2017

Knowledge Migration Strategies for Optimization of Multi-Population Cultural Algorithm

Panth Parikh
University of Windsor

Knowledge Migration Strategies for Optimization of Multi-Population Cultural Algorithm

by

Panth Parikh

A Thesis

Submitted to the Faculty of Graduate Studies

through the School of Computer Science

in Partial Fulfillment of the Requirements for

the Degree of Master of Science at the

University of Windsor

Windsor, Ontario, Canada

2017

Knowledge Migration Strategies for Optimization of Multi-Population Cultural Algorithm

by

Panth Parikh

Approved by

M. Khalid

Department of Electrical and Computer Engineering

M. Kargar

School of Computer Science

Z. Kobti, Advisor

School of Computer Science

26^{th} *April*, 2017

Declaration of Previous Publication

This thesis includes 1 original paper that have been previously published/ submitted for publication in peer reviewed conference, as follows:

Section	Full Citation	Publication status
3.3.1	Panth Parikh and Ziad Kobti. "Comparative strategies for knowledge migration in multiobjective optimization problems". In 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering , Windsor, Canada, 2017.	Accepted

I certify that I have obtained a written permission from the copyright owner to include the above published material in my thesis. I certify that the above material describes work completed during my registration as a graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any propriety rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extend that I have included copyright material that surpasses the bounds of fair dealing within the meaning of Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner to include such material in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies Office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

Evolutionary Algorithms (EAs) are meta-heuristic algorithms used for optimization of complex problems. Cultural Algorithm (CA) is one of the EA which incorporates knowledge for optimization. CA with multiple population spaces each incorporating culture and genetic evolution to obtain better solutions are known as Multi-Population Cultural Algorithm (MPCA). MPCA allows to introduce a diversity of knowledge in a dynamic and heterogeneous environment. In an MPCA each population represents a solution space. An individual belonging to a given population could migrate from one population to another for the purpose of introducing new knowledge that influences other individuals in the population. In this thesis, we provide different migration strategies which are inspired from game theory model to improve the quality of solutions. Migration among the different population in MPCA can address the problem of knowledge sharing among population spaces. We have introduced five different migration strategies which are related to the field of economics. The principal idea behind incorporating these strategies is to improve the rate of convergence, increase diversity, better exploration of the search space, to avoid premature convergence and to escape from local optima. Strategies are particularly taken from the economics background as it allows the individual and the population to use their knowledge and make a decision whether to cooperate or to defect with other individuals and populations. We have tested the proposed algorithms against CEC 2015 expensive benchmark problems. These problems are a set of 15 functions which includes varied function categories. Results depict that it leads a to better solution when proposed algorithms used for problems with complex nature and higher dimensions. For 10 dimensional problems the proposed strategies have 7 out 15 better results and for 30 dimensional problems we have 12 out of 15 better results when compared to the existing algorithms.

Dedication

I would like to dedicate this to my friends and family.

Acknowledgments

There are many people to whom I would like to acknowledge for their help and support for my journey of the master thesis.

First and foremost I would pay my gratitude to my supervisor Dr. Ziad Kobti. Under his guidance, I had enjoyed a lot working on my research work. It was a great pleasure to work and discuss with him. Without his support, this won't have been possible. I would also like to appreciate the amount of time he invested in me, the funding he provided and also the knowledge he shared with me.

I would also like to thank Mrs. Gloria Menash, secretary of the director, who always helped me arrange the meetings with my supervisor and constantly reminding about the meetings I had with my advisor. Also, my sincere thanks to Mrs. Karen Bourdeau, for help and support, and taking care of all the other issues related to my master's degree.

I am also very thankful to my friends for their moral support and listening to my problems for long hours.

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

Panth Parikh

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Optimization is finding the best result by maximizing the desired factors and minimizing the undesired ones. Optimization problems are the problems to find the best solution from all the feasible solutions [5]. The optimization problem is applied to a wide range of areas like energy utilization, supply chain management, job scheduling, solving mathematical problems and much more. Optimization problems are distinguished by their goals of minimization and maximization. Evolutionary algorithms (EA) has been used widely by the researchers to solve the optimization problems. EA optimizes the problem efficiently as it contains the search space and searches for the best possible solution in it [48]. The solutions can be either near optimal or optimal. EA allows the exploration and exploitation of the search space. Exploration helps to search the whole space and exploitation helps the solution to mutate and generate offspring. The problem with EA is that it can fall into local optima (solutions think its optimal solution, but it is not) and easily lose diversity (solutions creates clones). Diversity can be maintained among the population by using Multi-Population Cultural Algorithm(MPCA). MPCA is a class of EA which is most widely used to solve multi-objective problems. Introducing migration in MPCA can address the issue of falling into local optima as migrating the individuals

from one population to another enhances the searching of unsearched spaces which shows good potential for better solutions.

Game Theory strategies are the strategies which are used in games by the players to cooperate or defect with the other players playing the game [13]. Game theory strategies bring in the social factor which makes the player make a decision about cooperating or defecting with other players for their benefit and increase their playoffs [46]. Using game theory strategies for migration in MPCA can provide the balance between exploration and exploitation in evolutionary algorithms. It can make better use of the knowledge possessed by the individuals in the population to make a decision of cooperating with some other individuals for their benefit and to generate better results. The combination of this two different fields can cover the major aspects of diversity, escaping from local optima, premature convergence, exploration, and exploitation. This combination can lead to good results and efficiently solve the optimization problems.

1.1 Evolutionary Computation

Evolutionary Computation (EC) is a set of an algorithm which is inspired by the biological model of evolution. EC is sub-branch of artificial intelligence which is used for metaheuristic and stochastic optimization of complex problems [48]. There are various algorithms which come under EC, such as:

1. Cultural Algorithms
2. Genetic Algorithms
3. Differential Evolution
4. Particle Swarm Optimization
5. Ant Colony Optimization Algorithm

Evolutionary algorithms (EA) is a subset of EC, and hence they are also considered as optimization algorithms. The common underlying concept in each evolutionary algorithm is the same: given a set of the population under environmental pressure causes natural selection. The fitness function measures the fitness of the candidates, and the better candidates survive for the next generation, discarding the worst ones. Evolution of each individual is carried out by applying mutation and recombination operators on it. Mutation is applied on one candidate and as a result, we get one new candidate while in recombination two candidates (called parents) are selected, and it results in one or more new candidates (called offsprings). Mutation and recombination operators lead to a new set of candidates (offsprings) which replace the existing old candidates for the next generation. This process iterates until a termination condition is achieved. Figure 1 depicts the pseudocode of the evolutionary algorithm [14]. EA have been used by many researchers to solve the multi-objective problems [8], data mining [16], management applications [3] and much more.

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```

Figure 1.1: Pseudo-code for EA

EA incorporating genetics into the process of evolution are known as Genetic Algorithms (GA). GA are heuristic search algorithms based on evolutionary ideas of natural selection. GA was first introduced by Holland [24] which represents an

abstract model of Darwinian evolution theory and biological genetics [24]. GA is used by many researchers to solve various search and optimization problems. However, a simple GA converges to a single optimum and is not suitable for multi-modal optimization [2]. GA uses coevolution to evolve complex structures which include explicit notions of modularity to provide a fair chance for complex structures to evolve in the form of co-adopted subcomponents. This structure is noticed when there is a need for rule hierarchies in classifier systems and subroutines in genetic programming [45]. Coevolution can be described as two or more individuals reciprocally affect each other in evolution. The major drawback of coevolution is that it has a good chance of losing the diversity among the population.

Differential Evolution (DE) is an EA which was designed by Storn and Price [45] to solve the global optimization problems. Canonical DE was designed to deal with continuous domain although it shows great performance on combinatorial optimization problems. Although, it is not possible to apply DE directly on continuous domains. Overall DE shows good performance on both space trajectory optimization [49] and multi-area economic dispatch [43] as global optimization problems and also on some permutation problems [34, 40, 51]. DE is one of the popular EA due to its strong search space exploration [27]. DE makes use of differential formulation mechanism for generating offspring from current generations. All of the EAs are used to solve complex search and optimization problems, but none of them uses knowledge of the agent to do so. To make use of the knowledge possessed by the individuals or population Reynolds [41] introduced Cultural Algorithms (CA). CA is an EA which incorporates knowledge to direct the search process. CA show a large number of successful applications which depicts the performance of knowledge-based EA. In CA the knowledge is extracted and incorporated to revise its search mechanism. The extracted knowledge helps the CA

to find solutions with better quality and improves the convergence rate. CA is inspired from the biological model of human culture and beliefs. Unlike other EAs, CA has two components: population space and belief space. Population space is consist of individuals in the population and belief space stores the knowledge of the best individual of the population in the current generation. CA incorporates with different knowledge like situational, topological, historical, normative and domain. CAs with single population have a high chance of losing diversity and can be difficult to implement on real world problems with the dynamic population. To overcome this Multi-population Cultural Algorithm (MPCA) were introduced. The major problem with standard EA used for dynamic optimization problems appears to be that EA eventually converges to an optimum and loses its diversity which is necessary for exploring the search space. Also, they also lose their ability to adapt to the change in the environment. Therefore MPCA were introduced. Digalakis et. al [11] first introduced MPCA to solve the electric generator scheduling problem. MPCA consist of multiple populations which increases diversity in the population. They resemble more with the real world problems where the nature of problems is more dynamic and continuously varying over a range. In MPCA there are more parameters which can be adjusted when compared to CA. MPCA also allows exploring the large region of search space due to its widespread population. Incorporating different sub-population can solve the complex optimization problems with dynamic nature. To increase the convergence rate and to escape from local optima, migration can be incorporated in MPCA. Migration can also increase the diversity in the population and allows the topological knowledge to play its role in searching the unexplored search region which shows the potential for better results.

1.2 Game Theory

Game theory is the study of how two players play the game with a defined set or rules and regulations with an outcome [46]. Each player in a game would like to have an outcome which provides him with a large number of playoffs as possible. The player has some control over the outcome as his choice of strategy will influence it. However, the outcome is not only influenced by one player choice but also from the other players playing that game [46]. The player might cooperate or defect depending upon the strategy they use. There is a good chance of conflict between the players as a different player can use a different strategy. There is even a good chance of cooperation as players may decide to cooperate with each other to obtain an outcome with good playoffs for each player. Rational play can include a complicated individual decision as each player will decide a strategy which is favorable to the player itself, and also will be aware of the fact that other players will choose a strategy which will be favorable to them [13]. It will also involve the social decisions about how and with whom the player wants to cooperate for its betterment.

Game theory is so useful because it comes from the generalization and abstraction of the traditional games like chess, bridge, and poker. This abstraction and generalization are powerful enough to deal with many varied social issues of the society [46]. Companies following corporate strategies try to play a game. Similarly, the political candidates try to win an election; the political parties try to defeat or pass a bill, nations manipulating in the international arena and much more scenarios. In our work, the biological species can be considered as a player in a game in which payoff is a chance to pass the gene to the future generations.

1.3 Research Motivation

The major motivation for the research has come from observing the optimization techniques to solve the complex problems. While working on the optimization techniques, we found there are many algorithms which can be used. The main difficulty with most of the algorithm was that they were more problem specific and less general. The existing algorithms try to solve the problem in static rather than in dynamic way. After working in this field, we realized Multi-population Cultural Algorithms (MPCA) shows a lot of potentials to solve the complex problems and they resemble much with human culture. Exchange of knowledge among the individuals in the environment can help them to explore and exploit the conditions around them in a better way. We have tried to implement this idea in our work by introducing some strategies for migration in MPCA for the better quality of results. The migration strategies are inspired from the Game Theory model as they try to solve economic decisions and game moves. In our thesis, we mostly focus on implementing different migration strategies in MPCA for the better quality of results.

1.4 Thesis Contribution

In our work, we aim to develop and evaluate different migration strategies to improve the optimization of Multi-Population Cultural Algorithm. The migration strategies are inspired from the Game Theory model. Migration strategies are triggered at both individual and population level. Different migration strategies are compared with each other to evaluate and identify the better strategy on its performance of optimizing the complex problems. In our work, we hypothesize that migration strategies in MPCA will lead to better performance in search optimization in heterogeneous and dynamic spaces. In our study we hypothesize

when individuals migrate from one subpopulation to another by using a particular strategy will affect the whole population in a better way. We have developed our MPCA framework based on the work done by Raessi [27] and implemented different migration strategies to exchange cultural knowledge among populations. CEC 2015 [31] expensive benchmark functions have been used to test our framework and compare it with other existing algorithms. Testing is done on both 10 and 30-dimensional functions of CEC. The functions consist of different types like unimodal, simple multimodal, hybrid and composite functions.

1.5 Thesis Outline

The chapters of our research are organized in the following manner:

Chapter 1 contains the background, motivation and contribution of our research.

Chapter 2 describes in details the related work done in this field. It contains literature review of Cultural Algorithms, Multi-population Cultural Algorithms and Game theory model.

Chapter 3 describes the proposed algorithm with providing details of all the strategies and its implementations.

Chapter 4 provides all the details of experimental approach which contains outline of CEC functions, experimental setup and all the assumptions made.

Chapter 5 includes the evaluated results of the data mentioned in above chapter.

Chapter 6 includes the discussion in detail.

Chapter 7 contains the conclusion of our dissertation.

Chapter 2

Related Work

This chapter consists of all the related work used for the building of the fundamental concepts, developing of our framework and architecture of our thesis. In this section, we explain the literature related to Multi-Population Cultural Algorithm, Migration and Game theory strategies. Also the research motivation and idea we got from the published papers. The first section contains the Literature review of the related algorithms like Cultural Algorithm, Genetic Algorithm, and Multi-Population Cultural Algorithm. The second section of this chapter contains details of the game theory strategies which we have used for migration in our algorithm. The third section contains paper related to Multi-Population Cultural Algorithm and migration while the last section consists of adoption of Game Theory in EA.

2.1 Literature Review

This section consists of detailed explanation about the evolutionary algorithm (EA), different types of EA, Genetic Algorithm, Differential Evolution Cultural Algorithm and Multi-Population Cultural Algorithm.

2.1.1 Evolutionary Algorithm

Evolutionary algorithms (EAs) are a subset of those methods which has been successfully used in the past for optimization problems. EAs are generic population-based meta-heuristic optimization algorithms inspired by the biological model of evolution and the process of natural selection [14]. In EAs the population is randomly initialized over specific search space which is called the initial population. Then it incorporates evolutionary operators which include mutation and crossover. This operator creates new offsprings from the parent in the population. The selection operator selects the population with greater fitness from the parent and offspring which serves as population for next generation. The left over individuals are discarded from the population. This process continuous until the termination criteria is fulfilled which can be either reaching a maximum number of predefined generations or CPU time. EA are based on the simplified model of biological evolution [14]. To solve a problem, a particular environment can be created where potential solutions can evolve. Parameters of the problem shape up the environment which helps to evolve a good solution. EAs are a group of a probabilistic algorithm which is similar to the biological systems and artificial systems. There are many types of EA such as:

1. Genetic Algorithm
2. Differential Evolution
3. Cultural Algorithm
4. Multi-population Cultural Algorithm

2.1.2 Genetic Algorithm

Genetic Algorithms (GA) are a subset of EA; hence they are population-based evolutionary algorithms. GA were first introduced by Holland [23] but became popular after the works of Goldberg [4]. GA is mainly used to solve the search related and other optimization problems. They are very useful when very less is known about the domain. GA is consist of a group of individuals known as population, and these individuals are used to find the optimal solution within the specified search space. An initial random population is generated over the search space and evolutionary operators like mutation, recombination and selection are applied to them. In GAs after each generation, the best individuals are selected for mutation, recombination, selection, and crossover. The individuals also exchange knowledge among them by making use of the operators. GAs are very simple to code, and the population is not initialized at one point. Instead, they are spread across the search space for exploration. GAs use mutation, crossover, and selection operator to achieve an optimal solution and enhance exploration and exploitation.

1. Selection :

   The selection operator behaves similarly to the natural selection that is found in biological systems. They select the best individuals in the current generation based on their fitness. The individuals who are fitter are selected, and the weaker are discarded from the generation. The fitter individuals have a high chance of passing the knowledge from current generation to the next generation.

2. Crossover :

   This operator works similar to the biological model of reproduction. To individuals are selected from the current generation (parents) on their fitness

basis and are allowed to generate a new individual (offspring) in the next generation. This operator enhances the exploration in the search space.

3. Mutation :

   This operator is used to change or flip the solution of the individual, and hence it is rarely used in GA.

GA are used to solve both constrained and non-constrained problems which are based on natural selection. GA continuously modify the population in each generation to achieve an optimal solution. GA have been used in many applications like laser technology, image processing, VLSI, etc. GA are designed to solve the stationary optimization problems, and according to this history, they have not impressed much when applied to real world problems. GA also have disappointing results when applied to the dynamic and heterogeneous environment. If GA is applied to the dynamic environment, then there is a high chance of losing diversity in the population.

2.1.3 Differential Evolution

We have used DE as our evolutionary algorithm in MPCA. DE has the mutation, selection and crossover operators like every other EA. The major benefit of using DE is that it has strong search space exploration. It incorporates a differential formulation mechanism to generate offspring from the current individuals of the generation [27]. This can provide diversity among the population and help to escape from falling into local optima.

Differential Evolution (DE) is a subset of EA designed by Storn and Price [45] for solving global optimization problems. The canonical DE was initially designed to deal with continuous domains, but it also shows good performance on combinatorial optimization problems. DE shows remarkable performance on both space trajectory

optimization [49] and multi-economic dispatch [43] as global optimization problems. DE is one of the popular EA due to its good search space exploration [27]. It makes use of differential formulation to generate offspring from the current generation. It uses both mutation and crossover operators which are applied to all individuals of all generations. Each individual who is also represented as a solution in the search space is represented as a D-dimensional vector of real number values. This vector is called the target vector denoted by $X_{i,g}$, the i^{th} target vector of generation g:

$$X_{i,g} = [x_{1,i,g}, x_{2,i,g}, ..., x_{D,i,g}] \tag{2.1}$$

Where $x_{j,i,g}$ depicts the value for target vector $X_{i,g}$ of dimension j in range 1 to D. In each generation g the mutation operator is applied on each target vector $X_{i,g}$ in the generation to create the mutant vector $V_{i,g}$. There are many mutation strategies for DE, while the most general one is defined in equation 3.2.

$$V_{i,g} = X_r1, g + F * (X_r2, g - X_r3, g) \tag{2.2}$$

Where $X_{r1,g}$, $X_{r2,g}$, and $X_{r3,g}$ are three different vectors which are randomly selected within the same generation g. $X_{r1,g}$ is known as the base vector and two other are known as perturbation vectors. F is a scale factor which is used to determine the perturbation of the base vector $X_{r1,g}$.

In the equation 3.2 the mutant vector $V_{i,g}$ is calculated without the consideration of any value of target vector $X_{i,g}$, but modern equations may use the target vector values or any of its information like locality. Since the general mutation strategy includes random selected base vector and one level of perturbation it is called $DE/rand/1$. Many different strategies are introduced by Price et al. [39].

A comparative study is been provided by Mezura-Montes et al. [33] to compare eight other mutation and crossover strategies incorporating DEs. The authors claim

that $DE/best/1/bin$ is the most competitive DE regardless of the problem domain. Here, bin refers to the crossover operator which is described as follows and the $DE/best/1$ is described in equation 3.3.

$$V_{i,g} = X_best, g + F * (X_r1, g - X_r2, g) \tag{2.3}$$

Where $X_{r1,g}$ and $X_{r2,g}$ are randomly selected target vectors within generation g, $X_{best,g}$ represents the best solution within the same generation, and F is a scale factor.

After implementing the mutation operator and generating the mutant of the target vector, crossover operator is applied on both target and mutant vectors to generate a trail vector $Z_{i,g}$. The most popular and used crossover strategy is binomial crossover for DE which is shown in equation 3.4.

$$Z_{j,i,g} = \begin{cases} V_{j,i,g} & \text{if} r_j \leq C_r \text{ or } j = j_{rand} \\ X_{j,i,g} & \text{otherwise} \end{cases} \tag{2.4}$$

Where r_j is random number uniformly distributed in (0,1] interval for j^{th} dimension. C_r is the crossover probability which could be kept fixed for all generations or changing over the generations and j_{rand} is a randomly selected index to make sure that the trail vector $Z_{i,g}$ is different from target vector $X_{i,g}$ in at least one component.

Like all other evolutionary algorithms the final step is to implement the selection operator in each generation. The selection operator selects the better solution between the target vector $X_{i,g}$ and trail vector $Z_{i,g}$ by comparing their objective values. The selected solution is considered as target vector for the next generation denoted by $X_{i,g+1}$.

$$X_{i,g+1} = \begin{cases} Z_{i,g} & \text{if } f(Z_{i,g}) \leq f(X_{i,g}) \\ X_{i,g} & \text{otherwise} \end{cases} \tag{2.5}$$

2.1.4 Cultural Algorithms

Cultural Algorithm (CA) is an EA which is inspired by the model of the human evolution process. It incorporates knowledge which is used to direct the search spaces [41]. The knowledge is extracted by CA and then incorporated for benefiting its search mechanism. The extracted knowledge helps the CA to find solutions with better quality and also helps in improving the convergence rate.

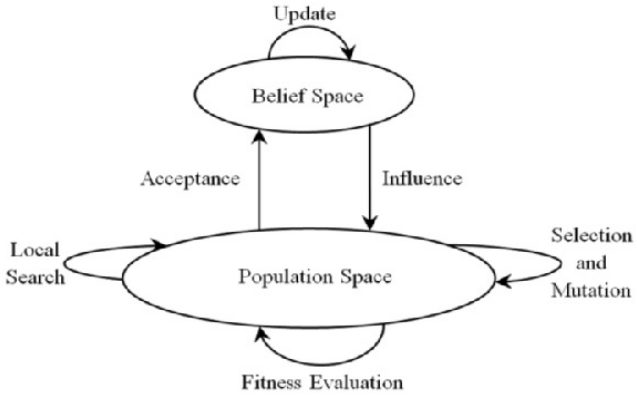

Figure 2.1: Architecture of CA

Figure 2.1 [42] illustrates the architecture of CA. As displayed in the figure, CA has population space, unlike any other EA where individuals reside. This space is managed by the EA like GA or DE. CA also has belief space which incorporates with knowledge. This space stores and update the knowledge extracted over generations. Both of the space communicate with each other by using the acceptance and influence operators. The knowledge circulation is defined as below.

1. The belief space receives the top performing individuals in the generation g from the population space by making the use of acceptance function.

2. The belief space updates its knowledge.

3. The belief space sends the update knowledge to the population space using influence function in the next generation *g+1*.

4. The population uses the knowledge to generate offsprings for next generation *g+1* from current generation *g*.

5. The top individuals from the next generation *g+1* are sent to the belief space to update its knowledge.

This cycle continuous until the termination condition is reached. The population of CA works like other EAs, but instead of using the random operators it uses knowledge-based evolutionary operators. Cultural Algorithm consists of two components [41].

1. Belief Space

2. Population Space

Belief Space

Belief space consists of different kinds of knowledge which are relevant to solving the problem. Due to this belief space is divided into separate categories. These categories contain different kinds of knowledge depending on which the population poses in the search space. The belief space is a repository where the knowledge is stored and is used by the population to obtain an optimal result. The belief space is updated after every iteration by the best individual in the search space. Other individuals in the population make use of this knowledge to move towards better search space. Artificial belief space stores the knowledge which is gained during the execution of the algorithm and makes use of it in the next generation and for its generic evolution. There are different types of knowledge in the belief space [22].

1. Situational Knowledge
2. Normative Knowledge
3. Topological Knowledge
4. Historical Knowledge
5. Domain Knowledge

Population Space

Population component is the space which consist of the individual in the population. The population component of CA is similar to that of GA. There are two function which allows the individual to move from population space to belief space and vice versa. The acceptance function transfers the best individual of the population space into belief space. After that the belief space updates its knowledge and updates the population space by making use of influence function. The individuals in the population space makes use of this knowledge to generate individuals for the next generation [28].

2.1.5 Multi-population Cultural Algorithm

Multi-population Cultural Algorithm (MPCA) can be considered as an extension of cultural algorithms. They are used to solve the optimization problems similar to CA. MPCA is CA incorporating multiple populations. The principle use of MPCA is to solve the knowledge migration/ sharing problem faced by CA. Digalakis et al. [11] were the first to introduce MPCA in their work to solve the electric generator scheduling problem. In the first model of MPCA, only the best solutions coming from each sub-populations were exchanged regarding migration rules. However, the best solutions only accounted for the current limited optimal information. MPCA

has a number of parameters to optimize when they are compared with the traditional CAs. For example, the number of the subpopulations, the size of a subpopulation, the migration rules and a number of individuals migrating. Guo et. al [20] successfully implemented MPCA for the multi-modal optimization problem, Yi-nan et al. [19] for interactive optimization and constrained optimization problems. Alami et. al [2] also proposed a method of dividing the sub-population based on fuzzy clustering and introduced the concept of cultural exchange between the subpopulations. According to them, the cultural exchange meant to exchange information among belief space of sub-populations. Hylnka et. al [22] also implemented a method to migrate agent among sub-population for the optimization problem. Raessi et. al [27] introduced a new concept to solve the optimization problem in which the subpopulations remained the same. Instead, the optimization parameters were divided among them.

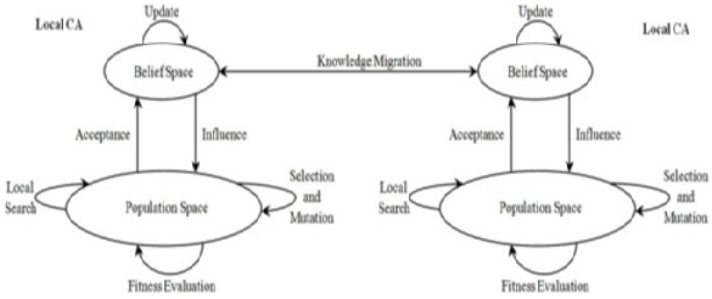

Figure 2.2: Architecture of MPCA

There are many versions of MPCA like Multi-Population Cultural Genetic Algorithms (MCGA), Multi-Population Cultural Differential Evolution (MCDE) and Multi-Population Cooperative Particle Swarm Cultural Algorithm (MCPSCA). The architecture of MPCA is depicted in figure 2.2 [20].

2.2 Game Theory Strategies

To incorporate migration in our Multi-Population Cultural Algorithm, we have made use of the game theory strategies. The conflict between the players in the game involves difficult and complicated decisions which also involves the social issues. We have used game theory strategies which are related to the field of economics [46]. The strategies used for migration among subpopulations are prisoners dilemma, oligopoly, duopoly, fair division and intra-household bargaining [13]. We have in particular made use of this strategies out of the vivid range of game theory strategies because we are trying to incorporate cooperative migration among the sub-population. But in the real world, it is not the case that every person wants to cooperate with each other, there might be many cases where people defect with each other. As we have made use of cultural algorithms which resembles very much with human culture and nature, so we have also introduced the defect nature in each migration strategy. This phenomenon allows generating a migration which is very much similar to human culture. It also involves a social impact where humans are in a dilemma to think about themselves (which is a defect in our case) or to think for the whole of the community (which is cooperate in our case). The migration takes place in each strategy by the combined decision of the whole population, by the dominators of the population or by the personal interest of the individual. All the three scenarios when analyzed properly can provide us with results from which we can decide which can be better for the individuals of the population.

2.2.1 Prisoner's Dilemma

Prisoners Dilemma is most widely used and important strategy of game theory [37] In this strategy, two criminals have committed a crime and are interrogated by the cops differently. The two criminals do not have any source of communication

between them. If both the criminals accept their crime, then both of them are punished for 3 years of imprisonment. If both of them do not accept the crime, then the punishment of imprisonment is 1 month as they are exempted and if one of them confesses to the crime than the confessed one is set free, and 5 years punishment of prison for other [13].

This strategy is incorporated in our algorithm as it tries to address the dilemma which humans make while making a decision, i.e., whether to be selfish or think for the welfare of all others affected by that decision. By introducing this strategy, we can notice whether the individuals in the population makes a decision to cooperate with the whole population or they focus on themselves during migration.

This strategy involves both exploitation and exploration as when individuals are migrated they discover the new search spaces, or else they guide the population towards their position.

2.2.2 Oligopoly

Oligopoly theory is concerned with market structures in which the actions of an individual firm affect all other firms and are affected by other firm actions. Oligopoly is a common market strategy where all the firms are in competition with each other. It can result in various kind of collusion which can reduce competition and can lead to increase in price for the consumers [13]. All oligopolistic are aware of the actions of each other. Oligopolistic competition can give rise to many outcomes. In some situations, the firm may use restrictive trade practice to raise the price and restrict production. Firms often collude in an attempt to stabilize unstable markets, so as to reduce the risks inherent in these markets for investment and product development. In some situations, the competition between firms can be fierce which can lead to low prices and high production.

2.2.3 Duopoly

Duopoly is a market in which two firms sell a product to a large number of consumers. Consumers are too small to affect the market price of the product, i.e.; the market is competitive on the buyers side. Each of the two sellers is a rational decision maker as its action will affect both himself and his rival [13]. Even though the interest of the seller is different, they are not wholly coincident or wholly in conflict. The sellers must concern them with the action of other seller is likely to do in the competitive market. The situation faced by the seller is non-cooperative in a sense as they are barred from making binding agreements with one another.

2.2.4 Fair Division

The traditional fair division has its origin in papers by Steinhaus [44] and Dubins and Spanier [12] who described the theory for sharing a perfect divisible cake among *'n'* people. In the method described by Steinhaus, each person cuts a slice from the cake and pass it to another person. This continues until every person has its slice or the cake is diminished. After the last person has its slice, the process continues again from the first person. The very similar method described by Dubins and Spainer, one person passes a knife continuously over the cake and at each instance determine a well-defined piece of cake which gradually increases over the time. The method got revised by many researchers over the period. According to the modern theory of fair division, an allocation is fair if and only if no person prefers any other persons share of cake to its own [Foley 1967].

2.2.5 Intra-household bargaining

Intra-household bargaining is a subfield of microeconomics which argues on the weakness of the New Home Economics does not only lie in the lack of recognition of

systematic, gender and age-based power relations which tend to structure household resource allocation [26]. Cooperative household bargaining model depicts the effectiveness of capturing preference and externally-derived bargaining power heterogeneity among family members; they treat individuals equally on their voice. According to its cooperative model, every individual in the house has equal opportunity to raise the voice in the decision of the resource allocation in the household, and if they deflect, then the bargaining also takes place with symmetric nature. This model of microeconomics mainly focuses on the feminine power for the household bargaining and the lack of equal opportunity provided to them which can lead to bad results. The results of the resource allocation of the house can improve if every individual has equal opportunity as if they even defect than the bargaining power does not only reside in the dominating individual of the house and each has an equal negotiation in the bargaining.

2.3 Multi-Population Cultural Algorithm

2.3.1 A multi-population cultural algorithm for the electrical generator scheduling problem

Authors Digalakis and Margaritis [11] were the first to introduce the Multi-Population Cultural Algorithm. They used MPCA to solve the electrical generator scheduling problem. The authors referred to the work of Mendes et al. [32] and proposed a guided local search (GLS) based parallel cultural algorithm which is a hybrid algorithm of GA and GLS procedure. The proposed algorithm is called Parallel Co-operating Cultural Algorithm (PARCA) in which the CAs were executed concurrently by the search programs. In this, the network of workstations was divided into two processors: a master processor and a slave processor. The master processor was in charge of initializing the population, managing the

population, performing the selection, mutation, and recombination. The slave processor was used to evaluate their simulations dispatched by the master processor. The population was divided into several sub-populations and were isolated from each other and managed their own local CA. The exchange of information between the populations allowed them to co-operate and to explore the promising areas of the search space, and also to reintroduce the previously lost genetic materials in the population. The populations also exchange their best individuals to enhance the search in the space. The architecture of PARCA is shown in figure 2.3 [11].

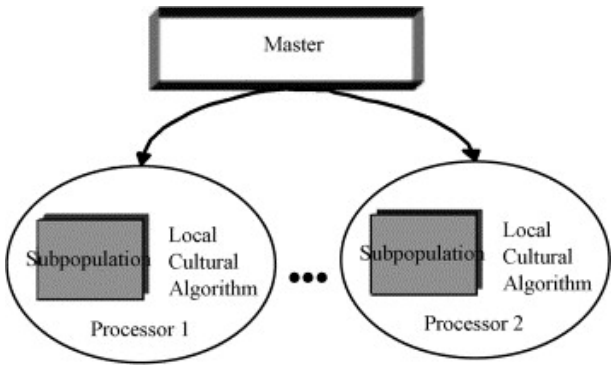

Figure 2.3: PARCA model

The authors implemented the PARCA using the message passing interface (MPI) standard. The configurations of their system were: SGI Origin 200 and 6 Pentium (P5/100 MHz) cluster with interconnection through Ethernet (100 MB/s) [11]. According to the authors, the algorithm showed better results of optimization but the cost and execution time was slightly more than the existing algorithms at that time.

2.3.2 Heterogeneous Multi-population Cultural Algorithm

Author Raessi et. al [27] stated that a group of sub-population which consists of different cultural algorithm do not directly communicate with each other so to overcome this problem the MPCA was introduced. Their work was inspired from

the work of Digalakis et. al [11], Holland et al. [24], Koza et. al [30] and Reynolds [41]. According to them, the evolutionary algorithms were successfully applied to solve the optimization problems, but the issue with them was they had good chances of immature convergence and falling into local optima. The major reason behind this was they were not able to preserve diversity among the population over the course of generations. The authors proposed a new framework of MPCA in which the subpopulations remained same, but the optimization parameters were divided among the sub-populations. Each sub-population optimized their parameters, and a set of the partial solution was generated. These partial solutions were combined to make the whole solution later. A detailed figure of the proposed architecture is depicted in figure 2.4 [27].

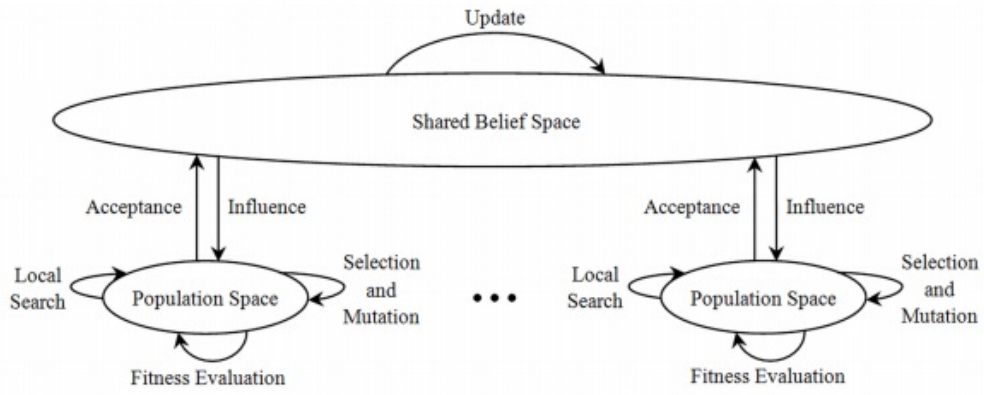

Figure 2.4: HMP-CA Architecture

The Heterogeneous Multi-population Cultural Algorithm was implemented using the JAVA platform by the authors. In their experiments, the population size was 1000 with 30 sub-populations, and each sub-population had 33 individuals. The experiments were carried out for 10000 generations and 10 iterations. CEC 2012 benchmark problems were used to test the proposed algorithm, and the experiments were carried out on 8 functions. The authors were successful in getting minimum results on 7 out of the 8 functions. The minimum value of only one function was not

found over the given time. The authors explained that only one function minimum value was not found over 10000 generations for five subpopulations, but they could have found it if the number of generations was increased. The authors claimed to find the minimum values of the numerical optimization functions and also their model was efficient in both time and space complexity.

2.3.3 A novel Multi-population Cultural Algorithm Adopting Knowledge Migration

Guo et. al [20] stated that in MPCA the information is exchanged among the sub-populations but at the individual level and not at the population level. The migration at the individual level does not consist of complete knowledge about the subpopulation which can make the evolution process slow. The authors in their work referred to the work of Reynold [41], Jin et. al [25] and Bin [36]. According to them most of the researchers did not take into account the exchange of implicit knowledge in MPCA. They also stated individuals of different subpopulation exchanged their knowledge in the belief space, but the method was not entirely clear.

The authors proposed a new model of MPCA which adopted knowledge migration known as Multi-Population Cultural Algorithm Adopting Knowledge Migration (MCAKM). In the proposed algorithm the knowledge is exchanged implicitly among the subpopulations instead of migrating individuals. In MPCA there are n number of subpopulations, and each adopts their cultural algorithm and the information among them is exchanged by migrating implicit knowledge at regular intervals. The proposed algorithm is shown in figure 2.5 [20].

To justify their algorithm the authors implemented it on high dimensional benchmark functions, and the performance of their algorithm was analyzed and compared with other proposed algorithms. The experiments were done by taking population size of 30 and 3 sub-populations with 0.3 as selection proportion and

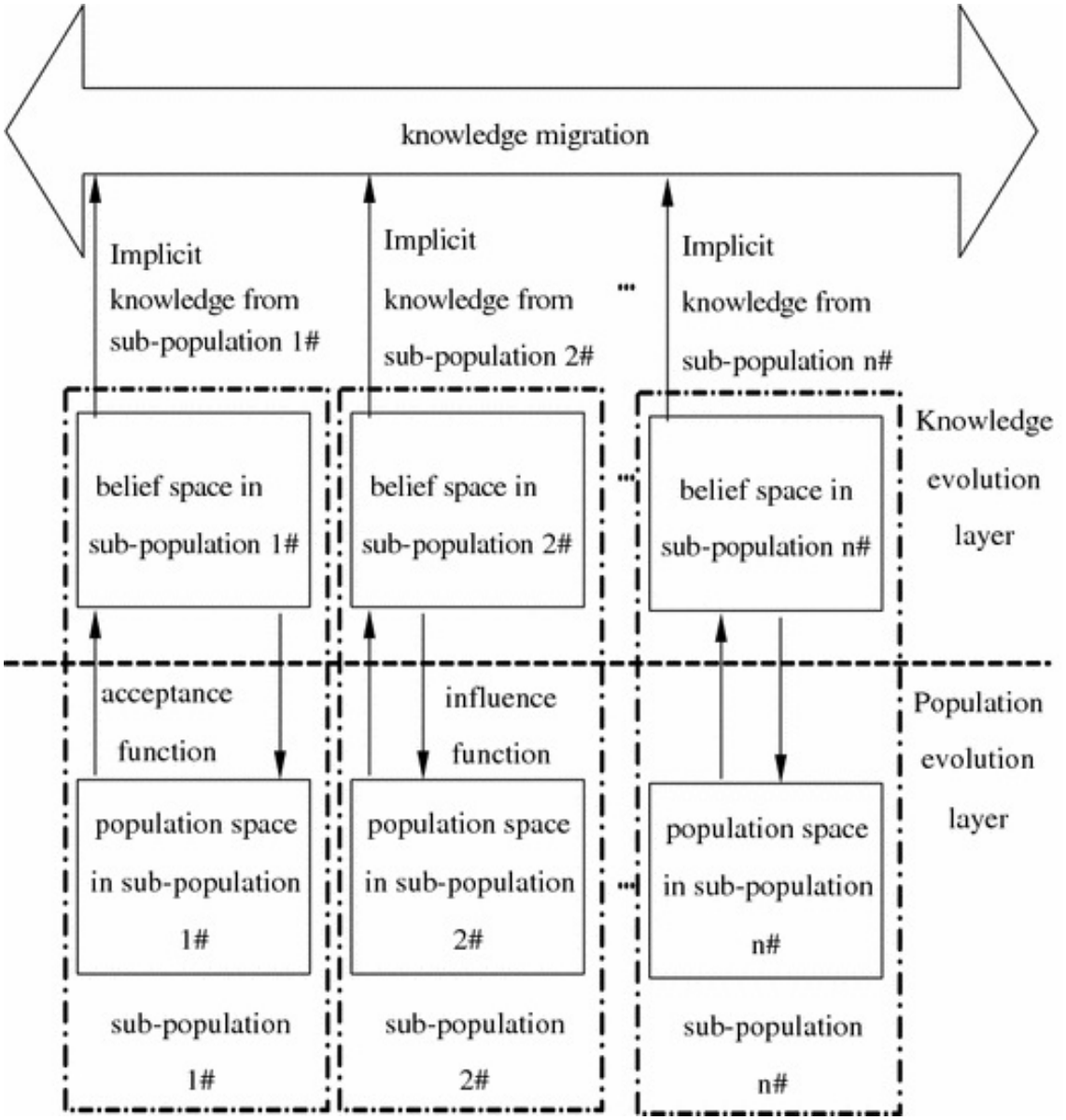

Figure 2.5: The structure of MCAKM

0.08 as the mutation probability. The experiments were run for 20 times with 100 iterations.

The authors validated their algorithm by comparing it with other cultural algorithm and MPCAs adopting influence range. They found their algorithm performed better compared to others and also had faster convergence speed with better solutions.

The new MPCA was inspired form the human cultural interactions and the

knowledge extracted from the evolution process were more efficient than others.

2.3.4 Knowledge Sharing Through Agent Migration with Multi-Population Cultural Algorithm

Hlynka et. al [22] in their work explained that sharing of knowledge at individual/ population level was always a problem and tried to solve this in their work. Reynold et. al [42], Kobti et. al [29], Guo et. al [20] and Raeesi et al.[27] works were referred by the authors. The authors mentioned other researchers did not use the implicit knowledge appropriately for migration and developed an MPCA with migration at the agent level. The authors ran their experiment using the Repast Simulation tool. Moving Peaks cone worlds domain was used to test their algorithm. Subpopulations performance was calculated by transferring the agent from one subpopulation to another over a period of time. The experiments were done by transferring 1%, 5%, 10%, 20% and 50% of the population and calculated the changes in the sub-populations performance.

They authors found the best and worst run time for topological and situational knowledge. They author claimed that their model performed better for only 1% transfer of agent among the sub-populations and also mentioned when a small group of agents moves the consistency of both the sub-populations can improve.

2.3.5 Promoting diversity using migration strategies in distributed genetic algorithms

Power et. al [38] in their work focused on increasing diversity in the population by introducing different migration strategies in Distributed Genetic Algorithm (DGA). The works of Grosso [18], Tanese [47] and Braun [6] was referred by the authors. The authors have stated that maintaining diversity among the population is

important for problems consisting dynamic landscapes or landscapes with lots of local optima.

In their work, the authors have focused on diversity measure when selecting an individual for migration among the parallel GAs. The individuals to migrate were not only selected on their fitness score but also on their location within the population. The clones of the individuals in the population were eliminated in each generation.

The authors have evaluated their algorithm on a number of the fitness function. The function used for testing the fitness was the onemax problem, multi-peaked three-dimensional landscape problems which include five hills and four valleys problem, waves problem, one center and four neighbors problem and six hump camel back problem. They have also experimented their algorithm on the benchmark functions used by Digalakis et. al [11] in their work. They have compared their migration strategy with the standard migration strategy and results depicts the author's strategy have outperformed on all the optimization problems they tested.

2.4 Game Theory in Evolutionary Algorithms

In this chapter, we discuss all the papers related to Game Theory Model and the techniques implemented in Evolutionary Algorithms.

2.4.1 An Evolutionary Game-Theoretical Approach to Particle Swarm Optimization

Author Chio et. al [10] integrated Prisoner's Dilemma, namely cooperate and defect into the Particle Swarm Optimization (PSO) algorithm. The authors referred to the work of Franken et. al [15], Adbelbar et. al [1], Pavlidis et. al [35] and Cui et. al [9]. The authors claimed that unlike other researchers they utilized the Prisoners

Dilemma (PD) framework, associating to each particle one of the two well-known strategies (cooperate or defect) which are interpreted as behaviors of particles. The authors claimed that they had a preliminary investigation in the direction of integrating PSO and Evolutionary Game Theory.

The authors stated that the underlying idea of their work is to modify the PSO algorithm, to make each particle in the swarm associated with a behavior (or strategy) incorporated from the PD framework (cooperate or defect) which is used to compute its position in next generation. Cooperate is considered as stronger social component and the defect is considered as a stronger individual component. The authors incorporated two strategies: flip-strategy, where a particle compares its fitness with its previous fitness and keeps its strategy if it improves or else changes it to another one. The other one was always C (cooperate), or D (defect) where the particle compares its fitness with its fitness in the previous generation and keeps C if it improves or keeps/changes to D if it decreases.

The authors compared their algorithm with 72 different PSO-EG variants. The experiments were done on Ackley, Griewank, Sphere, Rastrigin and Rosenbrock benchmark functions. Each experiment was done over 100 runs for both function values and normalized error. The results showed that the changing behavior is beneficial for the swarm especially when the dimension are increased and also for the multimodal problems.

2.4.2 Coevolution of cooperation and layer selection strategies in multiplex networks

Authors Hayashi et. al [21] in their work referred to the work of Gardens et. al [17], Wang et. al [50] and Buldyrev et. al [7]. Authors in their model have developed a co-evolutionary model of cooperation and layer selection strategies. Gardens et .al in their worked found that the evolution of cooperation was facilitated by the

multiplex structure of networks only when the temptation to defect was large [17]. In the model of authors, each individual has a layer selection strategy and prisoner's dilemma game (PDG) strategy (cooperate or defect). Each individual plays PD with the neighbor in its layer or in another layer in which it wants to move. If the fitness of the neighbor is better than the individual, then the individual imitates its neighbor strategy. The imitation probability is linear to the difference between the fitness values. If the individual fitness is higher than its neighbor than the individual keeps its strategy or else imitates its neighbor strategy. Schematic image of the authors model is depicted in figure 2.6 [21].

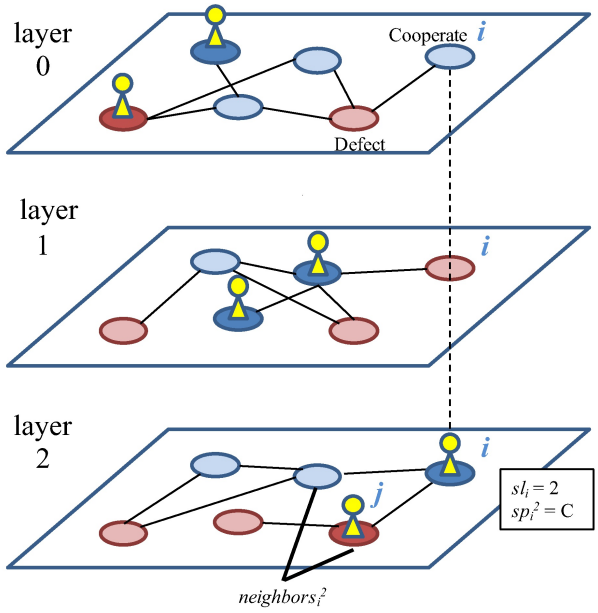

Figure 2.6: Schematic image of the model

The authors evaluated their work by having 100 individuals in the population, M = 1, 3,., 19 layers and b=1.1, ., 2.1 which is the temptation to defect. The experiments were done for five trials for each combination of layers and the temptation to defect. The authors claimed from their experiment results that the proportion of cooperative strategies has increased with increasing the number of layers and is not dependent on the degree of the dilemma. Also, the increase in cooperative strategies which is caused due to the cyclic coevolution process of layer

selection strategies and game theory strategies.

2.5 Conclusion

From works mentioned above, we can see the Evolutionary Algorithms work efficiently for the optimization problems. MPCA, in particular, is effective for a dynamic population with multiple cultures. Incorporating migration in MPCA has shown better convergence rate than the traditional MPCA. Using game theory concepts for migration is a new idea and has shown real potential. By using the game theory concepts for migration in MPCA can provide us with better results when implemented to optimization problems. Integrating this two different concepts can allow us to apply it to the heterogenous population on dynamic search space.

Chapter 3

Proposed Approach

In this chapter, we will introduce the pseudo-code and framework of our proposed algorithm. We will also discuss the design, belief space and population space used in our algorithm in detail. Later we will introduce knowledge migration among the subpopulations, and different migration strategies in this chapter.

3.1 Multi-population Cultural Algorithm with Migration

In this section, we will discuss the framework of our Multi-population Cultural Algorithm.

We have incorporated different migration strategies with the MPCA for having a strategical migration among the subpopulations. Implementing the strategies can provide us with better solutions in the population. Migration will be carried out by using all the introduced strategies. Based on the nature of strategy the migration can be of two types: cooperative and non-cooperative. The proposed model known as Knowledge Migration Strategies for Optimization of Multi-population Cultural Algorithm incorporates a local belief space for each population and a global belief

space which is shared by all the populations.

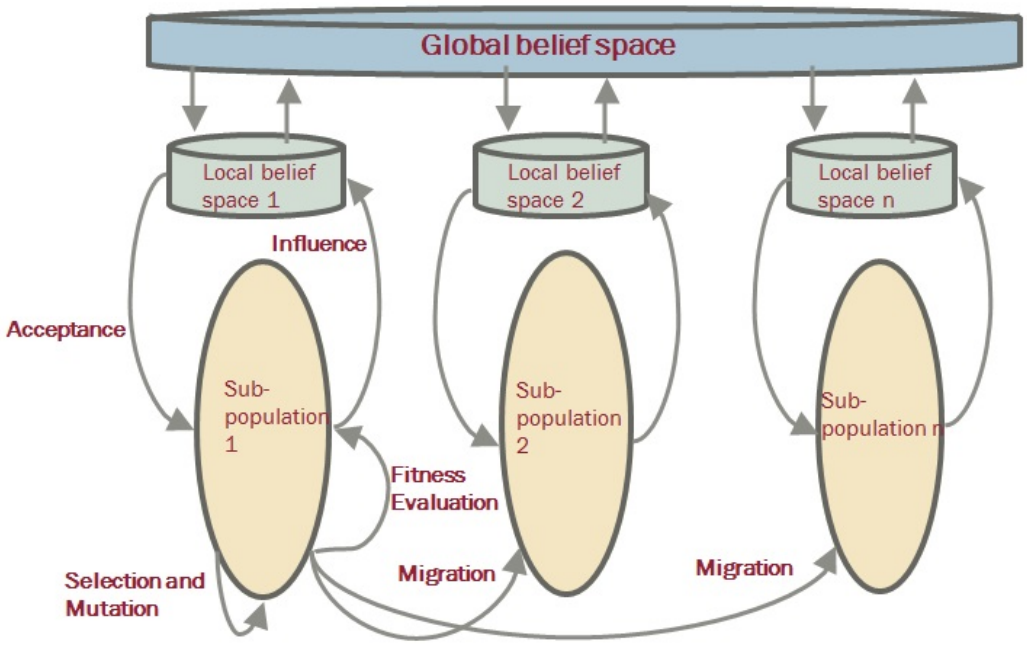

Figure 3.1: Architecture of our algorithm

Figure 3.1 illustrates the architecture of the proposed algorithm which is composed of a number of subpopulations which has its own belief space connected to a global belief space. The local belief space is very simple as it stores the best individual of that subpopulation for the current generation. Besides, these the global belief space stores the best individual of each subpopulation. The proposed model in this architecture incorporates the DE (DE/best/1 presented in equation 2.3 and binomial crossover illustrated in equation 2.4) as the evolutionary operators for every local CA. The crossover probability for the DE is random in the range of [0,1] for each generation, and the scale factor is selected randomly from intervals [0.5,2.5]

for each generation.

For our MPCA framework, we have made use of two different knowledge type: situational and topological. In the local belief space, the situational knowledge is stored as the solution of the best agent of the current generation resides in it. The knowledge possessed by the best agent is stored by using the acceptance function, and it is later used for guiding the other agents of the population to move towards a better region in the search space.On the other hand, the global belief space makes use of the topological knowledge during the migration process. All the best individuals of each population are stored in global belief space. Each individual in the global belief space is aware of its neighbor position which makes the migration process simpler and effective.

3.2 Knowledge Migration Strategy for Optimization of MPCA

In this section, we will explain all the steps to implement our proposed algorithm in detail.

- The population is initialized in the search space randomly.
- Then, the entire population is segregated into a number of subpopulations where each subpopulation has an equal number of agents.
- Each agent is initially assigned a strategy (i.e., cooperate or defect) during its initialization.
- Every subpopulation has its local belief space which is connected with common global belief space.

- DE is applied on each subpopulation to carry out evolution and generate offspring's for next generations.
- Migration is done using the strategy when the migration factor is satisfied.
- This process continuous until the termination condition is satisfied.

The pseudo-code of the model is described below.

Input: Test Problems and Algorithm Parameters;

Output: Optimal or near Optimal result;

---

1. Initialize number of Local CA as LocalCANo;

2. Generate initial population;

3. Initialize strategy C (Cooperate) or D (Defect) to each agent in population;

4. Divide population equally among LocalCAs;

5. **for** *Max_Generation* **do**

    6. **for** *(Each sub-population)* **do**

        7. Apply DE operator to generate offsprings;

        8. Evaluate fitness of all agent;

        9. Update Local Belief Space;

        10. Update Global Belief Space;

        11. **if** *Migration_Factor* **then**

            12. Apply Migration Strategy;

        end

    end

    Output the best individual found so far;

end

Algorithm 1: Framework of MPCA for Knowledge Migration Strategy

The implementation of the migration strategies is described in the next section.

3.3 Migration Strategies

3.3.1 Prisoners Dilemma

In this migration strategy, an agent to migrate is randomly chosen from the best performing subpopulation. Then one more agent is selected randomly from the same subpopulation. Both of the selected agents play prisoners dilemma with each other. In this, the fitness of both the agents is compared with each other. If the fitness of the migrating agent is greater than the randomly selected agent, then the migrating agent will keep its own strategy (i.e., to cooperate or defect) or else imitates the randomly selected agent strategy. If the migration agent strategy is changed and if it is to cooperate then it will migrate to subpopulation who's average fitness is less than its own fitness value having least difference. If the changed strategy is to defect then the agent will move to the subpopulation which contains the best performing agent in the whole population. On the other hand, if the migration strategy of the migrating agent is not changed when compared with its neighbor and it is C than then the migrating agent will move to least performing subpopulation in the context of average fitness. Else the agent will migrate to the sub-population with the best-performing individual in the entire population. Below is the pseudo-code of prisoner's dilemma for the minimization problem.

3.3.2 Oligopoly

In this strategy, initially, the subpopulations are assigned a strategy either high, medium or low in a random manner. Then the subpopulation are sorted according to their average fitness, and the best performing 30% of subpopulations are considered as the seller, and the rest are considered as buyers. The strategy of the best subpopulation from the seller is chosen as selling strategy, and the strategy of the best subpopulation from the buyer is chosen as buying strategy. Then a

Input : Subpopulations;
Output : Agent to migrate;

```
1. Sort LocalCA according to Avg_Fitness;
2. Select LocalCA with best Avg_Fitness;
3. Select a Random Agent to Migrate (Migrate_Agent);
4. Select another Random Agent from same sub-population (Random_Agent);
5. if Fitness(Migrate_Agent ≤ Random_Agent) then
   6. Migrate_Agent keeps own Strategy (C or D);
else
   7. Migrate_Agent(Strategy) = Random_Agent(Strategy);
   8. Strategy_Change = 1;
end
9. if Migrate_Agent(Strategy) = C then
   10. if Strategy_Change = 0 then
      11. Destination = LocalCA with highest Avg_Fitness;
   else
      12. Destination = LocalCA with high Avg_Fintess where difference
       between Migrate_Agent and LocalCA(Avg_Fintess) is least
   end
else
   13. Destination = LocalCA with best Agent in Entire Population;
   14. if Destination = Migrate_Agent(LocalCA) then
      15. Destination = Next best Agent in whole population;
   end
end
16. Migrate_Agent moves to destination;
```

Algorithm 2: Prisoner's Dilemma Migration Strategy

subpopulation is chosen at random from the sellers, and if the selling strategy is high, then the best individual is to be migrated from that subpopulation if the strategy is medium than the average performing agent and if low then the least performing agent is to be migrated. Then a buyer is selected randomly as the destination of the migrating agent. After the migration takes place, the buyer subpopulation will use the buying strategy to migrate an agent from its subpopulation to the selling subpopulation as a price of the product. It follows the same rule of high, medium and low followed by the seller subpopulation to migrate an agent. Below is the pseudo-code of oligopoly.

Input : Subpopulations;

Output : Agent to migrate;

1. Initially Strategy is assigned to all LocalCA as H(high), M(medium) and L(low);

2. Subpopulations are arranged according to Avg_Fitness;

3. Top performing 30% are considered as sellers and rest are buyers;

4. Seller_Strategy = Best_LocalCA_Strategy;

5. Buyer_Strategy = Best_LocalCA(Buyer)_Strategy;

6. A seller is selected at random from seller LocalCA;

7. A Buyer is selected at random from buyer LocalCA;

if *8. Seller_Strategy = H* **then**

    9. Best agent of seller LocalCA is migrated;

else

    if *10. Seller_Strategy = M* **then**

        11. Average agent of seller LocalCA is migrated;

    else

        12. Worst agent of seller LocalCA is migrated;

    end

end

13. Buyer returns an agent back to seller localCA by using Buyer_Strategy and follows same method of migration as seller;

Algorithm 3: Oligopoly Migration Strategy

Duopoly

In this strategy, the subpopulations are arranged according to their average fitness similar to the oligopoly strategy. The best two performing subpopulations are

selected to migrate an agent. Both the subpopulations will choose an agent to migrate. The destination of the migrating agent is selected randomly from the rest of the subpopulations. The best performing subpopulation will migrate its selected individual to any other subpopulation, and in return, it will take the best performing individual of that subpopulation. The same process is followed by the second best-performing subpopulation for migration. Below is the pseudo-code for the duopoly.

Input : Subpopulations;

Output : Agent to migrate;

---

1. *Arrange subpopulations according to Avg_Fitness;*
2. *Pick the best two subpopulations for migration;*
3. *Both of them will choose an agent to migrate;*
4. *Best subpopulation will migrate an agent randomly to any other subpopulation;*
5. *In return the best subpopulation will take the best individual from the destination subpopulation;*
6. *Second best sub-population will follow the same strategy as best subpopulation;*

Algorithm 4: Duopoly Migration Strategy

Fair Division

As the name suggests this strategy focuses more on fair migration technique where the better performing subpopulations migrate their individuals to the subpopulations performing below average. In this way, all the subpopulations have a fair chance of improving their fitness. In this strategy, the subpopulations are sorted according to their average fitness in descending order. If there are n subpopulations, then they are divided into two halves. The first "n/2" will migrate

their agents to the other "n/2 " subpopulations. The migration takes place in an organized manner where the best performing subpopulation migrates its agent to the worst performing subpopulation, the second best to the second worst, the third best to the third worst and so on. The best individuals of the subpopulation are migrated. This process makes sure that the subpopulation which performs weakest gets the best agent to help them to increase their fitness. Also, the one who is performing a bit less than average gets an agent from the subpopulation which is performing a bit more than average. This way can allow every subpopulation to improve their fitness considering every subpopulation equal and can eliminate the dominating factor from the subpopulations. The exploration rate can improve, and the diversity can also be maintained as two vastly different subpopulations need to cooperate with each other. The strategy focuses on cooperate nature of a society where everyone not only thinks about themselves, instead think for the whole of the society. Below is the pseudo-code for the fair division.

Input : Subpopulations;

Output : Agent to migrate;

---

1. *Arrange subpopulations according to Avg_Fitness;*
2. *Migrate best agent from best-performing subpopulation to worst-performing subpopulation;*
3. *Migrate best agent from second best subpopulation to second worst subpopulation;*
4. *Migrate best agent from third best subpopulation to third worst subpopulation;*
5. *And so on;*

Algorithm 5: Fair Division Strategy

Intra household Bargaining

In this strategy, the best subpopulation migrates an agent to a randomly selected subpopulation. The agents have a strategy initialized initially (cooperate or defect). The decision of each agent is taken into account. If more individuals have cooperate (C) strategy than a defect (D), then the subpopulation follows a C strategy to migrate or else vice versa. If they cooperate than the best individual of the subpopulation is migrated to the randomly selected subpopulation. If the subpopulation takes the decision to defect than the agent is selected randomly and the agent strategy is used to migrate (C or D). If the individual has the strategy to cooperate than the agent is migrated to the least performing subpopulation. If the individual has the strategy to defect than the individual will migrate to the second best-performing subpopulation. If the number of individuals for cooperating and defect are same, then the subpopulation follows the same strategy of migration as they follow for the defect. Below is the pseudo-code for intra-household bargaining.

Input : Subpopulations;

Output : Agent to migrate;

---

1. Best performing subpopulation is choose to migrate an agent;

2. Every individual will be assigned a strategy (cooperate or defect);

if *3. Individuals_cooperate* $\geq$ *Individuals_defect* **then**

    4. Destination subpopulation is selected randomly and best agent is migrated;

else

    5. One random agent is selected from the migrating subpopulation;

    if *6. Individual_strategy is Cooperate* **then**

        7. Agent is migrated to worst-performing subpopulation;

    else

        8. Agent is migrated to second best performing subpopulation;

    end

end

Algorithm 6: Intra Household Bargaining Strategy

Chapter 4

Experiments

In this chapter, we will firstly introduce the benchmark optimization functions used for evaluation of our algorithms. Then we describe the details of the experimental setup. Later we will summarize the results and analyze it.

4.1 Benchmark Optimization Functions

Most commonly used benchmark optimization functions are used to evaluate our algorithm and to compare it with the already existing algorithms. We have used CEC 2015 expensive benchmark functions which contain 15 functions. All the functions used are minimal functions, so we are looking to find the minimum results. Some functions are non-convex, and some are convex. All the test functions are dimension wise scalable. For our experiments, we have used different types of functions like:

1. Unimodal functions
2. Simple multimodal functions
3. Hybrid functions

4. Composite functions

4.1.1 Unimodal Functions

The functions below are extension of the basic functions. Few functions are shifted and rotated.

$$o_{i1} = [o_{i1}, o_{i2}, ..., o_{iD}]^T \tag{4.1}$$

is the shifted global optimum, which is randomly distributed in [-80,80]^D. Each below function has shift data for CEC'15. All the test functions are shifted to o and scalable.

F_1(Rotated Bent Cigar Function): Rotated bent cigar function is extended from the bent cigar function. The function is featured as unimodal, non-separable and dimension-wise scalable. As seen from figure 4.1 [31] it has smooth but narrow bridge.

$$f(x_1 \cdots x_n) = f_1(M(x - o_1)) + 100 \tag{4.2}$$

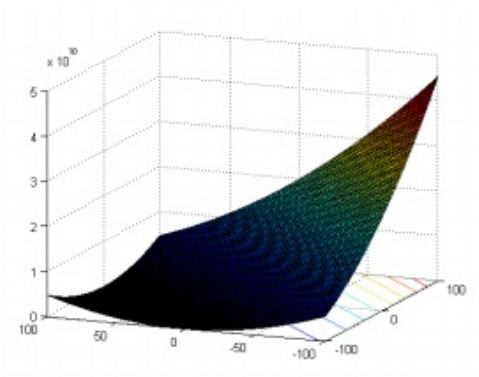

Figure 4.1: 3-D map for Rotated Bent Cigar Function [31]

F_2(Rotated Discus Function): Rotated discus function is extended from discus function. It featured as unimodal, non-separable and dimension-wise scalable. As

depicted in figure 4.2 [31] the function has one sensitive direction.

$$f(x_1 \cdots x_n) = f_2(M(x - o_2)) + 200 \tag{4.3}$$

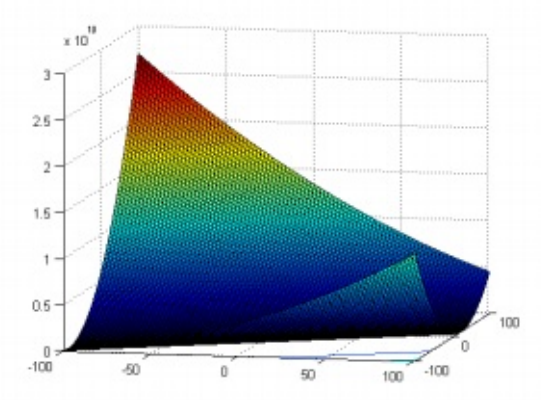

Figure 4.2: 3-D map for Rotated Discus Function [31]

4.1.2 Simple Multimodal Functions

F_3(Shifted and Rotated Weierstrass Function): The shifted and rotated weierstrass function is an extension of weierstrass function. It is featured as multi-modal, non-separable and dimension-wise scalable. As depicted in figure the function is continuous and differentiable only on a set of points.

$$f(x_1 \cdots x_n) = f_3(M(\frac{0.5(x - o_3)}{100})) + 300 \tag{4.4}$$

F_4(Shifted and Rotated Schwefel's Function): The shifted and rotated schwefel's function is extension of schwefel's function. It is featured as multi-modal, non-separable and dimension-wise scalable. As seen from the figure 4.3 [31] the function has a lot of local optima and the second best local optima is far from the global optima.

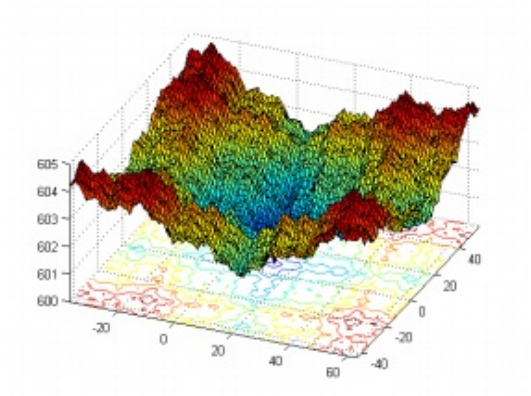

Figure 4.3: 3-D map for Shifted Rotated Schwefel's Function [31]

$$f(x_1 \cdots x_n) = f_4(M(\frac{1000(x - o_4)}{100})) + 400 \tag{4.5}$$

F_5(Shifted and Rotated Katsuura Function): The shifted and rotated katsuura function is an extension of katsuura function. It is featured as multi-modal, non-separable and dimension-wise scalable. It is seen in the figure 4.4 [31] that the function is continuous everywhere and it is not differentiable anywhere.

$$f(x_1 \cdots x_n) = f_5(M(\frac{5(x - o_5)}{100})) + 500 \tag{4.6}$$

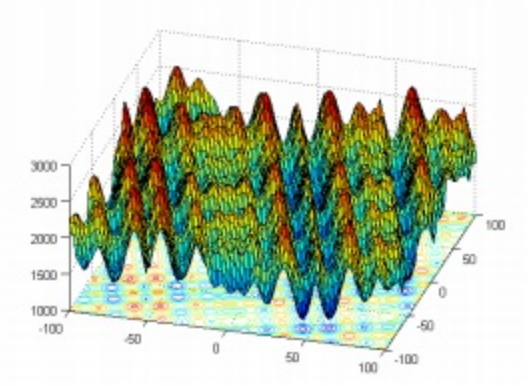

Figure 4.4: 3-D map for Shifted and Rotated Katsuura Function [31]

F_6(Shifted and Rotated HappyCat Function): The shifted and rotated happycat function is an extension of happycat function. It is featured as

mutli-modal, separable and dimension-wise scalable.

$$f(x_1 \cdots x_n) = f_6(M(\frac{5(x - o_6)}{100})) + 600 \tag{4.7}$$

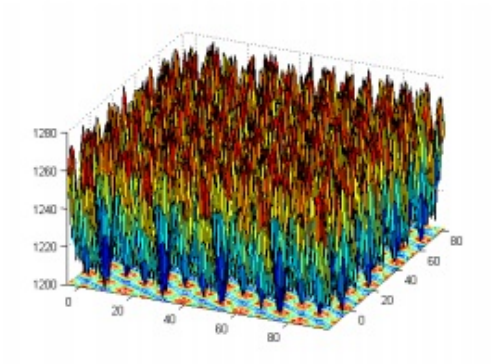

Figure 4.5: 3-D map for Shifted and Rotated HappyCat Function [31]

F_7(Shifted and Rotated HGBat Function): The shifted and rotated HGBat function is an extenion of HGBat function. It is featured as multi-modal, non-separable and dimension-wise scalable.

$$f(x_1 \cdots x_n) = f_7(M(\frac{5(x - o_7)}{100})) + 700 \tag{4.8}$$

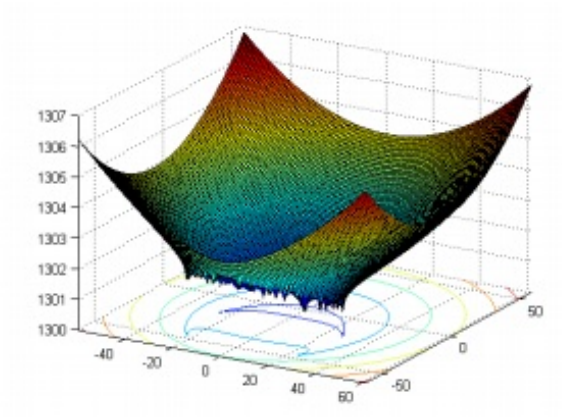

Figure 4.6: 3-D map for Shifted and Rotated HGBat Function [31]

F_8(Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function): The function is an extension and expanded version of two functions :

griewank's and rosenbrock's function. The function is multi-modal, non-separable and dimension-wise scalable.

$$f(x_1 \cdots x_n) = f_8(M(\frac{5(x - o_8)}{100}) + 1) + 800 \tag{4.9}$$

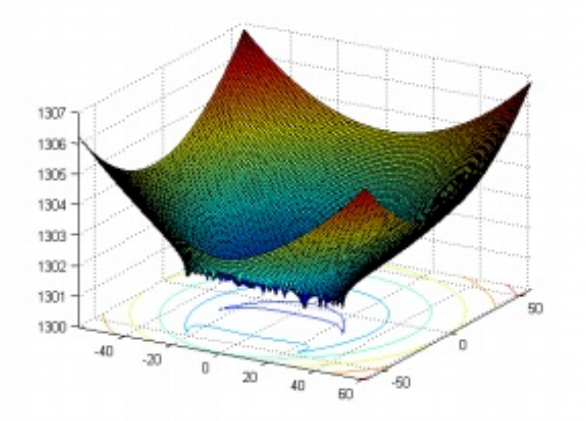

Figure 4.7: 3-D map for Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function [31]

F_9(Shifted and Rotated Expanded Scaffer's F6 Function): Shifted and rotated expanded scaffer's F6 function is an extension of expanded scaffer's F6 function. It is featured as multi-modal, non-separable and dimension-wise scalable.

$$f(x_1 \cdots x_n) = f_9(M(x - o_9) + 1) + 900 \tag{4.10}$$

4.1.3 Hybrid Functions

The hybrid functions are inspired from the real-world optimization problems. As in real-world optimization problems, different subset of variables possess different properties. Similarly in hybrid functions, the variables are divided randomly into some subsets and each subsets will have different basic functions operating on them.

$$F(x) = g_1(M_1 z_1) + g_2(M_2 z_2) + ... + g_N(M_N z_N) + f^*(x) \tag{4.11}$$

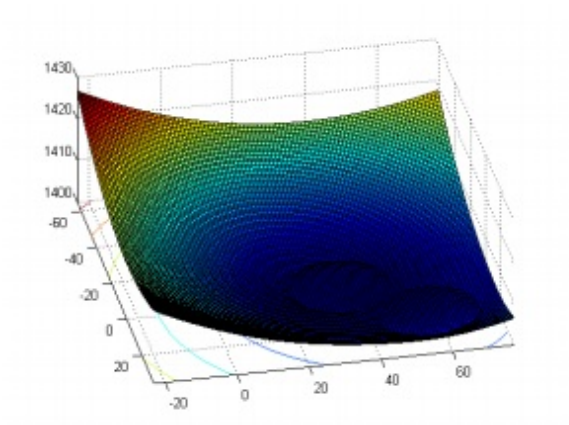

Figure 4.8: 3-D map for Shifted and Rotated Expanded Scaffer's F6 Function [31]

f*(15) = 1000

f*(16) = 1100

f*(17) = 1200

F(x) : hybrid function

$g_1(x)$: i^{th} basic function used to construct the hybrid function

N: number of functions

$$z = [z_1, z_2, ..., z_N], z_1 = [y_{s_1}, y_{s_2}, ..., y_{s_m}], z_2 = [y_{s_m+1}, y_{s_m+2}, ..., y_{s_m+n2}], ..., \\ z_N = [y_{s_{\sum_{i=1}^{N-1} n_i+1}}, y_{s_{\sum_{i=1}^{N-1} n_i+2}}, ..., y_{s_D}] \tag{4.12}$$

where, y= x-o_i and S = $randperm(1{:}D)$

p_i : used to control the percentage of $g_i(x)$

n_i : dimension for each basic function $\sum_{i=1}^{N} n_i = D$

$$n_i = [p_1 D], n_2 = [p_2 D], ..., n_{N-1} = [p_{N-1} D], n_N = D - \sum_{i=1}^{N-1} n_i \tag{4.13}$$

F_{10}(Hybrid Function 1) (N=3)

p=[0.3,0.3,0.4]

g_1 : Modified Schwefel's Function

g_2 : Rastrigin's Function

g_3 : High Conditioned Elliptic Function

F_{11}(Hybrid Function 2) (N=4)

p=[0.2,0.2,0.3,0.3]

g_1 : Griewank's Function

g_2 : Weierstrass Function

g_3 : Rosenbrock's Function

g_4 : Scaffer's F6 Function

F_{12}(Hybrid Function 3) (N=5)

p=[0.1,0.2,0.2,0.2,0.3]

g_1 : Katsuura Function

g_2 : HappyCat Function

g_3 : Expanded Griewank's plus Rosenbrock's Function

g_4 : Modified Schwefel's Function

g_5 : Ackley's Function

4.1.4 Composite Functions

$$F(x) = \sum_{i=1}^{N} \omega_i * [\lambda_i g_i(x) + bias_i] + f^* \tag{4.14}$$

$f^*(18) = 1300$

$f^*(19) = 1400$

$f^*(20) = 1500$

F(x) : composition function

g_i : i^{th} basic function used to construct the composition function

N: number of basic function

o_i : new shifted optimum position for each g_i(x), define the global and local optima's position

$bias_i$: defines which optimum is global optimum

σ_i : used to control each $g_i(x)$'s coverage range, a small σ_i give a narrow range for that $g_i(x)$

λ_i : used to control each $g_i(x)$'s height

W_i : weight value for each $g_i(x)$, calculated as below:

$$W_i = \frac{1}{\sqrt{\sum_{j=1}^{D}}} exp(\frac{-\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D{\sigma_i}^2}) \tag{4.15}$$

Then normalize the weight $\omega_i = w_i \ / \ \sum_{i=1}^{n} w_i$

So when $x = o_i$, $\omega_j =$

$$\left\{ \begin{array}{ll} 1 & j=i \\ 0 & j \neq i \end{array} \right\}$$

for j= 1, 2, ..., N , $f(x) = bias_i + f^*$

The optimum which has the smallest bias value is the global optimum. The composition function merges the properties of the sub-function better and maintains continuity around the global/local optima.

F_{13}(Composition Function 1) (N=5)

N = 5

σ = [10, 20, 30, 40, 50]

λ = [1, 1e-6, 1e-26, 1e-6, 1e-6]

bias = [0, 100, 200, 300, 400]

g_1 : Rotated Rosenbrock's Function

g_2 : High Conditioned Elliptic Function

g_3 : Rotated Bent Cigar Function

g_4 : Rotated Discus Function

g_5 : High Conditioned Elliptic Function

The function is featured as multi-modal, non-separable, asymmetrical and

dimension-wise scalable. The function has different properties around different local optima.

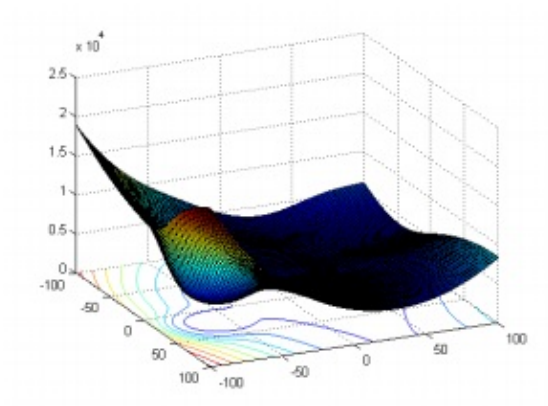

Figure 4.9: 3-D map for Composition Function 1 [31]

F_{14}(Composition Function 2) (N=3)

N = 3

σ = [10, 20, 30]

λ = [0.25, 1, 1e-7]

bias = [0, 100, 200]

g_1 : Rotated Schwefel's Function

g_2 : Rotated Rastrigin's Function

g_3 : Rotated High Conditioned Elliptic Function

The function is featured as multi-modal, non-separable, asymmetrical and dimension-wise scalable. The function has different properties around different local optima.

F_{15}(Composition Function 3) (N=5)

N = 5

σ = [10, 10, 30, 40, 50]

λ = [10, 10, 2.5, 2.5, 1e-6]

bias = [0, 100, 200, 300, 400]

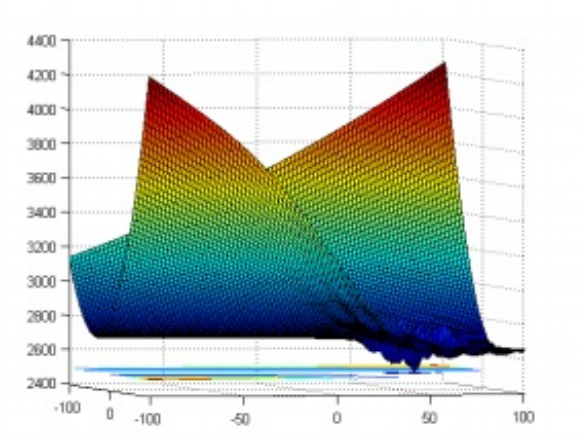

Figure 4.10: 3-D map for Composition Function 2 [31]

g_1 : Rotated HGBat Function

g_2 : Rotated Rastrigin's Function

g_3 : Rotated Schwefel's Function

g_4 : Rotated Weierstrass Function

g_5 : Rotated High Conditioned Elliptic Function

The function is featured as multi-modal, non-separable, asymmetrical and dimension-wise scalable. The function has different properties around different local optima.

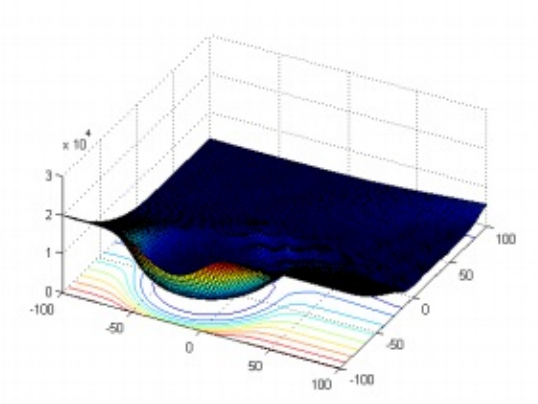

Figure 4.11: 3-D map for Composition Function 3 [31]

4.2 Experimental Setup

We have performed the experiments to compare the performance of MPCA, GA, DE, Heritage Dynamic Cultural Algorithm (HDCA) and MPCA with the proposed migration strategies explained in section 3. All the proposed strategies like Prisoners Dilemma, Oligopoly, Duopoly, Fair Division and Intrahousehold Bargaining has been compared with each other and also with the above mentioned known algorithms. The strategies explained in section 3 are abbreviated as below.

- M1: Genetic Algorithm
- M2: Differential Evolution
- M3: Multi-population Cultural Algorithm
- M4: Multi-population Cultural Algorithm with Random Migration
- M5: Heritage Dynamic Cultural Algorithm
- M6: Multi-population Cultural Algorithm with Prisoners Dilemma
- M7: Multi-population Cultural Algorithm with Oligopoly
- M8: Multi-population Cultural Algorithm with Duopoly
- M9: Multi-population Cultural Algorithm with Fair Division
- M10: Multi-population Cultural Algorithm with Intra-household Bargaining

The 10 algorithms listed above are compared with each. To carry out a fair comparison, the parameters used for execution of all the algorithm are same. The values of the parameters are listed in the table 4.1. All the algorithm are tested 20 times individual on all the fitness functions to get an accurate solution.
The performance of the algorithm was done using the following criteria:

Parameters	Values
Size of population	100
Number of subpopulation	10
Size of subpopulation	10
Maximum number of generations	100
Dimensions	10 &30
Independent run times	20
f: scalar factor in equation 3.2 & 3.3	[0.5,2.5]
C_r: Crossover probability in equation 3.4	0.5

Table 4.1: Parameter values for algorithms

- Mean fitness value (Mean): mean value of the solutions got at the maximum generation in 100 runs.
- Standard deviation (Std.): standard deviation of the mean fitness
- Best individual fitness (Best): best fitness value of the solution in the whole population in all the generations.
- Average number of generations (Gen): average number of generations to find the best solution.

4.3 Results and Analysis

In this section we will compare all the proposed strategies incorporated with MPCA, MPCA incorporating random migration, original MPCA, GA, DE and HDCA. The comparisons are done on both low dimension (10D) and high dimension (30D) on all the benchmark problems mentioned in section 4.1.

Table 4.2: Results for M1 - M10 on F1- F7 for 10D.

	F1	F2	F3	F4	F5	F6	F7
M1							
Mean	1.12E11	6.76E10	3.19E02	5.06E03	5.15E02	6.16E02	1.48E+03
Std.	7.18E09	7.65E09	0.743	2.01E02	0.654	2.077	5.60E+01
Best	1.43E10	3.66E07	3.11E02	1.64E03	5.02E02	6.01E02	7.97E02
Gen	20	16	22	27	13	43	13

Continuation of Table 4.2							
	F1	F2	F3	F4	F5	F6	F7
M2							
Mean	2.49E09	2.45E05	3.10E02	**8.86E02**	**5.02E02**	6.02E02	7.18E02
Std	5.73E08	1.54E05	0.832	1.56E02	0.304	0.472	5.576
Best	7.57E07	3.62E04	3.09E02	5.11E02	5.01E02	**6.00E02**	**7.00E02**
Gen	88	44	6	85	38	77	87
M3							
Mean	**1.44E09**	1.22E05	**3.10E02**	1.22E03	**5.02E02**	**6.01E02**	**7.11E02**
Std	3.15E08	9.47E04	0.712	8.15E01	0.218	0.192	2.97
Best	1.75E06	**1.07E04**	3.06E02	7.40E02	5.01E02	**6.00E02**	**7.00E02**
Gen	83	31	83	69	62	94	74
M4							
Mean	1.49E09	**1.06E05**	**3.10E02**	1.04E03	**5.02E02**	**6.01E02**	7.12E02
Std	3.92E08	7.73E04	1.22E00	7.68E01	0.279	0.186	2.71E00
Best	3.01E06	1.91E04	**3.04E02**	4.51E02	**5.01E02**	**6.00E02**	**7.00E02**
Gen	96	34	47	95	2	63	92
M5							
Mean	6.91E09	3.17E05	3.12E02	1.20E03	5.03E02	6.05E02	7.50E02
Std	2.35E09	2.80E05	0.741	2.19E02	0.503	0.854	1.20E01
Best	**1.32E09**	1.43E04	3.09E02	5.18E02	**5.01E02**	6.02E02	7.12E02
Gen	60	78	32	35	33	94	90

Continuation of Table 4.2							
	F1	F2	F3	F4	F5	F6	F7
M6							
Mean	1.55E09	1.70E05	**3.10E02**	1.07E03	**5.02E02**	6.02E02	**7.11E02**
Std	2.92E08	1.49E05	1.051	9.46E01	0.246	0.229	2.863
Best	3.01E06	3.04E04	3.05E02	4.77E02	**5.01E02**	**6.00E02**	7.00E02
Gen	97	29	59	91	54	93	86
M7							
Mean	1.51E09	1.55E05	3.11E02	1.12E03	**5.02E02**	6.02E02	7.12E02
Std	2.71E08	8.85E04	0.773	9.57E01	0.306	0.221	2.671
Best	4.72E06	3.65E04	3.06E02	4.84E02	**5.01E02**	**6.00E02**	**7.00E02**
Gen	93	55	80	94	71	90	87
M8							
Mean	1.58E09	1.95E05	**3.10E02**	9.88E02	**5.02E02**	**6.01E02**	**7.11E02**
Std.	4.20E08	1.12E05	0.653	6.36E01	0.314	0.205	3.302
Best	1.92E06	3.47E04	3.07E02	**4.24E02**	**5.01E02**	**6.00E02**	**7.00E02**
Gen	97	11	67	96	59	84	84
M9							
Mean	1.67E09	1.78E05	**3.10E02**	1.24E03	**5.02E02**	6.02E02	7.14E02
Std.	2.73E08	1.46E05	0.876	1.07E02	0.287	0.206	4.294
Best	5.58E07	2.85E04	3.08E02	6.49E02	**5.01E02**	**6.00E02**	**7.00E02**
Gen	77	19	62	86	8	59	73

Continuation of Table 4.2							
	F1	F2	F3	F4	F5	F6	F7
M10							
Mean	**1.44E09**	1.31E05	**3.10E02**	1.10E03	**5.02E02**	**6.01E02**	7.12E02
Std.	2.62E08	9.73E04	0.838	7.34E01	0.262	0.252	2.947
Best	2.09E06	3.31E04	3.05E02	4.37E02	**5.01E02**	**6.00E02**	**7.00E02**
Gen	97	77	73	95	52	90	84

Table 4.3: Results for M1 - M10 on F8- F15 for 10D.

	F8	F9	F10	F11	F12	F13	F14	F15
M1								
Mean	7.79E09	9.05E02	4.83E10	4.70E03	1.88E10	2.46E04	1.87E04	6.43E04
Std	2.43E09	0.029	4.63E09	2.63E02	1.07E10	1.62E03	1.16E04	4.28E03
Best	8.66E04	9.04E02	5.77E07	1.43E03	1.97E03	2.60E03	8.31E03	2.00E04
Gen	19	23	29	14	24	9	1	1
M2								
Mean	1.24E04	**9.04E02**	2.12E06	**5.70E02**	**1.40E03**	1.70E03	1.94E03	4.21E03
Std	1.19E04	0.183	1.31E06	1.58E02	7.09E01	2.66E01	6.83E01	6.34E02
Best	**8.03E02**	**9.03E02**	5.84E04	**2.66E01**	1.29E03	1.63E03	1.64E03	**1.50E03**
Gen	72	25	44	95	32	66	43	96
M3								
Mean	4.83E03	**9.04E02**	1.01E06	1.11E03	1.48E03	**1.66E03**	**1.61E03**	1.98E03
Std	3.67E03	0.113	4.11E05	2.735	5.73E01	1.53E01	1.61E00	1.12E02
Best	8.04E02	9.04E02	**1.23E04**	1.10E03	1.29E03	1.62E03	**1.60E03**	1.56E03
Gen	80	75	81	50	69	97	93	57

Continuation of Table 4.3								
	F8	F9	F10	F11	F12	F13	F14	F15
M4								
Mean	5.55E03	**9.04E02**	1.13E06	1.11E03	1.51E03	**1.66E03**	**1.61E03**	**1.97E03**
Std	4.26E03	0.170	6.90E05	2.093	6.47E01	1.30E01	1.887	1.01E02
Best	**8.03E02**	9.04E02	4.99E04	1.10E03	1.29E03	1.62E03	**1.60E03**	1.58E03
Gen	89	18	66	75	78	95	97	89
M5								
Mean	1.62E04	**9.04E04**	3.66E06	8.14E02	1.43E03	1.79E03	7.33E03	9.28E03
Std	1.48E04	0.110	1.77E06	1.77E02	1.10E02	6.04E01	2.30E02	1.43E03
Best	8.39E04	9.03E02	1.82E04	2.50E02	1.25E03	1.64E03	6.38E03	4.30E03
Gen	58	65	27	32	23	70	82	24
M6								
Mean	6.07E03	**9.04E02**	1.06E06	1.11E03	1.48E03	**1.66E03**	**1.61E03**	2.02E03
Std	4.66E03	0.101	5.72E05	1.901	8.51E01	1.67E01	2.502	5.12E01
Best	**8.03E02**	9.04E02	8.49E04	1.10E03	**1.24E03**	**1.61E03**	**1.60E03**	1.93E03
Gen	80	79	86	86	61	97	93	69
M7								
Mean	4.94E03	**9.04E02**	1.08E06	1.11E03	1.48E03	**1.66E03**	**1.61E03**	1.98E03
Std	2.83E03	0.113	5.90E05	3.268	5.45E01	1.47E01	2.059	7.58E01
Best	8.04E02	9.04E02	4.49E04	1.10E03	1.27E03	**1.61E03**	**1.60E03**	1.65E03
Gen	92	57	69	95	86	93	91	62

Continuation of Table 4.3								
	F8	F9	F10	F11	F12	F13	F14	F15
M8								
Mean	**3.56E03**	**9.04E02**	1.10E06	1.11E03	1.50E03	**1.66E03**	**1.61E03**	2.02E03
Std	2.11E03	0.101	5.11E05	2.254	7.51E01	1.54E01	1.963	4.87E01
Best	**8.03E02**	9.04E02	1.71E04	1.10E03	1.26E03	**1.61E03**	**1.60E03**	1.92E03
Gen	94	69	49	91	68	93	85	45
M9								
Mean	5.60E03	**9.04E02**	1.08E06	1.11E03	1.49E03	1.67E03	**1.61E03**	2.00E03
Std	3.81E03	0.115	4.12E05	3.231	6.03E01	1.57E01	2.741	6.97E01
Best	8.04E02	9.04E02	7.98E04	1.10E03	1.26E03	1.62E03	**1.60E03**	1.66E03
Gen	63	75	55	82	60	83	75	20
M10								
Mean	5.55E03	**9.04E02**	**9.12E05**	1.11E03	1.49E03	**1.66E03**	**1.61E03**	1.99E03
Std	3.21E03	0.239	3.65E05	2.443	6.53E01	1.30E01	2.138	8.15E01
Best	**8.03E02**	**9.03E02**	3.29E04	1.10E03	1.28E03	1.62E03	**1.60E03**	1.67E03
Gen	88	24	71	93	92	94	81	89

As seen from Table 4.2 and 4.3 the proposed migration strategies when compared with the other existing algorithms gives similar kind of results. On functions like F5, F6, F7, F9 and F14 the results for the best solution of existing algorithms are similar when compared to our migration strategies, but the number of generations seized by proposed strategies to reach the best solution is lesser. The proposed strategies have performed better on Function 4 which is a multi-modal function, function 12 and function 13 which are hybrid and composition function respectively. On all other functions, the results are equivalent except the unimodal functions. The table 4.2 and 4.3 depicts that the proposed migration strategies have good

results on few complex problems and apart from that does not have acute instead, equivalent or similar kind of results. The proposed migration strategies take less number of generations to get the best solution whereas other existing algorithms gets similar results but takes more generations. M9 is best migration strategy which works on 10-dimensional problems when compared with other migration strategies, due to its fair migration nature where every subpopulation holds the chance to improve its fitness.

In Table 4.4 and 4.5, the algorithms are evaluated on 30-dimensional problems which are more complex in nature than 10-dimensional problems.

Table 4.4: Results for M1 - M10 on F1- F7 for 30D.

	F1	F2	F3	F4	F5	F6	F7
M1							
Mean	3.12E11	7.67E10	3.59E02	1.23E04	5.15E02	6.11E02	1.28E03
Std.	1.05E10	1.29E10	1.359	5.32E02	0.396	0.946	3.55E01
Best	1.03E11	4.69E07	3.45E02	5.81E03	5.04E02	6.02E02	9.35E02
Gen	15	1	28	1	0	60	5
M2							
Mean	5.35E10	3.15E05	3.44E02	**4.82E03**	**5.04E02**	6.06E02	8.28E02
Std.	8.02E09	1.46E05	0.870	3.57E02	0.407	0.391	2.54E01
Best	1.38E10	1.43E05	3.41E02	3.77E03	5.03E02	6.03E02	7.31E02
Gen	92	0	64	77	25	84	91
M3							
Mean	3.18E10	**2.16E05**	**3.43E02**	5.57E03	**5.04E02**	6.04E02	7.74E02
Std.	4.99E09	7.50E04	1.277	2.53E02	0.457	6.342	1.10E01
Best	2.60E09	1.01E05	**3.36E02**	**2.29E03**	**5.02E02**	**6.01E02**	7.04E02
Gen	98	31	87	98	36	97	96

Continuation of Table 4.4							
	F1	F2	F3	F4	F5	F6	F7
M4							
Mean	2.83E10	2.35E05	**3.43E02**	5.33E03	**5.04E02**	6.04E02	7.68E02
Std.	2.84E09	8.91E04	1.247	3.09E02	0.330	0.350	6.923
Best	2.60E09	1.28E05	**3.36E02**	2.29E03	**5.02E02**	**6.01E02**	7.04E02
Gen	96	29	47	96	43	92	97
M5							
Mean	7.79E10	2.78E05	3.45E02	5.57E03	5.05E02	6.07E02	8.82E02
Std.	7.34E09	1.59E05	1.206	2.69E02	0.632	0.461	1.57E01
Best	4.27E10	**7.83E04**	3.38E02	3.56E03	5.03E02	6.05E02	8.00E02
Gen	39	23	66	41	64	21	73
M6							
Mean	3.01E10	2.67E05	**3.43E02**	5.28E03	**5.04E02**	6.04E02	7.70E02
Std.	4.07E09	1.31E05	1.458	3.24E02	0.402	0.504	7.554
Best	**1.99E09**	1.05E05	3.39E02	2.57E03	5.03E02	**6.01E02**	7.04E02
Gen	97	2	42	95	30	87	96
M7							
Mean	3.00E10	2.43E05	**3.43E02**	5.56E03	**5.04E02**	6.04E02	7.70E02
Std.	4.49E09	9.78E04	1.622	2.64E02	0.367	0.443	1.13E01
Best	3.65E09	1.19E05	**3.36E02**	2.59E03	**5.02E02**	**6.01E02**	7.05E02
Gen	97	0	71	98	44	93	98

Continuation of Table 4.4							
	F1	F2	F3	F4	F5	F6	F7
M8							
Mean	**2.65E10**	2.41E05	**3.43E02**	5.20E03	**5.04E02**	**6.03E02**	**7.64E02**
Std.	3.70E09	1.21E05	1.355	2.59E02	0.418	0.379	9.391
Best	2.05E09	1.25E05	3.37E02	2.03E03	**5.02E02**	**6.01E02**	**7.02E02**
Gen	97	21	38	99	32	95	94
M9							
Mean	3.27E10	2.45E05	**3.43E02**	5.83E03	**5.04E02**	6.04E02	7.80E02
Std.	4.47E09	1.17E05	1.544	3.91E02	0.343	0.444	7.866
Best	7.59E09	9.68E04	**3.36E02**	3.35E03	5.03E02	**6.01E02**	7.15E02
Gen	98	2	52	81	60	97	97
M10							
Mean	2.95E10	2.29E05	**3.43E02**	5.44E03	**5.04E02**	6.04E02	7.70E02
Std.	3.49E09	7.54E04	1.546	2.75E02	0.374	0.351	8.931
Best	3.01E09	1.50E05	**3.36E02**	**1.83E03**	5.03E02	**6.01E02**	7.07E02
Gen	97	37	72	99	72	91	96

Table 4.5: Results for M1 - M10 on F8- F15 for 30D.

	F8	F9	F10	F11	F12	F13	F14	F15
M1								
Mean	1.11E12	9.15E02	7.54E10	1.20E04	7.51E11	4.11E04	8.22E04	1.60E05
Std	4.67E11	0.020	8.84E09	6.37E02	3.58E11	3.52E03	4.70E03	8.58E03
Best	1.78E08	9.14E02	6.62E08	5.48E03	1.07E06	4.27E03	2.88E04	7.28E04
Gen	11	31	12	4	4	9	2	5

Continuation of Table 4.5								
	F8	F9	F10	F11	F12	F13	F14	F15
M2								
Mean	4.96E07	**9.14E02**	1.35E08	4.41E03	1.61E04	2.55E03	5.83E03	4.23E04
Std	2.51E07	0.107	4.60E07	3.76E02	1.87E04	2.00E02	7.39E02	5.47E03
Best	2.54E06	9.14E02	4.08E07	3.24E03	2.68E03	1.84E03	3.51E03	1.12E04
Gen	88	20	14	93	29	95	80	92
M3								
Mean	1.41E07	9.15E02	7.06E07	**1.33E03**	9.48E03	2.12E03	**1.78E03**	**2.91E03**
Std	6.34E06	0.155	1.98E07	4.52E01	7.65E03	9.97E01	2.71E01	2.15E01
Best	1.01E04	**9.13E02**	**9.58E06**	**1.14E03**	1.94E03	**1.68E03**	1.65E03	2.81E03
Gen	98	22	76	96	75	97	95	39
M4								
Mean	1.54E07	**9.14E02**	7.28E07	**1.33E03**	9.25E03	**2.06E03**	1.79E03	2.93E03
Std	9.03E06	0.182	2.40E07	3.78E01	9.68E03	6.28E01	2.03E01	3.33E01
Best	1.01E04	**9.13E02**	9.74E06	**1.14E03**	2.05E03	**1.68E03**	1.68E03	2.81E03
Gen	98	29	89	96	88	93	94	35
M5								
Mean	7.47E07	**9.14E02**	1.65E08	5.16E03	2.36E04	3.09E03	1.96E04	5.62E04
Std	2.56E07	0.093	5.42E07	3.66E02	1.95E04	1.88E02	1.26E03	5.01E03
Best	6.91E06	**9.13E02**	2.76E07	3.47E03	2.06E03	2.07E03	1.38E04	3.53E04
Gen	37	37	64	69	41	91	36	14

Continuation of Table 4.5								
	F8	F9	F10	F11	F12	F13	F14	F15
M6								
Mean	**1.26E07**	**9.14E02**	**6.47E07**	1.35E03	**5.64E03**	2.13E03	1.79E03	2.94E03
Std	4.94E06	0.119	2.38E07	3.83E01	3.45E03	9.22E01	3.91E01	3.01E01
Best	**2.10E03**	9.14E02	**1.50E06**	1.15E03	2.05E03	1.69E03	**1.63E03**	**2.78E03**
Gen	91	37	81	98	69	90	95	50
M7								
Mean	1.68E07	**9.14E02**	6.94E07	1.37E03	9.27E03	2.12E03	1.79E03	2.94E03
Std	8.31E06	0.072	1.93E07	5.54E01	6.92E03	9.20E01	2.63E01	2.91E01
Best	4.04E04	9.14E02	**8.78E06**	1.15E03	**1.81E03**	1.73E03	1.66E03	2.84E03
Gen	93	53	82	97	68	98	92	82
M8								
Mean	1.57E07	**9.14E02**	8.06E07	1.36E03	5.81E03	2.09E03	1.79E03	2.94E03
Std	6.13E06	0.141	2.23E07	4.45E01	3.33E03	7.65E01	3.46E01	2.14E01
Best	3.21E03	9.14E02	1.76E07	1.15E03	2.08E03	1.69E03	**1.64E03**	2.83E03
Gen	96	7	66	99	92	98	91	44
M9								
Mean	1.78E07	**9.14E02**	7.52E07	1.36E03	8.68E03	2.16E03	1.79E03	2.95E03
Std	8.50E06	0.108	1.92E07	5.84E01	9.00E03	1.02E02	3.34E01	3.18E01
Best	7.90E04	**9.13E02**	1.14E07	1.16E03	2.11E03	1.77E03	1.65E03	2.82E03
Gen	98	63	63	78	75	92	85	22

Continuation of Table 4.5								
	F8	F9	F10	F11	F12	F13	F14	F15
M10								
Mean	1.53E07	**9.14E02**	8.29E07	1.36E03	6.19E03	2.07E03	1.79E03	2.93E03
Std	5.99E06	0.119	1.65E07	4.63E01	3.34E03	9.01E01	3.28E01	3.22E01
Best	2.52E03	9.14E02	1.95E07	**1.14E03**	2.21E03	1.69E03	1.66E03	**2.80E03**
Gen	98	11	74	96	70	98	84	67

As seen from the results of Table 4.4 and 4.5 we can easily notice the proposed strategies especially prisoners dilemma, in particular, performs better. The proposed strategies have better results on most of the functions whether it is unimodal, multi-modal, hybrid or composition function problem. The functions on which the results are not better does not even have disappointing results. The results are equivalent or similar to the other functions. The number of generations taken by the proposed algorithms is less than other algorithms on function F5 and F6.

From analyzing the table 4.2, 4.3, 4.4 and 4.5, it can be easily seen the proposed strategies for migration are good for solving the complex problems than the simple one. The strategies show good results when the dimensions are high. Even the results for the 10-dimensional problem are equivalent of the proposed and existing algorithms, but the number of generations taken by proposed strategies are less for many functions. The major aspect to focus is when the proposed strategies are performing better than the other algorithms they show very good results and beat the existing algorithm results by a good margin. This shows us that the proposed migration strategies are better when compared with the other already existing algorithm for optimizing of complex problems. Besides this the migration strategies also searches in large space and maintain diversity which we will discuss in more detail in the next chapter. The proposed migration strategies are even versatile as

they solve most of the problems either better or similar to the other algorithms, while the other algorithms perform well on few functions and fail to have good results on many other functions.

The execution time to evaluate the migration strategies on above functions have also been noted and it takes about 1.20 sec to run F1, 1.10 sec for F2, 7.34 sec for F3, 0.89 sec for F4, 1.43 sec for F5, 2.10 for F6, 0.76 for F7, 1.00 sec for F8, 1.32 sec for F9, 2.01 sec for F10, 2.11 sec for F11, 2.11 sec for F12, 3.10 for F13, 2.89 sec for F14 and 2.00 sec for F15. The time noted is time taken by single run of algorithm on optimization function.

4.4 Migration Strategy Analysis

In the previous section, we have compared different migration strategies with the existing algorithms on various benchmark functions. The benchmark functions included many different categories of problems like unimodal, multi-modal, hybrid and composition. These different categories of functions can help us to decide which migration strategy can be used when we need to solve the particular type of problem. Table 4.6 describes which will be the best migration strategy to use when we have to solve a certain type of problem.

Problem Category	Migration Strategy
Unimodal	Prisoner's Dilemma
Multi- Modal	Prisoner's Dilemma, Oligopoly, Duopoly, Fair Division, Intra-household Bargaining
Hybrid	None
Composition	Prisoner's Dilemma, Oligopoly, Duopoly

Table 4.6: Comparison of problem category against Migration strategy

From the table 4.6 we can see when unimodal functions are to solved prisoner's dilemma works best compared to all the strategies. On multi-modal functions, all

the strategies show real potential to solve the problems. On hybrid functions, none of the strategies have provided real evidence to use them to get better results. While on composition functions prisoner's dilemma, oligopoly, and duopoly show good potential to solve them.

Chapter 5

Discussions, Comparisons and Analysis

In this section, we will discuss different migration strategies in the context of their characteristics. We will discuss the role of migration for particular strategy and the convergence speed to reach the near optimal solution.

5.1 Comparison between M3, M4 and M6

Firstly we will compare M3 Vs M4 Vs M6. M6 is prisoner's dilemma strategy, and we will compare it against the other two better performing algorithm MPCA and MPCA with random migration on the 30-dimensional problem. We will compare them against the fitness function 8.

The figure 5.1 demonstrates the performance of M6 with M3 and M4 on function F8, regarding fitness value against a number of generations. M6 convergence quickly towards a better solution than other algorithms. The drop in the fitness value of M6 demonstrates that when an agent is migrated after the number of generations the overall fitness of the population and also of the agents improves. After every 10 generations migration is carried out and we can see from the figure that after every

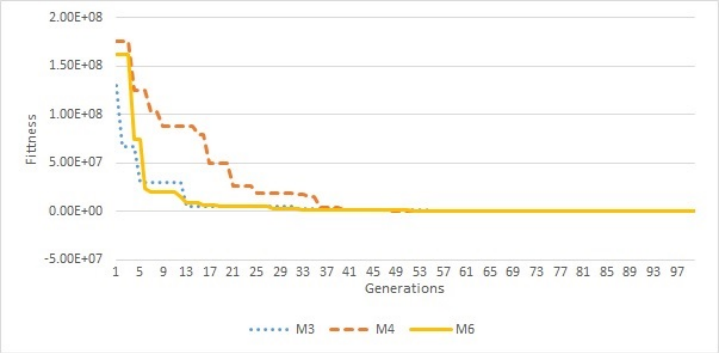

Figure 5.1: Convergence performance of M3, M4 and M6 for F8 (30D)

10 generations the fitness is improving quickly. While for the other algorithms like M3 and M4 it takes a longer number of generations to reach a good solution. As there is no migration in M3, the agent takes more time to reach better space in the search region. While M5 needs less generation than M3 because, it incorporates migration, but it is random and takes more time for the agent to find a better region. The migration strategy, prisoner's dilemma works better on the multi-modal function as the migration carried out by it tends to solve the multi-objective problems in a better way. Function F8 has a large number of local optima spread around the search region. M6 performs better as the migration in it allows the better performing agent to take a decision to cooperate or defect and the agent migrates to the population which requires that agent to guide them to find better search space and escaping from local optima.

5.2 Comparison between M2, M5 and M7

The figure 5.2 demonstrates the comparison between M2, M5, and M7 on function F10 for 30 dimensions. The function is a hybrid in nature and has many local minima. It is a complex, non-convex and non-separable function. It is difficult for the agents in the population to escape from local optima and explore new search

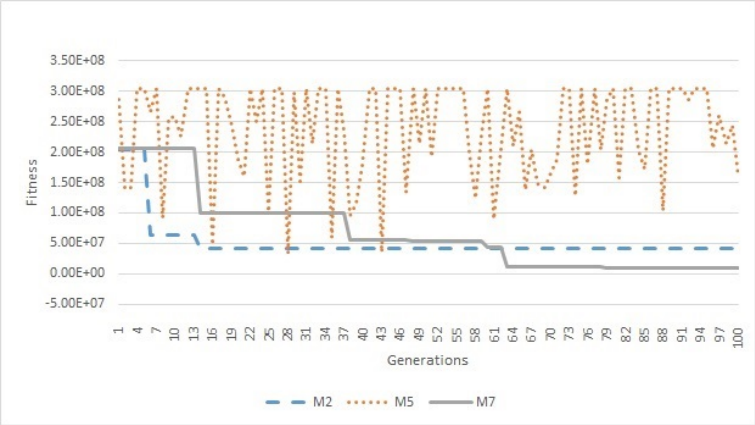

Figure 5.2: Convergence performance of M2, M5 and M7 for F10 (30D)

space with a good potential solution. This is the reason for the algorithm to have the same solution over generations. M2 quickly gets a good solution and then gets stuck into it. It falls into local optima and follows the same solution over the generations. While HDCA gets a good solution and then again gets into global minima due to the heritage knowledge it follows. The knowledge of the past generation inherited by the future generation makes them fall into global minima. The graph of M7 shows it takes the agents some time to find the proper subpopulation for migrating an agent. After every migration, the fitness of the population gets better as the migration is done by using the oligopoly strategy. The migration is done by the dominating individuals of the population which allow the whole population to make better decisions and escape from local optima.

5.3 Comparison between M2, M4 and M8

The figure 5.3 demonstrates the comparison between M2, M4, and M8 on fitness function F7 for 30 dimensions. All the algorithms follow similar kind of pattern except M1 which is GA. GA gets a good solution but in the next generation, the offspring's need to start over again to find a better solution in the search space. While M3 and M8 follow a similar pattern for over the generations but after many

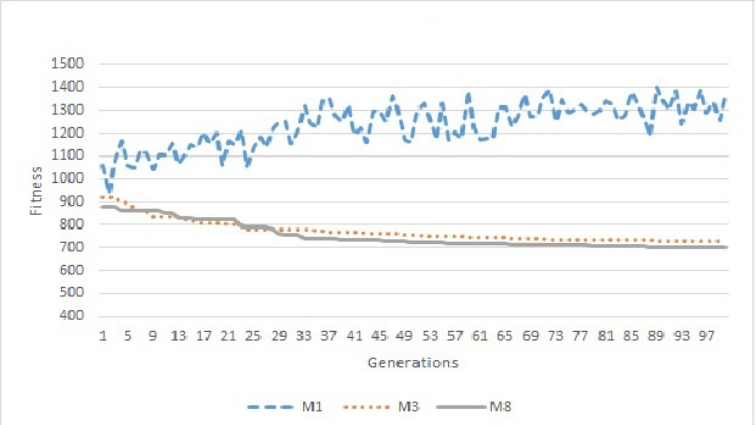

Figure 5.3: Convergence performance of M1, M3 and M8 for F7 (30D)

generations, M3 falls into local optima, but M8 continues to find a better solution in the search space. Duopoly strategy allows finding a better solution as dominating populations try to explore the search space for their better fitness, and this also benefits the other population. Migration of better individuals to not so good performing population helps that population to move towards better search space.

5.4 Comparison between M2, M4 and M9

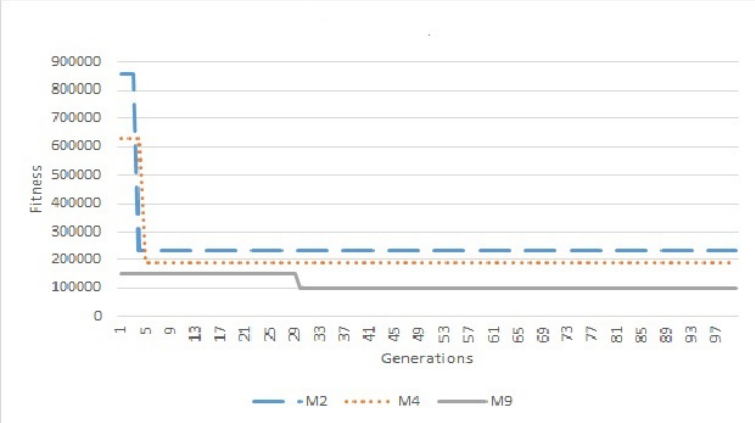

Figure 5.4: Convergence performance of M2, M4 and M9 for F2 (30D)

The figure 5.4 demonstrates the comparison between M2, M4, and M8 on fitness function F2 for 30 dimensions. It is seen from the graph M9 has the best solution

when compared to the other algorithm. The function is non-separable with one sensitive direction, so it is tough for the algorithm to optimize it and converge it quickly. The algorithm falls into local optima and cannot escape from it. M9 has a better solution as migration in fair division allows all the populations to migrate in a fair way by cooperating with each other. The cooperation allows the agents to search the space with good potential and explore it.

5.5 Comparison between M3, M4 and M10

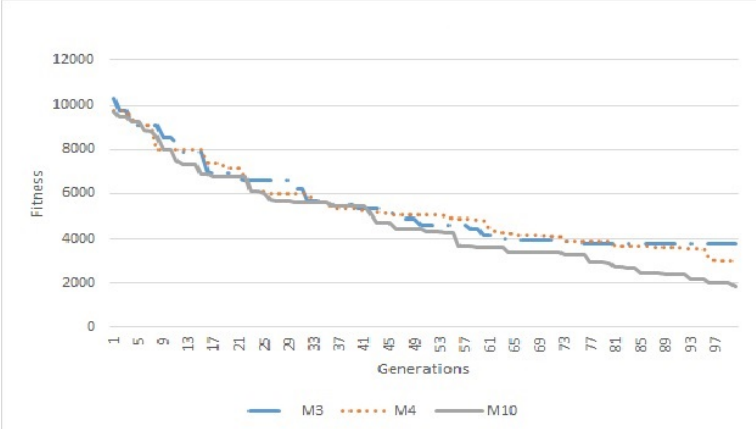

Figure 5.5: Convergence performance of M3, M4 and M10 for F4 (30D)

The figure 5.5 demonstrates the comparison between M3, M4, and M10 on fitness function F4 for 30 dimensions. All the algorithms almost follows similar kind of pattern in optimizing the function. While M3 convergences at 60 generations and fall into local optima. While M4 converges gradually but it is not able to reach the near optimal solution. M10 has the best result as the migration helps the agents to explore the unsearched space with a good solution. The convergence speed is not high as the population also possess diversity but avoiding to just move towards the best solution. The intro-household migration technique allows all the individuals in the subpopulation to take equal part in migration decision. This migration method helps the algorithm to search for new spaces even after every other algorithm

reaches to a near optimal solution and falls into premature convergence.

Chapter 6

Conclusions

Our primary focus is to show the impact of migration on different populations in MPCA. For witnessing the impact of migration, we have carried out the migration in a strategical way rather than doing it in a random manner. We have introduced five different migration strategies which are inspired by the game theory model. The strategies are selected from the economics background as it brings in the social factor among the individuals in the population. DE is used as the evolutionary algorithm for the evolution of the population as DE is good for better exploration of the search space. We have used CEC 2015 expensive benchmark problems to evaluate the performance of our algorithm and compared them with the existing algorithms. The results depict that the proposed strategies perform better with higher dimension than the lower dimension problems. The results of smaller dimension problems are not acute, instead are similar or equivalent. The results also show that when migration of individual is done from one population to another, both the populations have a good impact due to it. Migration allows the population to converge quickly and also maintain diversity. Graphs depict when an individual has migrated the fitness of the destination population has improved drastically. Prisoners Dilemma, Oligopoly, Duopoly, Fair Division and Intrahousehold

Bargaining are the game theory techniques which are used as migration strategies to improve the performance of MPCA. Strategies not only improve the performance but also in addition to it increases diversity among the population, helps to escape from local optima and avoids premature convergence. Prisoner's Dilemma, in particular, shows the best result among all the strategies when evaluated against complex functions with high dimensions (30D).

In future work, more strategies can be introduced either from the game theory concepts or any other field. Other game theory concepts apart from economics background can also be considered for migration strategy as it can bring in other factors apart from social influence. The new strategies can focus on performing on both high and low dimensional problems. More complex and benchmark functions can be used to evaluate the performance of the migration techniques. The proposed strategy shows good results mainly on the complex problems like multi-modal and composition. This procedure can be used in real world applications like the social networks. The networks of the social networking sites are heterogeneous in nature and complex. By implementing the strategical migration, we can witness the impact and then predict which strategy can work for better optimization with the different type of social network. It can also be used in the field of security where the intentions of the migrating person can be known by using the strategies.

Bibliography

[1] Ashraf M Abdelbar, Sherif Ragab, and Sara Mitri. Co-evolutionary particle swarm optimization applied to the 7/spl times/7 seega game. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 1, pages 243–248. IEEE, 2004.

[2] J Alami, A El Imrani, and A Bouroumi. A multipopulation cultural algorithm using fuzzy clustering. *Applied Soft Computing*, 7(2):506–519, 2007.

[3] Jörg Biethahn and Volker Nissen. *Evolutionary algorithms in management applications*. Springer Science & Business Media, 2012.

[4] Lashon B. Booker, David E. Goldberg, and John H. Holland. Classifier systems and genetic algorithms. *Artificial intelligence*, 40(1-3):235–282, 1989.

[5] Stephen P Boyd and Lieven Vandenberghe. Convex optimization (pdf). *Np: Cambridge UP*, 2004.

[6] Heinrich Braun. On solving travelling salesman problems by genetic algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 129–133. Springer, 1990.

[7] Sergey V Buldyrev, Roni Parshani, Gerald Paul, H Eugene Stanley, and Shlomo Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, 2010.

[8] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.

[9] Zhihua Cui, Xingjuan Cai, Jianchao Zeng, and Guoji Sun. Predicted-velocity particle swarm optimization using game-theoretic approach. In *International Conference on Intelligent Computing*, pages 145–154. Springer, 2006.

[10] Cecilia Di Chio, Paolo Di Chio, and Mario Giacobini. An evolutionary game-theoretical approach to particle swarm optimisation. In *Workshops on Applications of Evolutionary Computation*, pages 575–584. Springer, 2008.

[11] Jason G Digalakis and Konstantinos G Margaritis. A multipopulation cultural algorithm for the electrical generator scheduling problem. *Mathematics and Computers in Simulation*, 60(3):293–301, 2002.

[12] Lester E Dubins and Edwin H Spanier. How to cut a cake fairly. *The American Mathematical Monthly*, 68(1):1–17, 1961.

[13] John Eatwell, Murray Milgate, and Peter Newman. *Game theory*. Springer, 1989.

[14] Agoston E Eiben, James E Smith, et al. *Introduction to evolutionary computing*, volume 53. Springer, 2003.

[15] Nelis Franken and Andries Petrus Engelbrecht. Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma. *IEEE Transactions on Evolutionary Computation*, 9(6):562–579, 2005.

[16] Alex A Freitas. *Data mining and knowledge discovery with evolutionary algorithms*. Springer Science & Business Media, 2013.

[17] Jesús Gómez-Gardenes, Irene Reinares, Alex Arenas, and Luis Mario Floría. Evolution of cooperation in multiplex networks. *Scientific reports*, 2:620, 2012.

[18] Paul Bryant Grosso. Computer simulations of genetic adaptation: Parallel subcomponent interaction in a multilocus model. 1985.

[19] Yi Nan Guo, Yuan Yuan Cao, and Dan Dan Liu. Multi-population multi-objective cultural algorithm. In *Advanced Materials Research*, volume 156, pages 52–55. Trans Tech Publ, 2011.

[20] Yi-nan Guo, Jian Cheng, Yuan-yuan Cao, and Yong Lin. A novel multi-population cultural algorithm adopting knowledge migration. *Soft computing*, 15(5):897–905, 2011.

[21] Katsuki Hayashi, Reiji Suzuki, and Takaya Arita. Coevolution of cooperation and layer selection strategy in multiplex networks. *Games*, 7(4):34, 2016.

[22] Andrew William Hlynka and Ziad Kobti. Knowledge sharing through agent migration with multi-population cultural algorithm. In *FLAIRS Conference*, 2013.

[23] JH Holland. Adaptation in natural and artificial systems. 1 edigao. *Ann Arbor, USA: The University of Michigan Press*, 1975.

[24] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[25] Xidong Jin and Robert G Reynolds. Using knowledge-based evolutionary computation to solve nonlinear constraint optimization problems: a cultural algorithm approach. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3, pages 1672–1678. IEEE, 1999.

[26] Elizabeth Katz. The intra-household economics of voice and exit. *Feminist economics*, 3(3):25–46, 1997.

[27] Ziad Kobti et al. Heterogeneous multi-population cultural algorithm. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 292–299. IEEE, 2013.

[28] Ziad Kobti, RG Reynodls, and Tim Kohler. A multi-agent simulation using cultural algorithms: The effect of culture on the resilience of social systems. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 3, pages 1988–1995. IEEE, 2003.

[29] Ziad Kobti, Anne W Snowdon, Shamual Rahaman, Tina Dunlop, and Robert D Kent. A cultural algorithm to guide driver learning in applying child vehicle safety restraint. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 1111–1118. IEEE, 2006.

[30] John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

[31] JJ Liang, BY Qu, PN Suganthan, and Q Chen. Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization. *Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2014.

[32] Alexandre S Mendes, Felipe M Müller, Paulo M França, and Pablo Moscato. Comparing meta-heuristic approaches for parallel machine scheduling problems. *Production Planning & Control*, 13(2):143–154, 2002.

[33] Efrñn Mezura-Montes, Jesús Velázquez-Reyes, and Carlos A Coello Coello. A comparative study of differential evolution variants for global optimization. In

Proceedings of the 8th annual conference on Genetic and evolutionary computation, pages 485–492. ACM, 2006.

[34] Godfrey Onwubolu and Donald Davendra. Scheduling flow shops using differential evolution algorithm. *European Journal of Operational Research*, 171(2):674–692, 2006.

[35] NG Pavlidis, Kostantinos E Parsopoulos, and Michael N Vrahatis. Computing nash equilibria through computational intelligence methods. *Journal of Computational and Applied Mathematics*, 175(1):113–136, 2005.

[36] Bin Peng. Knowledge and population swarms in cultural algorithms for dynamic environments. 2005.

[37] William Poundstone. *Prisoner's Dilemma/John von Neumann, Game Theory and the Puzzle of the Bomb*. Anchor, 1993.

[38] David Power, Conor Ryan, and R Muhammad Atif Azad. Promoting diversity using migration strategies in distributed genetic algorithms. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1831–1838. IEEE, 2005.

[39] Kenneth Price, Rainer M Storn, and Jouni A Lampinen. *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media, 2006.

[40] Bin Qian, Ling Wang, Rong Hu, Wan-Liang Wang, De-Xian Huang, and Xiong Wang. A hybrid differential evolution method for permutation flow-shop scheduling. *The International Journal of Advanced Manufacturing Technology*, 38(7-8):757–777, 2008.

[41] Robert G Reynolds. An introduction to cultural algorithms. In *Proceedings of the third annual conference on evolutionary programming*, volume 131139. Singapore, 1994.

[42] Robert G Reynolds and SM Saleem. The impact of environmental dynamics on cultural emergence. *Perspectives on Adaptions in Natural and Artificial Systems*, pages 253–280, 2005.

[43] Manisha Sharma, Manjaree Pandit, and Laxmi Srivastava. Reserve constrained multi-area economic dispatch employing differential evolution with time-varying mutation. *International journal of electrical power & energy systems*, 33(3):753–766, 2011.

[44] Hugo Steinhaus. The problem of fair division. *Econometrica*, 16(1), 1948.

[45] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[46] Philip D Straffin. *Game theory and strategy*, volume 36. MAA, 1993.

[47] R Tanese. Distributed genetic algorithms for function optimization, unpublished doctoral dissertation, university of michigan. *Ann Arbor*, 1989.

[48] Santosh Upadhyayula. Dominance in multi-population cultural algorithms. 2015.

[49] Massimiliano Vasile, Edmondo Minisci, and Marco Locatelli. An inflationary differential evolution algorithm for space trajectory optimization. *IEEE Transactions on Evolutionary Computation*, 15(2):267–281, 2011.

[50] Zhen Wang, Attila Szolnoki, and Matjaž Perc. Evolution of public cooperation on interdependent networks: The impact of biased utility functions. *EPL (Europhysics Letters)*, 97(4):48001, 2012.

[51] Rui Zhang and Cheng Wu. A hybrid differential evolution and tree search algorithm for the job shop scheduling problem. *Mathematical Problems in Engineering*, 2011, 2011.

Vita Auctoris

NAME:	Panth Parikh
PLACE OF BIRTH:	Surat, India
YEAR OF BIRTH:	1993
EDUCATION:	UKA Tarsadia University, Surat, India
	Bachelor of Technology, Computer Engineering 2011-2015
	University of Windsor, Windsor ON, Canada
	Master of Science, Computer Science 2016-2017