristics and data clustering.

Chapter 2: Literature Review

This chapter contains a detailed review of the literature on multi-objective based data clustering, feature selection, and simultaneously clustering and feature selection approaches. Literature shows that finding the relation between data clusters is the best way to deal with the problem of ever increasing data. This chapter also explains other approaches that are being used frequently by researchers and developers using multi-objective metaheuristics.

Chapter 3: Bio-inspired based Metaheuristic Optimization Techniques

In this chapter, the proposed approaches for single-objective metaheuristics are introduced. Spotted Hyena Optimizer (SHO) mimics the behaviors of spotted hyenas in nature. Further, SHO is tested on real-life engineering design problems to show its efficiency. The second algorithm named as Emperor Penguin Optimizer (EPO) is proposed which mimics the huddling behavior of emperor penguins. This algorithm is validated on well-known data sets and engineering applications.

This chapter also proposes the multi-objective version of SHO algorithm named

as Multi-objective Spotted Hyena Optimizer (MOSHO). It is used to optimize the multiple objectives problems. In the proposed algorithm, a fixed-sized archive is employed for storing the non-dominated Pareto optimal solutions. The roulette wheel mechanism is used to select the effective solutions from archive to simulate the social and hunting behaviors of spotted hyenas. The proposed algorithm is tested on 24 benchmark test functions and compared with six recently developed metaheuristic algorithms. The proposed algorithm is then applied on six constrained engineering design problems to demonstrate its applicability on real-life problems.

Chapter 4: Automatic Clustering using Multi-objective Spotted Hyena Optimizer

In this chapter, a novel clustering technique is proposed using multi-objective spotted hyena optimizer. This technique is named as Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO). The two new concepts namely weighted cluster-centroid computation and threshold setting are also introduced to refine cluster centroids. A novel fitness function strategy is also proposed for efficient searching. The proposed technique is compared with various well-known multi-objective based clustering approaches. The results reveal that the proposed technique performs better than other competitive approaches.

Chapter 5: Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer

In this chapter, a novel feature selection technique is proposed using multi-objective spotted hyena optimizer. This technique is named as Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer (AFMOSHO). The optimal features are selected using cluster-centroid based encoding scheme. A novel threshold concept has been developed for feature ranking. A novel fitness function has been designed to evaluate the fitness of agents. The proposed

technique is compared with various multi-objective based feature selection approaches. The results show that the proposed technique performs better than other approaches.

Chapter 6: Simultaneously Clustering and Feature Selection using Multi-objective Optimization

In this chapter, a novel Algorithm for Clustering and Feature Selection is proposed based on Astrophysics concepts using Multi-objective Spotted Hyena Optimizer named as AMOSHO_CFS. The proposed technique has shown proficiency in finding the optimal feature subsets and number of clusters. The novel concept of dynamic threshold setting is developed to get the optimal number of clusters and feature sub-set. The fitness function is used to search optimal clusters and features in the proposed technique. Validation of the proposed technique is performed using nine real- life data sets. The statistical significance of proposed technique is attributed by unpaired t - test and Wilcoxon's rank sum test. It is also applied on solving image segmentation and micro array data analysis problems. It has been observed that proposed AMOSHO_CFS outperforms the other competitive techniques in terms of the optimal number of clusters and finding an optimal subset of features.

Chapter 7: Conclusion and Future Directions

This chapter concludes the thesis by providing a brief overview of proposed approaches and providing an insight into future work. The summarization of future directions in the area of metaheuristics, clustering, and feature selection will enlighten the research community regarding research areas in the field of optimization. It highlights the contributions and enlists the directions that need to be worked in the near future.

Chapter2

Literature Review

“I still believe in the possibility of a model of reality, that is to say, of a theory, which represents things themselves and not merely the probability of their occurrence.”

By Albert Einstein

This chapter provides a review of the major issues related to clustering. The literature is reviewed from three different points of view. First section deals with the review of multi-objective metaheuristic techniques for data clustering. Thereafter, a brief overview of research work based on multi-objective feature selection techniques is covered. The third section describes the brief review of data clustering and feature selection using multi-objective metaheuristic techniques.

2.1 Review on Multi-objective based Clustering Techniques

Handl and Knowles^[57] proposed a Multi-objective Clustering with Automatic K -determination (MOCK) approach. It consists of two main phases, namely initial clustering and model-selection. In the initial phase, it optimizes the two complementary clustering objectives using Multi-objective Evolutionary Algorithm (MOEA). In the model-selection phase, it analyzes the shape of tradeoff curve. Based on these phases, the algorithm provides an estimate of the quality of

all individual clustering solutions and determines a set of potentially promising clustering solutions.

Liu et al.^[58] developed an automatic clustering approach based on genetic algorithm, called Multi-objective Genetic K -means Algorithm (MOKGA), which minimizes two objective functions such as the number of clusters and the partitioning error. Additionally, a multiple Pareto optimal front layer ranking method was proposed to maintain a relatively consistent population size in the genetic process. The authors analyzed the clustering results by using several cluster validity techniques.

Matake et al.^[59] showed that the computational cost of MOCK is too high when the technique is applied to large data sets. Thus, they developed a data-clustering algorithm for web mining based on scalable automatic K -determination scheme. The main contributions were: (i) a new automatic K -determination scheme that was able to discover the appropriate number of clusters at low cost, and (ii) a new strategy for K -determination that did not require a null model. These improvements allow the application of MOCK to large data sets. This technique reduced the Pareto size that was helpful to determine the appropriate number of clusters.

Bandyopadhyay et al.^[60] proposed a Non-dominated Sorting Genetic Algorithm 2 (NSGA-II) based clustering technique which simultaneously optimized two cluster validity measures. A real encoding with centroid based representation was used. This technique was generated better clustering performance because it simultaneously optimized the indexes rather than individually.

Suresh et al.^[61] applied Differential Evolution (DE) algorithm on automatic clustering in a multi-objective optimization framework. The study simultaneously optimized the two conflicting fuzzy validity indices. A real-coded representation of the search variables was employed. The gap statistic was applied in choosing the most interesting solutions from the Pareto front. The final clustering quality

was evaluated by considering two measures, i.e., Rand index and Silhouette index. The performance of this approach was better than MOCK and NSGA-II in terms of compactness.

Kundu et al.^[62] proposed a hybrid clustering approach GADE based on Genetic Algorithm (GA) and Differential Evolution (DE). GADE incorporated GA operators into the multi-objective framework. Xie-Beni index and Fuzzy C Means (FCM) measure were used as the objective functions to be optimized. A real-coded representation of the search variables was employed for accommodating the variable number of cluster centers. The performance of GADE was tested on well-known real-life data sets.

A clustering methodology based on a Multi-objective Particle Swarm Optimization (MOPSO) was presented by Paoli et al.^[63]. This approach was proposed to solve three different issues related to the segmentation of hyperspectral images: (i) estimation of class statistical parameters, (ii) detection of the best discriminative bands without requiring their number a priori, and (iii) estimation of the number of data classes (or number of clusters) characterizing the image being considered. MOPSO is guided by three optimization criteria, namely Log-likelihood Function (LF), Bhattacharyya Statistical Distance (BD) between classes, and Minimum Description Length (MDL). An experimental analysis was done on both simulated and real hyperspectral images. In general, MOPSO provided satisfactory classification accuracy while reducing the number of bands used for the classification task. However, its main drawback is the time required for the optimization process, which may reach several hours depending on the image size.

Ma et al.^[64] developed the Immune Dominance Clonal Multi-objective Clustering (IDCMC) algorithm based on clonal selection with immune dominance. This approach divided the single population into three sub-populations according to three different measurements. It adopted distinct evolution and selection

strategies for every sub-population. The experiments were considered on six artificial data sets with manifold structures and handwritten digit data sets. The main advantages of this approach were efficiency and non-linear time complexity.

Saha and Bandyopadhyay^[65] proposed stability induced *Sym* index and Simulated Annealing based Multi-objective Optimization technique (SSym-AMOSA). It simultaneously optimized two objective functions, one reflecting the total cluster symmetry and the other reflecting the stability of the obtained partitions over different bootstrap samples of the data set. The authors proposed a semi-supervised method to select the best solution from a large number of non-dominated solutions on the final Pareto front. The performance of SSym-AMOSA was compared with those of four existing clustering techniques. SSym-AMOSA was able to determine the correct number of clusters for almost all the tested data sets. Besides this, it had some deficiencies such as low convergence rate and high computational cost.

In Automatic Multi-objective Clustering technique with Point Symmetry based distance (VAMOSa)^[66], both XB and Sym indices are defined as objective functions to determine the appropriate number of clusters. SSym-AMOSA is also used to select the best solutions from the Pareto front. The effectiveness of VAMOSa was tested on seven artificial and six real-life data sets. VAMOSa, as well as SSym-AMOSA, were capable to detect clusters with point symmetry.

Saha et al.^[67] proposed another fuzzy clustering technique based on Multi-objective Differential Evolution (MODE). This technique, called Multi-objective Modified Differential Evolution based Fuzzy Clustering (MOMoDEFC), encodes the cluster centers and optimizes XB index and FCM measure. The performance of MOMoDEFC was compared with that of multi-objective DE based fuzzy clustering, multi-objective GA based fuzzy clustering, single-objective DE based fuzzy clustering, and the fuzzy c-means algorithm. The performance of MOMoDEFC was tested on large image data sets

which showed that this technique can solve the data mining applications.

Liu et al.^[68] proposed an approach based on Multi-objective Invasive Weed Optimization algorithm (MOIWO) to solve the clustering problem in which the number of clusters was uncertain. MOIWO simultaneously optimized XB index and FCM measure. It adopted a variable length real-coded scheme in which strings encode cluster-centroids. To maintain the diversity of weeds, a mechanism called feedback update was introduced to update those individuals whose corresponding number of centroids had been eliminated in one generation. Finally, SI index was used to select the best solution. The performance of MOIWO was better than the other algorithms in terms of determining the correct number of clusters and the appropriate partition. However, in non-globular data sets, MOIWO was unable to select the correct clustering from the Pareto optimal solutions.

Similarly, Nanda and Panda^[69] proposed an automatic multi-objective clustering algorithm based on clonal selection algorithm called MOCLONAL. It used the operators of cloning, hypermutation, and re-selection of immune cells. Furthermore, K -means algorithm was incorporated to provide initial solutions, which are guided by MOCLONAL.

NSGA-II was adopted as the basis for Multi-objective Evolutionary Approach based on Soft Subspace Clustering (MOEASSC) proposed by Xia et al.^[70]. The projection similarity validity index ($PSVIndex$) was used, which assumed that data objects belonging to the same cluster were similar in relevant dimensions. MOEASSC was compared with four clustering algorithms by considering three artificial and thirteen real-life data sets. MOEASSC had better clustering accuracy than the other competitor algorithms and was scalable to large data sets.

Zhong and Zhang^[71] proposed an automatic fuzzy clustering method based on Adaptive Multi-objective Differential Evolution (AFCMDE) to optimize XB index and FCM measure by means of two-layer fuzzy clustering system. A self-adaptive strategy was employed to improve the efficiency of differential evolution and

reduce the complexity of high-dimensional data sets for clustering.

Nanda and Panda^[72] introduced a hybrid multi-objective algorithm based on Multi-objective Immunized Particle Swarm Optimization (MOIMPSO). This approach combined the operators from Clonal Selection Principle (CSP) and Particle Swarm Optimization (PSO). It automatically evaluated the cluster centers and simultaneously optimized the two objective functions to evaluate the effectiveness of solutions in the multi-objective domain. MOIMPSO provided single best solution from the Pareto optimal front, which mostly satisfied the user requirement. MOIMPSO provided more accurate results at low computational cost and performed better for overlapping clusters.

Abubaker et al.^[73] presented a hybrid multi-objective clustering algorithm based on Multi-objective Particle Swarm Optimization and Simulated Annealing (MOPSOSA). This algorithm combined the features from Particle Swarm Optimization (PSO) and Simulated Annealing (SA). MOPSOSA optimized simultaneously three objective functions such as (i) DB index, (ii) Sym index, and (iii) Conn index which was measured by relative neighborhood graph concept. The performance of MOPSOSA was superior over other algorithms and obtained the efficient number of clusters in a reasonable amount of time.

Saha and Bandyopadhyay^[74] developed a Generalized Automatic Clustering Algorithm in a Multi-objective framework (GenClustMOO). It optimized the three objective functions such as (i) the total compactness of the partitioning based on the Euclidean distance, (ii) the total symmetry of the clusters based on the point symmetry distance, and (iii) the cluster connectedness. For multi-objectivity, AMOSA algorithm was used to obtain a set of Pareto optimal solutions. GenClustMOO employed a mechanism to merge clusters from a variable number of global clusters. GenClustMOO was able to detect the correct number of clusters and the appropriate partitioning in data sets.

2.2 Review on Multi-objective based Feature Selection Techniques

Huang and Wang^[75] proposed GA based feature selection algorithm, which identified an optimal subset of features and optimized Support Vector Machine (SVM) kernel parameters. Their experimental analysis showed that the proposed algorithm outperforms traditional searching methods.

Muni et al.^[76] proposed a feature selection approach using multi-tree GP algorithm (GP_{mtfs}). For a given c -class problem, each classifier in GP_{mtfs} was assigned c trees. Two new crossover operations were introduced to suit the feature selection process. GP_{mtfs} achieved better results than the other methods^[77].

Hamdani et al.^[78] proposed a feature selection approach using NSGA-II. It is used to improve accuracy and minimize the features. However, the performance of proposed approach did not compare with other well-known feature selection algorithms^[79].

Zhu et al.^[80] presented an improved feature selection algorithm approach (WFFSA) using the nature-inspired algorithm. Ranking of features was done using GA which prioritized the features. GP relevance measure (GPRM) metric was used to evaluate the ranking of features in subsets. However, WFFSA had limited performance when the number of features was increased^[81].

Wang et al.^[82] designed a hybrid feature selection approach which was based on PSO and rough sets theory. The particle was allocated as the relevance degree between the class labels and features^[83]. However, it overruled the use of rough sets in feature selection problems due to high computational time.

Sivagaminathan et al.^[84] developed a hybrid approach based on Ant Colony Optimization (ACO) and Artificial Neural Networks (ANNs). The artificial neural network was used to figure out the classification performance. The proposed

algorithm selected a small number of features and thus able to achieve better classification performance than all features in most of the cases.

Lin et al.^[85] developed a feature selection approach to optimize the parameters in support vector machine and select the subset of features. It simultaneously determined the parameter values while discovering a subset of features without reducing SVM classification accuracy. The proposed approach performed better than the competitive algorithms in terms of classification accuracy rate.

Soto et al.^[86] proposed wrapper feature selection algorithm using NSGA-II and SPEA2. NSGA-II is one of the most widely used multi-objective optimization algorithm in various research domains. To validate the performance of proposed technique, four different learning algorithms were trained. This methodology was suitable for linear and non-linear QSAR/QSPR models and selection of subsets.

Guillén et al.^[87] developed a multi-objective approach that simultaneously optimized the parameters and selects the variables of Radial Basis Function Neural Networks (RBFNNs). The proposed algorithm allowed to obtain very accurate models using the most significant input variables.

Feature selection in binary problems of classification was implemented by Genetic Programming (GP) based filter approach by Neshatian and Zhang^[88]. This algorithm was able to find hidden relationships between subsets of features and target classes. It had been proved experimentally that the proposed algorithm improved the classification performance of classifiers while decreased their complexity. However, the proposed method was not appropriate for the problems where the best feature subset was expected to have a very large number of features.

A wrapper based multi-objective feature selection algorithm for customer churn prediction in telecommunications by using a modified NSGA-II was provided by Huang et al.^[89]. In this method, the three objectives, i.e., true positive rate, true negative rate, and the overall classification rate were used. The data set used for

the testing of this algorithm was churn prediction data set in telecommunications which resulted in a good performance for classification.

Esseghir et al.^[90] developed a filter-wrapper feature selection method using PSO, which utilized the strength of both filters and wrappers approaches. Each particle was encoded using a score value in the proposed approach and feature class dependency was also ensured using filter approach. The performance of the proposed approach was evaluated by using classification accuracy.

Unler and Murat^[91] developed the adaptive selection based wrapper feature selection algorithm. Feature importance or relevancy was not only articulated using likelihood multi-objective evolutionary algorithms but feature contribution in the overall classification was also considered as selection criteria. Experiments revealed that the approach outperformed tabu and scatter search algorithms.

Liu et al.^[92] developed a feature selection algorithm using Multi-Swarm Particle Swarm Optimization (MSPSO) algorithm to obtain the subset of features. Moreover, parameter optimization was performed in Support Vector Machine (SVM)^[93,94]. The proposed approach had shown promising results as compared to other competing algorithms.

2.3 Review on Multi-objective based Clustering and Feature Selection Techniques

In recent years, there were a few methods which use multi-objective optimization to solve the unsupervised feature selection problem. The unsupervised feature selection problem was solved using multi-objective wrapper algorithm as developed by Morita et al.^[95]. The multi-objective evolutionary algorithm such as Non-dominated Sorting Genetic Algorithm 2 (NSGA-II) was employed as the underlying optimization method. The number of features and Davies-Bouldin

index were the two objective functions related to clustering are deployed for the clustering task.

Saha et al.^[96] proposed a technique for simultaneously clustering and feature selection named as Semi-FeaClustMOO. It conceals the number of features and number of cluster centers in the form of a string. Further, the points were being assigned based on the features present in the string to different clusters using a point symmetry based distance^[97]. In order to determine the values of internal cluster validity indices, some distance computations had to be implemented on it. The final Pareto optimal front gathered a set of solutions having various feature combinations and cluster centers. The appropriate subset of features, proper number of clusters, and the proper partitioning were automatically determined.

Anusha^[98] proposed a clustering algorithm which was based on an optimization approach to optimize the several objectives simultaneously (MOP). The proposed technique employed the neighbourhood learning using a genetic algorithm to maximize accuracy and efficiency of the solution through the constraint feature selection. A new population was generated using Neighbourhood Learning using *K*-means Genetic Algorithm (NLMOGA) and a constrained feature selection was applied to each sub-population. This method was developed to determine a suitable or closest set of objects from the group objects which will improve the robustness of NLMOGA for different instances of MOPs. The effectiveness of this approach was evaluated with various real-life benchmark gene expression data sets.

Hoz et al.^[99] proposed a multi-objective feature selection approach for an unsupervised clustering procedure based on Growing Hierarchical Self-organising Maps (GHSOMs). In the network anomaly detection problem, this multi-objective approach makes it possible not only to differentiate between normal and anomalous traffic but also among different anomalies. The efficiency had been evaluated by using well-known data sets that contain extracted features and

labeled attacks from around two million connections.

Dutta^[100] proposed an evolutionary clustering algorithm for simultaneously clustering and feature selection. The feature sub-set selection criteria can improve the quality of clustering. It can also improve the understandability and scalability.

Saha^[101] proposed a new framework based on Multi-objective Optimization (MOO), namely FeaClusMOO, which was used to identify the accurate partitioning as well as the most relevant set of features from the data set. A newly developed multi-objective simulated annealing based optimization technique namely Archived Multi-objective Simulated Annealing (AMOSA) was used as the background strategy for optimization. The three objectives were optimized simultaneously using AMOSA in order to detect the appropriate subset of features, appropriate number of clusters, and appropriate partitioning. The effectiveness of the proposed FeaClusMOO in comparison with other clustering techniques had been evaluated on high-dimensional data sets.

Yuwono^[102] proposed a technique to find the automatic feature selection using multi-objective cluster optimization. A knowledge-discovery approach was developed for discovering characteristic features using Multi-objective Clustering Rapid Centroid Estimation (MOC-RCE). The proposed method had been tested on experimental fault data from the American Society of Heating, Refrigerating and Air-conditioning Engineers (ASHRAE) research project 1312-RP winter 2008 data set.

Table 2.1: Classification of multi-objective metaheuristic algorithms and its applications in data clustering and feature selection.

Year	Type	Algorithm	Author(s)	Approach
2001	PESA and PESA-II	Evolutionary	Corne et al. ^[103,104]	Clustering
2004	VIENNA	Evolutionary	Handl and Knowles ^[105]	Clustering
2005	MOCK-am	Evolutionary	Handl and Knowles ^[106]	Clustering
2007	MOCK	Evolutionary	Handl and Knowles ^[57]	Clustering
2007	MOGA	Evolutionary	Bandyopadhyay et al. ^[60]	Clustering
2008	MEPSO	Bio-inspired	Das et al. ^[107]	Clustering
2009	MOGA-SVM	Evolutionary	Maulik et al. ^[108]	Clustering
2009	MOACO	Bio-inspired	Santos et al. ^[109]	Clustering
2009	MODE	Evolutionary	Suresh et al. ^[110]	Clustering
2009	MOPSO	Bio-inspired	Paoli et al. ^[63]	Clustering
2009	MODEGA	Evolutionary	Kundu et al. ^[62]	Clustering
2010	MOSA	Physics	Saha and Bandyopadhyay ^[111,66]	Clustering
2010	NSGA-II	Evolutionary	Huang et al. ^[112]	Feature selection
2011	NSGA-II	Evolutionary	Mukhopadhyay and Maulik ^[113]	Clustering
2012	MOSSC	Evolutionary	Zhu et al. ^[114]	Clustering
2012	MOPSO	Bio-inspired	Xue et al. ^[115]	Clustering
2012	MOIWO	Bio-inspired	Liu et al. ^[68]	Clustering
2012	MOPSO	Bio-inspired	Ali et al. ^[116]	Clustering
2013	MOGA	Evolutionary	Vignolo et al. ^[117]	Feature selection
2013	MOSA	Physics	Saha and Bandyopadhyay ^[74]	Clustering
2013	MO-IMPESO	Bio-inspired	Nanda and Panda ^[72]	Clustering
2013	MOEA	Evolutionary	Dutta et al. ^[118]	Clustering and Feature selection
2014	MOEA	Evolutionary	Tan et al. ^[119]	Feature selection
2014	MOACO	Bio-inspired	Wu et al. ^[120]	Clustering
2014	MOSA	Physics	Saha et al. ^[121]	Feature selection
2014	MOEA	Evolutionary	Dutta et al. ^[100]	Clustering and Feature selection
2015	MO-PSOSA	Bio-inspired	Abubaker et al. ^[73]	Clustering
2015	MO-Cuckoo	Bio-inspired	George et al. ^[122]	Clustering
2016	MOGA-Feature	Evolutionary	Kashyap et al. ^[123]	Clustering and Feature selection
2017	ClusSMOG-II	Evolutionary	Heloulou et al. ^[124]	Clustering
2017	IMCPSO	Bio-inspired	Gong et al. ^[125]	Clustering
2017	ENORA	Evolutionary	Jiménez et al. ^[126]	Clustering
2017	PSO-FS	Bio-inspired	Zhang et al. ^[127]	Feature selection
2018	MOCDP	Evolutionary	Li et al. ^[128]	Clustering and Feature selection
2018	ADNSGA2-FCM	Evolutionary	Dong et al. ^[129]	Clustering

2.4 Research Gaps

Based on above literature, the following research gaps are identified :

(1) **Efficient multi-objective metaheuristic based technique**

The existing techniques use the repository and Pareto dominance concept to store the non-dominated solutions. However, these techniques suffer from algorithmic complexity and take a lot of time to converge. These techniques suffer from local optima. Hence, there is need to develop an efficient metaheuristic based technique to solve the multi-objective optimization problem.

(2) **Efficient feature selection/extraction technique**

The existing multi-objective metaheuristic algorithm did not provide satisfactory results due to redundant and noisy features present in the given data set. Feature selection technique plays a vital role to solve the above-mentioned problem. The feature selection technique not only reduce the dimensionality of the problem but also speed up the processing time. Therefore, there is an imperative need to establish novel feature selection technique for multi-objective metaheuristic technique.

(3) **Automatic clustering technique**

The another challenge of data clustering is to identify groups (or clusters) based on the characteristics of data points present in the given data set. In most of the real life application, it is impossible to detect the number of cluster in advance. Hence, there is a need to develop an automatic clustering technique which can determine the number of cluster during the run time.

(4) **Automatic clustering and feature selection technique**

Data clustering become more cumbersome when number of features and clusters

are unknown. Both of these greatly affect the efficiency of clustering techniques. If the number of features increases beyond a certain limit, then there may be a chance to increase the number of clusters. Hence, there is need to develop an automatic clustering and feature selection during simulation run.

2.5 Objectives

Based on above research gaps, the following research objectives are identified :

- (1) To design some novel multi-objective metaheuristic technique(s) for effective data clustering and feature selection.
- (2) To propose a technique for automatic data clustering using multi-objective metaheuristic approaches.
- (3) To develop a technique for simultaneous clustering and feature selection based on multi-objective metaheuristic approaches.
- (4) To apply and validate the proposed techniques on some engineering applications.

2.6 Research Methodology

The main intention of this research is to propose a novel multi-objective metaheuristic based clustering and feature selection techniques. The general methodology utilized in this regard is given below (see Fig. [2.1](#)):

Phase 1: Data pre-processing techniques

The removal of noisy and irregularity from the given data set will be done in this phase. For this purpose, data pre-processing technique is used and compared with the other existing techniques.

Phase 2: Feature selection/extraction

In this phase, the important features will be extracted/selected from the given data set. This phase play an important role in determining the number of clusters in given data set. It can also reduce the time and space complexity of data clustering technique. For this purpose, a novel multi-objective metaheuristic based feature selection technique will be used and compared with the recently developed techniques.

Phase 3: Cluster estimation using metaheuristic based technique

The number of cluster in a given data set is depend upon the number of features and data points exist in the given data set. After selecting the important features from Phase 2, the number of clusters will be determined using multi-objective metaheuristic technique during run time.

Phase 4: Cluster formation using metaheuristic based technique

In this phase, data points will be assigned to number of clusters determined from Phase 3. The assignment of data points will be done through cluster center point and symmetry based distance measure. A novel multi-objective metaheuristic based clustering technique is used. The cluster formulation can be further refined through novel developed multi-objective metaheuristic technique.

Phase 5: Cluster validation using validity indices

A novel cluster validity indices will be used to optimize and validate the clusters formed through multi-objective metaheuristic based clustering technique which is proposed in Phase 4.

Phase 6: **Interpretation of the results**

The evaluation of proposed approaches will be done using well known clustering metrics such as inter-cluster distance and intra-cluster distance. The proposed approaches will be tested on real-life engineering problems.

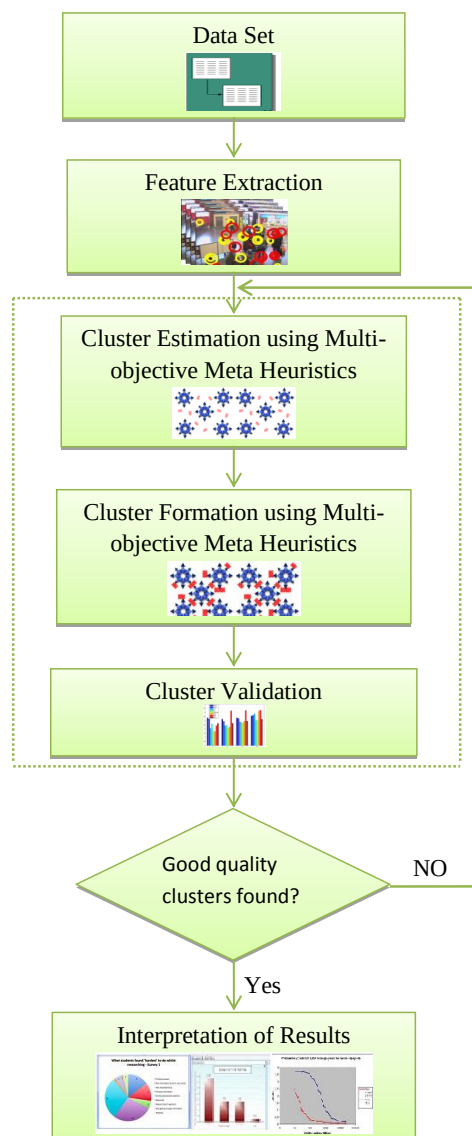


Fig. 2.1. General Methodology for Data Clustering.

Chapter3

Bio-inspired based Metaheuristic Optimization Techniques

“The purpose of computing is insight, not numbers.”

By Richard Hamming

3.1 Introduction

Most of the metaheuristic algorithms are based on the collective behavior of social creatures. The collective intelligence is inspired by the interaction of swarm with each other and their environment. Generally, these are easier to implement than evolutionary based algorithms due to fewer operators (i.e., selection, crossover, and mutation). The main components of metaheuristic algorithms are exploration and exploitation^[14,130]. Exploration ensures the algorithm to reach different promising regions of the search space, whereas exploitation ensures the searching for optimal solutions within the promising region^[23]. The fine tuning of these components is required to achieve the optimal solution for a given problem. However, an optimizer is able to solve a particular set of problem. The main motivation of this chapter is to develop novel metaheuristic algorithms for solving engineering design problems.

3.2 Single-objective Metaheuristic Algorithms

In this section, two novel metaheuristics namely Spotted Hyena Optimizer (SHO) and Emperor Penguin Optimizer (EPO) are proposed.

3.2.1 Spotted Hyena Optimizer (SHO)

Hyenas are large dog-like carnivores. They live in savannas, grasslands, sub-deserts, and forests of both Africa and Asia. There are four known species of hyena. These are spotted, striped, brown, and aardwolf. They differ in size, behavior, and type of diet. These species have a bear-like attitude as the front legs are longer than the back legs. Spotted hyenas are skillful hunters and largest among three hyena species (i.e., striped, brown, and aardwolf). They are so called because there are spots on their fur which is reddish brown in color with black spots. Spotted hyenas are complicated, intelligent, and highly social animals with really dreadful reputation. They have the ability to fight endlessly over territory and food. In spotted hyena's family, female members are dominant and live in their clan. However, male members leave their clan when they become adult to search and join a new clan. In this new family, they are lowest ranking members to get their share of meal. A male member who has joined the clan always stays with the same members (friends) for a long time. Whereas, a female is always assured of a stable place. An interesting fact about spotted hyenas is that they produce a sound alert to communicate with each other when a new food source is found.

According to Ilany *et al.* ^[131] spotted hyenas usually live and hunt in groups, rely on a network of trusted friends have more than 100 members. Spotted hyenas are social animals that can communicate with each other through specialized calls such as postures and signals. They use multiple sensory procedures to recognize

their kin and other individuals. They can also recognize third party kin and rank the relationships between their clan mates during social decision making. The spotted hyena track prey by sight, hearing, and smell. Fig. 3.1 shows the tracking, chasing, encircling, and attacking mechanism of spotted hyenas. Cohesive clusters are helpful for an efficient co-operation between spotted hyenas and also maximize the fitness. The hunting technique and social relation of spotted hyenas are mathematically modeled.

3.2.1.1 Mathematical Modeling

The mathematical modeling of searching, encircling, hunting, and attacking prey is provided. Thereafter, SHO algorithm is presented.

3.2.1.1.1 Encircling Prey

Spotted hyenas can be familiar with the location of prey and encircle them. To mathematically model the social hierarchy of spotted hyenas, the current best candidate solution is considered as the target prey or objective. The other search agents will try to update their positions according to the best optimal candidate solution. The mathematical model of this behavior is represented by the following equations:

$$\vec{D}_h = | \vec{B} \cdot \vec{P}_p(x) - \vec{P}(x) | \quad (3.1)$$

$$\vec{P}(x+1) = \vec{P}_p(x) - \vec{E} \cdot \vec{D}_h \quad (3.2)$$

where $\vec{D}_h$ represents the distance between the prey and spotted hyena, x indicates the current iteration, $\vec{B}$ and $\vec{E}$ are co-efficient vectors, $\vec{P}_p$ indicates the position vector of prey, $\vec{P}$ is the position vector of spotted hyena. However, $||$ and $\cdot$ is the absolute value and multiplication with vectors, respectively.

$\vec{B}$ and $\vec{E}$ are calculated as follows:

$$\vec{B} = 2 \cdot r\vec{d}_1 \quad (3.3)$$

$$\vec{E} = 2\vec{h} \cdot r\vec{d}_2 - \vec{h} \quad (3.4)$$

$$\vec{h} = \left(5 - \left(\text{Iteration} * \left(\frac{5}{\text{MaxIteration}} \right) \right) \right) \quad (3.5)$$

where, $\text{Iteration} = 1, 2, 3, \dots, \text{MaxIteration}$

For proper balancing between exploration and exploitation, $\vec{h}$ is linearly decreased from 5 to 0 during the course of iterations. Further, this mechanism promotes more exploitation as the iteration value increases. However, $r\vec{d}_1$ and $r\vec{d}_2$ are random vectors that lie in the range of [0, 1]. Fig. 3.2 shows the effects of Eqs. (3.1) and (3.2) in two-dimensional environment. In this figure, the spotted hyena (A, B) can update their position towards the position of prey (A^*, B^*). By adjusting the value of vectors $\vec{B}$ and $\vec{E}$, there are a different number of places which can be reached about the current position. The probably updated positions of a spotted hyena in 3D environment are shown in Fig. 3.5. By using Eqs. (3.1) and (3.2), a spotted hyena can update its position randomly around the prey. Therefore, the same concept can further be extended with n -dimensional search space.

3.2.1.1.2 Hunting

Spotted hyenas usually live and hunt in groups and rely on a network of trusted friends. They have the ability to recognize the location of prey. To define the hunting behavior of spotted hyenas, the best search agent has knowledge about the location of prey. The other search agents make a cluster of trusted friends towards the best search agent and saved the best solutions obtained so far to update their positions. The following equations are proposed in this mechanism:

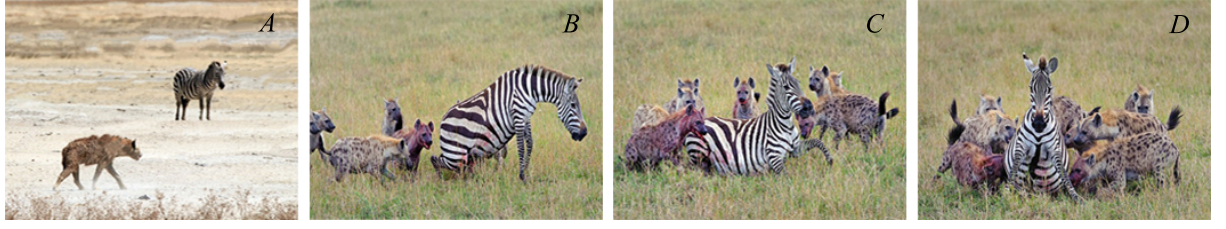


Fig. 3.1. Hunting behavior of spotted hyenas: (A) Searching and tracking; (B) Chasing; (C) Troublesome and encircling; (D) Immobile situation and attack on prey.

$$\vec{D}_h = | \vec{B} \cdot \vec{P}_h - \vec{P}_k | \quad (3.6)$$

$$\vec{P}_k = \vec{P}_h - \vec{E} \cdot \vec{D}_h \quad (3.7)$$

$$\vec{C}_h = \vec{P}_k + \vec{P}_{k+1} + \dots + \vec{P}_{k+N_s} \quad (3.8)$$

Here, N_s defines the number of spotted hyenas which can be calculated as follows:

$$N_s = \text{count}_{nos}(\vec{P}_h, \vec{P}_{h+1}, \vec{P}_{h+2}, \dots, (\vec{P}_h + \vec{M})) \quad (3.9)$$

$$\vec{P}(x+1) = \frac{\vec{C}_h}{N_s} \quad (3.10)$$

where $\vec{M}$ is a random vector that lies in the range of $[0.5, 1]$, nos defines the number of solutions and count all candidate solutions, after addition of $\vec{M}$, which are far similar to the best optimal solution. $\vec{C}_h$ is a group or cluster of N_s number of optimal solutions. $\vec{P}(x+1)$ saves the best optimal solution which is obtained so far and update the positions of other search agents.

3.2.1.1.3 Attacking Prey (Exploitation)

The variation in vector $\vec{E}$ is decreased to change the value in vector $\vec{h}$ which can decrease from 5 to 0 over the course of iterations. It is observed from Fig. 3.4 that $|\vec{E}| < 1$ forces the group of spotted hyenas to assault towards the prey. The mathematical formulation for attacking the prey is as follows:

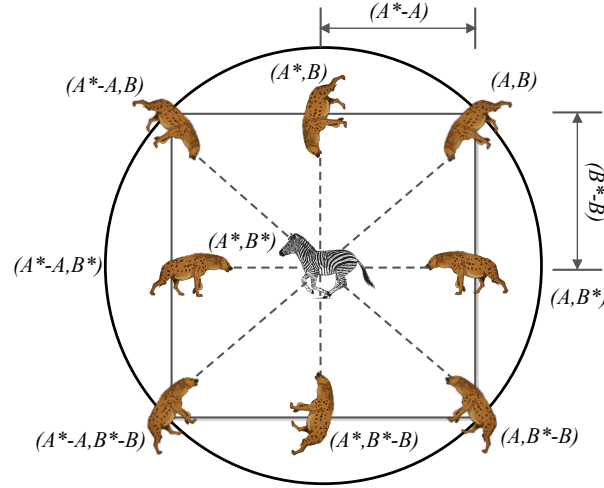


Fig. 3.2. 2D position vectors of spotted hyena.

$$\vec{P}(x+1) = \frac{\vec{C}_h}{N_s} \quad (3.11)$$

where $\vec{P}(x+1)$ save the best solution and updates the positions of other search agents. SHO algorithm allows its search agents to update their position and attack towards the prey.

3.2.1.1.4 Search for Prey(*Exploration*)

Spotted hyenas mostly search the prey according to the position of group or cluster of spotted hyenas which reside in vector $\vec{C}_h$. They move away from each other to search and attack the prey. Therefore, $\vec{E}$ with random values, which are greater than 1 or less than 1, are used to force the search agents to move far away from the prey. This mechanism allows SHO algorithm to search globally. To find prey, Fig. 3.3 shows that $|\vec{E}| > 1$ facilitates the spotted hyenas to move away from the prey. Another constituent of SHO algorithm which makes possible for exploration is $\vec{B}$. In Eq. (3.3), $\vec{B}$ contains random values which provide the random weights of prey. To show the random behavior of SHO algorithm, assume $\vec{B} > 1$ precedence than $\vec{B} < 1$ to demonstrate the effect in the distance as seen in Eq. (3.3). This will helpful for exploration and local optima avoidance. Depending on the position of

spotted hyenas, it can randomly decide a weight to the prey and possibly makes it rigid or beyond to reach for spotted hyenas. $\vec{B}$ is required to provide random values for exploration. This mechanism is very helpful to avoid local optima problem. Finally, SHO algorithm is stopped by satisfying termination criteria.

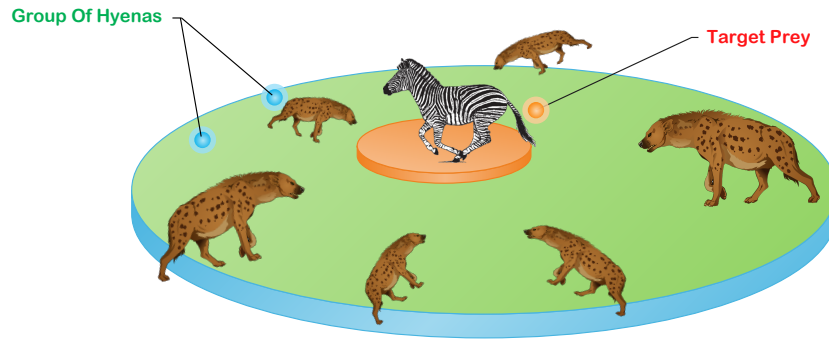


Fig. 3.3. Searching for prey ($|\vec{E}| > 1$).

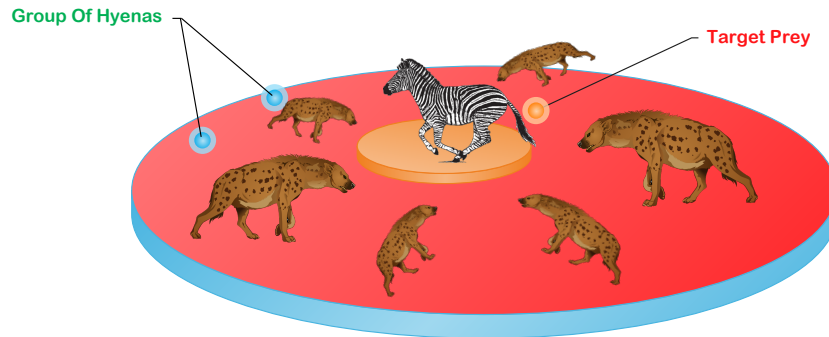


Fig. 3.4. Attacking for prey ($|\vec{E}| < 1$).

- The proposed algorithm saves the best solutions obtained so far over the course of iterations.

- around the solutions which can be extended to higher dimensions.

- with different random positions.

- probable position of the prey.

- vectors $\vec{E}$ and $\vec{h}$.

- ($|\vec{E}| \geq 1$) and another half of iterations are devoted to hunting (exploitation) ($|\vec{E}| \leq 1$).

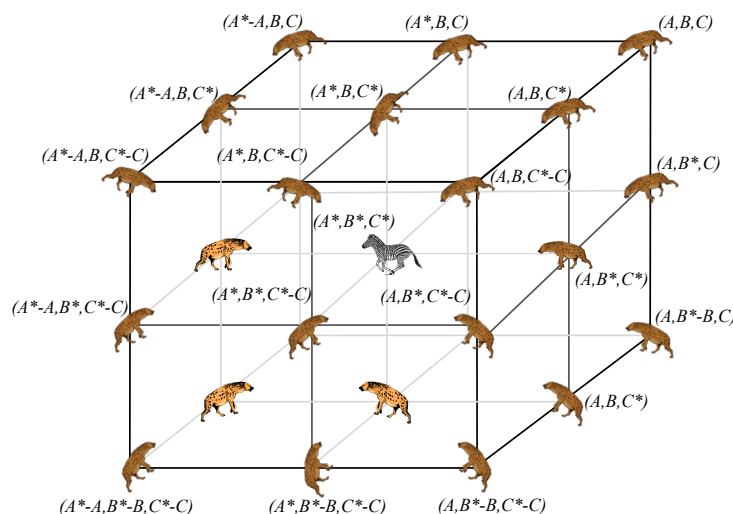


Fig. 3.5. 3D position vectors and possible next locations of spotted hyena.

3.2.1.2 Steps of SHO Algorithm

The steps of SHO are summarized as follows:

Step 1: Initialize the spotted hyenas population $\vec{P}_i$, where $i = 1, 2, \dots, n$.

Step 2: Choose the initial parameters of SHO namely $\vec{h}, \vec{B}, \vec{E}$, and N_s .

Step 3: Calculate the fitness value of each search agent.

Step 4: The best search agent is explored in the given search space.

Step 5: Define the group of optimal solutions, i.e., cluster using Eqs. (3.8) and (3.9).

Step 6: Update the positions of search agents using Eq. (3.11).

Step 7: Check whether any search agent goes beyond the boundary in the given search space and adjust it. It is done by adding and subtracting the values of search agents in such a way so that the solution not goes beyond the region of a search space.

Step 8: Compute the fitness value of search agent and update the vector $\vec{P}_h$ if there is a better solution than the previous optimal solution.

Step 9: Update the group of spotted hyenas $\vec{C}_h$ according to updated search agents.

Step 10: If the stopping criterion is satisfied, then the algorithm will be stopped. Otherwise, goto Step 5.

Step 11: Return the best optimal solution after the stopping criterion is satisfied, which is obtained so far.

Algorithm 1 Spotted Hyena Optimizer (SHO)

Input: Spotted hyenas population $\vec{P}_i$ ($i = 1, 2, \dots, n$)
Output: Best search agent

- 1: **procedure** SHO
- 2: Initialize the parameters $\vec{h}, \vec{B}, \vec{E}$, and N_s
- 3: Calculate the fitness of each search agent
- 4: $\vec{P}_h \leftarrow$ the best search agent
- 5: $\vec{C}_h \leftarrow$ the group or cluster of all far optimal solutions
- 6: **while** ($x < Max_{Iteration}$) **do**
- 7: **for** each search agent **do**
- 8: Update the position of current agent by Eq. (3.11)
- 9: **end for**
- 10: Update $\vec{h}, \vec{B}, \vec{E}$, and N_s
- 11: Check if any search agent goes beyond the given search space and then adjust it
- 12: Calculate the fitness of each search agent
- 13: Update $\vec{P}_h$ if there is a better solution than previous optimal solution
- 14: Update the group $\vec{C}_h$ w.r.t $\vec{P}_h$
- 15: $x \leftarrow x + 1$
- 16: **end while**
- 17: **return** $\vec{P}_h$
- 18: **end procedure**

3.2.1.3 Computational Complexity

The computational complexity of the proposed algorithm is discussed. Both the time and space complexities of the proposed algorithm is given below.

3.2.1.3.1 Time Complexity

1. Initialization of SHO population needs $\mathcal{O}(n \times dim)$ time, where n indicates the number of search agents and dim indicates the dimension of a test function.
2. The computation of fitness of each search agent requires $\mathcal{O}(Max_{Iteration} \times f_n \times pos)$ time, where $Max_{Iteration}$ is the maximum number of iterations to simulate the proposed algorithm, f_n is the test function, and pos is the position of a spotted hyena.

3. The group of spotted hyenas requires $\mathcal{O}(Max_{Iteration} \times N_c)$ time, where N_c is the counting value of spotted hyenas.
4. Steps 2 - 3 are repeated until the termination criterion is satisfied, which needs $\mathcal{O}(n)$ time.

Therefore, the overall time complexity of SHO algorithm is $\mathcal{O}(n \times f_n \times pos \times N_c \times Max_{Iteration})$.

The computational complexities of GWO, PSO, MFO, MVO, and SCA algorithms are $\mathcal{O}(N \times Max_{iterations} \times n_o \times n_p \times o_f)$. The GSA algorithm requires $\mathcal{O}(f^2)$ time for calculating the force component and thus, overall requires $\mathcal{O}(N \times f^2 \times Max_{iterations} \times n_o \times n_p \times o_f)$. The computational complexity of GA algorithm is $\mathcal{O}(N \times Max_{iterations} \times n_o \times n_p \times o_f \times (c_s + m_t))$, where c_s and m_t indicate the crossover and mutation operators, respectively. Finally, the HS algorithm requires $\mathcal{O}(N \times Max_{iterations} \times n_o \times n_p \times o_f \times BW)$, where BW is bandwidth and one of the deciding factor for better exploration and exploitation.

3.2.1.3.2 Space Complexity

The space complexity of SHO algorithm is the maximum amount of space used at any one time which is considered during its initialization process. Thus, the total space complexity of SHO algorithm is $\mathcal{O}(n \times dim)$.

3.2.1.4 Experimental Results and Discussions

This section describes the experimentation to evaluate the performance of proposed algorithm. Twenty-nine standard benchmark test functions are utilized for evaluating the performance of proposed algorithm. The results are evaluated and compared with eight well-known metaheuristic algorithms.

3.2.1.4.1 Benchmark Test Functions and Algorithms used for Comparisons

The twenty-nine benchmark functions are applied on proposed algorithm to demonstrate its efficiency. These functions are divided into four main categories such as Unimodal^[132], Multimodal^[133], Fixed-dimension Multimodal^[132,133], and Composite functions^[134]. These are described in Appendix A. In Appendix A, *Dim* and *Range* indicate the dimension of function and boundary of search space, respectively.

Tables A.1 and A.2 show the characteristics of unimodal and multimodal benchmark functions, respectively. The description of fixed-dimension modal and composite test functions are tabulated in Tables A.3 and A.4, respectively. The seven test functions ($F_1 - F_7$) are unimodal test functions. There is only one global optimum and has no local optima in the first group of test functions which make them highly appropriate for analyzing the convergence speed and exploitation ability of an algorithm. The second category consists of nine test functions ($F_8 - F_{16}$). The third category includes ten test functions ($F_{14} - F_{23}$). In the second and third category of test functions, there are multiple local solutions besides the global optimum which are useful for examining the local optima avoidance and the explorative ability of an algorithm. The fourth category comprises six composite test functions ($F_{24} - F_{29}$). These composite benchmark functions are the shifted, rotated, expanded, and combined version of classical functions^[135]. All of these benchmark test functions are minimization problems.

To validate the performance of the proposed algorithm, the eight well-known algorithms are chosen for comparison. These are Grey Wolf Optimizer (GWO)^[136], Particle Swarm Optimization (PSO)^[27], Moth-Flame Optimization (MFO)^[137], Multi-Verse Optimizer (MVO)^[138], Sine Cosine Algorithm (SCA)^[139], Gravitational Search Algorithm (GSA)^[140], Genetic Algorithm (GA)^[141], and Harmony Search (HS)^[142]. The number of search agents for the above-mentioned algorithms are set according to the literature.

Consequently, most of the researchers have employed these algorithms for performance evaluation of their work. Based on these grounds, we too choose these algorithms for demonstrating the efficiency of the proposed algorithm.

3.2.1.4.2 *Experimental Setup*

The parameter settings of proposed SHO and the above-mentioned algorithms are mentioned in Table 3.1. All of these parameters are set according to the reported literature. The same initialization technique is used to compare SHO with the above-mentioned metaheuristic algorithms. The experimentation and algorithms are implemented in MATLAB R2014a (8.3.0.532) version and run it in the environment of Microsoft Windows 8.1 with 64 bits on Core i5 processor with 2.40 GHz and 4 GB memory. The average and standard deviation results of objective functions values are in normal and italic forms, respectively. Whereas, the bold results represent that a particular algorithm is best as compared to other algorithms on employed benchmark test function. For each benchmark function, SHO algorithm utilizes 30 independent runs where each run employs 1000 times of iterations.

3.2.1.4.3 *Performance Comparison*

In order to demonstrate the performance of proposed algorithm, its results are compared with eight well-known metaheuristic algorithms on the above-mentioned benchmark test functions.

- Evaluation of functions $F_1 - F_7$

The functions $F_1 - F_7$ are unimodal and allow to assess the exploitation capability of the metaheuristic based algorithms. Table 3.2 shows the results obtained from SHO and other algorithms for unimodal test functions. It shows SHO is very competitive as compared to the other reported methods.

In particular, SHO is the most efficient algorithm for F_1, F_2, F_3, F_5 , and F_7 functions. The proposed SHO algorithm provides best exploitation than the others.

- Evaluation of functions $F_8 - F_{23}$

Tables 3.3 and 3.4 show the results obtained from SHO and other metaheuristics for functions $F_8 - F_{23}$ (multimodal and fixed-dimension multimodal functions) which can indicate that SHO has good exploration capability. According to these tables, SHO is most efficient in seven test problems (i.e., $F_9, F_{11}, F_{15}, F_{16}, F_{17}, F_{18}$, and F_{22}). In fact, SHO algorithm is the most efficient best algorithm in the majority of these test problems. These results show that SHO algorithm has better exploration capability than the other metaheuristics.

- Evaluation of functions $F_{24} - F_{29}$

The optimization of composite benchmark functions is a very competitive task because of balance between exploration and exploitation which can keep away from the local optima problem. Table 3.5 shows that SHO algorithm was the best optimizer for functions F_{26}, F_{28} , and F_{29} and very competitive in the other cases.

The results for functions $F_1 - F_{29}$ show that SHO is an efficient algorithm than the other optimization methods.

3.2.1.4.4 Convergence Analysis

The intention behind the convergence analysis is to increase the better understanding of different explorative and exploitative mechanisms in SHO algorithm. The best run convergence curves of SHO and other algorithms are provided and compared in Fig. 3.6. It can be observed that SHO is best competitive among the other existing metaheuristic algorithms. When optimizing the test functions, SHO shows three different convergence behaviors. In the initial steps

of iterations, SHO converges more rapidly for the promising regions of search space due to its adaptive mechanism. This behavior is apparent in test functions F_1, F_3, F_9, F_{11} , and F_{26} . In the second step, SHO converges towards the optimum only in final iterations which are shown in F_7 test function. The last step is the express convergence from the initial stage of iterations as seen in F_2 and F_{17} . These results show that SHO algorithm maintains a balance between exploration and exploitation to find the global optima.

3.2.1.4.5 Scalability Study

The next experiment is performed to see the effect of scalability on various benchmark test functions. The dimensionality of the test functions is changed by different number of ranges. As shown in Fig. 3.7, the results indicate that the performance of SHO algorithm provides different behaviors on various dimensionality. The performance of the proposed algorithm is degraded with the increase in the dimensionality as depicted in Fig. 3.7. It is also observed that the degradation in the performance is not too much. Therefore, it shows the applicability of proposed algorithm on a highly scalable environment.

3.2.1.4.6 Statistical Testing

Besides the basic statistical analysis, ANOVA test has been conducted for comparison of above-mentioned algorithms. ANOVA test has been used to test whether the results obtained from proposed algorithms differ from the results of other competitive algorithms in a statistically significant way. The sample size for ANOVA test takes as 30 with 95% confidence of interval. The result analysis of ANOVA test for benchmark functions is tabulated in Table 3.6. The results reveal that SHO is statistically significant as compared to other competitor algorithms.

Table 3.1: Parameters settings of algorithms.

Algorithms	Parameters	Values
<i>For all algorithms</i>	Number of Generations	1000
Spotted Hyena Optimizer (SHO)	$\vec{h}$ Constant	[5, 0]
	$\vec{M}$ Constant	[0.5, 1]
	$\vec{r}d_1, \vec{r}d_2$	[0, 1]
Emperor Penguin Optimizer (EPO)	Temperature Profile (T')	[1, 1000]
	$\vec{A}$ Constant	[-1.5, 1.5]
	Function $S()$	[0, 1.5]
	Parameter M	2
	Parameter f	[2, 3]
	Parameter l	[1.5, 2]
Grey Wolf Optimizer (GWO)	Alpha and Beta	0.9, 0.01
	Exploration Rate	0.1
Particle Swarm Optimization (PSO)	Inertia Coefficient	0.75
	Cognitive and Social Coeff	1.8, 2
Moth-Flame Optimization (MFO)	Convergence Constant	[-1, -2]
	Logarithmic Spiral	0.75
Multi-Verse Optimizer (MVO)	Max and Min WEP	1, 0.2
	Exploitation Accuracy	6
Sine Cosine Algorithm (SCA)	Fitness	1
	r_1 Constant	[2, 0]
Gravitational Search Algorithm (GSA)	Gravitational Constant	100
	Alpha Coefficient	20
Genetic Algorithm (GA)	Crossover	0.9
	Mutation	0.05
Harmony Search (HS)	Harmony Rate	0.95
	Neighbouring Value Rate	0.30
	Discrete Set and Fret Width	17700, 1

Table 3.2: Results of unimodal benchmark functions.

F	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_1	0.00E+00 (0.00E+00)	4.69E-59 (7.30E-59)	4.98E-09 (1.40E-08)	3.15E-04 (5.99E-04)	2.81E-01 (1.11E-01)	3.55E-02 (1.06E-01)	1.16E-16 (6.10E-17)	1.95E-12 (2.01E-11)	7.86E-10 (8.11E-09)
F_2	0.00E+00 (0.00E+00)	1.20E-34 (1.30E-34)	7.29E-04 (1.84E-03)	3.71E+01 (2.16E+01)	3.96E-01 (1.41E-01)	3.23E-05 (8.57E-05)	1.70E-01 (9.29E-01)	6.53E-18 (5.10E-17)	5.99E-20 (1.11E-17)
F_3	0.00E+00 (0.00E+00)	1.00E-14 (4.10E-14)	1.40E+01 (7.13E+00)	4.42E+03 (3.71E+03)	4.31E+01 (8.97E+00)	4.91E+03 (3.89E+03)	4.16E+02 (1.56E+02)	7.70E-10 (7.36E-09)	9.19E-05 (6.16E-04)
F_4	7.78E-12 (8.96E-12)	2.02E-14 (2.43E-14)	6.00E-01 (1.72E-01)	6.70E+01 (1.06E+01)	8.80E-01 (2.50E-01)	1.87E+01 (8.21E+00)	1.12E+00 (9.89E-01)	9.17E+01 (5.67E+01)	8.73E-01 (1.19E-01)
F_5	8.59E+00 (5.53E-01)	2.79E+01 (1.84E+00)	4.93E+01 (3.89E+01)	3.50E+03 (3.98E+03)	1.18E+02 (1.43E+02)	7.37E+02 (1.98E+03)	3.85E+01 (3.47E+01)	5.57E+02 (4.16E+01)	8.91E+02 (2.97E+02)
F_6	2.46E-01 (1.78E-01)	6.58E-01 (3.38E-01)	9.23E-09 (1.78E-08)	1.66E-04 (2.01E-04)	3.15E-01 (9.98E-02)	4.88E+00 (9.75E-01)	1.08E-16 (4.00E-17)	3.15E-01 (9.98E-02)	8.18E-17 (1.70E-18)
F_7	3.29E-05 (2.43E-05)	7.80E-04 (3.85E-04)	6.92E-02 (2.87E-02)	3.22E-01 (2.93E-01)	2.02E-02 (7.43E-03)	3.88E-02 (5.79E-02)	7.68E-01 (2.77E+00)	6.79E-04 (3.29E-03)	5.37E-01 (1.89E-01)

Table 3.3: Results of multimodal benchmark functions.

F	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_8	-1.16E+03 (2.72E+02)	-6.14E+03 (9.32E+02)	-6.01E+03 (1.30E+03)	-8.04E+03 (8.80E+02)	-6.92E+03 (9.19E+02)	-3.81E+03 (2.83E+02)	-2.75E+03 (5.72E+02)	-5.11E+03 (4.37E+02)	-4.69E+02 (3.94E+02)
F_9	0.00E+00 (0.00E+00)	4.34E-01 (1.66E+00)	4.72E+01 (1.03E+01)	1.63E+02 (3.74E+01)	1.01E+02 (1.89E+01)	2.23E+01 (3.25E+01)	3.35E+01 (1.19E+01)	1.23E-01 (4.11E+01)	4.85E-02 (3.91E+01)
F_{10}	2.48E+00 (1.41E+00)	1.63E-14 (3.14E-15)	3.86E-02 (2.11E-01)	1.60E+01 (6.18E+00)	1.15E+00 (7.87E-01)	1.55E+01 (8.11E+00)	8.25E-09 (1.90E-09)	5.31E-11 (1.11E-10)	2.83E-08 (4.34E-07)
F_{11}	0.00E+00 (0.00E+00)	2.29E-03 (5.24E-03)	5.50E-03 (7.39E-03)	5.03E-02 (1.74E-01)	5.74E-01 (1.12E-01)	3.01E-01 (2.89E-01)	8.19E+00 (3.70E+00)	3.31E-06 (4.23E-05)	2.49E-05 (1.34E-04)
F_{12}	3.68E-02 (1.15E-02)	3.93E-02 (2.42E-02)	1.05E-10 (2.06E-10)	1.26E+00 (1.83E+00)	1.27E+00 (1.02E+00)	5.21E+01 (2.47E+02)	2.65E-01 (3.14E-01)	9.16E-08 (4.88E-07)	1.34E-05 (6.23E-04)
F_{13}	9.29E-01 (9.52E-02)	4.75E-01 (2.38E-01)	4.03E-03 (5.39E-03)	7.24E-01 (1.48E+00)	6.60E-02 (4.33E-02)	2.81E+02 (8.63E+02)	5.73E-32 (8.95E-32)	6.39E-02 (4.49E-02)	9.94E-08 (2.61E-07)

Table 3.4: Results of fixed-dimension multimodal benchmark functions.

F	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_{14}	9.68E+00 (3.29E+00)	3.71E+00 (3.86E+00)	2.77E+00 (2.32E+00)	2.21E+00 (1.80E+00)	9.98E-01 (9.14E-12)	1.26E+00 (6.86E-01)	3.61E+00 (2.96E+00)	4.39E+00 (4.41E-02)	6.79E+00 (1.12E+00)
F_{15}	9.01E-04 (1.06E-04)	3.66E-03 (7.60E-03)	9.09E-04 (2.38E-04)	1.58E-03 (3.50E-03)	7.15E-03 (1.26E-02)	1.01E-03 (3.75E-04)	6.84E-03 (7.37E-03)	7.36E-03 (2.39E-04)	5.15E-03 (3.45E-04)
F_{16}	-1.06E+01 (2.86E-11)	-1.03E+00 (7.02E-09)	-1.03E+00 (0.00E+00)	-1.03E+00 (0.00E+00)	-1.03E+00 (4.74E-08)	-1.03E+00 (3.23E-05)	-1.03E+00 (0.00E+00)	-1.04E+00 (4.19E-07)	-1.03E+00 (3.64E-08)
F_{17}	3.97E-01 (2.46E-01)	3.98E-01 (7.00E-07)	3.97E-01 (9.03E-16)	3.98E-01 (1.13E-16)	3.98E-01 (1.15E-07)	3.99E-01 (7.61E-04)	3.98E-01 (1.13E-16)	3.98E-01 (3.71E-17)	3.99E-01 (9.45E-15)
F_{18}	3.00E+00 (9.05E+00)	3.00E+00 (7.16E-06)	3.00E+00 (6.59E-05)	3.00E+00 (4.25E-15)	5.70E+00 (1.48E+01)	3.00E+00 (2.25E-05)	3.01E+00 (3.24E-02)	3.01E+00 (6.33E-07)	3.00E+00 (1.94E-10)
F_{19}	-3.75E+00 (4.39E-01)	-3.86E+00 (1.57E-03)	-3.90E+00 (3.37E-15)	-3.86E+00 (3.16E-15)	-3.86E+00 (3.53E-07)	-3.86E+00 (2.55E-03)	-3.22E+00 (4.15E-01)	-3.30E+00 (4.37E-10)	-3.29E+00 (9.69E-04)
F_{20}	-1.44E+00 (5.47E-01)	-3.27E+00 (7.27E-02)	-3.32E+00 (2.66E-01)	-3.23E+00 (6.65E-02)	-3.23E+00 (5.37E-02)	-2.84E+00 (3.71E-01)	-1.47E+00 (5.32E-01)	-2.39E+00 (4.37E-01)	-2.17E+00 (1.64E-01)
F_{21}	-2.08E+00 (3.80E-01)	-9.65E+00 (1.54E+00)	-7.54E+00 (2.77E+00)	-6.20E+00 (3.52E+00)	-7.38E+00 (2.91E+00)	-2.28E+00 (1.80E+00)	-4.57E+00 (1.30E+00)	-5.19E+00 (2.34E+00)	-7.33E+00 (1.29E+00)
F_{22}	-1.61E+01 (2.04E-04)	-1.04E+01 (2.73E-04)	-8.55E+00 (3.08E+00)	-7.95E+00 (3.20E+00)	-8.50E+00 (3.02E+00)	-3.99E+00 (1.99E+00)	-6.58E+00 (2.64E+00)	-2.97E+00 (1.37E-02)	-1.00E+00 (2.89E-04)
F_{23}	-1.68E+00 (2.64E-01)	-1.05E+01 (1.81E-04)	-9.19E+00 (2.52E+00)	-7.50E+00 (3.68E+00)	-8.41E+00 (3.13E+00)	-4.49E+00 (1.96E+00)	-9.37E+00 (2.75E+00)	-3.10E+00 (2.37E+00)	-2.46E+00 (1.19E+00)

Table 3.5: Results of composite benchmark functions.

F	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_{24}	2.30E+02 (1.37E+02)	8.39E+01 (8.42E+01)	6.00E+01 (8.94E+01)	1.18E+02 (7.40E+01)	1.40E+02 (1.52E+02)	1.20E+02 (3.11E+01)	4.49E-17 (2.56E-17)	5.97E+02 (1.34E+02)	4.37E+01 (2.09E+01)
F_{25}	4.08E+02 (9.36E+01)	1.48E+02 (3.78E+01)	2.44E+02 (1.73E+02)	9.20E+01 (1.36E+02)	2.50E+02 (1.44E+02)	1.14E+02 (1.84E+00)	2.03E+02 (4.47E+02)	4.09E+02 (2.10E+01)	1.30E+02 (3.34E+00)
F_{26}	3.39E+02 (3.14E+01)	3.53E+02 (5.88E+01)	3.39E+02 (8.36E+01)	4.19E+02 (1.15E+02)	4.05E+02 (1.67E+02)	3.89E+02 (5.41E+01)	3.67E+02 (8.38E+01)	9.30E+02 (8.31E+01)	5.07E+02 (4.70E+01)
F_{27}	7.26E+02 (1.21E+02)	4.23E+02 (1.14E+02)	4.49E+02 (1.42E+02)	3.31E+02 (2.09E+01)	3.77E+02 (1.28E+02)	4.31E+02 (2.94E+01)	5.32E+02 (1.01E+02)	4.97E+02 (3.24E+01)	3.48E+02 (2.07E+02)
F_{28}	1.06E+02 (1.38E+01)	1.36E+02 (2.13E+02)	2.40E+02 (4.25E+02)	1.13E+02 (9.27E+01)	2.45E+02 (9.96E+01)	1.56E+02 (8.30E+01)	1.44E+02 (1.31E+02)	1.90E+02 (5.03E+01)	8.43E+02 (6.94E+02)
F_{29}	5.97E+02 (4.98E+00)	8.26E+02 (1.74E+02)	8.22E+02 (1.80E+02)	8.92E+02 (2.41E+01)	8.33E+02 (1.68E+02)	6.06E+02 (1.66E+02)	8.13E+02 (1.13E+02)	6.65E+02 (3.37E+02)	7.37E+02 (2.76E+01)

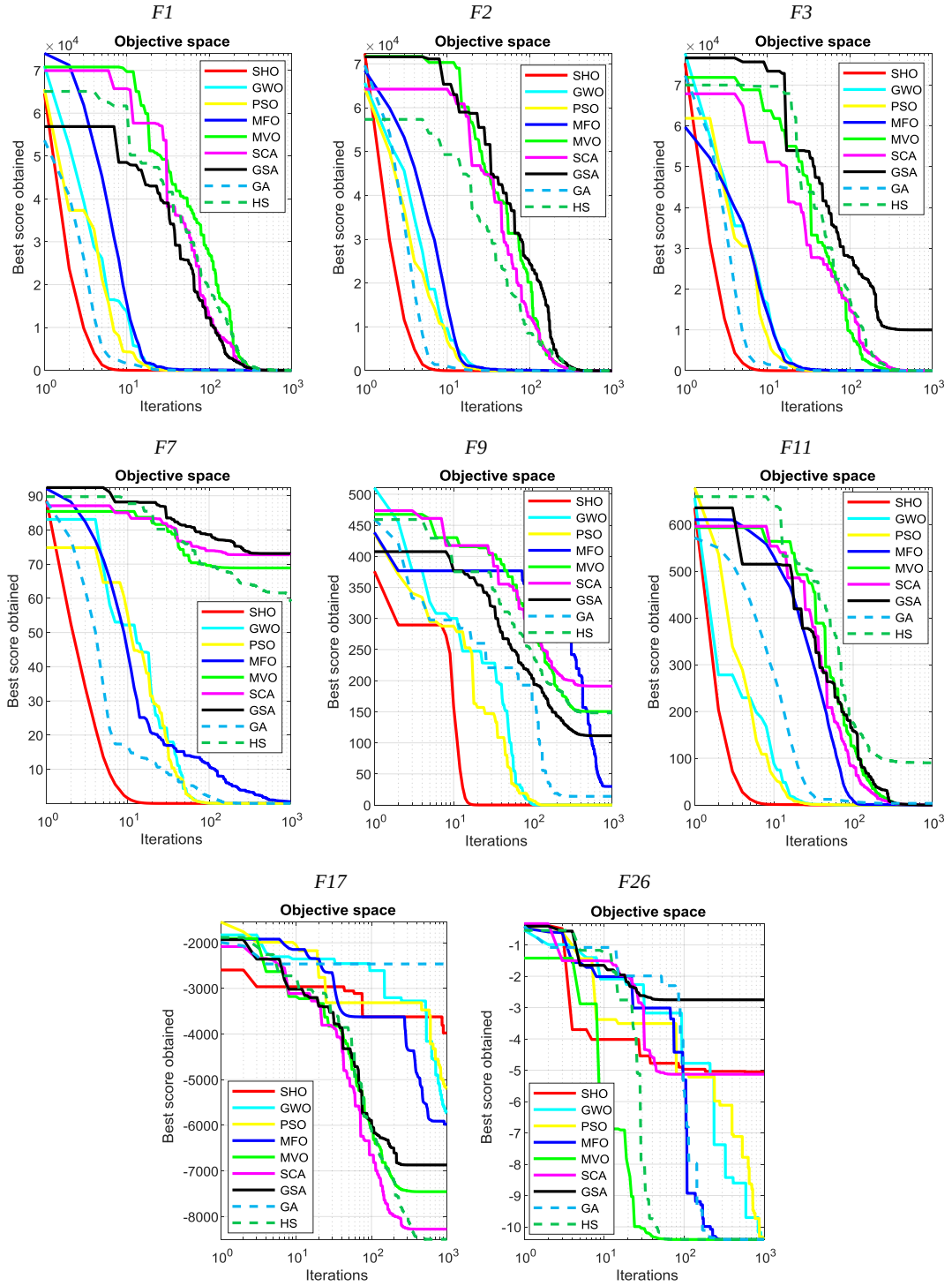


Fig. 3.6. The convergence curves of SHO and the existing metaheuristics on benchmark test problems.

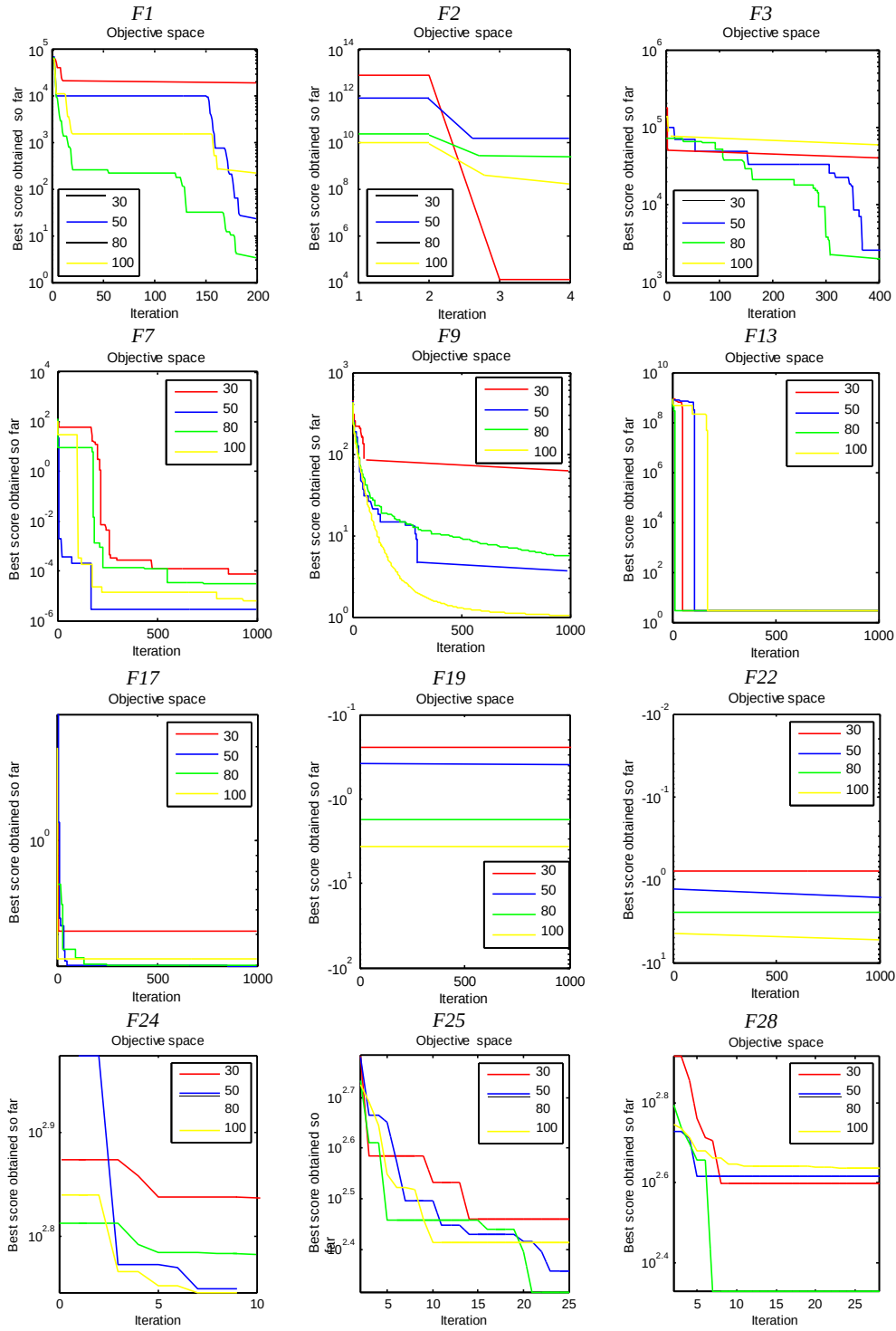


Fig. 3.7. Effect of scalability on the performance of SHO algorithm.

Table 3.6: ANOVA test results.

F	p -value	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_1	1.92E-74	GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, GA	SHO, GWO, PSO, MFO, MVO, SCA, GSA, GA
F_2	3.61E-72	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, GSA, GA, HS	SHO, GWO, MFO, SCA, GSA, GA, HS	SHO, GWO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA
F_3	3.56E-38	GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA	SHO, GWO, PSO, MVO, SCA, GSA, GA	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MVO, SCA, GSA, GA
F_4	2.85E-174	GWO, PSO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, HS	SHO, PSO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, SCA, GSA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA
F_5	2.16E-22	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, HS	SHO, GWO, PSO, SCA, GSA, HS	SHO, PSO, MFO, SCA, GSA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, MFO, MVO, SCA
F_6	1.97E-147	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, SCA, GSA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, HS	SHO, PSO, MFO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, MFO, MVO, SCA, GA
F_7	1.53E-01	GWO, PSO, MFO, MVO, SCA, GSA, GA	SHO, MFO, MVO, SCA, GA, HS	SHO, MFO, MVO, SCA, GSA, GA, HS	SHO, GWO, MVO, MVO, SCA, GA, HS	SHO, GWO, MFO, SCA, GSA, GA, HS	SHO, GWO, MFO, SCA, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, MFO, SCA, GSA, GA
F_8	1.56E-122	GWO, PSO, MFO, MVO, SCA, HS	SHO, MFO, MVO, SCA, GSA, HS	SHO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, PSO, MFO, MVO, SCA, GSA, MFO, SCA, GSA, GA
F_9	1.87E-112	GWO, PSO, MFO, GSA, GA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, SCA, GSA, GA	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, GA
F_{10}	1.28E-86	GWO, PSO, MFO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA, HS	SHO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA	SHO, GWO, PSO, MFO, MVO, GSA, GA	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, GA
F_{11}	1.86E-77	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, MFO, SCA, GSA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, GA	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA

(Continued...)

F	p -value	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA	HS
F_{12}	2.39E-01	GWO, PSO, MVO, SCA, GSA, HS	SHO, PSO, SCA, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MVO, SCA, GA, HS	SHO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, GWO, PSO, SCA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA
F_{13}	1.90E-03	PSO, MFO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA	SHO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MVO, SCA, GSA, GA, HS	SHO, GWO, MFO, SCA, GSA, HS	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA	SHO, GWO, MVO, SCA, GSA, GA
F_{14}	6.37E-47	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA	SHO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA
F_{15}	1.15E-05	GWO, PSO, MVO, SCA, GSA, GA	SHO, PSO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, GSA, GA, HS	SHO, PSO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA	SHO, GWO, PSO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA
F_{16}	2.66E-40	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{17}	1.81E-12	GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, GA
F_{18}	4.60E-02	MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA	SHO, GWO, PSO, MVO, SCA, GSA, GA	SHO, GWO, MFO, SCA, GSA, HS	SHO, PSO, MFO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, HS	SHO, GWO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GA
F_{19}	9.51E-45	GWO, PSO, MFO, MVO, SCA, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GSA, GA	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA	SHO, GWO, MFO, MVO, GSA, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, SCA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GA
F_{20}	1.97E-98	GWO, MFO, MVO, SCA, GSA, GA, HS	SHO, MFO, MVO, SCA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA, HS	SHO, PSO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA, HS	SHO, PSO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, MFO, MVO, GSA, HS	SHO, GWO, MFO, MVO, SCA, GSA, GA
F_{21}	2.55E-31	GWO, PSO, MFO, MVO, SCA, GA, HS	SHO, PSO, MVO, SCA, GSA, GA, HS	SHO, GWO, MFO, MVO, SCA, GSA	SHO, GWO, PSO, MVO, SCA, GSA, HS	SHO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MVO, GSA, GA, HS	SHO, GWO, PSO, MVO, MVO, SCA, HS	SHO, GWO, PSO, MFO, MVO, SCA, GSA	SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{22}	8.10E-31	GWO, PSO, MFO, MVO, SCA, GSA, GA	SHO, PSO, MFO, SCA, GSA, GA, HS	SHO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, SCA, GA, HS	SHO, PSO, MFO, SCA, GSA, HS	SHO, GWO, PSO, MFO, MVO, GSA, GA	SHO, GWO, MFO, MVO, SCA, GA, HS	SHO, GWO, PSO, SCA, GSA, HS	SHO, MFO, MVO, SCA, GSA, GA
F_{23}	2.79E-33	PSO, MFO, MVO, SCA, GSA, GA	SHO, PSO, MFO, SCA, GA, HS	SHO, MFO, MVO, SCA, GSA, GA, HS	SHO, GWO, PSO, SCA, GSA, GA, HS	SHO, GWO, PSO, MFO, SCA, GSA, GA	SHO, GWO, PSO, MFO, MVO, GA, HS	SHO, GWO, PSO, MFO, MVO, SCA, HS	SHO, GWO, PSO, MFO, MVO, SCA	SHO, GWO, PSO, MFO, MVO, SCA, GSA

3.2.1.5 Applications of SHO on Engineering Design Problems

In this section, SHO was tested with five constrained and one unconstrained engineering design problems such as welded beam, tension/compression spring, pressure vessel, speed reducer, rolling bearing element, and displacement of loaded structure.

3.2.1.5.1 Constraint Handling

Constraint handling is one of the biggest challenges in solving optimization problems using metaheuristic techniques. There are five constraint handling techniques^[143] namely penalty functions, hybrid methods, separation of objective functions and constraints, repair algorithms, and special operators. Among these techniques, the penalty functions are simple and easy to implement. There are numerous penalty functions such as static, annealing, adaptive, co-evolutionary, and death penalty. These approaches convert constraint problems into unconstrained problem by adding some penalty values. In this thesis, a static penalty approach is employed to handle constraints in optimization problems. Static penalty function does not depend on the current generation number and always remains constant during the entire computational process. This will help to reduce the time complexity of an algorithm and explore better in a search space.

$$\zeta(z) = f(z) \pm \left[\sum_{i=1}^m l_i \cdot \max(0, t_i(z))^\alpha + \sum_{j=1}^n o_j \cdot |U_j(z)|^\beta \right] \quad (3.12)$$

where $\zeta(z)$ is the modified objective function. l_i and o_j are positive penalty values. $t_i(z)$ and $U_j(z)$ are constraint functions. l_i and o_j are positive constants. The values of α and β are 1 and 2, respectively. This approach assigns the penalty value for each infeasible solution. In death penalty approach, a large value is assigned to the objective function of infeasible solution. Therefore, the static penalty function is employed which helps the search agents to move towards the feasible search

space of the problem.

3.2.1.5.2 Welded Beam Design Problem

The main objective of this problem is to minimize the fabrication cost of welded beam as shown in Fig. 3.8. The optimization constraints are shear stress (τ), bending stress (θ), buckling load (P_c) on the bar, and end deflection (δ) of the beam.

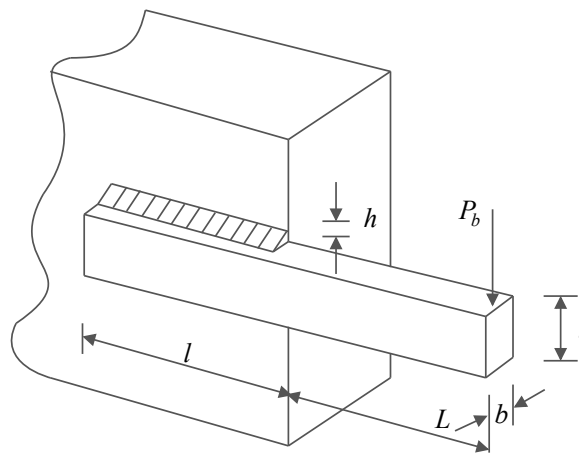


Fig. 3.8. Schematic view of welded beam problem.

There are four optimization variables of this problem which are as follows:

- Thickness of weld (h)
- Length of the clamped bar (l)
- Height of the bar (t)
- Thickness of the bar (b)

The mathematical formulation of this problem is formulated in Appendix C.1.3. The optimization methods previously applied to this problem include GWO, PSO, MFO, MVO, SCA, GSA, GA, and HS. The comparison for the best solution obtained

Table 3.7: Comparison results for welded beam design.

Algorithms	Optimum variables				Optimum cost
	h	t	l	b	
SHO	0.205563	3.474846	9.035799	0.205811	1.725661
GWO	0.205678	3.475403	9.036964	0.206229	1.726995
PSO	0.197411	3.315061	10.00000	0.201395	1.820395
MFO	0.203567	3.443025	9.230278	0.212359	1.732541
MVO	0.205611	3.472103	9.040931	0.205709	1.735472
SCA	0.204695	3.536291	9.004290	0.210025	1.759173
GSA	0.147098	5.490744	10.00000	0.217725	2.172858
GA	0.164171	4.032541	10.00000	0.223647	1.873971
HS	0.206487	3.635872	10.00000	0.203249	1.836250

Table 3.8: Comparison of SHO with other algorithms in terms of statistical results for welded beam design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	1.725661	1.725828	1.726064	0.000287	1.725787
GWO	1.726995	1.727128	1.727564	0.001157	1.727087
PSO	1.820395	2.230310	3.048231	0.324525	2.244663
MFO	1.732541	1.775231	1.802364	0.012397	1.812453
MVO	1.725472	1.729680	1.741651	0.004866	1.727420
SCA	1.759173	1.817657	1.873408	0.027543	1.820128
GSA	2.172858	2.544239	3.003657	0.255859	2.495114
GA	1.873971	2.119240	2.320125	0.034820	2.097048
HS	1.836250	1.363527	2.035247	0.139485	1.9357485

by such algorithms is presented in Table 3.7. The results indicate that SHO converges towards the best design. The comparison of statistical results is given in Table 3.8.

The results show that SHO performs better in terms of mean (average), median, standard deviation, and requires less number of analyses to find the best optimal design. The success rate of SHO algorithm is high as compared to other methods of solving this optimization problem.

3.2.1.5.3 Tension/compression Spring Design Problem

The objective of this problem is to minimize the tension/ compression spring weight as shown in Fig. 3.9.

Table 3.9: Comparison results for tension/compression spring design.

Algorithms	Optimum variables			Optimum cost
	d	D	N	
SHO	0.051144	0.343751	12.0955	0.012674000
GWO	0.050178	0.341541	12.07349	0.012678321
PSO	0.05000	0.310414	15.0000	0.013192580
MFO	0.05000	0.313501	14.03279	0.012753902
MVO	0.05000	0.315956	14.22623	0.012816930
SCA	0.050780	0.334779	12.72269	0.012709667
GSA	0.05000	0.317312	14.22867	0.012873881
GA	0.05010	0.310111	14.0000	0.013036251
HS	0.05025	0.316351	15.23960	0.012776352



Fig. 3.9. Schematic view of tension/compression spring problem.

The optimization constraints of this problem are:

- Shear stress.
- Surge frequency.
- Minimum deflection.

There are three design variables of this problem: such as wire diameter (d), mean coil diameter (D), and the number of active coils (N).

The mathematical formulation of this problem is formulated in Appendix C.1.4. The comparison of best optimal solution among several algorithms is given in Table 3.9. The results indicate that SHO performs better than other optimization methods. The statistical results of tension/compression spring design is given in Table 3.10. The results show that SHO performs better and require less number of analyses to find the best optimal design.

Table 3.10: Comparison of SHO with other algorithms in terms of statistical results for tension/compression spring design.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	0.012674000	0.012684106	0.012715185	0.000027	0.012687293
GWO	0.012678321	0.012697116	0.012720757	0.000041	0.012699686
PSO	0.013192580	0.014817181	0.017862507	0.002272	0.013192580
MFO	0.012753902	0.014023657	0.017236590	0.001390	0.013896512
MVO	0.012816930	0.014464372	0.017839737	0.001622	0.014021237
SCA	0.012709667	0.012839637	0.012998448	0.000078	0.012844664
GSA	0.012873881	0.013438871	0.014211731	0.000287	0.013367888
GA	0.013036251	0.014036254	0.016251423	0.002073	0.013002365
HS	0.012776352	0.013069872	0.015214230	0.000375	0.012952142

3.2.1.5.4 Pressure Vessel Design Problem

This problem was proposed by Kannan and Kramer^[144] to minimize the total cost, including cost of material, forming and welding of cylindrical vessel which are capped at both ends by hemispherical heads as shown in Fig. 3.10.

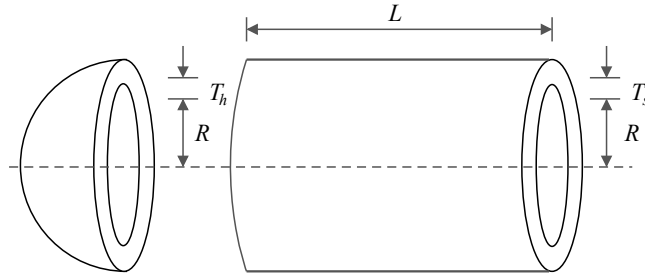


Fig. 3.10. Schematic view of pressure vessel problem.

There are four variables in this problem ($x_1 - x_4$):

- T_s (x_1 , thickness of the shell).
- T_h (x_2 , thickness of the head).
- R (x_3 , inner radius).
- L (x_4 , length of the cylindrical section without considering the head).

Among these four design variables, R and L are continuous variables while T_s and T_h are integer values which are multiples of 0.0625 in. The mathematical formulation of this problem is formulated in Appendix C.1.1. Table 3.11 represents

Table 3.11: Comparison results for pressure vessel design problem.

Algorithms	Optimum variables				Optimum cost
	T_s	T_h	R	L	
SHO	0.778210	0.384889	40.315040	200.00000	5885.5773
GWO	0.779035	0.384660	40.327793	199.65029	5889.3689
PSO	0.778961	0.384683	40.320913	200.00000	5891.3879
MFO	0.835241	0.409854	43.578621	152.21520	6055.6378
MVO	0.845719	0.418564	43.816270	156.38164	6011.5148
SCA	0.817577	0.417932	41.74939	183.57270	6137.3724
GSA	1.085800	0.949614	49.345231	169.48741	11550.2976
GA	0.752362	0.399540	40.452514	198.00268	5890.3279
HS	1.099523	0.906579	44.456397	179.65887	6550.0230

Table 3.12: Comparison of SHO with other algorithms in terms of statistical results for pressure vessel design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	5885.5773	5887.4441	5892.3207	002.893	5886.2282
GWO	5889.3689	5891.5247	5894.6238	013.910	5890.6497
PSO	5891.3879	6531.5032	7394.5879	534.119	6416.1138
MFO	6055.6378	6360.6854	7023.8521	365.597	6302.2301
MVO	6011.5148	6477.3050	7250.9170	327.007	6397.4805
SCA	6137.3724	6326.7606	6512.3541	126.609	6318.3179
GSA	11550.2976	23342.2909	33226.2526	5790.625	24010.0415
GA	5890.3279	6264.0053	7005.7500	496.128	6112.6899
HS	6550.0230	6643.9870	8005.4397	657.523	7586.0085

the comparisons of best optimal solution for SHO and other reported methods. According to this table, SHO is able to find an optimal design with minimum cost. Table 3.12 shows the statistical results obtained from proposed algorithm and other algorithms on pressure vessel design problem. The results show that SHO performs better than the other algorithms in terms of best optimal solution.

3.2.1.5.5 Speed Reducer Design Problem

The design of speed reducer is a more challenging benchmark because it can be associated with seven design variables^[145].

In this optimization problem (see Fig. 3.11) the weight of speed reducer is to be minimized with subject to constraints^[146]:

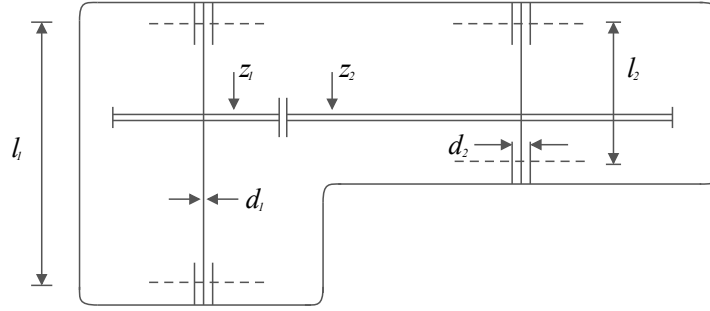


Fig. 3.11. Schematic view of speed reducer problem.

- Bending stress of the gear teeth.
- Surface stress.
- Transverse deflections of the shafts.
- Stresses in the shafts.

There are seven optimization variables ($x_1 - x_7$) of this problem which can represent as the face width ($b = x_1$), module of teeth ($m = x_2$), number of teeth in the pinion ($z = x_3$), length of the first shaft between bearings ($l_1 = x_4$), length of the second shaft between bearings ($l_2 = x_5$), the diameter of first ($d_1 = x_6$) shafts, and the diameter of second shafts ($d_2 = x_7$). The third variable, number of teeth in the pinion (z), is of integer values. Therefore, all other variables (excluding x_3) are continuous.

The mathematical formulation of this problem is formulated in Appendix C.1.2. The comparison of the best optimal solution with various optimization methods is given in Table 3.13. The statistical results of eight optimization methods are compared with the proposed method which is given in Table 3.14. According to Table 3.13, SHO provides the best optimal solution so far and able to find a design with minimum cost.

Table 3.13: Comparison results for speed reducer design problem.

Algorithms	Optimum variables							Optimum cost
	b	m	z	l_1	l_2	d_1	d_2	
SHO	3.50159	0.7	17	7.3	7.8	3.35	5.28874	2998.5507
GWO	3.506690	0.7	17	7.4	7.8	3.36	5.286768	3001.288
PSO	3.500019	0.7	17	8.3	7.8	3.35	5.286715	3005.763
MFO	3.507524	0.7	17	7.3	7.8	3.32	5.287524	3009.571
MVO	3.508502	0.7	17	7.4	7.8	3.36	5.286777	3002.928
SCA	3.508755	0.7	17	7.3	7.8	3.46	5.289213	3030.563
GSA	3.600000	0.7	17	8.3	7.8	3.37	5.289224	3051.120
GA	3.510253	0.7	17	8.35	7.8	3.36	5.287723	3067.561
HS	3.520124	0.7	17	8.37	7.8	3.37	5.288719	3029.002

Table 3.14: Comparison of SHO with other algorithms in terms of statistical results for speed reducer design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	2998.5507	2999.640	3003.889	1.93193	2999.187
GWO	3001.288	3005.845	3008.752	5.83794	3004.519
PSO	3005.763	3105.252	3211.174	79.6381	3105.252
MFO	3009.571	3021.256	3054.524	11.0235	3020.524
MVO	3002.928	3028.841	3060.958	13.0186	3027.031
SCA	3030.563	3065.917	3104.779	18.0742	3065.609
GSA	3051.120	3170.334	3363.873	92.5726	3156.752
GA	3067.561	3186.523	3313.199	17.1186	3198.187
HS	3029.002	3295.329	3619.465	57.0235	3288.657

3.2.1.5.6 Rolling Element Bearing Design Problem

The main objective of this problem is to maximize the dynamic load carrying capacity of a rolling element bearing as shown in Fig. 3.12.

There are 10 decision variables of this problem which are pitch diameter (D_m), ball diameter (D_b), number of balls (Z), inner (f_i) and outer (f_o) raceway curvature coefficients, K_{Dmin} , K_{Dmax} , ε , e , and ζ (see Fig. 3.12).

The mathematical formulation of this problem is formulated in Appendix C.1.5.

Table 3.15: Comparison results for rolling element bearing design problem.

Algorithms	Optimum variables										Opt. cost
	D_m	D_b	Z	f_i	f_o	K_{Dmin}	K_{Dmax}	ε	e	ζ	
SHO	125	21.40732	10.93268	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85054.532
GWO	125.6	21.35129	10.98781	0.515	0.515	0.5	0.7	0.3	0.03	0.6	84807.111
PSO	125	20.75388	11.17342	0.515	0.515	0.5	0.6	0.3	0.05	0.6	81691.202
MFO	125	21.03287	10.96571	0.515	0.515	0.5	0.7	0.3	0.02	0.6	84002.524
MVO	125.6	21.32250	10.97338	0.515	0.515	0.5	0.7	0.3	0.04	0.6	84491.266
SCA	125	21.14834	10.96928	0.515	0.515	0.5	0.7	0.3	0.03	0.6	83431.117
GSA	125	20.85417	11.14989	0.515	0.517	0.5	0.6	0.3	0.02	0.6	82276.941
GA	125	20.77562	11.01247	0.515	0.515	0.5	0.6	0.3	0.05	0.6	82773.982
HS	125	20.87123	11.16697	0.515	0.516	0.5	0.6	0.3	0.05	0.6	81569.527

Table 3.16: Comparison of SHO with other algorithms in terms of statistical results for rolling element bearing design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	85054.532	85024.858	85853.876	0186.68	85040.241
GWO	84807.111	84791.613	84517.923	0137.186	84960.147
PSO	81691.202	50435.017	32761.546	13962.150	42287.581
MFO	84002.524	82357.493	83979.254	1401.524	84497.397
MVO	84491.266	84353.685	84100.834	0392.431	84398.601
SCA	83431.117	81005.232	77992.482	1710.777	81035.109
GSA	82276.941	78002.107	71043.110	3119.904	78398.853
GA	82773.982	81198.753	80687.239	1679.367	8439.728
HS	81569.527	80397.998	79412.779	1756.902	8347.009

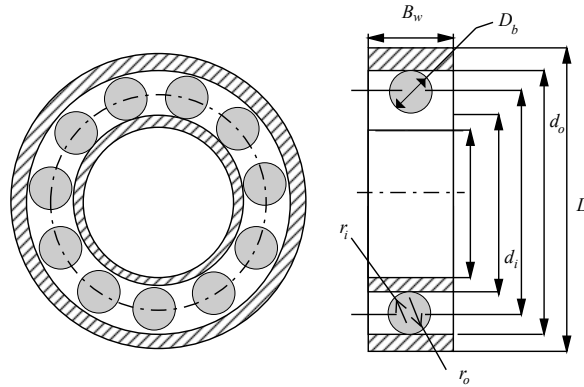


Fig. 3.12. Schematic view of rolling element bearing problem.

Table 3.15 shows the comparison of best optimal solution obtained from different optimization methods. The statistical optimization results obtained from the reported algorithms are reported in Table 3.16. From Table 3.15, the proposed approach provides the best optimal solution than the other optimizers.

Table 3.17: Comparison results for displacement of loaded structure.

Algorithms	Optimum cost (π)
SHO	168.8889
GWO	170.3645
PSO	170.5960
MFO	173.3654
MVO	169.3023
SCA	169.0032
GSA	176.3697
GA	171.3674
HS	172.0324

3.2.1.5.7 Displacement of Loaded Structure Problem

Displacement is a vector which defines the shortest distance between the initial and the final position of a given point (see Fig. 3.13). This problem is described as the minimization of potential energy to reduce the excess load of structure and overcome the problem of structural failure. It consists of minimizing the potential energy ($f(z)$) of loaded structure.

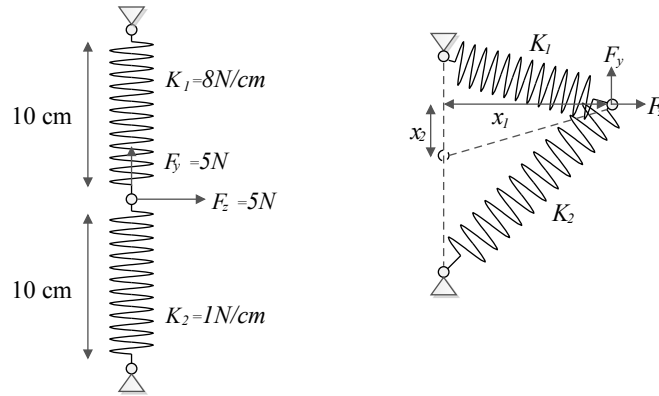


Fig. 3.13. Schematic view of displacement of loaded structure.

Table 3.17 presents the comparisons of best optimal solution for SHO and other competitor methods. According to this table, SHO is able to minimize the potential energy for displacement of loaded structure problem. Table 3.18 shows the statistical results obtained from proposed algorithm and other algorithms. It shows that SHO performs better than other competitor approaches.

Table 3.18: Comparison of SHO statistical for displacement of loaded structure.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
SHO	168.8889	170.3659	173.6357	023.697	169.6710
GWO	170.3645	171.3694	174.3970	196.037	173.3694
PSO	170.5960	174.6354	175.3602	236.036	173.9634
MFO	173.3654	178.6972	180.6357	323.364	175.3670
MVO	169.3023	171.0034	174.3047	202.753	170.0032
SCA	169.0032	171.7530	174.4527	129.047	170.3647
GSA	176.3697	178.7521	179.5637	113.037	174.367
GA	171.3674	172.0374	174.0098	212.703	172.0097
HS	172.0324	173.4327	174.0399	080.111	171.3697

3.2.2 Emperor Penguin Optimizer (EPO)

The emperor penguin, scientifically named as *Aptenodytes forsteri*, is the tallest and heaviest in all of the penguin species. The male and female emperor penguins are very much similar in terms of plumage and size. The dorsal side and head are black with a white belly, pale yellow breast, and bright yellow ear patches.

Emperor penguins spend their lives in open ice and breeds during the winter season. During the breeding season, emperor penguins come ashore in huge colonies which include hundreds of thousands of emperor penguins. Females lay a single egg and can travel 50 miles to reach the ocean for hunting. The emperor penguin is a social like animal by its foraging behavior and hunting together in a group.

In the sea, emperor penguins can dive up to 1,900 feet deeper and stay under sea for more than 25 minutes. Emperor penguins are flightless like other penguins species with stiffened and flattened wings.

Emperor penguins are the only species that huddles to survive during the Antarctic winter. The huddling behavior of emperor penguins is decomposed into four phases^[147]:

- Generate and determine the huddle boundary of emperor penguins.
- Calculate the temperature profile around the huddle.
- Determine the distance between emperor penguins.

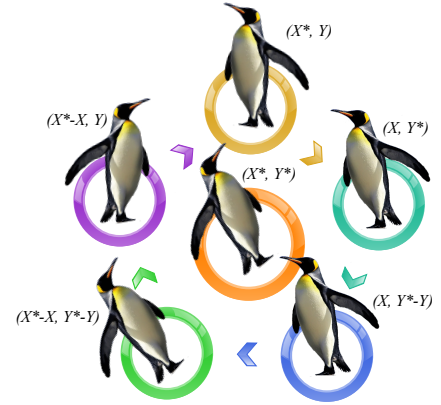


Fig. 3.14. Huddling behavior of emperor penguins.

- Relocate the effective mover.

An important feature of this huddling behavior is that each penguin has an equal opportunity to the warmth of huddle.

Fig. 3.14 shows the huddling behavior of emperor penguins. In this figure, the emperor penguin (X^*, Y^*) can update its position towards the positions of other emperor penguins and reaches at different number of places about the current position.

3.2.2.1 Mathematical Model and Optimization Algorithm

In this subsection, the huddling behavior of emperor penguins is mathematically modeled. The main aim of this model is to find an effective mover. The huddle is assumed to be situated on two dimensional L -shape polygon plane. Firstly, emperor penguins generate the huddle boundary randomly. Thereafter, the temperature profile around the huddle is computed. The distance between emperor penguins is also calculated which will be helpful for more exploration and exploitation. Finally, the effective mover, i.e., the best optimal solution is obtained and recompute the boundary of huddle with updated positions of emperor penguins (or search agents). These steps are explained in the preceding subsections.

3.2.2.1.1 Generate and Determine the Huddle Boundary

During huddling, emperor penguins generally position themselves on a polygon shape grid boundary. The emperor penguins have at least two neighbours which are chosen randomly in the huddle as shown in Fig. 3.14. The wind flow around the huddle is determined to find the huddle boundary around a polygon. However, the wind flow is faster than the movement of an emperor penguin. The concepts of complex variables are used to describe the randomly generated huddle boundary of the emperor penguin.

Let Φ defines the wind velocity and Ψ be the gradient of Φ .

$$\Psi = \nabla \Phi \quad (3.13)$$

Vector Ω is combined with Φ to generate the complex potential.

$$F = \Phi + i\Omega \quad (3.14)$$

where i denotes the imaginary constant and F is an analytical function on the polygon plane. Fig. 3.14 shows the effect of Eq. (3.14) in two-dimensional environment. In this figure, the emperor penguins can update their position randomly towards the position of emperor penguin which is to be situated at the center of L -shaped polygon region with highest effective fitness rate during iteration process.

3.2.2.1.2 Temperature Profile Around the Huddle

The emperor penguins generate huddle to conserve energy and maximize the ambient temperature in the huddle. To mathematically model this situation, it is assumed that the temperature $T = 0$ when the radius of polygon $R > 0.5$ and temperature $T = 1$ when the radius becomes $R \leq 0.5$. This temperature profile

is responsible for exploration and exploitation process for emperor penguins with different locations. The temperature profile around the huddle T' is computed as follows:

$$T' = \left(T - \frac{Max_{iteration}}{x - Max_{iteration}} \right)$$

$$T = \begin{cases} 0, & \text{if } R > 0.5 \\ 1, & \text{if } R < 0.5 \end{cases} \quad (3.15)$$

where x defines the current iteration, $Max_{iteration}$ denotes the maximum number of iterations, R is the radius, and T is the time for finding best optimal solution in a search space.

3.2.2.1.3 Distance between Emperor Penguins

The distance between emperor penguin and best obtained optimal solution is computed after the generation of huddle boundary. The current best optimal solution is the solution whose fitness value is close to the optimum. The other search agents (or emperor penguins) will update their positions according to the current best optimal solution which is mathematically defined as follows:

$$\vec{D}_{ep} = Abs \left(S(\vec{A}) \cdot \vec{P}(x) - \vec{C} \cdot \vec{P}_{ep}(x) \right) \quad (3.16)$$

where $\vec{D}_{ep}$ represents the distance between the emperor penguin and best fittest search agent (i.e., best emperor penguin whose fitness value is less), x indicates the current iteration. $\vec{A}$ and $\vec{C}$ are used to avoid the collision between neighbours (i.e., other emperor penguins). $\vec{P}$ defines the best optimal solution (i.e., fittest emperor penguin), $\vec{P}_{ep}$ indicates the position vector of the emperor penguin. $S()$ defines the social forces of emperor penguins that is responsible to move towards the direction of best optimal search agent. The vectors $\vec{A}$ and $\vec{C}$ are computed as follows:

$$\vec{A} = (M \times (T' + P_{grid}(Accuracy)) \times Rand()) - T' \quad (3.17)$$

$$P_{grid}(Accuracy) = Abs(\vec{P} - \vec{P}_{ep}) \quad (3.18)$$

$$\vec{C} = Rand() \quad (3.19)$$

where M is the movement parameter that maintains a gap between search agents for collision avoidance. The value of parameter M is set to 2. T' is the temperature profile around the huddle, $P_{grid}(Accuracy)$ defines the polygon grid accuracy by comparing the difference between emperor penguins, and $Rand()$ is a random function lies in the range of $[0, 1]$.

The function $S()$ is calculated as follows:

$$S(\vec{A}) = \left(\sqrt{f \cdot e^{-x/l} - e^{-x}} \right)^2 \quad (3.20)$$

where e defines the expression function. f and l are control parameters for better exploration and exploitation. The values of f and l lie in the range of $[2, 3]$ and $[1.5, 2]$, respectively. Note that it has been observed that the proposed algorithm provides better results between these ranges.

3.2.2.1.4 Relocate the Mover

The positions of emperor penguins are updated according to the best obtained optimal solution, i.e., mover. This mover is responsible for changing the positions of other search agents in a given search space and vacates its current position. The following equation is proposed to update the next position of an emperor penguin:

$$\vec{P}_{ep}(x+1) = P(x) - \vec{A} \cdot \vec{D}_{ep} \quad (3.21)$$

where $\vec{P}_{ep}(x+1)$ represents the next updated position of emperor penguin. During iteration process, the huddling behavior of emperor penguins is recomputed once the mover has been re-located.

The pseudo code of EPO algorithm is shown in Algorithm 2. There are some interesting points about the proposed EPO algorithm which are given below:

- The proposed algorithm saves the best solutions which are obtained so far during the course of iterations.
- The proposed polygon grid mechanism defines an L-shaped polygon grid around the solutions which can be extended to higher dimensions as shown in Fig. 3.15.
- $\vec{A}$ and $\vec{C}$ assist the candidate solutions to behave more randomly in a search space and responsible to avoid the collisions between search agents.
- The proposed distance method allows candidate solutions to locate the possible position of the best fittest emperor penguin.
- The convergence behaviors of common optimization algorithms suggests that exploitation tends to increase the speed of convergence, while exploration tends to decrease the convergence rate of the algorithm. Therefore, the possibility of better exploration and exploitation is done by the adjusted values of vectors $\vec{A}$ and $\vec{C}$. These behaviors are shown in Fig. 3.16 which shows that half of the iterations are responsible for exploration and the other half are responsible for exploitation in a given search space.
- The swarm behavior of emperor penguins in a search region defines the effectively collective behavior of EPO algorithm as shown in Fig. 3.17.

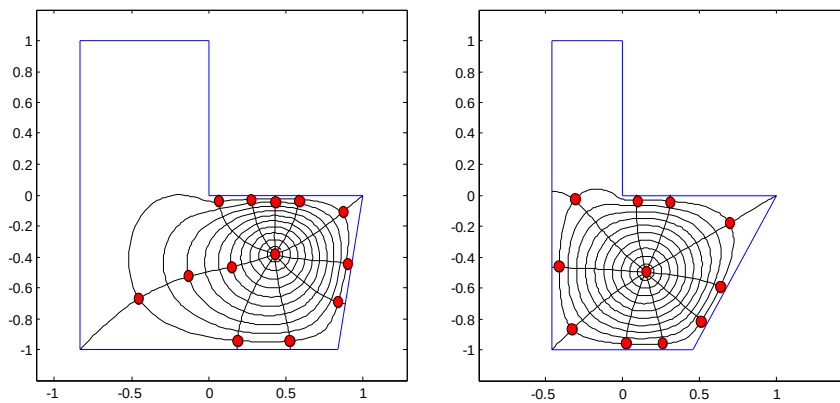


Fig. 3.15. *L*-shaped polygon huddling boundary of emperor penguins.

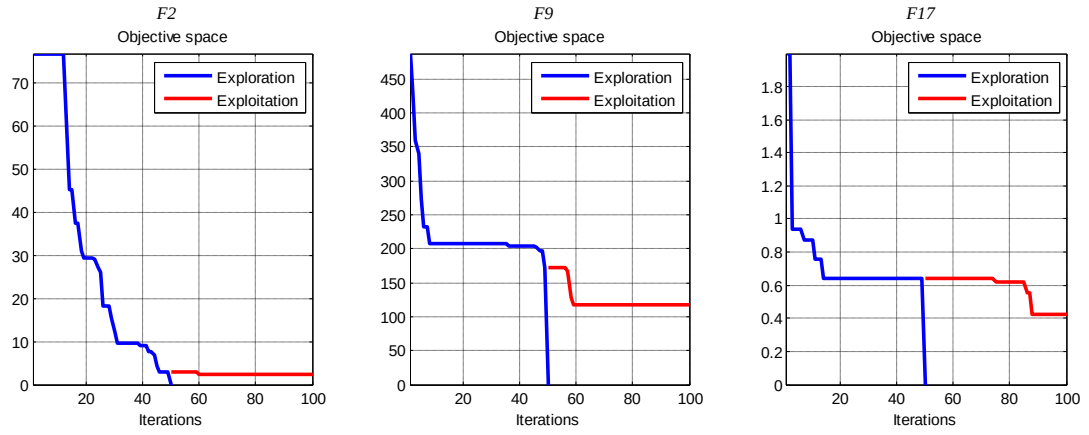


Fig. 3.16. Exploration and Exploitation phases of EPO on F_2 , F_9 , and F_{17} benchmark test functions.

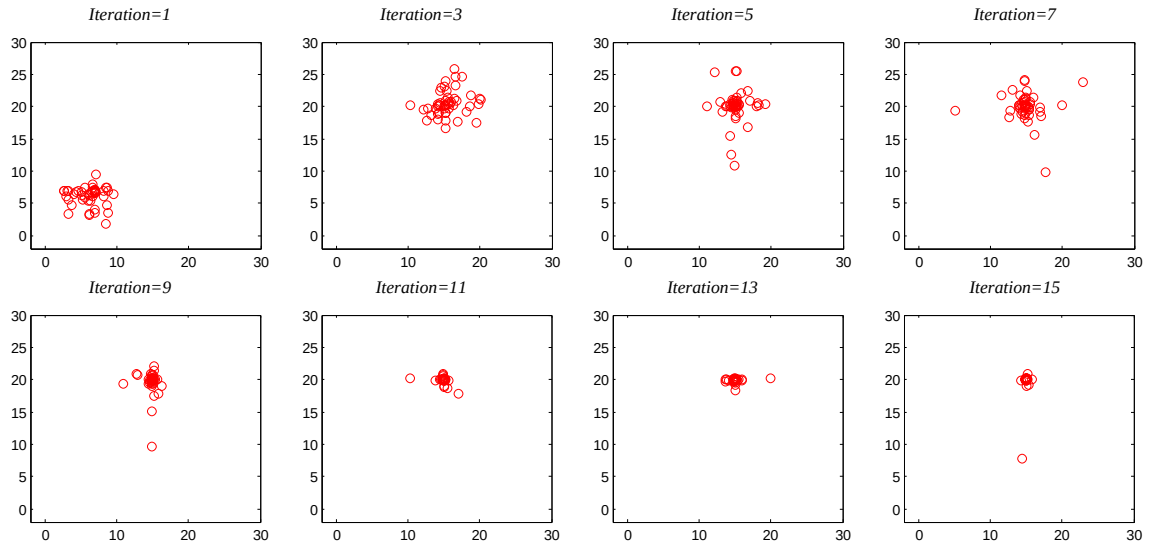


Fig. 3.17. Swarm behaviors of emperor penguins in two-dimensional environment.

3.2.2.2 Steps and Flowchart of Proposed EPO Algorithm

The steps and flowchart (see Fig. 3.18) of proposed EPO are summarized as follows:

Step 1: Initialize the emperor penguins population $P_{ep}(\vec{x})$, where $x = 1, 2, \dots, n$.

Step 2: Choose the initial parameters: $T', \vec{A}, \vec{C}, S(), R$, and $Max_{iteration}$.

Step 3: Calculate the fitness value of each search agent.

Step 4: Determine the huddle boundary of emperor penguins using Eqs. (3.13) and (3.14).

Step 5: Calculate the temperature profile T' around the huddle using Eq. (3.15).

Step 6: Compute the distance between the emperor penguins using Eqs. (3.16)-(3.20).

Step 7: Update the positions of other search agents using Eq. (3.21).

Step 8: Check whether any search agent goes beyond the boundary in a given search space and then amend it.

Step 9: Calculate the updated search agent fitness value and update the position of previously obtained optimal solution.

Step 10: The algorithm will be stopped until the stopping criterion is satisfied. Otherwise, go to Step 5.

Step 11: Return the best optimal solution after the termination criteria, which is obtained so far.

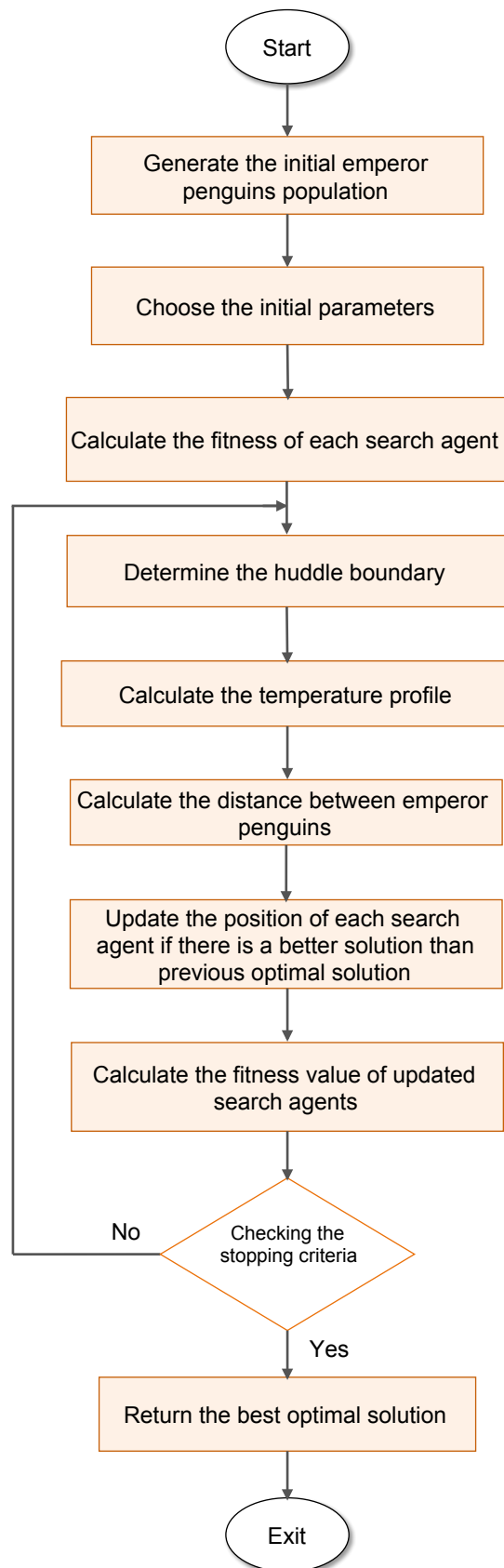


Fig. 3.18. Flowchart of the proposed EPO algorithm.

Algorithm 2 Emperor Penguin Optimizer (EPO)

Input: the emperor penguins population $P_{ep}(x)$ ($x \leftarrow 1, 2, \dots, n$)
Output: the best obtained search agent $\vec{P}$

```
1: procedure EPO
2: Initialize the parameters  $T', \vec{A}, \vec{C}, S(), R$ , and  $Max_{iteration}$ 
3:   while ( $x < Max_{iteration}$ ) do
4:      $FITNESS(P_{ep})$  /* Compute the fitness of each search agent using FITNESS
       function */
5:      $R \leftarrow Rand()$  /* Generate random number in range [0, 1] */
6:     if ( $R > 0.5$ ) then
7:        $T \leftarrow 0$ 
8:     else
9:        $T \leftarrow 1$ 
10:    end if
11:     $T' \leftarrow \left( T - \frac{Max_{iteration}}{x - Max_{iteration}} \right)$  /* Compute the temperature profile around
       the huddle */
12:    for  $i \leftarrow 1$  to  $n$  do
13:      for  $j \leftarrow 1$  to  $n$  do
14:        Compute the vectors  $\vec{A}$  and  $\vec{C}$ 
15:        Compute the function  $S(\vec{A})$ 
16:        Update the position of current agent using Eq. (3.21)
17:      end for
18:    end for
19:    Update parameters  $T', \vec{A}, \vec{C}$ , and  $S()$ 
20:    Amend search agent which goes beyond the region of search space
21:     $FITNESS(P_{ep})$  /* Again compute the fitness value of updated search agents
       using FITNESS function */
22:    Update  $\vec{P}$  if there is a better solution than previous optimal solution i.e., ( $FIT_{best}$ )
23:     $x \leftarrow x + 1$ 
24:  end while
25: return  $\vec{P}$ 
26: end procedure

27: procedure  $FITNESS(P_{ep})$ 
28:   for  $i \leftarrow 1$  to  $n$  do
29:      $FIT[i] \leftarrow FITNESS\_FUNCTION(P_{ep})$  /* Compute the fitness of each
       individual */
30:   end for
31:    $FIT_{best} \leftarrow BEST(FIT[])$  /* Compute the best fitness value using BEST function
       */
32:   return  $FIT_{best}$ 
33: end procedure

34: procedure  $BEST(FIT[])$ 
35:    $best \leftarrow FIT[0]$ 
36:   for  $i \leftarrow 1$  to  $n$  do
37:     if ( $FIT[i] < best$ ) then
38:        $best \leftarrow FIT[i]$ 
39:     end if
40:   end for
41:   return  $best$  /* Return the best fitness value */
42: end procedure
```

3.2.2.3 Computational Complexity

In this subsection, the computational complexity of proposed EPO algorithm is discussed. Both the time and space complexities of the proposed algorithm are given below.

3.2.2.3.1 Time Complexity

1. Population initialization process requires $\mathcal{O}(n \times d)$ time, where n indicates the population size and d indicates the dimension of a given problem.
2. The fitness of each agent requires $\mathcal{O}(Max_{iteration} \times n \times d)$ time, where $Max_{iteration}$ is the maximum number of iteration to simulate the proposed algorithm.
3. The function $S()$ requires $\mathcal{O}(N)$ time, where N defines the social forces of emperor penguins for better exploration and exploitation.
4. Steps 2 - 3 are repeated until the termination criteria is satisfied which needs $\mathcal{O}(k)$ time.

Hence, the total complexity of Steps 2 - 3 is $\mathcal{O}(n \times Max_{iteration} \times d \times N)$. Therefore, the overall time complexity of EPO algorithm is $\mathcal{O}(k \times n \times Max_{iteration} \times d \times N)$.

3.2.2.3.2 Space Complexity

The space complexity of EPO algorithm is the maximum amount of space used at any one time which is considered during its initialization process. Thus, the total space complexity of EPO algorithm is $\mathcal{O}(n \times d)$.

3.2.2.4 Experimental Results and Discussions

This section describes the experimentation on forty-four standard benchmark test functions to evaluate the performance of proposed algorithm. The results are compared with eight well-known metaheuristic algorithms.

3.2.2.4.1 Benchmark Test Functions

The forty-four benchmark test functions are applied on proposed algorithm to demonstrate its applicability and efficiency. These functions are divided into five main categories: Unimodal^[132], Multimodal^[133], Fixed-dimension Multimodal^[132,133], Composite functions^[134], and IEEE CEC-2015 functions^[148]. These functions are tabulated in Appendix A. In Appendix A, *Dim* and *Range* indicate the dimension of the function and boundary of the search space, respectively whereas f_{min} denotes the minimization function.

3.2.2.4.2 Algorithms Involved for Comparison

To validate the performance of the proposed EPO algorithm, the eight well-known optimization algorithms such as Spotted Hyena Optimizer (SHO)^[149], Grey Wolf Optimizer (GWO)^[136], Particle Swarm Optimization (PSO)^[27], Multi-Verse Optimizer (MVO)^[138], Sine Cosine Algorithm (SCA)^[139], Gravitational Search Algorithm (GSA)^[140], Genetic Algorithm (GA)^[141], Harmony Search (HS)^[142] are used for comparison. The number of search agents for the above-mentioned algorithms are set according to the literature.

3.2.2.4.3 Experimental Setup

The parameter settings of proposed EPO and competitor metaheuristic algorithms, i.e., SHO, GWO, PSO, MVO, SCA, GSA, GA, and HS are tabulated in Table 3.1. The

parameter values of these algorithms are set as they are recommended in their original papers. The average and standard deviation of the best optimal solution are mentioned in tables. For each benchmark test function, EPO algorithm utilizes 30 independent runs in which each run employs 1000 iterations.

3.2.2.4.4 Performance Comparison

In order to demonstrate the effectiveness of proposed algorithm, it is compared with eight well-known optimization algorithms on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions.

- Evaluation of test functions $F_1 - F_7$

The functions $F_1 - F_7$ are unimodal test problems that are used to assess the exploitation capability of the metaheuristic algorithms. It is observed from Table 3.19 that EPO is very competitive as compared to the other competitor algorithms. The results reveal that EPO is able to determine the optimal solution for functions, i.e., F_4, F_5, F_6 , and F_7 . EPO has better exploitation capability and able to find the best optimal solution very efficiently.

- Evaluation of test functions $F_8 - F_{23}$

Tables 3.20 and 3.21 depict the performance of above-mentioned algorithms on multimodal test functions ($F_8 - F_{13}$) and fixed-dimension multimodal test functions ($F_{14} - F_{23}$). From these tables, it can be seen that EPO is able to find optimal solution for nine test problems (i.e., $F_8, F_{10}, F_{13}, F_{14}, F_{15}, F_{17}, F_{18}, F_{19}$, and F_{22}) and also obtains competitive results in majority of test problems. The results reveal that EPO algorithm has better exploration capability.

- Evaluation of test functions $F_{24} - F_{29}$

The composite benchmark functions is very challenging task because these functions require balance between exploration and exploitation. These

test functions are shifted, rotated, expanded, and combined version of classical functions. The local optima avoidance can be observed in these test functions. Table 3.22 depicts that EPO algorithm is an effective optimizer for F_{25} , F_{26} , and F_{27} test functions and very competitive in other test cases.

- Evaluation of IEEE CEC-2015 test functions ($CEC1 - CEC15$)

Table 3.23 shows the performance of EPO and other algorithms on IEEE CEC-2015 test functions. It is observed from Table 3.23 that proposed algorithm is efficient for $CEC - 1, CEC - 3, CEC - 7, CEC - 8, CEC - 9, CEC - 10, CEC - 11, CEC - 12, CEC - 13, CEC - 14$, and $CEC - 15$ test functions. Fig. 3.20 shows the box plot for the proposed EPO and other competitor algorithms on IEEE CEC-2015 benchmark test functions. It can be seen from this figure that EPO is the best efficient optimizer for most of the benchmark test functions.

3.2.2.5 Convergence Analysis

The main intention behind the convergence analysis is to understand the behavior of proposed EPO algorithm. Fig. 3.19 shows the best run convergence curves of EPO and the other metaheuristic algorithms. EPO has three different convergence behaviors. In the initial stage of iterations, EPO converges more quickly in the search space due to its adaptive mechanism. This behavior is shown in test functions (i.e., F_1, F_9, F_{12} , and F_{18}). In the second step, EPO converges towards the optimum when final iteration reaches which is shown in F_{23} and F_{27} test functions. The last step shows the express convergence from the initial step of iterations which is shown in F_5 and F_{15} test functions. These results reveal that EPO algorithm maintains a proper balance between exploration and exploitation to find the global optimum. Hence, the results reveal different behaviors of EPO algorithm which show that the success rate of EPO is better than the other competitor algorithms for solving optimization problems.

Table 3.19: Results of unimodal benchmark test functions.

F	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_1	5.71E-28 (8.31E-29)	0.00E+00 (0.00E+00)	4.61E-23 (7.37E-23)	4.98E-09 (1.40E-08)	2.81E-01 (1.11E-01)	3.55E-02 (1.06E-01)	1.16E-16 (6.10E-17)	1.95E-12 (2.01E-11)	7.86E-10 (8.11E-09)
F_2	6.20E-40 (3.32E-40)	0.00E+00 (0.00E+00)	1.20E-34 (1.30E-34)	7.29E-04 (1.84E-03)	3.96E-01 (1.41E-01)	3.23E-05 (8.57E-05)	1.70E-01 (9.29E-01)	6.53E-18 (5.10E-17)	5.99E-20 (1.11E-17)
F_3	2.05E-19 (9.17E-20)	0.00E+00 (0.00E+00)	1.00E-14 (4.10E-14)	1.40E+01 (7.13E+00)	4.31E+01 (8.97E+00)	4.91E+03 (3.89E+03)	4.16E+02 (1.56E+02)	7.70E-10 (7.36E-09)	9.19E-05 (6.16E-04)
F_4	4.32E-18 (3.98E-19)	7.78E-12 (8.96E-12)	2.02E-14 (2.43E-14)	6.00E-01 (1.72E-01)	8.80E-01 (2.50E-01)	1.87E+01 (8.21E+00)	1.12E+00 (9.89E-01)	9.17E+01 (5.67E+01)	8.73E-01 (1.19E-01)
F_5	5.07E+00 (4.90E-01)	8.59E+00 (5.53E-01)	2.79E+01 (1.84E+00)	4.93E+01 (3.89E+01)	1.18E+02 (1.43E+02)	7.37E+02 (1.98E+03)	3.85E+01 (3.47E+01)	5.57E+02 (4.16E+01)	8.91E+02 (2.97E+02)
F_6	7.01E-19 (4.39E-20)	2.46E-01 (1.78E-01)	6.58E-01 (3.38E-01)	9.23E-09 (1.78E-08)	3.15E-01 (9.98E-02)	4.88E+00 (9.75E-01)	1.08E-16 (4.00E-17)	3.15E-01 (9.98E-02)	8.18E-17 (1.70E-18)
F_7	2.71E-05 (9.26E-06)	3.29E-05 (2.43E-05)	7.80E-04 (3.85E-04)	6.92E-02 (2.87E-02)	2.02E-02 (7.43E-03)	3.88E-02 (5.79E-02)	7.68E-01 (2.77E+00)	6.79E-04 (3.29E-03)	5.37E-01 (1.89E-01)

Table 3.20: Results of multimodal benchmark test functions.

F	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_8	-8.76E+02 (5.92E+01)	-1.16E+02 (2.72E+01)	-6.14E+02 (9.32E+01)	-6.01E+02 (1.30E+02)	-6.92E+02 (9.19E+01)	-3.81E+02 (2.83E+01)	-2.75E+02 (5.72E+01)	-5.11E+02 (4.37E+01)	-4.69E+01 (3.94E+01)
F_9	6.90E-01 (4.81E-01)	0.00E+00 (0.00E+00)	4.34E-01 (1.66E+00)	4.72E+01 (1.03E+01)	1.01E+02 (1.89E+01)	2.23E+01 (3.25E+01)	3.35E+01 (1.19E+01)	1.23E-01 (4.11E+01)	4.85E-02 (3.91E+01)
F_{10}	8.03E-16 (2.74E-14)	2.48E+00 (1.41E+00)	1.63E-14 (3.14E-15)	3.86E-02 (2.11E-01)	1.15E+00 (7.87E-01)	1.55E+01 (8.11E+00)	8.25E-09 (1.90E-09)	5.31E-11 (1.11E-10)	2.83E-08 (4.34E-07)
F_{11}	4.20E-05 (4.73E-04)	0.00E+00 (0.00E+00)	2.29E-03 (5.24E-03)	5.50E-03 (7.39E-03)	5.74E-01 (1.12E-01)	3.01E-01 (2.89E-01)	8.19E+00 (3.70E+00)	3.31E-06 (4.23E-05)	2.49E-05 (1.34E-04)
F_{12}	5.09E-03 (3.75E-03)	3.68E-02 (1.15E-02)	3.93E-02 (2.42E-02)	1.05E-10 (2.06E-10)	1.27E+00 (1.02E+00)	5.21E+01 (2.47E+02)	2.65E-01 (3.14E-01)	9.16E-08 (4.88E-07)	1.34E-05 (6.23E-04)
F_{13}	0.00E+00 (0.00E+00)	9.29E-01 (9.52E-02)	4.75E-01 (2.38E-01)	4.03E-03 (5.39E-03)	6.60E-02 (4.33E-02)	2.81E+02 (8.63E+02)	5.73E-32 (8.95E-32)	6.39E-02 (4.49E-02)	9.94E-08 (2.61E-07)

Table 3.21: Results of fixed-dimension multimodal benchmark test functions.

F	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_{14}	1.08E+00 (4.11E-02)	9.68E+00 (3.29E+00)	3.71E+00 (3.86E+00)	2.77E+00 (2.32E+00)	9.98E+01 (9.14E-12)	1.26E+00 (6.86E-01)	3.61E+00 (2.96E+00)	4.39E+00 (4.41E-02)	6.79E+00 (1.12E+00)
F_{15}	8.21E-03 (4.09E-04)	9.01E-03 (1.06E-03)	3.66E-02 (7.60E-02)	9.09E-03 (2.38E-03)	7.15E-02 (1.26E-01)	1.01E-02 (3.75E-03)	6.84E-02 (7.37E-02)	7.36E-02 (2.39E-03)	5.15E-02 (3.45E-03)
F_{16}	-1.02E+00 (9.80E-07)	-1.03E+00 (2.86E-11)	-1.02E+00 (7.02E-09)	-1.02E+00 (0.00E+00)	-1.02E+00 (4.74E-08)	-1.02E+00 (3.23E-05)	-1.02E+00 (0.00E+00)	-1.02E+00 (4.19E-07)	-1.01E+00 (3.64E-08)
F_{17}	3.98E-01 (5.39E-05)	3.98E-01 (2.46E-01)	3.98E-01 (7.00E-07)	3.98E-01 (9.03E-16)	3.98E-01 (1.15E-07)	3.98E-01 (7.61E-04)	3.98E-01 (1.13E-16)	3.98E-01 (3.71E-17)	3.98E-01 (9.45E-15)
F_{18}	3.00E+00 (1.15E-08)	3.00E+00 (9.05E+00)	3.00E+00 (7.16E-06)	3.00E+00 (6.59E-05)	3.00E+00 (1.48E+01)	3.00E+00 (2.25E-05)	3.00E+00 (3.24E-02)	3.00E+00 (6.33E-07)	3.00E+00 (1.94E-10)
F_{19}	-3.86E+00 (6.50E-07)	-3.71E+00 (4.39E-01)	-3.84E+00 (1.57E-03)	-3.80E+00 (3.37E-15)	-3.77E+00 (3.53E-07)	-3.75E+00 (2.55E-03)	-3.86E+00 (4.15E-01)	-3.81E+00 (4.37E-10)	-3.73E+00 (9.69E-04)
F_{20}	-2.81E+00 (7.11E-01)	-1.44E+00 (5.47E-01)	-3.27E+00 (7.27E-02)	-3.32E+00 (2.66E-01)	-3.23E+00 (5.37E-02)	-2.84E+00 (3.71E-01)	-1.47E+00 (5.32E-01)	-2.39E+00 (4.37E-01)	-2.17E+00 (1.64E-01)
F_{21}	-8.07E+00 (2.29E+00)	-2.08E+00 (3.80E-01)	-9.65E+00 (1.54E+00)	-7.54E+00 (2.77E+00)	-7.38E+00 (2.91E+00)	-2.28E+00 (1.80E+00)	-4.57E+00 (1.30E+00)	-5.19E+00 (2.34E+00)	-7.33E+00 (1.29E+00)
F_{22}	-10.01E+00 (3.97E-02)	-1.61E+00 (2.04E-04)	-1.04E+00 (2.73E-04)	-8.55E+00 (3.08E+00)	-8.50E+00 (3.02E+00)	-3.99E+00 (1.99E+00)	-6.58E+00 (2.64E+00)	-2.97E+00 (1.37E-02)	-1.00E+00 (2.89E-04)
F_{23}	-3.41E+00 (1.11E-02)	-1.68E+00 (2.64E-01)	-1.05E+01 (1.81E-04)	-9.19E+00 (2.52E+00)	-8.41E+00 (3.13E+00)	-4.49E+00 (1.96E+00)	-9.37E+00 (2.75E+00)	-3.10E+00 (2.37E+00)	-2.46E+00 (1.19E+00)

Table 3.22: Results of composite benchmark test functions.

F	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_{24}	2.33E+02 (9.22E+01)	2.30E+02 (1.37E+02)	8.39E+01 (8.42E+01)	6.00E+01 (8.94E+01)	1.40E+02 (1.52E+02)	1.20E+02 (3.11E+01)	4.49E-17 (2.56E-17)	5.97E+02 (1.34E+02)	4.37E+01 (2.09E+01)
F_{25}	4.10E+01 (3.09E+01)	4.08E+02 (9.36E+01)	1.48E+02 (3.78E+01)	2.44E+02 (1.73E+02)	2.50E+02 (1.44E+02)	1.14E+02 (1.84E+00)	2.03E+02 (4.47E+02)	4.09E+02 (2.10E+01)	1.30E+02 (3.34E+00)
F_{26}	3.39E+02 (5.03E+01)	3.39E+02 (3.14E+01)	3.53E+02 (5.88E+01)	3.39E+02 (8.36E+01)	4.05E+02 (1.67E+02)	3.89E+02 (5.41E+01)	3.67E+02 (8.38E+01)	9.30E+02 (8.31E+01)	5.07E+02 (4.70E+01)
F_{27}	3.44E+02 (2.17E+01)	7.26E+02 (1.21E+02)	4.23E+02 (1.14E+02)	4.49E+02 (1.42E+02)	3.77E+02 (1.28E+02)	4.31E+02 (2.94E+01)	5.32E+02 (1.01E+02)	4.97E+02 (3.24E+01)	3.48E+02 (2.07E+02)
F_{28}	1.46E+02 (7.06E+01)	1.06E+02 (1.38E+01)	1.36E+02 (2.13E+02)	2.40E+02 (4.25E+02)	2.45E+02 (9.96E+01)	1.56E+02 (8.30E+01)	1.44E+02 (1.31E+02)	1.90E+02 (5.03E+01)	8.43E+02 (6.94E+02)
F_{29}	7.43E+02 (9.09E+02)	5.97E+02 (4.98E+00)	8.26E+02 (1.74E+02)	8.22E+02 (1.80E+02)	8.33E+02 (1.68E+02)	6.06E+02 (1.66E+02)	8.13E+02 (1.13E+02)	6.65E+02 (3.37E+02)	7.37E+02 (2.76E+01)

Table 3.23: Results of IEEE CEC-2015 benchmark test functions.

F	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
$CEC-1$	1.50E+05 (1.21E+06)	2.28E+06 (2.18E+06)	2.02E+06 (2.08E+06)	4.37E+05 (4.73E+05)	1.47E+06 (2.63E+06)	6.06E+05 (5.02E+05)	7.65E+06 (3.07E+06)	3.20E+07 (8.37E+06)	8.89E+06 (6.95E+06)
$CEC-2$	6.70E+06 (1.34E+08)	3.13E+05 (4.19E+05)	5.65E+06 (6.03E+06)	9.41E+03 (1.08E+04)	1.97E+04 (1.46E+04)	1.43E+04 (1.03E+04)	7.33E+08 (2.33E+08)	4.58E+03 (1.09E+03)	2.97E+05 (2.85E+03)
$CEC-3$	3.20E+02 (1.16E-03)	3.20E+02 (3.76E-02)	3.20E+02 (7.08E-02)	3.20E+02 (8.61E-02)	3.20E+02 (9.14E-02)	3.20E+02 (3.19E-02)	3.20E+02 (7.53E-02)	3.20E+02 (1.11E-05)	3.20E+02 (2.78E-02)
$CEC-4$	4.10E+02 (5.61E+01)	4.11E+02 (1.71E+01)	4.16E+02 (1.03E+01)	4.09E+02 (3.96E+00)	4.26E+02 (1.17E+01)	4.18E+02 (1.03E+01)	4.42E+02 (7.72E+00)	4.39E+02 (7.25E+00)	6.99E+02 (6.43E+00)
$CEC-5$	9.81E+02 (2.06E+02)	9.13E+02 (1.85E+02)	9.20E+02 (1.78E+02)	8.65E+02 (2.16E+02)	1.33E+03 (3.45E+02)	1.09E+03 (2.81E+02)	1.76E+03 (2.30E+02)	1.75E+03 (2.79E+02)	1.26E+03 (1.86E+02)
$CEC-6$	2.05E+03 (1.05E+04)	1.29E+04 (1.15E+04)	2.26E+04 (2.45E+04)	1.86E+03 (1.93E+03)	7.35E+03 (3.82E+03)	3.82E+03 (2.44E+03)	2.30E+04 (2.41E+04)	3.91E+06 (2.70E+06)	2.91E+05 (1.67E+05)
$CEC-7$	7.02E+02 (5.50E-01)	7.02E+02 (6.76E-01)	7.02E+02 (7.07E-01)	7.02E+02 (7.75E-01)	7.02E+02 (1.10E+00)	7.02E+02 (9.40E-01)	7.06E+02 (9.07E-01)	7.08E+02 (1.32E+00)	7.08E+02 (2.97E+00)
$CEC-8$	1.47E+03 (2.34E+03)	1.86E+03 (1.98E+03)	3.49E+03 (2.04E+03)	3.43E+03 (2.77E+03)	9.93E+03 (8.74E+03)	2.58E+03 (1.61E+03)	6.73E+03 (3.36E+03)	6.07E+05 (4.81E+05)	5.79E+04 (2.76E+04)
$CEC-9$	1.00E+03 (1.51E+01)	1.00E+03 (1.43E-01)	1.00E+03 (1.28E-01)	1.00E+03 (7.23E-02)	1.00E+03 (2.20E-01)	1.00E+03 (5.29E-02)	1.00E+03 (9.79E-01)	1.00E+03 (5.33E+00)	1.00E+03 (3.97E+00)
$CEC-10$	1.23E+03 (2.51E+04)	2.00E+03 (2.73E+03)	4.00E+03 (2.82E+03)	3.27E+03 (1.84E+03)	8.39E+03 (1.12E+04)	2.62E+03 (1.78E+03)	9.91E+03 (8.83E+03)	3.42E+05 (1.74E+05)	4.13E+04 (2.39E+04)
$CEC-11$	1.35E+03 (1.41E+01)	1.38E+03 (2.42E+01)	1.40E+03 (5.81E+01)	1.35E+03 (1.12E+02)	1.37E+03 (8.97E+01)	1.39E+03 (5.42E+01)	1.35E+03 (1.11E+02)	1.41E+03 (7.73E+01)	1.36E+03 (5.39E+01)
$CEC-12$	1.30E+03 (7.50E+00)	1.30E+03 (7.89E-01)	1.30E+03 (6.69E-01)	1.30E+03 (6.94E-01)	1.30E+03 (9.14E-01)	1.30E+03 (8.07E-01)	1.31E+03 (1.54E+00)	1.31E+03 (2.05E+00)	1.31E+03 (1.65E+00)
$CEC-13$	1.30E+03 (6.43E-05)	1.30E+03 (2.76E-04)	1.30E+03 (1.92E-04)	1.30E+03 (5.44E-03)	1.30E+03 (1.04E-03)	1.30E+03 (2.43E-04)	1.30E+03 (3.78E-03)	1.35E+03 (4.70E+01)	1.35E+03 (3.97E+01)
$CEC-14$	3.22E+03 (2.12E+03)	4.25E+03 (1.73E+03)	7.29E+03 (2.45E+03)	7.10E+03 (3.12E+03)	7.60E+03 (1.29E+03)	7.34E+03 (2.47E+03)	7.51E+03 (1.52E+03)	9.30E+03 (4.04E+02)	8.96E+03 (6.32E+03)
$CEC-15$	1.60E+03 (5.69E+01)	1.60E+03 (3.76E+00)	1.61E+03 (4.94E+00)	1.60E+03 (2.66E-07)	1.61E+03 (1.13E+01)	1.60E+03 (1.80E-02)	1.62E+03 (3.64E+00)	1.64E+03 (1.12E+01)	1.63E+03 (3.67E+01)

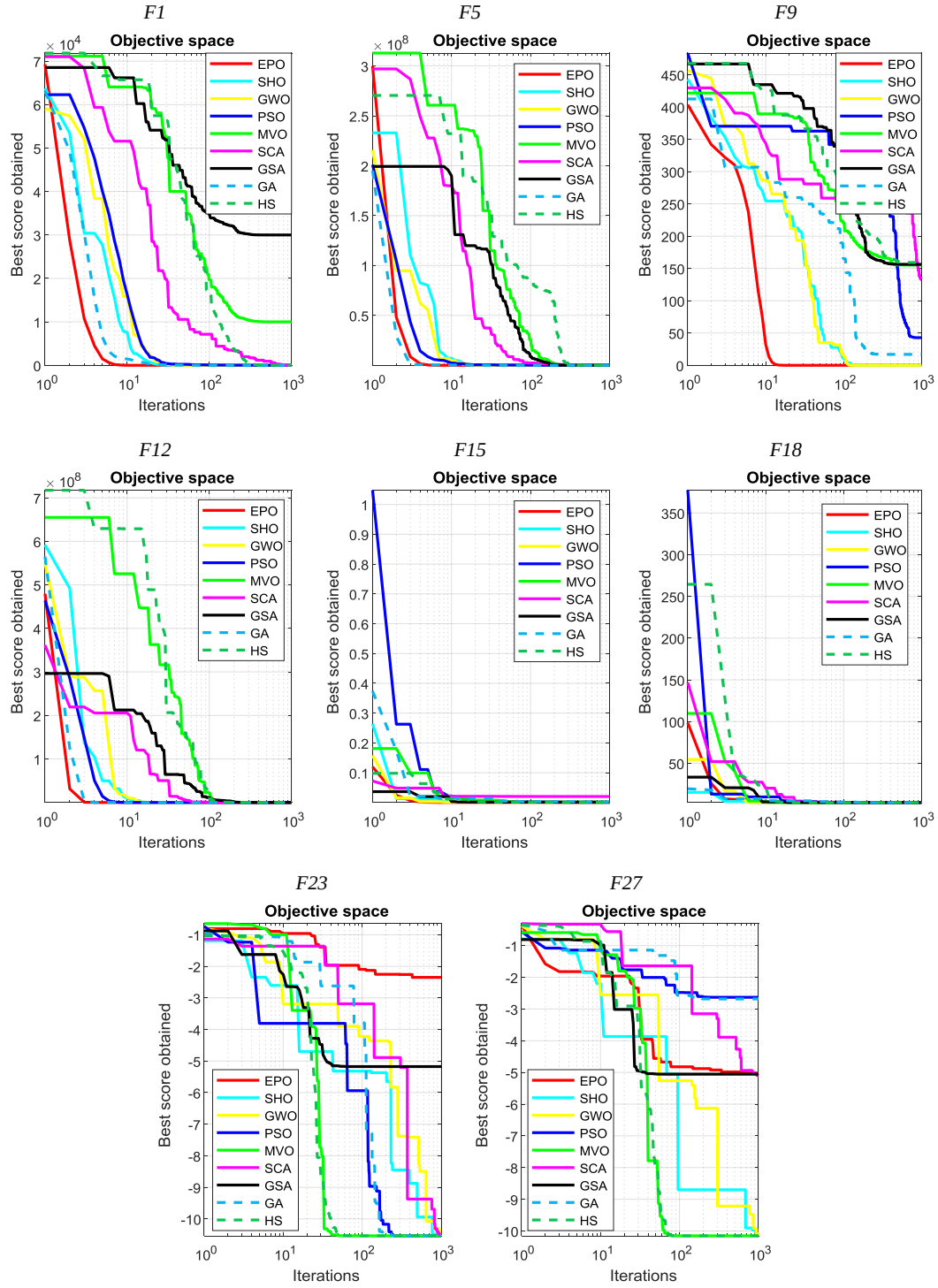


Fig. 3.19. Convergence analysis between the proposed EPO and other competitive algorithms on benchmark test problems.

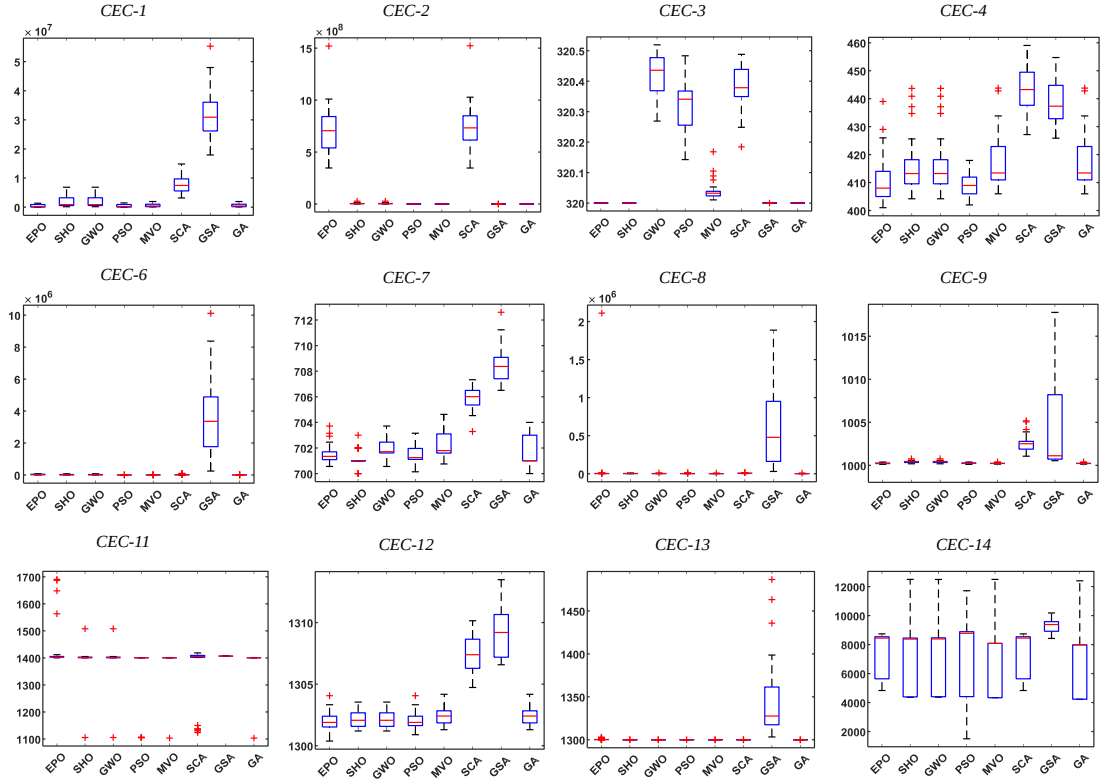


Fig. 3.20. Box plot obtained from the proposed EPO and other competitor algorithms on IEEE CEC-2015 benchmark test functions.

3.2.2.6 Sensitivity Analysis

The proposed EPO algorithm involves four parameters: maximum number of iterations, number of search agents, parameter f , and parameter l . The sensitivity investigation of these parameters has been discussed by varying their values and keeping other parameters fixed.

1. Maximum number of iterations: EPO algorithm was run for the different number of iterations. The values of $Max_{iteration}$ used in experimentation are 100, 500, 800, and 1000. Table 3.24 and Fig. 3.21(a) show the effect of number of iterations over benchmark test functions. The results show that EPO converges towards the optimum when the number of iterations is increased.

2. Number of search agents: To investigate the effect of the number of search agents on benchmark test functions, EPO algorithm was executed for 30, 50, 80, and 100. Table 3.25 and Fig. 3.21(b) show the effect of the number of search agents on benchmark test functions. The results show that EPO provides best optimal solutions when the number of search agent is set to 80.
3. Variation in parameter M : The proposed EPO was executed for different values of M keeping other parameters fixed, i.e., the maximum number of iterations and number of search agents. The values of parameter M are varied from [1,2,3,4]. Table 3.26 and Fig. 3.21(c) show the effect of M over different benchmark test functions. The results demonstrate that EPO achieves near optimal solutions when the value of M is set to 2.
4. Variation in parameter f : EPO algorithm was run for different values of f keeping other parameters fixed, i.e., the maximum number of iterations and number of search agents. The values of f used in experimentation are lies in the ranges of [1,2], [2,3], [3,4], and [4,5]. Table 3.27 and Fig. 3.21(d) show the effect of f over benchmark test functions. The results reveal that EPO provides better results when the value of f is set between the range of [2,3].
5. Variation in parameter l : The values used in experimentation for l are lie between the ranges of [0.5,1], [1,1.5], [1.5,2], and [2,2.5]. Table 3.28 and Fig. 3.21(e) show the effect of l on different benchmark test functions. It is been observed that EPO provides optimal results when the value of l is fixed between the range of [1.5,2].

Table 3.24: The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions using different simulation runs (i.e., 100, 500, 800, and 1000).

Iterations	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	F_{24}	$CEC - 1$
100	2.71E-17	7.07E+00	3.21E-03	9.07E-01	-2.11E+00	3.53E+02	2.60E+06
500	5.44E-20	6.01E+00	1.20E-03	7.97E-01	-2.97E+00	3.15E+02	2.56E+05
800	4.00E-25	5.52E+00	7.22E-04	6.12E-01	-3.11E+00	2.95E+02	2.01E+05
1000	2.42E-29	5.04E+00	4.20E-05	3.88E-01	-3.47E+00	2.35E+02	1.50E+05

Table 3.25: The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations is fixed as 1000. The number of search agents are varied from 30 to 100.

Search agents	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	F_{24}	$CEC - 1$
30	1.51E-16	7.01E+00	3.11E-03	5.27E-01	-2.53E+00	3.33E+02	2.86E+06
50	5.43E-19	6.31E+00	2.30E-03	7.20E-01	-2.80E+00	3.20E+02	2.22E+05
80	2.22E-28	5.00E+00	4.24E-05	3.22E-01	-3.49E+00	2.29E+02	1.47E+05
100	5.97E-23	5.42E+00	8.32E-04	6.18E-01	-3.13E+00	2.96E+02	2.18E+05

Table 3.26: The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter M is varied from [1, 2, 3, 4].

M	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	F_{24}	$CEC - 1$
1	4.41E-17	6.10E+00	2.41E-04	5.76E-01	-2.21E+00	3.36E+02	2.53E+06
2	4.00E-22	5.13E+00	1.19E-06	4.87E-02	-3.50E+00	2.24E+02	1.47E+05
3	7.48E-17	7.83E+00	2.40E-03	5.26E-01	-2.93E+00	4.02E+02	3.43E+05
4	1.76E-19	7.02E+00	4.26E-04	5.49E-01	-3.08E+00	2.76E+02	2.19E+05

Table 3.27: The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter f is varied from [1, 2] to [4, 5].

f	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	F_{24}	$CEC - 1$
[1, 2]	6.23E-18	7.05E+00	3.34E-03	8.79E-01	-2.14E+00	3.47E+02	2.53E+06
[2, 3]	2.22E-29	5.03E+00	4.22E-05	3.85E-01	-3.50E+00	2.33E+02	1.48E+05
[3, 4]	3.42E-19	6.00E+00	1.17E-03	7.86E-01	-2.89E+00	3.11E+02	2.50E+05
[4, 5]	4.00E-23	5.60E+00	7.40E-04	6.10E-01	-3.17E+00	2.93E+02	2.07E+05

Table 3.28: The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions where number of iterations and search agents are fixed as 1000 and 80, respectively. The parameter f is fixed as $[2, 3]$. The parameter l is varied from $[0.5, 1]$ to $[2, 2.5]$.

l	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	F_{24}	$CEC-1$
$[0.5, 1]$	5.46E-19	7.18E+00	3.42E-03	8.92E-01	-2.23E+00	3.40E+02	2.55E+06
$[1, 1.5]$	1.41E-21	5.86E+00	1.34E-03	7.76E-01	-2.94E+00	3.19E+02	2.75E+05
$[1.5, 2]$	2.28E-28	5.01E+00	4.13E-05	3.80E-01	-3.50E+00	2.31E+02	1.51E+05
$[2, 2.5]$	8.95E-24	5.55E+00	7.43E-04	6.00E-01	-3.07E+00	2.80E+02	2.00E+05

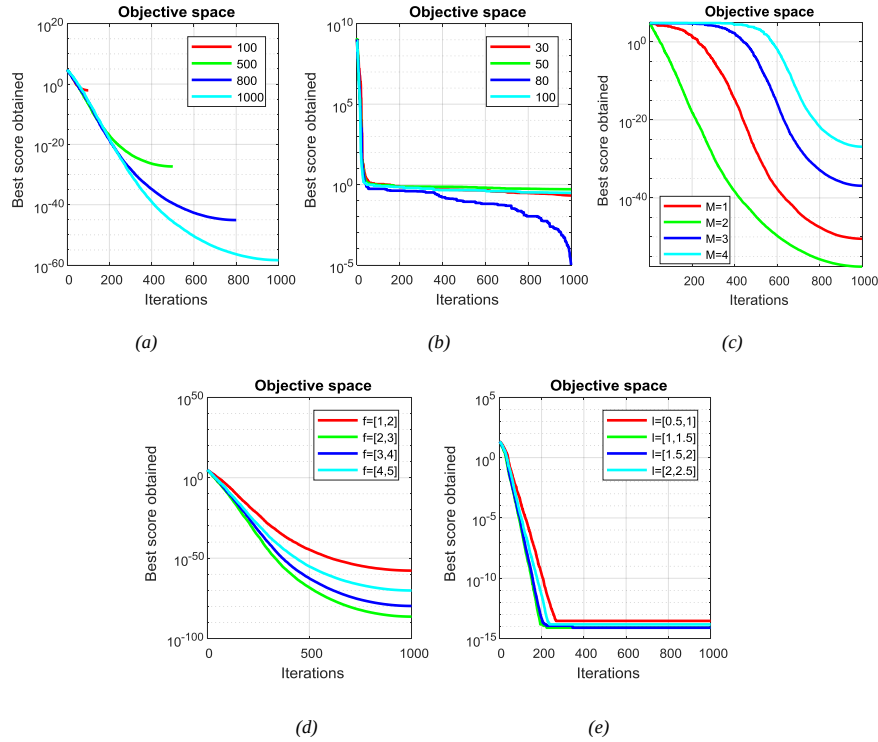


Fig. 3.21. Sensitivity analysis of proposed EPO algorithm, (a) Number of iterations; (b) Number of search agents; (c) Control parameter M ; (d) Control parameter f ; (e) Control parameter l .

3.2.2.7 Scalability Study

This subsection describes the effect of scalability is tested on various test functions using the proposed EPO. The dimensionality of the test functions varies between different ranges. Fig. 3.22 shows the performance of EPO algorithm on scalable benchmark test functions. It is observed that the performance of EPO is not too much degraded when the dimensionality of search space is increased. The results reveal that the performance of EPO is least affected by the increase in the dimensionality of search space.

3.2.2.8 Statistical Testing

Apart from standard statistical analysis such as mean and standard deviation, ANOVA test has been conducted. ANOVA test is used to determine whether the results obtained from proposed algorithm are different from other competitor algorithms in a statistically significant way. The sample size for ANOVA test is 30 with 95% confidence of interval. The results of ANOVA test are tabulated in Table 3.29 which shows that EPO is statistically significant from other algorithms for all benchmark test functions.

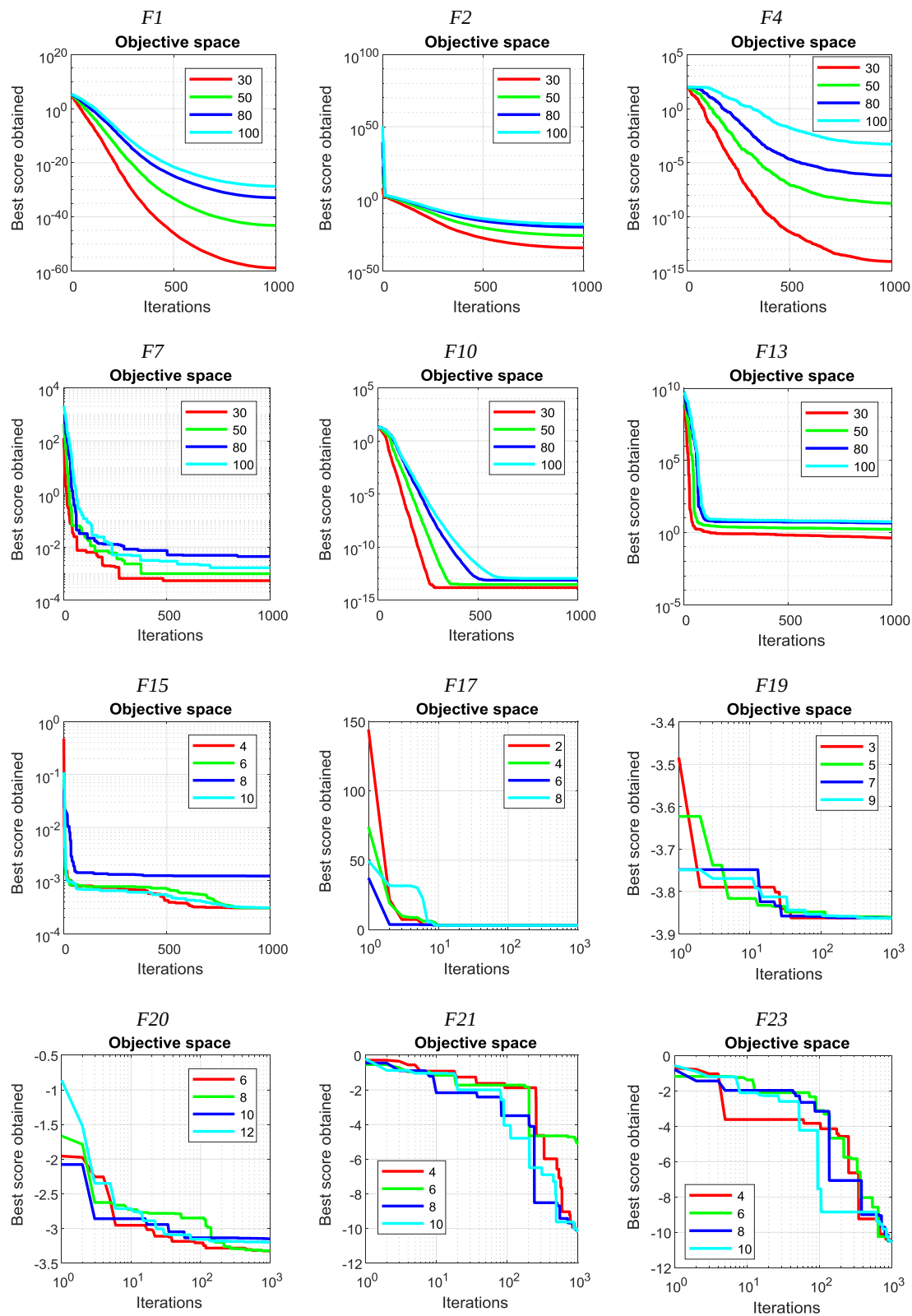


Fig. 3.22. Effect of scalability on the performance of EPO algorithm.

Table 3.29: ANOVA test results.

F	p -value	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_1	2.72E-24	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_2	3.60E-21	SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA
F_3	5.53E-30	SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, GWO, PSO, SCA, GSA, GA	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA
F_4	1.80E-74	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_5	2.11E-11	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_6	4.97E-19	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, HS	EPO, SHO, GWO, PSO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA
F_7	1.13E-85	SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GA, HS	EPO, SHO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, MVO, SCA, GSA	EPO, SHO, GWO, PSO, SCA, GSA, GA
F_8	7.56E-14	SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, PSO, MVO, SCA, GA, HS	EPO, SHO, MVO, SCA, GSA, GA	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, GSA, HS	EPO, SHO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, PSO, MVO, SCA
F_9	1.00E-95	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, SCA, GSA, GA, HS	EPO, SHO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA
F_{10}	1.21E-81	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, GSA	EPO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, SCA	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{11}	9.36E-07	SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA

(Continued...)

F	p -value	EPO	SHO	GWO	PSO	MVO	SCA	GSA	GA	HS
F_{12}	2.99E-53	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA
F_{13}	4.90E-38	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, SCA, GSA, GA, HS	EPO, SHO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA
F_{14}	4.34E-40	SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA
F_{15}	3.15E-37	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA	EPO, SHO, GWO, PSO, MVO, GA, HS	EPO, SHO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA	EPO, SHO, GWO, PSO, MVO, SCA
F_{16}	2.07E-10	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA	EPO, SHO, GWO, PSO, MVO, SCA	EPO, SHO, GWO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA
F_{17}	1.70E-76	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GA, HS	EPO, SHO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA
F_{18}	4.69E-32	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, SCA, GSA, HS	EPO, SHO, GWO, PSO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, PSO, MVO, SCA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{19}	4.46E-39	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, HS	EPO, SHO, PSO, SCA, GSA, HS	EPO, GWO, PSO, MVO, SCA, GSA, GA
F_{20}	4.90E-109	SHO, GWO, PSO, MVO, SCA, GSA	EPO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, GSA, GA	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{21}	2.11E-39	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GA	EPO, SHO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{22}	6.13E-68	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, GSA, GA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, PSO, SCA, GSA, GA, HS	EPO, SHO, PSO, MVO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA
F_{23}	2.19E-39	SHO, GWO, PSO, MVO, SCA, GSA, GA, HS	EPO, GWO, PSO, MVO, SCA, GSA, HS	EPO, SHO, PSO, MVO, SCA, GSA, GA, HS	EPO, SHO, GWO, MVO, SCA, GSA, GA, HS	EPO, SHO, PSO, GSA, GA, HS	EPO, SHO, GWO, PSO, MVO, GSA, GA	EPO, SHO, GWO, MVO, SCA, GA, HS	EPO, SHO, GWO, PSO, MVO, SCA, GSA	EPO, SHO, GWO, PSO, MVO, SCA, GSA, GA

Table 3.30: Comparison results for pressure vessel design problem.

Algorithms	Optimum variables				Optimum cost
	T_s	T_h	R	L	
EPO	0.778099	0.383241	40.315121	200.00000	5880.0700
SHO	0.778210	0.384889	40.315040	200.00000	5885.5773
GWO	0.779035	0.384660	40.327793	199.65029	5889.3689
PSO	0.778961	0.384683	40.320913	200.00000	5891.3879
MVO	0.845719	0.418564	43.816270	156.38164	6011.5148
SCA	0.817577	0.417932	41.74939	183.57270	6137.3724
GSA	1.085800	0.949614	49.345231	169.48741	11550.2976
GA	0.752362	0.399540	40.452514	198.00268	5890.3279
HS	1.099523	0.906579	44.456397	179.65887	6550.0230

3.2.2.9 Applications of EPO on Engineering Design Problems

In this section, the proposed EPO algorithm has been applied to five constrained and one unconstrained engineering design problems. These are pressure vessel, speed reducer, welded beam, tension/compression spring, rolling element bearing, and displacement of loaded structure. All of these optimization problems have different constraints with different nature. EPO algorithm uses death penalty function to handle both constrained and unconstrained optimization problems.

3.2.2.9.1 Pressure Vessel Design Problem

This problem was first proposed by Kannan and Kramer^[144] to minimize the total fabrication cost (see Fig. 3.10). The mathematical formulation of this problem is formulated in Appendix C.1.1. Table 3.30 shows the comparison of best obtained optimal solution from EPO and other algorithms such as SHO, GWO, PSO, MVO, SCA, GSA, GA, and HS. The proposed EPO provides optimal solution at $z^* = (0.778099, 0.383241, 40.315121, 200.00000)$ with corresponding fitness value equal to $f(z^*) = 5880.0700$. From this table, it can be concluded that EPO is able to find the best optimal design with minimum cost. The statistical results of pressure vessel design problem are presented in Table 3.31. The results reveal that EPO performs better than the other competitor algorithms in terms of best, mean, and median.

Table 3.31: Statistical results obtained from different algorithms for pressure vessel design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	5880.0700	5884.1401	5891.3099	024.341	5883.5153
SHO	5885.5773	5887.4441	5892.3207	002.893	5886.2282
GWO	5889.3689	5891.5247	5894.6238	013.910	5890.6497
PSO	5891.3879	6531.5032	7394.5879	534.119	6416.1138
MVO	6011.5148	6477.3050	7250.9170	327.007	6397.4805
SCA	6137.3724	6326.7606	6512.3541	126.609	6318.3179
GSA	11550.2976	23342.2909	33226.2526	5790.625	24010.0415
GA	5890.3279	6264.0053	7005.7500	496.128	6112.6899
HS	6550.0230	6643.9870	8005.4397	657.523	7586.0085

Table 3.32: Comparison results for speed reducer design problem.

Algorithms	Optimum variables							Optimum cost
	b	m	p	l_1	l_2	d_1	d_2	
EPO	3.50123	0.7	17	7.3	7.8	3.33421	5.26536	2994.2472
SHO	3.50159	0.7	17	7.3	7.8	3.35127	5.28874	2998.5507
GWO	3.506690	0.7	17	7.380933	7.815726	3.357847	5.286768	3001.288
PSO	3.500019	0.7	17	8.3	7.8	3.352412	5.286715	3005.763
MVO	3.508502	0.7	17	7.392843	7.816034	3.358073	5.286777	3002.928
SCA	3.508755	0.7	17	7.3	7.8	3.461020	5.289213	3030.563
GSA	3.600000	0.7	17	8.3	7.8	3.369658	5.289224	3051.120
GA	3.510253	0.7	17	8.35	7.8	3.362201	5.287723	3067.561
HS	3.520124	0.7	17	8.37	7.8	3.366970	5.288719	3029.002

3.2.2.9.2 Speed Reducer Design Problem

The speed reducer design problem is a challenging benchmark due to its seven design variables^[145] as shown in Fig. 3.11. The main objective of this problem is to minimize the weight of speed reducer. The mathematical formulation of this problem is formulated in Appendix C.1.2. The comparison of the best obtained optimal solution with various optimization algorithms is given in Table 3.32. The proposed EPO provides optimal solution at $z^* = (3.50123, 0.7, 17, 7.3, 7.8, 3.33421, 5.26536)$ with corresponding fitness value equal to $f(z^*) = 2994.2472$. However, the statistical results of proposed and eight competitive optimization algorithms including SHO, GWO, PSO, MVO, SCA, GSA, GA, and HS are given in Table 3.33. The results indicate that EPO outperforms than other competitor algorithms.

Table 3.33: Statistical results obtained from different algorithms for speed reducer design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	2994.2472	2997.482	2999.092	1.78091	2996.318
SHO	2998.5507	2999.640	3003.889	1.93193	2999.187
GWO	3001.288	3005.845	3008.752	5.83794	3004.519
PSO	3005.763	3105.252	3211.174	79.6381	3105.252
MVO	3002.928	3028.841	3060.958	13.0186	3027.031
SCA	3030.563	3065.917	3104.779	18.0742	3065.609
GSA	3051.120	3170.334	3363.873	92.5726	3156.752
GA	3067.561	3186.523	3313.199	17.1186	3198.187
HS	3029.002	3295.329	3619.465	57.0235	3288.657

Table 3.34: Comparison results for welded beam design problem.

Algorithms	Optimum variables				Optimum cost
	h	l	t	b	
EPO	0.205411	3.472341	9.035215	0.201153	1.723589
SHO	0.205563	3.474846	9.035799	0.205811	1.725661
GWO	0.205678	3.475403	9.036964	0.206229	1.726995
PSO	0.197411	3.315061	10.00000	0.201395	1.820395
MVO	0.205611	3.472103	9.040931	0.205709	1.725472
SCA	0.204695	3.536291	9.004290	0.210025	1.759173
GSA	0.147098	5.490744	10.00000	0.217725	2.172858
GA	0.164171	4.032541	10.00000	0.223647	1.873971
HS	0.206487	3.635872	10.00000	0.203249	1.836250

3.2.2.9.3 Welded Beam Design Problem

The objective of this design problem is to minimize the fabrication cost of welded beam (see Fig. 3.8). The mathematical formulation of this problem is formulated in Appendix C.1.3. The comparison for best solution obtained from proposed EPO and other algorithms is presented in Table 3.34. Among other algorithms, the proposed EPO provides optimal solution at $z^* = (0.205411, 3.472341, 9.035215, 0.201153)$ with corresponding fitness value equal to $f(z^*) = 1.723589$. The results indicate that EPO converges towards the best design. The statistical comparison of the proposed algorithm with other competitor algorithms is depicted in Table 3.35. It is observed from Table 3.35 that EPO surpassed other algorithms for providing the best solution in terms of best, mean, and median.

Table 3.35: Statistical results obtained from different algorithms for welded beam design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	1.723589	1.725124	1.727211	0.004325	1.724399
SHO	1.725661	1.725828	1.726064	0.000287	1.725787
GWO	1.726995	1.727128	1.727564	0.001157	1.727087
PSO	1.820395	2.230310	3.048231	0.324525	2.244663
MVO	1.725472	1.729680	1.741651	0.004866	1.727420
SCA	1.759173	1.817657	1.873408	0.027543	1.820128
GSA	2.172858	2.544239	3.003657	0.255859	2.495114
GA	1.873971	2.119240	2.320125	0.034820	2.097048
HS	1.836250	1.363527	2.035247	0.139485	1.9357485

Table 3.36: Comparison results for tension/compression spring design problem.

Algorithms	Optimum variables			Optimum cost
	d	D	P	
EPO	0.051087	0.342908	12.0898	0.012656987
SHO	0.051144	0.343751	12.0955	0.012674000
GWO	0.050178	0.341541	12.07349	0.012678321
PSO	0.05000	0.310414	15.0000	0.013192580
MVO	0.05000	0.315956	14.22623	0.012816930
SCA	0.050780	0.334779	12.72269	0.012709667
GSA	0.05000	0.317312	14.22867	0.012873881
GA	0.05010	0.310111	14.0000	0.013036251
HS	0.05025	0.316351	15.23960	0.012776352

3.2.2.9.4 Tension/compression Spring Design Problem

The main objective of this problem is to minimize the tension/ compression spring weight as shown in Fig. 3.9. The mathematical formulation of this problem is formulated in Appendix C.1.4. The comparison for the best solution obtained from the proposed EPO and other competitor algorithms is presented in Table 3.36. The best solution was obtained by EPO at design variables $z^* = (0.051087, 0.342908, 12.0898)$ with an objective function value of $f(z^*) = 0.012656987$. The results reveal that EPO again performs better than other optimization algorithms. The statistical results of tension/compression spring design are compared and tabulated in Table 3.37. From Table 3.37, it can be seen that provides better statistical results than the other algorithms in terms of best, mean, and median.

Table 3.37: Statistical results obtained from different algorithms for tension/compression spring design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	0.012656987	0.012678903	0.012667902	0.001021	0.012676002
SHO	0.012674000	0.012684106	0.012715185	0.000027	0.012687293
GWO	0.012678321	0.012697116	0.012720757	0.000041	0.012699686
PSO	0.013192580	0.014817181	0.017862507	0.002272	0.013192580
MVO	0.012816930	0.014464372	0.017839737	0.001622	0.014021237
SCA	0.012709667	0.012839637	0.012998448	0.000078	0.012844664
GSA	0.012873881	0.013438871	0.014211731	0.000287	0.013367888
GA	0.013036251	0.014036254	0.016251423	0.002073	0.013002365
HS	0.012776352	0.013069872	0.015214230	0.000375	0.012952142

Table 3.38: Comparison results for rolling element bearing design problem.

Algorithms	Optimum variables										Opt. cost
	D_m	D_b	Z	f_i	f_o	K_{Dmin}	K_{Dmax}	ε	e	ζ	
EPO	125	21.41890	10.94113	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85067.983
SHO	125	21.40732	10.93268	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85054.532
GWO	125.6199	21.35129	10.98781	0.515	0.515	0.5	0.68807	0.300151	0.03254	0.62701	84807.111
PSO	125	20.75388	11.17342	0.515	0.515000	0.5	0.61503	0.300000	0.05161	0.60000	81691.202
MVO	125.6002	21.32250	10.97338	0.515	0.515000	0.5	0.68782	0.301348	0.03617	0.61061	84491.266
SCA	125	21.14834	10.96928	0.515	0.515	0.5	0.7	0.3	0.02778	0.62912	83431.117
GSA	125	20.85417	11.14989	0.515	0.517746	0.5	0.61827	0.304068	0.02000	0.624638	82276.941
GA	125	20.77562	11.01247	0.515	0.515000	0.5	0.61397	0.300000	0.05004	0.610001	82773.982
HS	125	20.87123	11.16697	0.515	0.516000	0.5	0.61951	0.301128	0.05024	0.614531	81569.527

3.2.2.9.5 Rolling Element Bearing Design Problem

The objective of this problem is to maximize the dynamic load carrying capacity of a rolling element bearing as demonstrated in Fig. 3.12. The mathematical formulation of this problem is formulated in Appendix C.1.5. Table 3.38 shows the performance comparison of best optimal solution obtained from several algorithms. The proposed EPO provides optimal solution at $z^* = (125, 21.41890, 10.94113, 0.515, 0.515, 0.4, 0.7, 0.3, 0.02, 0.6)$ with corresponding fitness value equal to $f(z^*) = 85067.983$. The statistical results obtained for rolling element bearing design problem are compared and shown in Table 3.39. It reveals that EPO is capable to achieve the best optimal solution.

3.2.2.9.6 Displacement of Loaded Structure Problem

Displacement is a vector which defines the shortest distance between the initial and the final position of a given point. The main objective of this problem is to

Table 3.39: Statistical results obtained from different algorithms for rolling element bearing design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	85067.983	85042.352	86551.599	1877.09	85056.095
SHO	85054.532	85024.858	85853.876	0186.68	85040.241
GWO	84807.111	84791.613	84517.923	0137.186	84960.147
PSO	81691.202	50435.017	32761.546	13962.150	42287.581
MVO	84491.266	84353.685	84100.834	0392.431	84398.601
SCA	83431.117	81005.232	77992.482	1710.777	81035.109
GSA	82276.941	78002.107	71043.110	3119.904	78398.853
GA	82773.982	81198.753	80687.239	1679.367	8439.728
HS	81569.527	80397.998	79412.779	1756.902	8347.009

Table 3.40: Comparison results for displacement of loaded structure problem.

Algorithms	Optimum cost (π)
EPO	168.8231
SHO	168.8889
GWO	170.3645
PSO	170.5960
MVO	169.3023
SCA	169.0032
GSA	176.3697
GA	171.3674
HS	172.0324

minimize the potential energy for reducing the excess load of structure as shown in Fig. 3.13.

Table 3.40 shows the comparison of best optimal solution obtained from EPO and other competitor approaches such as SHO, GWO, PSO, MVO, SCA, GSA, GA, and HS. The proposed EPO provides best optimum cost at $\pi = 168.8231$. The results reveal that EPO is able to minimize the potential energy for loaded structure problem.

The statistical results for the reported algorithms are depicted in Table 3.41. From Table 3.41, it is noticed that the results obtained from EPO are far better than the other algorithms in terms of best, mean, and median.

Table 3.41: Statistical results obtained from different algorithms for displacement of loaded structure problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
EPO	168.8231	170.1309	173.9721	211.861	169.4214
SHO	168.8889	170.3659	173.6357	023.697	169.6710
GWO	170.3645	171.3694	174.3970	196.037	173.3694
PSO	170.5960	174.6354	175.3602	236.036	173.9634
MVO	169.3023	171.0034	174.3047	202.753	170.0032
SCA	169.0032	171.7530	174.4527	129.047	170.3647
GSA	176.3697	178.7521	179.5637	113.037	174.367
GA	171.3674	172.0374	174.0098	212.703	172.0097
HS	172.0324	173.4327	174.0399	080.111	171.3697

3.3 Multi-objective Metaheuristic Technique

The another contribution of this chapter is the development of multi-objective version of SHO algorithm.

3.3.1 Multi-objective Spotted Hyena Optimizer

To develop a multi-objective version of SHO, two components are introduced. These components are very similar to MOPSO^[150]. The first component is an archive which is accountable for storing the best non-dominated Pareto optimal solutions. The second component is a group selection approach to choose nearby solutions corresponding to the position of prey from the archive. These components are described in the preceding subsections.

3.3.1.1 Archive

The archive is a storage part of all non-dominated Pareto optimal solutions which are obtained so far. This archive component has an ability to evenly spread around the Pareto front if it is concave, convex, and disconnected. It consists of two main parts such as archive controller and grid.

3.3.1.1.1 *The Archive Controller*

The main function of this controller is to decide whether the solution should be added or not in the archive. There are some important points regarding the archive update mechanism which are given below:

- The solution should not be allowed to enter the archive when it is dominated by at least one of the archive members.
- When a new solution dominates one or more members in the archive, the new solution will be able to enter the archive after omitting the dominated solution(s) in the archive.
- The new member should be added in the archive when neither the new solution nor archive solutions dominate each other.
- The grid method should be run to omit one of the most crowded solution section and insert new solutions to improve the diversity corresponding to Pareto optimal front when the archive is full.

3.3.1.1.2 *The Grid*

The adaptive grid mechanism is used to produce the distributed Pareto fronts^[151]. The objective function space is divided into different regions as shown in Fig. 3.23. The grid has to be recalculated and relocated each individual if the inserted individual into population lies outside the current bounds of the grid^[150] (see Fig. 3.24). The adaptive grid is a space formed by hypercubes and is used to distribute in a uniform way.

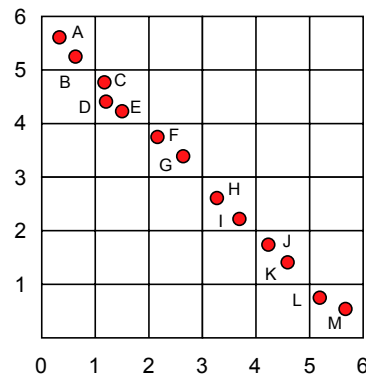


Fig. 3.23. The adaptive grid mechanism of individuals.

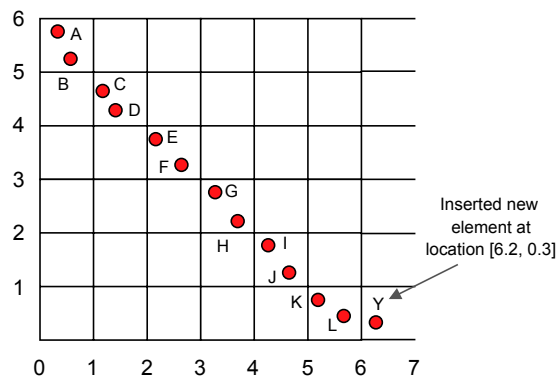


Fig. 3.24. Relocate the individuals if the new element lies outside the boundary of the grid.

3.3.1.1.3 Group Selection Mechanism

In a multi-objective search space, the challenging task is to compare the solutions with archive members. To overcome this issue, a group selection mechanism is designed. The group selection strategy chooses the least crowded section of search space and offers one of its non-dominated solutions to group of nearby solutions as shown in Fig. 3.25. The selection is done on the basis of roulette-wheel method

with probability which is defined as:

$$H_k = \frac{f}{S_k} \quad (3.22)$$

where f is a constant number that should be greater than one and S is the number of Pareto optimal solutions which are obtained so far in the k th segment. The roulette-wheel selection method is a classic selection operator in the proportional type. The fitness value of each individual in the population corresponds to the range on the roulette wheel proportion. In Eq. (3.22), it can be seen that choosing the number of solutions is directly proportional to vector $\vec{M}$. The proposed MOSHO algorithm includes all the characteristics of SHO algorithm. The main difference between MOSHO and SHO is that MOSHO searches in a set of archive members, while SHO saves the group of optimal solutions.

The foremost difference between MOSHO and NSGA-II is archive. The motivation behind the use of archive is to reduce the possibility of deteriorate non-dominated solutions. MOSHO provides optimal solutions as compared to other archive based algorithms. The archive based algorithms usually utilize selection operator such as crossover, mutation, etc. These operators bias the search towards the members of archive. In MOSHO, the variables are exchanged among a solution in the search space. This exchange operation increase the exploration capability of MOSHO. Whereas, it reduces the convergence property. To overcome this problem, MOSHO uses a group selection approach to select at least one non-dominated solution in the given search space.

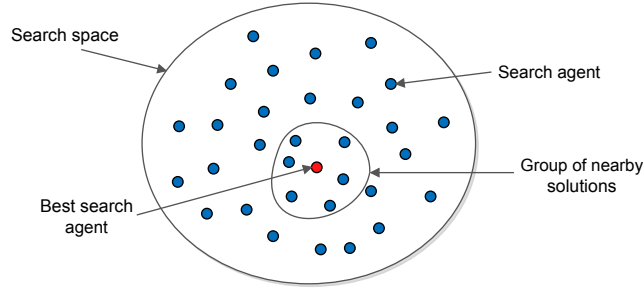


Fig. 3.25. Group selection mechanism.

Algorithm 3 Multi-objective Spotted Hyena Optimizer (MOSHO)

Input: Spotted hyenas population $\vec{P}_i$ ($i = 1, 2, \dots, n$)

Output: Archive of non-dominated optimal solutions

- 1: **procedure** MOSHO
 - 2: Initialize the vectors $\vec{h}, \vec{B}, \vec{E}$, and N_s
 - 3: Calculate the objective values of each search agent
 - 4: Find all the non-dominated solutions and initialize these solutions to archive
 - 5: $\vec{P}_h \leftarrow$ best search agent from archive
 - 6: $\vec{C}_h \leftarrow$ group or cluster of all far optimal solutions with respect to $\vec{P}_h$ (archive)
 - 7: **while** ($x < MaxIteration$) **do**
 - 8: **for** each search agent **do**
 - 9: Update the position of current search agent
 - 10: **end for**
 - 11: Update $\vec{h}, \vec{B}, \vec{E}$, and N_s
 - 12: Calculate the objective values of all search agents
 - 13: Find the non-dominated solutions from updated search agents
 - 14: Update the obtained non-dominated solutions to archive
 - 15: **if** archive is full **then**
 - 16: Grid method should be run to omit one of the most crowded archive members
 - 17: Add new solution to the archive
 - 18: **end if**
 - 19: Check if any search agent goes beyond the search space and then adjust it
 - 20: Calculate the objective values of each search agent
 - 21: Update $\vec{P}_h$ if there is a better solution than the previous optimal solution from archive
 - 22: Update the group $\vec{C}_h$ with respect to $\vec{P}_h$ (archive)
 - 23: $x \leftarrow x + 1$
 - 24: **end while**
 - 25: **return** archive
 - 26: **end procedure**
-

3.3.1.2 Computational Complexity

In this subsection, the computational complexity of proposed algorithm is discussed. The time and space complexities of MOSHO is described below.

3.3.1.2.1 Time Complexity

1. Initialization of MOSHO population needs $\mathcal{O}(n_o \times n_p)$ time, where n_o and n_p represents the number of objectives and population size, respectively.
2. The fitness calculation of each search agent requires $\mathcal{O}(Max_{iterations} \times n_o \times n_p)$ time, where $Max_{iterations}$ is the maximum number of iterations to simulate the proposed MOSHO algorithm.
3. The algorithm requires $\mathcal{O}(M)$ time to define the group of spotted hyenas, where M indicates the counting value of spotted hyenas.
4. It requires $\mathcal{O}(n_o \times (n_{ns} + n_p))$ time to update the archive of non-dominated solutions.
5. Repeat Steps 2 - 4 until the termination criteria is satisfactory.

Therefore, the overall time complexity of proposed MOSHO algorithm is $\mathcal{O}(Max_{iterations} \times n_o \times (n_p + n_{ns}) \times M)$.

The computational complexities corresponding to MOPSO, NSGA-II, PESA-II, SPEA2, and MOACO are $\mathcal{O}(Max_{iterations} \times n_o \times n_p^2)$, $\mathcal{O}(Max_{iterations} \times n_o \times n_p^2)$, $\mathcal{O}(Max_{iterations} \times n_o \times (n_p + n_{ns})^2)$, $\mathcal{O}(Max_{iterations} \times n_o \times n_p^2)$, and $\mathcal{O}(Max_{iterations} \times n_o \times n_p^2)$, respectively. Whereas, the computational complexity of MOEA/D is $\mathcal{O}(Max_{iterations} \times n_o \times n_p \times T)$, where T represents the Tchebycheff approach to decompose the optimization problem into subproblems.

3.3.1.2.2 Space Complexity

The space complexity of MOSHO algorithm is considered during its initialization process which requires space at any one time. Hence, the total space complexity of MOSHO algorithm is $\mathcal{O}(n_o \times n_p)$.

3.3.1.3 Experimental Results and Discussions

This section compares the performance of MOSHO with six recently developed algorithms and validates over 24 benchmark test functions. To generate the results for each benchmark test functions, MOSHO algorithm utilizes 30 independent runs where each run employs 1000 times of iterations.

3.3.1.3.1 Benchmark Test Functions

The twenty-four well-known benchmark test functions are applied on MOSHO algorithm to demonstrate its efficiency. The IEEE CEC-2009 special session benchmark test suit^[152] is used which consists of ten unconstrained test functions as described in Appendix B. These unconstrained functions are able to solve the bound constrained multi-objective problems.

The popular five ZDT^[153] and nine DTLZ^[154] benchmark test suites are also utilized which are described in the Appendices B.1 and B.2, respectively. These unconstrained functions involve a feature to converge the Pareto optimal front very efficiently.

In Appendix B.3, the characteristics of IEEE CEC-2009 benchmark test functions are described. There are ten unconstrained test functions ($UF1 - UF10$) which are included in this test suite. The detailed characteristics of ZDT and DTLZ unconstrained test functions are described in Appendices B.1 and B.2, respectively. There are five ZDT test functions ($ZDT1 - ZDT6$) and nine DTLZ test functions ($DTLZ1 - DTLZ9$).

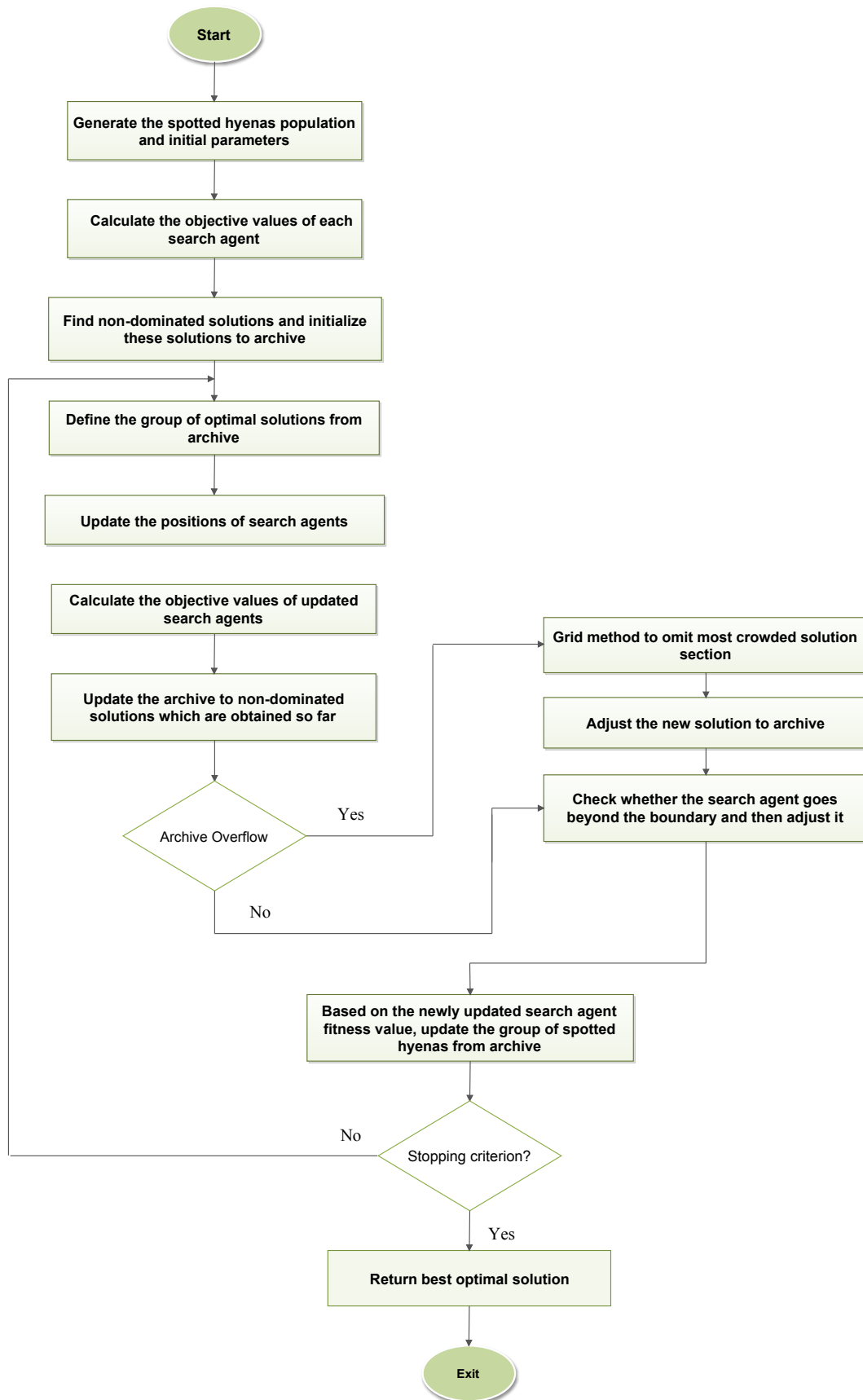


Fig. 3.26. Flowchart of the proposed MOSHO algorithm.

3.3.1.3.2 *Experimental Setup*

To validate the performance of MOSHO algorithm, it is compared with six well-known optimization algorithms such as Multi-objective Particle Swarm Optimization (MOPSO)^[155], Non-dominated Sorting Genetic Algorithm 2 (NSGA-II)^[156], Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D)^[157], Pareto Envelope based Selection Algorithm 2 (PESA-II)^[158], Strength Pareto Evolutionary Algorithm 2 (SPEA2)^[159], and Multi-objective Ant Colony Optimization (MOACO)^[160].

The parameter settings of these algorithms are set as they are recommended in their original papers^[155,156,157,104,159,161]. The population size and the maximum number of iterations for above-mentioned algorithms are set to 100 and 1000, respectively. Due to the stochastic nature of these algorithms, the results are averaged over 30 independent runs under 30 different random seeds. The mean best-of-run solution and standard deviation of best solution in the last iteration are reported in tables.

3.3.1.3.3 *Performance Metrics*

The four well-known performance metrics are used to investigate the performance of above-mentioned algorithms. These are Hypervolume (HV)^[50,51], $\Delta_p(p = 1)$ ^[52,53,54], Spread^[55,51], and Epsilon (ϵ)^[56,51].

The best Pareto optimal solutions set which is obtained by each algorithm is compared with the proposed technique. The above-mentioned algorithms run 30 times on 24 benchmark test problems.

3.3.1.3.4 Performance Evaluation

This subsection discuss the results of proposed MOSHO algorithm on IEEE CEC-2009, ZDT, and DTLZ benchmark test suites.

- Results on IEEE CEC-2009 ($UF1 - UF10$) benchmark test functions

Table 3.42 shows the performance comparison of the proposed MOSHO with the above-mentioned existing techniques for IEEE CEC-2009 benchmark test problems. The results reveal that MOSHO is able to determine the optimal solutions for $UF1, UF2, UF4, UF5$, and $UF6$ test functions. For $UF3$ test function, MOPSO provides better value of Spread than the others. MOEA/D provides better results than the other competitive algorithms in terms of Δ_p and Epsilon. For $UF7, UF9$, and $UF10$ test functions, MOSHO provides competitive results in comparison with the others. For $UF8$ test function, NSGA-II gives the better value of Δ_p and Epsilon as compared with other algorithms. The Pareto optimal solutions of MOSHO on UF benchmark problems are illustrated in Fig. 3.27.

Table 3.42: Results of multi-objective algorithms on the IEEE CEC-2009 ($UF1 - UF10$) test functions.

F	Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
UF1	<i>Hypervolume</i>	4.69E-00	4.80E-01	5.10E-01	6.61E-01	4.83E-01	2.75E-01	1.46E-01
		(7.30E-04)	(2.04E-02)	(4.36E-02)	(2.80E-03)	(5.97E-02)	(1.20E-01)	(7.81E-02)
	Δ_p	4.07E-09	3.21E-04	4.44E-03	2.67E-04	6.42E-03	1.60E-02	3.67E-03
		(8.11E-11)	(1.11E-04)	(1.35E-03)	(1.04E-04)	(2.73E-03)	(1.69E-03)	(2.91E-03)
	<i>Spread</i>	1.27E-01	9.99E-01	1.50E+00	4.14E-01	1.41E+00	1.05E+00	1.01E+00
		(3.12E-02)	(2.13E-01)	(1.24E-01)	(1.85E-01)	(1.65E-01)	(1.05E-01)	(2.03E-01)
UF2	<i>Epsilon</i>	5.90E-03	2.06E-01	2.18E-01	1.19E-02	2.76E-01	6.58E-01	2.83E-01
		(1.11E-03)	(3.10E-02)	(5.50E-02)	(1.17E-02)	(1.04E-01)	(1.69E-01)	(1.20E-01)
	<i>Hypervolume</i>	1.07E-00	6.12E-01	6.10E-01	6.51E-01	5.94E-01	5.27E-01	4.16E-01
		(1.93E-03)	(2.95E-03)	(7.45E-03)	(3.89E-03)	(1.65E-02)	(1.07E-01)	(2.28E-01)
	Δ_p	1.13E-04	1.51E-03	6.37E-03	3.45E-04	5.63E-03	3.21E-03	3.10E-03
		(2.10E-04)	(3.86E-05)	(5.63E-04)	(2.75E-04)	(5.86E-04)	(2.11E-03)	(2.98E-03)
UF3	<i>Spread</i>	4.09E-02	4.98E-01	6.04E-01	5.43E-01	8.86E-01	1.04E+00	2.88E-01
		(2.93E-02)	(1.23E-01)	(4.70E-02)	(6.76E-02)	(8.12E-02)	(1.53E-01)	(1.49E-01)
	<i>Epsilon</i>	2.05E-02	9.23E-02	1.41E-01	6.85E-02	1.79E-01	3.35E-01	4.71E-02
		(5.19E-04)	(9.73E-03)	(3.68E-02)	(1.51E-02)	(5.61E-02)	(1.50E-01)	(3.43E-03)
	<i>Hypervolume</i>	5.09E-01	3.96E-01	3.24E-01	6.14E-01	3.55E-01	3.00E-00	4.74E-01
		(3.70E-02)	(3.51E-02)	(4.40E-02)	(2.24E-02)	(3.36E-02)	(3.75E-02)	(1.71E-02)
UF4	Δ_p	2.63E-03	5.88E-03	8.96E-03	1.97E-03	1.12E-02	1.69E-02	2.88E-02
		(1.61E-03)	(4.31E-04)	(1.60E-03)	(5.01E-04)	(1.89E-03)	(1.99E-03)	(1.40E-03)
	<i>Spread</i>	8.02E-01	6.86E-01	1.02E+00	7.03E-01	1.10E+00	1.09E+00	3.48E+00
		(6.99E-02)	(1.05E-01)	(5.47E-02)	(2.04E-01)	(1.03E-01)	(9.40E-02)	(2.77E-02)
	<i>Epsilon</i>	2.50E-01	1.95E-01	3.84E-01	1.48E-01	4.78E-01	5.55E-01	3.96E-01
		(4.11E-02)	(2.10E-02)	(5.94E-02)	(6.18E-02)	(6.32E-02)	(8.06E-02)	(4.48E-02)
UF5	<i>Hypervolume</i>	1.00E-00	2.40E-01	2.46E-01	2.37E-01	2.40E-01	1.52E-01	2.41E-01
		(2.32E-03)	(7.71E-03)	(3.19E-03)	(1.01E-02)	(3.13E-03)	(6.50E-02)	(2.97E-02)
	Δ_p	4.15E-04	2.66E-03	1.31E-03	3.89E-03	2.39E-03	5.60E-03	4.99E-03
		(2.69E-05)	(1.04E-04)	(4.45E-05)	(5.69E-04)	(1.32E-04)	(3.64E-03)	(2.81E-04)
	<i>Spread</i>	1.00E-01	4.01E-01	5.15E-01	5.02E-01	8.24E-01	1.09E+00	2.74E-01
		(3.13E-02)	(6.83E-02)	(5.11E-02)	(9.46E-02)	(5.32E-02)	(1.20E-01)	(4.24E-02)
UF6	<i>Epsilon</i>	1.10E-02	7.48E-02	6.99E-02	7.80E-02	8.39E-02	3.63E-01	4.71E-02
		(3.33E-03)	(6.03E-03)	(9.11E-03)	(9.14E-03)	(8.61E-03)	(3.10E-01)	(6.85E-03)
	<i>Hypervolume</i>	2.62E-01	3.51E-03	2.70E-02	4.72E-02	6.20E-02	3.96E-02	4.82E-02
		(1.23E-02)	(1.57E-02)	(4.06E-02)	(5.74E-02)	(6.01E-02)	(4.77E-02)	(2.09E-02)
	Δ_p	1.05E-01	2.63E-01	2.89E-01	1.07E-01	3.10E-01	1.35E-01	2.42E-01
		(1.08E-02)	(1.50E-01)	(3.89E-02)	(2.91E-02)	(1.38E-02)	(2.21E-02)	(3.85E-02)
UF7	<i>Spread</i>	1.90E-01	8.71E-01	1.26E+00	1.20E+00	1.14E+00	1.13E+00	1.21E+00
		(4.78E-02)	(9.50E-02)	(9.95E-02)	(5.02E-02)	(1.34E-01)	(8.40E-02)	(5.72E-02)
	<i>Epsilon</i>	1.11E-01	1.28E+00	6.93E-01	6.41E-01	7.82E-01	9.71E-01	4.90E-01
		(1.00E-01)	(5.28E-01)	(1.89E-01)	(1.43E-01)	(1.76E-01)	(2.21E-01)	(3.47E-01)
	<i>Hypervolume</i>	2.62E-01	3.51E-03	2.70E-02	4.72E-02	6.20E-02	3.96E-02	4.82E-02
		(1.23E-02)	(1.57E-02)	(4.06E-02)	(5.74E-02)	(6.01E-02)	(4.77E-02)	(2.09E-02)

(Continued...)

F	Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
UF6	<i>Hypervolume</i>	1.60E-00	2.02E-02	1.96E-01	2.08E-01	2.09E-01	1.15E-01	2.87E-01
		(1.38E-02)	(2.56E-02)	(6.99E-02)	(8.26E-02)	(6.55E-02)	(8.61E-02)	(3.75E-02)
	Δ_p	4.31E-03	4.56E-03	6.78E-03	4.44E-03	1.94E-02	2.21E-02	1.71E-02
		(1.10E-03)	(1.56E-03)	(2.11E-03)	(1.67E-03)	(3.79E-03)	(2.45E-03)	(8.29E-03)
	<i>Spread</i>	7.09E-01	9.79E-01	1.19E+00	1.14E+00	1.27E+00	1.10E+00	2.11E+00
		(4.40E-02)	(6.74E-02)	(1.24E-01)	(9.47E-02)	(2.03E-01)	(8.48E-02)	(4.66E-02)
UF7	<i>Epsilon</i>	2.02E-01	5.95E-01	4.22E-01	3.55E-01	5.48E-01	8.70E-01	5.39E-01
		(1.20E-01)	(1.82E-01)	(1.49E-01)	(1.48E-01)	(1.91E-01)	(2.67E-01)	(2.09E-01)
	<i>Hypervolume</i>	2.33E-01	3.95E-01	3.29E-01	4.91E-01	2.49E-01	1.50E-00	2.86E-01
		(6.36E-02)	(3.57E-02)	(8.89E-02)	(3.34E-03)	(8.82E-02)	(8.80E-02)	(3.99E-02)
	Δ_p	3.61E-03	6.22E-03	7.78E-03	1.51E-04	6.32E-02	3.67E-02	4.82E-02
		(1.45E-03)	(2.67E-03)	(2.06E-03)	(2.44E-05)	(2.71E-03)	(3.30E-03)	(1.75E-03)
UF8	<i>Spread</i>	2.81E-01	8.52E-01	1.07E+00	3.49E-01	1.18E+00	1.01E+00	4.88E-01
		(2.17E-02)	(6.63E-02)	(2.11E-01)	(1.75E-01)	(8.77E-02)	(2.58E-02)	(2.67E-02)
	<i>Epsilon</i>	2.05E-02	1.82E-01	4.28E-01	4.40E-02	6.08E-01	8.01E-01	4.73E-01
		(1.78E-02)	(1.07E-01)	(2.42E-01)	(3.17E-02)	(2.19E-01)	(1.37E-01)	(2.81E-01)
	<i>Hypervolume</i>	1.60E-00	3.05E-02	1.16E-01	1.81E-01	1.93E-02	1.29E-01	1.00E-01
		(2.47E-02)	(2.69E-02)	(4.07E-02)	(1.01E-01)	(3.32E-02)	(8.96E-02)	(2.79E-02)
UF9	Δ_p	3.34E-03	2.31E-03	2.11E-03	3.16E-01	5.11E-03	4.87E-03	5.95E-03
		(6.11E-04)	(2.01E-04)	(1.45E-04)	(5.17E-03)	(4.86E-04)	(9.09E-04)	(2.86E-03)
	<i>Spread</i>	6.07E-01	8.27E-01	7.77E-01	4.69E-01	7.07E-01	8.62E-01	5.79E-01
		(5.13E-01)	(8.20E-02)	(8.05E-02)	(7.90E-01)	(1.11E-01)	(2.29E-01)	(1.58E-01)
	<i>Epsilon</i>	8.03E-01	7.69E-01	6.24E-01	7.01E-01	9.70E-01	8.05E-01	6.97E-01
		(3.14E-01)	(8.51E-02)	(1.11E-01)	(9.54E-01)	(5.22E-02)	(1.31E-01)	(3.66E-01)
UF10	<i>Hypervolume</i>	4.09E-02	6.89E-02	2.54E-01	2.76E-00	9.43E-02	4.57E-01	3.39E-01
		(6.33E-01)	(5.70E-02)	(6.17E-02)	(4.47E-01)	(3.51E-02)	(4.78E-02)	(2.11E-02)
	Δ_p	4.39E-03	2.68E-03	5.35E-03	6.90E-03	4.78E-03	1.20E-03	3.23E-03
		(6.10E-04)	(3.27E-04)	(2.51E-04)	(4.48E-03)	(3.31E-04)	(1.04E-04)	(1.76E-03)
	<i>Spread</i>	1.01E-01	8.21E-01	8.52E-01	6.63E-01	7.55E-01	7.36E-01	4.59E-01
		(7.37E-02)	(5.05E-02)	(6.83E-02)	(6.41E-02)	(7.45E-02)	(6.10E-02)	(3.46E-02)
UF10	<i>Epsilon</i>	3.03E-01	8.21E-01	5.14E-01	6.56E-01	8.95E-01	4.81E-01	6.85E-01
		(5.15E-02)	(1.42E-01)	(6.46E-02)	(5.77E-02)	(4.16E-02)	(2.92E-02)	(5.68E-02)
	<i>Hypervolume</i>	4.60E-02	0.00E+00	4.53E-00	6.90E-02	4.05E-02	3.38E-02	4.90E-02
		(1.31E-02)	(0.00E+00)	(1.16E-02)	(5.11E-01)	(2.97E-02)	(2.86E-02)	(3.00E-02)
	Δ_p	6.78E-03	3.79E-02	3.13E-03	4.01E-03	1.83E-03	6.50E-03	4.22E-03
		(4.91E-03)	(6.73E-03)	(3.00E-03)	(2.81E-03)	(4.18E-04)	(3.47E-04)	(1.46E-03)
UF10	<i>Spread</i>	2.62E-01	6.62E-01	7.66E-01	5.21E-01	9.18E-01	1.24E+00	4.53E-01
		(5.82E-02)	(5.07E-02)	(6.28E-02)	(8.60E-01)	(9.13E-02)	(2.23E-01)	(1.86E-01)
	<i>Epsilon</i>	4.61E-01	1.82E+00	1.05E+00	5.33E-01	8.86E-01	8.68E-01	6.00E-01
		(5.53E-01)	(2.45E-01)	(1.55E-01)	(7.17E-01)	(1.07E-01)	(1.03E-01)	(2.85E-01)

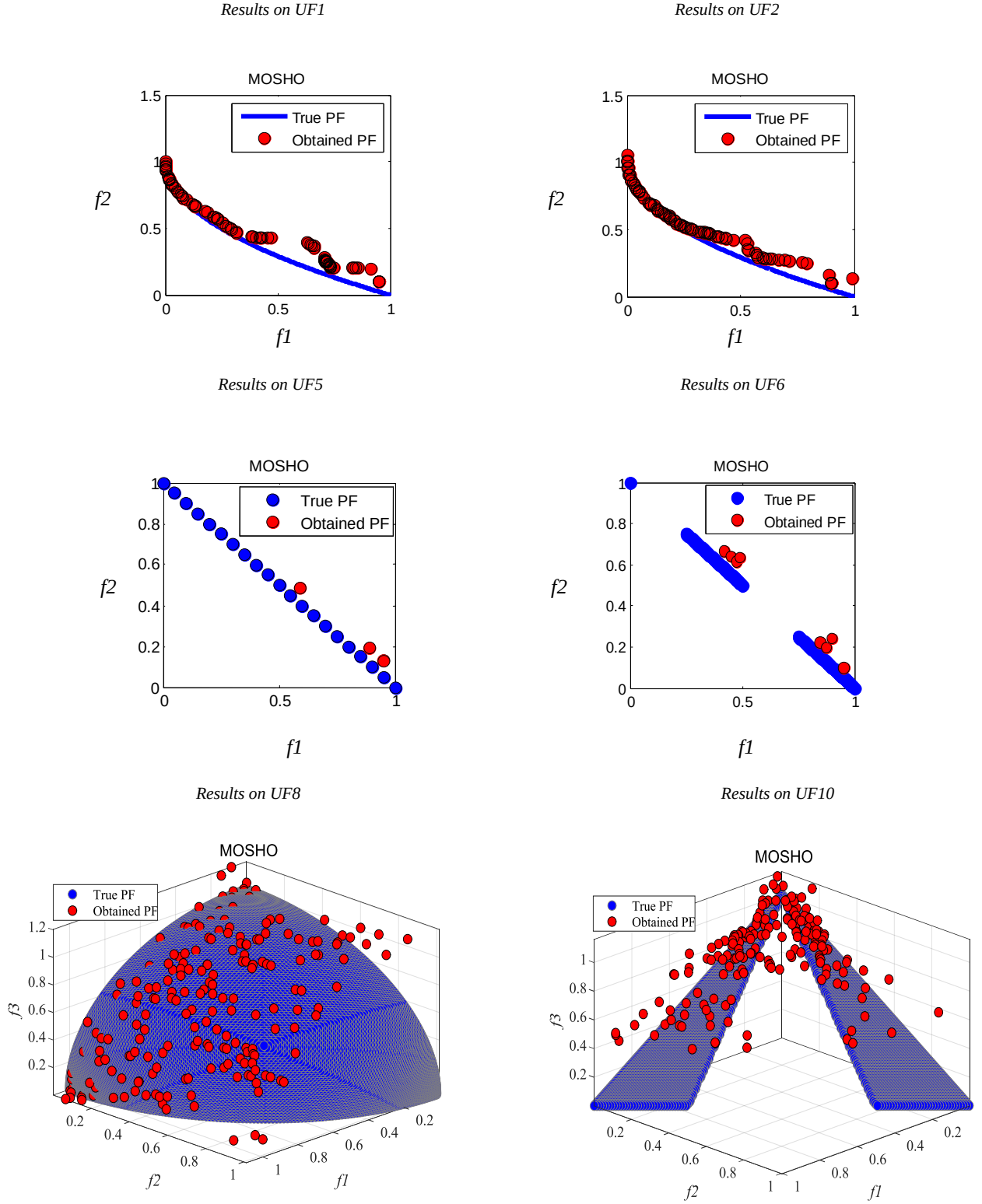


Fig. 3.27. Best Pareto optimal solutions obtained by the proposed MOSHO algorithm on IEEE CEC-2009 benchmark test problems.

- Results on ZDT ($ZDT1 - ZDT6$) benchmark test functions

The performance comparison of MOSHO with various algorithms on ZDT benchmark test problems are tabulated in Table 3.43. It has been observed from the table that MOSHO outperforms the other algorithms on the majority of ZDT test functions in terms of Δ_p , Spread, and Epsilon. The best Pareto optimal solutions obtained from the above-mentioned algorithms is shown in Fig. 3.28. This figure illustrates that MOSHO provides better true Pareto optimal front. It is also observed that NSGA-II, MOEA/D, and SPEA2 perform poorer results on $ZDT2$ test problem. MOSHO is able to provide efficient results on $ZDT2$, $ZDT3$, and $ZDT6$ test problems while all of the other algorithms fail to achieve Pareto optimal solutions. Fig. 3.29 shows the box plot of MOSHO is considerably narrower than those of MOPSO, NSGA-II, MOEA/D, PESA-II, SPEA2, and MOACO.

Table 3.43: Results of multi-objective algorithms on the ZDT benchmark (*ZDT1* – *ZDT6*) test functions.

F	Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
ZDT1	<i>Hypervolume</i>	6.56E-00 (3.82E-04)	6.62E-01 (1.40E-04)	6.60E-01 (2.62E-04)	6.64E-01 (3.56E-04)	6.56E-01 (8.58E-04)	6.60E-01 (3.05E-04)	6.60E-01 (2.46E-04)
	Δ_p	3.34E-04 (4.04E-05)	2.33E-04 (5.00E-07)	1.18E-04 (3.37E-06)	6.32E-05 (2.44E-06)	4.03E-04 (1.44E-04)	2.11E-04 (3.21E-06)	4.66E-04 (2.00E-06)
	<i>Spread</i>	2.30E-02 (1.11E-03)	6.87E-02 (1.04E-02)	3.77E-01 (2.58E-02)	2.85E-01 (2.82E-03)	6.38E-01 (3.78E-02)	1.43E-01 (1.60E-02)	2.31E-01 (2.20E-02)
	<i>Epsilon</i>	5.08E-02 (6.16E-02)	5.52E-03 (1.58E-04)	1.34E-02 (2.00E-03)	3.01E-03 (3.82E-04)	2.58E-02 (1.18E-02)	8.90E-03 (6.68E-04)	7.08E-02 (3.41E-04)
	<i>Hypervolume</i>	3.29E-01 (3.71E-05)	3.28E-01 (8.09E-05)	3.26E-01 (3.11E-04)	3.31E-01 (1.84E-04)	3.24E-00 (6.54E-04)	3.26E-01 (6.14E-04)	3.27E-01 (2.09E-04)
	Δ_p	2.01E-05 (1.34E-06)	1.45E-04 (2.57E-06)	1.37E-04 (3.41E-05)	4.98E-05 (6.00E-06)	3.52E-04 (2.35E-05)	2.22E-04 (4.21E-06)	1.30E-04 (2.76E-06)
ZDT2	<i>Spread</i>	2.02E-01 (9.10E-02)	6.68E-02 (7.85E-03)	3.70E-01 (4.46E-02)	1.45E-01 (1.21E-02)	6.76E-01 (4.20E-02)	1.49E-01 (1.46E-02)	2.56E-01 (3.87E-02)
	<i>Epsilon</i>	1.20E-03 (2.19E-04)	5.46E-03 (1.37E-04)	1.27E-02 (2.30E-03)	2.68E-03 (3.62E-04)	3.29E-02 (1.88E-02)	8.49E-03 (6.16E-04)	3.07E-03 (8.73E-04)
	<i>Hypervolume</i>	5.11E-01 (2.22E-04)	5.15E-01 (3.87E-04)	5.15E-01 (1.58E-04)	5.16E-01 (3.09E-05)	5.10E-00 (2.80E-03)	5.14E-01 (2.31E-04)	5.13E-01 (5.80E-04)
	Δ_p	4.13E-05 (3.31E-06)	2.57E-04 (2.08E-06)	1.33E-04 (4.49E-06)	4.14E-05 (5.55E-06)	1.00E-03 (5.29E-04)	3.21E-04 (2.88E-05)	2.40E-04 (4.99E-05)
	<i>Spread</i>	1.00E-01 (2.17E-03)	7.05E-01 (3.38E-03)	7.42E-01 (1.32E-02)	1.03E+00 (2.47E-03)	9.02E-01 (2.62E-02)	7.09E-01 (4.74E-03)	6.06E-01 (2.53E-03)
	<i>Epsilon</i>	4.91E-03 (6.74E-04)	5.71E-03 (5.49E-04)	8.87E-03 (1.56E-03)	5.33E-03 (1.27E-04)	1.30E-01 (1.54E-01)	9.60E-03 (1.14E-03)	5.94E-03 (2.48E-03)
ZDT3	<i>Hypervolume</i>	6.60E-01 (4.44E-04)	1.00E+00 (0.00E+00)	6.54E-01 (3.53E-03)	6.65E-01 (4.95E-05)	6.50E-01 (4.64E-03)	6.49E-01 (8.60E-03)	6.45E-01 (3.46E-02)
	Δ_p	4.30E-04 (1.80E-04)	3.41E-01 (2.38E-02)	3.78E-04 (5.99E-05)	4.18E-05 (3.04E-05)	3.10E-04 (1.10E-04)	4.71E-03 (1.20E-03)	1.69E-04 (1.00E-04)
	<i>Spread</i>	2.00E-01 (1.98E-03)	8.86E-01 (7.18E-02)	3.98E-01 (3.61E-02)	2.85E-01 (2.00E-03)	9.10E-01 (2.88E-01)	3.06E-01 (1.36E-01)	2.01E-01 (1.07E-02)
	<i>Epsilon</i>	2.09E-03 (1.00E-04)	5.14E+00 (1.91E+00)	1.76E-02 (8.12E-03)	2.94E-03 (1.03E-04)	2.87E-02 (1.06E-02)	6.37E-02 (5.29E-02)	3.28E-02 (2.02E-02)
	<i>Hypervolume</i>	3.19E-00 (4.33E-05)	4.01E-01 (5.40E-05)	3.88E-01 (1.46E-03)	4.05E-01 (6.11E-08)	3.91E-01 (1.34E-03)	3.78E-01 (2.70E-03)	2.26E-01 (1.86E-03)
	Δ_p	2.70E-04 (4.11E-05)	2.31E-04 (2.09E-06)	4.81E-04 (7.07E-05)	3.61E-05 (6.41E-08)	3.07E-04 (3.80E-05)	4.95E-04 (3.37E-05)	3.27E-04 (1.30E-05)
ZDT4	<i>Spread</i>	1.11E-01 (1.24E-04)	8.48E-01 (4.79E-01)	3.66E-01 (3.99E-02)	1.50E-01 (1.86E-04)	7.59E-01 (1.93E-01)	2.38E-01 (3.08E-02)	3.47E-01 (1.94E-02)
	<i>Epsilon</i>	1.20E-03 (3.13E-04)	4.86E-03 (5.16E-04)	1.58E-02 (2.06E-03)	1.64E-03 (1.43E-01)	1.89E-02 (3.66E-03)	2.56E-02 (5.35E-06)	3.00E-02 (2.17E-04)
	<i>Hypervolume</i>	3.19E-00 (4.33E-05)	4.01E-01 (5.40E-05)	3.88E-01 (1.46E-03)	4.05E-01 (6.11E-08)	3.91E-01 (1.34E-03)	3.78E-01 (2.70E-03)	2.26E-01 (1.86E-03)
	Δ_p	2.70E-04 (4.11E-05)	2.31E-04 (2.09E-06)	4.81E-04 (7.07E-05)	3.61E-05 (6.41E-08)	3.07E-04 (3.80E-05)	4.95E-04 (3.37E-05)	3.27E-04 (1.30E-05)
	<i>Spread</i>	1.11E-01 (1.24E-04)	8.48E-01 (4.79E-01)	3.66E-01 (3.99E-02)	1.50E-01 (1.86E-04)	7.59E-01 (1.93E-01)	2.38E-01 (3.08E-02)	3.47E-01 (1.94E-02)
	<i>Epsilon</i>	1.20E-03 (3.13E-04)	4.86E-03 (5.16E-04)	1.58E-02 (2.06E-03)	1.64E-03 (1.43E-01)	1.89E-02 (3.66E-03)	2.56E-02 (5.35E-06)	3.00E-02 (2.17E-04)
ZDT6	<i>Hypervolume</i>	3.19E-00 (4.33E-05)	4.01E-01 (5.40E-05)	3.88E-01 (1.46E-03)	4.05E-01 (6.11E-08)	3.91E-01 (1.34E-03)	3.78E-01 (2.70E-03)	2.26E-01 (1.86E-03)
	Δ_p	2.70E-04 (4.11E-05)	2.31E-04 (2.09E-06)	4.81E-04 (7.07E-05)	3.61E-05 (6.41E-08)	3.07E-04 (3.80E-05)	4.95E-04 (3.37E-05)	3.27E-04 (1.30E-05)
	<i>Spread</i>	1.11E-01 (1.24E-04)	8.48E-01 (4.79E-01)	3.66E-01 (3.99E-02)	1.50E-01 (1.86E-04)	7.59E-01 (1.93E-01)	2.38E-01 (3.08E-02)	3.47E-01 (1.94E-02)
	<i>Epsilon</i>	1.20E-03 (3.13E-04)	4.86E-03 (5.16E-04)	1.58E-02 (2.06E-03)	1.64E-03 (1.43E-01)	1.89E-02 (3.66E-03)	2.56E-02 (5.35E-06)	3.00E-02 (2.17E-04)
	<i>Hypervolume</i>	3.19E-00 (4.33E-05)	4.01E-01 (5.40E-05)	3.88E-01 (1.46E-03)	4.05E-01 (6.11E-08)	3.91E-01 (1.34E-03)	3.78E-01 (2.70E-03)	2.26E-01 (1.86E-03)
	Δ_p	2.70E-04 (4.11E-05)	2.31E-04 (2.09E-06)	4.81E-04 (7.07E-05)	3.61E-05 (6.41E-08)	3.07E-04 (3.80E-05)	4.95E-04 (3.37E-05)	3.27E-04 (1.30E-05)

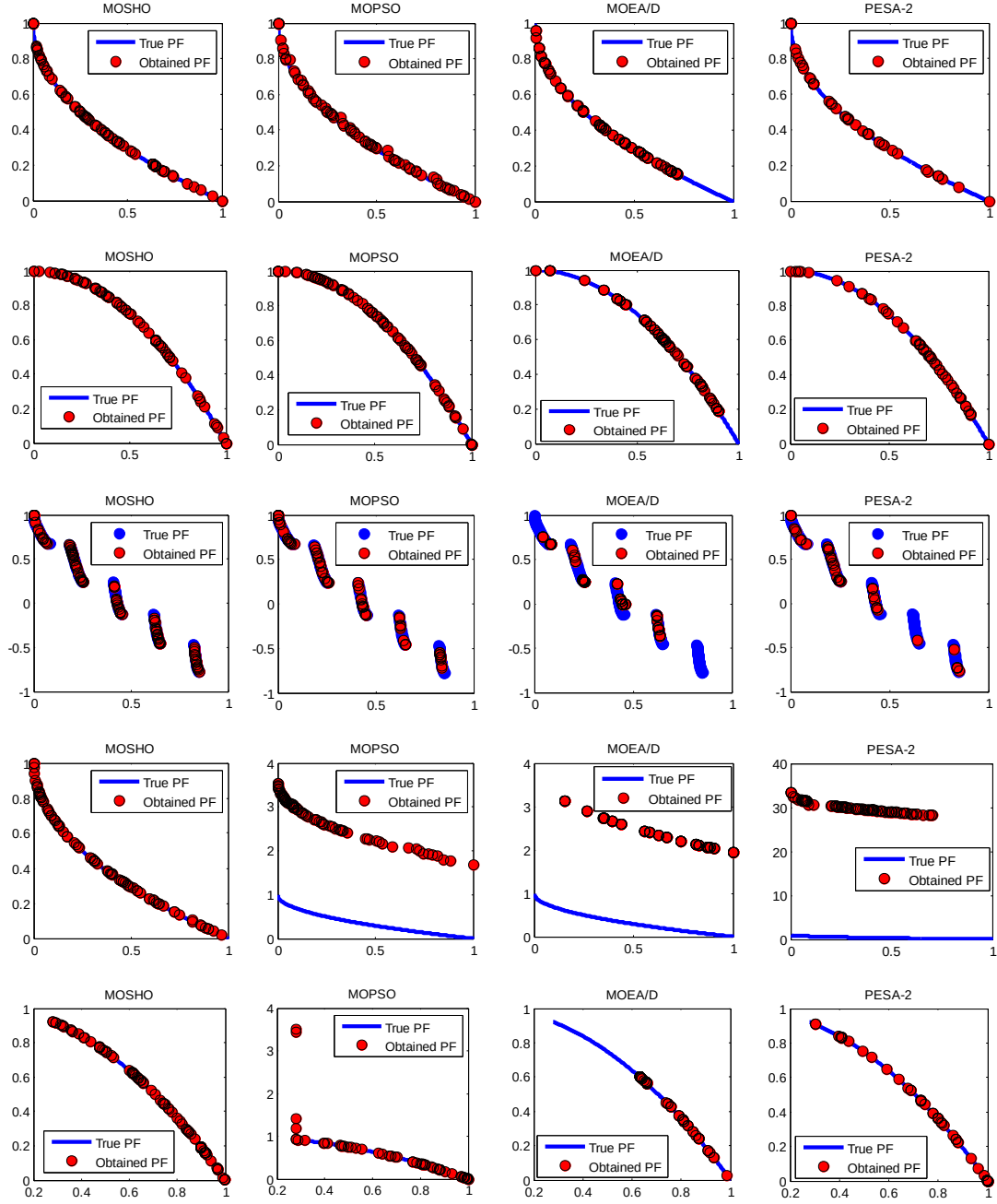


Fig. 3.28. Best Pareto optimal solutions obtained by multi-objective algorithms on (a) ZDT1, (b) ZDT2, (c) ZDT3, (d) ZDT4, and (e) ZDT6 test problems.

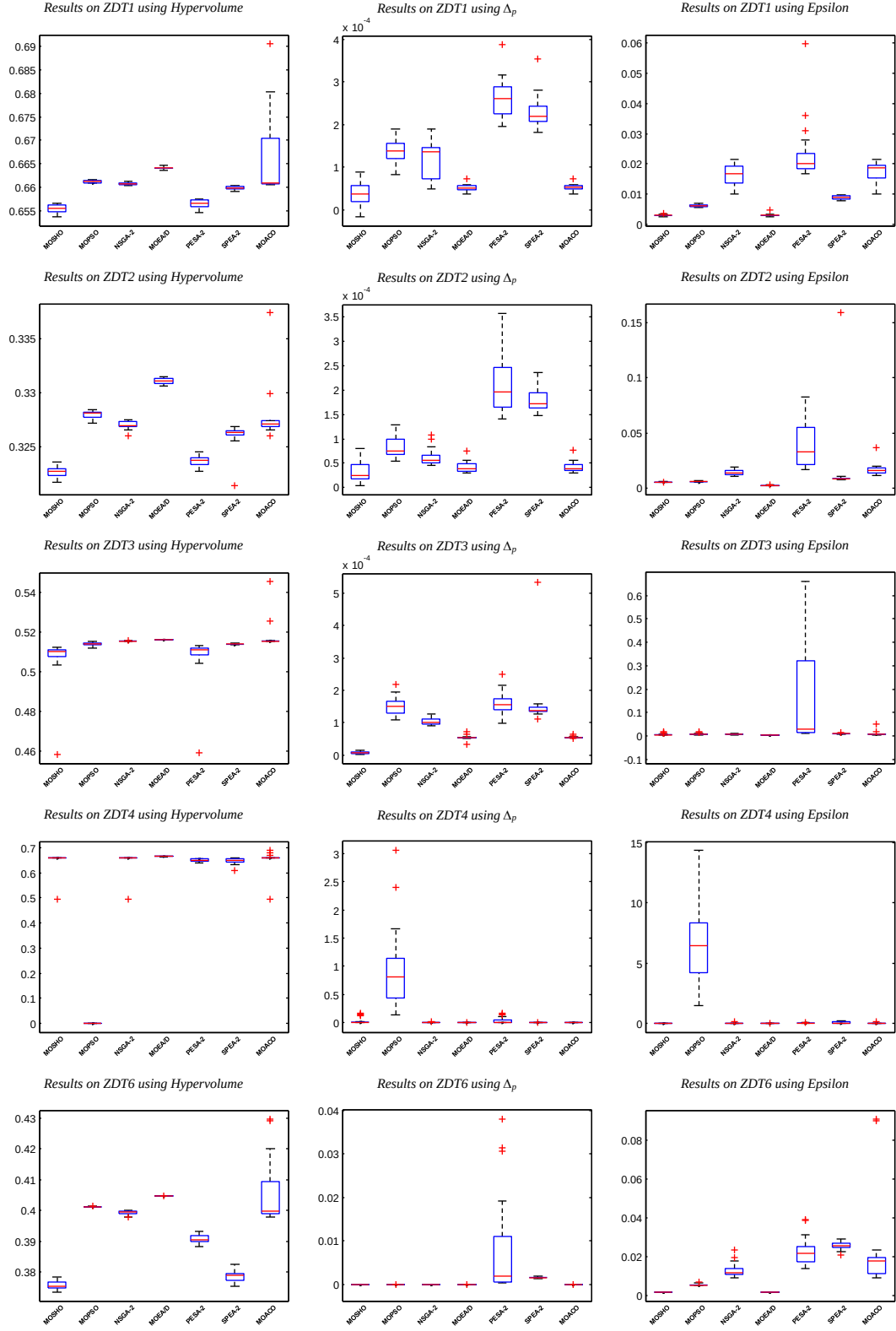


Fig. 3.29. Box plot of the statistical results for Hypervolume, Δ_p , Spread, and Epsilon on ZDT benchmark test problems.

- Results on DTLZ ($DTLZ1 - DTLZ9$) benchmark test functions

The statistical results of MOPSO, NSGA-II, MOEA/D, PESA-II, SPEA2, and MOACO on DTLZ benchmark problems are shown in Table 3.44. For $DTLZ1, DTLZ2, DTLZ3, DTLZ5, DTLZ6$, and $DTLZ9$ test functions, MOSHO provides better statistical results than the others for most of the performance measures. For $DTLZ4$ test function, MOEA/D and NSGA-II give the better value of Δ_p and Epsilon over other algorithms. PESA-II provides the better value of Hypervolume and Spread than the others.

The results show that the proposed algorithm has better convergence ability for most of the $DTLZ$ benchmark test functions. Fig. 3.30 shows the shapes of the obtained Pareto optimal fronts of MOSHO which are closer to the true Pareto optimal front. However, these results prove that the durability of the proposed MOSHO algorithm is considerably high on these unconstrained benchmark test problems. Therefore, MOSHO shows superior convergence on all of the above benchmark test problems.

Table 3.44: Results of the multi-objective algorithms on the DTLZ benchmark ($DTLZ1 - DTLZ9$) test functions.

F	Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
DTLZ1	<i>Hypervolume</i>	4.60E-01 (4.33E-02)	3.63E-01 (5.56E-02)	6.62E-01 (3.95E-01)	3.56E-00 (5.94E-03)	5.93E-01 (3.73E-02)	7.56E-01 (1.97E-02)	4.11E-01 (2.00E-02)
	Δ_p	3.45E-04 (1.11E-05)	2.53E-03 (2.49E-04)	3.78E-03 (3.81E-04)	4.24E-04 (5.40E-05)	5.00E-03 (2.00E-04)	4.46E-04 (7.21E-06)	5.79E-04 (4.11E-04)
	<i>Spread</i>	1.33E-01 (3.17E-01)	5.95E-01 (2.93E-01)	2.02E+00 (2.66E-01)	2.85E-01 (4.24E-01)	3.72E+00 (3.89E-01)	9.15E-01 (2.97E-01)	4.97E-01 (1.08E-01)
	<i>Epsilon</i>	3.60E-03 (5.83E-02)	3.96E-02 (3.93E-02)	2.10E-02 (5.23E-01)	4.35E-02 (3.73E-02)	2.04E-03 (2.67E-01)	5.39E-02 (1.49E-02)	2.07E-02 (2.00E-02)
	<i>Hypervolume</i>	1.09E-00 (1.07E-03)	1.94E-01 (2.63E-03)	2.63E-02 (3.96E-03)	3.08E-01 (5.23E-02)	5.70E-02 (1.12E-02)	4.04E-01 (1.87E-03)	3.97E-01 (2.05E-03)
	Δ_p	3.54E-04 (1.11E-05)	3.48E-03 (8.06E-04)	3.67E-04 (2.00E-05)	6.90E-04 (6.57E-05)	3.19E-02 (5.39E-03)	4.73E-04 (4.11E-05)	4.95E-04 (4.06E-05)
DTLZ2	<i>Spread</i>	3.30E-01 (5.78E-03)	6.71E-01 (2.73E-01)	4.83E-01 (7.54E-03)	7.40E-01 (3.30E-02)	7.56E-01 (5.73E-01)	5.46E-01 (3.05E-02)	4.83E-01 (6.28E-02)
	<i>Epsilon</i>	3.69E-02 (6.16E-02)	7.73E-02 (8.33E-03)	2.37E-01 (8.08E-02)	5.05E-02 (3.57E-01)	3.73E-01 (4.73E-01)	8.41E-02 (1.13E-02)	6.05E-02 (4.74E-02)
	<i>Hypervolume</i>	2.09E-02 (8.37E-03)	3.63E-01 (2.63E-02)	6.24E-02 (5.25E-04)	3.74E-01 (2.74E-02)	5.23E-01 (2.35E-03)	2.00E+00 (0.00E+00)	4.03E-01 (4.46E-03)
	Δ_p	3.47E-04 (2.10E-04)	5.22E-03 (3.14E-03)	5.05E-04 (3.33E-04)	2.49E-03 (6.41E-04)	3.03E-01 (3.90E-02)	1.00E-01 (4.59E-02)	3.04E-02 (2.41E-02)
	<i>Spread</i>	1.39E-02 (3.33E-02)	5.73E-01 (3.73E-01)	1.32E+00 (6.34E-02)	4.63E-01 (2.63E-02)	2.43E-02 (4.03E-01)	1.20E+00 (1.09E-01)	2.92E-01 (1.63E-01)
	<i>Epsilon</i>	2.09E-02 (1.56E-02)	3.74E-01 (4.38E-02)	5.73E-01 (5.77E-02)	3.84E-02 (6.63E-02)	3.22E-01 (8.74E-02)	6.37E+00 (2.70E+00)	3.20E-01 (2.83E-02)
DTLZ3	<i>Hypervolume</i>	2.60E-02 (1.31E-02)	2.74E-01 (3.73E-03)	3.45E-02 (6.89E-02)	2.74E-01 (3.63E-02)	4.53E-00 (5.23E-03)	3.12E-01 (9.77E-02)	4.96E-02 (2.38E-02)
	Δ_p	2.33E-04 (4.11E-05)	4.61E-02 (2.00E-04)	3.69E-02 (5.31E-03)	2.14E-04 (1.09E-04)	4.79E-02 (6.49E-03)	4.35E-03 (3.20E-03)	3.16E-02 (4.92E-03)
	<i>Spread</i>	2.78E-02 (5.11E-02)	3.08E-01 (3.53E-02)	5.73E-02 (6.63E-01)	6.64E-01 (3.21E-02)	4.23E-03 (6.66E-02)	4.68E-01 (1.29E-01)	5.05E-02 (4.80E-02)
	<i>Epsilon</i>	5.05E-02 (8.70E-02)	5.13E-02 (6.63E-02)	6.73E-03 (9.14E-03)	3.63E-02 (9.74E-02)	3.53E-01 (8.33E-03)	3.23E-01 (2.86E-01)	4.60E-02 (3.94E-02)
	<i>Hypervolume</i>	1.49E-00 (1.32E-04)	1.63E-03 (2.83E-04)	3.13E-02 (5.06E-02)	6.05E-01 (3.83E-02)	5.93E-02 (5.23E-02)	1.96E-01 (2.47E-02)	2.42E-02 (1.99E-02)
	Δ_p	1.46E-03 (2.81E-04)	1.35E-01 (7.36E-02)	3.21E-01 (1.73E-02)	5.64E-03 (6.88E-04)	4.73E-02 (2.32E-02)	3.77E-01 (2.31E-02)	2.90E-02 (1.04E-03)
DTLZ4	<i>Spread</i>	3.88E-01 (5.00E-02)	5.63E-01 (5.73E-02)	1.36E+00 (7.96E-03)	6.27E-01 (5.22E-02)	1.04E+00 (4.14E-04)	2.43E-02 (5.47E-02)	8.33E-01 (4.29E-02)
	<i>Epsilon</i>	2.03E-01 (2.78E-01)	3.73E+00 (5.28E-01)	5.23E-01 (1.69E-01)	6.94E-01 (5.92E-01)	5.82E-01 (1.02E-01)	7.75E-01 (3.20E-01)	4.06E-01 (1.95E-01)
	<i>Hypervolume</i>	1.49E-00 (1.32E-04)	1.63E-03 (2.83E-04)	3.13E-02 (5.06E-02)	6.05E-01 (3.83E-02)	5.93E-02 (5.23E-02)	1.96E-01 (2.47E-02)	2.42E-02 (1.99E-02)
	Δ_p	1.46E-03 (2.81E-04)	1.35E-01 (7.36E-02)	3.21E-01 (1.73E-02)	5.64E-03 (6.88E-04)	4.73E-02 (2.32E-02)	3.77E-01 (2.31E-02)	2.90E-02 (1.04E-03)
	<i>Spread</i>	3.88E-01 (5.00E-02)	5.63E-01 (5.73E-02)	1.36E+00 (7.96E-03)	6.27E-01 (5.22E-02)	1.04E+00 (4.14E-04)	2.43E-02 (5.47E-02)	8.33E-01 (4.29E-02)
	<i>Epsilon</i>	2.03E-01 (2.78E-01)	3.73E+00 (5.28E-01)	5.23E-01 (1.69E-01)	6.94E-01 (5.92E-01)	5.82E-01 (1.02E-01)	7.75E-01 (3.20E-01)	4.06E-01 (1.95E-01)
DTLZ5	<i>Hypervolume</i>	1.49E-00 (1.32E-04)	1.63E-03 (2.83E-04)	3.13E-02 (5.06E-02)	6.05E-01 (3.83E-02)	5.93E-02 (5.23E-02)	1.96E-01 (2.47E-02)	2.42E-02 (1.99E-02)
	Δ_p	1.46E-03 (2.81E-04)	1.35E-01 (7.36E-02)	3.21E-01 (1.73E-02)	5.64E-03 (6.88E-04)	4.73E-02 (2.32E-02)	3.77E-01 (2.31E-02)	2.90E-02 (1.04E-03)
	<i>Spread</i>	3.88E-01 (5.00E-02)	5.63E-01 (5.73E-02)	1.36E+00 (7.96E-03)	6.27E-01 (5.22E-02)	1.04E+00 (4.14E-04)	2.43E-02 (5.47E-02)	8.33E-01 (4.29E-02)
	<i>Epsilon</i>	2.03E-01 (2.78E-01)	3.73E+00 (5.28E-01)	5.23E-01 (1.69E-01)	6.94E-01 (5.92E-01)	5.82E-01 (1.02E-01)	7.75E-01 (3.20E-01)	4.06E-01 (1.95E-01)
	<i>Hypervolume</i>	1.49E-00 (1.32E-04)	1.63E-03 (2.83E-04)	3.13E-02 (5.06E-02)	6.05E-01 (3.83E-02)	5.93E-02 (5.23E-02)	1.96E-01 (2.47E-02)	2.42E-02 (1.99E-02)
	Δ_p	1.46E-03 (2.81E-04)	1.35E-01 (7.36E-02)	3.21E-01 (1.73E-02)	5.64E-03 (6.88E-04)	4.73E-02 (2.32E-02)	3.77E-01 (2.31E-02)	2.90E-02 (1.04E-03)

(Continued...)

F	Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
DTLZ6	Hypervolume	2.63E-00	3.02E-03	1.92E-01	5.04E-02	3.09E-01	1.05E-01	2.53E-02
		(1.11E-03)	<i>(2.56E-02)</i>	<i>(7.83E-02)</i>	<i>(3.16E-03)</i>	<i>(5.55E-02)</i>	<i>(6.61E-02)</i>	<i>(2.00E-02)</i>
	Δ_p	3.00E-03	2.33E-02	3.50E-02	5.21E-03	3.67E-02	3.11E-02	2.10E-02
		(2.17E-03)	<i>(3.66E-03)</i>	<i>(5.12E-03)</i>	<i>(2.51E-03)</i>	<i>(6.36E-03)</i>	<i>(3.10E-03)</i>	<i>(4.96E-03)</i>
	Spread	3.45E-01	6.09E-01	1.53E+00	4.10E-01	1.26E-01	2.10E+00	3.53E-01
		<i>(6.77E-02)</i>	(3.32E-02)	<i>(1.94E-01)</i>	<i>(5.07E-01)</i>	<i>(2.63E-01)</i>	<i>(7.18E-02)</i>	<i>(4.75E-02)</i>
DTLZ7	Hypervolume	2.47E-03	6.95E-01	4.63E-03	3.65E-02	3.43E-01	7.72E-01	4.34E-02
		<i>(7.16E-03)</i>	<i>(1.82E-03)</i>	<i>(3.63E-01)</i>	(3.48E-04)	<i>(6.36E-01)</i>	<i>(2.73E-01)</i>	<i>(6.80E-03)</i>
	Δ_p	2.09E-02	3.53E-01	6.29E-03	4.11E-01	3.49E-00	2.51E-01	5.03E-02
		<i>(1.32E-02)</i>	<i>(3.63E-02)</i>	(3.09E-04)	<i>(3.83E-03)</i>	<i>(5.82E-01)</i>	<i>(8.38E-02)</i>	<i>(7.72E-03)</i>
	Spread	4.10E-03	5.50E-03	6.98E-03	3.38E-03	4.91E-02	1.89E-02	6.42E-03
		<i>(3.77E-04)</i>	(1.13E-05)	<i>(3.48E-04)</i>	<i>(3.27E-03)</i>	<i>(6.94E-03)</i>	<i>(2.81E-03)</i>	<i>(3.92E-03)</i>
DTLZ8	Hypervolume	4.67E-02	6.31E-01	1.83E+00	4.35E-01	3.73E-03	1.03E-01	2.28E-02
		<i>(7.17E-02)</i>	<i>(8.04E-02)</i>	<i>(2.53E-03)</i>	<i>(3.37E-01)</i>	(3.27E-04)	<i>(3.91E-02)</i>	<i>(5.40E-03)</i>
	Δ_p	7.78E-02	8.63E-01	3.28E-02	3.63E-02	3.08E-01	4.04E-01	2.28E-01
		<i>(5.18E-02)</i>	<i>(1.07E-01)</i>	(3.74E-04)	<i>(3.35E-02)</i>	<i>(3.33E-02)</i>	<i>(1.93E-01)</i>	<i>(4.32E-02)</i>
	Spread	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
DTLZ9	Hypervolume	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Δ_p	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Spread	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
DTLZ9	Hypervolume	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Δ_p	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Spread	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
DTLZ9	Hypervolume	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Δ_p	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Spread	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
DTLZ9	Hypervolume	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Δ_p	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Spread	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
DTLZ9	Hypervolume	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Δ_p	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Spread	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
DTLZ9	Hypervolume	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Δ_p	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Spread	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
DTLZ9	Hypervolume	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Δ_p	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Spread	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
DTLZ9	Hypervolume	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Δ_p	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Spread	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
DTLZ9	Hypervolume	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Δ_p	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Spread	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
DTLZ9	Hypervolume	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Δ_p	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
	Spread	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
DTLZ9	Hypervolume	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Δ_p	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.76E-03)</i>
	Spread	6.46E-03	3.18E-04	5.03E-03	4.75E-02	4.69E-03	3.11E-03	4.53E-03
		<i>(2.81E-04)</i>	<i>(1.00E-03)</i>	<i>(2.68E-03)</i>	<i>(2.30E-03)</i>	(6.68E-05)	<i>(1.47E-03)</i>	<i>(8.95E-04)</i>
DTLZ9	Hypervolume	3.87E-02	3.27E-01	2.37E-03	3.39E-01	5.07E-02	5.02E-02	1.37E-02
		<i>(7.77E-02)</i>	<i>(3.20E-02)</i>	<i>(8.53E-03)</i>	<i>(3.90E-01)</i>	<i>(3.14E-02)</i>	<i>(2.99E-02)</i>	(4.05E-03)
	Δ_p	1.01E-02	3.69E-01	6.25E-03	3.01E-01	3.24E-02	6.09E-01	2.23E-02
		<i>(6.19E-02)</i>	<i>(3.51E-02)</i>	(1.24E-03)	<i>(4.23E-01)</i>	<i>(2.22E-02)</i>	<i>(1.82E-01)</i>	<i>(6.97E-03)</i>
	Spread	7.60E-03	4.50E-00	1.84E-01	4.88E-03	2.03E-02	2.98E-01	1.39E-01
		<i>(2.47E-02)</i>	(2.69E-04)	<i>(3.94E-02)</i>	<i>(2.01E-02)</i>	<i>(4.23E-02)</i>	<i>(6.93E-02)</i>	<i>(2.</i>

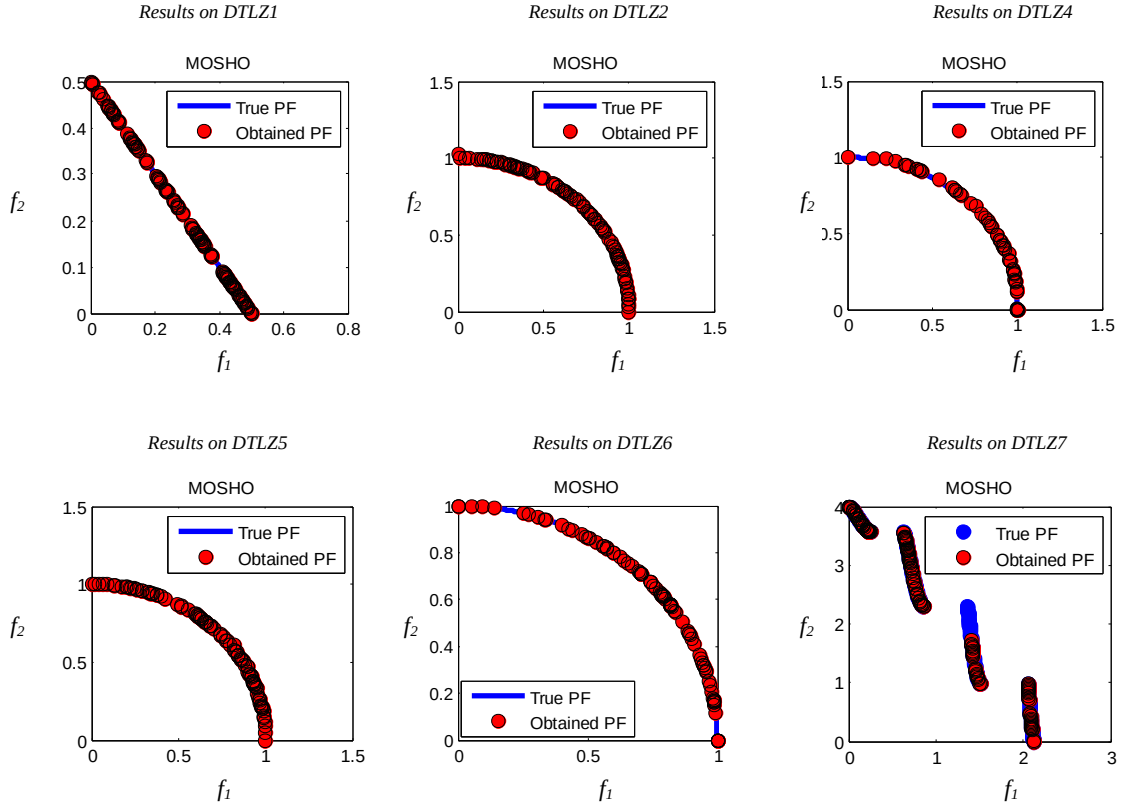


Fig. 3.30. Best Pareto optimal solutions obtained by the proposed MOSHO algorithm on DTLZ benchmark test problems.

3.3.1.3.5 Sensitivity Analysis

- Effect of an archive on proposed algorithm

The behavior of archive during consecutive generations is illustrated in Table 3.45. To show the effect of an archive on proposed MOSHO, it has been tested on $ZDT1$, $DTLZ6$, and $ZDT3$ benchmark test functions. $ZDT1$, $DTLZ6$, and $ZDT3$ have concave, convex, and disconnected properties, respectively. The archive size of these test problems is fixed at 10. The detailed convergence analysis is depicted in Fig. 3.31. It is observed that the proposed MOSHO achieves optimal values of these test functions during the course of iterations.

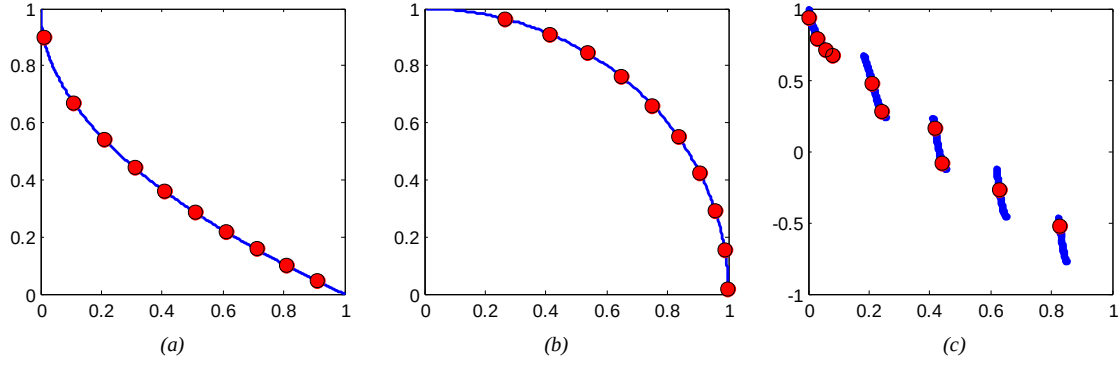


Fig. 3.31. Convergence analysis of proposed archive on (a) Concave, (b) Convex, and (c) Disconnected Pareto fronts.

- Effect of selection mechanism on proposed algorithm

The roulette wheel and tournament selection approaches are used to analyze the performance of proposed MOSHO. For experimentation, $ZDT1$, $ZDT3$, and $ZDT6$ test functions are taken into consideration. Fig. 3.32 shows the convergence analysis of roulette wheel and tournament selection approaches. From Fig. 3.32, it can be concluded that the convergence of roulette wheel selection approach is better than the tournament selection approach. It is also observed that the roulette wheel selection approach is able to converge towards the optimal solutions.

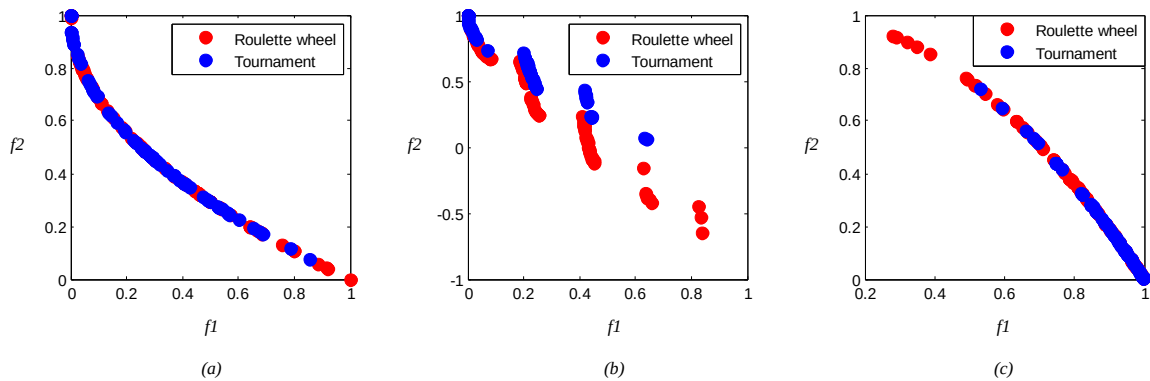


Fig. 3.32. Convergence analysis of selection approaches on (a) $ZDT1$, (b) $ZDT3$, and (c) $ZDT6$ test functions.

Table 3.45: Archive state in different generations for ZDT1, DTLZ6, and ZDT3 test functions using MOSHO.

Iterations	ZDT1 (<i>Concave</i>)				DTLZ6 (<i>Convex</i>)			
	Archive		Objective value		Archive		Objective value	
	x	y	f_1	f_2	x	y	f_1	f_2
<i>1</i>	0.999	0.015	0.861	0.937	0.256	3.616	0.648	0.382
	0.987	0.156			0.005	5.989		
	0.955	0.294			0.005	5.989		
	0.904	0.425			0.030	5.337		
	0.835	0.549			0.030	5.337		
	0.750	0.661			0.005	5.989		
	0.649	0.760			0.090	4.112		
	0.535	0.844			0.030	5.337		
	0.411	0.911			0.256	3.616		
	0.263	0.964			0.030	5.337		
<i>50</i>	0.842	0.010	0.762	0.753	0.136	2.025	0.590	0.233
	0.862	0.140			0.023	3.234		
	0.823	0.118			0.007	4.023		
	0.843	0.346			0.039	4.237		
	0.705	0.413			0.039	3.007		
	0.639	0.519			0.085	4.462		
	0.556	0.431			0.128	3.096		
	0.448	0.746			0.025	3.039		
	0.401	0.831			0.223	2.219		
	0.248	0.763			0.010	4.414		
<i>100</i>	0.018	0.900	0.801	0.566	0.999	0.157	0.568	0.967
	0.117	0.670			0.987	0.156		
	0.215	0.541			0.955	0.294		
	0.316	0.442			0.904	0.425		
	0.418	0.359			0.835	0.549		
	0.519	0.285			0.750	0.661		
	0.612	0.218			0.649	0.760		
	0.714	0.157			0.535	0.844		
	0.818	0.100			0.411	0.911		
	0.916	0.045			0.263	0.964		

Table 3.46: Comparison of best solution obtained from different algorithms for welded beam design problem.

Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
<i>Hypervolume</i>	5.18E-00	8.09E-01	7.17E-01	7.94E-01	9.93E-01	6.40E-01	8.59E-01
	(3.35E-01)	(6.93E-01)	(5.34E-01)	(6.97E-01)	(8.96E-01)	(4.87E-01)	(5.70E-01)
Δ_p	3.71E-02	2.47E-01	1.69E-01	4.45E-02	6.84E-01	7.92E-02	4.34E-02
	(5.07E-03)	(6.55E-02)	(2.87E-02)	(2.30E-02)	(6.87E-02)	(5.50E-02)	(3.48E-02)
<i>Spread</i>	1.37E-01	2.14E-01	9.90E-01	1.78E+00	1.81E-01	6.59E-01	3.66E-01
	(5.96E-02)	(8.29E-02)	(1.78E-01)	(1.01E+00)	(7.20E-02)	(2.18E-01)	(2.97E-01)
<i>Epsilon</i>	9.64E-03	1.04E-01	9.64E-02	4.22E-02	1.15E-01	1.52E-02	4.68E-02
	(2.41E-03)	(9.64E-02)	(1.36E-02)	(9.75E-03)	(9.63E-02)	(8.76E-03)	(3.05E-03)

3.3.1.4 Applications of MOSHO on Engineering Design Problems

In order to investigate the performance of proposed MOSHO, it has been applied on five real-life constrained engineering design problems such as welded beam design, multiple-disk clutch brake design, pressure vessel design, speed reducer design, and gear train design. Due to low computational cost and simplicity, MOSHO algorithm is equipped with death penalty function to handle multiple constraints. The proposed MOSHO is compared with six well-known algorithms such as MOPSO, NSGA-II, MOEA/D, PESA-II, SPEA2, and MOACO.

3.3.1.4.1 Welded Beam Design Problem

The main objective of this problem is to minimize the fabrication cost and simultaneously minimize the vertical deflection at the end of the welded beam as shown in Fig. 3.8. The mathematical formulation of this problem is given in Appendix C.2.1. Table 3.46 shows the comparisons of different algorithms for the best obtained solutions. The results reveal that MOSHO performs better than the other existing algorithms. By observing the results, MOSHO achieves the near optimal solution during course of iterations and durability of the proposed MOSHO algorithm is high on this constrained engineering problem.

3.3.1.4.2 Multiple-disk Clutch Brake Design Problem

The objective of this problem is to minimize the stopping time (f_1) as well as minimize the mass of the brake system (f_2) as shown in Fig. 3.33. There are five decision variables of this problem.

- Inner radius in mm (r_i).
- Outer radius in mm (r_o).
- Thickness of discs in mm (t).
- Actuating force (F).
- Number of friction surfaces (Z).

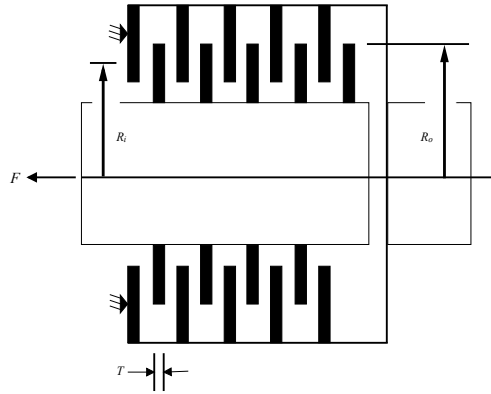


Fig. 3.33. Schematic view of multiple-disk clutch brake design problem.

The mathematical formulation of this problem is given in Appendix C.2.2. The obtained results of best optimal solution among several algorithms are given in Table 3.47. Among these algorithms, the proposed MOSHO gives the optimal value of decision variables. It is worth mentioning that MOSHO provides better statistical results in terms of Hypervolume, Δ_p , Spread, and Epsilon.

3.3.1.4.3 Pressure Vessel Design Problem

This problem was proposed by Kannan and Kramer^[144] to minimize the total cost (f_1) including cost of material, forming and welding of cylindrical vessel

Table 3.47: Comparison of best solution obtained from different algorithms for multiple-disk clutch brake design problem.

Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
<i>Hypervolume</i>	4.53E-00	7.86E-01	7.19E-01	9.64E-01	9.03E-01	9.64E-01	6.40E-01
	(2.49E-01)	(5.32E-01)	(5.63E-01)	(7.63E-01)	(8.14E-01)	(7.96E-01)	(4.02E-01)
Δ_p	2.73E-02	5.16E-02	2.13E-01	7.43E-02	2.21E-01	9.72E-02	5.13E-02
	(1.48E-03)	(5.99E-03)	(2.52E-02)	(6.03E-02)	(9.91E-02)	(3.17E-02)	(4.52E-02)
<i>Spread</i>	1.96E-01	2.36E-01	8.64E-01	1.54E+00	9.52E-01	8.30E-01	4.42E-01
	(1.10E-01)	(1.35E-01)	(5.24E-01)	(1.35E+00)	(5.21E-01)	(4.40E-01)	(2.38E-01)
<i>Epsilon</i>	1.76E-02	1.14E-01	7.52E-02	1.39E-01	1.10E-01	2.93E-02	2.08E-02
	(5.24E-03)	(9.84E-02)	(2.44E-02)	(8.32E-02)	(7.30E-02)	(1.31E-02)	(1.58E-02)

Table 3.48: Comparison of best solution obtained from different algorithms for pressure vessel design problem.

Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
<i>Hypervolume</i>	3.03E-00	7.30E-01	6.25E-01	8.95E-01	8.36E-01	9.60E-01	4.00E-01
	(2.00E-01)	(4.23E-01)	(4.51E-01)	(7.13E-01)	(7.01E-01)	(8.95E-01)	(3.79E-01)
Δ_p	2.67E-03	6.16E-01	4.31E-02	2.59E-01	4.00E-02	3.67E-01	2.28E-02
	(1.00E-03)	(2.19E-02)	(4.45E-03)	(5.98E-02)	(3.22E-02)	(9.90E-02)	(1.63E-02)
<i>Spread</i>	1.77E-01	2.01E-01	5.21E-01	1.44E+00	8.45E-01	9.63E-01	3.07E-01
	(1.20E-01)	(1.56E-01)	(4.03E-01)	(1.04E+00)	(6.38E-01)	(6.35E-01)	(2.89E-01)
<i>Epsilon</i>	1.19E-02	1.04E-01	9.52E-02	1.54E-01	1.40E-01	3.00E-02	2.58E-02
	(4.21E-03)	(8.43E-02)	(7.84E-02)	(1.35E-01)	(1.04E-01)	(1.93E-02)	(2.07E-02)

which are capped at both ends by hemispherical heads, and to maximize the storage capacity (f_2) as shown in Fig. 3.10. The mathematical formulation of this problem is formulated in Appendix C.2.3. Table 3.48 shows the comparisons of best optimal solution for MOSHO and other reported algorithms. According to this table, MOSHO is able to find the optimal design with minimum cost and maximum storage and able to find a good distribution of points.

3.3.1.4.4 Speed Reducer Design Problem

The design of speed reducer is a most challenging test problem which is associated with seven design variables^[145]. In this optimization problem (see Fig. 3.11), the weight of speed reducer (f_1) and stress of a speed reducer (f_2) should

Table 3.49: Comparison of best solution obtained from different algorithms for speed reducer design problem.

Metrics	MOSHO	MOPSO	NSGA-II	MOEA/D	PESA-II	SPEA2	MOACO
<i>Hypervolume</i>	3.22E-00 (2.20E-01)	6.97E-01 (4.35E-01)	6.00E-01 (5.62E-01)	8.63E-01 (6.69E-01)	9.95E-01 (8.03E-01)	8.64E-01 (6.14E-01)	5.11E-01 (4.00E-01)
Δ_p	4.13E-03 (1.57E-03)	5.07E-02 (4.84E-03)	3.37E-02 (3.03E-02)	7.68E-02 (2.00E-02)	5.00E-01 (8.93E-02)	6.42E-03 (4.66E-03)	5.99E-02 (3.57E-03)
<i>Spread</i>	1.69E-01 (1.22E-01)	3.36E-01 (2.15E-01)	7.63E-01 (4.32E-01)	1.97E+00 (1.45E+00)	4.52E-01 (6.32E-01)	6.95E-01 (4.44E-01)	3.11E-01 (2.52E-01)
<i>Epsilon</i>	1.00E-02 (5.09E-03)	1.09E-01 (1.02E-01)	1.39E-01 (8.62E-02)	1.96E-01 (7.63E-02)	6.53E-02 (4.25E-02)	1.34E-01 (1.96E-01)	2.75E-01 (4.54E-02)

be minimized with eleven constraints. The mathematical formulation of this problem is formulated in Appendix C.2.4. Table 3.49 shows the comparison of best solutions for six optimization algorithms and proposed MOPSO in terms of function values. From Table 3.49, it is evident that MOSHO produced best solution with considerable improvement as compared to other algorithms. MOSHO algorithm is able to find a design with minimum weight and stress, simultaneously. The MOSHO algorithm achieves better results and verified by finding a set of Pareto solutions.

3.4 Summary

In this chapter, two novel bio-inspired metaheuristic optimization algorithms are proposed named as spotted hyena optimizer (SHO) and emperor penguin optimizer (EPO). The fundamental concepts which motivate these methods are inspired by the social hierarchy of spotted hyenas and huddling behavior of emperor penguins in nature. The performance of the proposed SHO and EPO algorithms are tested on unimodal, multimodal, fixed-dimension multimodal, composite, and IEEE CEC-2015 benchmark test functions. Moreover, six real-life engineering design problems are employed to further examine the efficiency of SHO and EPO algorithms. The results demonstrate that SHO and EPO provide

extremely competitive results as compared to other well-known metaheuristics.

This chapter also presented the multi-objective version of SHO algorithm called multi-objective spotted hyena optimizer (MOSHO). The two components are introduced to solve multi-objective problem. The first component is an archive for retrieving and storing the best non-dominated solutions which are obtained so far and the second component is a group selection mechanism of nearby optimal solutions from the archive. MOSHO has been tested on twenty-four well-known benchmark test functions. The proposed method used for solving the six real-life optimization problems perhaps require less computational efforts to find the optimum solutions.

Chapter4

Automatic Clustering using Multi-objective Spotted Hyena Optimizer

“It is not the answer that enlightens, but the question.”

By Eugene Lonesco

4.1 Introduction

The main focus of this thesis is the partition based clustering technique. The most of partitional clustering algorithms require predefined number of clusters. However, the number of clusters can not be determined in advance for most of the real-life engineering problems. If the estimated number of cluster is less than the actual number of cluster, then dissimilar data points may be combined in one cluster. If the number of estimated clusters is greater than the actual number of cluster, then clusters get decomposed into smaller clusters. Therefore, the number of clusters is an important parameter for clustering and it greatly affects the performance of clustering algorithms. Researchers used metaheuristic techniques to solve this problem. Thus, an attempt has made in this thesis to determine the number of clusters automatically using metaheuristics.

4.2 Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO)

This section defines the inspiration and mathematical justification of proposed automatic data partitioning technique using multi-objective spotted hyena optimizer.

4.2.1 Inspiration

The main contribution of this work is to design an approach for automatic data partitioning without the prior knowledge of the number of clusters. A variable string length based clustering approach is proposed for finding the optimal number of clusters. The variable string is used for encoding the clusters centers with their threshold values. A novel method is proposed to set the value of threshold for cluster centers. The threshold value for cluster centers is set to their intra-cluster variation. However, the adjustment of cutoff threshold value for a given data set is computed as the average value of threshold setting for cluster rather than fixing a particular value. For data points assignment to different clusters, a weighted Euclidean distance is used in place of Euclidean distance to incorporate the effect of their cluster centers. The mathematical formulation of weighted Euclidean distance is as follows:

$$D_e^w(x_i, x_j) = \left(\sum_{l=1}^D w_i^2 |x_{jl} - m_{il}|^2 \right)^{1/2} \quad (4.1)$$

where D_e is the Euclidean distance, w_i represents the threshold value assigned to cluster center m_i , x_{jl} represents the l th dimension value of x_j . Moreover, MOSHO algorithm is used for computing the fitness of agents. The agent's fitness is taken

as a parameter to perform the search process.

4.2.2 Proposed Clustering Technique

The proposed Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO) steps are described in Algorithm 4.

Algorithm 4 Automatic Clustering using Multi-objective Spotted Hyena Optimizer (ACMOSHO)

- 1: Initialization of control parameters such as maximum number of iterations, number of clusters, and population size.
 - 2: Initialization of search agents is done in such a way that it contains the maximum number of cluster centers and their corresponding values of activation margins (see Section 4.2.2.1).
 - 3: Compute cluster centers which are active in each agent (see Section 4.2.2.1).
 - 4: Calculate the weighted Euclidean distance from all active cluster centers for each data point x_i .
 - 5: Assign x_i to cluster center with minimum distance compared to other cluster center.
 - 6: If data points are less than 2 then cluster centers re-initialization is done by using average computation method as described in Section 4.2.4.2.
 - 7: Calculate the fitness of search agents using MOSHO algorithm (see Section 4.2.4.1).
 - 8: Repeat Steps 3 - 7 till maximum number of iterations is reached.
 - 9: Optimal cluster centers are identified by best search agent.
-

4.2.2.1 Cluster Encoding

The proposed ACMOSHO is based on cluster-centroid based encoding technique to represent the clustering solutions. An agent is a vector of real numbers having dimension $K + K \times D$ for n data points where each having D dimensions. K is the user-defined clusters. The vector $(V_i(t))$ is defined in Fig. 4.1.

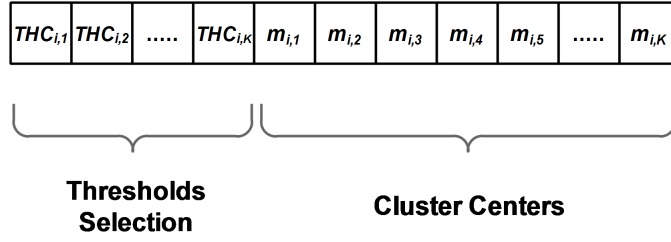


Fig. 4.1. Agent encoding scheme.

where $m_{i,j}$ is the j th cluster center of the i th agent. $THC_{i,j}$ is the threshold value of cluster-centroid $m_{i,j}$.

Example: Consider an example that assumes threshold cutoff value is taken as 0.5. As shown in Fig. 4.2, two thresholds values are greater than 0.5 (i.e., 0.7 and 0.6). The first (8.1, 9.6) and third (6.4, 2.3) cluster centers are selected for partitioning. During the iteration process, it may be possible that none of the threshold values are higher than cutoff threshold value. In this case, two thresholds are randomly selected from a data set.

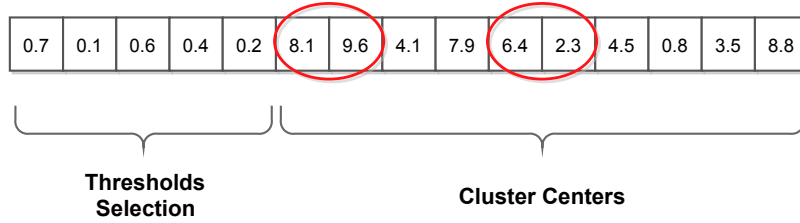


Fig. 4.2. Selection of thresholds and cluster centers.

4.2.3 Threshold Setting Computation

A novel threshold setting method for cluster center computation is proposed which is based on within-cluster variation and is defined as:

$$THC_l = \sqrt{\left(\frac{1}{n} \sum_{j=1}^n (x_j^l - C_l)^2 \right)} \quad (4.2)$$

where $l = 2, 3, \dots, K$. THC_l is the selection threshold with respect to cluster l . n represents the number of data points and x_j^l represents the j th data point that

belongs to cluster C_l .

4.2.4 Threshold Cutoff Computation

The cluster cutoff threshold value is set according to its mean values. The cutoff threshold for cluster selection is defined as:

$$T_{cutt_clus} = \frac{1}{K} \sum_{l=1}^K THC_l \quad (4.3)$$

4.2.4.1 Fitness Evaluation

I index^[12] and Sym index^[13] are chosen for clustering criteria. The first quantifies the amount of symmetry present in a particular partitioning whereas the latter one measures the compactness of partitioning.

The penalty function is used to ensure that at least two clusters present in a data set. By combining this function, the new clustering criterion is defined as:

$$Fit(K) = I(K) \times Sym(K) \times \frac{1}{K-1} \quad (4.4)$$

4.2.4.2 Cluster Center Validation

There may be a probability that at least two data points are given to cluster centers. To handle this situation, an average computation method is used to re-initialize the cluster center positions of particular agents^[162].

Example: Let 150 data points having 3-dimensional feature space in a given data set. Assume there are three active number of clusters and cluster centers for a particular agent are ((0.9, 0.42, -0.01), (4.0, 3.1, 0.8), (6.1,1.2, 0.8)). Here, cluster center (0.9, 0.42, -0.01) have no data point assignment. Average computational formula is used to assign data points to nearest clusters. The

updated cluster centers are (3.128, 2.269, 0.601), (2.900, 1.832, 0.266), and (4.789, 2.456, 0.876).

4.2.5 Computational Complexity

The space and time complexities of proposed ACMOSHO algorithm are discussed below.

4.2.5.1 Time Complexity

1. Initialization of ACMOSHO population needs $\mathcal{O}(n_a \times s_l)$ time, where n_a is the number of search agents and s_l is the length of each encoded agent.
2. It requires $\mathcal{O}(n_a \times K)$ for active cluster extraction.
3. The data points which are given to different clusters requires $\mathcal{O}(n^2 \times K)$.
4. For fitness calculation, the algorithm requires $\mathcal{O}(n \times K)$ time.
5. Steps 3 - 4 are repeated for all agents till the maximum number of iterations.

Hence, the time complexity of proposed ACMOSHO algorithm for maximum number of iterations is $\mathcal{O}(n^2 \times K \times n_a \times Max_{iteration})$.

4.2.5.2 Space Complexity

The space complexity of proposed algorithm is computed during its initial step. Therefore, the space complexity of proposed algorithm is $\mathcal{O}(n_a \times s_l)$.

4.3 Experimental Results and Discussions

This section evaluates the performance of proposed approach on nine real-life data sets. The findings of proposed technique are evaluated and compared with four well-known multi-objective clustering algorithms.

4.3.1 Data sets and Algorithms used for Comparisons

The clustering techniques are tested on nine data sets taken from UCI database^[163]. Table 4.1 presents a detailed description of these data sets. The four well-known multi-objective clustering algorithms are chosen to validate the performance of proposed ACMOSHO. These are GenClustMOO^[74], Multi-objective Invasive Weed Optimization Algorithm (MOIWO)^[68], Automatic Clustering using Multi-objective Particle Swarm and Simulated Annealing (MOPSOSA)^[73], and an Evolutionary Approach to Multi-objective Clustering (MOCK)^[57].

Table 4.1: Description of data sets.

Data sets	No. of data points	No. of features	No. of classes
Iris	150	4	3
Glass	214	9	6
Wine	178	13	3
Vowel	871	3	6
Breast Cancer	683	9	2
Bupa	345	6	2
Haberman	306	3	2
CMC	1473	9	3
WDBC	576	30	2

4.3.2 Performance Comparison

Table 4.2 shows the optimal clustering results obtained from ACMOSHO, GenClustMOO, MOIWO, MOPSOSA, and MOCK. GenClustMOO, MOIWO, and MOPSOSA generate 2 clusters for Iris data set whereas ACMOSHO and MOCK produce 3 clusters. For Wine data set, ACMOSHO generates 3 clusters during simulation runs. For Glass data set, the proposed ACMOSHO, MOCK, and GenClustMOO provide 6 partitions during simulation run. ACMOSHO, MOCK, and MOIWO provide 2 clusters for Breast Cancer data set. For Vowel data set, ACMOSHO generates 6 clusters. However, GenClustMOO, MOIWO, and MOPSOSA are unable to generate the optimal number of clusters on this data set. ACMOSHO, MOCK, GenClustMOO, MOIWO, and MOPSOSA provide 2 partitions

for Haberman data set. For Bupa data set, ACMOSHO, MOCK, and MOIWO provide 2 clusters during simulation runs. For WDBC data set, ACMOSHO, MOCK, GenClustMOO, MOIWO, and MOPSOSA provide 2 clusters in each run. For CMC data set, ACMOSHO, MOCK, GenClustMOO, and MOPSOSA provide 3 clusters whereas MOIWO provides 4 clusters in each iteration. Tables 4.3 and 4.4 show the intra- and inter-cluster distances generated by various clustering approaches. The results show that the proposed ACMOSHO obtains optimal clusters for almost all data sets. Hence, it has been concluded that ACMOSHO performs better than the other approaches.

The statistical significance of the proposed approach is done using unpaired t -test. The results are compared by best outperforming and second best outperforming clustering algorithms as shown in Table 4.5. The sample size is set to 40 for unpaired t -test. It can be seen that ACMOSHO is statistically significant for almost all data sets.

Table 4.2: Average and standard deviation of number of partitions obtained by the proposed ACMOSHO and competitive approaches on nine data sets over 40 independent runs.

Data sets	ACMOSHO	MOCK	GenClustMOO	MOIWO	MOPSOSA
Iris	2.943(1.217)	3.257(1.215)	2.237(1.412)	2.357(1.125)	2.507(1.932)
Wine	3.000(1.001)	3.257(1.048)	3.057(1.040)	2.957(1.012)	3.507(1.021)
Glass	6.000(1.253)	6.057(1.020)	5.957(1.069)	5.857(1.029)	5.607(1.069)
Breast Cancer	2.014(1.002)	2.007(1.092)	2.257(1.059)	2.007(1.009)	2.257(1.030)
Vowel	6.011(1.007)	5.757(1.0635)	7.257(1.020)	5.057(1.008)	7.507(1.049)
Haberman	2.003(0.000)	2.085(0.000)	2.412(0.055)	2.0475(0.010)	2.412(0.020)
Bupa	2.095(0.012)	2.142(0.085)	2.260(0.060)	2.102(0.015)	2.200(0.036)
WDBC	2.000(0.040)	2.012(0.086)	2.263(0.123)	2.007(0.010)	2.003(0.008)
CMC	3.000(0.000)	3.113(0.026)	3.050(0.032)	4.412(0.165)	3.453(0.179)

4.4 Application to Image Segmentation

Image segmentation is a technique of decomposing an image into different regions with similar features. Image segmentation problem can be contemplated as a

Table 4.3: Average and standard deviation of intra-cluster distance obtained by the proposed ACMOSHO and competitor algorithms on nine data sets over 40 independent runs.

Data sets	ACMOSHO	MOCK	GenClustMOO	MOIWO	MOPSOSA
Iris	1.35856(0.180)	3.1167(0.033)	3.6527(1.195)	3.5677(2.792)	3.9447(1.874)
Wine	1.0600(0.000)	4.0467(0.002)	4.8517(0.184)	4.1637(1.929)	4.9497(1.232)
Glass	561.4395(7.71)	563.2477(134.2)	599.5357(10.34)	594.6737(30.62)	608.8377(20.92)
Breast Cancer	6.0955(0.120)	4.2447(0.143)	4.8517(0.373)	4.9947(0.904)	4.6947(0.654)
Vowel	1412.5241(2.860)	1441.637(0.792)	1482.517(3.973)	1495.137(12.334)	1493.727(10.833)
Haberman	4.0052(0.123)	4.0125(0.102)	3.5241(0.442)	4.416(0.853)	4.2352(0.411)
Bupa	4.0044(0.096)	4.1254(0.102)	4.4121(0.255)	3.7740(0.863)	3.4410(0.500)
WDBC	4.0900(0.086)	4.1422(0.115)	4.6652(0.302)	4.7522(0.744)	4.2152(0.462)
CMC	1.5524(0.147)	2.1045(0.030)	3.4415(1.142)	3.5085(2.700)	2.6859(1.769)

Table 4.4: Average and standard deviation of inter-cluster distance obtained by the proposed ACMOSHO and competitor algorithms on nine real life data sets over 40 independent runs.

Data sets	ACMOSHO	MOCK	GenClustMOO	MOIWO	MOPSOSA
Iris	4.3012(1.540)	2.5937(1.028)	2.2107(1.770)	2.5057(1.524)	2.1157(1.000)
Wine	3.5000(1.000)	3.1487(1.056)	2.6117(1.052)	2.8057(1.300)	2.6117(1.258)
Glass	22.8221(2.524)	853.627(8.111)	889.327(3.125)	869.937(1.452)	891.827(3.854)
Breast Cancer	6.7120(1.254)	3.2577(1.524)	2.3617(1.669)	2.3947(1.856)	2.9647(1.968)
Vowel	620.8542(1.500)	2724.857(1.025)	1923.937(1.005)	1944.387(1.524)	2434.457(2.754)
Haberman	6.5241(0.108)	3.4125(0.120)	2.2411(0.020)	2.3412(1.652)	2.7745(1.300)
Bupa	6.4128(0.152)	3.2000(0.126)	2.2457(0.046)	2.3524(1.740)	2.7469(1.368)
WDBC	6.3324(0.141)	3.1245(0.147)	2.3241(0.034)	2.3412(1.5227)	2.5231(1.350)
CMC	4.2000(0.412)	2.2253(0.041)	2.2458(0.885)	2.5412(1.412)	2.2145(1.102)

clustering problem. It is mathematically described as follows:

$$\sum_{i=1}^n R_i = I \quad (4.5)$$

such that,

$$R_i \cap R_j = \phi \quad (4.6)$$

The proposed algorithm is tested on four color images such as Mandrill, Airplane (F-16), Peppers, and House. The size of each image is 512×512 . The number of data points are 262144 for each image. The original images and clustered images are demonstrated in Fig. 4.3. For aforementioned images, the optimal range of clusters is given in Ref. [164]. Table 4.6 shows the results obtained from the proposed and other clustering techniques on color images. For Mandrill and

Table 4.5: Unpaired t -test between the best and second best algorithms on nine data sets.

Data sets	Standard error	t	95% confidence interval	Two-tailed P	Significance
Iris	0.033	8.4980	-0.37022 to -0.22970	<0.0001	Significant
Wine	0.007	40.5400	-0.26220 to -0.23765	<0.0001	Significant
Glass	0.089	0.3055	-0.22452 to -0.16455	<0.0001	Significant
Breast Cancer	0.003	0.8000	-0.23800 to -0.22176	<0.0001	Significant
Vowel	0.011	21.0531	0.22600 to 0.27354	<0.0001	Significant
Haberman	0.006	55.2000	-0.2412 to -0.23025	<0.0001	Significant
Bupa	0.002	12.1256	-0.22365 to -0.21254	<0.0001	Significant
WDBC	0.003	55.5247	-0.24578 to -0.23365	<0.0001	Significant
CMC	0.005	35.4587	-0.25241 to -0.23625	<0.0001	Significant

Airplane (F-16) images, ACMOSHO generates 6 clusters. ACMOSHO produces 7 and 5 partitions for Peppers and House images, respectively. The results are compared with MOCK, GenClustMOO, MOIWO, and MOPSOSA algorithms. Further, the significance of ACMOSHO is proved by using unpaired t -test as shown in Table 4.7. These results show the supremacy of proposed ACMOSHO over other approaches.

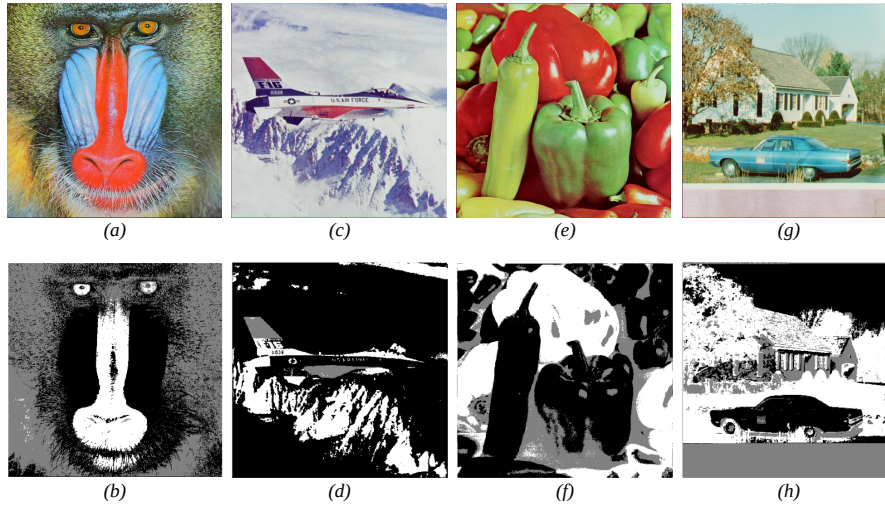


Fig. 4.3. (a) Mandrill image; (b) Segmented image by ACMOSHO; (c) Airplane (F-16) image; (d) Segmented image by ACMOSHO; (e) Peppers image; (f) Segmented image by ACMOSHO; (g) House image; (h) Segmented image by ACMOSHO.

Table 4.6: Average and standard deviation of clustering obtained by the proposed ACMOSHO and competitive algorithms on four color images over 40 independent runs.

Images	Cluster range	ACMOSHO	MOCK	GenClustMOO	MOIWO	MOPSOSA
Mandrill	5-10	6.000(0.696)	4.100(0.201)	4.001(0.126)	4.524(0.400)	3.000(0.000)
Airplane (F-16)	5-10	6.010(0.000)	4.017(0.750)	3.200(0.075)	6.310(0.090)	3.498(0.055)
Peppers	5-10	7.007(0.000)	6.007(0.028)	6.810(0.060)	3.152(0.366)	5.007(0.060)
House	5-10	5.017(1.203)	5.285(0.070)	4.657(0.674)	7.400(0.041)	5.225(0.013)

Table 4.7: Unpaired t -test between the outperforming and second outperforming algorithms on coloured images.

Image	Standard error	t	95% confidence interval	Two-tailed P	Significance
Mandrill	0.095	2.5800	-0.44586 to -0.05211	0.0121	Significant
Airplane (F-16)	0.196	5.4102	-1.51003 to -0.67569	<0.0001	Significant
Peppers	0.078	2.5204	-0.44125 to -0.05115	0.0112	Significant
House	0.230	1.7526	-0.86160 to 0.06160	0.0864	Not significant

4.5 Summary

In this chapter, a novel multi-objective based automatic clustering approach is proposed which utilizes two new concepts viz., weighted cluster-centroid computation and dynamic threshold setting. A variable string length method is used for encoding the clusters centers with their threshold values. A novel method is proposed to set the value of threshold for cluster centers. The experimental outcome indicates that the proposed technique is able to efficiently identify the optimal number of clusters and outperforms than the other clustering techniques.

Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer

“The essence of mathematics is not to make simple things complicated, but to make complicated things simple.”

By Stan Gudder

5.1 Introduction

Technological enhancement has led to tremendous growth in data due to increase in social media platforms and computational hardware. Such enormous data deals with various issues, i.e., high-dimensional nature of data and missing data values^[165]. In order to process high-dimensional data and find an optimal subset of feature is a challenging task^[166]. Data collection of such high-dimensional data may turn into redundant, irrelevant, and noisy data. Dimensionality reduction techniques can be employed to remove noisy, redundant data, and irrelevant features^[167]. These techniques are broadly classified into two categories namely feature extraction and feature selection. Feature extraction helps to convert high-dimensional data space into lower space. Feature selection techniques are used to minimize the feature subset, remove redundant, and correlated

features. Feature selection techniques are more reliable in terms of reliability and interpretability. It is the most widely used technique in various research fields such as image processing^[168], pattern recognition^[169], and so on^[170]. Feature selection algorithms are classified into two categories such as supervised and unsupervised^[171]. Mostly feature selection is used in supervised algorithms to classify the given data. But, it becomes critical to define feature selection strategy in the unsupervised environment as class labels are not provided. Unsupervised feature selection is one of the most challenging tasks. These are categorized into two main classes such as filter and wrapper^[172,171]. Filter method relies on the data set to define the relevant subset of features without considering the clustering algorithm. Whereas, wrapper method depends on unsupervised algorithm to find the optimal subset of features. Most of the literature is available on the wrapper feature selection methods. The performance of wrapper feature selection algorithms depends on the clustering criterion.

5.2 Proposed Feature Selection Technique

In this chapter, an automatic feature selection technique has been proposed for handling high-dimensional data sets. The optimal features are selected using cluster-centroid based encoding scheme. A novel threshold concept has been developed for feature ranking. A novel fitness function has been designed to evaluate the fitness of agents. The proposed Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer (AFSMOSHO) approach is described in Algorithm 5.

Algorithm 5 Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer (AFSMOSHO)

- 1: Initialize the population size, maximum number of iterations, and parameters of multi-objective spotted hyena optimizer.
 - 2: Search agents are initialized in such a way that it may contain the cluster centers with their corresponding threshold activation values (see Section 5.2.1).
 - 3: The active features in each cluster are found by evaluating the threshold limits (see Section 5.2.1).
 - 4: Euclidean distance is calculated from all active cluster centers to each data point.
 - 5: Each data point is assigned to a cluster with minimum distance with respect to others.
 - 6: If the number of data points assigned to a cluster is less than one (i.e., no data points belong to a particular cluster), then re-initialize the cluster centers of the agent using the procedure as described in Section 5.2.5.
 - 7: The fitness of agents is calculated using MOSHO algorithm (see Section 5.2.4).
 - 8: The values of selected thresholds are allocated to each feature (see Section 5.2.2).
 - 9: The thresholds cutoff are computed using selection threshold which is calculated in the previous step (see Section 5.2.3).
 - 10: Steps 3 - 9 are repeated until it reaches the maximum number of iterations.
 - 11: The best search agent returns the number of cluster centers and features.
-

5.2.1 Agent Representation and Initialization

The proposed AFSMOSHO employs centroid based scheme to encode the features and cluster centers. An agent is a vector having dimension $D + K \times D$ and n number of data points. Here, D represents the dimension and K represents the number of clusters. The values assigned to first D entries are positive numbers in the range $[0, 1]$. It indicates that the corresponding feature is activated or not activated in each cluster center during clustering. The rest entries are employed for K clusters centers having D dimension. The vector $V_i(t)$ of i th agent at particular time t is illustrated in Fig. 5.1.

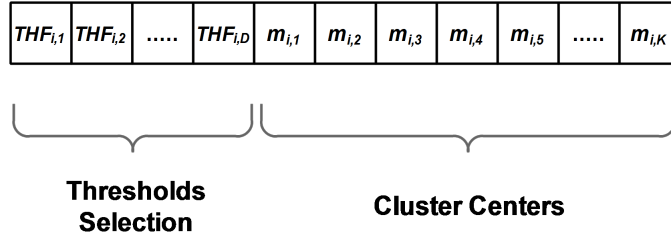


Fig. 5.1. Agent representation scheme.

where $THF_{i,D}$ is the threshold value of D th feature of i th agent and $m_{i,K}$ is the K th cluster center of i th agent. The rules for active feature extraction is defined below:

If $THF_{i,j} > T_{cutt_feat}$ **then**

j th feature in the i th agent is active

Else

j th feature in the i th agent is not active

Example: Let us assume that the threshold cutoff value T_{cutt_feat} is set as 0.5. Therefore, two thresholds values are greater than 0.5. These values are 0.6 and 0.8. The equivalent second and fourth features are selected for partitioning as shown in Fig. 5.2.



Fig. 5.2. Selection of features from four-dimensional data set.

5.2.2 Threshold Setting Computation

In this thesis, a new threshold setting method is proposed for feature selection which is computed as below:

$$THF_p = \frac{1}{K} \sum_{i=1}^K \left(\frac{Vr_p - Vr_{p,i}}{Vr_p} \right), \quad \text{where } p = 1, 2, \dots, D \quad (5.1)$$

where THF_p is the threshold value of p th feature and Var_p is the variance of p th feature in data set. $Var_{p,i}$ represents the variance of p th feature in the i th cluster. K represents the clusters that are used for partitioning the data set.

5.2.3 Threshold Cutoff Computation

The cutoff feature threshold value T_{cutoff_feat} is set according to the mean value of sum of thresholds in each feature. These are computed overall dimensions in case of feature thresholds. The cutoff threshold for feature selection is mathematically defined as:

$$T_{cutoff_feat} = \frac{1}{D} \sum_{p=1}^D THF_p \quad (5.2)$$

The cutoff threshold T_{cutoff_feat} is set to 0.5 during the initial step of algorithm.

5.2.4 Fitness Function Evaluation

I index^[12] and Sym index^[13] are chosen for clustering criteria. The first quantifies the amount of symmetry present in a particular partitioning. The latter one measures the compactness of partitioning.

The penalty function is included to ensure that at least two clusters present in a data set. The mathematical formulation of fitness function is given below:

$$Fit(K) = I(K) \times Sym(K) \times PF \quad (5.3)$$

where PF is a penalty function which is described below:

$$PF = \frac{D - d}{D - 1} \quad (5.4)$$

where D represents the total number of features and d is the number of selected features.

5.2.5 Cluster Center Validation

There may be a possibility that there are no data points are assigned to a cluster center, i.e., the data points are less than two. To handle this situation, cluster center positions of particular agents are re-initialized by an average computation method^[162].

Example: Let us assume that number of 180 data points having 3-dimensional feature space in a given data set. Assume there are three active number of clusters and cluster centers for a particular agent are $((0.8, 0.45, -0.05), (5.1, 3.3, 0.4), (5.7, 1.3, 0.7))$. Here, cluster center $(0.8, 0.45, -0.05)$ have no data point assignment. The average computational formula is employed to assign data points to nearest clusters. The updated cluster centers are $(3.324, 2.255, 0.672)$, $(2.756, 1.822, 0.271)$, and $(4.908, 2.576, 0.903)$.

5.3 Computational Complexity

In this section, both the time and space complexities of proposed AFSMOSHO are discussed.

5.3.1 Time Complexity

1. Initialization of AFSMOSHO population needs $\mathcal{O}(n_a \times s_l)$ time, where n_a represents the number of agents and s_l represents the length of encoded search agent, respectively.
2. It requires $\mathcal{O}(n_a \times K \times D)$ for active feature extraction.
3. To calculate the minimum distance of data points which are assigned to different clusters, requires $\mathcal{O}(n^2 \times K)$. The updated cluster centers needs $\mathcal{O}(K)$ time.

4. For fitness calculation, the algorithm requires $\mathcal{O}(n \times K)$ time.

5. Repeat Steps 3 - 4 for all agents until the termination criteria is satisfactory.

Therefore, the overall time complexity of proposed AFSMOSHO algorithm for maximum number of iterations is $\mathcal{O}(n^2 \times K \times n_a \times Max_{iteration})$.

5.3.2 Space Complexity

The space complexity of proposed algorithm is computed during its initialization process. Therefore, the space complexity of proposed algorithm is $\mathcal{O}(n_a \times s_l)$.

5.4 Experimental Results and Discussions

This section evaluates the performance of proposed approach on twelve real-life benchmark functions. The results are evaluated and compared with five state-of-the-art multi-objective feature selection approaches.

5.4.1 Data sets used for Comparisons

The proposed AFSMOSHO is tested on twelve real-life data sets from UCI machine learning database^[163]. These data sets are divided into three categories: low-dimensional data set, i.e., $D \leq 10$, medium-dimensional data set, i.e., $10 \leq D \leq 100$, and high-dimensional data set, i.e., $D > 100$. Table 5.1 describes the detailed description of these data sets.

5.4.2 Experimental Setup

The algorithms are implemented and tested on MATLAB R2017a version. The maximum number of iterations and search agents are fixed as 500 and 80, respectively. The results have been measured over 40 independent simulation runs.

Table 5.1: Description of data sets.

Dimension	Data sets	No. of data points	No. of features	No. of classes
Low	Haberman	306	3	2
	Iris	150	4	3
	Diabetes	768	8	2
	Glass	214	9	6
	Cancer	683	9	2
Medium	Wine	178	13	3
	Waveform	5000	22	3
	Spambase	4601	57	2
	Splice	1000	60	2
	Movement Libras	360	90	15
High	Isolet	7797	617	26
	Multiple features	2000	649	10

5.4.3 Results and Discussions

The two sets of experimentations are implemented to find the relevance of feature selection in clustering. In the first set, all features of the data sets are considered and then clustering without feature selection is performed. In the second set, the proposed AFSMOSHO has been validated over data set for selecting relevant features and then cluster centers are selected according to the nominated features. Afterward, the clustering is done on the basis of nominated features. The results are analyzed by well-known cluster quality measures such as recall, precision, and F-measure^[173]. The average values of these measures are tabulated in Tables 5.2 - 5.4. Larger value of these metrics signifies efficient clustering. AFSMOSHO has advantage of selecting the small number of features for given data sets which are demonstrated in Tables 5.2 - 5.4. For Haberman, Iris, and Diabetes data sets, the proposed AFSMOSHO identifies 2 features during multiple runs. For Glass and Cancer data sets, AFSMOSHO identifies 4 features. In the case of Splice data set, it selects only 3 features from all the features. The proposed AFSMOSHO selects 4 features for Wine data set. In the case of Spambase and Libras data sets, AFSMOSHO identifies 36 and 47 features, respectively. For Waveform data set, AFSMOSHO selects 9 features. For Isolet and Multi features data sets, the proposed AFSMOSHO selects 268 and 283 features, respectively. Therefore, for

low-, medium-, and high-dimensional data sets, there is 44%, 41%, and 43% reduction in features on an average, respectively.

Table 5.2: The average(mean) performance of proposed AFSMOSHO on low-dimensional data sets.

Data sets		No. of features	Precision	Recall	FM
Haberman	Without FS	3.00	0.5744	0.6080	0.5540
	FS	2.00	0.6240	0.6452	0.6258
Iris	Without FS	4.00	0.2263	0.3263	0.3000
	FS	2.00	0.9613	0.9600	0.9593
Diabetes	Without FS	8.00	0.5075	0.5000	0.2620
	FS	2.30	0.5730	0.5042	0.4192
Glass	Without FS	9.00	0.2522	0.1181	0.1202
	FS	3.80	0.3300	0.2240	0.2310
Cancer	Without FS	9.00	0.9430	0.9010	0.9170
	FS	4.60	0.9540	0.9320	0.9412

Table 5.3: The average(mean) performance of proposed AFSMOSHO on medium-dimensional data sets.

Data sets		No. of features	Precision	Recall	FM
Wine	Without FS	13.00	0.2871	0.2341	0.2592
	FS	4.70	0.6058	0.5993	0.5608
Waveform	Without FS	21.00	0.3931	0.4563	0.4350
	FS	9.32	0.4555	0.4611	0.4450
Spambase	Without FS	57.00	0.1960	0.4991	0.3132
	FS	36.00	0.2300	0.4995	0.3145
Splice	Without FS	60.00	0.4530	0.4790	0.4000
	FS	3.00	0.5235	0.5217	0.5167
Libras	Without FS	90.00	0.0214	0.0360	0.0272
	FS	47.52	0.0750	0.1346	0.0800

Table 5.4: The average(mean) performance of proposed AFSMOSHO on high-dimensional data sets.

Data sets		No. of features	Precision	Recall	FM
Isolet	Without FS	616.00	0.0298	0.0498	0.0350
	FS	268.00	0.0371	0.0494	0.0380
Multiple features	Without FS	646.00	0.1020	0.0900	0.0955
	FS	283.00	0.1560	0.1200	0.1318

5.4.4 Relevance of Selected Features

The selected features can be estimated by the frequency of selected features and their Laplacian Score (LS). The feature frequency N_{f_i} is computed as:

$$f_r(N_{f_i}) = \frac{R_{N_{f_i}}}{R_I} \quad (5.5)$$

where $R_{N_{f_i}}$ represents the count of a particular selected feature and R_I represents the count of distinct runs. Table 5.5 shows the features frequency nominated for different data sets such as Haberman, Iris, Diabetes, Glass, Cancer, and Wine. The results show that AFSMOSHO nominates 1 and 3 feature of Haberman data set. The frequency of nomination of these features is one or approximately one. It can be articulated that LS of selected features was large. AFSMOSHO selects features 3 and 4 very often in Iris data set because LS of these features was high. The most frequently nominated features are 4 and 5 in Diabetes data set. These have a high value of LS . The features 1, 3, 4, and 7 of Glass data set are nominated frequently. These features have a large LS value. In case of Cancer data set, the often nominated features are 1, 2, 3, 6, and 7 as LS of these features are large. The features 4, 5, 8, and 9 have a low LS value. In Wine data set, the features 1, 6, 7, and 13 differ often. These features have a high LS value. The results show that AFSMOSHO can nominate features that comprise more information.

5.4.5 Performance Evaluation

The performance of proposed AFSMOSHO has been compared with five multi-objective feature selection approaches. These are Fuzzy rule based (FR-FS) method^[174], Bat algorithm based method (BBA-FS)^[175], Ranking based method (RB-FS)^[176], Differential Evolution based Multi-objective Feature Selection (DEMOFS)^[177], and Multi-objective Genetic Algorithm (MOGA-FS) based technique^[178].

Table 5.5: The obtained features frequency and their corresponding Laplacian Score (LS) using AFSMOSHO algorithm.

Data sets		Feature number												
		1	2	3	4	5	6	7	8	9	10	11	12	13
Haberman	fr	1.00	0.19	0.78	-	-	-	-	-	-	-	-	-	-
	LS	0.97	0.88	0.94	-	-	-	-	-	-	-	-	-	-
Iris	fr	0.00	0.00	1.00	1.00	-	-	-	-	-	-	-	-	-
	LS	0.94	0.83	0.97	0.95	-	-	-	-	-	-	-	-	-
Diabetes	fr	0.05	0.12	0.23	0.80	0.65	0.05	0.06	0.24	-	-	-	-	-
	LS	0.35	0.92	0.85	0.85	0.97	0.56	0.07	0.70	-	-	-	-	-
Glass	fr	0.73	0.00	1.00	0.93	0.00	0.05	0.81	0.17	0.00	-	-	-	-
	LS	0.75	0.75	0.93	0.71	0.70	0.47	0.90	0.54	0.14	-	-	-	-
Cancer	fr	0.93	1.00	1.00	0.00	0.00	1.00	0.67	0.00	0.00	-	-	-	-
	LS	0.90	0.92	0.92	0.90	0.83	0.94	0.87	0.90	0.75	-	-	-	-
Wine	fr	0.72	0.23	0.05	0.30	0.17	0.61	0.73	0.06	0.06	0.30	0.18	0.18	1.00
	LS	0.40	0.14	0.10	0.22	0.53	0.37	0.44	0.14	0.15	0.17	0.17	0.30	0.97

The comparative results by means of aforesaid methods for various data sets are depicted in Tables 5.6 and 5.7. It is perceived from Table 5.6 that Representational Entropy (RE) values for Iris and Cancer data sets are 1.47 and 1.81, respectively. The AFSMOSHO and competing approaches, i.e., FR-FS, RB-FS, and MOGA-FS nominated the features 3 and 4 in Iris data set. Therefore, the results demonstrated in Table 5.6 are found to be identical for these algorithms. In case of Cancer data set, the RE value is greater than those of FR-FS, BBA-FS, RB-FS, DEMOFS, and MOGA-FS. In case of medium-dimensional data set, that is Spambase and Waveform, it is perceived that the efficiency of AFSMOSHO is superior than the other algorithms. In case of multiple features, the performance of RB-FS is superior than the other approaches. In case of Isolet data set, the RE value is perceived to be greater than those of MOGA-FS, DEMOFS, and BBA-FS. In Table 5.7, it is seen that AFSMOSHO provides improved Class Separability (CS) as compared to other feature selection approaches for different types of data sets. In case of Iris data set, CS value is lesser than those of FR-FS, BBA-FS, RB-FS, DEMOFS, and MOGA-FS.

Table 5.6: The comparison results between feature selection approaches using Representational Entropy (RE).

Data sets	AFSMOSHO	FR-FS	BBA-FS	RB-FS	DEMOFS	MOGS-FS
Isolet	2.96	2.88	2.45	2.80	2.56	2.13
Multiple features	2.77	2.47	2.15	2.00	2.09	2.22
Spambase	2.12	2.00	1.04	1.53	1.47	1.36
Waveform	2.98	2.75	2.44	2.19	1.46	1.77
Cancer	1.81	0.96	1.38	0.92	1.47	1.53
Iris	1.47	0.99	0.41	1.33	1.21	1.16

Table 5.7: The comparison results between feature selection approaches using Class Separability (CS).

Data sets	AFSMOSHO	FR-FS	BBA-FS	RB-FS	DEMOFS	MOGS-FS
Isolet	0.90	1.78	1.40	1.80	1.56	1.11
Multiple features	1.65	1.97	2.10	2.18	1.76	1.82
Spambase	1.10	1.96	1.74	1.50	2.00	1.95
Waveform	0.90	0.95	1.13	1.05	1.01	1.17
Cancer	1.20	1.94	1.90	1.53	1.22	1.42
Iris	1.46	1.91	1.55	1.74	1.86	1.66

5.5 Summary

In this chapter, the proposed technique uses MOSHO algorithm for selecting the optimal number of features. The proposed technique helps to deal with high-dimensional data and automatically selects the optimal subset of features. The proposed technique introduces the concept of dynamic threshold setting and cutoff to identify the optimal feature subset. A novel fitness function is used to define the fitness of each feature subset. The proposed technique is validated against 12 real life data set with diversity in dimensionality. The results reveal that the proposed technique outperforms the existing technique in terms of representational entropy and class separability.

Chapter6

Simultaneously Clustering and Feature Selection using Multi-objective Spotted Hyena Optimizer

“Sometimes the questions are complicated and the answers are simple.”

By Dr. Seuss

6.1 Introduction

The major focus of this chapter is partitional based multi-objective clustering technique. It is used to deal with two major clustering issues such as determining the number of cluster prior the clustering process and finding the corresponding best cluster partitions. To solve these problems, partitional based multi-objective clustering technique run repeatedly with different number of classes as input and then to select the partitioning of the data that result in the best cluster validity indices. Single cluster validity index is unable to detect the optimal number of clusters. Hence, there is a need to build a novel fitness function which incorporates more than one cluster validity indices to maintain the inter- and intra-cluster properties. An efficient feature selection technique is required to enhance the

efficiency of algorithm and minimize the redundancy for unsupervised data sets. Feature selection in unsupervised learning is a difficult problem because of unavailability of class labels that can be used as a counsellor for searching relevant information. Hence, a novel multi-objective metaheuristic based technique is required for efficient clustering and feature selection.

6.2 Proposed Automatic Clustering and Feature Selection Approach using Multi-objective Spotted Hyena Optimizer

This section describes the contribution followed by the proposed automatic clustering and feature selection technique.

6.2.1 Contribution

The main contribution of this chapter is to automate clustering and feature selection using astrophysics based multi-objective spotted hyena optimizer. A variable composite agent representation scheme is designed for encoding both clusters and feature subsets efficiently. A novel threshold concept is also proposed to find out the optimal number of clusters and features separately. The proposed threshold inhibits the variance present in the given data set. The composite agent encodes cluster centers and features with their corresponding threshold values. A novel fitness function is proposed to compute the agent's fitness values.

6.2.2 Astrophysics Concept

The proposed technique is based on the fact that planets move in an elliptical path around the sun (see Fig. [6.1](#)). The planets closer to sun has the potential in becoming candidate solutions in exploitation. The elliptical shape helps in better

exploration and exploitation. Here, the sun is considered as prey and planets are analogous to spotted hyenas. Figs. 6.2(a) and 6.2(b) show the spherical and the elliptical orbit concepts, respectively. These are responsible for both exploration and exploitation in MOSHO.

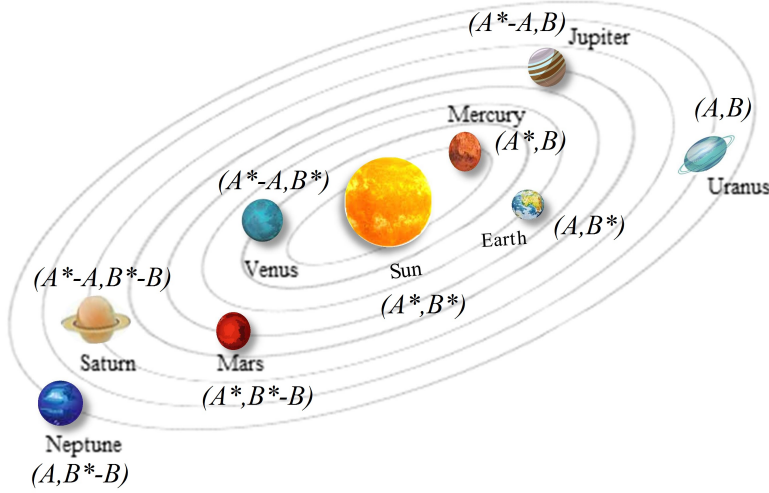


Fig. 6.1. Concept of astrophysics strategy.

6.2.3 Astrophysics based Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)

In this section, the concept of path followed by planets around the sun is presented. Firstly, R numbers of spotted hyenas are randomly generated. Thereafter, the positions of these R spotted hyenas ($\vec{P}_k, k = 1, 2, \dots, R$) are updated based on the

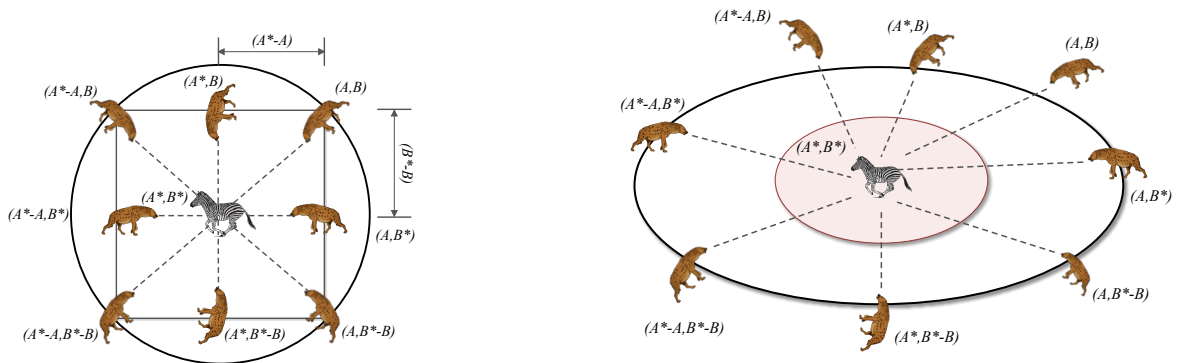


Fig. 6.2. Position vectors of spotted hyenas; (a) Spherical; (b) Elliptical.

position of best optimal solution ($\vec{P}(x+1)$). Hence, the new positions of R spotted hyenas are computed as:

$$\vec{P}(x+1) = \frac{\vec{C}_h}{N} + (E_l(P_k) \times D_e) \quad (6.1)$$

$$E_l(P_k) = (P_1 \times P_2 \times \dots \times P_R) \times \pi \quad (6.2)$$

where $E_l(P_k)$ indicates the elliptical positions of spotted hyenas and D_e is the Euclidean distance between the positions of spotted hyenas to remove the conflict between each other. $\vec{P}(x+1)$ saves the best optimal solution which is obtained so far and N is the number of optimal solutions.

6.2.4 Automatic Clustering and Feature Selection using Astrophysics based Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)

The proposed Astrophysics based Multi-objective Spotted Hyena Optimizer for Automatic Clustering and Feature Selection (AMOSHO_CFS) approach is described in Algorithm 6.

6.2.4.1 Agent Representation and Initialization

The proposed AMOSHO_CFS employed a variable composite agent to encode both features and clusters. An agent consists of real numbers of dimension $(K_{max} + D) + (K_{max} \times D)$ for n number of data points each having D dimensions, and K_{max} as user-defined clusters. The value of K_{max} is set as follows:

$$K_{max} = \sqrt{D} \quad (6.3)$$

The values assigned to $(K_{max} + D)$ are positive numbers in range of $[0, 1]$. The first K_{max} entry indicates that the belonging cluster is activated or not during clustering

Algorithm 6 Automatic Clustering and Feature Selection using Multi-objective Spotted Hyena Optimizer (AMOSHO_CFS)

- 1: The parameters such as number of population, maximum number of iterations, and user-defined maximum number of clusters (K_{max}) are initialized.
 - 2: Search agents are initialized in such a way that it may contain the cluster centers with their corresponding threshold activation values (see Section 6.2.4.1).
 - 3: The active cluster centers and features are found by evaluating the threshold limits (see Section 6.2.4.4).
 - 4: Euclidean distance is calculated from all active cluster centers to each data point.
 - 5: Each data point is assigned to a cluster that having minimum distance with respect to other clusters.
 - 6: However, if the data points are less than one (i.e., no data points belong to a particular cluster), then re-initialize the cluster centers of the agent using the procedure mentioned in Section 6.2.4.5.
 - 7: Fitness of agents is calculated using astrophysics based MOSHO algorithm (see Section 6.2.4.6).
 - 8: The values of selected thresholds are assigned to clusters and feature separately (see Section 6.2.4.2).
 - 9: The thresholds cutoff are computed for each cluster center and feature based on selection threshold which is calculated in the previous step (see Section 6.2.4.3).
 - 10: Steps 3 - 9 are repeated until the termination criterion is satisfied.
 - 11: The best search agent will return the optimal feature subset, cluster centers, and number of clusters.
-

process. D entry indicates that the features are activated or not. $(K_{max} \times D)$ represents K_{max} cluster centers of D dimension. The vector $V_i(t)$ of i th agent at particular time t is shown in Fig. 6.3.

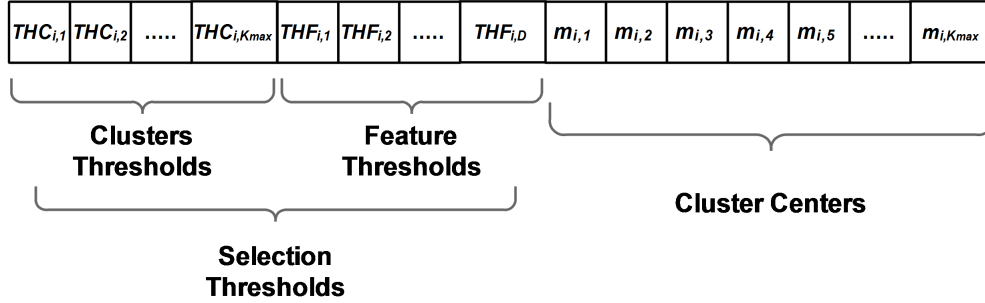


Fig. 6.3. Agent representation scheme.

where $cr_{i,j}$ is the j th cluster center of i th agent. THC_{ij} represents the value of threshold of cluster center $cr_{i,j}$. $THF_{i,j}$ is the value of threshold of j th feature in each cluster center of i th agent. $THC_{i,j}$ and $THF_{i,j}$ are selection thresholds for selecting active cluster centers and features, respectively.

Example 1: Let us assume that the user-defined number of clusters $K_{max} = 4$ in three dimensional space, i.e., $D = 3$.

The first entries (i.e., 0.5, 0.6, 0.3, 0.2) indicate the cluster selection thresholds with respect to four clusters. Then, the next three entries (i.e, 0.6, 0.2, 0.7) represent the feature selection thresholds and the last entries represent four cluster centers, i.e., (4.8, 3.1, 1.5), (5.6, 4.3, 0.9), (6.8, 2.9, 4.8), and (7.6, 3.0, 2.3).

6.2.4.2 Threshold Setting Computation

A novel threshold setting method for cluster center computation is proposed which is based on within-cluster variation and is defined as:

$$THC_l = \sqrt{\left(\frac{1}{n} \sum_{j=1}^n (x_j^l - C_l)^2 \right)} \quad (6.4)$$

where $l = 2, 3, \dots, K_{max}$. THC_l is the selection threshold with respect to cluster l . n represents the number of data points and x_j^l represents the j th data point that belongs to cluster C_l .

To determine the number of clusters automatically, it is necessary that the same feature corresponds to each cluster. Hence, the threshold setting mechanism is proposed for finding an efficient number of features. It is computed as:

$$THF_p = \frac{1}{K} \sum_{i=1}^K \left(\frac{Vr_p - Vr_{p,i}}{Vr_p} \right), \quad \text{where } p = 1, 2, \dots, D \quad (6.5)$$

where THF_p is threshold value of p th feature and Vr_p is variance of p th feature. $Vr_{p,i}$ represents the variance of p th feature in i th cluster. K represents the clusters that are used for partitioning the data set. THF_p is the average value of significance of p th feature.

6.2.4.3 Threshold Cutoff Computation

The cutoff feature and cluster threshold values are set according to their mean values. These cutoff threshold values are computed the overall dimensions and clusters in case of feature thresholds and cluster thresholds, respectively. The cutoff threshold for cluster selection is defined as:

$$T_{cutt_clus} = \frac{1}{K_{max}} \sum_{l=1}^{K_{max}} THC_l \quad (6.6)$$

The cutoff threshold for feature selection is defined as:

$$T_{cutt_feat} = \frac{1}{D} \sum_{p=1}^D THF_p \quad (6.7)$$

In the initial stage of algorithm, the cutoff thresholds T_{cutt_clus} and T_{cutt_feat} are set to 0.5.

Example 2: The cluster center and feature selection thresholds are considered

as given in Example 1 (i.e., 0.5, 0.6, 0.3, 0.2) and (0.6, 0.2, 0.7), respectively. Based on the threshold cutoff computation approach, T_{cutt_clus} and T_{cutt_feat} are 0.5, respectively.

6.2.4.4 Active Cluster Center and Feature Extraction

The selection of cluster centers and features in an agent depend upon their cutoff threshold values, i.e., T_{cutt_clus} and T_{cutt_feat} . The rules for active cluster center and feature extraction are described as:

If $THC_{i,j} > T_{cutt_clus}$ **then**

j th cluster in the i th agent is active (i.e, $m_{i,j}$)

Else

j th cluster in the i th agent is not active

EndIf

If $THF_{i,j} > T_{cutt_feat}$ **then**

j th feature in each cluster centers for i th agent is active

Else

j th feature in each cluster centers for i th agent is not active

EndIf

There is a possibility that no thresholds are greater than the cutoff threshold values, i.e, T_{cutt_clus} and T_{cutt_feat} . In this situation, two thresholds values are selected randomly and re-initialize them.

Example 3: The cluster center and feature selection thresholds are (0.5, 0.6, 0.3, 0.2) and (0.6, 0.2, 0.7), respectively. The thresholds cutoff for both cluster center and feature selection is 0.5. Firstly, the cluster center thresholds are employed to select the number of clusters. Only two thresholds values (i.e., 0.5 and 0.6)

are greater than 0.5. Therefore, the thresholds of feature selection are used to choose the features from their corresponding clusters (see Fig. 6.4). Hence, two thresholds (i.e., 0.6 and 0.7) are greater than the cutoff threshold value (i.e., 0.5) for feature selection.

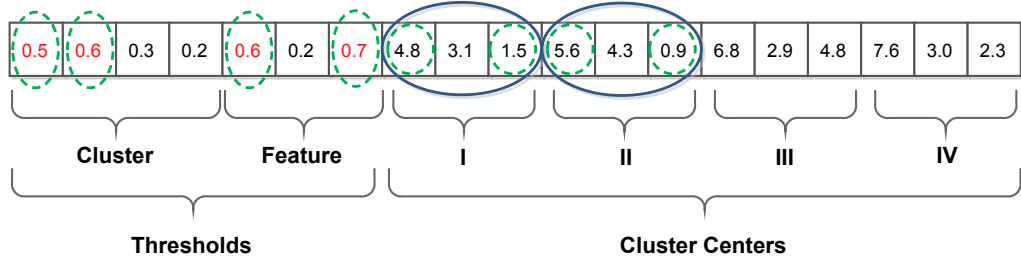


Fig. 6.4. Proposed agent encoding scheme.

6.2.4.5 Cluster Center Validation

The proposed technique may suffer from empty clusters problems. It means that the cluster has no data point. This issue generally arises because of misplacement of cluster center as compared to data distribution. To deal with empty clusters, the re-initialization of cluster centers can be done. The average computation formula is used to allocate n/K data points to the nearest cluster center.

Example 4: Let us assume that the number of 150 data points having 3-dimensional feature space in a given data set. There are three active number of clusters and cluster centers for a particular agent are $((0.8, 0.70, -0.08), (8.0, 5.6, 0.9), (8.4, 2.8, 0.2))$. Here, cluster center $(0.8, 0.70, -0.08)$ have no data point assignment. This is the case with outside cluster data point allocation. Average computational formula is used to assign data points to nearest clusters. The updated cluster centers are $(3.532, 2.115, 0.653)$, $(2.769, 1.748, 0.297)$, and $(4.752, 2.790, 0.787)$.

6.2.4.6 Fitness Function Evaluation

I index^[12] and Sym index^[13] are as cluster quality measures. I index measures the amount of symmetry present in a particular partitioning. Sym index measures the compactness of partitioning.

The penalty function P_{term1} is introduced to find the optimal number of clusters during run time and is mathematically defined as:

$$P_{term1} = K_{max} - K \quad (6.8)$$

Most of the clustering criteria do not consider the appropriate number of features. Hence, another penalty function P_{term2} is employed for selecting the optimal number of features d from total features D . It is defined as:

$$P_{term2} = \frac{D - d}{D - 1} \quad (6.9)$$

By combining these terms, a novel clustering criterion is defined as follows:

$$Fit(K) = I(K) \times Sym(K) \times P_{term1} \times P_{term2} \quad (6.10)$$

6.2.5 Computational Complexity

The space and time complexities of AMOSHO_CFS are discussed below.

6.2.5.1 Time Complexity

1. Initialization of AMOSHO_CFS population needs $\mathcal{O}(n_a \times d_s)$ time, where n_a is the number of agents and d_s is the distance of each encoded agent, respectively.
2. It requires $\mathcal{O}(n_a \times K_{max})$ for active cluster and feature extraction.

3. The data points are assigned to different clusters requires $\mathcal{O}(n^2 \times K_{max})$.
4. For fitness calculation, the algorithm requires $\mathcal{O}(n \times K_{max})$ time.
5. Steps 3 - 4 are repeated for all agents until the termination criteria is satisfied (i.e., maximum number of iterations).

Hence, the total time complexity of AMOSHO_CFS algorithm for maximum number of iterations is $\mathcal{O}(n^2 \times K_{max} \times n_a \times Max_{iteration})$.

6.2.5.2 Space Complexity

The space complexity of proposed algorithm is computed during its initial step. Therefore, the space complexity of the proposed AMOSHO_CFS algorithm is $\mathcal{O}(n_a \times d_s)$.

6.3 Experimental Results and Discussions

This section presents the performance of proposed approach on eight real-life benchmark functions. The results are evaluated and compared with four well-known multi-objective algorithms namely FeaClusMOO^[179], FeaClusMOO_EUC^[179], VGAPS^[12], and K -means^[180].

6.3.1 Data sets used

The efficiency and effectiveness of proposed AMOSHO_CFS is tested on eight real-life data sets namely Iris, Wine, Glass, Breast Cancer, Vowel, Haberman, Bupa, and CMC^[163]. The characteristics of these data sets are tabulated in Table 6.1.

Table 6.1: Description of data sets.

Data sets	No. of data points	No. of features	No. of classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
Breast Cancer	683	9	2
Vowel	871	3	6
Haberman	306	3	2
Bupa	345	6	2
CMC	1473	9	3

6.3.2 Experimental Setup

The above-mentioned algorithms are implemented and tested in MATLAB R2016a version. The platform used for experimentation is Microsoft Windows 10 having 64 bits and core i7 processor. The search agents and maximum number of iterations of the proposed approach are set as 30 and 100, respectively.

6.3.3 Relevance of Selected Features

The selected features are estimated by the frequency of selected features and their Laplacian Score (LS). The feature frequency N_{f_i} is defined as:

$$f_r(N_{f_i}) = \frac{R_{N_{f_i}}}{R_I} \quad (6.11)$$

where $R_{N_{f_i}}$ represents the number of times a particular feature is selected and R_I is the total number of independent runs. Table 6.2 shows the features frequency selected from AMOSHO_CFS and their LS for eight data sets. From Table 6.2, it is shown that AMOSHO_CFS selects features 3 and 4 of Iris data set. These features have a high LS value. For Wine data set, AMOSHO_CFS selects features 1, 6, 7, and 13. For Glass data set, the frequently selected features are 1, 3, 4, and 7. For Haberman data set, the selected features are 1 and 3. For Bupa data set, AMOSHO_CFS selects features 2 and 5 as LS values of these features are high. For Cancer data set, the selected features are 1, 2, 3, 6 and 7. For Vowel data set, the

selected feature is 2. For CMC data set, the features 1 and 4 are selected. The LS values of these features are high.

6.3.4 Relevance of Selected Clusters

The frequency of number of clusters with their Classification Accuracy (AC) values is selected to differentiate the important clusters from others. The mathematical formulation of frequency of clusters N_{c_i} is as follows:

$$Fr(N_{c_i}) = \frac{R_{N_{c_i}}}{R_I} \quad (6.12)$$

where $R_{N_{c_i}}$ represents the number of times a particular cluster is selected and R_I is the total number of independent runs. Table 6.3 demonstrates the count of selected clusters and their corresponding classification accuracy for eight data sets such as Iris, Wine, Glass, Haberman, Bupa, Cancer, Vowel, and CMC. For Iris data set, the proposed AMOSHO_CFS generates 3 clusters in each run with its corresponding high classification accuracy. For Wine data set, AMOSHO_CFS generates 3 clusters based on classification accuracy. The cluster count for Glass data set is 5. The cluster counts for Haberman data set is 2. The number of clusters for Bupa data set is 2 and their cluster accuracy is high. For Cancer data set, the proposed AMOSHO_CFS produces 6 clusters. For Vowel data set, the number of clusters is 2. The cluster count for CMC data set is 3 with high classification accuracy.

6.3.5 Performance Evaluation

Table 6.5 shows the mean and standard deviation of cluster quality measures obtained from the proposed and other clustering algorithms.

For Iris data set, AMOSHO_CFS obtains the optimal classification accuracy as compared to other competitor approaches such as FeaClusMOO,

Table 6.2: The obtained features frequency and their corresponding Laplacian Score (LS) using AMOSHO_CFS algorithm.

Data sets		Feature number												
		1	2	3	4	5	6	7	8	9	10	11	12	13
Iris	fr	0.00	0.00	1.00	1.00	-	-	-	-	-	-	-	-	-
	LS	0.95	0.84	0.98	0.96	-	-	-	-	-	-	-	-	-
Wine	fr	0.74	0.24	0.05	0.30	0.18	0.62	0.74	0.05	0.05	0.30	0.18	0.18	1.00
	LS	0.40	0.14	0.10	0.23	0.54	0.38	0.45	0.13	0.15	0.18	0.17	0.30	0.98
Glass	fr	0.74	0.00	1.00	0.93	0.00	0.05	0.81	0.17	0.00	-	-	-	-
	LS	0.76	0.75	0.94	0.71	0.70	0.50	0.91	0.55	0.14	-	-	-	-
Haberman	fr	0.90	0.00	0.10	-	-	-	-	-	-	-	-	-	-
	LS	0.97	0.88	0.95	-	-	-	-	-	-	-	-	-	-
Bupa	fr	0.07	0.80	0.46	0.04	0.70	0.00	-	-	-	-	-	-	-
	LS	0.35	0.88	0.82	0.70	0.95	0.27	-	-	-	-	-	-	-
Cancer	fr	0.93	1.00	1.00	0.00	0.00	1.00	0.68	0.00	0.00	-	-	-	-
	LS	0.90	0.93	0.92	0.90	0.83	0.95	0.88	0.92	0.76	-	-	-	-
Vowel	fr	0.00	1.00	0.08	-	-	-	-	-	-	-	-	-	-
	LS	0.95	0.98	0.97	-	-	-	-	-	-	-	-	-	-
CMC	fr	0.80	0.08	0.08	0.80	0.08	0.00	0.13	0.00	0.00	-	-	-	-
	LS	0.98	0.76	0.70	0.90	0.35	0.30	0.70	0.70	0.22	-	-	-	-

FeaClusMOO_EUC, VGAPS, and KM. The proposed algorithm selects the features 3 and 4 during the simulation run. For Wine data set, AMOSHO_CFS achieves the best classification accuracy as compared to other competitive methods. The number of clusters and number of features obtained from AMOSHO_CFS is 3 and 4, respectively.

For Glass data set, the proposed AMOSHO_CFS algorithm produces the number of clusters as 6. The number of features obtained from FeaClusMOO, FeaClusMOO_EUC, VGAPS, KM, and AMOSHO_CFS is 4. For Haberman data set, AMOSHO_CFS provides the optimal classification accuracy. The classification accuracies obtained from FeaClusMOO and FeaClusMOO_EUC are 56.9 and 55.8%, respectively. The classification accuracies obtained from KM and VGAPS are 56.7% and 56.3%, respectively. For Bupa data set, AMOSHO_CFS obtains the number of clusters as 2 and number of features as 2. Therefore, the classification accuracy obtained from proposed AMOSHO_CFS is better than the other approaches. For Cancer data set, all the above-mentioned approaches produce the accurate number of clusters and features. The classification accuracy obtained from AMOSHO_CFS is better than the other algorithms. For Vowel and

Table 6.3: The obtained clusters frequency and their corresponding Classification Accuracy (AC) using AMOSHO_CFS algorithm.

Data sets		Number of clusters											
		2	3	4	5	6	7	8	9	10	11	12	13
Iris	Fr	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-	-	-
	AC	0.95	0.80	0.68	0.60	0.51	0.50	0.45	0.45	0.40	-	-	-
Wine	Fr	0.85	0.08	0.00	0.00	0.00	0.00	0.00	0.00	-	-	-	-
	AC	0.65	0.55	0.53	0.37	0.37	0.32	0.30	0.30	-	-	-	-
Glass	Fr	0.00	0.00	0.17	0.65	0.12	0.00	0.00	0.00	-	-	-	-
	AC	0.36	0.40	0.40	0.80	0.55	0.52	0.52	0.48	-	-	-	-
Haberman	Fr	0.26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-	-	-	-
	AC	0.45	0.35	0.26	0.26	0.26	0.22	0.20	0.17	-	-	-	-
Bupa	Fr	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-	-	-	-
	AC	0.45	0.43	0.37	0.34	0.33	0.30	0.27	0.20	-	-	-	-
Cancer	Fr	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	AC	0.92	0.75	0.70	0.65	0.62	0.55	0.50	0.50	0.45	0.42	0.30	0.25
Vowel	Fr	0.00	0.00	0.09	0.80	0.08	0.00	0.00	0.00	0.00	0.00	0.00	-
	AC	0.30	0.36	0.45	0.55	0.50	0.45	0.42	0.40	0.35	0.34	0.35	-
CMC	Fr	0.90	0.04	0.00	0.00	0.00	0.00	0.00	0.00	-	-	-	-
	AC	0.40	0.30	0.25	0.23	0.20	0.17	0.15	0.15	-	-	-	-

CMC data sets, AMOSHO_CFS produces best classification accuracy followed by FeaClusMOO, FeaClusMOO_EUC, VGAPS, and KM.

Table 6.4: p -values obtained form Wilcoxon's rank sum test by comparing AMOSHO_CFS with other competitor algorithms.

Data sets	KM	VGAPS	FeaClusMOO_EUC	FeaClusMOO
Iris	3.54E-05	1.03E-04	2.70E-02	2.43E-02
Wine	6.04E-01	5.71E-01	2.43E-01	3.17E-01
Glass	1.57E-01	1.20E-03	4.71E-04	1.42E-05
Haberman	2.51E-04	4.80E-04	7.20E-02	2.80E-03
Bupa	1.20E-04	8.42E-02	1.23E-04	1.67E-05
Cancer	1.18E-04	1.13E-05	1.35E-03	7.50E-02
Vowel	1.45E-03	8.77E-02	6.44E-02	1.60E-03
CMC	1.80E-04	1.19E-03	2.20E-03	1.30E-02

6.3.6 Statistical Testing

The comparison of algorithms does not guarantee the efficacy and superiority of the proposed algorithm. The possibility of getting good results, by chance, can not be ignored. For this, Wilcoxon's rank sum test is performed at 5% level of significance and the p -values are tabulated in Table 6.4. The results show that the

classification accuracy obtained from the proposed AMOSHO_CFS is statistically significant as compared to other algorithms.

6.4 AMOSHO_CFS for Real-life Applications

The proposed AMOSHO_CFS is validated and tested on two real-life problems such as image segmentation and microarray data analysis.

6.4.1 Application to Image Segmentation

The proposed algorithm is tested on four color images namely Mandrill, Airplane (F-16), Peppers, and House. The size of each image is 512×512 . The number of data points for each image is 262144. The original images and clustered images are shown in Fig. 6.5. For above-mentioned images, the optimal range of clusters is given in Ref.^[164]. Table 6.6 shows the results obtained from the proposed and other clustering techniques on color images. For Mandrill and Airplane (F-16) images, AMOSHO_CFS generates 6 clusters. AMOSHO_CFS gives 7 and 5 partitions in case of Peppers and House images, respectively. Table 6.7 shows the significance of AMOSHO_CFS using unpaired t -test. These results show the supremacy of proposed AMOSHO_CFS over the other approaches.

6.4.2 Microarray Data Analysis

The proposed technique is also tested on microarray data sets. The three well-known data sets are employed for experimentation. These data sets are Yeast Sporulation^[181], Human Fibroblasts Serum^[182], and Rat CNS^[183]. The results are tested and compared with AMOSHO_CFS, KM, VGAPS, FeaClusMOO_EUC, and FeaClusMOO algorithms. The parameter settings of above-mentioned algorithms are set same as are in their original papers.

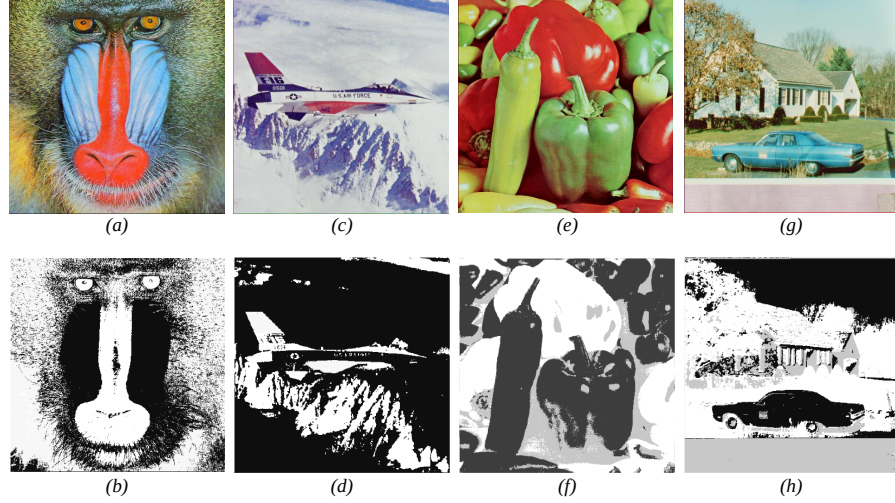


Fig. 6.5. (a) Actual Mandrill image; (b) Clustered image by AMOSHO_CFS; (c) Actual Airplane (F-16) image; (d) Clustered image by AMOSHO_CFS; (e) Actual Peppers image; (f) Clustered image by AMOSHO_CFS; (g) Actual House image; (h) Clustered image by AMOSHO_CFS.

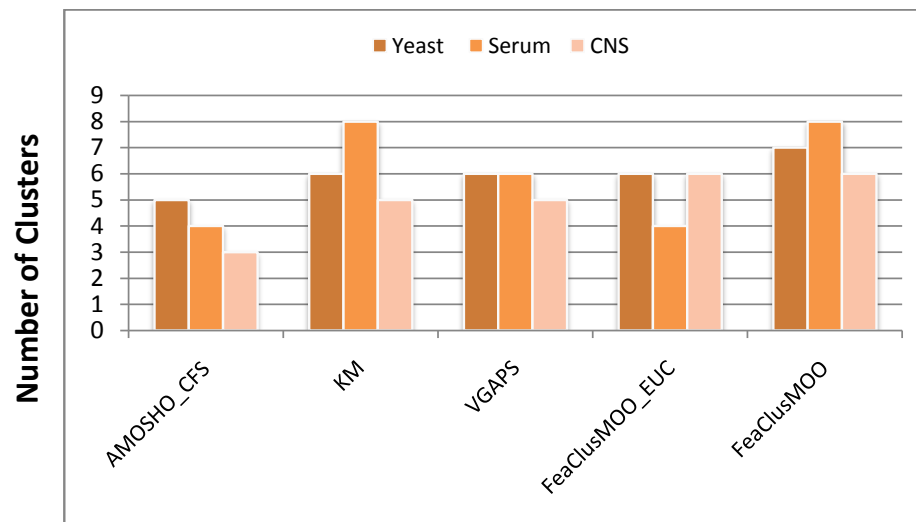
Silhouette index (SC)^[184] is used to measure the separation and compactness of clusters. The higher value of SC means the better performance of clustering algorithm. However, all the clustering algorithms have been run 10 times independently.

Fig. 6.6 shows the number of clusters and average values of Silhouette index obtained by the proposed and competitive algorithms. The results reveal that AMOSHO_CFS has generated 5 clusters for Yeast Sporulation data set. KM, VGAPS, and FeaClusMOO_EUC yielded 6 clusters. The number of clusters generated by FeaClusMOO is 7. The proposed AMOSHO_CFS produces a better value of SC as 0.6985.

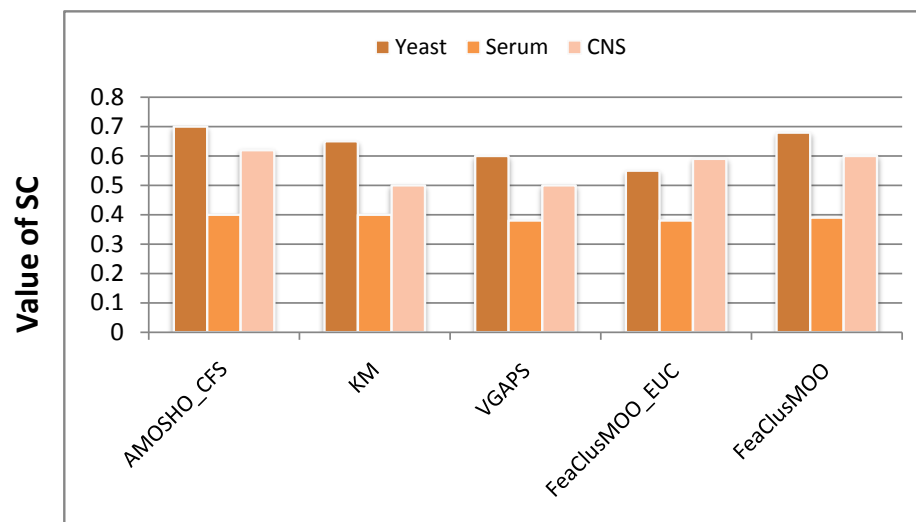
For Human Fibroblasts Serum data set, the proposed AMOSHO_CFS and FeaClusMOO_EUC produced the number of clusters as 4. KM and FeaClusMOO generated 8 clusters. VGAPS produced the number of clusters as 6. The value of SC obtained from AMOSHO_CFS is 0.4003.

For Rat CNS data set, the proposed AMOSHO_CFS generated the number of clusters as 3. KM and VGAPS produced the number of clusters as 5. The number of

clusters obtained from both FeaClusMOO_EUC and FeaClusMOO is 6. The value of SC produced by proposed AMOSHO_CFS is 0.6254. The results reveal that AMOSHO_CFS algorithm outperforms than the other competitor approaches in terms of the number of clusters and SC.



(a)



(b)

Fig. 6.6. Comparison of proposed AMOSHO_CFS approach with competitor approaches in terms of: (a) Number of clusters; (b) SC values.

Table 6.5: The obtained average and standard deviation of cluster quality measures using the proposed and other competitive algorithms.

Data sets	Methods	No. of clusters	No. of features	Classification accuracy
Iris	AMOSHO_CFS	3.0 (0.0)	2.0 (0.0)	96.5 (0.4)
	KM	3.0 (0.0)	2.7 (0.9)	94.8 (2.4)
	VGAPS	3.0 (0.0)	2.2 (0.3)	95.4 (1.9)
	FeaClusMOO_EUC	3.0 (0.0)	1.9 (0.2)	95.9 (1.5)
	FeaClusMOO	3.0 (0.0)	2.0 (0.3)	96.0 (1.6)
Wine	AMOSHO_CFS	3.05 (0.6)	3.8 (1.1)	66.4 (4.1)
	KM	3.8 (2.3)	6.2 (2.6)	64.7 (9.1)
	VGAPS	3.2 (2.0)	5.7 (2.9)	65.0 (8.7)
	FeaClusMOO_EUC	3.5 (1.1)	5.8 (1.5)	65.6 (8.9)
	FeaClusMOO	3.4 (2.5)	6.0 (3.1)	65.2 (9.8)
Glass	AMOSHO_CFS	5.9 (0.8)	3.8 (0.8)	52.2 (3.2)
	KM	6.2 (1.9)	4.3 (1.6)	44.3 (4.7)
	VGAPS	6.0 (1.2)	4.0 (0.6)	44.7 (5.1)
	FeaClusMOO_EUC	5.9 (1.5)	3.5 (1.1)	43.1 (5.4)
	FeaClusMOO	5.7 (2.0)	3.0 (1.9)	46.4 (4.9)
Haberman	AMOSHO_CFS	2.3 (0.4)	1.0 (0.0)	62.5 (9.7)
	KM	2.9 (1.7)	2.0 (0.0)	56.7 (6.4)
	VGAPS	2.2 (0.6)	1.9 (0.6)	56.3 (5.8)
	FeaClusMOO_EUC	2.5 (0.5)	1.3 (0.5)	55.8 (8.1)
	FeaClusMOO	2.7 (1.2)	1.7 (0.9)	56.9 (7.5)
Bupa	AMOSHO_CFS	2.2 (0.3)	2.2 (0.4)	54.7 (2.2)
	KM	2.9 (1.2)	2.5 (0.4)	52.4 (2.4)
	VGAPS	2.6 (1.1)	3.0 (0.5)	51.7 (3.1)
	FeaClusMOO_EUC	3.0 (0.7)	2.4 (0.8)	51.5 (3.9)
	FeaClusMOO	2.6 (1.9)	1.9 (1.2)	52.1 (2.5)
Cancer	AMOSHO_CFS	2.0 (0.0)	3.8 (0.9)	95.2 (0.3)
	KM	2.4 (1.6)	4.3 (1.5)	94.3 (2.1)
	VGAPS	2.4 (1.3)	3.9 (2.0)	93.9 (2.6)
	FeaClusMOO_EUC	2.2 (0.4)	4.1 (1.4)	94.6 (2.9)
	FeaClusMOO	2.0 (1.1)	4.0 (1.8)	94.8 (1.2)
Vowel	AMOSHO_CFS	6.0 (0.6)	1.1 (0.3)	55.5 (2.2)
	KM	6.2 (0.9)	2.0 (0.0)	53.9 (4.5)
	VGAPS	6.3 (1.6)	2.9 (0.7)	52.8 (3.9)
	FeaClusMOO_EUC	6.0 (1.4)	1.1 (0.3)	53.4 (3.6)
	FeaClusMOO	6.0 (2.0)	1.8 (0.6)	54.1 (4.1)
CMC	AMOSHO_CFS	3.0 (0.5)	2.2 (0.4)	43.0 (3.1)
	KM	3.5 (1.1)	3.9 (1.2)	41.8 (3.8)
	VGAPS	3.2 (1.9)	3.8 (1.5)	41.5 (4.3)
	FeaClusMOO_EUC	2.9 (0.8)	3.6 (1.4)	40.4 (3.1)
	FeaClusMOO	3.2 (1.4)	3.0 (1.0)	41.9 (4.2)

Table 6.6: Average and standard deviation of clustering results obtained from different clustering algorithms on four color images over 40 independent runs.

Image	Cluster range	AMOSHO_CFS	KM	VGAPS	FeaClusMOO_EUC	FeaClusMOO
Mandrill	5-10	6.000(0.696)	4.100(0.201)	4.001(0.126)	4.524(0.400)	3.000(0.000)
Airplane (F-16)	5-10	6.010(0.000)	4.017(0.750)	3.200(0.075)	6.310(0.090)	3.498(0.055)
Peppers	5-10	7.007(0.000)	6.007(0.028)	6.810(0.060)	3.152(0.366)	5.007(0.060)
House	5-10	5.017(1.203)	5.285(0.070)	4.657(0.674)	7.400(0.041)	5.225(0.013)

Table 6.7: Unpaired t -test between the best outperforming and second best outperforming algorithms for color images.

Image	Standard error	t	95% confidence interval	Two-tailed P	Significance
Mandrill	0.095	2.5800	-0.44586 to -0.05211	0.0121	Significant
Airplane (F-16)	0.196	5.4102	-1.51003 to -0.67569	<0.0001	Significant
Peppers	0.078	2.5204	-0.44125 to -0.05115	0.0112	Significant
House	0.230	1.7526	-0.86160 to 0.06160	0.0864	Not significant

6.5 Summary

This chapter presents a novel astrophysics based multi-objective approach for automatic finding the number of clusters and features. A variable real-coded agent is used to encode both number of features and clusters. The proposed approach uses novel concept of threshold setting and cutoff for determining the optimal number of features and clusters. A novel fitness function is designed for efficient searching. The proposed technique is validated on real-life data sets. The performance of proposed AMOSHO_CFS is also tested on two real-life applications to demonstrate its effectiveness.

Conclusions and Future Scope

“Everything should be made as simple as possible, but not simpler.”

By Albert Einstein

The intent of this research has been on clustering and in particular the determination of cluster number and its features automatically using multi-objective metaheuristic approach. Two new optimization algorithms are proposed for solving real-life engineering design problems. These are named as Spotted Hyena Optimizer (SHO) and Emperor Penguin Optimizer (EPO). The fundamental concepts which motivate the SHO algorithm are inspired by social hierarchy and hunting behavior of spotted hyenas. The SHO algorithm is tested on twenty-nine benchmark test functions to analyze the exploration, exploitation, local optima avoidance, and convergence behavior. Moreover, five real-life engineering design problems are employed to further examine the efficiency of SHO algorithm. The results demonstrate that SHO provides extremely competitive results as compared to other well-known metaheuristics. Similarly, EPO algorithm is inspired by the huddling behavior of emperor penguins in nature. The proposed EPO algorithm has been tested on forty-four benchmark test functions. Moreover, five real-life engineering design problems are employed to further determine the efficiency of proposed algorithm. Despite, the computational complexity has also been analyzed in terms of time and space complexity. The results reveal that EPO provides very competitive results as compared with other well-known metaheuristics.

Furthermore, an extension of Spotted Hyena Optimizer (SHO) called Multi-objective Spotted Hyena Optimizer (MOSHO) is proposed to solve multi-objective optimization problems. The two components are introduced to solve multi-objective problem. The first component is an archive for retrieving and storing the best non-dominated solutions which are obtained so far and the second component is a group selection mechanism of nearby optimal solutions from the archive. MOSHO has been tested on twenty-four well-known benchmark test functions. The proposed method is applied to five real-life multi-objective optimization problems. The results demonstrate that the proposed method is able to handle many constraints and offers better solutions than other optimizers.

The automatic data clustering is employed for searching an optimal number of clusters so that the obtained partitions get optimized. For this, a novel automatic data clustering approaches called ACMOSHO has been proposed which is based on Multi-objective Spotted Hyena Optimizer (MOSHO). The proposed approach employs the two novel concepts, i.e., weighted cluster-centroid computation and dynamic threshold setting. The experimental outcome reveals that ACMOSHO outperformed its competitors in terms of the number of clusters. Furthermore, the clusters generated by ACMOSHO are well separated and more compact on the basis of intra-cluster as well as inter-cluster distances. In addition, ACMOSHO is utilized for automatic segmentation of coloured images and its efficacy is compared with other prevailing algorithms.

A novel approach named AFSMOSHO to automatically select the features in an unsupervised manner. The proposed approach is based on MOSHO algorithm and selects the optimal number of features of the data set. In addition, a novel threshold setting concept based on the statistical property of data set is developed to further refine the features. AFSMOSHO has been evaluated on twelve real-life data sets with diversity in dimensionality. The high-dimensional data sets resulted in 43% average reduction in features whereas the medium data sets achieved 41% average reduction in features. Further to that, the average reduction in features

for low-dimensional data sets is 44%. The experimental results indicate that the proposed algorithm achieved superior performance than other approaches in terms of class separability and representational entropy.

A novel approach based on astrophysics concepts named AMOSHO_CFS employing Multi-objective Spotted Hyena Optimizer (MOSHO), is proposed for automatic clustering and feature selection. In this approach, a composite agent representation mechanism is utilized for encoding both the number of features having varying number of clusters as well as cluster centers. A novel threshold setting concept is proposed to obtain the optimal number of features and clusters at run time without requiring any a priori knowledge regarding the number of clusters. In addition, a new clustering criterion is projected to make the search more efficient. The performance of proposed AMOSHO_CFS is also tested on two real-life applications to demonstrate its effectiveness. The experimental results revealed that AMOSHO_CFS is capable of generating the optimal number of features as well as clusters for majority of the data sets.

The work mentioned in this thesis can be extended in the following directions:

- The proposed emperor penguin optimizer can be extended for multi-objective optimization problems.
- The proposed emperor penguin optimizer can be applied to clustering and feature selection problems.
- Other approaches can be explored for threshold setting and cutoff for automatic clustering and feature selection.
- The symmetry based distance measures can be incorporated in automatic clustering and feature selection techniques.
- The concept of information theory can be incorporated in the proposed multi-objective based automatic clustering and feature selection approaches.
- The proposed approaches can also be applied to electric power and carbon price forecasting problems.

AppendixA

Single-objective Benchmark Test Functions

This appendix provides the details of single-objective benchmark test functions which are used in this thesis.

A.1 Unimodal Benchmark Test Functions

The detail description of seven well-known unimodal benchmark test functions ($F_1 - F_7$) is mentioned in Table A.1.

Table A.1: Unimodal benchmark test functions.

Function name	Mathematical formulation	Dim	Range	f_{min}
Sphere Model	$F_1(\vec{z}) = \sum_{i=1}^N z_i^2$	30	[-100, 100]	0
Schwefel's Problem 2.22	$F_2(\vec{z}) = \sum_{i=1}^N z_i + \prod_{i=1}^N z_i $	30	[-10, 10]	0
Schwefel's Problem 1.2	$F_3(\vec{z}) = \sum_{i=1}^N (\sum_{j=1}^i z_j)^2$	30	[-100, 100]	0
Schwefel's Problem 2.21	$F_4(\vec{z}) = \max_i \{ z_i , 1 \leq i \leq N\}$	30	[-100, 100]	0
Generalized Rosenbrock's Function	$F_5(\vec{z}) = \sum_{i=1}^{N-1} [100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2]$	30	[-30, 30]	0
Step Function	$F_6(\vec{z}) = \sum_{i=1}^N (z_i + 0.5)^2$	30	[-100, 100]	0
Quartic Function	$F_7(\vec{z}) = \sum_{i=1}^N iz_i^4 + \text{random}[0, 1]$	30	[-1.28, 1.28]	0

A.2 Multimodal Benchmark Test Functions

The detail description of six well-known multimodal benchmark test functions ($F_8 - F_{13}$) is mentioned in Table A.2.

Table A.2: Multimodal benchmark test functions.

Function name	Mathematical formulation	Dim	Range	f_{min}
Generalized Schwefel's Problem 2.26	$F_8(\vec{z}) = \sum_{i=1}^N -z_i \sin(\sqrt{ z_i })$	30	[-500, 500]	-12569.5
Generalized Rastrigin's Function	$F_9(\vec{z}) = \sum_{i=1}^N [z_i^2 - 10 \cos(2\pi z_i) + 10]$	30	[-5.12, 5.12]	0
Ackley's Function	$F_{10}(\vec{z}) = -20 \exp \left(-0.2 \sqrt{\frac{1}{N} \sum_{i=1}^N z_i^2} \right) - \exp \left(\frac{1}{N} \sum_{i=1}^N \cos(2\pi z_i) \right) + 20 + e$	30	[-32, 32]	0
Generalized Griewank Function	$F_{11}(\vec{z}) = \frac{1}{4000} \sum_{i=1}^N z_i^2 - \prod_{i=1}^N \cos \left(\frac{z_i}{\sqrt{i}} \right) + 1$	30	[-600, 600]	0
Generalized Penalized Functions	$F_{12}(\vec{z}) = \frac{\pi}{N} \{ 10 \sin(\pi x_1) + \sum_{i=1}^{N-1} (x_i - 1)^2 [1 + 10 \sin^2(\pi x_{i+1})] + (x_N - 1)^2 \} + \sum_{i=1}^N u(z_i, 10, 100, 4)$ $x_i = 1 + \frac{z_i + 1}{4}$ $u(z_i, a, k, m) = \begin{cases} k(z_i - a)^m & z_i > a \\ 0 & -a < z_i < a \\ k(-z_i - a)^m & z_i < -a \end{cases}$	30	[-50, 50]	0
	$F_{13}(\vec{z}) = 0.1 \{ \sin^2(3\pi z_1) + \sum_{i=1}^N (z_i - 1)^2 [1 + \sin^2(3\pi z_i + 1)] + (z_N - 1)^2 [1 + \sin^2(2\pi z_N)] \} + \sum_{i=1}^N u(z_i, 5, 100, 4)$	30	[-50, 50]	0

A.3 Fixed-dimension Multimodal Benchmark Test Functions

The detail description of ten well-known fixed-dimension multimodal benchmark test functions ($F_{14} - F_{23}$) is mentioned in Table A.3.

A.4 Composite Benchmark Test Functions

The detail description of six well-known composite benchmark test functions ($F_{24} - F_{29}$) is mentioned in Table A.4.

A.5 IEEE CEC-2015 Benchmark Test Functions

The detail description of fifteen well-known composite benchmark test functions ($CEC - 1 - CEC15$) is mentioned in Table A.5.

Table A.3: Fixed-dimension multimodal benchmark test functions.

Function name	Mathematical formulation	Dim	Range	f_{min}
Shekel's Foxholes Function	$F_{14}(\vec{z}) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (z_i - a_{ij})^6} \right)^{-1}$	2	[-65, 65]	1
Kowalik's Function	$F_{15}(\vec{z}) = \sum_{i=1}^{11} \left[a_i - \frac{z_1(b_i^2 + b_i z_2)}{b_i^2 + b_i z_3 + z_4} \right]^2$	4	[-5, 5]	0.00030
Six-Hump Camel-Back Function	$F_{16}(\vec{z}) = 4z_1^2 - 2.1z_1^4 + \frac{1}{3}z_1^6 + z_1z_2 - 4z_2^2 + 4z_2^4$	2	[-5, 5]	-1.0316
Branin Function	$F_{17}(\vec{z}) = \left(z_2 - \frac{5.1}{4\pi^2} z_1^2 + \frac{5}{\pi} z_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos z_1 + 10$	2	[-5, 5]	0.398
Goldstein-Price Function	$F_{18}(\vec{z}) = [1 + (z_1 + z_2 + 1)^2(19 - 14z_1 + 3z_1^2 - 14z_2 + 6z_1z_2 + 3z_2^2)] \times [30 + (2z_1 - 3z_2)^2 \times (18 - 32z_1 + 12z_1^2 + 48z_2 - 36z_1z_2 + 27z_2^2)]$	2	[-2, 2]	3
Hartman's Family	$F_{19}(\vec{z}) = -\sum_{i=1}^4 c_i \exp(-\sum_{j=1}^3 a_{ij}(z_j - p_{ij})^2)$	3	[1, 3]	-3.86
	$F_{20}(\vec{z}) = -\sum_{i=1}^4 c_i \exp(-\sum_{j=1}^6 a_{ij}(z_j - p_{ij})^2)$	6	[0, 1]	-3.32
Shekel's Foxholes Function	$F_{21}(\vec{z}) = -\sum_{i=1}^5 [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.1532
	$F_{22}(\vec{z}) = -\sum_{i=1}^7 [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.4028
	$F_{23}(z) = -\sum_{i=1}^1 0[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.536

Table A.4: Composite benchmark test functions.

Functions	Dim	Range	f_{min}
$F_{24}(CF1)$: $f_1, f_2, f_3, \dots, f_{10} = \text{Sphere Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [1, 1, 1, \dots, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] = [5/100, 5/100, 5/100, \dots, 5/100]$	10	[-5, 5]	0
$F_{25}(CF2)$: $f_1, f_2, f_3, \dots, f_{10} = \text{Griewank's Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [1, 1, 1, \dots, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] = [5/100, 5/100, 5/100, \dots, 5/100]$	10	[-5, 5]	0
$F_{26}(CF3)$: $f_1, f_2, f_3, \dots, f_{10} = \text{Griewank's Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [1, 1, 1, \dots, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] = [1, 1, 1, \dots, 1]$	10	[-5, 5]	0
$F_{27}(CF4)$: $f_1, f_2 = \text{Ackley's Function}$ $f_3, f_4 = \text{Rastrigin's Function}$ $f_5, f_6 = \text{Weierstras's Function}$ $f_7, f_8 = \text{Griewank's Function}$ $f_9, f_{10} = \text{Sphere Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [1, 1, 1, \dots, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] = [5/32, 5/32, 1, 1, 5/0.5, 5/0.5, 5/100, 5/100, 5/100, 5/100]$	10	[-5, 5]	0
$F_{28}(CF5)$: $f_1, f_2 = \text{Rastrigin's Function}$ $f_3, f_4 = \text{Weierstras's Function}$ $f_5, f_6 = \text{Griewank's Function}$ $f_7, f_8 = \text{Ackley's Function}$ $f_9, f_{10} = \text{Sphere Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [1, 1, 1, \dots, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] = [1/5, 1/5, 5/0.5, 5/0.5, 5/100, 5/100, 5/32, 5/32, 5/100, 5/100]$	10	[-5, 5]	0
$F_{29}(CF6)$: $f_1, f_2 = \text{Rastrigin's Function}$ $f_3, f_4 = \text{Weierstras's Function}$ $f_5, f_6 = \text{Griewank's Function}$ $f_7, f_8 = \text{Ackley's Function}$ $f_9, f_{10} = \text{Sphere Function}$ $[\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{10}] = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$ $[\beta_1, \beta_2, \beta_3, \dots, \beta_{10}] =$ $[0.1 * 1/5, 0.2 * 1/5, 0.3 * 5/0.5, 0.4 * 5/0.5, 0.5 * 5/100, 0.6 * 5/100, 0.7 * 5/32, 0.8 * 5/32, 0.9 * 5/100, 1 * 5/100]$	10	[-5, 5]	0

Table A.5: IEEE CEC-2015 benchmark test functions.

No.	Functions	Related basic functions	Dim	f_{min}
<i>CEC</i> – 1	Rotated Bent Cigar Function	Bent Cigar Function	30	100
<i>CEC</i> – 2	Rotated Discus Function	Discus Function	30	200
<i>CEC</i> – 3	Shifted and Rotated Weierstrass Function	Weierstrass Function	30	300
<i>CEC</i> – 4	Shifted and Rotated Schwefel's Function	Schwefel's Function	30	400
<i>CEC</i> – 5	Shifted and Rotated Katsuura Function	Katsuura Function	30	500
<i>CEC</i> – 6	Shifted and Rotated HappyCat Function	HappyCat Function	30	600
<i>CEC</i> – 7	Shifted and Rotated HGBat Function	HGBat Function	30	700
<i>CEC</i> – 8	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	Griewank's Function	30	800
		Rosenbrock's Function		
<i>CEC</i> – 9	Shifted and Rotated Expanded Scaffer's F6 Function	Expanded Scaffer's F6 Function	30	900
<i>CEC</i> – 10	Hybrid Function 1 ($N = 3$)	Schwefel's Function	30	1000
		Rastrigin's Function		
		High Conditioned Elliptic Function		
<i>CEC</i> – 11	Hybrid Function 2 ($N = 4$)	Griewank's Function	30	1100
		Weierstrass Function		
		Rosenbrock's Function		
		Scaffer's F6 Function		
<i>CEC</i> – 12	Hybrid Function 3 ($N = 5$)	Katsuura Function	30	1200
		HappyCat Function		
		Expanded Griewank's plus Rosenbrock's Function		
		Schwefel's Function		
		Ackley's Function		
<i>CEC</i> – 13	Composition Function 1 ($N = 5$)	Rosenbrock's Function	30	1300
		High Conditioned Elliptic Function		
		Bent Cigar Function		
		Discus Function		
<i>CEC</i> – 14	Composition Function 2 ($N = 3$)	High Conditioned Elliptic Function	30	1400
		Schwefel's Function		
		Rastrigin's Function		
<i>CEC</i> – 15	Composition Function 3 ($N = 5$)	High Conditioned Elliptic Function	30	1500
		HGBat Function		
		Rastrigin's Function		
		Schwefel's Function		
		Weierstrass Function		
		High Conditioned Elliptic Function		

AppendixB

Multi-objective Benchmark Test Functions

The multi-objective benchmark test functions are divided into three categories: ZDT^[153], DTLZ^[154], and IEEE CEC-2009 special session^[152]. The detail description of these benchmark test functions is described below.

B.1 ZDT Benchmark Test Functions

B.1.1 ZDT1

Minimize: $f_1(\vec{z}) = z_1$

Minimize: $f_2(\vec{z}) = g(\vec{z}) \times h(f_1(\vec{z}), g(\vec{z}))$

where,

$$g(\vec{z}) = 1 + \frac{9}{N-1} \sum_{i=2}^N z_i$$
$$h(f_1(\vec{z}), g(\vec{z})) = 1 - \sqrt{\frac{f_1(\vec{z})}{g(\vec{z})}}$$
$$0 \leq z_i \leq 1, 1 \leq i \leq 30$$

B.1.2 ZDT2

Minimize: $f_1(\vec{z}) = z_1$

Minimize: $f_2(\vec{z}) = g(\vec{z}) \times h(f_1(\vec{z}), g(\vec{z}))$

where,

$$\begin{aligned}
g(\vec{z}) &= 1 + \frac{9}{N-1} \sum_{i=2}^N z_i \\
h(f_1(\vec{z}), g(\vec{z})) &= 1 - \left(\frac{f_1(\vec{z})}{g(\vec{z})} \right)^2 \\
0 \leq z_i &\leq 1, 1 \leq i \leq 30
\end{aligned}$$

B.1.3 ZDT3

Minimize: $f_1(\vec{z}) = z_1$

Minimize: $f_2(\vec{z}) = g(\vec{z}) \times h(f_1(\vec{z}), g(\vec{z}))$

where,

$$\begin{aligned}
g(\vec{z}) &= 1 + \frac{9}{29} \sum_{i=2}^N z_i \\
h(f_1(\vec{z}), g(\vec{z})) &= 1 - \sqrt{\frac{f_1(\vec{z})}{g(\vec{z})}} \\
&\quad - \left(\frac{f_1(\vec{z})}{g(\vec{z})} \right) \sin(10\pi f_1(\vec{z})) \\
0 \leq z_i &\leq 1, 1 \leq i \leq 30
\end{aligned}$$

B.1.4 ZDT4

Minimize: $f_1(\vec{z}) = z_1$

Minimize: $f_2(\vec{z}) = g(\vec{z}) \times [1 - (z_1/g(\vec{z}))^2]$

where,

$$\begin{aligned}
g(\vec{z}) &= 1 + 10(n-1) + \sum_{i=2}^n (z_i^2 - 10 \cos(4\pi z_i)) \\
0 \leq z_1 &\leq 1, -5 \leq z_i \leq 5, i = 2, \dots, n
\end{aligned}$$

B.1.5 ZDT6

Minimize: $f_1(\vec{z}) = 1 - e^{-4z_1} \times \sin^6(6\pi z_1)$

Minimize: $f_2(\vec{z}) = 1 - \left(\frac{f_1(\vec{z})}{g(\vec{z})} \right)^2$

where,

$$\begin{aligned}
g(\vec{z}) &= 1 + 9 \left[\frac{\left(\sum_{i=2}^n z_i \right)}{(n-1)} \right]^{0.25} \\
0 \leq z_i &\leq 1, i = 1, 2, \dots, n
\end{aligned}$$

B.2 DTLZ Benchmark Test Functions

B.2.1 DTLZ1

$$\text{Minimize: } f_1(\vec{z}) = \frac{1}{2}z_1(1 + g(\vec{z}))$$

$$\text{Minimize: } f_2(\vec{z}) = \frac{1}{2}(1 - z_1)(1 + g(\vec{z}))$$

where,

$$g(\vec{z}) = 100 \left[|\vec{z}| + \sum_{z_i \in \vec{z}} (z_i - 0.5)^2 - \cos(20\pi(z_i - 0.5)) \right]$$
$$0 \leq z_i \leq 1, i = 1, 2, \dots, n$$

B.2.2 DTLZ2

$$\text{Minimize: } f_1(\vec{z}) = (1 + g(\vec{z}))\cos\left(z_1 \frac{\pi}{2}\right)$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z}))\sin\left(z_1 \frac{\pi}{2}\right)$$

where,

$$g(\vec{z}) = \sum_{z_i \in \vec{z}} (z_i - 0.5)^2$$
$$0 \leq z_i \leq 1, i = 1, 2, \dots, n$$

B.2.3 DTLZ3

$$\text{Minimize: } f_1(\vec{z}) = (1 + g(\vec{z}))\cos\left(z_1 \frac{\pi}{2}\right)$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z}))\sin\left(z_1 \frac{\pi}{2}\right)$$

where,

$$g(\vec{z}) = 100 \left[|\vec{z}| + \sum_{z_i \in \vec{z}} (z_i - 0.5)^2 - \cos(20\pi(z_i - 0.5)) \right]$$
$$0 \leq z_i \leq 1, i = 1, 2, \dots, n$$

B.2.4 DTLZ4

$$\text{Minimize: } f_1(\vec{z}) = (1 + g(\vec{z}))\cos\left(z_1^\alpha \frac{\pi}{2}\right)$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z}))\sin\left(z_1^\alpha \frac{\pi}{2}\right)$$

where,

$$g(\vec{z}) = \sum_{z_i \in \vec{z}} (z_i - 0.5)^2$$

$$\alpha = 100$$

$$0 \leq z_i \leq 1, i = 1, 2, \dots, n$$

B.2.5 DTLZ5

$$\text{Minimize: } f_1(\vec{z}) = (1 + g(\vec{z})) \cos\left(\frac{1 + 2g(\vec{z})z_1}{4(1 + g(\vec{z}))} \times \frac{\pi}{2}\right)$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z})) \sin\left(\frac{1 + 2g(\vec{z})z_1}{4(1 + g(\vec{z}))} \times \frac{\pi}{2}\right)$$

where,

$$g(\vec{z}) = \sum_{z_i \in \vec{z}} (z_i - 0.5)^2$$

$$0 \leq z_i \leq 1, i = 2, 3, \dots, n$$

B.2.6 DTLZ6

$$\text{Minimize: } f_1(\vec{z}) = (1 + g(\vec{z})) \cos\left(\frac{1 + 2g(\vec{z})z_1}{4(1 + g(\vec{z}))} \times \frac{\pi}{2}\right)$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z})) \sin\left(\frac{1 + 2g(\vec{z})z_1}{4(1 + g(\vec{z}))} \times \frac{\pi}{2}\right)$$

where,

$$g(\vec{z}) = \sum_{z_i \in \vec{z}} z_i^{0.1}$$

$$0 \leq z_i \leq 1, i = 2, 3, \dots, n$$

B.2.7 DTLZ7

$$\text{Minimize: } f_1(\vec{z}) = z_1$$

$$\text{Minimize: } f_2(\vec{z}) = (1 + g(\vec{z}))h(f_1(\vec{z}), g(\vec{z}))$$

where,

$$g(\vec{z}) = 1 + \frac{9}{|\vec{z}|} \sum_{z_i \in \vec{z}} z_i$$

$$h(f_1(\vec{z}), g(\vec{z})) = M - \frac{f_1(\vec{z})}{1 + g(\vec{z})} (1 + \sin(3\pi f_1(\vec{z})))$$

$$0 \leq z_i \leq 1, 1 \leq i \leq n$$

B.3 IEEE CEC-2009 Benchmark Test Functions

Table B.1: IEEE CEC-2009 benchmark test functions.

Name	Mathematical formulation	Properties
UF1	$f_1 = x_1 + \frac{2}{ J_1 } \sum_{j \in J_1} [x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)]^2$ $f_2 = 1 - \sqrt{x} + \frac{2}{ J_2 } \sum_{j \in J_2} [x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)]^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$	Bi-objective
UF2	$f_1 = x_1 + \frac{2}{ J_1 } \sum_{j \in J_1} y_j^2$ $f_2 = 1 - \sqrt{x} + \frac{2}{ J_2 } \sum_{j \in J_2} y_j^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$	Bi-objective
UF3	$y_j = \begin{cases} x_j - [0.3x_1^2 \cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1] \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & \text{if } j \in J_1 \\ x_j - [0.3x_1^2 \cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1] \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & \text{if } j \in J_2 \end{cases}$ $f_1 = x_1 + \frac{2}{ J_1 } \left(4 \sum_{j \in J_1} y_j^2 - 2 \prod_{j \in J_1} \cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2\right)$ $f_2 = \sqrt{x_1} + \frac{2}{ J_2 } \left(4 \sum_{j \in J_2} y_j^2 - 2 \prod_{j \in J_2} \cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2\right)$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$ $y_j = x_j - x_1^{0.5(1.0 + \frac{3(j-2)}{n-2})}, j = 2, 3, \dots, n$	Bi-objective
UF4	$f_1 = x_1 + \frac{2}{ J_1 } \sum_{j \in J_1} h(y_j)$ $f_2 = 1 - x_2 + \frac{2}{ J_2 } \sum_{j \in J_2} h(y_j)$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$ $y_j = x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), j = 2, 3, \dots, n, \quad h(t) = \frac{ t }{1 + e^{2 t }}$	Bi-objective
UF5	$f_1 = x_1 + \left(\frac{1}{2N} + \epsilon\right) \sin(2N\pi x_1) + \frac{2}{ J_1 } \sum_{j \in J_1} h(y_i)$ $f_2 = 1 - x_1 + \left(\frac{1}{2N} + \epsilon\right) \sin(2N\pi x_1) + \frac{2}{ J_2 } \sum_{j \in J_2} h(y_i)$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$ $\epsilon > 0, \quad y_j = x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), j = 2, 3, \dots, n, \quad h(t) = 2t^2 - \cos(4\pi t) +$	Bi-objective

(Continued...)

Name	Mathematical formulation	Properties
UF6	$f_1 = x_1 + \max\{0, 2\left(\frac{1}{2N} + \epsilon\right)\sin(2N\pi x_1)\} + \frac{2}{ J_1 }\left(4\sum_{j \in J_1} y_j^2 - 2\prod_{j \in J_1} \cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 1\right)$ $f_2 = 1 - x_1 + \max\{0, 2\left(\frac{1}{2N} + \epsilon\right)\sin(2N\pi x_1)\} + \frac{2}{ J_2 }\left(4\sum_{j \in J_2} y_j^2 - 2\prod_{j \in J_2} \cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 1\right)$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$ $\epsilon > 0, \quad y_j = x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), j = 2, 3, \dots, n$	Bi-objective
UF7	$f_1 = \sqrt[5]{x_1} + \frac{2}{ J_1 }\sum_{j \in J_1} y_j^2$ $f_2 = 1 - \sqrt[5]{x_1} + \frac{2}{ J_2 }\sum_{j \in J_2} y_j^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j \mid j \text{ is even and } 2 \leq j \leq n\}$ $\epsilon > 0, \quad y_j = x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), j = 2, 3, \dots, n$	Bi-objective
UF8	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{ J_1 }\sum_{j \in J_1} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{ J_2 }\sum_{j \in J_2} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_3 = \sin(0.5x_1\pi) + \frac{2}{ J_3 }\sum_{j \in J_3} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $J_1 = \{j \mid 3 \leq j \leq n, \text{ and } j-1 \text{ is a multiplication of } 3\}$ $J_2 = \{j \mid 3 \leq j \leq n, \text{ and } j-2 \text{ is a multiplication of } 3\}$ $J_3 = \{j \mid 3 \leq j \leq n, \text{ and } j \text{ is a multiplication of } 3\}$	Tri-objective
UF9	$f_1 = 0.5[\max\{0, (1 + \epsilon)(1 - 4(2x_1 - 1)^2)\} + 2x_1]x_2 + \frac{2}{ J_1 }\sum_{j \in J_1} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_2 = 0.5[\max\{0, (1 + \epsilon)(1 - 4(2x_1 - 1)^2)\} + 2x_1]x_2 + \frac{2}{ J_2 }\sum_{j \in J_2} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $f_3 = 1 - x_2 + \frac{2}{ J_3 }\sum_{j \in J_3} \left(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right)\right)^2$ $J_1 = \{j \mid 3 \leq j \leq n, \text{ and } j-1 \text{ is a multiplication of } 3\}$ $J_2 = \{j \mid 3 \leq j \leq n, \text{ and } j-2 \text{ is a multiplication of } 3\}$ $J_3 = \{j \mid 3 \leq j \leq n, \text{ and } j \text{ is a multiplication of } 3\}, \quad \epsilon = 0.1$	Tri-objective
UF10	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{ J_1 }\sum_{j \in J_1} [4y_j^2 - \cos(8\pi y_j) + 1]$ $f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{ J_2 }\sum_{j \in J_2} [4y_j^2 - \cos(8\pi y_j) + 1]$ $f_3 = \sin(0.5x_1\pi) + \frac{2}{ J_3 }\sum_{j \in J_3} [4y_j^2 - \cos(8\pi y_j) + 1]$ $J_1 = \{j \mid 3 \leq j \leq n, \text{ and } j-1 \text{ is a multiplication of } 3\}$ $J_2 = \{j \mid 3 \leq j \leq n, \text{ and } j-2 \text{ is a multiplication of } 3\}$ $J_3 = \{j \mid 3 \leq j \leq n, \text{ and } j \text{ is a multiplication of } 3\}$	Tri-objective

AppendixC

Engineering Design Problems

This appendix provides the details of single and multi-objective engineering design problems which are used in this thesis.

C.1 Single-objective Engineering Design Problems

The five well-known single-objective engineering design problems are pressure vessel design, speed reducer design, welded beam design, tension/compression spring design, and rolling element bearing design. The detail description of these problems is mentioned below.

C.1.1 Pressure Vessel Design Problem

Consider $\vec{z} = [z_1 \ z_2 \ z_3 \ z_4] = [T_s \ T_h \ R \ L]$,

Minimize $f(\vec{z}) = 0.6224z_1z_3z_4 + 1.7781z_2z_3^2 + 3.1661z_1^2z_4 + 19.84z_1^2z_3$,

Subject to:

$$g_1(\vec{z}) = -z_1 + 0.0193z_3 \leq 0,$$

$$g_2(\vec{z}) = -z_3 + 0.00954z_3 \leq 0, \tag{C.1}$$

$$g_3(\vec{z}) = -\pi z_3^2 z_4 - \frac{4}{3}\pi z_3^3 + 1,296,000 \leq 0,$$

$$g_4(\vec{z}) = z_4 - 240 \leq 0,$$

where,

$$1 \times 0.0625 \leq z_1, \ z_2 \leq 99 \times 0.0625, \ 10.0 \leq z_3, \ z_4 \leq 200.0.$$

C.1.2 Speed Reducer Design Problem

Consider $\vec{z} = [z_1 \ z_2 \ z_3 \ z_4 \ z_5 \ z_6 \ z_7] = [b \ m \ p \ l_1 \ l_2 \ d_1 \ d_2]$,

$$\begin{aligned} \text{Minimize } f(\vec{z}) = & 0.7854z_1z_2^2(3.3333z_3^2 + 14.9334z_3 - 43.0934) \\ & - 1.508z_1(z_6^2 + z_7^2) + 7.4777(z_6^3 + z_7^3) + 0.7854(z_4z_6^2 + z_5z_7^2), \end{aligned} \quad (\text{C.2})$$

Subject to:

$$\begin{aligned} g_1(\vec{z}) &= \frac{27}{z_1z_2^2z_3} - 1 \leq 0, \\ g_2(\vec{z}) &= \frac{397.5}{z_1z_2^2z_3^2} - 1 \leq 0, \\ g_3(\vec{z}) &= \frac{1.93z_4^3}{z_2z_6^4z_3} - 1 \leq 0, \\ g_4(\vec{z}) &= \frac{1.93z_5^3}{z_2z_7^4z_3} - 1 \leq 0, \\ g_5(\vec{z}) &= \frac{[(745(z_4/z_2z_3))^2 + 16.9 \times 10^6]^{1/2}}{110z_6^3} - 1 \leq 0, \\ g_6(\vec{z}) &= \frac{[(745(z_5/z_2z_3))^2 + 157.5 \times 10^6]^{1/2}}{85z_7^3} - 1 \leq 0, \\ g_7(\vec{z}) &= \frac{z_2z_3}{40} - 1 \leq 0, \\ g_8(\vec{z}) &= \frac{5z_2}{z_1} - 1 \leq 0, \\ g_9(\vec{z}) &= \frac{z_1}{12z_2} - 1 \leq 0, \\ g_{10}(\vec{z}) &= \frac{1.5z_6 + 1.9}{z_4} - 1 \leq 0, \\ g_{11}(\vec{z}) &= \frac{1.1z_7 + 1.9}{z_5} - 1 \leq 0, \end{aligned}$$

where,

$$2.6 \leq z_1 \leq 3.6, \ 0.7 \leq z_2 \leq 0.8, \ 17 \leq z_3 \leq 28, \ 7.3 \leq z_4 \leq 8.3,$$

$$7.3 \leq z_5 \leq 8.3, \ 2.9 \leq z_6 \leq 3.9, \ 5.0 \leq z_7 \leq 5.5.$$

C.1.3 Welded Beam Design Problem

Consider $\vec{z} = [z_1 \ z_2 \ z_3 \ z_4] = [h \ l \ t \ b]$,

$$\text{Minimize } f(\vec{z}) = 1.10471z_1^2z_2 + 0.04811z_3z_4(14.0 + z_2), \quad (\text{C.3})$$

Subject to:

$$g_1(\vec{z}) = \tau(\vec{z}) - 13,600 \leq 0,$$

$$g_2(\vec{z}) = \sigma(\vec{z}) - 30,000 \leq 0,$$

$$g_3(\vec{z}) = \delta(\vec{z}) - 0.25 \leq 0,$$

$$g_4(\vec{z}) = z_1 - z_4 \leq 0,$$

$$g_5(\vec{z}) = 6,000 - P_c(\vec{z}) \leq 0,$$

$$g_6(\vec{z}) = 0.125 - z_1 \leq 0,$$

$$g_7(\vec{z}) = 1.10471z_1^2 + 0.04811z_3z_4(14.0 + z_2) - 5.0 \leq 0,$$

where,

$$0.1 \leq z_1, 0.1 \leq z_2, z_3 \leq 10.0, z_4 \leq 2.0,$$

$$\tau(\vec{z}) = \sqrt{(\tau')^2 + (\tau'')^2 + (l\tau'\tau'')}/\sqrt{0.25(l^2 + (h+t)^2)},$$

$$\tau' = \frac{6,000}{\sqrt{2}hl}, \quad \sigma(\vec{z}) = \frac{504,000}{t^2b}, \quad \delta(\vec{z}) = \frac{65,856,000}{(30 \times 10^6)bt^3},$$

$$\tau'' = \frac{6,000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2[0.707hl(l^2/12 + 0.25(h+t)^2)]}, P_c(\vec{z}) = 64,746.022(1 - 0.0282346t)tb^3.$$

C.1.4 Tension/compression Spring Design Problem

Consider $\vec{z} = [z_1 \ z_2 \ z_3] = [d \ D \ P]$,

Minimize $f(\vec{z}) = (z_3 + 2)z_2z_1^2$,

Subject to:

$$g_1(\vec{z}) = 1 - \frac{z_2^3z_3}{71785z_1^4} \leq 0,$$

$$g_2(\vec{z}) = \frac{4z_2^2 - z_1z_2}{12566(z_2z_1^3 - z_1^4)} + \frac{1}{5108z_1^2} \leq 0, \tag{C.4}$$

$$g_3(\vec{z}) = 1 - \frac{140.45z_1}{z_2^2z_3} \leq 0,$$

$$g_4(\vec{z}) = \frac{z_1 + z_2}{1.5} - 1 \leq 0,$$

where,

$$0.05 \leq z_1 \leq 2.0, 0.25 \leq z_2 \leq 1.3, 2.0 \leq z_3 \leq 15.0.$$

C.1.5 Rolling Element Bearing Design Problem

$$\text{Maximize } C_d = \begin{cases} f_c Z^{2/3} D_b^{1.8}, & \text{if } D \prec 25.4mm \\ C_d = 3.647 f_c Z^{2/3} D_b^{1.4}, & \text{if } D \succ 25.4mm \end{cases}$$

Subject to:

$$g_1(\vec{z}) = \frac{\phi_0}{2\sin^{-1}(D_b/D_m)} - Z + 1 \leq 0,$$

$$g_2(\vec{z}) = 2D_b - K_{Dmin}(D - d) \geq 0,$$

$$g_3(\vec{z}) = K_{Dmax}(D - d) - 2D_b \geq 0,$$

$$g_4(\vec{z}) = \zeta B_w - D_b \leq 0,$$

$$g_5(\vec{z}) = D_m - 0.5(D + d) \geq 0,$$

$$g_6(\vec{z}) = (0.5 + e)(D + d) - D_m \geq 0,$$

$$g_7(\vec{z}) = 0.5(D - D_m - D_b) - \varepsilon D_b \geq 0,$$

$$g_8(\vec{z}) = f_i \geq 0.515,$$

$$g_9(\vec{z}) = f_o \geq 0.515,$$

(C.5)

where,

$$x = [\{(D - d)/2 - 3(T/4)\}^2 + \{D/2 - T/4 - D_b\}^2 - \{d/2 + T/4\}^2]$$

$$y = 2\{(D - d)/2 - 3(T/4)\}\{D/2 - T/4 - D_b\}$$

$$f_c = 37.91 \left[1 + \left\{ 1.04 \left(\frac{1 - \gamma}{1 + \gamma} \right)^{1.72} \left(\frac{f_i(2f_o - 1)}{f_o(2f_i - 1)} \right)^{0.41} \right\}^{10/3} \right]^{-0.3}$$

$$\times \left[\frac{\gamma^{0.3}(1 - \gamma)^{1.39}}{(1 + \gamma)^{1/3}} \right] \left[\frac{2f_i}{2f_i - 1} \right]^{0.41}, \quad \phi_o = 2\pi - 2\cos^{-1} \left(\frac{x}{y} \right)$$

$$\gamma = \frac{D_b}{D_m}, \quad f_i = \frac{r_i}{D_b}, \quad f_o = \frac{r_o}{D_b}, \quad T = D - d - 2D_b$$

$$D = 160, \quad d = 90, \quad B_w = 30, \quad r_i = r_o = 11.033$$

$$0.5(D + d) \leq D_m \leq 0.6(D + d), \quad 0.15(D - d) \leq D_b$$

$$\leq 0.45(D - d), \quad 4 \leq Z \leq 50, \quad 0.515 \leq f_i \text{ and } f_o \leq 0.6,$$

$$0.4 \leq K_{Dmin} \leq 0.5, \quad 0.6 \leq K_{Dmax} \leq 0.7, \quad 0.3 \leq e \leq 0.4,$$

$$0.02 \leq e \leq 0.1, \quad 0.6 \leq \zeta \leq 0.85.$$

C.2 Multi-objective Engineering Design Problems

The five well-known multi-objective engineering design problems are welded beam design, multiple-disk clutch brake design, pressure vessel design, speed reducer design, and gear train design. The detail description of these problems is mentioned below.

C.2.1 Welded Beam Design Problem

$$\text{Minimize } f_1(\vec{z}) = C = 1.10471h^2l + 0.04811tb(14.0 + l),$$

$$\text{Minimize } f_2(\vec{z}) = D = \frac{2.1952}{t^3b},$$

Subject to:

$$g_1(\vec{z}) = 13,600 - \tau(\vec{z}) \geq 0,$$

$$g_2(\vec{z}) = 30,000 - \sigma(\vec{z}) \geq 0,$$

$$g_3(\vec{z}) = b - h \geq 0,$$

$$g_4(\vec{z}) = P_c(\vec{z}) - 6,000 \geq 0,$$

Variable range,

(C.6)

$$0.125 \leq h, b \leq 5.0\text{in.},$$

$$0.1 \leq l, t \leq 10.0\text{in.},$$

where,

$$\tau(\vec{z}) = \sqrt{(\tau')^2 + (\tau'')^2 + (l\tau'\tau'')/\sqrt{0.25(l^2 + (h+t)^2)}},$$

$$\tau' = \frac{6,000}{\sqrt{2hl}}, \sigma(\vec{z}) = \frac{504,000}{t^2b},$$

$$\tau'' = \frac{6,000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2[0.707hl(l^2/12 + 0.25(h+t)^2)]},$$

$$P_c(\vec{z}) = 64,746.022(1 - 0.0282346t)tb^3.$$

C.2.2 Multiple-disk Clutch Brake Design Problem

$$\text{Minimize } f_1(\vec{z}) = M = \pi(r_o^2 - r_i^2)t(Z+1)p_m,$$

$$\text{Minimize } f_2(\vec{z}) = T = \frac{I_z w}{M_h + M_f},$$

Subject to:

$$g_1(\vec{z}) = r_o - r_i - \Delta R \geq 0,$$

$$g_2(\vec{z}) = L_{max} - (Z+1)(t+\delta) \geq 0,$$

$$g_3(\vec{z}) = p_{max} - p_{rz} \geq 0,$$

$$g_4(\vec{z}) = p_{max}V_{sr,max} - p_{rz}V_{sr} \geq 0,$$

$$g_5(\vec{z}) = V_{sr,max} - V_{sr} \geq 0,$$

$$g_6(\vec{z}) = M_h - sM_s \geq 0,$$

$$g_7(\vec{z}) = T \geq 0,$$

$$g_8(\vec{z}) = T_{max} - T \geq 0,$$

$$60 \leq r_i \leq 80 \text{ mm},$$

$$90 \leq r_o \leq 110 \text{ mm},$$

$$1.5 \leq t \leq 3 \text{ mm},$$

$$0 \leq F \leq 1000 \text{ N},$$

$$2 \leq Z \leq 9$$

where,

$$p_m = 0.0000078 \text{ kg/mm}^3, p_{max} = 1 \text{ MPa}, \mu = 0.5, V_{sr,max} = 10 \text{ m/s},$$

$$s = 1.5, T_{max} = 15 \text{ s}, n = 250 \text{ rpm}, M_s = 40 \text{ Nm}, M_f = 3 \text{ Nm},$$

$$I_z = 55 \text{ kg-m}^2, \delta = 0.5 \text{ mm}, \Delta R = 20 \text{ mm}, L_{max} = 30 \text{ mm},$$

$$M_h = \frac{2}{3}\mu F Z \frac{r_o^3 - r_i^3}{r_o^2 - r_i^2} \text{ N-mm}, w = \frac{\pi n}{30} \text{ rad/s}, R_{sr} = \frac{2}{3} \frac{r_o^3 - r_i^3}{r_o^2 - r_i^2} \text{ mm},$$

$$A = \pi(r_o^2 - r_i^2) \text{ mm}^2,$$

$$p_{rz} = \frac{F}{A} \text{ N/mm}^2,$$

$$V_{sr} = \frac{\pi R_{sr} n}{30} \text{ mm/s},$$

(C.7)

C.2.3 Pressure Vessel Design Problem

$$\text{Minimize } f_1(\vec{z}) = 0.6224T_sLR + 1.7781T_hR^2 + 3.1661T_s^2L + 19.84T_s^2R,$$

$$\text{Minimize } f_2(\vec{z}) = -(\pi R^2L + 1.333\pi R^3),$$

Subject to:

$$g_1(\vec{z}) = 0.0193R - T_s \leq 0,$$

$$g_2(\vec{z}) = 0.00954R - T_h \leq 0,$$

$$g_3(\vec{z}) = 0.0625 - T_s \leq 0,$$

$$g_4(\vec{z}) = T_s - 5 \leq 0, \tag{C.8}$$

$$g_5(\vec{z}) = 0.0625 - T_h \leq 0,$$

$$g_6(\vec{z}) = T_h - 5 \leq 0,$$

$$g_7(\vec{z}) = 10 - R \leq 0,$$

$$g_8(\vec{z}) = R - 200 \leq 0,$$

$$g_9(\vec{z}) = 10 - L \leq 0,$$

$$g_{10}(\vec{z}) = L - 240 \leq 0,$$

C.2.4 Speed Reducer Design Problem

$$\text{Minimize } f_1(\vec{z}) = 0.7854z_1z_2^2(3.3333z_3^2 + 14.9334z_3 - 43.0934)$$

$$- 1.508z_1(z_6^2 + z_7^2) + 7.4777(z_6^3 + z_7^3) + 0.7854(z_4z_6^2 + z_5z_7^2),$$

$$\text{Minimize } f_2(\vec{z}) = \frac{\sqrt{\frac{745z_4^2}{z_2z_3} + 1.69 \times 10^7}}{0.1z_6^3}, \tag{C.9}$$

Subject to:

$$g_1(\vec{z}) = \frac{27}{z_1z_2^2z_3} - 1 \leq 0,$$

$$g_2(\vec{z}) = \frac{397.5}{z_1z_2^2z_3^2} - 1 \leq 0,$$

$$\begin{aligned}
g_3(\vec{z}) &= \frac{1.93z_4^3}{z_2z_6^4z_3} - 1 \leq 0, \\
g_4(\vec{z}) &= \frac{1.93z_5^3}{z_2z_7^4z_3} - 1 \leq 0, \\
g_5(\vec{z}) &= \frac{[(745(z_4/z_2z_3))^2 + 16.9 \times 10^6]^{1/2}}{110z_6^3} - 1 \leq 0, \\
g_6(\vec{z}) &= \frac{[(745(z_5/z_2z_3))^2 + 157.5 \times 10^6]^{1/2}}{85z_7^3} - 1 \leq 0, \\
g_7(\vec{z}) &= \frac{z_2z_3}{40} - 1 \leq 0, \\
g_8(\vec{z}) &= \frac{5z_2}{z_1} - 1 \leq 0, \\
g_9(\vec{z}) &= \frac{z_1}{12z_2} - 1 \leq 0, \\
g_{10}(\vec{z}) &= \frac{1.5z_6 + 1.9}{z_4} - 1 \leq 0, \\
g_{11}(\vec{z}) &= \frac{1.1z_7 + 1.9}{z_5} - 1 \leq 0,
\end{aligned}$$

where,

$$2.6 \leq z_1 \leq 3.6, 0.7 \leq z_2 \leq 0.8, 17 \leq z_3 \leq 28, 7.3 \leq z_4 \leq 8.3,$$

$$7.3 \leq z_5 \leq 8.3, 2.9 \leq z_6 \leq 3.9, 5.0 \leq z_7 \leq 5.5$$

C.2.5 Gear Train Design Problem

$$\text{Minimize } f_1(T) = \left| 6.931 - \frac{T_a}{T_d} \frac{T_f}{T_b} \right|$$

$$\text{Minimize } f_2(T) = \max(T_d, T_b, T_a, T_f)$$

Subject to:

$$\frac{f_1(T)}{6.931} \leq 0.5,$$

$$12 \leq T_d, T_b, T_a, T_f \leq 60,$$

T_d, T_b, T_a, T_f are integers

(C.10)

References

- [1] Taha, A. & Hadi, A. S. Pair-wise association measures for categorical and mixed data, *Information Sciences*, **346**, 73–89, 2016.
- [2] Sazonov, E. Clustering [book review], *IEEE Pulse*, **1**(1), 74–76, 2010.
- [3] Everitt, B. S., Landau, S., & Leese, M. *Cluster Analysis*. Wiley Publishing, 4th edition, 2009.
- [4] Xu, R. & Wunsch, D. Survey of clustering algorithms, *IEEE Transactions on neural networks*, **16**(3), 645–678, 2005.
- [5] Tan, P.-N. et al. *Introduction to data mining*. Pearson Education India, 2007.
- [6] Fligner, M. A. & Verducci, J. S. Distance based ranking models, *Journal of the Royal Statistical Society. Series B (Methodological)*, **48**, 359–369, 1986.
- [7] Potolea, R., Cacoveanu, S., & Lemnaru, C. Meta-learning framework for prediction strategy evaluation, in International Conference on Enterprise Information Systems. Springer, 280–295.
- [8] Das, S., Abraham, A., & Konar, A. *Metaheuristic clustering*, volume 178. Springer, 2009.
- [9] Davies, D. L. & Bouldin, D. W. A cluster separation measure, *IEEE transactions on pattern analysis and machine intelligence*, **1**(2), 224–227, 1979.
- [10] Xie, X. L. & Beni, G. A validity measure for fuzzy clustering, *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **13**(8), 841–847, 1991.
- [11] Pakhira, M. K., Bandyopadhyay, S., & Maulik, U. Validity index for crisp and fuzzy clusters, *Pattern recognition*, **37**(3), 487–501, 2004.

- [12] Bandyopadhyay, S. & Saha, S. A point symmetry-based clustering technique for automatic evolution of clusters, *IEEE Transactions on Knowledge and Data Engineering*, **20**(11), 1441–1457, 2008.
- [13] Maulik, U. & Bandyopadhyay, S. Performance evaluation of some clustering algorithms and validity indices, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(12), 1650–1654, 2002.
- [14] Alba, E. & Dorronsoro, B. The exploration/exploitation tradeoff in dynamic cellular genetic algorithms, *IEEE transactions on evolutionary computation*, **9**(2), 126–142, 2005.
- [15] Kala, R., Vazirani, H., Shukla, A., & Tiwari, R. Offline handwriting recognition using genetic algorithm, *International Journal of Computer Science*, **7**, 2010.
- [16] Manshahia, M. S., Dave, M., & Singh, S. Firefly algorithm based clustering technique for wireless sensor networks, in *Wireless Communications, Signal Processing and Networking (WiSPNET)*, International Conference on. IEEE, 1273–1276.
- [17] Manshahia, M. S., Dave, M., & Singh, S. B. Congestion control in wireless sensor networks based on bioluminescent firefly behavior, *Wireless Sensor Network*, **7**(12), 149–156, 2015.
- [18] Yolcu, U., Cagcag, O., Aladag, C. H., & Egrioglu, E. An enhanced fuzzy time series forecasting method based on artificial bee colony, *Journal of Intelligent & Fuzzy Systems*, **26**(6), 2627–2637, 2014.
- [19] Aladag, C. H., Yolcu, U., Egrioglu, E., & Bas, E. Fuzzy lagged variable selection in fuzzy time series with genetic algorithms, *Applied Soft Computing*, **22**, 465–473, 2014.

- [20] Li, H., Guo, S., Zhao, H., Su, C., & Wang, B. Annual electric load forecasting by a least squares support vector machine with a fruit fly optimization algorithm, *Energies*, **5**(11), 4430–4445, 2012.
- [21] Li, H.-Z., Guo, S., Li, C.-J., & Sun, J.-Q. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm, *Knowledge-Based Systems*, **37**, 378–387, 2013.
- [22] Kaur, T., Saini, B. S., & Gupta, S. A joint intensity and edge magnitude-based multilevel thresholding algorithm for the automatic segmentation of pathological mr brain images, *Neural Computing and Applications*, **30**(4), 1317–1340, 2018.
- [23] Lozano, M. & García-Martínez, C. Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report, *Computers & Operations Research*, **37**(3), 481–497, 2010.
- [24] Abrol, P., Gupta, S., & Kaur, K. Analysis of resource management and placement policies using a new nature inspired meta heuristic sscwa avoiding premature convergence in cloud, in *Computational Techniques in Information and Communication Technologies (ICCTICT)*, 2016 International Conference on. IEEE, 653–658.
- [25] Wolpert, D. H. & Macready, W. G. No free lunch theorems for optimization, *IEEE transactions on evolutionary computation*, **1**(1), 67–82, 1997.
- [26] Assad, A. & Deep, K. A hybrid harmony search and simulated annealing algorithm for continuous optimization, *Information Sciences*, **450**, 246–266, 2018.
- [27] Kennedy, J. Particle swarm optimization, in *Encyclopedia of machine learning*, 760–766. Springer, 2011.

- [28] Lalwani, S., Kumar, R., & Deep, K. Multi-objective two-level swarm intelligence approach for multiple rna sequence-structure alignment, *Swarm and evolutionary computation*, **34**, 130–144, 2017.
- [29] Branke, J., Deb, K., Dierolf, H., & Osswald, M. Finding knees in multi-objective optimization, in International conference on parallel problem solving from nature. Springer, 722–731.
- [30] Deb, K. Advances in evolutionary multi-objective optimization, in International Symposium on Search Based Software Engineering. Springer, 1–26.
- [31] Castillo, E., Mínguez, R., Castillo, C., & Hadi, A. S. Improving parameter estimation using constrained optimization methods, *Journal of Statistical Computation and Simulation*, **82**(10), 1471–1499, 2012.
- [32] Marler, R. T. & Arora, J. S. Survey of multi-objective optimization methods for engineering, *Structural and multidisciplinary optimization*, **26**(6), 369–395, 2004.
- [33] Zăvoianu, A.-C., Lughofer, E., Bramerdorfer, G., Amrhein, W., & Klement, E. P. Decmo2: a robust hybrid and adaptive multi-objective evolutionary algorithm, *soft Computing*, **19**(12), 3551–3569, 2015.
- [34] Zăvoianu, A.-C., Bramerdorfer, G., Lughofer, E., Silber, S., Amrhein, W., & Klement, E. P. Hybridization of multi-objective evolutionary algorithms and artificial neural networks for optimizing the performance of electrical drives, *Engineering Applications of Artificial Intelligence*, **26**(8), 1781–1794, 2013.
- [35] Yang, X.-S., Karamanoglu, M., & He, X. Flower pollination algorithm: a novel approach for multiobjective optimization, *Engineering Optimization*, **46**(9), 1222–1237, 2014.

- [36] Yang, X.-S. & Hossein Gandomi, A. Bat algorithm: a novel approach for global engineering optimization, *Engineering Computations*, **29**(5), 464–483, 2012.
- [37] Kala, R., Shukla, A., & Tiwari, R. Fusion of probabilistic a* algorithm and fuzzy inference system for robotic path planning, *Artificial Intelligence Review*, **33**(4), 307–327, 2010.
- [38] Yang, X.-S. *Engineering optimization: an introduction with metaheuristic applications*. John Wiley & Sons, 2010.
- [39] Madavan, N. K. & Biegel, B. A. Multiobjective optimization using a pareto differential evolution approach, in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, 1145–1150.
- [40] Coello, C. C. Evolutionary multi-objective optimization: a historical view of the field, *IEEE computational intelligence magazine*, **1**(1), 28–36, 2006.
- [41] Luh, G.-C. & Chueh, C.-H. Multi-objective optimal design of truss structure with immune algorithm, *Computers & structures*, **82**(11-12), 829–844, 2004.
- [42] Kipouros, T., Jaeggi, D. M., Dawes, W. N., Parks, G. T., Savill, A. M., & Clarkson, P. J. Biobjective design optimization for axial compressors using tabu search, *AIAA journal*, **46**(3), 701–711, 2008.
- [43] Chen, M. & Hammami, O. A system engineering conception of multi-objective optimization for multi-physics system, in *Multiphysics Modelling and Simulation for Systems Design and Monitoring*, 299–306. Springer, 2015.
- [44] Hadi, A. S. Identifying multiple outliers in multivariate data, *Journal of the Royal Statistical Society: Series B (Methodological)*, **54**(3), 761–771, 1992.

- [45] Handl, J., Kell, D. B., & Knowles, J. Multiobjective optimization in bioinformatics and computational biology, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, **4**(2), 279–292, 2007.
- [46] Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A., et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.
- [47] Coello, C. A. C. Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored, *Frontiers of Computer Science in China*, **3**(1), 18–30, 2009.
- [48] Edgeworth, F. Y. *Mathematical psychics: An essay on the application of mathematics to the moral sciences*, volume 10. Kegan Paul, 1881.
- [49] Pareto, V. *Cours d'économie politique*, volume 1. Librairie Droz, 1964.
- [50] Zitzler, E. & Thiele, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach, *IEEE transactions on Evolutionary Computation*, **3**(4), 257–271, 1999.
- [51] Coello, C. A. C., Dhaenens, C., & Jourdan, L. Multi-objective combinatorial optimization: Problematic and context, in *Advances in multi-objective nature inspired computing*, 1–21. Springer, 2010.
- [52] Rudolph, G., Schütze, O., Grimme, C., Domínguez-Medina, C., & Trautmann, H. Optimal averaged hausdorff archives for bi-objective problems: theoretical and numerical results, *Computational Optimization and Applications*, **64**(2), 589–618, 2016.
- [53] Schutze, O., Esquivel, X., Lara, A., & Coello, C. A. C. Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization, *IEEE Transactions on Evolutionary Computation*, **16**(4), 504–522, 2012.

- [54] Schütze, O., Laumanns, M., Tantar, E., Coello, C. A. C., & Talbi, E.-G. Computing gap free pareto front approximations with stochastic search algorithms, *Evolutionary Computation*, **18**(1), 65–96, 2010.
- [55] Li, M. & Zheng, J. Spread assessment for evolutionary multi-objective optimization, in International Conference on Evolutionary Multi-Criterion Optimization. Springer, 216–230.
- [56] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Da Fonseca, V. G. Performance assessment of multiobjective optimizers: An analysis and review, *IEEE Transactions on evolutionary computation*, **7**(2), 117–132, 2003.
- [57] Handl, J. & Knowles, J. An evolutionary approach to multiobjective clustering, *IEEE transactions on Evolutionary Computation*, **11**(1), 56–76, 2007.
- [58] Liu, Y., Özyer, T., Alhajj, R., & Barker, K. Integrating multi-objective genetic algorithm and validity analysis for locating and ranking alternative clustering, *Informatica*, **29**(1), 2005.
- [59] Matake, N., Hiroyasu, T., Miki, M., & Senda, T. Multiobjective clustering with automatic k-determination for large-scale data, in Proceedings of the 9th annual conference on Genetic and evolutionary computation. ACM, 861–868.
- [60] Bandyopadhyay, S., Maulik, U., & Mukhopadhyay, A. Multiobjective genetic clustering for pixel classification in remote sensing imagery, *IEEE transactions on Geoscience and Remote Sensing*, **45**(5), 1506–1511, 2007.
- [61] Suresh, K., Kundu, D., Ghosh, S., Das, S., & Abraham, A. Automatic clustering with multi-objective differential evolution algorithms, in Evolutionary Computation, 2009. CEC’09. IEEE Congress on. IEEE, 2590–2597.

- [62] Kundu, D., Suresh, K., Ghosh, S., Das, S., Abraham, A., & Badr, Y. Automatic clustering using a synergy of genetic algorithm and multi-objective differential evolution, in International Conference on Hybrid Artificial Intelligence Systems. Springer, 177–186.
- [63] Paoli, A., Melgani, F., & Pasolli, E. Clustering of hyperspectral images based on multiobjective particle swarm optimization., *IEEE Transactions on Geoscience and Remote Sensing*, **47**(12-2), 4175–4188, 2009.
- [64] Ma, W., Jiao, L., & Gong, M. Immunodominance and clonal selection inspired multiobjective clustering, *Progress in Natural Science*, **19**(6), 751–758, 2009.
- [65] Saha, S. & Bandyopadhyay, S. A new multiobjective clustering technique based on the concepts of stability and symmetry, *Knowledge and Information Systems*, **23**(1), 1–27, 2010.
- [66] Saha, S. & Bandyopadhyay, S. A symmetry based multiobjective clustering technique for automatic evolution of clusters, *Pattern recognition*, **43**(3), 738–751, 2010.
- [67] Saha, I., Maulik, U., & Plewczynski, D. A new multi-objective technique for differential fuzzy clustering, *Applied Soft Computing*, **11**(2), 2765–2776, 2011.
- [68] Liu, R., Wang, X., Li, Y., & Zhang, X. Multi-objective invasive weed optimization algorithm for clustering, in Evolutionary Computation (CEC), 2012 IEEE Congress on. IEEE, 1–8.
- [69] Nanda, S. & Panda, G. Automatic clustering using monoclonal for classifying actions of 3d human models, in Humanities, Science and Engineering Research (SHUSER), 2012 IEEE Symposium on. IEEE, 945–950.

- [70] Xia, H., Zhuang, J., & Yu, D. Novel soft subspace clustering with multi-objective evolutionary approach for high-dimensional data, *Pattern Recognition*, **46**(9), 2562–2575, 2013.
- [71] Zhong, Y., Zhang, S., & Zhang, L. Automatic fuzzy clustering based on adaptive multi-objective differential evolution for remote sensing imagery, *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens*, **6**(5), 2290–2301, 2013.
- [72] Nanda, S. J. & Panda, G. Automatic clustering algorithm based on multi-objective immunized pso to classify actions of 3d human models, *Engineering Applications of Artificial Intelligence*, **26**(5-6), 1429–1441, 2013.
- [73] Abubaker, A., Baharum, A., & Alrefaei, M. Automatic clustering using multi-objective particle swarm and simulated annealing, *PloS one*, **10**(7), e0130995, 2015.
- [74] Saha, S. & Bandyopadhyay, S. A generalized automatic clustering algorithm in a multiobjective framework, *Applied Soft Computing*, **13**(1), 89–108, 2013.
- [75] Huang, C.-L. & Wang, C.-J. A ga-based feature selection and parameters optimization for support vector machines, *Expert Systems with applications*, **31**(2), 231–240, 2006.
- [76] Muni, D. P., Pal, N. R., & Das, J. Genetic programming for simultaneous feature selection and classifier design, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **36**(1), 106–117, 2006.
- [77] Li, D., Ma, C., Wang, Y., & Zhang, D. Design and optimization of an athermal asymmetric all-polymer arrayed waveguide grating, *Modern Physics Letters B*, **22**(04), 295–302, 2008.

- [78] Hamdani, T. M., Won, J.-M., Alimi, A. M., & Karray, F. Multi-objective feature selection with nsga ii, in International Conference on Adaptive and Natural Computing Algorithms. Springer, 240–247.
- [79] Winotai, P., Limsuwan, P., & Rittikulsittichai, S. Optimization of heat treatments of vietnamese rubies, *Modern Physics Letters B*, **17**(29n30), 1537–1546, 2003.
- [80] Zhu, Z., Ong, Y.-S., & Dash, M. Wrapper–filter feature selection algorithm using a memetic framework, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **37**(1), 70–76, 2007.
- [81] Tang, Z. Uncertainty based robust optimization in aerodynamics, *Modern Physics Letters B*, **23**(03), 477–480, 2009.
- [82] Wang, X., Yang, J., Teng, X., Xia, W., & Jensen, R. Feature selection based on rough sets and particle swarm optimization, *Pattern recognition letters*, **28**(4), 459–471, 2007.
- [83] Wang, F., Zhang, B., Chai, S., & Xia, Y. Community detection in complex networks using proximate support vector clustering, *Modern Physics Letters B*, **32**(07), 1850101, 2018.
- [84] Sivagaminathan, R. K. & Ramakrishnan, S. A hybrid approach for feature subset selection using neural networks and ant colony optimization, *Expert systems with applications*, **33**(1), 49–60, 2007.
- [85] Lin, S.-W., Ying, K.-C., Chen, S.-C., & Lee, Z.-J. Particle swarm optimization for parameter determination and feature selection of support vector machines, *Expert systems with applications*, **35**(4), 1817–1824, 2008.
- [86] Soto, A. J., Cecchini, R. L., Vazquez, G. E., & Ponzoni, I. Multi-objective feature selection in qsar using a machine learning approach, *QSAR & Combinatorial Science*, **28**(11-12), 1509–1523, 2009.

- [87] Guillén, A., Pomares, H., González, J., Rojas, I., Valenzuela, O., & Prieto, B. Parallel multiobjective memetic rbfns design and feature selection for function approximation problems, *Neurocomputing*, **72**(16-18), 3541–3555, 2009.
- [88] Neshatian, K. & Zhang, M. Pareto front feature selection: using genetic programming to explore feature space, in Proceedings of the 11th Annual conference on Genetic and evolutionary computation. ACM, 1027–1034.
- [89] Huang, B., Buckley, B., & Kechadi, T.-M. Multi-objective feature selection by using nsga-ii for customer churn prediction in telecommunications, *Expert Systems with Applications*, **37**(5), 3638–3646, 2010.
- [90] Esseghir, M. A., Goncalves, G., & Slimani, Y. Adaptive particle swarm optimizer for feature selection, in International Conference on Intelligent Data Engineering and Automated Learning. Springer, 226–233.
- [91] Unler, A. & Murat, A. A discrete particle swarm optimization method for feature selection in binary classification problems, *European Journal of Operational Research*, **206**(3), 528–539, 2010.
- [92] Liu, Y., Wang, G., Chen, H., Dong, H., Zhu, X., & Wang, S. An improved particle swarm optimization for feature selection, *Journal of Bionic Engineering*, **8**(2), 191–200, 2011.
- [93] GWIZDAŁŁA, T. M. The ising model studied using evolutionary approach, *Modern Physics Letters B*, **19**(04), 169–179, 2005.
- [94] Ovchinnikov, I. V. & Wang, K. L. Stochastic dynamics and combinatorial optimization, *Modern Physics Letters B*, **31**(31), 1750285, 2017.
- [95] Morita, M., Sabourin, R., Bortolozzi, F., & Suen, C. Y. Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition, in Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings., 666–670.

- [96] Saha, S., Alok, A. K., & Ekbal, A. Use of semisupervised clustering and feature-selection techniques for identification of co-expressed genes, *IEEE journal of biomedical and health informatics*, **20**(4), 1171–1177, 2016.
- [97] Bandyopadhyay, S. & Saha, S. Gaps: A clustering method using a new point symmetry-based distance measure, *Pattern recognition*, **40**(12), 3430–3451, 2007.
- [98] Anusha, M. & Sathiaselvan, J. Feature selection using k-means genetic algorithm for multi-objective optimization, *Procedia Computer Science*, **57**, 1074–1080, 2015.
- [99] De la Hoz, E., de la Hoz, E., Ortiz, A., Ortega, J., & Martínez-Álvarez, A. Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps, *Knowledge-Based Systems*, **71**, 322–338, 2014.
- [100] Dutta, D., Dutta, P., & Sil, J. Simultaneous feature selection and clustering with mixed features by multi objective genetic algorithm, *International Journal of Hybrid Intelligent Systems*, **11**(1), 41–54, 2014.
- [101] Saha, S., Spandana, R., Ekbal, A., & Bandyopadhyay, S. Simultaneous feature selection and symmetry based clustering using multiobjective framework, *Applied Soft Computing*, **29**, 479–486, 2015.
- [102] Yuwono, M., Su, S. W., Guo, Y., Li, J., West, S., & Wall, J. Automatic feature selection using multiobjective cluster optimization for fault detection in a heating ventilation and air conditioning system, in *Artificial Intelligence, Modelling and Simulation (AIMS)*, 2013 1st International Conference on. IEEE, 171–176.
- [103] Corne, D. W., Knowles, J. D., & Oates, M. J. The pareto envelope-based selection algorithm for multiobjective optimization, in *International conference on parallel problem solving from nature*. Springer, 839–848.

- [104] Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. Pesa-ii: Region-based selection in evolutionary multiobjective optimization, in Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation. Morgan Kaufmann Publishers Inc., 283–290.
- [105] Handl, J. & Knowles, J. Evolutionary multiobjective clustering, in International Conference on Parallel Problem Solving from Nature. Springer, 1081–1091.
- [106] Handl, J. & Knowles, J. D. Multiobjective clustering around medoids, in Congress on Evolutionary Computation, 632–639.
- [107] Das, S., Abraham, A., & Konar, A. Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm, *Pattern recognition letters*, **29**(5), 688–699, 2008.
- [108] Maulik, U., Mukhopadhyay, A., & Bandyopadhyay, S. Combining pareto-optimal clusters using supervised learning for identifying co-expressed genes, *BMC bioinformatics*, **10**(1), 27, 2009.
- [109] Santos, D. S., De Oliveira, D., & Bazzan, A. L. A multiagent, multiobjective clustering algorithm, in Data mining and multi-agent integration, 239–249. Springer, 2009.
- [110] Suresh, K., Kundu, D., Ghosh, S., Das, S., Abraham, A., & Han, S. Y. Multi-objective differential evolution for automatic clustering with application to micro-array data analysis, *Sensors*, **9**(5), 3981–4004, 2009.
- [111] Saha, S. & Bandyopadhyay, S. A new multiobjective simulated annealing based clustering technique using symmetry, *Pattern Recognition Letters*, **30**(15), 1392–1403, 2009.
- [112] Huang, B., Buckley, B., & Kechadi, T.-M. Multi-objective feature selection by using nsga-ii for customer churn prediction in telecommunications, *Expert Systems with Applications*, **37**(5), 3638–3646, 2010.

- [113] Mukhopadhyay, A. & Maulik, U. A multiobjective approach to mr brain image segmentation, *Applied Soft Computing*, **11**(1), 872–880, 2011.
- [114] Zhu, L., Cao, L., & Yang, J. Multiobjective evolutionary algorithm-based soft subspace clustering, in Evolutionary Computation (CEC), 2012 IEEE Congress on. IEEE, 1–8.
- [115] Xue, B., Zhang, M., & Browne, W. N. Multi-objective particle swarm optimisation (pso) for feature selection, in Proceedings of the 14th annual conference on Genetic and evolutionary computation. ACM, 81–88.
- [116] Ali, H., Shahzad, W., & Khan, F. A. Energy-efficient clustering in mobile ad-hoc networks using multi-objective particle swarm optimization, *Applied Soft Computing*, **12**(7), 1913–1928, 2012.
- [117] Vignolo, L. D., Milone, D. H., & Scharcanski, J. Feature selection for face recognition based on multi-objective evolutionary wrappers, *Expert Systems with Applications*, **40**(13), 5077–5084, 2013.
- [118] Dutta, D., Dutta, P., & Sil, J. Simultaneous continuous feature selection and k clustering by multi objective genetic algorithm, in Advance Computing Conference (IACC), 2013 IEEE 3rd International. IEEE, 937–942.
- [119] Tan, C. J., Lim, C. P., & Cheah, Y.-N. A multi-objective evolutionary algorithm-based ensemble optimizer for feature selection and classification with neural network models, *Neurocomputing*, **125**, 217–228, 2014.
- [120] Wu, C.-W., Chiang, T.-C., & Fu, L.-C. An ant colony optimization algorithm for multi-objective clustering in mobile ad hoc networks, in Evolutionary computation (CEC), 2014 IEEE congress on. IEEE, 2963–2968.
- [121] Saha, S., Ekbal, A., Alok, A. K., & Spandana, R. Feature selection and semi-supervised clustering using multiobjective optimization, *SpringerPlus*, **3**(1), 465, 2014.

- [122] George, G. & Parthiban, L. Multi objective fractional cuckoo search for data clustering and its application to medical field, *Journal of Medical Imaging and Health Informatics*, **5**(3), 423–434, 2015.
- [123] Kashyap, H., Das, S., Bhattacharjee, J., Halder, R., & Goswami, S. Multi-objective genetic algorithm setup for feature subset selection in clustering, in Recent Advances in Information Technology (RAIT), 2016 3rd International Conference on. IEEE, 243–247.
- [124] Heloulou, I., Radjef, M. S., & Kechadi, M. T. Automatic multi-objective clustering based on game theory, *Expert Systems with Applications*, **67**, 32–48, 2017.
- [125] Gong, C., Chen, H., He, W., & Zhang, Z. Improved multi-objective clustering algorithm using particle swarm optimization, *PloS one*, **12**(12), e0188815, 2017.
- [126] Jiménez, F., Sánchez, G., García, J. M., Sciavicco, G., & Miralles, L. Multi-objective evolutionary feature selection for online sales forecasting, *Neurocomputing*, **234**, 75–92, 2017.
- [127] Zhang, Y., Gong, D.-w., Sun, X.-y., & Guo, Y.-n. A pso-based multi-objective multi-label feature selection method in classification, *Scientific reports*, **7**(1), 376, 2017.
- [128] Li, X. & Wong, K.-C. Evolutionary multiobjective clustering and its applications to patient stratification, *IEEE Transactions on Cybernetics*, (99), 1–14, 2018.
- [129] Dong, Z., Jia, H., & Liu, M. An adaptive multiobjective genetic algorithm with fuzzy-means for automatic data clustering, *Mathematical Problems in Engineering*, **2018**, 2018.
- [130] Olorunda, O. & Engelbrecht, A. P. Measuring exploration/exploitation in particle swarms using swarm diversity, in Evolutionary Computation,

2008. CEC 2008.(IEEE World Congress on Computational Intelligence).
IEEE Congress on. IEEE, 1128–1134.
- [131] Ilany, A., Booms, A. S., & Holekamp, K. E. Topological effects of network structure on long-term social network dynamics in a wild mammal, *Ecology letters*, **18**(7), 687–695, 2015.
- [132] Digalakis, J. G. & Margaritis, K. G. On benchmarking functions for genetic algorithms, *International journal of computer mathematics*, **77**(4), 481–506, 2001.
- [133] Yang, X.-S. Firefly algorithm, stochastic test functions and design optimisation, *International Journal of Bio-Inspired Computation*, **2**(2), 78–84, 2010.
- [134] Liang, J.-J., Suganthan, P. N., & Deb, K. Novel composition test functions for numerical global optimization, in Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE. IEEE, 68–75.
- [135] Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y.-P., Auger, A., & Tiwari, S. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization, *KanGAL report*, **2005005**, 2005, 2005.
- [136] Mirjalili, S., Mirjalili, S. M., & Lewis, A. Grey wolf optimizer, *Advances in engineering software*, **69**, 46–61, 2014.
- [137] Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowledge-Based Systems*, **89**, 228–249, 2015.
- [138] Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. Multi-verse optimizer: a nature-inspired algorithm for global optimization, *Neural Computing and Applications*, **27**(2), 495–513, 2016.

- [139] Mirjalili, S. Sca: a sine cosine algorithm for solving optimization problems, *Knowledge-Based Systems*, **96**, 120–133, 2016.
- [140] Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. Gsa: a gravitational search algorithm, *Information sciences*, **179**(13), 2232–2248, 2009.
- [141] Bonabeau, E., Marco, D. d. R. D. F, Dorigo, M., Théraulaz, G., Theraulaz, G., et al. *Swarm intelligence: from natural to artificial systems*. Number 1. Oxford university press, 1999.
- [142] Geem, Z. W., Kim, J. H., & Loganathan, G. V. A new heuristic optimization algorithm: harmony search, *Simulation*, **76**(2), 60–68, 2001.
- [143] Coello, C. A. C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, *Computer methods in applied mechanics and engineering*, **191**(11-12), 1245–1287, 2002.
- [144] Kannan, B. & Kramer, S. N. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, *Journal of mechanical design*, **116**(2), 405–411, 1994.
- [145] Gandomi, A. H. & Yang, X.-S. Benchmark problems in structural optimization, in *Computational optimization, methods and algorithms*, 259–281. Springer, 2011.
- [146] Mezura-Montes, E. & Coello, C. A. C. Useful infeasible solutions in engineering optimization with evolutionary algorithms, in *Mexican International Conference on Artificial Intelligence*. Springer, 652–662.
- [147] Waters, A., Blanchette, F., & Kim, A. D. Modeling huddling penguins, *PLoS One*, **7**(11), e50277, 2012.

- [148] Chen, Q., Liu, B., Zhang, Q., Liang, J., Suganthan, P., & Qu, B. Problem definitions and evaluation criteria for cec 2015 special session on bound constrained single-objective computationally expensive numerical optimization, *Technical Report, Nanyang Technological University*, , 2014.
- [149] Dhiman, G. & Kumar, V. Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications, *Advances in Engineering Software*, **114**, 48–70, 2017.
- [150] Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. Handling multiple objectives with particle swarm optimization, *IEEE Transactions on evolutionary computation*, **8**(3), 256–279, 2004.
- [151] Knowles, J. D. & Corne, D. W. Approximating the nondominated front using the pareto archived evolution strategy, *Evolutionary computation*, **8**(2), 149–172, 2000.
- [152] Zhang, Q., Zhou, A., Zhao, S., Suganthan, P. N., Liu, W., & Tiwari, S. Multiobjective optimization test instances for the cec 2009 special session and competition, *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, **264**, 2008.
- [153] Zitzler, E., Deb, K., & Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical results, *Evolutionary computation*, **8**(2), 173–195, 2000.
- [154] Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. Scalable test problems for evolutionary multiobjective optimization, in *Evolutionary multiobjective optimization*, 105–145. Springer, 2005.
- [155] Coello Coello, C. & Lechuga, M. Mopso: a proposal for multiple objective particle swarm optimization, in *Proc., Evolutionary Computation*, 2002. CEC’02. Proceedings of the 2002 Congress on, 1051–1056.

- [156] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii, *IEEE transactions on evolutionary computation*, **6**(2), 182–197, 2002.
- [157] Zhang, Q. & Li, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition, *IEEE Transactions on evolutionary computation*, **11**(6), 712–731, 2007.
- [158] Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. Pesa-ii: Region-based selection in evolutionary multiobjective optimization, in *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., 283–290.
- [159] Zitzler, E., Laumanns, M., & Thiele, L. Spea2: Improving the strength pareto evolutionary algorithm, *TIK-report*, **103**, 2001.
- [160] Angus, D. & Woodward, C. Multiple objective ant colony optimisation, *Swarm intelligence*, **3**(1), 69–85, 2009.
- [161] Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. A comprehensive review of swarm optimization algorithms, *PLoS One*, **10**(5), e0122827, 2015.
- [162] Das, S., Abraham, A., & Konar, A. Automatic clustering using an improved differential evolution algorithm, *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, **38**(1), 218–237, 2008.
- [163] Blake, C. L. Uci repository of machine learning databases, irvine, university of california, **1**, 1998.
- [164] Jain, A. K., Murty, M. N., & Flynn, P. J. Data clustering: a review, *ACM computing surveys (CSUR)*, **31**(3), 264–323, 1999.
- [165] Şen, B. & Peker, M. Novel approaches for automated epileptic diagnosis using fcbf selection and classification algorithms, *Turkish Journal of Electrical Engineering & Computer Sciences*, **21**(Sup. 1), 2092–2109, 2013.

- [166] Javidi, M. M. & Emami, N. A hybrid search method of wrapper feature selection by chaos particle swarm optimization and local search, *Turkish Journal of Electrical Engineering & Computer Sciences*, **24**(5), 3852–3861, 2016.
- [167] Günal, S. Hybrid feature selection for text classification, *Turkish Journal of Electrical Engineering & Computer Sciences*, **20**(Sup. 2), 1296–1311, 2012.
- [168] Rui, Y., Huang, T. S., & Chang, S.-F. Image retrieval: Current techniques, promising directions, and open issues, *Journal of visual communication and image representation*, **10**(1), 39–62, 1999.
- [169] Jain, A. & Zongker, D. Feature selection: Evaluation, application, and small sample performance, *IEEE transactions on pattern analysis and machine intelligence*, **19**(2), 153–158, 1997.
- [170] BİRİCİK, G., Diri, B., & SÖNMEZ, A. C. Abstract feature extraction for text classification, *Turkish Journal of Electrical Engineering & Computer Sciences*, **20**(Sup. 1), 1137–1159, 2012.
- [171] Kumar, V., Chhabra, J. K., & Kumar, D. Automatic unsupervised feature selection using gravitational search algorithm, *IETE Journal of Research*, **61**(1), 22–31, 2015.
- [172] Yu, L. & Liu, H. Feature selection for high-dimensional data: A fast correlation-based filter solution, in Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03. AAAI Press, 856–863.
- [173] Buckland, M. & Gey, F. The relationship between recall and precision, *Journal of the American society for information science*, **45**(1), 12, 1994.
- [174] Chen, Y.-C., Pal, N. R., & Chung, I.-F. An integrated mechanism for feature selection and fuzzy rule extraction for classification, *IEEE Transactions on Fuzzy Systems*, **20**(4), 683–698, 2012.

- [175] Nakamura, R. Y., Pereira, L. A., Costa, K., Rodrigues, D., Papa, J. P., & Yang, X.-S. Bba: a binary bat algorithm for feature selection, in 25th SIBGRAPI conference on graphics, Patterns and Images. IEEE, 291–297.
- [176] Wang, Z., Sun, M., & Jiang, J. Automatically fast determining of feature number for ranking-based feature selection, *Electronics letters*, **48**(23), 1462–1463, 2012.
- [177] Xue, B., Fu, W., & Zhang, M. Multi-objective feature selection in classification: a differential evolution approach, in Asia-Pacific Conference on Simulated Evolution and Learning. Springer, 516–528.
- [178] Spolaor, N., Lorena, A. C., & Lee, H. D. Use of multiobjective genetic algorithms in feature selection, in Eleventh Brazilian Symposium on Neural Networks (SBRN). IEEE, 146–151.
- [179] Saha, S., Spandana, R., Ekbal, A., & Bandyopadhyay, S. Simultaneous feature selection and symmetry based clustering using multiobjective framework, *Applied Soft Computing*, **29**, 479–486, 2015.
- [180] Hartigan, J. A. & Wong, M. A. Algorithm as 136: A k-means clustering algorithm, *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **28**(1), 100–108, 1979.
- [181] Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P. O., & Herskowitz, I. The transcriptional program of sporulation in budding yeast, *Science*, **282**(5389), 699–705, 1998.
- [182] Iyer, V. R., Eisen, M. B., Ross, D. T., Schuler, G., Moore, T., Lee, J. C., Trent, J. M., Staudt, L. M., Hudson, J., Boguski, M. S., et al. The transcriptional program in the response of human fibroblasts to serum, *Science*, **283**(5398), 83–87, 1999.
- [183] Wen, X., Fuhrman, S., Michaels, G. S., Carr, D. B., Smith, S., Barker, J. L., & Somogyi, R. Large-scale temporal gene expression mapping of

central nervous system development, *Proceedings of the National Academy of Sciences*, **95**(1), 334–339, 1998.

- [184] Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, *Journal of computational and applied mathematics*, **20**, 53–65, 1987.

Author's Publications

The contribution of author's accepted publications in this thesis are mentioned below:

1. G. Dhiman and V. Kumar. Spotted Hyena Optimizer: A Novel Bio-inspired based Metaheuristic Technique for Engineering Applications. *Advances in Engineering Software (Elsevier)*, **114**, 48–70, 2017. **[SCI/SCIE Indexed, Impact Factor - 4.194]**
2. G. Dhiman and V. Kumar. Multi-objective Spotted Hyena Optimizer: A Multi-objective Optimization Algorithm for Engineering Problems. *Knowledge-Based Systems (Elsevier)*, **150**, 175–197, 2018. **[SCI/SCIE Indexed, Impact Factor - 5.101]**
3. G. Dhiman and V. Kumar. Emperor Penguin Optimizer: A Bio-inspired Algorithm for Engineering Problems. *Knowledge-Based Systems (Elsevier)*, **159**, 20–50, 2018. **[SCI/SCIE Indexed, Impact Factor - 5.101]**
4. G. Dhiman and V. Kumar. Astrophysics inspired Multi-objective Approach for Automatic Clustering and Feature Selection in Real-life Environment. *Modern Physics Letters B (World Scientific)*, **32**(31), 1–23, 2018. **[SCI/SCIE Indexed, Impact Factor - 0.9]**
5. G. Dhiman and V. Kumar. Spotted Hyena Optimizer for Solving Complex and Non-linear Constrained Engineering Problems. In *Proceedings of Advances in Intelligent Systems and Computing (Springer)*, **741**, 2018. **[Scopus Indexed]**

The contribution of author's communicated publications in this thesis are mentioned below:

1. G. Dhiman and V. Kumar. Automatic Clustering using Multi-objective Spotted Hyena Optimizer and its Application to Image Segmentation. *Neural Computing and Applications (Springer)*, **[SCI/SCIE Indexed, Impact Factor - 4.664] (Under Review)**

2. G. Dhiman and V. Kumar. Automatic Feature Selection using Multi-objective Spotted Hyena Optimizer. *Engineering with Computers*, [**SCI/SCIE Indexed, Impact Factor - 3.551**] (*Under Review*)

- ACMOSHO, [130](#)
- ACO, [27](#)
- AFCMDE, [25](#)
- AFSMOSHO, [142](#)
- Algorithm, [70](#)
- AMOS, [31](#)
- AMOSHO_CFS, [156](#)
- ANNs, [27](#)
- ANOVA, [51](#), [91](#)
- Astrophysics, [154](#)
- Benchmark, [48](#)
- Bhattacharyya Statistical Distance
(BD), [23](#)
- Breast Cancer, [163](#)
- Bupa, [163](#)
- Chebyshev Distance, [6](#)
- Classification, [1](#)
- Clustering, [1](#), [21](#), [130](#)
- CMC, [163](#)
- Composite, [80](#)
- Constraint Handling, [59](#)
- Convergence, [82](#)
- Cosine Distance, [5](#)
- Davis-Bouldin Index (DB Index), [7](#)
- DE, [22](#), [23](#)
- Displacement of Loaded Structure,
[68](#), [99](#)
- DTLZ, [107](#)
- Dunn Index (DI Index), [7](#)
- EPO, [38](#), [69](#)
- Epsilon, [15](#), [109](#)
- Euclidean Distance, [5](#)
- Evolutionary, [11](#)
- FCM, [23](#), [25](#)
- FeaClusMOO, [163](#)
- FeaClusMOO_EUC, [163](#)
- FeaClustMOO, [31](#)
- Feature Selection, [27](#), [142](#), [153](#)
- Fixed-dimension Multimodal, [80](#)
- GA, [23](#), [48](#)
- GADE, [23](#)
- GenClustMOO, [26](#), [135](#)
- Generational Distance (GD), [16](#)
- Glass, [163](#)
- GP, [28](#)
- GSA, [48](#)
- GWO, [48](#)
- Haberman, [163](#)
- HS, [48](#)
- Human Fibroblasts Serum, [168](#)
- HV, [15](#), [109](#)
- I Index, [9](#)
- IDCMC, [23](#)

IEEE CEC-2009, [107](#)
 IEEE CEC-2015, [80](#)
 Inverted Generational Distance
 (IGD), [16](#)
 Iris, [163](#)

 K-means, [163](#)

 Laplacian Score, [150](#)
 Log-likelihood Function (LF), [23](#)

 Metaheuristic, [10](#)
 MFO, [48](#)
 Microarray, [168](#)
 Minimum Description Length (MDL),
 [23](#)
 MOACO, [109](#)
 MOCK, [135](#)
 MOCLONAL, [25](#)
 MODE, [24](#)
 MOEA, [21](#)
 MOEA/D, [109](#)
 MOEASSC, [25](#)
 MOIMPSO, [26](#)
 MOIWO, [25](#), [135](#)
 MOKGA, [22](#)
 MOMoDEFC, [24](#)
 MOPSO, [23](#), [101](#), [109](#)
 MOPSOSA, [26](#), [135](#)
 MOSHO, [104](#)
 Multi-objective, [21](#)
 Multi-objective Optimization, [14](#)
 Multimodal, [80](#)
 Multiple-disk Clutch Brake Design,
 [125](#)
 MVO, [48](#)

 NLMOGA, [30](#)
 No Free Lunch (NFL), [10](#)
 NSGA-II, [22](#), [109](#)

 Optimization, [70](#)

 Pakhira-Bandyopadhyay-Maulik Index
 (PBM Index), [8](#)
 Pareto, [12](#), [22](#), [26](#), [30](#)
 Performance Metrics, [15](#)
 PESA-II, [109](#)
 Pressure Vessel Design, [63](#), [95](#)
 PSO, [12](#), [26](#), [48](#)

 Rand Index, [23](#)
 Rat CNS, [168](#)
 Rolling Element Bearing Design, [66](#),
 [99](#)

 SCA, [48](#)
 Segmentation, [136](#)
 Semi-FeaClustMOO, [30](#)
 SHO, [38](#), [42](#)
 Silhouette Index, [23](#), [168](#)
 Simulated Annealing, [24](#)
 Single-objective Optimization, [13](#)

SPEA2, [109](#)
 Spearman Distance, [6](#)
 Speed Reducer Design, [64](#), [96](#)
 Spread, [15](#), [109](#)
 SSym-AMOSA, [24](#)
 Standardized Euclidean Distance, [5](#)
 SVM, [27](#), [29](#)
 Swarm, [12](#)
 Sym Index, [9](#)
 Tension/compression Spring Design,
 [61](#), [98](#)
 Threshold, [133](#)
 Unimodal, [80](#)
 Validity Indices, [6](#)
 VAMOSA, [24](#)
 VGAPS, [163](#)
 Vowel, [163](#)
 Welded Beam Design, [60](#), [97](#)
 WFFSA, [27](#)
 Wine, [163](#)
 XB, [24](#), [25](#)
 Xie and Beni Index (XB Index), [8](#)
 Yeast Sporulation, [168](#)
 ZDT, [107](#)