10-19-2015

# Heuristics for Multi-Population Cultural Algorithm

Xinyu He
*University of Windsor*

# Heuristics for Multi-Population Cultural Algorithm

by

Xinyu He

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

2015

# Heuristics for Multi-Population Cultural Algorithm

by

Xinyu He

APPROVED BY:

---

K. Tepe, External Reader

Department of Electrical and Computer Engineering

---

D. Wu, Internal Reader

School of Computer Science

---

Z. Kobti, Advisor

School of Computer Science

October 15, 2015

# Authors Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyones copyright nor violate any proprietary 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 extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

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

Cultural Algorithm (CA) is one of the Evolutionary Algorithms (EAs) which derives from the cultural evolution process in nature. As an extended version of the CA, the Multi-population Cultural Algorithm (MPCA) has multiple population spaces. Since the evolutionary information can be exchanged among the sub-populations, the MPCA can obtain better results than the CA in optimization problems.

In this thesis, we introduce heuristics to improve the MPCA. The heuristic strategies target the existing weaknesses in MPCAs. Four strategies are developed addressing these weaknesses, including the individual memory heuristic, the social interaction heuristic, the dynamic knowledge migration interval heuristic and the population dispersion based knowledge migration interval heuristic.Five standard benchmark optimization functions with different characteristics are taken to test the efficiency of the heuristics. Simulation results show that each heuristic, to varying degrees, improves the MPCA in convergence speed, stability and precision. We compared different combinations of the strategies, and the results show that the MPCAs with social interaction based knowledge selection, as well as dynamic knowledge migration interval/population dispersion based knowledge migration interval, outperform the other combinations in both low-dimension functions and high-dimension functions.

# DEDICATION

*Dedicated to My Family and Friends*

# ACKNOWLEDGEMENTS

I would like to express my gratitude to all those who helped me during my graduate studies. My deepest gratitude goes first and foremost to my supervisor, Dr. Ziad Kobti for his constant support and guidance all throughout my graduate studies. He has walked me through all the stages of the writing of this thesis. Without his consistent and illuminating instruction, this thesis could not have reached its present form.

My heartfelt gratitude goes to Mrs. Gloria Mensah, secretary to the director, who has always helped setup meetings with my supervisor. In addition, my sincere thanks to Ms. Karen Bourdeau, who has always been there to take care of other issues related to my masters degree.

I am also indebted to NSERC Discovery Grants Program for partially supporting my research.

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

Xinyu He

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# 1 INTRODUCTION

## 1.1 Overview

Optimization problems are crucial in Computer Science. Mathematical approaches often meet failures in solving large-scale optimization problems. In addition to the mathematical optimization, linear programming techniques and dynamic programming techniques were developed to search for optimal solutions. However, these techniques encounter difficulties again in solving the optimization problems with a large number of variables as well as the optimization problems of non-linear objective functions[7]. To overcome these problems, EAs were proposed by researchers. EAs are a class of intelligent optimization algorithms that mimic the metaphor of gene evolution and/or social behaviour of species[3], such as the behaviour of birds foraging and behaviour of ants finding the route to the food source. However, EAs can only get near optimal solutions. CAs are one of the most recent evolutionary algorithms addressing these problems.

The term "culture" was first introduced by Edward B. Tylor in 1881. In his book *Primitive Culture*, he described culture as "that complex whole which includes knowledge, belief, art, morals, customs, and any other capabilities and habits acquired by man as a member of society "[8].

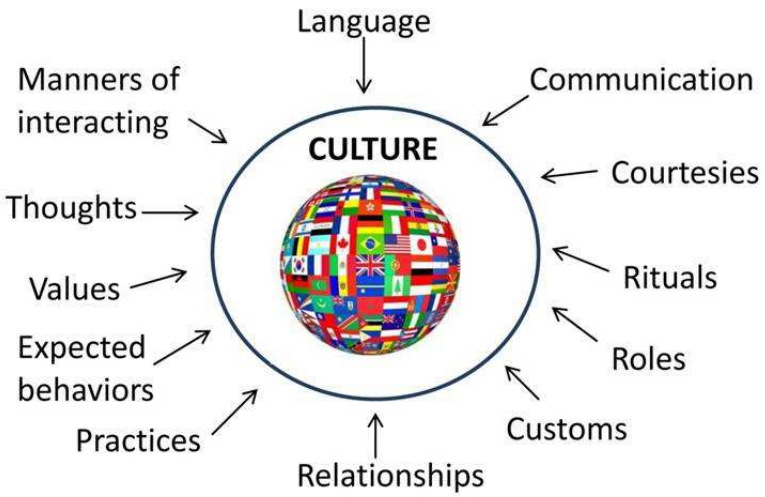


Fig. 1.1: What is culture[2]

Inspired by the cultural evolution process in nature, an evolutionary computational system that can store, accumulate and utilize individuals' experiences during the evolution was proposed by Reynolds in 1994[4]. A CA system is divided into two parts: population space and belief space. The two components evolve respectively and communicate with each other by the communication protocol. The dual inheritance structure makes the CA a self-adaptation system that enables global evolutionary information be more fully utilized.

The MPCA is an extended version of the basic CA. In the standard CAs, the influence function guides the evolution only by the knowledge from a single belief space, which may invalidate the structure of the CA and lead to poor global optimization and instability. Therefore, the MPCA, a CA system with multiple populations was proposed. The MPCA increases the validity of the knowledge by cooperating the implicit knowledge from different populations and at the same time provides the optimal model for the evolution by managing multiple sub-population hierarchically.

## 1.2 Motivation

The CA has been proved a promising EA that can be widely used in many fields such as complex global optimization problems, mechanical design[9], data mining[10][11] and semantic networks[12], etc. As an extension of the CA, the MPCA has better performance in avoiding premature convergence and convergence speed[13], so more and more attention is directed toward the development of MPCAs. At present, MPCAs still have some weaknesses that remain to be improved.

- The first weakness is that only the best solution coming from each sub-population is able to be exchanged with the other sub-populations in terms of the given communication rules. The best solution reflects not enough evolutionary information, thereby decreasing the validity of cooperated knowledge. This probably misleads the whole population converging to the local optima.

- The second one is the random knowledge selection strategy. When the MPCA comes to the phase when the individuals select knowledge sources to evolve the population, each individual selects the same knowledge source, and selects it in a random way. The different categories of knowledge sources always have the diverse influence on the individuals. For example, as the knowledge source used to narrow the feasible search space, the normative knowledge source usually can have more effect on the most individuals in the early stage of the evolution. The topographic knowledge source can affect individuals more in the late time, as it can direct the individuals to explore the search space more precisely. Therefore, a more rational knowledge selection strategy for the individuals in the MPCA need to be developed to improve the efficiency of the MPCA.

- The third weakness is that current MPCAs use the constant knowledge migration interval. Migration and blend of knowledge sources among sub-populations can pace up the convergence of MPCAs. However, migrating knowledge sources

at a too small interval may cause the loss of diversity for the population in the early phase of the evolution. Therefore, current MPCAs require a more reasonable strategy of knowledge migration interval that can manage knowledge sources to be migrated at dynamic interval for different time of the evolution. This weakness is also mentioned in the literature[13].

## 1.3 Problem Statement

Guo et al[13] developed a novel MPCA framework named Multi-Population Cultural Algorithm Adopting Knowledge Migration (MPCA-KM). MPCA-KM provides the approaches to overcoming the first drawback mentioned in Section 1.2. In this thesis, we supplement the coordination mechanism of the normative knowledge source for MPCA-KM and add a historic knowledge source to the algorithm. Then we apply the proposed heuristic strategies to MPCA-KM. Benchmark optimization functions with various properties are used to evaluate how these heuristics can improve the algorithm's efficiency in precision of solutions, stability and convergence speed. The heuristic strategies are described as follows:

- The individuals in the population select the knowledge sources based on social interaction.
- The individuals in the population choose the knowledge sources to based on individual memory.
- The implicit knowledge sources migrate between sub-populations at the dynamic interval.
- The implicit knowledge sources migrate between sub-populations at the interval based on the population dispersion degree.

Our target is to develop an enhanced MPCA by solving the existing weaknesses mentioned in section 1.2. We anticipate these heuristics to improve the MPCA in term of convergence speed, stability and precision.

## 1.4 Organization of Thesis

Chapter 2 contains a review of some typical EAs including Ant Colony Optimization, Genetic Algorithms, and Particle Swarm Optimization. A review of previous works on CAs and MPCAs is presented in Chapter 2. Chapter 3 details the descriptions of the four heuristic strategies developed in this thesis. Chapter 4 produces the test results and analysis of the four strategies and their combinations. Finally the conclusions and suggested future work are presented in Chapter 5.

# 2 LITERATURE REVIEW

## 2.1 Evolutionary Algorithm

EAs are computational approaches that apply the natural biological evolution and/or social behaviour of species to the problems of finding an optimal solution. In EAs, the best solution is yielded from a population of candidate solutions. Part or all of the population individuals (candidate solutions) would have to go through mutation, crossover, selection and reproduction in each iteration of EAs. The operation of mutation or crossover periodically makes changes in individuals of the current population, producing the new individuals. The selection process makes the "most fit" individuals survive, and eliminates the "least fit" members. Consequently, the surviving individuals will participate in the next iteration. As a result of these steps, better solutions will be produced along the iterations. However, EAs rarely reach the optimal solutions because a solution can only be "better " in comparison to the presently known solutions. Some well-known EAs are Genetic Algorithm(GA), Particle Swarm Optimization(PSO) and Ant Colony Optimization(ACO).

### 2.1.1 Genetic Algorithm

GAs originate from the idea of biological natural selection[14]. In GAs, solutions to an optimization problem are represented as "chromosomes ". A chromosome is made up of a set of "genes" that express variables of the optimization problem. GAs start with a random population of chromosomes. For each iteration of the GA, all the chromosomes should be evaluated via the fitness functions. The chromosomes with the best fitness value are selected to operate mutation and crossover so as to yield the offspring chromosomes, and the offspring chromosomes will be taken to compare with the parent chromosomes. Then, for steady versions of GAs, only the winning offspring chromosomes are used to replace the worst parent chromosomes. In contrast,

for unsteady GAs, all the offspring solutions are used to form the population of next generation without being compared with the parent solutions[3]. Usually, the process of GAs is continued until a near-optimal solution is obtained, or a certain number of generations is reached.

The main parameters used in GAs are population size, the maximum number of generations, mutation rate, and crossover rate. Large population size and a large number of maximum generations usually gain the probability to obtain a better solution[3]. A small mutation rate is usually used, and the crossover is given a rate that ranges from 0.6 to 1.0, traditionally[15].

A pseudocode for GAs is shown as follows[15].

**Algorithm 1** Genetic Algorithm[3]

```
1: Generate random population of P solutions (chromosomes);
2: for each individual i ∈ P do
3:    Calculate fitness(i);
4:    for each i = 1 to number of generations; do
5:       Randomly select an operation (crossover or mutation);
6:       if (crossover) then
7:          Select two parents at random i_a and i_b;
8:          Generate an offspring i_c = crossover (i_a and i_b);
9:       else
10:         if (mutation) then
11:            Select one chromosome i at random;
12:            Generate an offspring i_c = mutate (i);
13:         end if
14:      end if
15:      Calculate the fitness of the offspring i_c;
16:      if i_c is better than the worst chromosome then then
17:         replace the worst chromosome by i_c;
18:      end if
19:   end for
20: end for
21: Check if termination = true;
```

### 2.1.2 Particle Swarm Optimization

PSO was first proposed by Kennedy and Eberhart[16]. It is inspired by the social behavior of a flock of birds that try to reach an unknown destination. A bird in the flock is referred to a "particle" that is analogous to a chromosome in GAs. Particles are unable to reproduce offsprings. Instead of reproduction, the particles change their positions to evolve the populations. The particles have both private knowledge and global knowledge which will help the particles approach the desired position (optimal solution). In PSO, the particles can learn from their own experiment (local knowledge) and can communicate with the other particles around them (global knowledge)[3]. The pseudocode for PSO is given as follows[17]:

**Algorithm 2** Particle Swarm Optimization[17]

```
1: Generate random population of N solutions(particles)
2: for each individual i ∈ N do
3:    Calculate fitness(i);
4:    Initialize the value of the weight factor, w;
5:    for each particle do
6:       Set pBest as the best position of particle i;
7:       if fitness (i) is better than pBest; then
8:          pBest(i)=fitness(i);
9:       end if
10:   end for
11:   Set gBest as the best fitness of all particles;
12:   for each each particle; do
13:      Calculate particle velocity according to Equation 4;
14:      Update particle position according to Equation 3;
15:   end for
16:   Update the value of the weight factor, w;
17: end for
18: Check if termination=true;
```

A PSO system is initialized with a swarm of random particles in a S-demension space (S is the number of variables to the optimization problem). Each particle $i$ is assigned with three values: its current position ($X_i$), its velocity ($V_i$) and the best position it has reached ($P_i$) in previous cycles. The position of the best particle in

current cycle ($P_g$) is also known to each particle in the population. Then for each cycle, each particle updates its position according to the following formulas[18]:

$$New\ V_i = w \times current\ V_i + c_1 \times rand() \times (P_i - X_i) + c_2 \times Rand() \times (P_g - X_i) \quad (1)$$

$$\begin{aligned} New\ position\ X_i = current\ position\ X_i + new\ V_i\ ; \\ V_{max} \leq V_i \leq -V_{max} \end{aligned} \quad (2)$$

Here, $w$ denotes the inertia weight that can balance the global search and local search[18]. $c_1$ and $c_2$ are two positive constants called learning factors ,rand() and Rand() generate a random number in the rang[0,1]. $V_{max}$ is the upper bound to change of velocity[16]. Four main parameters for PSO are the population size (number of particles), number of generation intervals, the maximum change of the flying velocity $V_{max}$ and the inertia weight $w$.

### 2.1.3 Ant-colony Optimization

ACO was developed by Dorigo et al.[19]. It takes inspiration from the natural fact that ants are able to find the shortest path between their nest and the food source. It is because ants leave a chemical substance named pheromone wherever they travel, and ants follow the paths with more pheromone deposits. As shown in Figure 2.1, the initial ants just randomly rotate around the obstacle on their first trip between the nest and the food source. So the right direction and the left direction have the same pheromone deposits. The ants travel the shorter path will find the food and return earlier (assume all the ants have the same speed). Then the returning ants will be more likely to follow the shorter path, as the shorter path has accumulated more

pheromone. Moreover, the ants that later start out will choose the shorter trial and deposit more and more pheromone on the path. As a result of this positive feedback, all the ants will finally choose the shortest path over time[20].

The pseudo code (Algorithm 2.1) from [3] describes how the basic ACO works and there are many variations to this basic version[21].

**Algorithm 3** Ant Colony Optimization[3]

```
Initialize the pheromone trails and parameters;
Generate population of m solutions (ants);
for each individualantk i ∈ m do
    Calculate fitness(k);
    Determine its best position;
    Determine the best global ant;
    Update the pheromone trail;
end for
Check if termination = true;
```

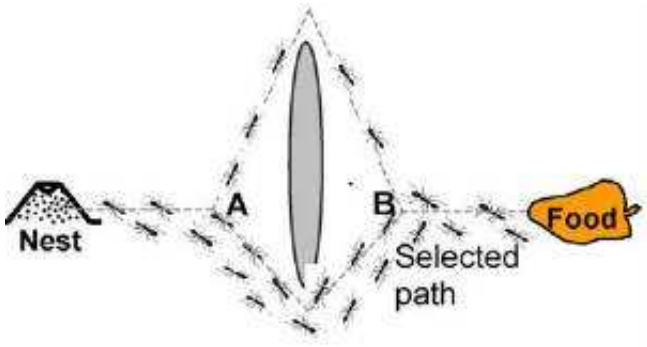


Fig. 2.1: Ants find the shortest path to the food[3]

Similar to PSO, an ACO system starts with m random ants, and each ant carries a solution string, with $n_i$ optional values for each variable i. The selected value for each variable represents the path which the ants will travel. The pheromone associated with each path (optional value) will change iteratively as the following equations shown [19]:

$$\tau_{ij}(t) = \rho\tau_{ij}(t-1) + \Delta\tau_{ij}; \quad t = 1, 2..., T$$

$$\Delta\tau_{ij} = \sum^{m} \begin{cases} R/fitness_k & \text{if option } l_{ij} \text{ is chosen by ant k} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Here, $\tau_{ij}(t)$ denotes the pheromone assigned with the ant i's $jth$ variable at the t$th$ iteration; $\rho$ is the pheromone evaporation rate (0-1)[22]; $\Delta\tau_{ij}$ expresses the increased pheromone deposits; R represents a constant number named pheromone reward factor[3] that is used in minimization problems and $l_{ij}$ denotes the $jth$ option for variable i. Then in the next iteration, the ant k will have probability $P_ij(k,t)$ to choose the path $l_{ij}$ according to the pheromone associated with the path.

$$P_{ij}(k,t) = \frac{[\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta}}{\sum_{l_{ij}} [\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta}]} \tag{4}$$

As Formula 4 shown, $\eta_{ij}$ denotes the heuristic factor that can indicate how good for ant k to choose option $l_{ij}$, $\alpha$ and $\beta$ are two positive numbers that control the importance of pheromone. The main parameters used in ACO are: the number of ants, the number of iteration, pheromone evaporation rate $\rho$, pheromone reward factor R, $\alpha$ and $\beta$.

### 2.1.4 Other EAs

Other EAs including Evolutionary Programming (EP), Memetic Algorithm (MA) and Shuffled Frog Leaping Algorithm (SFL) are briefly introduced as follows:

- **Evolutionary Programming**: The EP is a global optimization algorithm that is similar to the GA. However, it only uses the Gaussian-based mutation operator to generate new candidate solutions. EPs are usually applied in continuous function optimization[23].

- **Memetic Algorithm**: The MA was developed based on the notion of meme[24]. As the extension to the GA, the MA also has the chromosomes. However, the elements that constitute the chromosomes are called memes. MAs use local search technique, such as pair-wise interchange heuristics[25], to decrease the likehood of premature convergence.

- **Shuffled Frog Leaping Algorithm**: The SFL algorithm is a multi-population EA. In SFL algorithms, frogs(solutions) are divided into several sub-groups. Within each sub-group, the frogs can exchange ideas with each other through memetic evolution[26]. In addition to local search performing within each sub-group, the evolutionary information is also passed among sub-groups in a shuffling process.[27]

Many other EAs were developed besides the ones mentioned above. In general, we do not have an EA that can optimize all the optimization problems, various EAs have unequal performance for specific domains(see also the discussion on the no-free-lunch theorem for optimization [28]).

## 2.2 Cultural Algorithm

Inspired by the process of social and cultural changes, the CA was developed to enhance evolutionary computation. Besides the population component that evolutionary computation approaches have, there is an additional peer component belief space and a supporting communication protocol between these two components, which makes CAs perform better in some special optimal cases than other EAs. The following figure presents the basic CA framework[4].

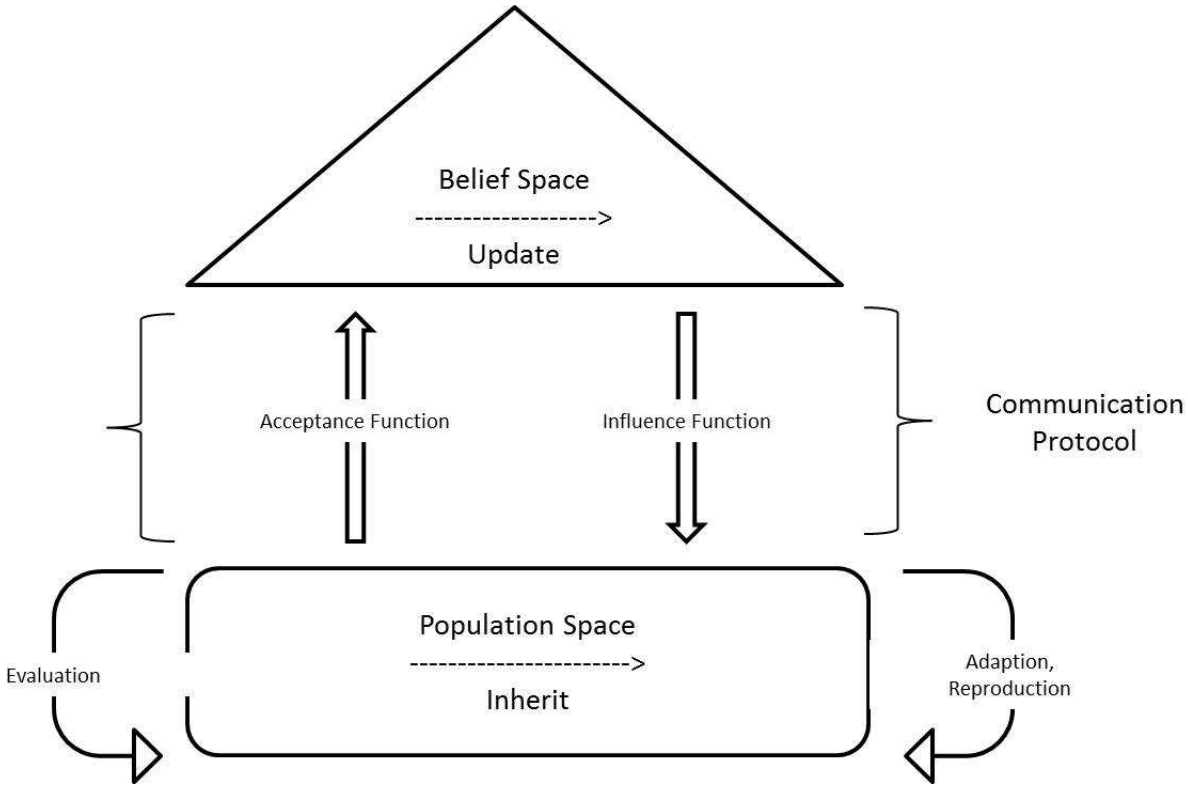


Fig. 2.2: CA framework[4]

As Figure 2.2 shown, the population space and the belief space can evolve respectively. The population space consists of the autonomous solution agents and the belief space is considered as a global knowledge repository. The evolutionary knowledge that stored in belief space can affect the agents in population space through influence function and the knowledge extracted from population space can be passed

to belief space by the acceptance function. The population space in CAs can be modeled by the population-based EAs, such as GAs[29], EPs[13] and PSO[30], etc.

In the process of the CA evolution, the population space is initialized with candidate solution agents at random, meanwhile, the initial knowledge sources in the belief space are built. At first, the two spaces evolve independently. Then the selected agents from the population space are used to update the belief space. After the knowledge sources being updated, the belief space will reversely guide the evolution of the population space. these procedures repeat till a termination condition has been reached. The CA pseudo code presented by [10] is given as follows:

**Algorithm 4** Cultural Algorithm[10]

```
1: t=0;
2: Initialize Population POP(t);
3: Initialize Belief Space BLF(t);
4: Repeat
5:    Evaluate Population POP(t);
6:    Adjust (BLF(t), Accept(POP(t)));
7:    Adjust (BLF (t));
8:    Variation(POP (t) from POP (t-1));
9: Until termination condition achieved
```

Different types of knowledge sources are used in the CA to solve different problems. The researchers have concluded five categories of knowledge sources and all available evolutionary information can be expressed by one of these five knowledge sources for a given domain. They are:

- **Situational Knowledge**: Chung[31] proposed the situational knowledge in 1997. It is designed to solve real-valued function optimization problems in the static environment. The situational knowledge can record the exemplars of successful and unsuccessful solutions.

- **Normative Knowledge**: The normative knowledge describes ranges of acceptable behaviors for the solution individuals[31]. The normative knowledge

can induce individuals to evolve within the domain that has the larger likehood to obtain optimal solutions.

- **Topographic Knowledge**: The topographic knowledge sources can express the spatial pattern of individual's behavior. The topographic knowledge is usually used to exploit the search space more precisely by dividing the space into smaller cells[32].

- **Domain Knowledge**: To solve dynamic optimization problems, Reynolds and Saleem introduced the domain knowledge to CAs[33]. It is used to monitor the changes of the environment and predict the evolutionary trend.

- **Historic Knowledge**: The historic knowledge was also proposed by Reynolds and Saleem[33]. The historic knowledge can be considered as the log in which the important events during the evolution of the population are recorded.

## 2.3 Multi-Population Cultural Algorithm

Researchers have developed many CAs with single population space. These CAs are powerful and perform well on a wide variety of problems. However, it has been proven that the CA system with multiple population spaces can reach better solutions to the optimization problems[34]. The MPCA enables implicit knowledge sources to be exchanged among sub-populations based on certain rules. With knowledge sources from different sub-populations migrated and cooperated with each other, the individuals can obtain more evolutionary information to evolve the population more quickly and more precisely. Consequently, the MPCA can improve the speed of convergence and have more probability to overcome premature convergence. The following diagram describes an MPCA framework proposed by Guo et al[13].

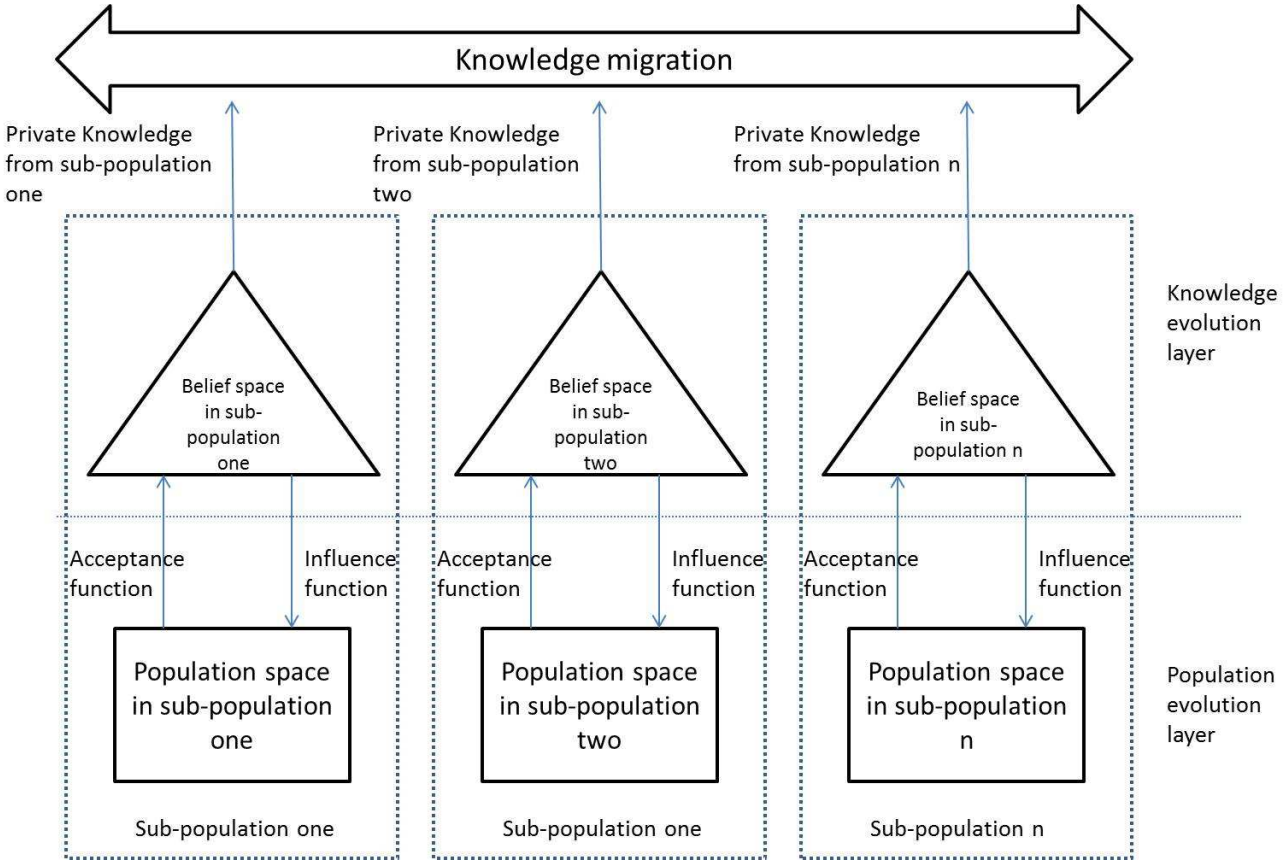


Fig. 2.3: MPCA framework

As the Figure 2.3 shown, an MPCA system contains n sub-populations. Each sub-populations is a basic CA system that consists of belief spaces and population spaces. Each sub-population evolves independently for a certain generations, and then the knowledge sources in the sup-populations will be migrated to each other. The migrated knowledge source from the other sub-population will blend with the private knowledge sources, as so to form the new private knowledge sources. By the way of knowledge migration, the evolutionary information extracted from different sub-populations will be shared with all the individuals. It is noted that, the knowledge migrations happen at a certain generation interval. The details of the migration strategies of different knowledge sources are discussed in Chapter 3.

The brief review of some current MPCAs is presented as follows:

- Digalakis and Margaritis[34] proposed an MPCA called the parallel co-operating cultural algorithm (PARCA). The PARCA consists of several sub-components and each of the them runs a CA with different hehaviour. Local search is adopted by each sub-component. They claimed that the exchange of information among the sub-CA systems allows them to co-operate and explore promising areas of the search space found by the other populations, and also to reintroduce previously lost cultural material in the population. However, the sub-populations in PARCA exchange the information extracted only from the best solution individual. This may reduce the accuracy and stability of the algorithm.

- Alami et al.[35] proposed an MPCA using fuzzy clustering. The proposed MPCA uses the fuzzy clustering technique to divide the initial population into sub-populations, and each sub-population can be managed by their own local CA. Besides, it introduces the cultural exchange concept that can be useful to mining new cultural. They declared that the experimental results indicated that their proposed model display better search performance than the sharing

fitness technique on four known multi-modal test functions. However, their model cannot support multiple types of knowledge sources for being migrated among sub-populations.

- MPCA-KM is a novel MPCA framework that was developed by Guo et al[13]. Guo et al. declared that MPCA-KM has faster convergence speed and better solutions than CA and PARCA for the high-dimensional static optimization problems. In MPCA-KM, the information is shared among sub-populations in the forms of different knowledge types. They introduce the knowledge cooperation strategy of topographic knowledge, but more cooperation strategies need to be developed for the other categories of knowledge sources. Moreover, the model does not consider the knowledge selection strategies when the MPCA-KM uses more than one knowledge source. They illustrated that the MPCA-KM requires a dynamic knowledge migration interval among sub-populations for the future work.

We have reviewed many EAs in this section. In summary, a GA has a simple structure that cannot store the individuals' knowledge over long time; For PSO, the individuals can communicate with each other, but only in one form where the individuals gather toward to the best solution; ACO is mainly used for discrete optimization problems that are not the domain that our work focuses on; CA can store and utilize various forms of knowledge, but the social contact is restricted to a single community. MPCA models have not only individual-to-individual social interaction, but community-to-community social interaction as well, that is more fit to natural system than single-population CA models. Therefore, our heuristics are built on the basis of MPCA which we will introduce in the next chapter.

# 3 PROPOSED ALGORITHM

In this chapter, we introduce the flowcharts of the proposed MPCA and the proposed MPCA will be compared with MPCA-KM. We subsequently discuss the design of knowledge sources that used in the proposed MPCA and how these knowledge sources migrate among the sub-populations. Last, the four heuristics that apply to the proposed MPCA are presented.

## 3.1 Algorihtm Flowchart

Figure 3.1 shows the flowcharts of both the MPCA-KM and the proposed MPCA. The parts wraped by the dotted line are the components to which the heuristic strategies apply.

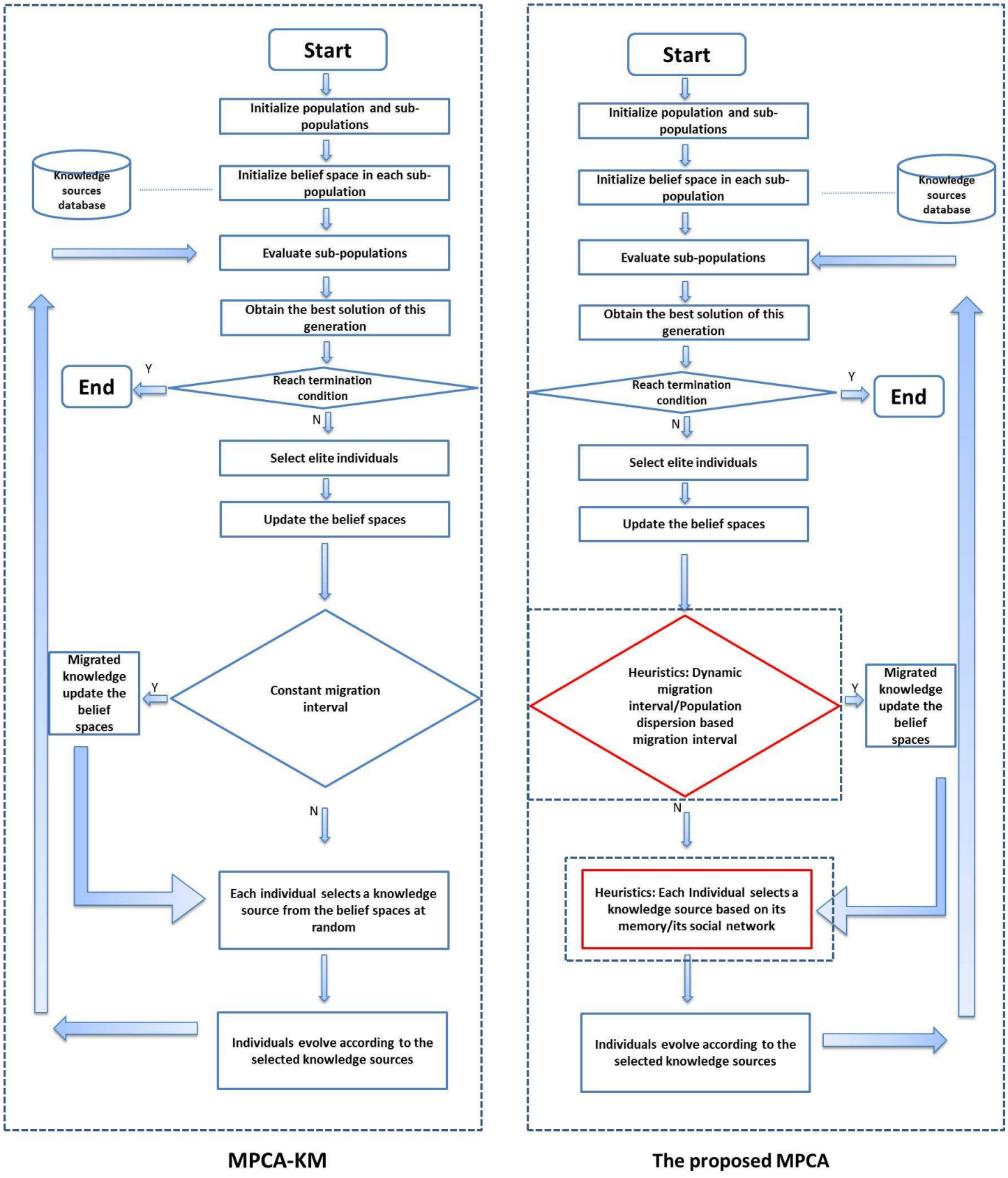


Fig. 3.1: Flowchart of the proposed MPCA

To observe the influence of the proposed heuristic strategies, each solution (individual) in the proposed MPCA is assigned with a unique ID that will not be changed during the whole process of the evolution as well as a knowledge-log that can record which and how the knowledge source impact the evolution of the individual in each generation. Figure 3.2 shows the sample of an individual in the proposed MPCA.

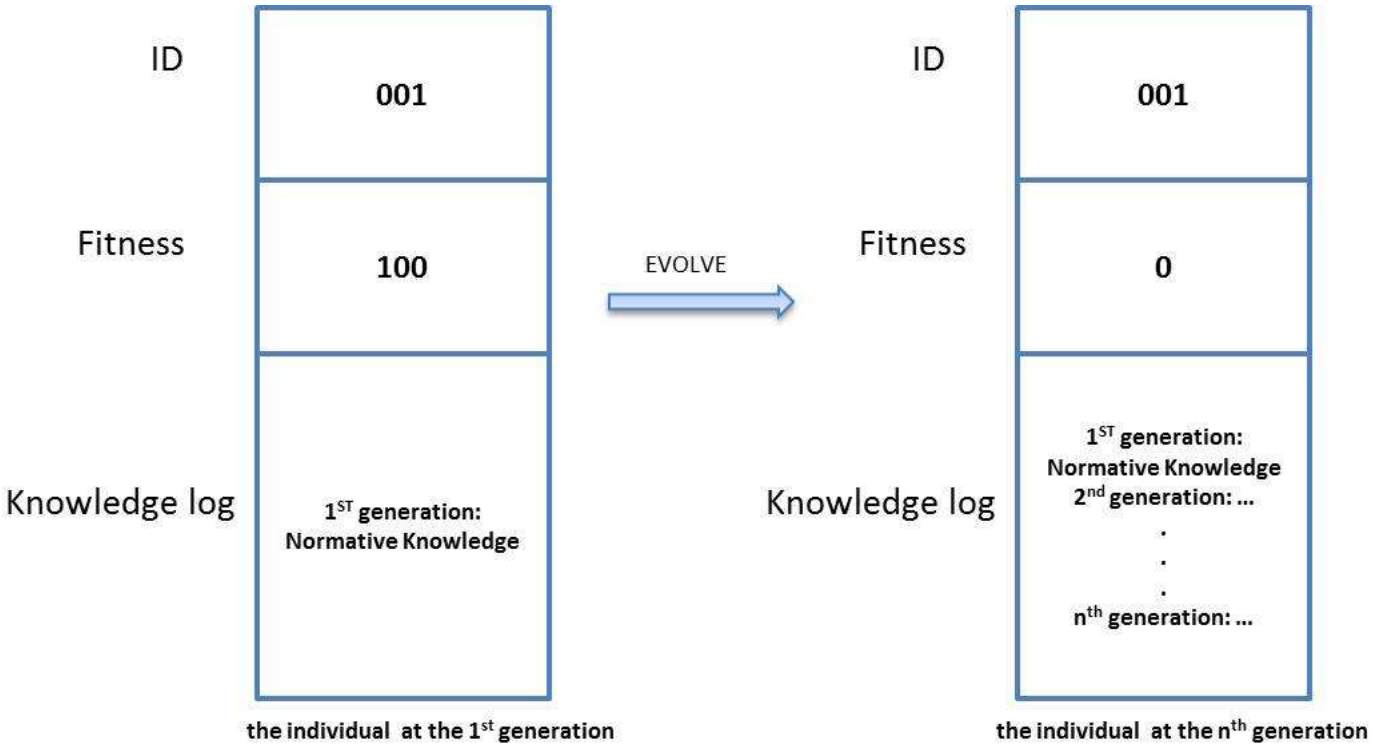


Fig. 3.2: Solution individual

Algorithm 5 describes the detail process of our proposed MPCA.

**Algorithm 5** MPCA with the proposed heuristics

Parameters initialize;
Generate random population of n solutions (individuals);
Initial population is divided into M sub-population denoted by $P^i$, i=1,2,...,M;
**for each** belief space $\in P^i$ **do**
  Initialize knowledge sources;
**end for**
**for each** individual x $\in P^i$, i=1,2,...,M **do**
  calculate fitness(x);
**end for**
**for each** $P^i$ **do**
  Sort the individuals according to fitness value;
  Select p% top individuals;
  **for each** private knowledge source **do**
    Update according to Formula (6)(7)(8)(9)(12)(17) in Section 3.2;
  **end for**
**end for**
[ ***Heuristics about knowledge migration interval*** ]
g $\leftarrow$ dynamic interval heuristics ***or*** population dispersion based interval heuristics
[ ***Heuristics about knowledge migration interval ends*** ]
**if** generation = g **then**
  **for each** sub-population having adjacent sub-populations **do**
    knowledge sources to be migrated to the adjacent sub-populations $\leftarrow$ private knowledge sources;
    new private knowledge sources $\leftarrow$ blend(private knowledge, migrated knowledge from the adjacent sub-populations) according to Formula (11)(15);
  **end for**
**end if**
[ ***Heuristics about knowledge selection*** ]
**for each** individual x $\in P^i$, i=1,2,...,M **do**
  Select one knowledge source from the belief space;
  [ ***Heuristics about knowledge selection ends*** ]
  $\bar{x}$ $\leftarrow$ evolve(x, the selected knowledge source) according to Formula (10)(14)(18);
  **if** fitness($\bar{x}$) is better than fitness(x) **then**
    $\bar{x}$ replaces x;
  **end if**
**end for**
generation $\leftarrow$ generation + 1;
Check if termination = true;

## 3.2 Knowledge Sources

The CA enables the evolutionary information to be collected, stored and updated in the belief space as knowledge source. Along with the development of CAs, it was concluded that five knowledge sources are are able to express all available knowledge for a given domain[36]. Each of these five knowledge sources is suitable to specific optimization problems. In our thesis, normative knowledge, topographic knowledge, and historic knowledge are used for the proposed MPCA, as the heuristic strategies primarily focus on the static environment. The update of the knowledge sources is discussed in this section, which refers to the $14th$ step in Algorithm 5. The evolution of individuals that controlled by the three knowledge sources refers to the $26th$ step in Algorithm 5 and the process of knowledge sources blend is related to the $21st$ step of Algorithm 5.

### 3.2.1 Normative Knowledge

Normative knowledge is used to define the feasible search space to an optimization problem. According to [31], the normative knowledge of the $ith$ sub-population $P^i$ is described as follows:

$$NK^i =< I_j^i, L_j^i, U_j^i > \tag{5}$$

Here, $j = 1, 2, ...m$, m denotes the number of variable of the optimization problem. $I_j^i = [l_j^i, u_j^i]$ records the bound of feasible search space to the $jth$ variable, and $l_j^i$ and $u_j^i$ express lower bound and upper bound, respectively. For instance, $I_1^1 = [-10, 10]$ means that lower limit and upper limit of the first variable in the first sub-population are -10 and 10. $L_j^i$ expresses the fitness score of the lower bound of the j$th$ variable. Similarly, $U_j^i$ denotes the fitness value of the upper bound of the j$th$ variable.

The normative knowledge sources in the belief spaces are updated by the selected

elite individuals as follows[31]:

$$l_j^i(t+1) = \begin{cases} x_{lj}^i(t) & if \quad x_{lj}^i(t) < l_j^i(t) \quad or \quad f(x_l^i(t)) < L_j^i(t) \\ l_j^i(t) & else \end{cases} \tag{6}$$

$$L_j^i(t+1) = \begin{cases} f(x_{lj}^i(t)) & if \quad x_{lj}^i(t) < l_i^i(t) \quad or \quad f(x_l^i(t)) < L_i^i(t) \\ l_j^i(t) & else \end{cases} \tag{7}$$

$$u_j^i(t+1) = \begin{cases} x_{lj}^i(t) & if \quad x_{lj}^i(t) > u_j^i(t) \quad or \quad f(x_l^i(t)) < U_j^i(t) \\ u_j^i(t) & else \end{cases} \tag{8}$$

$$U_j^i(t+1) = \begin{cases} f(x_{lj}^i(t)) & if \quad x_{lj}^i(t) > u_j^i(t) \quad or \quad f(x_l^i(t)) < U_j^i(t) \\ U_j^i(t) & else \end{cases} \tag{9}$$

Here, formula(6) and (7) represent the update for lower bound for the $jth$ variable and its lower fitness value at the (t+1)$th$ generation. Then equation(8) and (9) represent the update for upper bound of the $jth$ variable and update of the upper bound's fitness value. Within these formulas, $x_{lj}^i(t)$ denotes value of the j$th$ variable of the l$th$ individual in the i$th$ sub-population, and $f(x_l^i(t))$ indicates the fitness score of individual, $x_l^i(t)$.

Suppose $x_{lj}^i(t)$ select normative knowledge at the t$th$ iteration; it is induced to evolve by the normative knowledge source in the following way[31]:

$$\bar{x}_{lj}^i(t) = \begin{cases} x_{kj}^i(t) + \dfrac{\delta}{2}(u_j^i(t) - l_j^i(t)) & if \quad x_{lj}^i(t) \in I_j^i \\ \delta(u_j^i(t) - l_j^i(t)) + l_j^i(t) & if \quad x_{lj}^i(t) \notin I_j^i \end{cases} \tag{10}$$

Here, $\bar{x}_{lj}^i(t)$ is the value of the j$th$ variable to the individual $x_{lj}^i(t)$ after evolving, and $\delta$ is a random number between 0 and 1. As the formulas shown, normative knowledge utilizes the information that is carried by the elite individuals to reduce the feasible search space and direct the individuals to move into the feasible search

space, thereby avoiding inefficient search outside the dominant landscape.

In our proposed MPCA, each sub-population evolves independently. Therefore, the belief spaces of different sub-populations have varying degree of development. To increase the efficiency of the algorithm, sub-populations need to share their private knowledge with each other. Thereby the knowledge sources of more developed sub-populations need to cooperate with the knowledge sources of less developed sub-populations.

Suppose the normative knowledge source $NK^m$ in the belief space of the m$th$ sub-population migrates to the belief space of the i$th$ sub-population and cooperates with the normative knowledge source $NK^i$. This will cause the $NK^i$ to be updated as follows:

$$NK^i = \begin{cases} NK^m & if \quad \forall j \in [1,m], I_j^i \in I_j^m \; and \; L_j^m \leq L_j^i \; and \; U_j^m \leq U_j^i \\ NK^i & else \end{cases} \tag{11}$$

Figure 3.3 shows an example of normative knowledge sources that come from two sub-populations blend. The black boxes represent the feasible search space in sub-population A and sub-population B. Obviously, the sub-population B has a more developed normative knowledge source, which has narrowed the feasible space to a smaller region. After blending with the normative knowledge source from sub-population B, the normative source in sub-population A will evolve to the same range as the normative knowledge source in sub-population B. The migration of normative knowledge can speed up the evolution of the belief spaces in sub-populations so as to accelerate the convergence speed of the whole population.

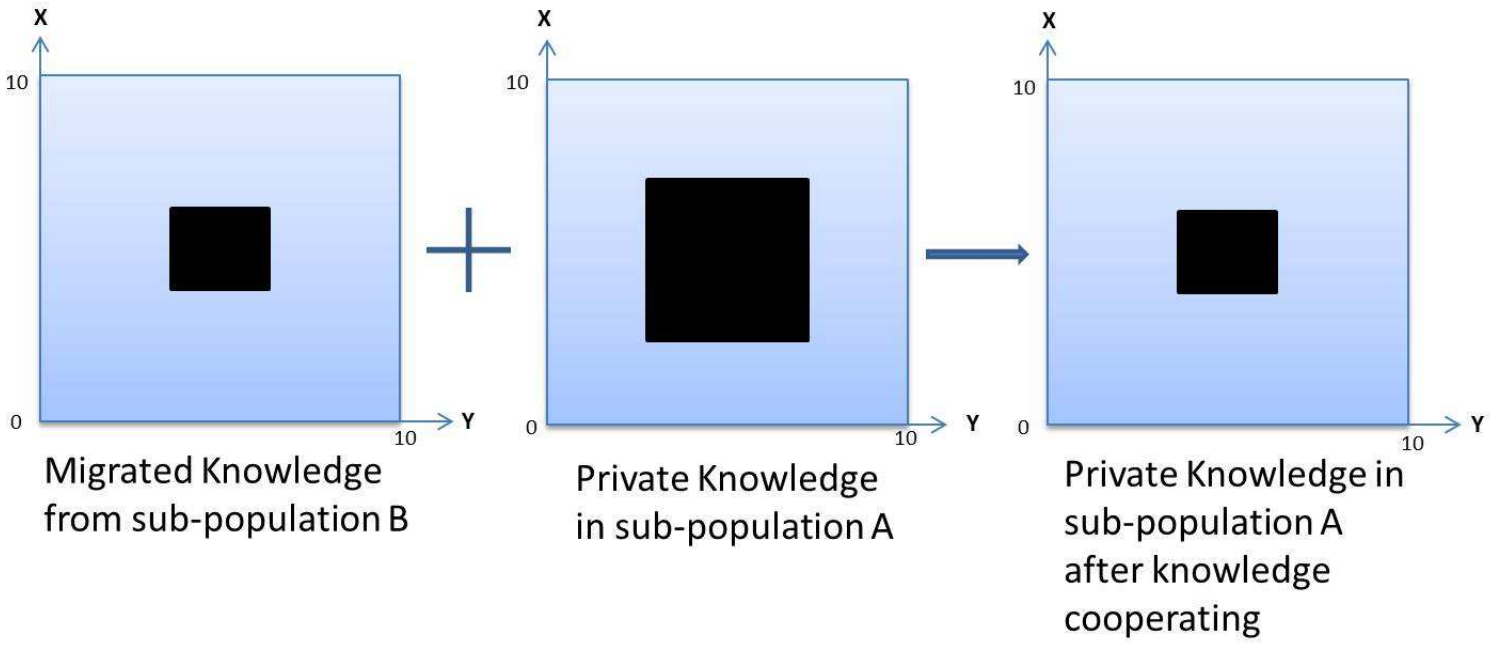


Fig. 3.3: Cooperate mechanism of normative knowledge (m = 2)

At the early stage of evolution, normative knowledge can guide the population explore the whole search space and quickly identify the dominant regions. However, after the individuals gather at the regions with the most potential, normative knowledge is unable to drive them into exploiting the regions more precisely.

### 3.2.2 Topographic Knowledge

Topographic knowledge describes the distribution of good solutions in the feasible search space. Within the search space recorded by normative knowledge, the area containing the best solution is uniformly divided into subspaces along each dimension by a binary tree. Thereby, the m-dimension hypercubes with the same size are formed. The hypercubes are called cells. Obviously, there are $2^m$ cells. Assume the best solution at the $t-1$ iteration is located in a 2-dimension area [(-10,10),(-10,10)], and four cells are obtained in this area, noted by $C_1^i(t-1), ..., C_4^i$ (Figure 3.4). Then, if the best solution at the $t$ iteration is located in $C_1^i(t-1)$, this area is divided into

four cells again, as shown in Figure 3.4.

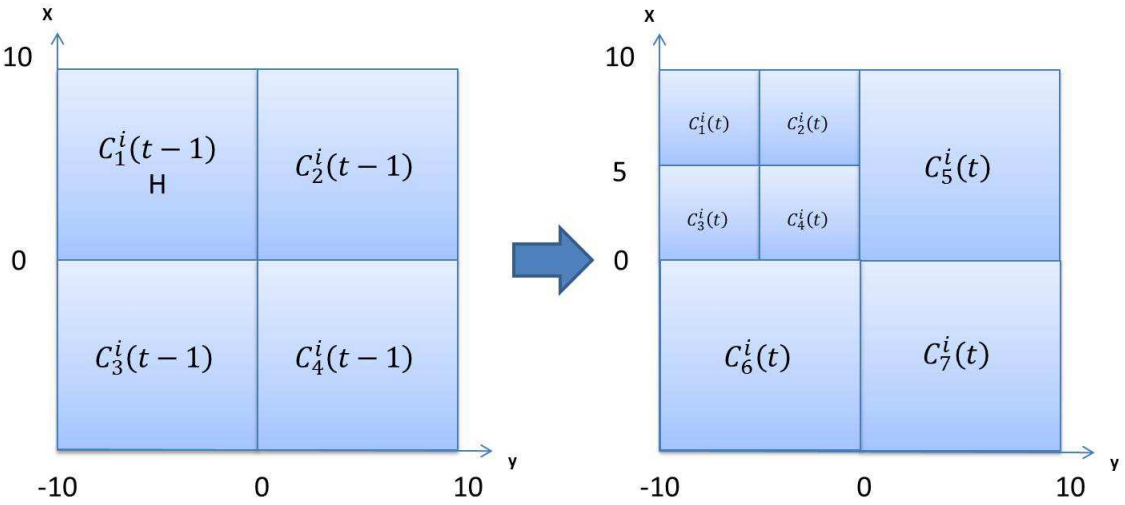


Fig. 3.4: Topographic knowledge(2-dimension)

Topographic knowledge composed of cells is described as follows[32]:

$$TK^i(t) =< C_1^i(t), C_2^i(t), ...C_k^i(t), ... > \tag{12}$$

Each cell is assigned an attribute about the potential to get the optimal solution.

$$\text{Cell attribute} = \begin{cases} High(H) & f(x_k^{i^*}) > \bar{f}(x_k^{i^*}) \\ Unknown(\#) & x^i(t) \notin C_k^i(t) \\ Low(L) & f(x_k^{i^*}) \leq \bar{f}(x_k^{i^*}) \end{cases} \tag{13}$$

In formula 13, $f(x_k^{i^*})$ denotes the fitness value of the best individual in the kth cell and then $\bar{f}(x_k^{i^*})$ expresses the average performance value of the best individuals from all cells. The cell attribute shows the possibility of a cell containing the global optimal solution. H means the cell is a possible subspace finding the better solutions. # means a cell has not been explored. In order to determine whether good solutions are located in the cells, cells with a # attribute shall be searched.

Topographic knowledge is used to induce individuals to exploit certain cells, namely local search. Topographic knowledge evolves the individuals in the following way[32]:

$$\bar{x}_{lj}^{i}(t) = \begin{cases} x_{kj}^{i}(t) + \delta \dfrac{f(x_l^i(t))}{\sum_{l=1}^{n} f(x_l^i(t))}(u_j^i(t) - l_j^i(t)) & if \quad (x_{lj}^i(t) \notin C_k^i(t)) \cap (attribute_k^i \neq L) \\ x_{lj}^{i}(t) + \dfrac{\delta}{m}\sqrt{f(x_{lj}^i(t))} & if \quad (x_{lj}^i(t) \in C_k^i(t)) \cap (attribute_k^i = H) \end{cases} \tag{14}$$

As above mentioned, cells with H or # attribute shall be mainly searched. If the individuals are located in the cells with H, they will search for the optimal solutions amongst these cells. If the individuals are located in cells with L attribute, they will move towards the cells with better potential.

Topographic knowledge directly influences the exploitable ability of the individuals to the dominant region. Different sub-populations may have topographic knowledge sources of different development. A reasonable strategy of cooperating topographic knowledge sources from different sub-populations is important. [13] provides a cooperate mechanism to topographic knowledge. Suppose $TK^j =< C_1^j(t), C_2^j(tC_3^j(t), C_4^j(t), C_5^j(t), C_6^j(tC_7^j(t), >$ is the topographic knowledge in the j$th$ sub-population at the t$th$ generation as shown in Figure 3.5, and $TK^i =< C_1^i(t), C_2^i(tC_3^i(t), C_4^i(t), C_5^i(t), C_6^i(tC_7^i(t)$ >is the topographic knowledge from the i$th$ sub-population as shown in Figure 3.5. Then, if the knowledge migration condition is satisfied at the the t$th$ generation,$TK^j$ and $TK^i$ will mix to form the new topographic knowledge $TK^{ij}(t)$.

$$TK^{ij}(t) =< C_1^j(t), C_2^j(t), C_3^j(t), C_4^j(t)...C_4^i(t), C_5^i(t), C_6^i(t), C_7^i(t)) > \tag{15}$$

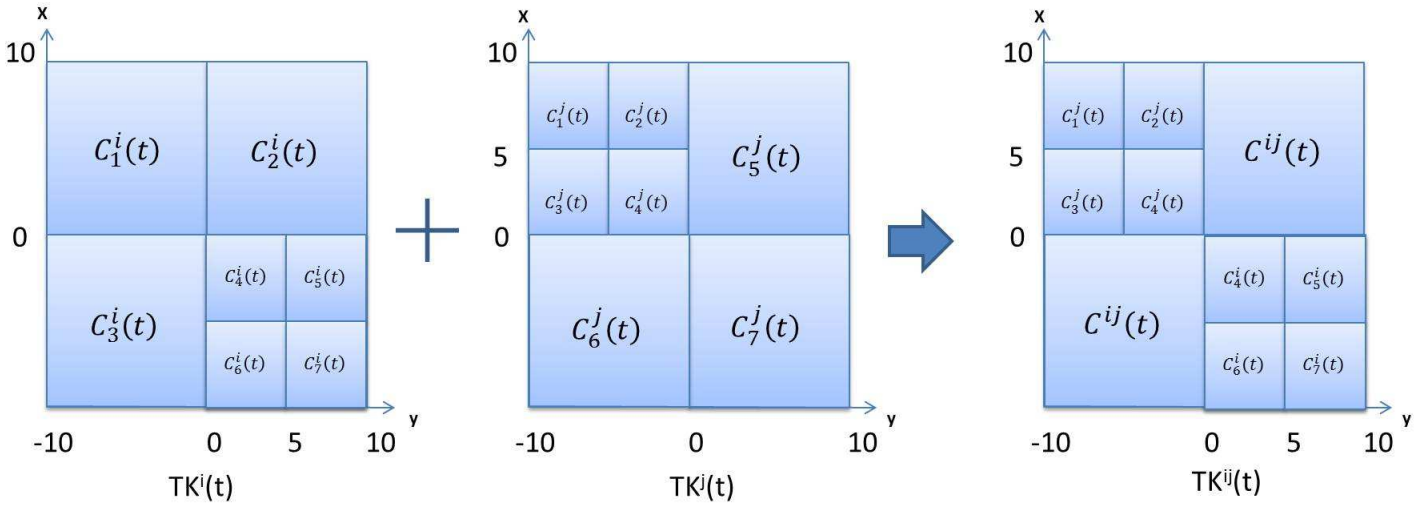


Fig. 3.5: Cooperate mechanism of topographic knowledge (m = 2)

### 3.2.3 Historic Knowledge

In this thesis, historic knowledge is designed to record the good solutions that have been found, which can be represented as follows:

$$HK^i(t) =< E_1^i, E_2^i, ...E_j^i, ....E_s^i > \tag{16}$$

Here, s is maximum length of $HK^i$, $E^i(t) = [x_t^i]$ , $x_t^i$ denotes the best individual of the $ith$ sub-population $t$ iterations ago and t≤s. The historic knowledge database can be described as a queue. Once the queue is full, the new coming record will replace the most recent record which has been stored in the queue. The update of historic knowledge is shown as follows:

$$HK^i(t+1) = \begin{cases} < E^i(1), E^i(2), ...E^i(t), E^i(t+1) > & if \quad t < s \\ < E^i(1), E^i(2), ...E^i(s) > & if \quad t \geq s \end{cases} \tag{17}$$

To avoid missing potential region, historic knowledge induce individuals to search for the former regions which once produced good solutions. When historic knowledge

sources work, Gaussian mutation[37] will be adopted in the vicinity of the best solutions at the early stage of the evolution. The influence function relevant to historic knowledge is given in Equation 18, $S^i$ denotes a random individual selected from $SK^i$, and $w_1$, $w_2$ represent the weight parameters, $w_1, w_2 \geq 0$ and $w_1 + w_2 = 1$. $\delta$ is a random number between 0 and 1.

$$x_j^i(t+1) = S^i(t)(w_1 + w_2\delta) \tag{18}$$

## 3.3 Heuristic Strategies

In order to improve the weakness in current MPCAs, which are the random knowledge sources selection mechanism and the constant knowledge migration interval, four heuristic strategies applied to the different components of MPCAs are proposed. The four heuristic strategies are knowledge selection based on individual memory, knowledge section based on social interaction, dynamic knowledge migration interval and population dispersion based knowledge migration interval.

### 3.3.1 Strategy One (S1): Knowledge Selection Based on Individual Memory

Utilizing more than one type of knowledge sources enable MPCAs to solve more complex optimization problems. However, different categories of knowledge sources have different effectiveness for the entire population from early time to latter time of the evolution. Furthermore, for an individual, the effectiveness of every type of knowledge source will change along with the evolution, since the individual's position will change over the evolution. For example, an individual that has been very close to the optimal solution can get more benefit from topographic knowledge sources than normative knowledge sources, since the topographic knowledge sources can guide the individual search for the space more precisely.

The individuals in existing MPCAs cannot identify the importance of knowledge sources, thereby they choose random knowledge sources to evolve the population, which may reduce the efficiency of MPCAs. Therefore, a novel knowledge selection heuristic strategy is introduced in this thesis. It is assumed that the individuals have memory of the knowledge source's performance, due to which they are more likely to select the knowledge source that has better performance. Figure 3.6 shows an example of how an individual that has memory selects the better knowledge source. The individual described as a face has been through three generations in which it selected

normative knowledge source, topographic knowledge source, and historic knowledge source in order. The height of the bar represents the fitness value of the individual. The topographic knowledge source gain the height fitness bar most. Therefore, the individual is most likely to select the topographic knowledge at the fourth generation.

It is noted that:

- The individuals only have short-term memory. That is only the most recent performance of each knowledge source can be kept for the individuals;
- The higher performance value one knowledge source has, in the larger possibility the individual will select the type of knowledge source in the next round;
- The initial performance value of all knowledge sources are zero;
- If the performance value of all three types of knowledge sources for the individual are zero, the individual will select each of the three types of knowledge sources once in the next three generations.

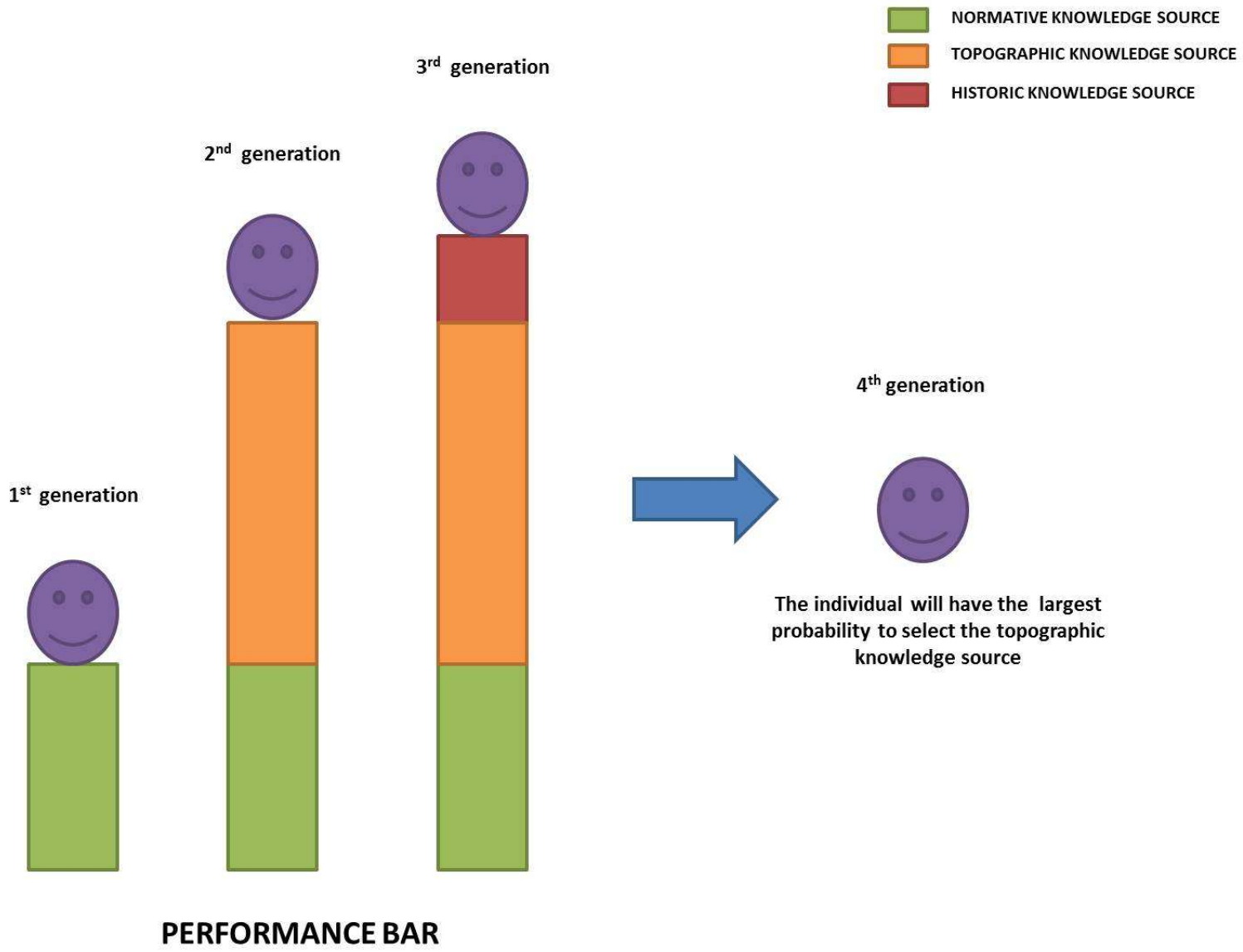


Fig. 3.6: The memory can affect the individual's decisions

Assume $m_{lj}^i$ denotes the performance value of the j$th$ knowledge source for the l$th$ individual in the i$th$ sub-population. Then the update of $m_{lj}^i$ is shown as follows:

$$m_{lj}^i = |fitness(x_l^i(t+1)) - fitness(x_l^i(t))| \quad \text{if } x_l^i \text{ selects the j}th \text{ knowledge source at the t}th \text{ generation} \tag{19}$$

The probability for the l$th$ individual to select the j$th$ knowledge source in the

next generation is defined as:

$$p_{lj}^{i} = \frac{m_{lj}^{i}}{\sum_{j=1}^{n} m_{lj}^{i}} \tag{20}$$

where n denotes the total number of available knowledge sources.

Algorithm 6 describes the MPCA with knowledge selection based on individual memory. Algorithm 6 refers to [ ***Heuristics about knowledge selection*** ] in Algorithm 5.

**Algorithm 6** MPCA with knowledge selection based on individual memory

1: **for each** individual x $\in P^i$, i=1,2,...,M **do**
2: update($m_{lj}^i$)
3: **if** $\forall j \in [1, n],\ m_{lj}^i = 0$ **then**
4: $x_l^i$ will select normative knowledge source, topographic knowledge source, and historic knowledge source in order in the next three generations;
5: **else**
6: $x_l^i$ selects the one knowledge source j according to probability $p_{lj}^i$;
7: **end if**
8: **end for**

### 3.3.2 Strategy Two (S2): Knowledge Selection Based on Social Interaction

Knowledge selection strategy based on social interaction is proposed for the same purpose as knowledge selection strategy based on individual memory. It is assumed that the individuals have social connections with each other in the population. The individuals' choices can be affected by their "neighbours" in the population[40]. For example, as the Figure 3.7 shown, it is a simple social network for a 5-individual population. The circle with different colours represents the individual that want to suggest the knowledge source of that color to its neighbours. Then, the individuals will transit their knowledge suggestions to their neighbours(adjacent individuals). For instance, individual A will transit the suggestion of selecting yellow knowledge source to individual C, individual B and individual D, so as the other individuals in

the network. Then every individual counts up the knowledge source suggestion bids collected (include the original knowledge suggestion hold by the individual itself ). The knowledge source type that has most votes will be selected by the each individual in next iteration. In the example of figure 3.7, individual A has collected two blue knowledge suggestions, one yellow knowledge bid and one green knowledge suggestion, so A will select blue knowledge source.

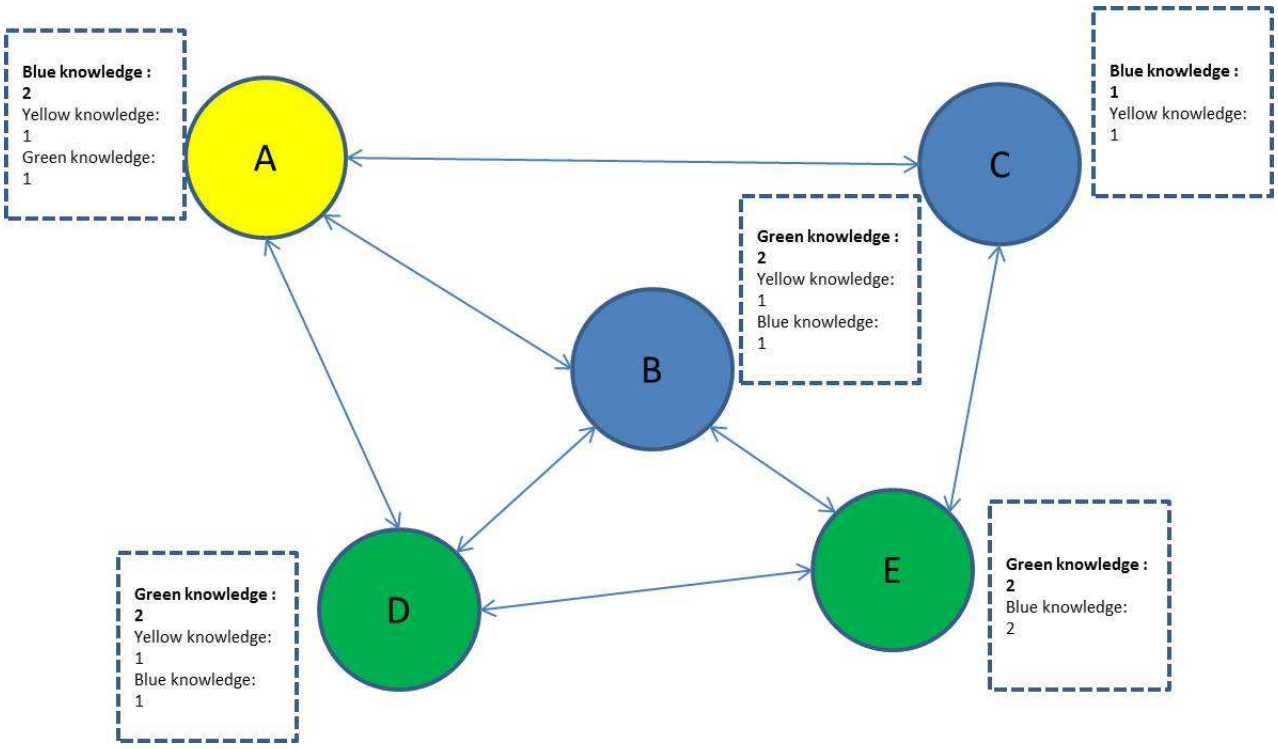


Fig. 3.7: Individuals select knowledge source based on social influence

For this heuristic strategy, it is noted that:

- Which knowledge source the individuals suggests to its neighbors is determined by the individual memory developed in strategy one;
- If there are ties between knowledge sources bids, the individual will select the knowledge source suggested by itself;

- There are different types of social network topology. The efficiency of different social network used in the algorithm will be discussed in the next chapter;
- Each sub-population has independent social network;

  Algorithm 7 shows the MPCA with knowledge selection based on social interaction. Algorithm 7 refers to [ ***Heuristics about knowledge selection*** ] in Algorithm 5.

**Algorithm 7** MPCA with knowledge selection based on social interaction

**for each** individual x $\in P^i$, i=1,2,...,M **do**
  update($m_{lj}^i$)
  **if** $\forall j \in [1,n],\ m_{lj}^i = 0$ **then**
    $x_l^i$ will suggest normative knowledge source, topographic knowledge source, and historic knowledge source in order over next three generations;
  **else**
    $x_l^i$ suggests the knowledge source j according to $p_{lj}^i$
  **end if**
**end for**
**for each** $P^i$ **do**
  **for each** Individual **do**
    Transmits the suggested knowledge source to the adjacent individuals.
    Counts the bids of collected knowledge sources
    **if** There are ties between knowledge sources **then**
      Selects the suggested knowledge source
    **else**
      Selects the knowledge source that has most votes
    **end if**
  **end for**
**end for**

### 3.3.3 Strategy Three (S3): Dynamic Knowledge Migration Interval

As discussed in section 1.2, constant knowledge migration rate that apply to the existing MPCAs may result in premature convergence and more convergence generations for optimization problems. At the latter time of evolution, the diversity to individuals is low, so more knowledge migrations among sub-populations are in demand to speed

up the pace on convergence. On the other hand, the frequent migration of knowledge at the early time decrease the diversity of individuals, this may lead the algorithm trapping in the local optima.

Therefore, a dynamic knowledge migration rate is introduced as followed[38]:

$$\eta(t) = floor(\eta_{max} N^{-t} + 1) \tag{21}$$

Here, $\eta$ denotes the number of iteration between two knowledge migrations ,$\eta_{max}$ denotes the maximum interval, $\eta \in [1, \eta_{max}]$ and N reflects the rate of interval change, $N > 1$.

Figure 3.8 shows an example of knowledge migration interval. The maximum interval is 50 generations at the beginning of evolution; then the interval gradually decreases to 1 at around the 200th generation.

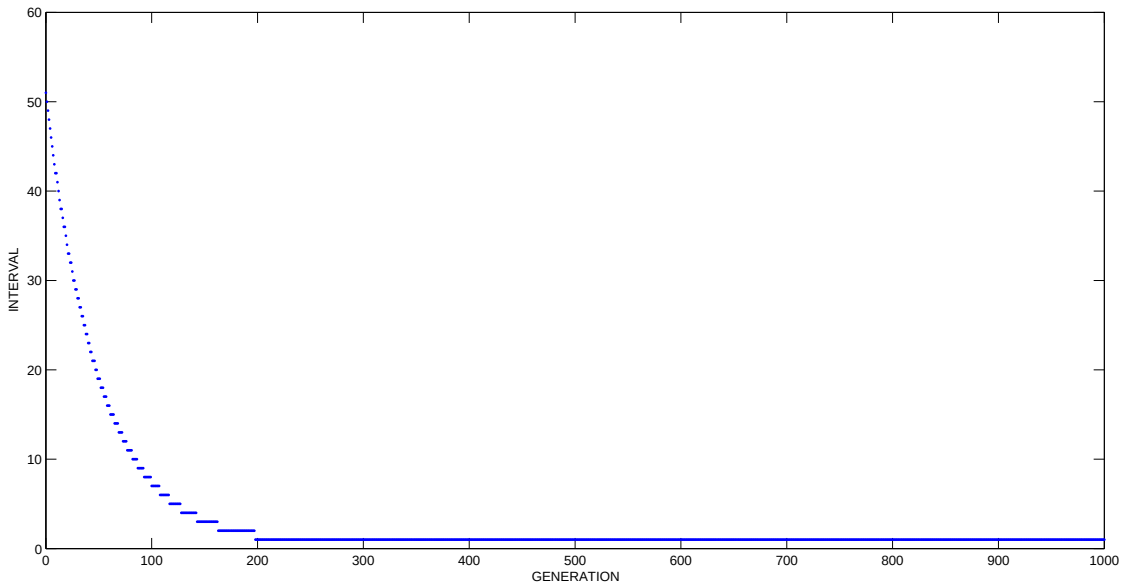


Fig. 3.8: Dynamic knowledge migration interval ($\eta_{max} = 50$, $N = 1.02$)

Algorithm 8 describes the MPCA with dynamic knowledge migration interval and refers to [ ***Heuristics about knowledge migration interval*** ] in Algorithm

5.

**Algorithm 8** MPCA with dynamic knowledge migration interval

1: update($\eta$);
2: g $\leftarrow$ $\eta$+generation of previous knowledge migration

### 3.3.4 Strategy Four (S4): Population Dispersion Based Knowledge Migration Interval

This strategy is put forward for the same purpose as strategy three. MPCAs need an indicator to determine when to migrate the knowledge sources among sub-populations. Therefore, we use population dispersion rate to decide the timing of knowledge migration. Population dispersion was proposed by Lisis [39]. It is used to measure the intensity of the population.

$$PD(t) = \sum_{j=1}^{n}[\frac{1}{popsize}\sum_{i=1}^{popsize}(x_{ij}(t) - \frac{1}{popsize}\sum_{i=1}^{popsize}x_{ij}(t))^2] \tag{22}$$

In formula 22, $PD(t)$ refers to the dispersion rate of the entire population at the t$th$ generation, $x_{lj}(t)$ represents the j$th$ variable of the l$th$ individual at the t$th$ generation.

It is assumed that:

- If $PD(t) > \tau$, $\tau$ is a positive number. The distribution of individuals in the population space is scattered, the knowledge sources of each sub-population do not migrate at the t$th$ iteration. This will maintain the high diversity of the population.

- If $PD(t) \leq \tau$, the distribution of individuals in population space is concentrated, the private knowledge of each sub-population will migrate to other sub-populations at this iteration.

Algorithm 9 shows the MPCA with dynamic knowledge migration interval and refers to [ ***Heuristics about knowledge migration interval*** ] in Algorithm 5.

**Algorithm 9** MPCA with population dispersion based knowledge migration

```
1: update(PD(t));
2: if PD(t) > τ then
3:    g ← 0;
4: else
5:    g ← generation;
6: end if
```

# 4 EXPERIMENT AND RESULT ANALYSIS

In this chapter, we first introduce the test functions for the experiment. Then we explain the detail of the experimental setup. Last, the summary results of the tests along with the analysis will be presented.

## 4.1 Benchmark Optimization Functions

Many commonly used benchmark optimization functions are used to evaluate and compare the performance of optimization algorithms. In our experiments, five static minimal optimization functions[5] with different characteristics are taken to test the proposed heuristic strategies. All of these test functions are dimension-wise scalable.

1. $F_1(x) = \sum_{i=1}^{D} x_i^2$
2. $F_2(x) = (\sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2)$
3. $F_3(x) = -20exp(-0.2\sqrt{\frac{1}{m}\sum_{i=1}^{D} x_i^2}) - exp[\frac{1}{m}\sum_{i=1}^{D} cos(2\pi x_i)] + 20 + e$
4. $F_4(x) = \sum_{i=1}^{D} (x_i^2 - 10cos(2\pi x_i) + 10)$
5. $F_5(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$

Table 4.1: Dimensions, search ranges,and global optimum values of the test functions

| **Function** | **Dimension** | **Search range** | **Optimal solution** | **Global minima** |
|---|---|---|---|---|
| $F_1 : Sphere\ function$ | 10/50 | $x_i \in [-100, 100]$ | $[0, 0, ..., 0]$ | 0 |
| $F_2 : Schwefel\ problem$ | 10/50 | $x_i \in [-100, 100]$ | $[0, 0, ..., 0]$ | 0 |
| $F_3 : Ackley\ function$ | 10/50 | $x_i \in [-30, 30]$ | $[0, 0, ..., 0]$ | 0 |
| $F_4 : Rastrigin\ function$ | 10/50 | $x_i \in [-5.12, 5.12]$ | $[0, 0, ..., 0]$ | 0 |
| $F_5 : Rosenbrock\ function$ | 10/50 | $x_i \in [-30, 30]$ | $[1, 1, ..., 1]$ | 0 |

$F_1$ **(Sphere Function)** : As the figure 4.1 shown, sphere function is simplest unimodal function in the test functions. It is featured with unimodal, separable and dimension-wise scalable.

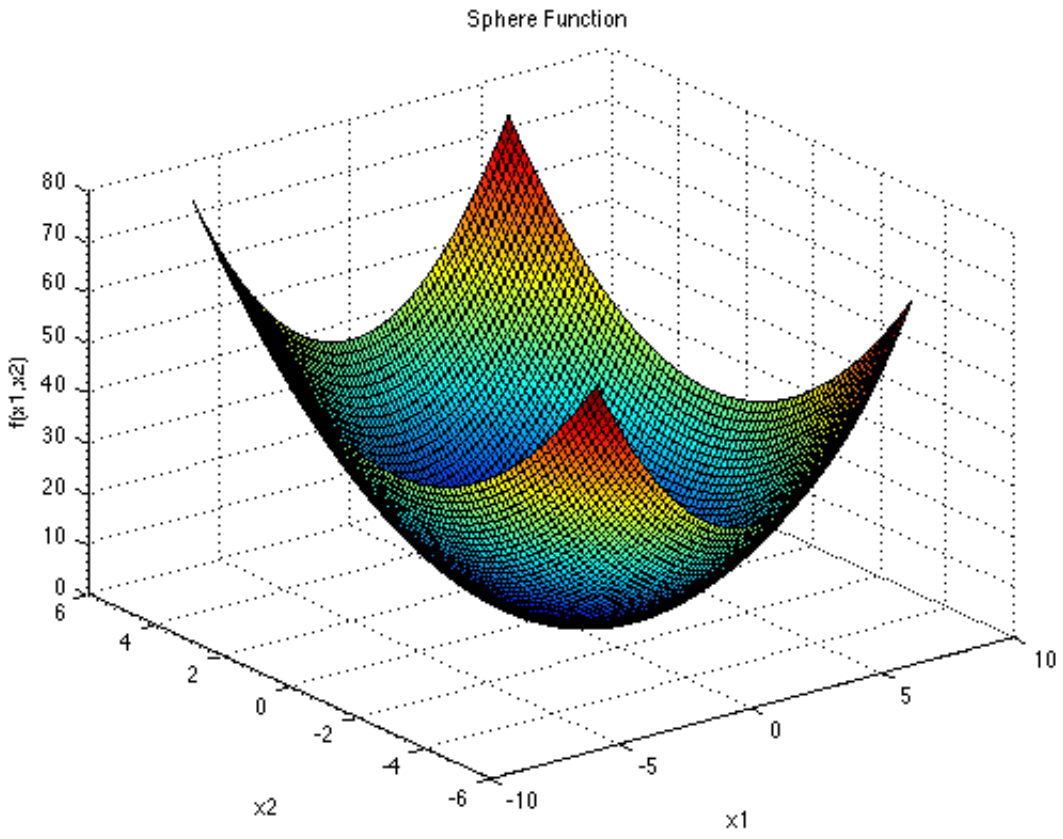


Fig. 4.1: 3-D image for 2-D Sphere Function[5].

$F_2$ **(Schwefel Problem 1.2 Function)** : As shown in figure 4.2, the global minimal solution is located in narrow and long area. Schwefel problem 1.2 function is featured with unimodal, separable, and dimension-wise scalable.

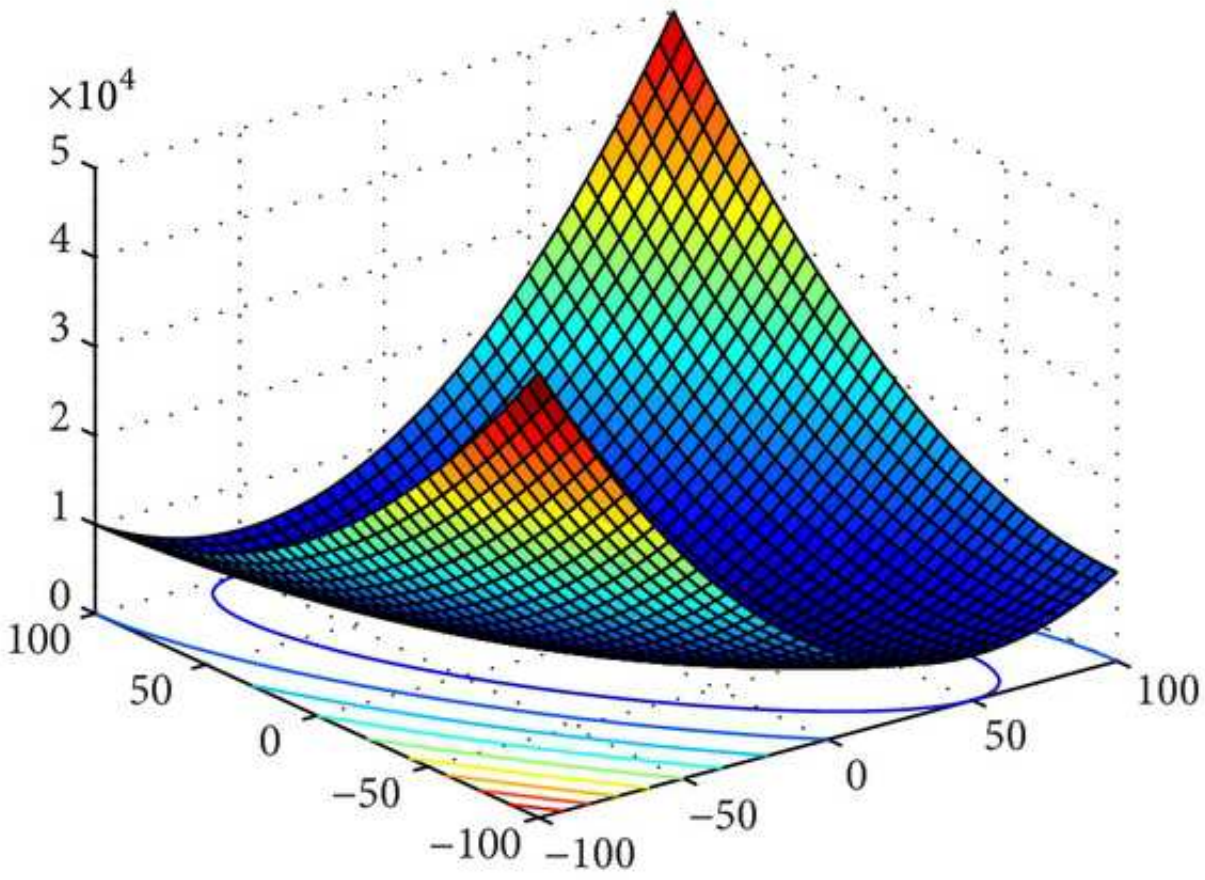


Fig. 4.2: 3-D image for 2-D Schwefel Problem 1.2 Function[5].

$F_3$ **(Ackley Function)** : Ackley function is a multi-modal optimization function which has a lot of local minima. A large number of modals(local minima) are located in a very flat area(figure 4.3) due to modulation of added amplified cosine wave. It is featured with multi-modal, non-separable, and dimension-wise scalable.

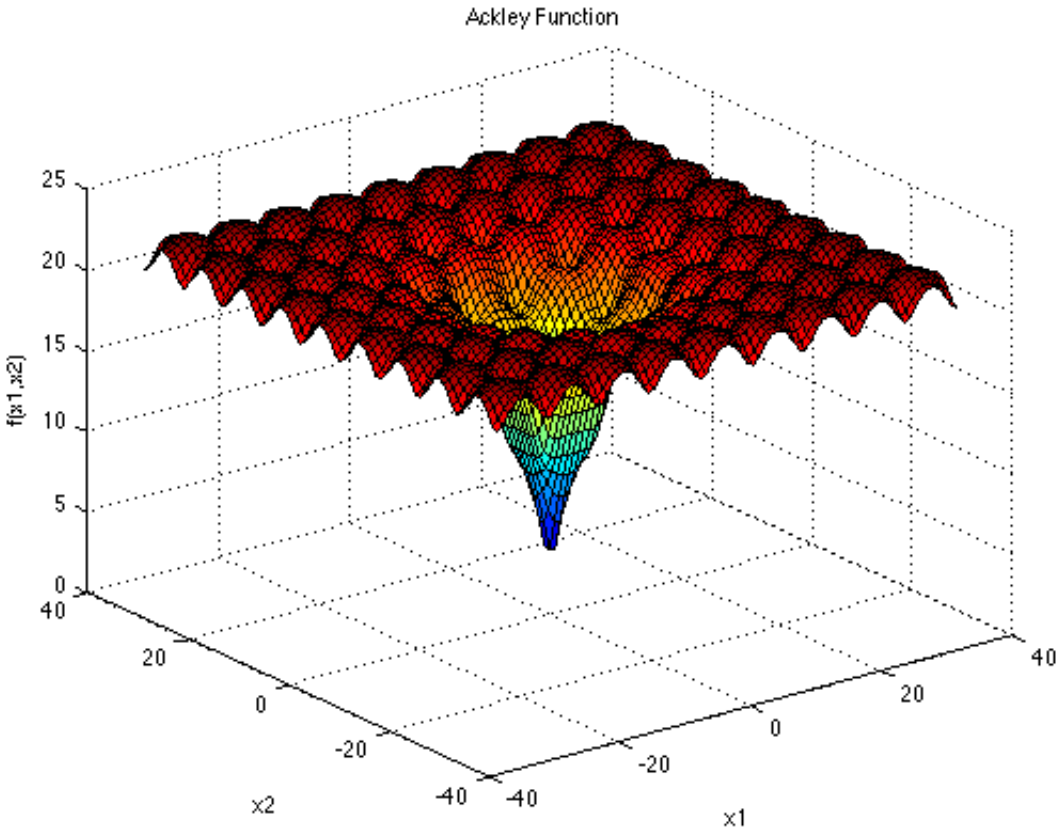


Fig. 4.3: 3-D image for 2-D Ackleys Function[5].

$F_4$ **(Rastrigin Function)** : Rastrigin function has lots of local minimas in the range [-5.12,5.12], seen from figure 4.4. It is featured with multi-modal, separable, dimension-wise scalable, and huge serried local optima.

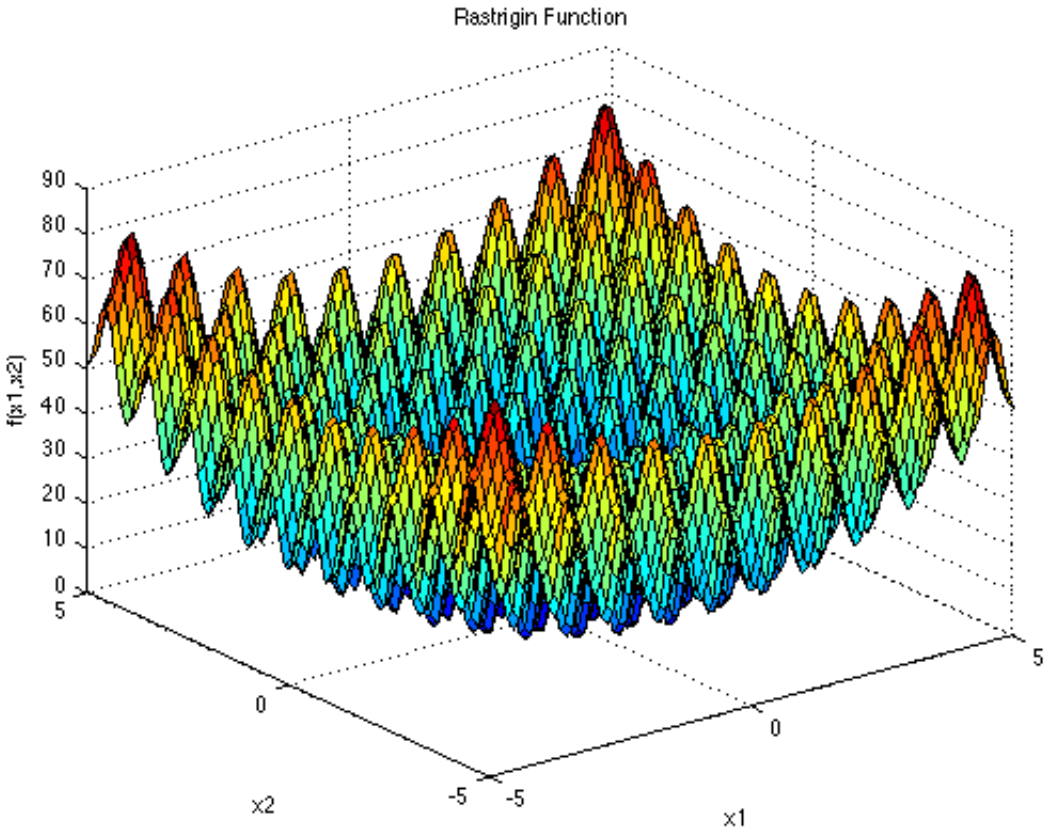


Fig. 4.4: 3-D image for 2-D Rastrigin Function[5].

$F_5$ **(Rosenbrock Function)** : Rosenbrock function is a multi-modal optimization function which the global minima is located in a very narrow and long bar-type area. It is featured with multi-modal,non-separable,dimension-wise scalable, Having a very narrow valley from local optimum to global optimum.

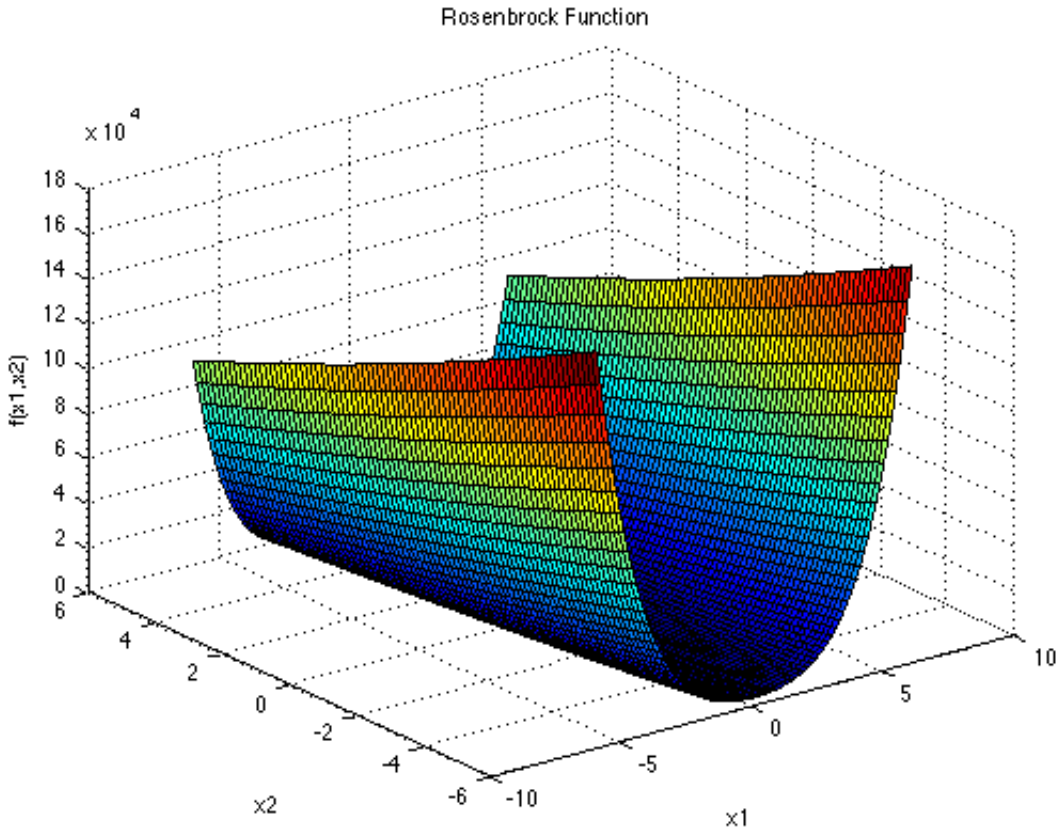


Fig. 4.5: 3-D image for 2-D Rosenbrock Function[5].

These different types of benchmark optimization test functions with different characteristics are appropriate for evaluating the performance of the proposed algorithm.

## 4.2 Experimental Setup

We perform the experiments to compare the performance between MPCA with the proposed heuristics and MPCA without the heuristics (MPCA-KM) on the five test functions. Furthermore, we compare the efficiency of different heuristic strategies. The combinations of the strategies explained in Chapter 3 and their abbreviations are listed as follows:

- **M1**: Basic MPCA without heuristics
- **M2**: MPCA with knowledge selection based on individual memory heuristics
- **M3**: MPCA with knowledge selection based on social interaction heuristics
- **M4**: MPCA with dynamic knowledge migration interval
- **M5**: MPCA with population dispersion based knowledge migration interval
- **M6**: MPCA with knowledge selection based on individual memory heuristics and dynamic knowledge migration interval
- **M7**: MPCA with knowledge selection based on individual memory heuristics and population dispersion based knowledge migration interval
- **M8**: MPCA with knowledge selection based on social interaction heuristics and dynamic knowledge migration interval
- **M9**: MPCA with knowledge selection based on social interaction heuristics and population dispersion based knowledge migration interval

The nine algorithms listed above fall into 3 sets: M1 vs M2 vs M3, M1 vs M4 vs M5 and M1 vs M6 vs M7 vs M8 vs M9. For a fair comparison, the algorithms in the same set use the same parameters given in Table 4.3, Table 4.5 and Table 4.7 . All the experiments are carried out with JAVA language. For the purpose of reducing statistic error, sixty trial runs are performed for each function. The performance of the different algorithms was compared using the following criteria:

1. Mean fitness: mean value of the solutions got at the maximum generation in 60 runs.
2. Standard deviation: standard deviation of mean fitness.

3. Interval number: The average value of iteration number to reach the target solution value in 60 runs. The target solution value is usually within an acceptive tolerance of the known global optima for an optimization function. The target values to the five test functions are summarized in table 4.2.

Table 4.2: Target values of 5 test problems [1]

| **Function** | **Target** |
|---|---|
| F1(10-Dimension) | $1 \times 10^{-8}$ |
| F1(50-Dimension) | $1 \times 10^{0}$ |
| F2(10-Dimension) | $1 \times 10^{-1}$ |
| F2(50-Dimension) | $1 \times 10^{3}$ |
| F3(10-Dimension) | $1 \times 10^{-12}$ |
| F3(50-Dimension) | $5 \times 10^{0}$ |
| F4(10-Dimension) | $1 \times 10^{1}$ |
| F4(50-Dimension) | $3 \times 10^{2}$ |
| F5(10-Dimension) | $1 \times 10^{-1}$ |
| F5(50-Dimension) | $1 \times 10^{2}$ |

It is noted that the different topologies used in the social network model for connection between individuals have the different effect on the efficiency of strategy two. There are three commonly used topologies supported in population-based EAs [40]. The efficiency of using the three topologies in Strategy two is discussed in the next chapter. The three topology types are:

- Ring topology: Each individual has two connections to other individuals in the population. (figure 4.6 b)
- Square topology: Each individual has four connections to other individuals in the population. (figure 4.6 c)

- Square topology: Each individual has connections to all other individuals in the population. (figure 4.6 a)

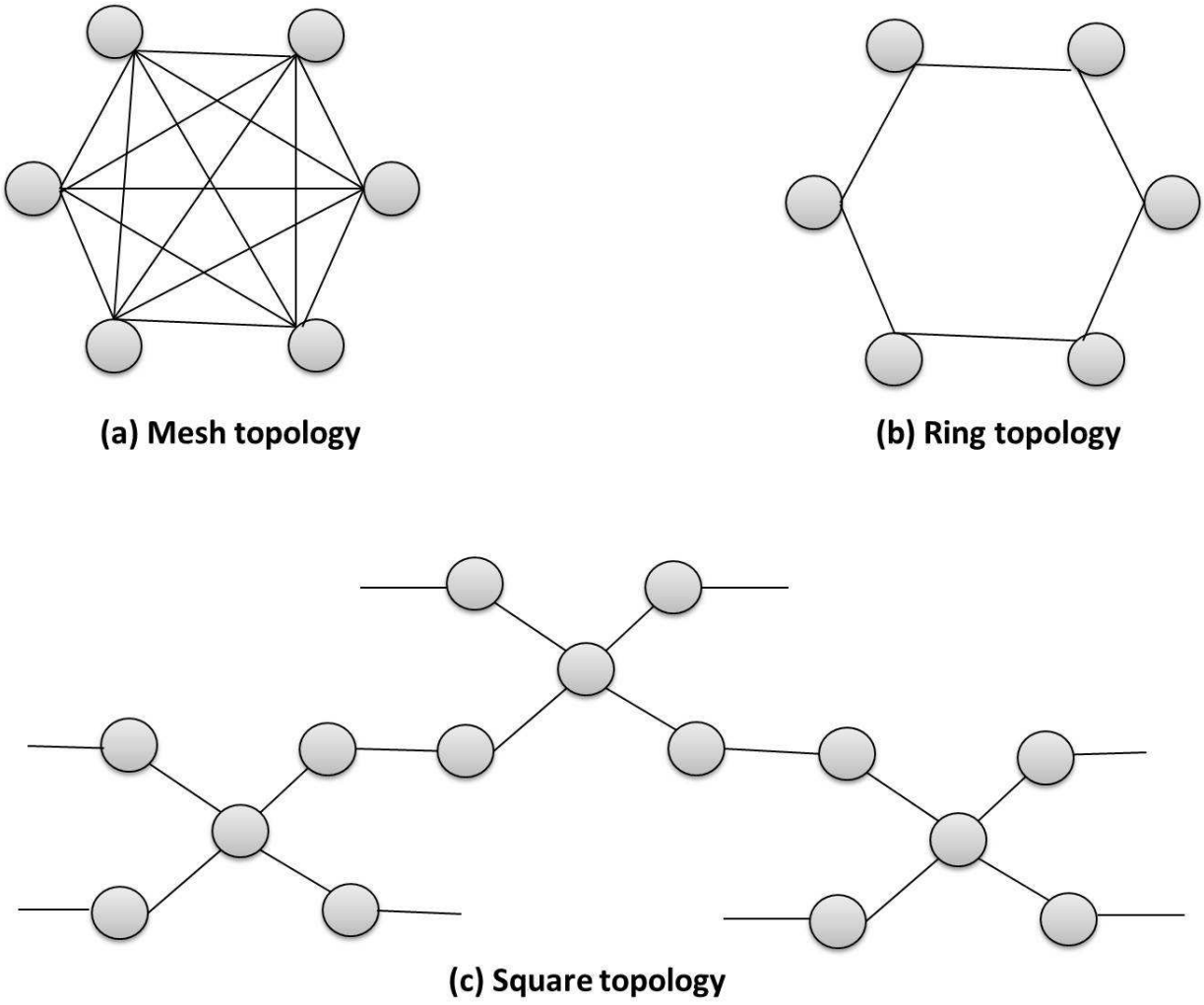


Fig. 4.6: Ring topology, mesh topology and square topology

## 4.3 Experimental Results and Analysis

### 4.3.1 M1 vs. M2 vs. M3

Table 4.3: Main parameters for M1 vs. M2 vs. M3

| **Parameter** | **Value** |
|---|---|
| Population size | 200 |
| Sub-population amount | 8 |
| Sub-population amount size | 25 |
| Selection proportion | 0.3 |
| $w_1$: weight parameter in Formula 18 | 0.7 |
| $w_2$: weight parameter in Formula 18 | 0.3 |
| Constant knowledge migration interval | 3 |
| Run times | 60 |
| Maximum iterations | 1000 |
| Dimension | 10 |

Table 4.4 show the test result of M1, M2, M3 using ring topology, M3 using square topology and M3 using mesh topology on benchmark optimization functions F1-F5. The best results got in 60 runs are typed in bold. It can be observed that the proposed two heuristic strategies in term of knowledge selection improve the performance of original MPCA in mean fitness value, standard deviation, and iteration number. Furthermore, all three M3s outperform M2 on the five test functions. For the comparison of the M3s adopting different topologies, M3(Square topology) can get the best fitness value and the smallest standard deviation in 1000 generations, and M3(Mesh topology) can use the minimal generations to reach the target fitness value on all the five optimization problems. However, the difference of test results between the M3s using different topologies is minor.

Figure 4.7 - Figure 4.11 show that over 1000 generations, how many times an individual select the three knowledge sources, respectively. Such as Figure 4.7 shown,

Table 4.4: Test result for M1 vs. M2 vs. M3

| | **F1** | **F2** | **F3** | **F4** | **F5** |
|---|---|---|---|---|---|
| **M1** | | | | | |
| Mean fitness | $4.6965 * 10^{-12}$ | $2.8535 * 10^{-3}$ | $1.6276 * 10^{-14}$ | $3.2569 * 10^{-1}$ | $2.3548 * 10^{-4}$ |
| Standard deviation | $1.4386 * 10^{-11}$ | $2.0140 * 10^{-2}$ | $1.4760 * 10^{-13}$ | $4.2503 * 10^{0}$ | $4.6988 * 10^{-3}$ |
| Iteration number | 293.4166 | 34.0000 | 432.4833 | 89.2833 | 189.0166 |
| **M2 (M1+S1)** | | | | | |
| Mean fitness | $2.2021 * 10^{-18}$ | $1.3759 * 10^{-3}$ | $7.2353 * 10^{-18}$ | $3.3358 * 10^{0}$ | $7.6621 * 10^{-5}$ |
| Standard deviation | $5.4823 * 10^{-17}$ | $1.8891 * 10^{-2}$ | $7.1158 * 10^{-17}$ | $9.5463 * 10^{0}$ | $9.1140 * 10^{-4}$ |
| Iteration number | 149.1000 | 25.2333 | 398.0833 | 77.7666 | 177.9833 |
| **M3 (M1+S2(Ring) )** | | | | | |
| Mean fitness | $1.7374 * 10^{-24}$ | $3.5455 * 10^{-6}$ | $3.7455 * 10^{-20}$ | $7.2541 * 10^{0}$ | $4.0259 * 10^{-6}$ |
| Standard deviation | $1.0090 * 10^{-23}$ | $6.1667 * 10^{-6}$ | $6.3029 * 10^{-19}$ | $5.8799 * 10^{0}$ | $5.2550 * 10^{-5}$ |
| Iteration number | 116.0333 | 23.0000 | 320.9833 | 72.4166 | 164.2000 |
| **M3 (M1+S2(Square))** | | | | | |
| Mean fitness | **$4.7279 * 10^{-26}$** | **$1.7333 * 10^{-6}$** | **$1.5679 * 10^{-20}$** | **$7.8965 * 10^{-1}$** | **$1.2533 * 10^{-6}$** |
| Standard deviation | **$1.8891 * 10^{-25}$** | **$2.1889 * 10^{-6}$** | **$5.6642 * 10^{-19}$** | **$2.3342 * 10^{-1}$** | **$2.3684 * 10^{-5}$** |
| Iteration number | 112.4333 | 21.8000 | 289.3500 | 70.8166 | 160.1333 |
| **M3 (M1+S2(Mesh))** | | | | | |
| Mean fitness | $2.7333 * 10^{-23}$ | $3.2273 * 10^{-5}$ | $6.2146 * 10^{-19}$ | $8.2064 * 10^{-1}$ | $5.2149 * 10^{-6}$ |
| Standard deviation | $1.5119 * 10^{-22}$ | $5.0152 * 10^{-5}$ | $4.2547 * 10^{-18}$ | $1.5580 * 10^{-1}$ | $8.1533 * 10^{-5}$ |
| Iteration number | **98.1500** | **18.3667** | **276.0166** | **69.7833** | **155.0333** |

for F1 test function, by average, an individual select normative knowledge 563 times, topographic knowledge 425 times and historic knowledge 12 times, in M3. Obviously, without knowledge selection strategies, the individual select almost same amount of three knowledge sources in original MPCA(M1). The two knowledge selection heuristics can drive the individuals to select more normative knowledge and topographic knowledge, and much less historic knowledge. This can improve performance of the MPCA, as normative knowledge, and topographic knowledge can give the individual much greater probability to find a better solution than historic knowledge. In the other word, historic is the weak knowledge compared with normative knowledge and topographic knowledge.

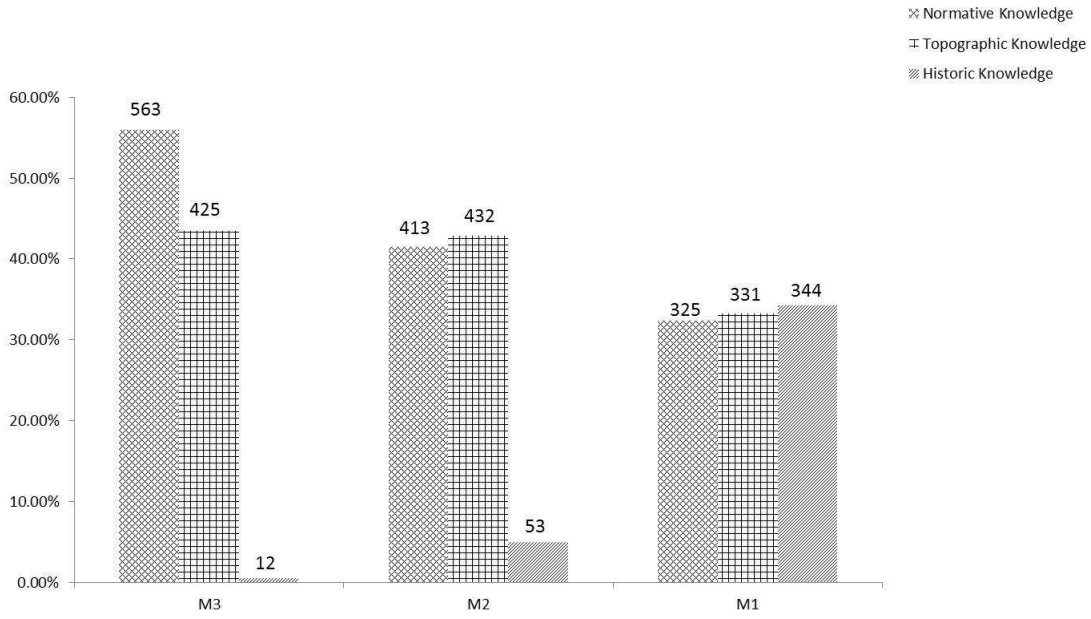


Fig. 4.7: Number of times three different knowledge sources being selected by individuals for F1

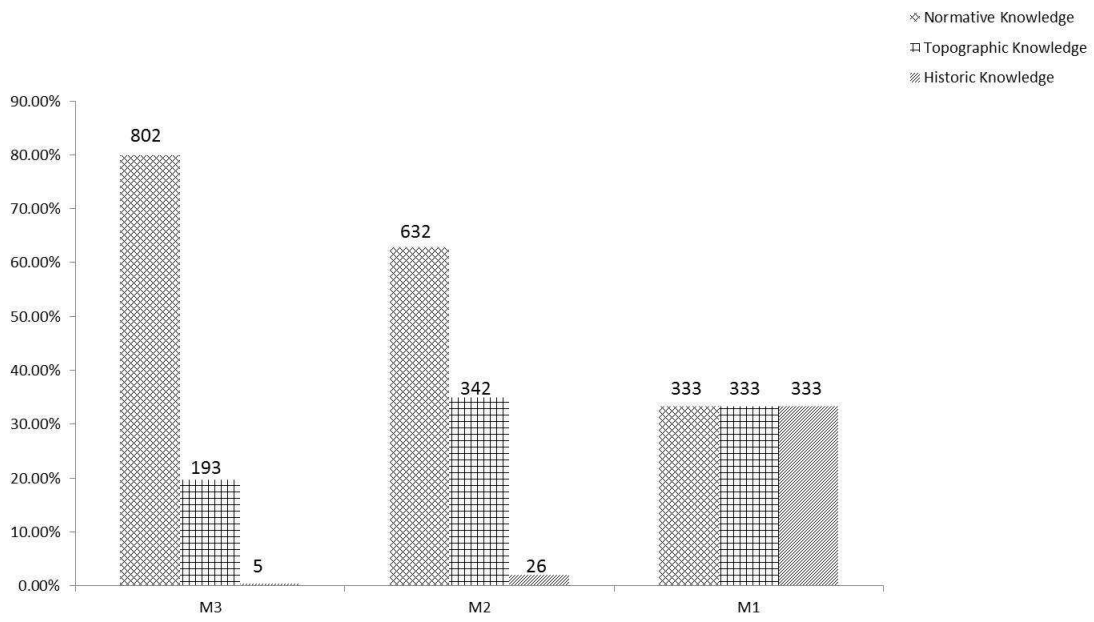


Fig. 4.8: Number of times three different knowledge sources being selected by individuals for F2

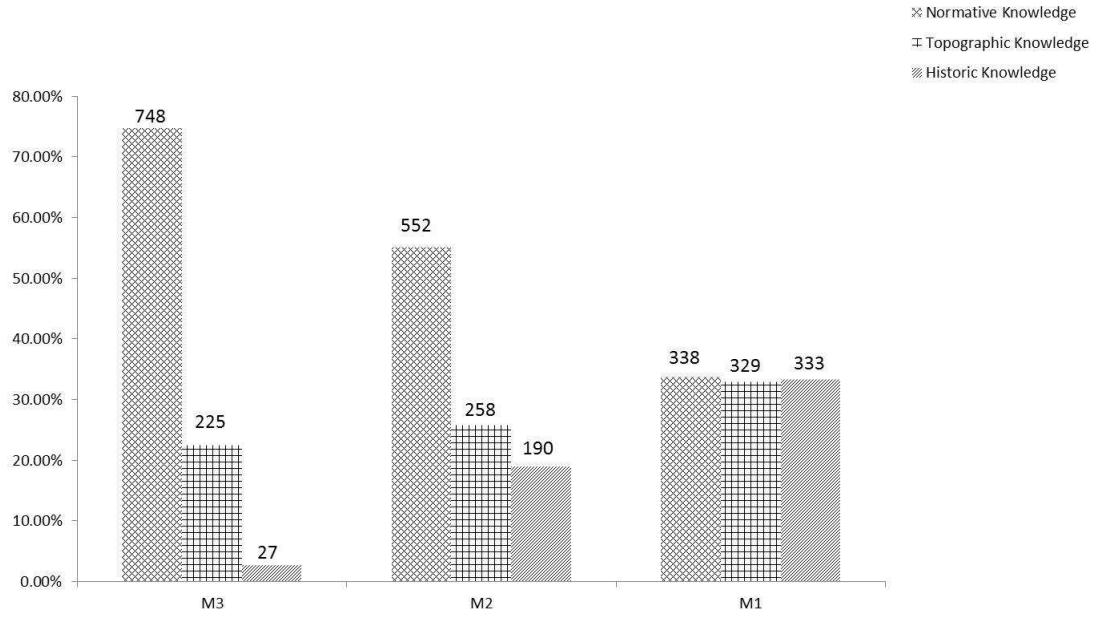


Fig. 4.9: Number of times three different knowledge sources being selected by individuals for F3

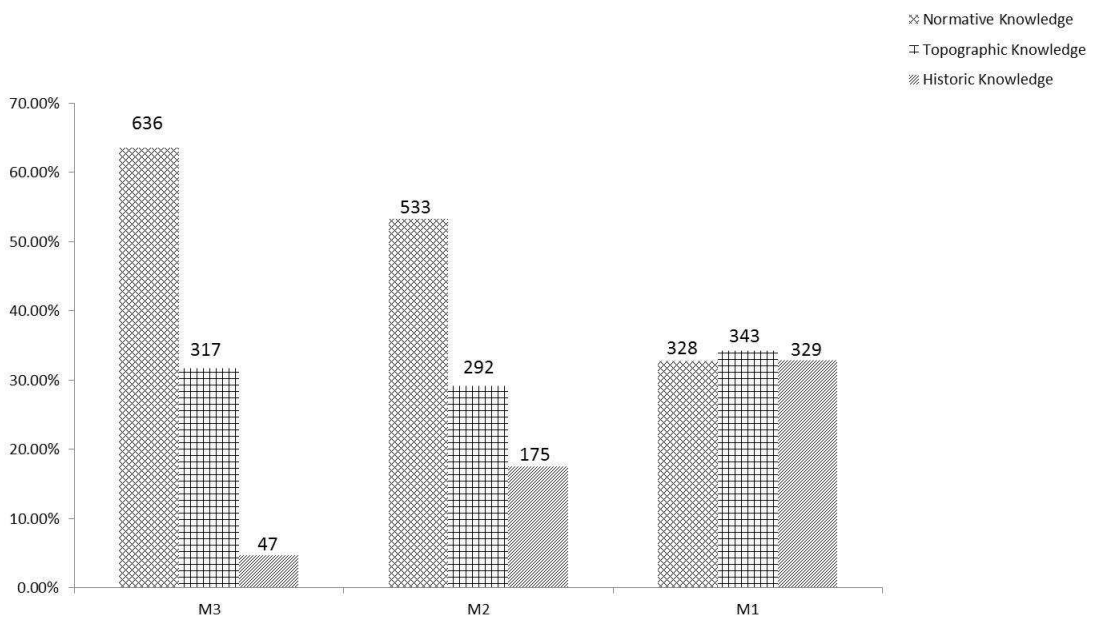


Fig. 4.10: Number of times three different knowledge sources being selected by individuals for F4

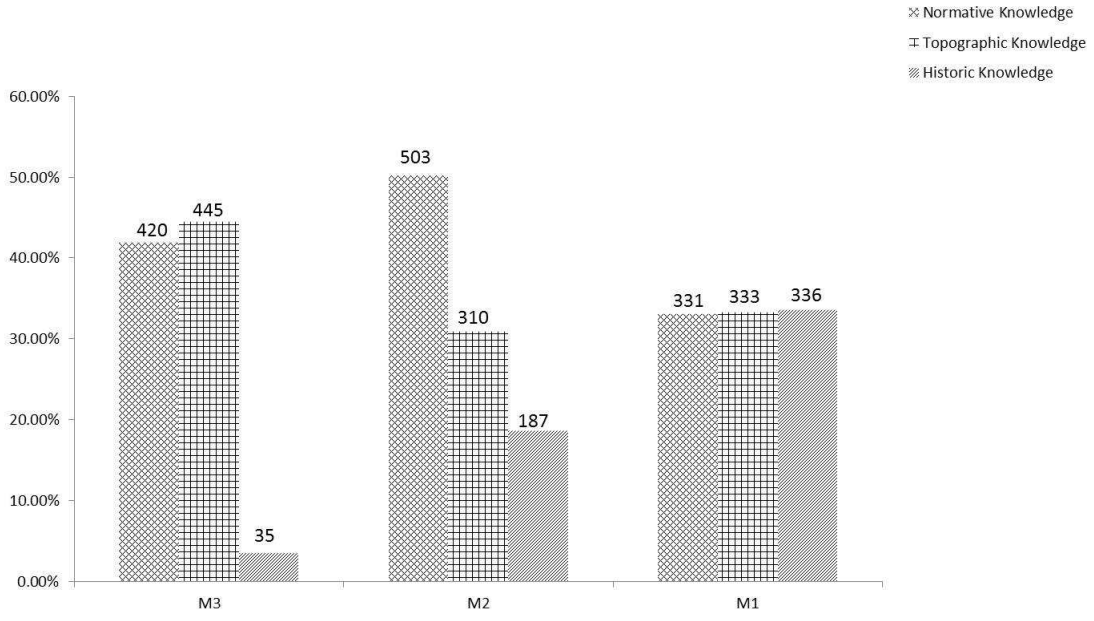


Fig. 4.11: Number of times three different knowledge sources being selected by individuals for F5

### 4.3.2 M1 vs. M4 vs. M5

Table 4.5: Main parameters for M1 vs. M4 vs. M5

| **Parameter** | **Value** |
|---|---|
| Population size | 200 |
| Sub-population amount | 8 |
| Sub-population amount size | 25 |
| Selection proportion | 0.3 |
| $w_1$: weight parameter in formula 18 | 0.7 |
| $w_2$: weight parameter in formula 18 | 0.3 |
| $\eta_{max}$ (formula 21) | 1000 |
| $\tau$ in formula22 | 0.01[39] |
| Run times | 60 |
| Maximum iteration | 1000 |
| Dimension | 10 |

Table 4.5 lists the main parameters used in the experiment of M1 vs. M4 vs. M5. To tune the parameter N in formula 21, we tested N at a interval of 0.02 from 1.01 to 2.00 on the five benchmark functions. Table 4.10 shows the test result of M1 with one knowledge migration interval, M1 with three knowledge migration intervals, M1 with six knowledge migration intervals, M2 and M3 on benchmark optimization functions F1-F5. The best results got in 60 runs are typed in bold. It can be seen that the proposed two heuristic strategies in term of knowledge migration interval can give the MPCA with constant knowledge migration interval slight promotion effects in mean fitness value, standard deviation, and iteration number. From the standard deviation value and iteration number value shown in the table, we can see that the stability and convergence speed of M5 is better than M4 in general. However, M4 can reach better optimal solutions in 1000 generations on most test functions(4 out of 5). Knowledge migrating at every generation makes the MPCA outperform the MPCAs with knowledge migrating at every three generations and at every six

generations in convergence speed. However, M1(constant interval = 1)has the worst global optimization ability among all the five algorithms listed in the table.

### 4.3.3 M1 vs. M6 vs. M7 vs. M8 vs. M9

In this section, in order to evaluate the efficiency of the proposed heuristic strategies to MPCA, all the heuristic strategies constitute four combinational strategies and compare with the original MPCA on both low-dimension test functions (10-dimension) and high-dimension test functions (50-dimension). Table 4.7 lists the main parameters used in the tests. The original MPCA in this set of tests have knowledge migrating at every three generations, constant knowledge migration interval of three generations shows the best performance in the experiments in Section 4.3.2.

Table 4.7: Main parameters for M1 vs. M6 vs. M7 vs. M8 vs. M9

| **Parameter** | **Value** |
|---|---|
| Population size (10-Dimension) | 200 |
| Sub-population amount (10-Dimension) | 8 |
| Sub-population amount size (10-Dimension) | 25 |
| Population size (50-Dimension) | 300 |
| Sub-population amount (50-Dimension) | 10 |
| Sub-population amount size (50-Dimension) | 30 |
| Selection proportion | 0.3 |
| $w_1$: weight parameter in formula 18 | 0.7 |
| $w_2$: weight parameter in formula 18 | 0.3 |
| $\tau$ in formula22 | 0.01 |
| $\eta_{max}$ (formula 21) | 1000 |
| Run times | 60 |
| Maximum iteration | 1000 |

As the test results shown from Table 4.8 and Table 4.9, all the combinational strategies can improve MPCA a lot. The MPCAs with combinational heuristic strate-

Table 4.6: Test result for M1 vs. M4 vs. M5

| | **F1** | **F2** | **F3** | **F4** | **F5** |
|---|---|---|---|---|---|
| **M1 (constant interval = 1)** | | | | | |
| Mean fitness | $6.7374 * 10^{-10}$ | $9.3654 * 10^{-3}$ | $5.6276 * 10^{-13}$ | $3.4896 * 10^{0}$ | $7.7471 * 10^{-4}$ |
| Standard deviation | $1.2258 * 10^{-9}$ | $8.2492 * 10^{-2}$ | $3.4760 * 10^{-13}$ | $7.5963 * 10^{0}$ | $8.2966 * 10^{-3}$ |
| Iteration number | 241.3166 | **30.7666** | 342.4833 | 80.2833 | 185.8333 |
| **M1 (constant interval = 3)** | | | | | |
| Mean fitness | $4.6965 * 10^{-12}$ | $2.8535 * 10^{-3}$ | $1.6276 * 10^{-14}$ | $3.2569 * 10^{-1}$ | $2.3548 * 10^{-4}$ |
| Standard deviation | $1.4386 * 10^{-11}$ | $2.0140 * 10^{-2}$ | $1.4760 * 10^{-13}$ | $4.2503 * 10^{0}$ | $4.6988 * 10^{-3}$ |
| Iteration number | 293.4166 | 34.0000 | 432.4833 | 89.2833 | 189.0166 |
| **M1 (constant interval = 6)** | | | | | |
| Mean fitness | $1.7374 * 10^{-11}$ | $6.3259 * 10^{-3}$ | $3.7455 * 10^{-14}$ | $1.2541 * 10^{0}$ | $6.3205 * 10^{-4}$ |
| Standard deviation | $1.0090 * 10^{-10}$ | $6.5210 * 10^{-2}$ | $5.3029 * 10^{-13}$ | $5.8799 * 10^{0}$ | $6.1497 * 10^{-3}$ |
| Iteration number | 325.8333 | 39.1833 | 551.7666 | 99.4166 | 195.6000 |
| **M4 (M1+S3)** | | | | | |
| Mean fitness | $4.7279 * 10^{-16}$ | $\mathbf{1.2346 * 10^{-3}}$ | $\mathbf{3.5679 * 10^{-15}}$ | $\mathbf{7.8965 * 10^{-2}}$ | $\mathbf{5.3278 * 10^{-5}}$ |
| Standard deviation | $1.8891 * 10^{-16}$ | $2.1745 * 10^{-2}$ | $6.6642 * 10^{-14}$ | $\mathbf{2.3342 * 10^{-1}}$ | $3.2077 * 10^{-4}$ |
| Iteration number | 240.6666 | 32.4833 | 334.7500 | 82.3666 | **185.3666** |
| **M5 (M1+S4)** | | | | | |
| Mean fitness | $\mathbf{2.7333 * 10^{-17}}$ | $2.6024 * 10^{-3}$ | $6.2146 * 10^{-15}$ | $8.2064 * 10^{-2}$ | $5.9936 * 10^{-5}$ |
| Standard deviation | $\mathbf{1.5119 * 10^{-17}}$ | $\mathbf{1.5692 * 10^{-2}}$ | $\mathbf{4.2547 * 10^{-14}}$ | $3.5580 * 10^{-1}$ | $\mathbf{2.4896 * 10^{-4}}$ |
| Iteration number | **235.1500** | 33.3667 | **320.9833** | **80.0166** | 186.1333 |

Table 4.8: Test result for M1 vs. M6 vs. M7 vs. M8 vs. M9(10-dimension)

| | **F1** | **F2** | **F3** | **F4** | **F5** |
|---|---|---|---|---|---|
| **M1** | | | | | |
| Mean fitness | $4.6965 * 10^{-12}$ | $2.8535 * 10^{-3}$ | $1.6276 * 10^{-14}$ | $3.2569* 10^{-1}$ | $2.3548* 10^{-4}$ |
| Standard deviation | $1.4386 * 10^{-11}$ | $2.0140 * 10^{-2}$ | $1.4760* 10^{-13}$ | $4.2503* 10^{0}$ | $4.6988* 10^{-3}$ |
| Iteration number | 293.4166 | 34.0000 | 432.4833 | 89.2833 | 189.0166 |
| **M6 (M1+S1+S3)** | | | | | |
| Mean fitness | $5.2364 * 10^{-30}$ | $2.9107 * 10^{-10}$ | $2.6656 * 10^{-21}$ | $2.7492 * 10^{-1}$ | $3.7452* 10^{-7}$ |
| Standard deviation | $3.2258 * 10^{-30}$ | $3.1702 * 10^{-9}$ | $6.1446 * 10^{-20}$ | $2.2777* 10^{-1}$ | $4.0288* 10^{-7}$ |
| Iteration number | 99.0166 | 20.9833 | 302.1833 | 71.5000 | 158.9833 |
| **M7 (M1+S1+S4)** | | | | | |
| Mean fitness | $1.3659 * 10^{-30}$ | $4.0323 * 10^{-10}$ | $3.9899 * 10^{-21}$ | $1.3131* 10^{-1}$ | $2.9634* 10^{-7}$ |
| Standard deviation | $2.3629 * 10^{-30}$ | $6.7421 * 10^{-10}$ | $5.2040 * 10^{-20}$ | $3.5618 * 10^{-1}$ | $7.5655* 10^{-7}$ |
| Iteration number | 95.5000 | 20.1833 | 316.8333 | 70.9833 | 149.4833 |
| **M8 (M1+S2+S3)** | | | | | |
| Mean fitness | $8.3621 * 10^{-43}$ | $2.6064 * 10^{-10}$ | $3.3359 * 10^{-22}$ | **$6.3367 * 10^{-2}$** | $9.3654* 10^{-8}$ |
| Standard deviation | $4.0996 * 10^{-42}$ | $7.2458 * 10^{-10}$ | **$2.4798 * 10^{-21}$** | **$4.2159 * 10^{-2}$** | **$8.9122* 10^{-8}$** |
| Iteration number | **89.7166** | 16.8666 | 209.7666 | 55.7500 | **98.8166** |
| **M9 (M1+S2+S4)** | | | | | |
| Mean fitness | **$1.6587 * 10^{-43}$** | **$1.8383 * 10^{-10}$** | **$1.7496 * 10^{-22}$** | $6.4896 * 10^{-2}$ | **$7.1015* 10^{-8}$** |
| Standard deviation | **$3.3397 * 10^{-42}$** | **$6.3373 * 10^{-10}$** | $2.8411 * 10^{-21}$ | $1.4420 * 10^{-1}$ | $2.6448* 10^{-8}$ |
| Iteration number | 91.8166 | **15.2833** | **199.4833** | **53.0166** | 112.7166 |

Table 4.9: Test result for M1 vs. M6 vs. M7 vs. M8 vs. M9 (50-dimension)

| | **F1** | **F2** | **F3** | **F4** | **F5** |
|---|---|---|---|---|---|
| **M1** | | | | | |
| Mean fitness | $3.7211 * 10^{-2}$ | $3.5469 * 10^{2}$ | $4.9960 * 10^{-1}$ | $5.2353 * 10^{1}$ | $9.5523 * 10^{-1}$ |
| Standard deviation | $6.3549 * 10^{-1}$ | $7.2149 * 10^{2}$ | $6.0463 * 10^{-1}$ | $1.5613 * 10^{1}$ | $5.2956 * 10^{-1}$ |
| Iteration number | 568.1333 | 121.7833 | 232.7666 | 159.7500 | 129.4166 |
| **M6 (M1+S1+S3)** | | | | | |
| Mean fitness | $2.1569 * 10^{-10}$ | $2.3658 * 10^{-2}$ | $2.8765 * 10^{-7}$ | $2.3120 * 10^{0}$ | $3.4716 * 10^{-3}$ |
| Standard deviation | $1.5896 * 10^{-9}$ | $2.8484 * 10^{-2}$ | $7.1092 * 10^{-6}$ | $3.7754 * 10^{0}$ | $3.2874 * 10^{-2}$ |
| Iteration number | 108.3166 | 55.8166 | 71.4833 | 124.0000 | 96.2666 |
| **M7 (M1+S1+S4)** | | | | | |
| Mean fitness | $1458 * 10^{-10}$ | $2.6548 * 10^{-2}$ | $2.6544 * 10^{-7}$ | $1.0104 * 10^{0}$ | $3.2298 * 10^{-3}$ |
| Standard deviation | $3.9596 * 10^{-9}$ | $8.5211 * 10^{-2}$ | $4.9077 * 10^{-6}$ | $3.6581 * 10^{1}$ | $6.2495 * 10^{-2}$ |
| Iteration number | 112.1833 | 69.6666 | 75.1833 | 149.9833 | 90.4833 |
| **M8 (M1+S2+S3)** | | | | | |
| Mean fitness | $4.7422 * 10^{-14}$ | $3.2896 * 10^{-4}$ | $3.8126 * 10^{-12}$ | $\mathbf{6.7633 * 10^{-1}}$ | $4.3927 * 10^{-4}$ |
| Standard deviation | $9.3221 * 10^{-13}$ | $1.7721 * 10^{-3}$ | $\mathbf{8.3908 * 10^{-11}}$ | $\mathbf{3.4463 * 10^{-1}}$ | $\mathbf{3.3821 * 10^{-3}}$ |
| Iteration number | **69.8333** | 36.8666 | 40.7666 | 130.7500 | **49.6000** |
| **M9 (M1+S2+S4)** | | | | | |
| Mean fitness | $\mathbf{3.6588 * 10^{-14}}$ | $\mathbf{2.6963 * 10^{-4}}$ | $\mathbf{2.6331 * 10^{-12}}$ | $7.6389 * 10^{-1}$ | $\mathbf{3.4922 * 10^{-4}}$ |
| Standard deviation | $\mathbf{1.0367 * 10^{-13}}$ | $\mathbf{1.2486 * 10^{-3}}$ | $8.5782 * 10^{-11}$ | $5.1098 * 10^{-1}$ | $6.1215 * 10^{-3}$ |
| Iteration number | 77.6833 | **34.8000** | **33.8166** | **129.3666** | 82.7166 |

gies can have better optimal solutions, better stability and faster convergence speed for most test functions, whether the function is unimodal or multi-modal, and whether high or low dimensional. The MPCA with knowledge selection based on social interaction and dynamic knowledge migration (M8), and MPCA with knowledge selection based on social interaction and population dispersion based knowledge migration interval (M9) show the best performance in the algorithms from M1 to M9 on both low-dimension test functions and high-dimension test functions. M8 and M9 have better performance than M6 and M7 in aspects of mean fitness value, standard deviation and iteration number to reach goal fitness value.

## 4.4 Discussion

Figure 4.12 illustrates the convergence performance in terms of mean fitness value of M1, M3 and M9 for F1(dimension = 10). Sphere function (F1) is the simplest unimodal test function. Therefore, it can be observed that M1, M3, and M9 quickly converge toward the optima, but M9 can get the best fitness value in 1000 iterations. It also can be observed that M9 converges faster than M3 over the second half of the evolution, as M9 is incorporated with the heuristic of knowledge migration interval that speeds up the convergence by migrating knowledge sources more frequently at latter stage of the evolution.

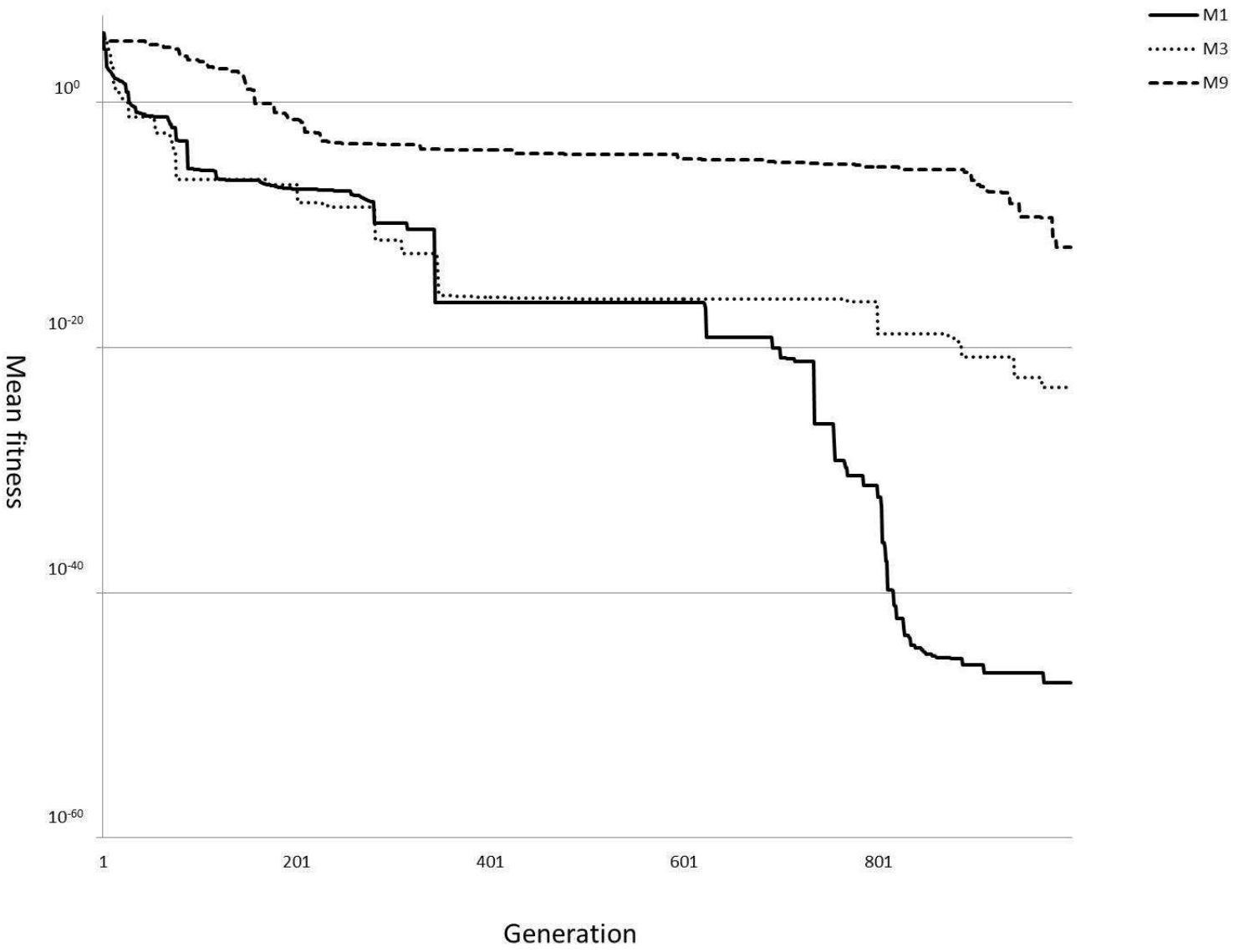


Fig. 4.12: Convergence performance of M1,M3 and M9 for F1(10-D)

Figure 4.13 demonstrates convergence performance of M1(constant knowledge migration interval = 1), M1(constant knowledge migration interval = 6) and M8 for Schwfel problem 1.2 function(F2) with 50 dimensions. For Schwfel problem 1.2 function, it is difficult to get the global optima, as the global minimum solution located in a long and narrow area. Therefore, the two M1s are trapped into the local optima. It can be seen from the figure, the heuristic strategies give M8 a better global opti-

mization ability than M1s even though a smaller knowledge migration interval makes M1(constant interval =1) drop quickly at first.

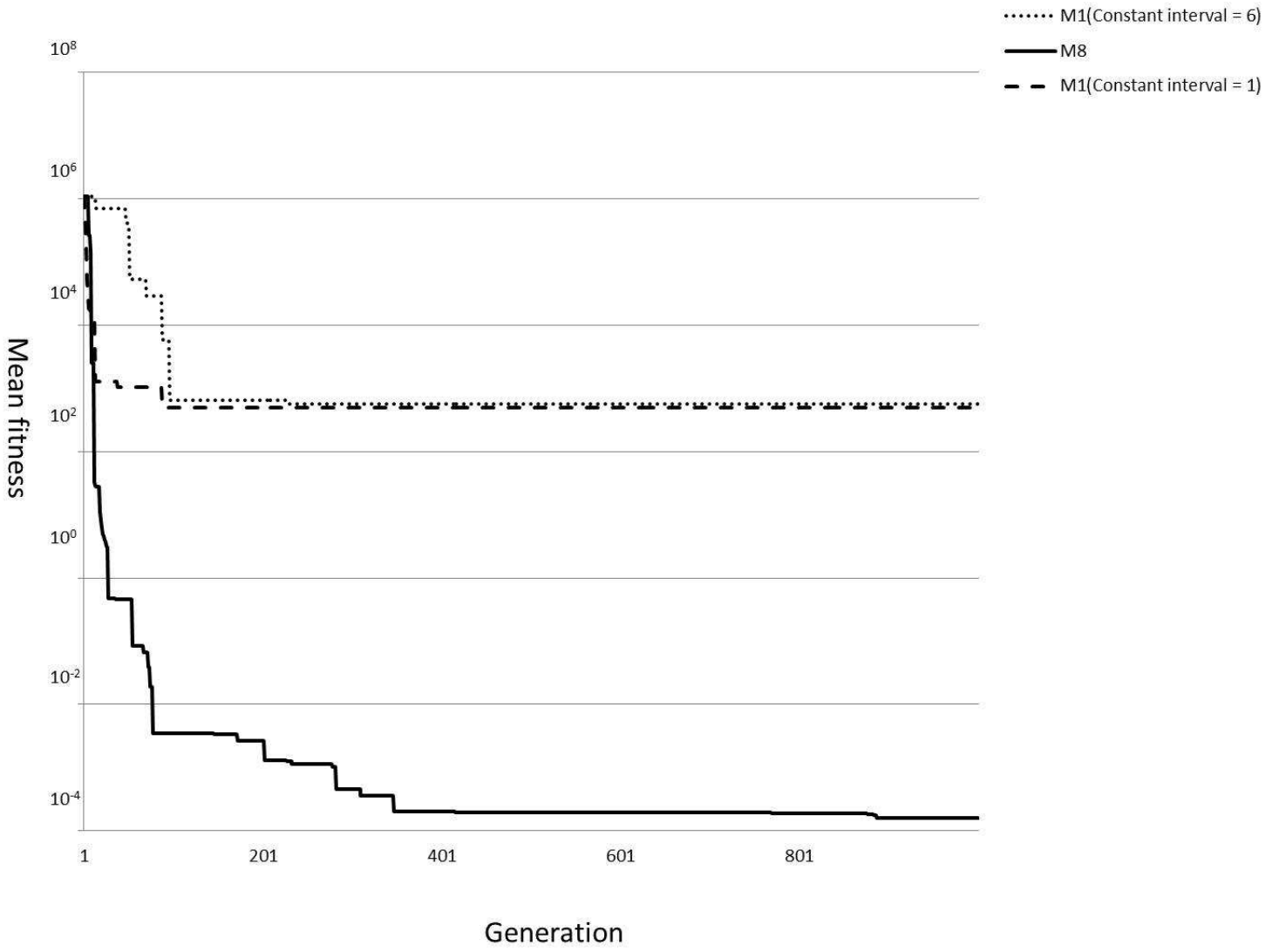


Fig. 4.13: Convergence performance of M1(constant knowledge migration interval = 1), M1(constant knowledge migration interval = 6) and M8 for F2(50-D)

Figure 4.14 shows the convergence curves of M1, M7, and M9 for the F3 function of 50-dimension. It is obviously from the figure, M7 and M9 have the faster drop

speed of optimization as well as the more precise optimal solutions than M1. M7 and M9 have different knowledge selection strategies and same knowledge migration interval strategies. It can be obtained that, the individuals can select more reasonable knowledge sources based on social interaction than individual memory, as M9 can reach better optimal solutions than M7.

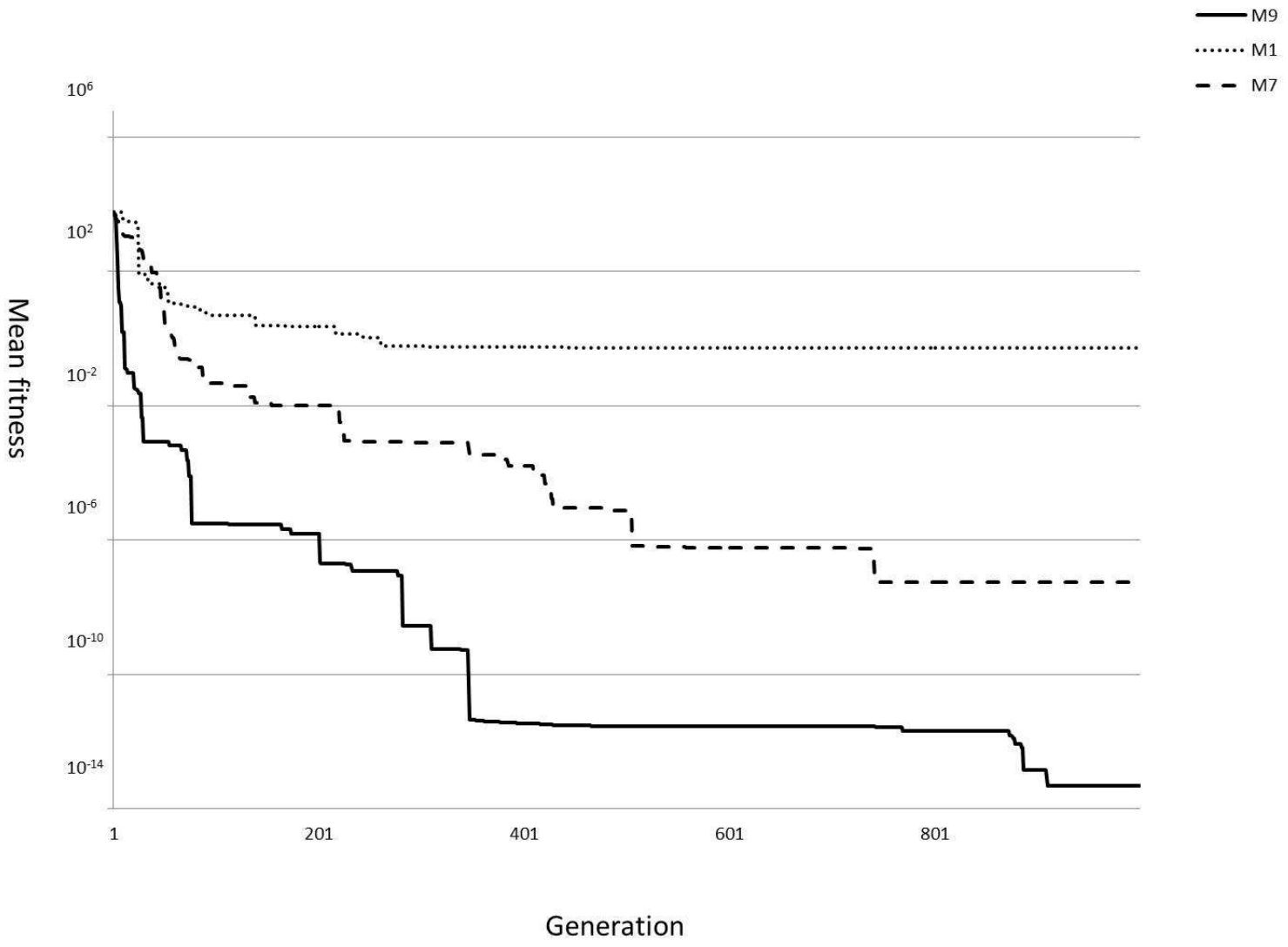


Fig. 4.14: Convergence performance of M1,M7 and M9 for F3 (50-D)

Figure 4.15 displays convergence performance of M1, M2 and M3 using square

topology for 50-dimensional F4 optimization function. It can be seen from the figure, three algorithms converge toward the same small range because rastrigin function has a large number of local minimas in a small region. Even though the three algorithms may converge to the local optima, M3 still uses the least generations to get target fitness value. This may prove that the strategy of knowledge selection based on social interaction is better than the strategy of knowledge selection based on individual memory.

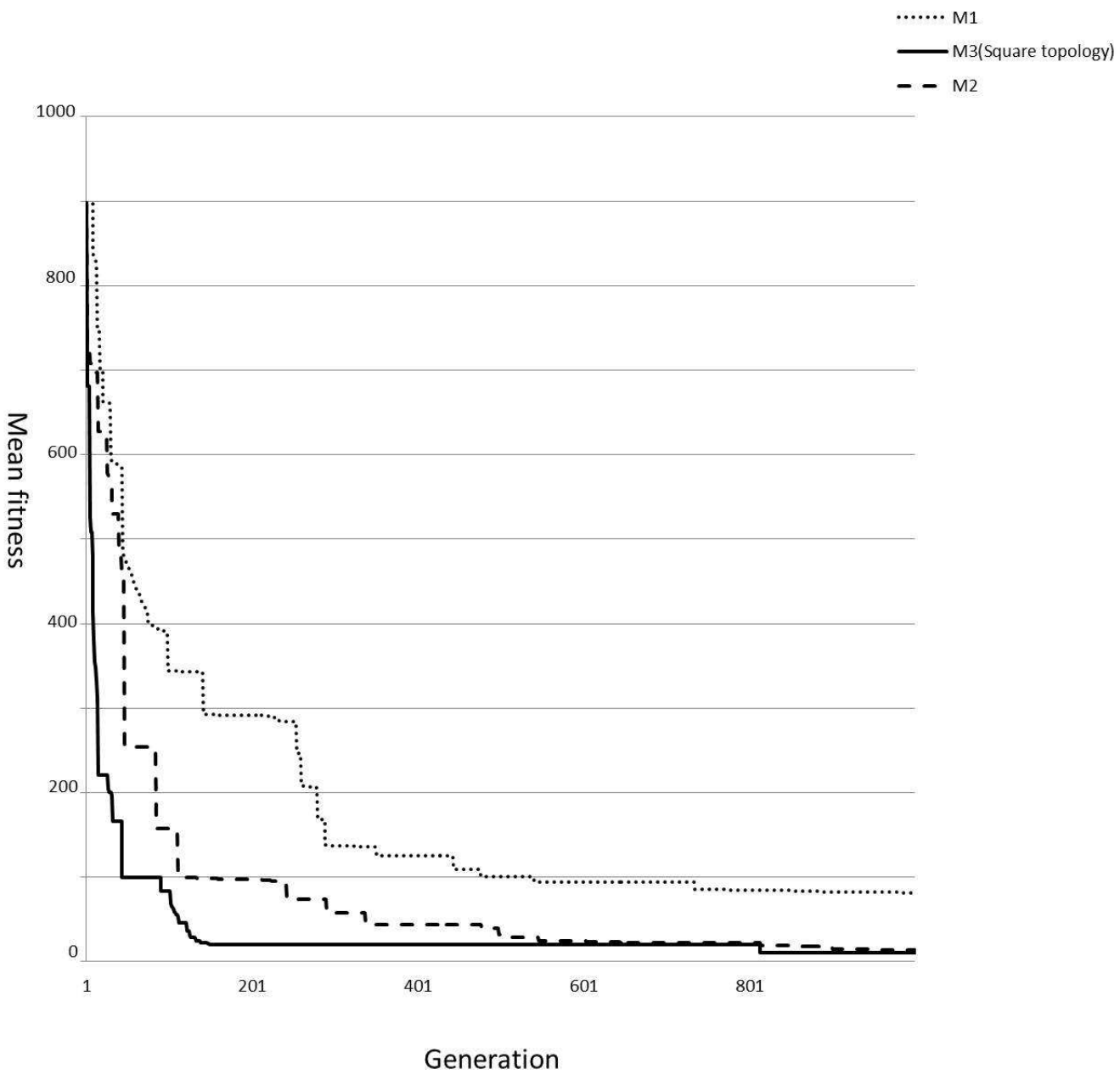


Fig. 4.15: Convergence performance of M1,M2 and M3 for F4(10-D)

Figure 4.16 illustrates the fitness value curves of M1, M8 and M9 over 1000 generation for F5(10-dimension). As we can see from the figure, M8 and M9 get close best fitness value in 1000 generations. M8 has the faster speed of convergence during the first half of evolution and M9 converges more quickly during the second half of evo-

lution. That is because M8 and M9 have different strategies for knowledge migration interval.

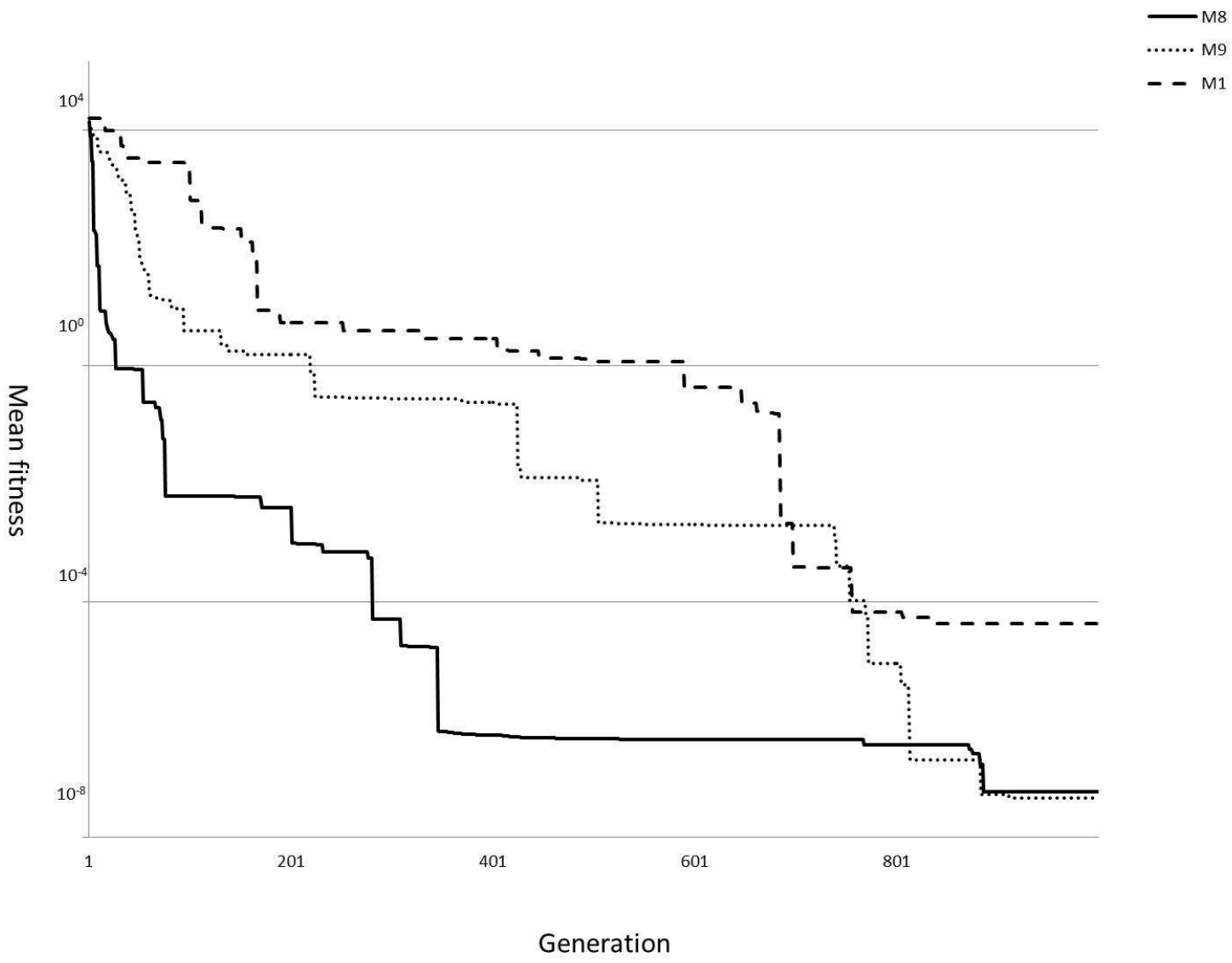


Fig. 4.16: Convergence performance of M1, M8 and M9 for F5(10-D)

## 4.5 A Real-World Application

In this subsection, the proposed MPCA is applied on a real-life problem. Spring design is a mechanical design problem [41] to minimize the weight of a compression spring (Fig.5.1). Design of the spring has constraints including shear stress, minimum deflection, surge frequency,and limits on outside diameter and on design variables. The design variables are mean coil diameter (x1), wire diameter (x2), and the number of active coils (x3), along with four inequality constraints[6]. The mathematical expression of this problem is as follows[6]:

$$\text{Minimize: } f(x) = (x_3 + 2)x_1x_2^2 \tag{23}$$

$$\text{Subject to: } \begin{array}{l} 1 - \dfrac{x_1^3 x_3}{71785x_2^4} \leq 0 \\ \dfrac{4x_1^2 - x_1x_2}{12566(x_1x_2^3 - x_1^3x_3)} - \dfrac{1}{5108x_2^2} - 1 \leq 0 \\ 1 - \dfrac{140.45x_2}{x_1^3x_3} \leq 0 \\ \dfrac{x_1 + x_2}{1.5} - 1 \leq 0 \end{array} \tag{24}$$

with the following limits on variables: $0.25 \leq x_1 \leq 1.3$, $0.05 \leq x_2 \leq 2.0$, and $2 \leq x_3 \leq 15$.

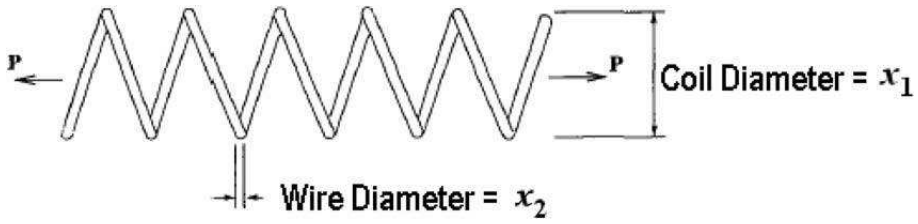


Fig. 4.17: Spring design problem[6]

Table shows the simulation results for the spring design problem using M9 (the MPCA with strategy two and strategy four) and M1(MPCA without the proposed heuristic). The setting of the algorithm is shown as Table 4.10 The decision variable

Table 4.10: Test result for spring design problem

| Algorithms | x1 | x2 | x3 | f(x) |
|---|---|---|---|---|
| M1(without heuristics) | 0.398 | 0.053 | 9.821 | 0.0132 |
| M9(S2+S4) | 0.352 | 0.050 | 11.716 | 0.0120 |

values and optimized solution are summarized in the table 4.11. As can be seen from the table, the MPCA with strategy two and strategy four (M9) finds the better optimum solution compared to the MPCA without heuristics(M1).

Table 4.11: Main parameters for spring design problem

| **Parameter** | **Value** |
|---|---|
| Population size | 200 |
| Sub-population amount | 8 |
| Sub-population amount size | 25 |
| Selection proportion | 0.3 |
| Run times | 20 |
| Maximum iterations | 1000 |

# 5 CONCLUSIONS AND FUTURE WORK

Our target is to use heuristics to improve MPCA. Four heuristic strategies are put forward in this thesis according to the weaknesses the existing MPCAs have, which are stochastic knowledge selection, and constant knowledge migration interval. We take a suit of benchmark optimization functions to test the efficiency of separate heuristics and combinational heuristics. The experimental results show that, MPCA with the heuristics have more efficiency in optimizing low-dimensional and unimodal optimization problems. The combinational strategies can promote the MPCA most in optimizing the unimodal function F1 over 1000 iteration.

M2(M1+S1): Individuals select knowledge sources based on individual memory in the M2 algorithm. Test results indicate that individuals can select the more reasonable knowledge sources during evolution, so as to improve the efficiency of the original MPCA (M1).

M3(M1+S2): Individuals select knowledge sources based on social interaction in the M3 algorithm. Different topologies of the social network in the population have different effects on the M3, and the M3 using square topology can get the best solutions compared to other topologies. M3 outperforms M2 in aspects of convergence speed, stability, and precision of solutions on the five test functions.

M4(M1+S3) and M5(M1+S4): Knowledge migrates among the sub-populations at dynamic intervals in the M4 algorithm. Alternately, with the M5 algorithm, knowledge migrates among the sub-populations at intervals based on the population dispersion. Both M4 and M5 can improve the original MPCA on 10-dimensional functions F1-F5 in mean fitness and standard deviation, but the performance of M2 and M3 is better than M4 and M5.

M6(M1+S1+S3), M7(M1+S1+S4), M8(M1+S2+S3) and M9(M1+S2+S4): Individuals select knowledge sources based on individual memory and knowledge migrate among the sub-populations at dynamic intervals in the M6 algorithm. With the

M7 algorithm, individuals select knowledge sources based on individual memory and knowledge migrate among the sub-populations at intervals based on the population dispersion in this algorithm. Alternately, the M8 algorithm sees individuals select knowledge sources based on social interaction, and knowledge migrates among the sub-populations at dynamic intervals in this algorithm. With the M9 algorithm, individuals select knowledge sources based on social interaction and knowledge migrates among the sub-populations at intervals based on the population dispersion in this algorithm. The four MPCAs with the proposed combinational heuristics can get better and more stable solutions on low-dimensional functions F1 to F5 compared to the four MPCAs with the single heuristic strategy.

We have also applied M9(MPCA with S2 and S4) on a real-world optimization problem, which demonstrates better performance compared to the MPCA without our proposed strategies.

In the future work, we can improve the testing to include more benchmark functions and dynamic environment can be tested as well.

# REFERENCES


[1] Sun, Y., Zhang, L., Gu, X.: A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. Neurocomputing **98** (2012) 76–89

[2] Lauren, D.: What is culture. `https://disqus.com/home/channel/thewriteway/discussion/channel-thewriteway/multicultural_behavior/` (2015)

[3] Elbeltagi, E., Hegazy, T., Grierson, D.: Comparison among five evolutionary-based optimization algorithms. Advanced engineering informatics **19**(1) (2005) 43–53

[4] Reynolds, R.G.: An introduction to cultural algorithms. In: Proceedings of the third annual conference on evolutionary programming, Singapore (1994) 131–139

[5] Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. KanGAL report **2005005** (2005)

[6] Daneshyari, M., Yen, G.G.: Constrained multiple-swarm particle swarm optimization within a cultural framework. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on **42**(2) (2012) 475–490

[7] Lovbjerg, M.: Improving particle swarm optimization by hybridization of stochastic search heuristics and self-organized criticality. Master's thesis, Department of Computer Science, University of Aarhus (2002)

[8] Tylor, E.B.: Primitive culture: researches into the development of mythology, philosophy, religion, art, and custom. Volume 2. Murray (1871)

[9] Coelho, L.D.S., Mariani, V.C.: An efficient particle swarm optimization approach based on cultural algorithm applied to mechanical design. In: Evolutionary Computation, 2006. CEC 2006. IEEE Congress on, IEEE (2006) 1099–1104

[10] Reynolds, R., Al-Shehri, H.: The use of cultural algorithms with evolutionary programming to control the data mining of large-scale spatio-temporal databases. In: Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on. Volume 5., IEEE (1997) 4098–4103

[11] Jin, X., Reynolds, R.G.: Data mining using cultural algorithms and regional schemata. In: Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings. 14th IEEE International Conference on, IEEE (2002) 33–40

[12] Rychtyckyi, N., Reynolds, R.G.: Using cultural algorithms to improve performance in semantic networks. In: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. Volume 3., IEEE (1999)

[13] Guo, Y.n., Cheng, J., Cao, Y.y., Lin, Y.: A novel multi-population cultural algorithm adopting knowledge migration. Soft computing **15**(5) (2011) 897–905

[14] Holland, J.H.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press (1992)

[15] Golberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addion wesley **1989** (1989)

[16] Kennedy, J.: Particle swarm optimization. In: Encyclopedia of Machine Learning. Springer (2010) 760–766

[17] Kennedy, J., Kennedy, J.F., Eberhart, R.C., Shi, Y.: Swarm intelligence. Morgan Kaufmann (2001)

[18] Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on, IEEE (1998) 69–73

[19] Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **26**(1) (1996) 29–41

[20] Dorigo, M., Gambardella, L.M.: Ant colonies for the travelling salesman problem. BioSystems **43**(2) (1997) 73–81

[21] Rajendran, C., Ziegler, H.: Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. European Journal of Operational Research **155**(2) (2004) 426–438

[22] Merkle, D., Middendorf, M., Schmeck, H.: Ant colony optimization for resource-constrained project scheduling. Evolutionary Computation, IEEE Transactions on **6**(4) (2002) 333–346

[23] Fogel, L.J.: Autonomous automata. Industrial Research **4**(2) (1962) 14–19

[24] Dawkins, R.: The selfish gene. Number 199. Oxford university press (2006)

[25] Merz, P., Freisleben, B.: A genetic local search approach to the quadratic assignment problem. In: Proceedings of the 7th international conference on genetic algorithms, Citeseer (1997) 1–1

[26] Liong, S.Y., Atiquzzaman, M.: Optimal design of water distribution network using shuffled complex evolution. Journal of The Institution of Engineers, Singapore **44**(1) (2004) 93–107

[27] Eusuff, M.M., Lansey, K.E.: Optimization of water distribution network design using the shuffled frog leaping algorithm. Journal of Water Resources Planning and Management **129**(3) (2003) 210–225

[28] Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. Evolutionary Computation, IEEE Transactions on **1**(1) (1997) 67–82

[29] Gao, F., Liu, H., Zhao, Q., Cui, G.: Hybrid model of genetic algorithm and cultural algorithms for optimization problem. In: Simulated Evolution and Learning. Springer (2006) 441–448

[30] Wang, Y.S., Ai, J.B., Shi, Y.J., Teng, H.F.: Cultural-based particle swarm optimization algorithm. JOURNAL-DALIAN UNIVERSITY OF TECHNOLOGY **47**(4) (2007) 539

[31] Chung, C.J.: Knowledge-based approaches to self-adaptation in cultural algorithms. (1997)

[32] Jin, X., Reynolds, R.G.: 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., IEEE (1999)

[33] Saleem, S.M.: Knowledge-based solution to dynamic optimization problems using cultural algorithms. Wayne State University (2001)

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

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

[36] Peng, B.: Knowledge and population swarms in cultural algorithms for dynamic environments. (2005)

[37] Hinterding, R.: Gaussian mutation and self-adaption for numeric genetic algorithms. In: Evolutionary Computation, 1995., IEEE International Conference on. Volume 1., IEEE (1995) 384

[38] GUO, Y.n., CHENG, J., CAO, Y.y., LIU, D.d.: Multi-population particle swarm cultural algorithms adopting chaotic knowledge migration. Control Theory & Applications **9** (2011) 003

[39] Lis, J.: Parallel genetic algorithm with the dynamic control parameter. In: Evolutionary Computation, 1996., Proceedings of IEEE International Conference on, IEEE (1996) 324–329

[40] Reynolds, R.G., Ali, M.: Computing with the social fabric: The evolution of social intelligence within a cultural framework. Computational Intelligence Magazine, IEEE **3**(1) (2008) 18–30

[41] Daneshyari, M., Yen, G.G.: Talent based social algorithm for optimization. In: Congress on evolutionary computation. (2004)

# VITA AUCTORIS

| | |
|---|---|
| NAME: | Xinyu He |
| PLACE OF BIRTH: | Fujian, China |
| YEAR OF BIRTH: | 1988 |
| EDUCATION: | Fujian University of Technology, Fujian, China |
| | Bachelor of Engineering, Computer Science 2008-2012 |
| | University of Windsor, Windsor ON, Canada |
| | Master of Science, Computer Science 2012-2015 |