EFFECT OF THE NUMBER OF PARENTS IN MULTI-PARENT GENETIC ALGORITHMS

BY

SENG PAN THAT PANN PHYU

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)

SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY

THAMMASAT UNIVERSITY

ACADEMIC YEAR 2016

EFFECT OF THE NUMBER OF PARENTS IN MULTI-PARENT GENETIC ALGORITHMS

BY

SENG PAN THAT PANN PHYU

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)

SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY

THAMMASAT UNIVERSITY

ACADEMIC YEAR 2016

EFFECT OF THE NUMBER OF PARENTS IN MULTI-PARENT GENETIC ALGORITHMS

A Thesis Presented

By

Seng Pan That Pann Phyu

Submitted to

Sirindhorn International Institute of Technology

Thammasat University

In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)

Approved as to style and content by

Advisor and Chairperson of Thesis Committee ______________________

(Asst. Prof. Dr. Gun Srijuntongsiri)

Committee Member ______________________

(Assoc. Prof. Dr. Bunyarit Uyyanonvara)

Committee Member ______________________

(Dr. Chanjira Sinthanayothin)

FEBRUARY 2017

Abstract

EFFECT OF THE NUMBER OF PARENTS IN MULTI-PARENT GENETIC ALGORITHMS

by

SENG PAN THAT PANN PHYU

Bachelor of Computer Technology, University of Computer Studies Mandalay, 2011
Master of Computer Technology, University of Computer Studies Mandalay, 2016
Master of Science (Engineering and Technology), Sirindhorn International Institute of Technology, Thammasat University, 2017

Genetic algorithm (GA) has been successfully applied to many numerical optimization problems in history. Multi-parent genetic algorithm (MPGA) is a generalized genetic algorithm. The crossover operator in MPGA uses more than two parents, differs from the original genetic algorithm, for the transformation of genetic information. Since MPGA has been increasing its interest in the family of genetic algorithms, it becomes an interesting algorithm to improve the solutions better than the traditional genetic algorithm. In this study, the effect of the number of parents in MPGA is investigated for solving ten multimodal high dimension benchmark functions and the problem of shuttle bus routing system (SBRS) in Thammasat University (Rangsit Campus). The experiment proves that using more parents in genetic algorithm yield better solution than the traditional genetic algorithm without taking too much on computation time.

Keywords: Genetic algorithm, multi-parent genetic algorithm, numerical optimization problems, shuttle bus routing system.

Acknowledgements

First of all, I would like to express my sincerest gratitude to my advisor Asst: Prof Dr. Gun Srijuntongsiri for his precious advises, suggestions and understanding throughout this study. Whenever I ran into a trouble and had a question about my research, he was always open, patient and steered me in the right direction. Being his advisee is very grateful and it would be impossible to complete this study without his guidance and support.

I would like to state my deepest appreciation to Assoc. Prof Dr. Bunyarit Uyyanonvara and Dr. Chanjira Sinthanayothin for their astonishing comments and suggestions in every single seminar and for their willingness to serve as the members of the thesis committee.

I would like to thank Prof. Dr. Thanaruk Theeramunkong for providing the data of shuttle buses from Thammasat University (Rangsit Campus).

I would also like to put my appreciation to Sirindhorn International Institute of Technology (SIIT), Thammasat University for providing Excellent Foreign Students (EFS) Scholarship.

I would like to acknowledge Center of Excellence in Intelligent Informatics, Speech and Language Technology and Service Innovation (CILS) from Thammasat University, Thailand for offering a partial research support.

In addition, I also thanks to all of my colleagues for their encouragement, share knowledges, suggestions, helps and care each other throughout this study.

Finally, I would like to express my profound gratitude to my beloved parents, U Aung Mya and Daw Ah Mar Kyi, and all the family members for their great encouragement, understanding, financial support, prayers and everything.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 General

Genetic algorithm (GA) is a search technique, which is inspired by the process of biological reproduction system. Genetic algorithm was developed by John Holland and his students in the 1970s according to the Darwinian notion of the survival of the fittest [14]. Genetic algorithm allows the population as a search space to optimize the fittest solution by the iterative process of randomly selected chromosomes and reproduces the better chromosomes in each generation.

Genetic algorithm (GA), a sub class of evolutionary algorithms (EAs), have been efficiently exploited for its robustness in many optimization problems [13]. Evolutionary algorithms are population-based metaheuristic optimization algorithms which are inspired by the nature of biology mechanisms such as genetic algorithms, evolution strategies and evolutionary programming [3]. Among these algorithms, genetic algorithm has been the most extensively investigated algorithm in the history of evolutionary optimization [5]. The common idea of all evolutionary algorithms is the iterative process of selecting the fittest solution in the boundary of search space with the variation of operators. The operators cause the natural selection process which modifies the candidates and choses the better fittest. The evolutionary process makes the better fittest solution in the search space. Unlike the other evolutionary algorithms, genetic algorithm has binary string representation for the candidate solutions. The string representation of genetic algorithm is powerful for mimicking the gene transformation of natural selection.

Since genetic algorithm successfully applied for solving optimization problems, there are many different types of genetic algorithms were evolved with different gene transformation methods. The results for each method are varied due to the optimization problems and gene selection process. Optimal solutions for different kinds of real world problems are still remain in the optimization area.

1.2 The concept of Genetic Algorithm

The routine of original genetic algorithm is straightforward. The genetic algorithm starts with a randomized initial population. In the terminology of genetic algorithm, the variables are called *genes*. In order to model the genes recombination, the variables are encoded as binary strings. A *chromosome* represents as a collection of *genes*. The size of population, genes, chromosomes and the rate of crossover and mutation are predefined before the genetic algorithm start. An example of binary chromosome can be represented as follow:

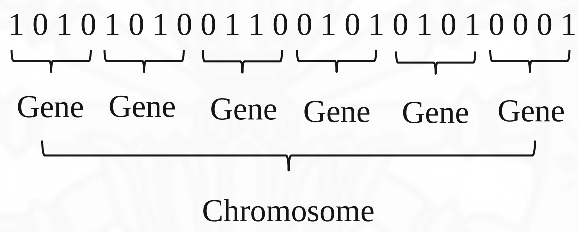

Figure 1.1 The representation of a binary chromosome

The optimization problem is called fitness function or objective function. Upon the type of optimization problems, the fitness function searches the minimize or maximize fitness value. The performance of genetic algorithm varies according to the problems and its nature. The genetic algorithm involves three major evolutionary mechanisms: selection, crossover and mutation. From these three mechanisms, the crossover plays the most important role in the process of genetic algorithm. The genetic algorithm ends its routine when the fitness value reaches to the optimal solution or a predefined number of generations are operated.

Selection, the first mechanism in the genetic algorithm, is a process in which a certain number of individual chromosomes are selected according to the fitness values and store into a mating pool. The mating pool is a temporary storage where the fittest chromosomes are selected by the selection technique. The traditional selection technique selects the best fittest chromosomes from the population and copies them into the mating pool. For example, the four best chromosomes are selected from the initial population according to their fitness values and keeps in the mating pool as listed in the Table 1.1. There are many other selection techniques. Some of which are based on the percentage of fitness values, best fitness of random selection, partially select from the best fittest individuals and so on.

Table 1.1: Sample of chromosome representation and fitness values

No.	Chromosome	Fitness Value
1	101101	55
2	100010	49
3	100101	34
4	101010	18

After the selection process, the crossover proceeds in two steps. First, two chromosomes are randomly selected from the mating pool. The selected chromosomes are regarded as the parent chromosomes to reproduce new child chromosomes or offspring. Second, each of two parents chromosomes are crossing over one another to reproduce offspring. For example, consider the chromosome number 2 and 3 are chosen as the parent chromosomes and crossing over both chromosomes according to the randomly generated single crossover point as shown in Figure 1.2. Then the new offspring are updated into the mating pool. The process of crossover process repeats until all new chromosomes are updated in the mating pool.

Parent 1	**1 0 0 | 0 1 0**
Parent 2	1 0 0 | 1 0 1
Offspring 1	**1 0 0** 1 0 1
Offspring 2	1 0 0 **0 1 0**

Figure 1.2 Simple crossover process with single crossover point

Mutation is the last mechanism in the process genetic algorithm. After crossover reproduced the new offspring by mixing the gene information from their parents, the mutation proceeds to change some random gene information in the chromosome from the mating pool. The process of mutation consists of two steps. First, a random variable, *r* is generated for each single bit of all chromosomes in the mating pool. Second, if *r* is less than the mutation rate *p*, which is predefined before, the current bit is flipped to its opposite value. Therefore, all new chromosomes are updated in the mating pool. A sample of mutation for an offspring from Figure 1.1 is demonstrated in Figure 1.2.

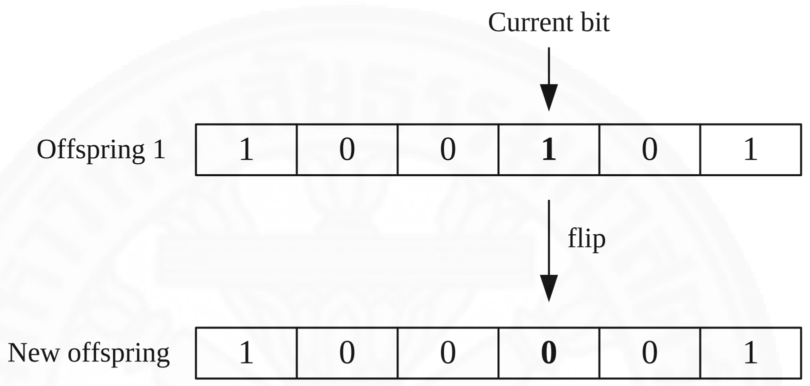

Figure 1.2 Mutation

By changing some of gene information in the mating pool, the mutation may affect the current solutions not to converge in the local optimum. The illustration for the effect of mutation as shown in Figure 1.3.

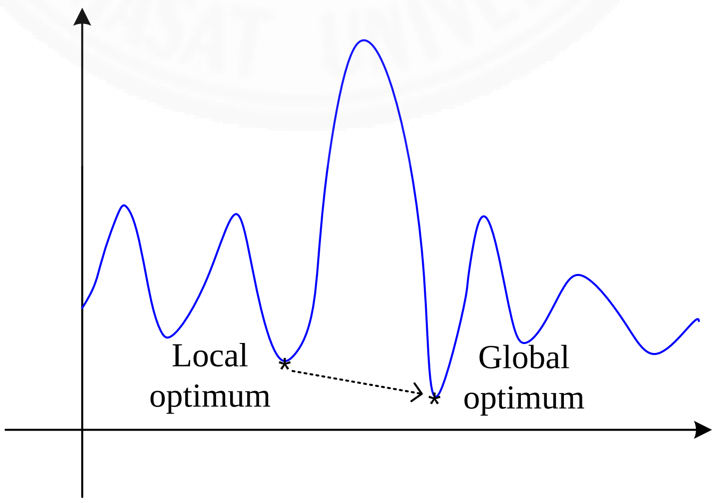

Figure 1.3 Effect of mutation

All updated chromosomes in the mating pool evaluates the fitness values to decide whether the best fitness value for current generation meets with a desire optimal solution or not. If the solution does not meet the optimal solution, repeats the genetic algorithm from the selection process. Otherwise, the iteration of generations in genetic algorithm stops its process and produces the best solution from whole generations. The most common stopping criteria in the traditional genetic algorithm are:

- The best fitness value in the current generation does not change in many generation.
- The genetic algorithm reaches a certain number of generations.

1.3 Statement of problems

The traditional genetic algorithm is easy to understand, but comprehensive. The selection chooses the best candidates from all possible solution spaces. The crossover reproduces the better candidates and the mutation might bring directly to the optimal solution. As the crossover mechanism carries out the best information from the parents which naturally adapts and delivers into the offspring, many researchers have been focused on various ways of crossover process rather than the selection and mutation.

Traditionally, the genetic algorithm chooses two chromosomes from the mating pool, as the parents to reproduce the new child chromosomes by the crossover operator. But, unlike in nature, there is nothing restricting us from using more than two parents to generate offspring. Since the results of genetic algorithms from the previous researches varied due to the correlation between the optimization problems and the crossover operators, the lack of complexity over the real-world problems still remains to challenge a new genetic algorithm among the previous traditional genetic algorithms. In this research, the genetic algorithm using more than two parent chromosomes, called multi-parent genetic algorithm (MPGA), is proposed to solve the numerical benchmark functions and the problem of shuttle bus routing system (SBRS).

A shuttle bus routing system (SBRS) provides the free shuttle buses service within the campus connected by a few routes. The bus stops are located along the different routes in order to reach every corner within the campus. In the proposed SBRS, the problem is to minimize the distance of a certain number of routes and to cover every bus stops inside the campus. Moreover, it can be considered as the nature of formulating the school bus routing system is similar with the nature of vehicle routing problems (VRP).

This research contains twofold. First, the proposed multi-parent genetic algorithm is compared with the traditional genetic algorithm by measuring their performance on high dimensional multimodal benchmark functions. Second, the problem of shuttle bus routing system inside the campus is successfully solved by the proposed multi-parent genetic algorithm and the traditional genetic algorithm. The experiment results are compared and analyzed in order to measure the performance in both genetic algorithms.

The rest of the book is organized as follows. All the related researches from the beginning of genetic algorithm to multi-parent genetic algorithm are described in Chapter 2 as the literature review. The component of the proposed multi-parent genetic algorithm (MPGA) is detailed in Chapter 3. The experiment results on both genetic algorithms are explained in Chapter 4. Finally, the conclusions are drawn in Chapter 5.

1.4 Objectives of study

The study aims to utilize the multi-parent genetic algorithm (MPGA) to the real world problem. The objectives of the study are

- To analyze the effect of the number of parents in genetic algorithm
- To compare the traditional genetic and multi-parent genetic algorithm
- To solve the problem of shuttle bus routing system by using the multi-parent genetic algorithm.

Chapter 2
Literature Review

2.1 Extended Genetic Algorithms

Since the performance of genetic algorithm is varied according to the crossover operator and optimization problems, many extended crossover methods are developed for solving the numerical optimization problems. Multi-parent genetic algorithm is one of the extended genetic algorithms from the family of evolutionary algorithms. Using more than two chromosomes as the parent for reproduction process of the genetic algorithm is not new in the history of genetic algorithms.

The first approach for a genetic algorithm with three parents uniform crossover was developed and analyzed to solve De Jong test functions [12]. The uniform crossover reproduces *n* new children from *n* parents. The algorithm generates three child chromosomes from the following three parents uniform crossover mask and its inverses as shown in Figure 2.1.

```
let k = length of the bit-string
for j = 1 to k do
        mask[j] = random(0,1,2)
        inverse_mask_1[j] = (mask[j]+1) MOD 3
        inverse_mask_2[j] = (mask[j]+2) MOD 3
endfor
```

Figure 2.1 Three parents uniform crossover operator and inverse mask construction

The function *random (0,1,2)* returns either 0, 1 or 2, each with probability one-third, then all the bit-level genetic information is maintained in the child chromosomes. The experiment proved that three parents genetic algorithm outperforms two parents traditional genetic algorithm when solving De Jong test functions. De Jong invented a test environment of five minimization functions [16]. The test functions are listed in

Figure 2.2. De Jong test functions still remains as a milestone in the development of genetic algorithm because of the following natures:

- Continuous or discontinuous
- Convex or nonconvex
- Unimodal or multimodal
- Quadratic or nonquadratic
- Low-dimensionality or high-dimensionality
- Deterministic or stochastic

Later, another non-traditional genetic algorithm with gene scanning crossover operators is analyzed for solving Traveling Salesman Problem (TSP), graph coloring and De Jong test functions [8]. The gene scanning crossover operators calculate the fitness values of the chromosome based on occurrence based scanning which selects the most occurrence genes in the parent. The fitness based scanning which chooses the fitness values. The uniform scanning in which genes are randomly chosen from the parents. The experiment on all the crossover operators concluded that using more parents in some of the De Jong test functions outperform the traditional genetic algorithm, but for the other problems, the traditional genetic algorithm outperforms the genetic algorithm with more than two parents.

f_1	$f_1(x_i) = \sum_{i=1}^{3} x_i^2,$	$-5.12 \leq x_i \leq 5.12$
f_2	$f_2(x_i) = 100(x_1^2 - x_2)^2 + (1 + x_1)^2,$	$-2.048 \leq x_i \leq 2.048$
f_3	$f_3(x_i) = \sum_{i=1}^{5} integer(x_i),$	$-5.12 \leq x_i \leq 5.12$
f_4	$f_4(x_i) = \sum_{i=1}^{30} ix_i^4 + Gauss(0,1),$	$-1.28 \leq x_i \leq 1.28$
f_5	$f_5(x_i) = 0.002 + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6},$	$-65.536 \leq x_i \leq 65.536$

Figure 2.2 De Jong test functions

In [9], diagonal crossover for different number of parents is investigated and compared to the scanning crossover. The fitness functions contain De Jong test functions. The scanning crossover is generalized by the uniform crossover which selects the most occurrence bit in the parents into the child chromosome as shown in Figure 2.3 and the diagonal crossover is generalized by the classic 1-point crossover which reproduce *n* offspring by *n* parents as shown in Figure 2.4. Each of the test functions, the optimal number of parents are always higher than two in both crossovers. The experiment shown that the diagonal crossover yields better results in all test functions when the scanning crossover failed in one test function.

Parent 1:	1 0 0 1 1 0
Parent 2:	0 1 0 1 0 1
Parent 3:	0 0 1 0 1 1
Offspring:	0 0 0 1 1 1

Figure 2.3 Occurrence based scanning crossover in three parents

Parent 1:	**1 0** 0 1 1 0
Parent 2:	0 1 **0 1** 0 1
Parent 3:	0 0 1 0 **1 1**
	↓
Offspring 1:	**1 0 0 1 1 1**
Offspring 2:	0 1 1 0 1 0
Offspring 3:	0 0 1 0 0 1

Figure 2.4 Diagonal crossover with three parents

The genetic algorithm with a new multi-parent crossover applied to solve different numbers of the constrained optimization problems [11]. The problems are introduced in the CEC2010 constrained optimization competition session [7]. A new randomized crossover operator is used to replace mutation, which is intended to escape the local optima and premature convergence. The crossover operator used three individuals to reproduce three offspring according to the crossover rate. The experiment shown better performance as it compares to the traditional genetic algorithm. The efficiency of genetic algorithm was improved by the randomized crossover operator.

In the operations research area, the quadratic assignment problem includes one of the most difficult problems. In [1], a genetic algorithm with new sequential crossover successfully solved the quadratic assignment problems. The research proved that the new sequential crossover operator improves the genetic algorithm rather than an existing multi-parent traditional sequential crossover for the problem.

The earlier research on multi-parent recombination genetic algorithm convinced that the effect of using more than two parents may differ during the combination of the gene transformation in the crossover operator and the nature of the optimization functions. The diagonal crossover performs better than the existing crossovers when solving the numerical test functions. Consequently, other multi-parent genetic algorithms are still remained to solve not only numerical optimization problems, but also for some applications.

2.2 Related Works

In the evolutionary algorithms, the traditional genetic algorithm is widely used for solving many numerical optimization problems and applications. There are different kind of crossovers which are suitable for solving different kind of problems. Multi-parent recombination in the crossover mechanism acquired better performance than the genetic algorithms when solving the optimization problems as in De Jong test suits. In order to adapt the real world complex problems, multi-parent genetic algorithm is a challenge to improve the traditional genetic algorithms.

Eiben investigated the performance of diagonal crossover for higher number of parents in genetic algorithms [10]. The research based on three hypotheses to explain why genetic algorithm increases when more parents are used, which are:

H1. Using more crossover points leads to better performance

H2. Bigger generational gap leads to better performance

H3. Using more parents leads to better performance

The test functions include the behavior of unimodal, multimodal and quasi-random landscapes. Two types of diagonal crossovers are used to examine the hypotheses. First, the diagonal crossover reproduces three offspring from three parents. Second, the diagonal crossover reproduces one offspring from three parents. The illustration for both diagonal crossovers are shown in Figure 2.5.

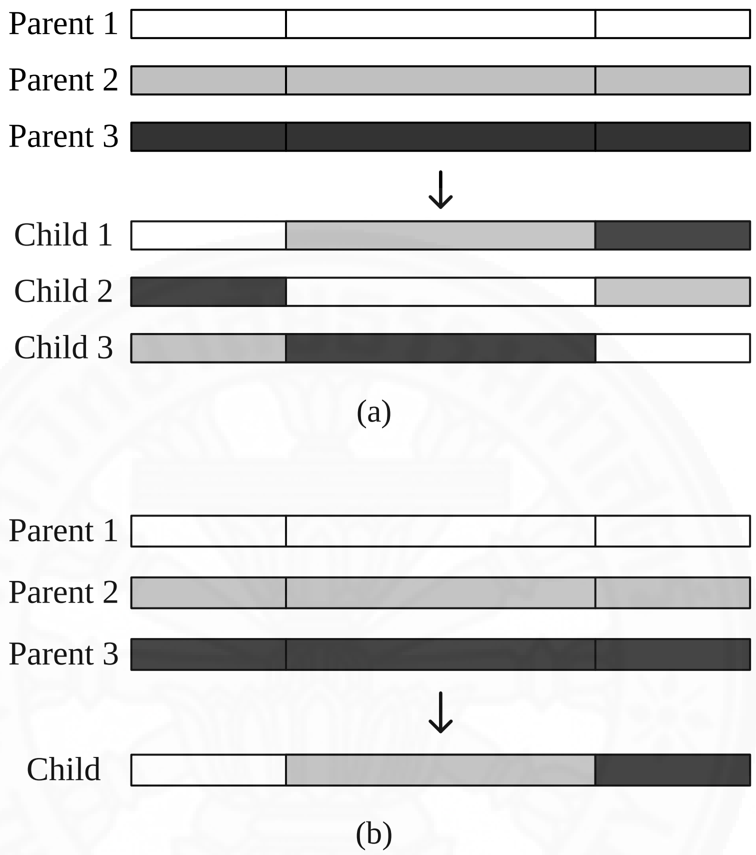

Figure 2.5 Diagonal Crossover with (a) three parents and three children and (b) three parents and one child

The experiment obverse that using the diagonal crossover with more crossover points becomes better in all test functions. Since both types of diagonal crossover applied to all test functions, one child from three parents did not solve in every function. From the results in some test functions, the performance of algorithm did not increase when using more parents recombination.

The original vehicle routing problem (VRP) is an optimization problem, generalized by traveling salesman problem (TSP) [6]. The goal of VRP is to find the minimum mileage of the gasoline delivery trucks between terminals and service stations. Vehicle routing problem with time windows (VRPTW) is an extension of vehicle routing problem with the extra time constraint [2]. The traditional genetic algorithm was applied for solving VRPTW in [17]. A real world application of school bus routing problem was experimented by the traditional genetic algorithm in [4] to minimize the capacity of buses, the cost of running, and the distance traveled to each route.

Yassen applied a genetic algorithm with multi-parent insertion crossover for vehicle routing problem with time windows [20]. The multi-parent insertion crossover is enhanced by two parents insertion crossover. The proposed problem finds the minimal cost of routes without violating the imposed constrains. The experiment compared with two parents crossover operator and obtained better solution.

In this study, new diagonal crossovers are generalized from the original diagonal crossover in both multi-parent genetic algorithm and traditional genetic algorithm. The crossover points in the chromosomes are generalized in order to study the effect of the higher parents in the proposed multi-parent genetic algorithm. The test suits are generalized to the high dimension multimodal bench mark test functions. Therefore, the multi-parent genetic algorithm is applied to solving the proposed school bus routing system in a college campus. The proposed problem minimizes the distance of each route in the campus without losing the number of bus stops within the college campus.

Chapter 3
Multi-parent Genetic Algorithm

3.1 Introduction to Multi-Parent Genetic Algorithm

Genetic algorithm (GA) is a randomized iterative search algorithm. The algorithm starts its routine by a random number of population and produces subsequent generations with three main genetic operators. Multi-parent genetic algorithm (MPGA) is a genetic algorithm, unlike the nature of genetic algorithm, in which the number of parent chromosomes in the crossover operator uses more than two chromosomes to inherit genetic information from the parent chromosomes to offspring chromosomes, from generation to generations.

The procedures of multi-parent genetic algorithm are as simple as the traditional genetic algorithm, beside the crossover operation procedure. In the terminology of multi-parent genetic algorithm, the term *parents* is used for the set of chromosomes selected as the gene donors in the crossover operator. The term *chromosome* is the representation of binary string, which is converted by the real values from the problem. The *fitness functions* are the problems to be optimized by using the multi-parent genetic algorithm. The *population* includes temporary random solutions to optimize within the search space. The *mating pool* is the place where the selected chromosomes are stored and replaces throughout the whole algorithm. Once all the mechanisms in the algorithm accomplished through one time in the genetic algorithm, a one *generation* is completed.

The multi-parent genetic algorithm starts with a random number of population. Follow by the initialization, all the variables from the fitness function are encoded and the three main mechanisms are implemented as follows:

- Selection operator
- Crossover operator
- Mutation operator

Afterwards, all the chromosomes are decoded back into the variables in order to evaluate their fitness values. The proposed multi-parent genetic algorithm terminates when the variation values of the optimal solutions are not changed for several generations.

3.2 Test Functions

The first fold of multi-parent genetic algorithm is regarded to investigate the effect of the different number of parents in the multi-parent genetic algorithm. According to experiment the algorithm on the large problem sets, all ten test functions are generalized from the benchmarks (test functions) in [18].

The benchmarks are commonly known in the literature. All functions are under consideration of multi-models functions and applied as the quality test for resistant optimization techniques like genetic algorithm, simulated annealing, traveling salesmen problem and so on. All the test functions are continuous minimization problems and generalized to the *n* dimension multi-modal test functions with huge number of local extremes. The overview of all ten test functions are illustrated in two dimensions. The explanations of each function are listed below.

3.2.1 Rastringin's Function

Ratringin's function is a non-linear multimodal function which is based on the De Jong functions with extra cosine modulation. Rastringin's function has many local minima. An overview of the Rastringin's function in two-dimension is illustrated in Figure 3.1. The definition of Rastringin's function is as follows:

$$f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\ \cos(2\pi x_i)), \tag{3.1}$$

where $-5.12 \leq x_i \leq 5.12$. Its global minimum is $f(x) = 0$ for $x_i = 0, (i = 1, \dots, n)$.

3.2.2 Schwefel's Function

Many local minimum in the Schwefel's function are closed to each other. It assembles the problem hard to find the global minimum among the local minimum points. The algorithm can converge earlier in the local minimum area. The Schwefel's function is illustrated in two-dimension as shown in Figure 3.2. The definition of Schwefel's function is as follows:

$$f(x) = \sum_{i=1}^{n} (-x_i \sin(\sqrt{|x_i|})), \tag{3.2}$$

where $-500 \le x_i \le 500$. The global minimum is $f(x) = -418.982n$ for $x_i = 420.9687, (i = 1, \dots, n)$.

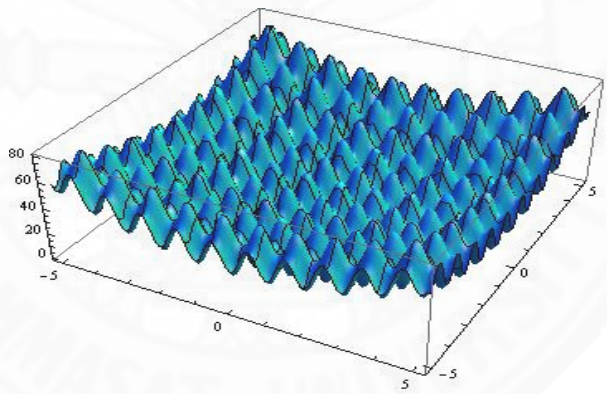

Figure 3.1 An overview of Rastrigin's function in two-dimension

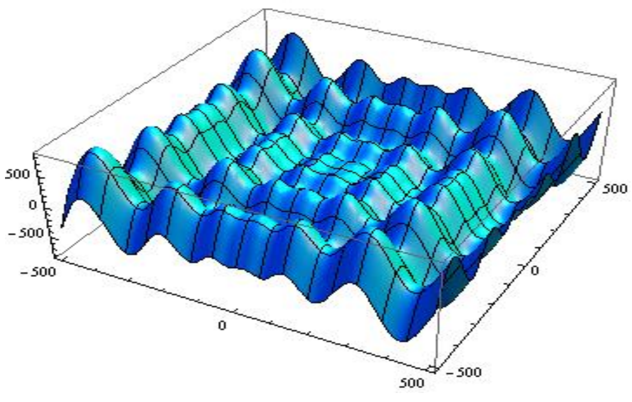

Figure 3.2 An overview of Schwefel's function in two-dimension

3.2.3 Ackley's Function

The Ackley's function contains many local optimum points. In its two-dimension graph in Figure 3.3, there is a large hole in the middle of a broad local minimum region. The mathematical definition of Ackley's function is as follows:

$$f(x) = -a.\exp\left(-b.\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1), \quad (3.3)$$

where $a = 20$, $b = 0.2$, $c = 2\pi$ and $-32.768 \leq x_i \leq 32.768$. Its global minimum is $f(x) = 0$ for $x_i = 0, (i = 1, \ldots, n)$.

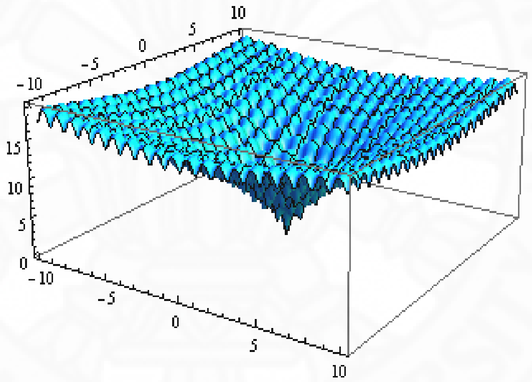

Figure 3.3 An overview of Ackley's function in two dimension

3.2.4 Griewangk's Function

The Griewangk's function includes many distributed local minimum points on its surface. The two dimension view of the Griewangk's function is illustrated in Figure 3.4. The definition of Griewangk's function is as follows:

$$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad (3.4)$$

where $-600 \leq x_i \leq 600$. The global minimum is $f(x) = 0$ for $x_i = 0, (i = 1, \ldots, n)$.

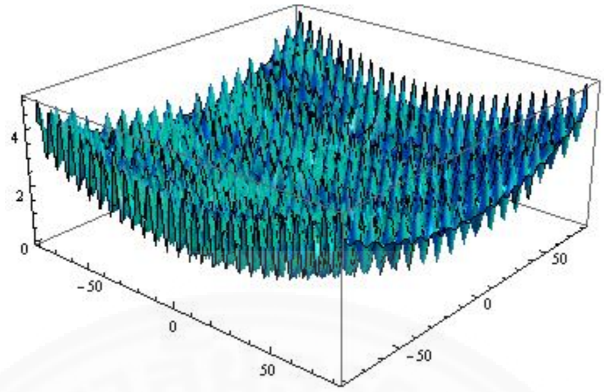

Figure 3.4 An overview of Griewangk's function in two dimension

3.2.5 Langermann's Function

The Langermann's function is a multimodal test function. An overview of the Langermann's function in two-dimension is illustrated in Figure 3.5. The definition of Langermann's function is as follows:

$$f(x) = \sum_{i=1}^{m} c_i \,.\exp\left(-\frac{1}{\pi}\sum_{\mathrm{j}=1}^{\mathrm{n}}\left(x_j - a_{ij}\right)^2\right)\cos\left(\pi\sum_{j=1}^{n}\left(x_j - a_{ij}\right)^2\right), \quad (3.5)$$

where $m = 5$ and $(c_i, i = 1, \ldots, m), (a_{ij}, j = 1, \ldots, n, i = 1, \ldots, m)$ are randomly chosen constants.

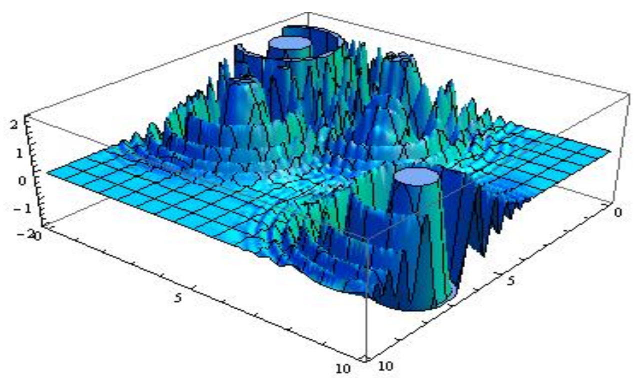

Figure 3.5 An overview of Langermann's function in two dimension

3.2.6 Michalewicz's Function

The Michalewicz's function is a multimodal function which possess a number of steepest edge according to the parameter m. An overview of the Michalewicz's function in two-dimension is illustrated in Figure 3.6. The definition of Michalewicz's function is as follows:

$$f(x) = -\sum_{i=1}^{n} \sin(x_i) \left(\sin\left(\frac{i x_i^2}{\pi} \right) \right)^{2m}, \tag{3.6}$$

where $m = 10, 0 \leq x_i \leq \pi$ and $i = 1, \ldots, n$. The global minimum value has been approximated by $f(x) = -4.687$ for $n = 5$.

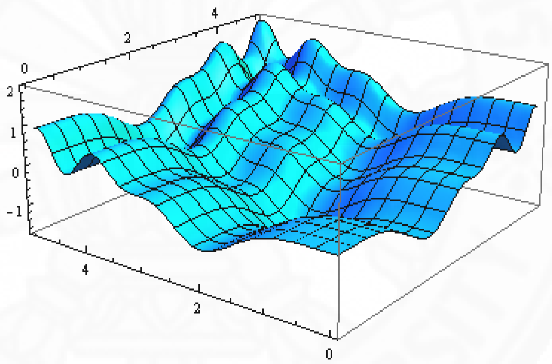

Figure 3.6 An overview of Michalewicz's function in two dimension

3.2.7 Easom's Function

The Easom's function has a long hole in the middle of many local minima. An overview of the Easom's function in two-dimension is illustrated in Figure 3.7. The definition of Easom's function is as follows:

$$f(x) = -\prod_{i=1}^{n} \cos(x_i) .\ \exp \sum_{i=1}^{n} \left(-(x_i - \pi^2) \right), \tag{3.7}$$

where $-100 \leq x_i \leq 100$. The global minimum value is $f(x) = -1$ for $x_i = \pi$.

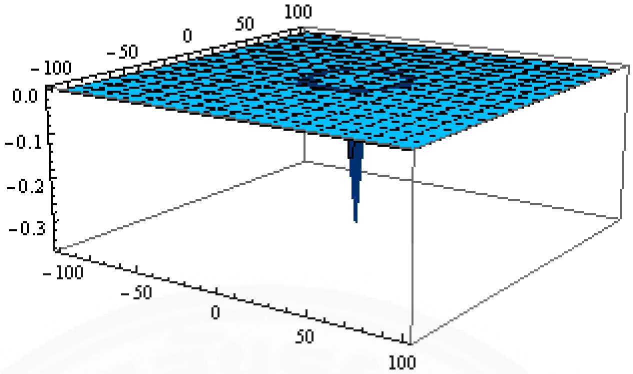

Figure 3.7 An overview of Easom's function in two dimension

3.2.8 Drop Wave Function

The Drop Wave function is a multi-complex function. An overview of the Drop Wave function in two-dimension is illustrated in Figure 3.8. The definition of Drop Wave function is as follows:

$$f(x) = -\frac{1 + \cos\left(12\sqrt{\sum_{i=1}^{n} x_i^2}\right)}{\frac{1}{2}\cdot\sum_{i=1}^{n} x_i^2 + 2}, \tag{3.8}$$

where $-5.12 \leq x_i \leq 5.12$. The global minimum value is $f(x) = -1$ for $x_i = 0$.

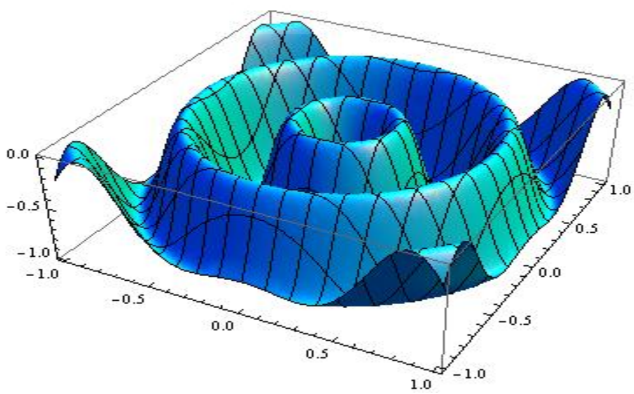

Figure 3.8 An overview of Drop Wave function in two dimension

3.2.9 Shubert's Function

The Shubert's function is a one of the multimodal test functions. An overview of the Shubert's function in two-dimension is illustrated in Figure 3.9. The definition of Shubert's function is as follows:

$$f(x) = -\sum_{i=1}^{5} i\cos\left((i+1).\sum_{i=1}^{n} x_i + 1\right).\sum_{i=1}^{5} i\cos\left((i+1).\sum_{i=1}^{n} x_i + 1\right), \qquad (3.9)$$

where $-5.12 \leq x_i \leq 5.12$. The global minimum value is $f(x) = -186.7309$.

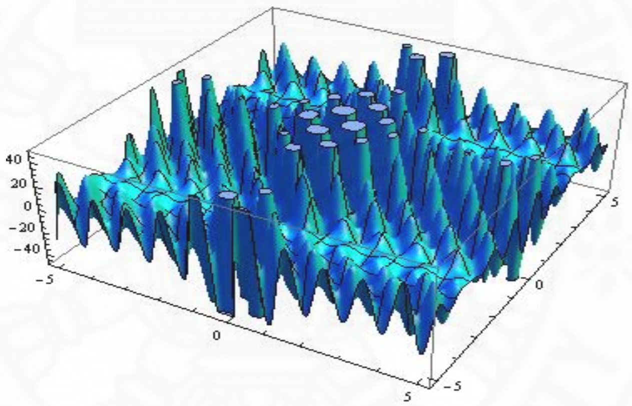

Figure 3.9 An overview of Shubert's function in two dimension

3.2.10 Rosenbrock's Function

The Rosenbrock's function is a famous test problem, also known as the second function of De Jong. An overview of the Rosenbrock's function in two-dimension is illustrated in Figure 3.10. The definition of Rosenbrock's function is as follows:

$$f(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2), \qquad (3.10)$$

where $-2.048 \leq x_i \leq 2.048$. Its global minimum value is $f(x) = 0 \; for \; x_i = 0, (i = 1, \dots, n)$.

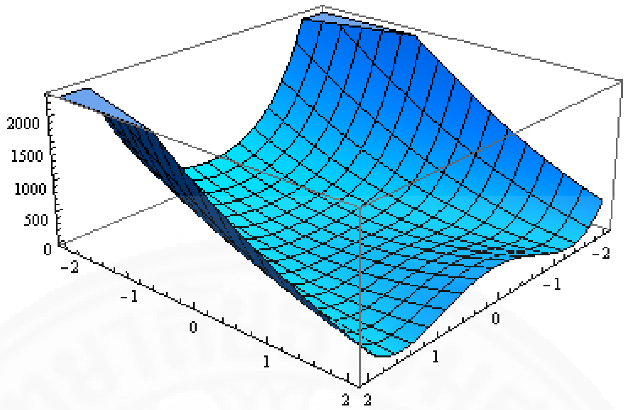

Figure 3.10 An overview of Rosenbrock's function in two dimension

3.3 Problem Formulation of SBRS

The second fold of the study of multi-parent genetic algorithm is to apply and analyze the effects of multi-parent crossover operator by solving the real-world problem. The problem is a shuttle bus routing system (SBRS) based on a college campus. The problem is formulated by considering the location of the bus stops in the Thammasat University (Rangsit Campus), Thailand. Currently, the campus is operating with three different shuttle bus routes. The total number of the bus stops is 47. The distance between the bus stops are different to each route. The location of the bus stops is taken by latitude and longitude. The coordinate points of the bus stops are listed in Table 3.1.

The objective function of SBRS is considered to eliminate the missing bus stops and to optimize the minimum requirement of the route distances. The routes $R_i = (r_{i1}, r_{i2}, \dots, r_{i,n_i})$ passes to the bus stops by the order in $r_{i1}, r_{i2}, \dots, r_{i,n_i}$. Haversine formula is used to calculate the distance $D(R_i)$ for the route i. The problem formulation of SBRS is as follows:

$$\min \sum_{i=1}^{n} D(R_i) + kM(R_i), \qquad (3.11)$$

where k is used as a penalty value for the problem. $M(R_i)$ is the minimum number of missing stops in the route R_i and n is the total number of route i.

Table 3.1: Coordinate points of the bus stops in Thammasat University (Rangsit Campus)

Bus Stops	Latitude	Longitude
1	14.07535	100.60165
2	14.07465	100.60165
3	14.07395	100.60165
4	14.07095	100.60165
5	14.06755	100.60325
6	14.06735	100.60895
7	14.06855	100.60745
8	14.06855	100.60635
9	14.06745	100.60585
10	14.06745	100.60415
11	14.06745	100.60325
12	14.06605	100.60055
13	14.06945	100.60155
14	14.06975	100.60155
15	14.07375	100.60155
16	14.06955	100.60175
17	14.07285	100.60155
18	14.07555	100.60155
19	14.07595	100.60065
20	14.07615	100.59875
21	14.07615	100.59735
22	14.07635	100.59575
23	14.07795	100.59485
24	14.07655	100.59565
25	14.07635	100.59735
26	14.07625	100.59985
27	14.07615	100.60025
28	14.07205	100.60195
29	14.07215	100.60355
30	14.07215	100.60515
31	14.07215	100.60815
32	14.07415	100.61595
33	14.07205	100.61325
34	14.07205	100.61185
35	14.07205	100.60825
36	14.07205	100.60525
37	14.07205	100.60335
38	14.07205	100.60205
39	14.07045	100.60125
40	14.07125	100.59795
41	14.075874	100.59665
42	14.074961	100.59665
43	14.074696	100.59675
44	14.074362	100.596715
45	14.073288	100.596743
46	14.072395	100.59699
47	14.071783	100.597254

The distance for two location points in two-dimension is easy to calculate. But the calculation of the distance for two locations on earth is required to recognize its sphere shape. Therefore, according to calculate the distance for two locations on arc which is expressed by latitude and longitude, Haversine formula is used to measure the distance. The formula was introduced by James Inman in 1835 [15]. The Harvisine formula is described as follows:

$$\Delta Lat = \alpha_2 - \alpha_1 \tag{3.12}$$

$$\Delta Long = \beta_2 - \beta_1 \tag{3.13}$$

$$a = \sin^2\left(\frac{\Delta Lat}{2}\right) + \cos(\alpha_1)\cos(\alpha_2)\sin^2\left(\frac{\Delta Long}{2}\right) \tag{3.14}$$

$$c = 2.\operatorname{atan2}(\sqrt{a}, \sqrt{1-a}) \tag{3.15}$$

$$d = Rc \tag{3.16}$$

where R is 6371 to consider the radius of the earth in kilometer. α_1, α_2 are the latitude points and β_1, β_2 are the longitude points of the geological location on earth. Then d defines the distance for two location points on earth.

3.4 Chromosome Representation

In numerical optimization problems, most of the variables are real numbers. In order to use the binary string chromosomes in the multi-parent genetic algorithm, all different type of variables need the binary encoding and decoding process. The representation of a randomized binary chromosome for two-dimensions problem is demonstrated in Figure 3.11.

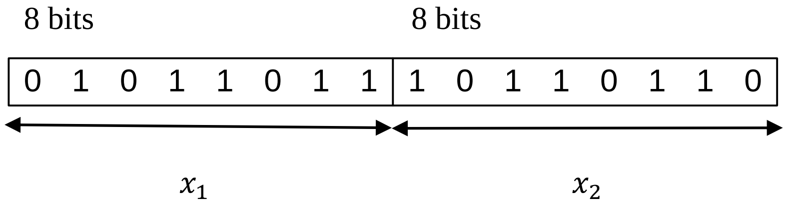

Figure 3.11 The representation of a binary chromosome

All the variables for the test functions and the number of bus stops are real numbers. Each value has its lower and upper bound. The conversion of binary and real number is as follows:

$$X = l + \left(\left(x + \frac{1}{2}\right) \cdot \left(\frac{(u-l)}{2^n}\right)\right), \quad (3.17)$$

where

$X \in \mathbb{R}$

$l =$ lower bound

$u =$ upper bound

$n =$ number of bits in a chromosome

$x =$ the decimal value of a binary chromosome

3.5 Procedure of Multi-Parent Genetic Algorithm

All the parameters for the multi-parent genetic algorithm are predefined before the algorithm starts processing the initialization. The predefined parameters are the parent number, the dimension number for the test function, the population size, the length of each chromosome, the size of mating pool, and the mutation rate.

The multi-parent genetic algorithm begins with a randomized initial population. The population consist of a collection of randomized binary chromosomes. Then the selection operator, the crossover operator, and the mutation operator proceeds to finish one generation. The algorithm processes until the termination criteria meet. The flowchart for the multi-parent genetic algorithm is illustrated in Figure 3.12.

3.5.1 Selection

Selection is the first operator of the multi-parent genetic algorithm. It also manipulates the performance of the algorithm to reproduce better chromosomes in each generation. The selection operator selects the best chromosomes from the initial population into the mating pool. Roulette wheel selection technique is a well-known selection operator in the history of genetic algorithm. The technique is based on the

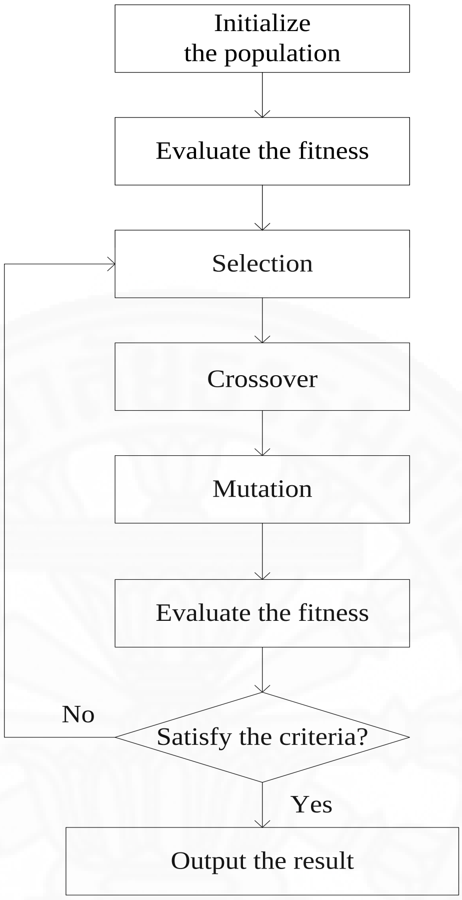

Figure 3.12 Flowchart of multi-parent genetic algorithm

probability of the fitness $f(i)$ in each chromosome i from the population [13]. The better chromosomes have the better probability of fitness. The formulation for the selection technique is as follows:

$$p(i) = \frac{f(i)}{\sum_{j=1}^{n} f(j)} \tag{3.18}$$

where n is the population size. In this research, the Roulette Wheel Selection is generalized according to the minimization nature of the test functions. That is, a chromosome x is chosen with its probability as follows:

$$p(j) = 1 - \frac{f(x)}{\sum_{j=1}^{n} f(j)} \tag{3.19}$$

where $f(x)$ is the value of fitness for the chromosome x.

3.5.2 Crossover

In the genetic algorithm, the process of crossover operator influences the performance of the genetic algorithm more than the other operators. The crossover operator in multi-parent genetic algorithm generates new n chromosomes by transforming inherited genes from the n parent chromosomes. The diagonal crossover operator has been conducted the most efficient and reliable crossover operator for the genetic algorithms in history. The original diagonal crossover chooses n parents with $n-1$ crossover points to reproduce n offspring as described in Figure 3.13.

Figure 3.13 Diagonal crossover with $n-1$ crossover points

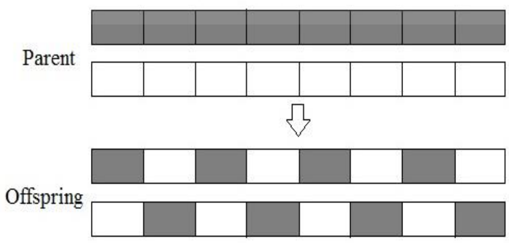

Figure 3.14 Diagonal crossover with multiple crossover points

Since the performance of genetic algorithm varies according to the crossover points in the diagonal crossover operator in the literature. In this study, the effect of multi-parent genetic algorithm analyzed with two crossover methods in diagonal crossover. First, the diagonal crossover with n parents and $n-1$ crossover points to reproduce n offspring. Second, the diagonal crossover with n parents using more than n crossover points to reproduce n offspring. The visualization of multi crossover points in the diagonal is illustrated in Figure 3.14.

3.5.3 Mutation

Mutation is the last mechanism in genetic algorithm. The effect of mutation operator is also important in genetic algorithm. Mutation operator proceeds to change some random genetic information in each chromosome within the mating pool. The process of mutation is straightforward, the operator uses mutation rate p_m to decide the opportunity of changing genetic information. The lower mutation rate makes the lower chance to mutate the genetic information. In the binary coded genetic algorithm, mutation operator is used to flip the bit from 0 to 1 or 1 to 0. Every single bit in each chromosome inside the mating pool is applied by a random number r. A bit from the chromosome is changed when a random number r is less than the mutation rate p_m. The pseudocode for the mutation operator is shown in Figure 3.15.

```
function Mutation:
        mutation_rate  p_m;
        r = random();
        while (current_position  ≤  length of chromosome)  do
                if ( r < p_m)
                        then change current_bit;
                        current_position++;
                endif;
        endwhile;
end.
```

Figure 3.15 Pseudocode for mutation operator

Not every bit in the chromosome is changed due to the mutation rate. Normally the probability of mutation sets quite low but changing some genetic information in the genetic algorithm makes benefit for some chromosomes not to stuck in the local minima.

Chapter 4
Experiments and Results

4.1 Multi-parent genetic algorithm with multimodal test functions

The first fold of experiment is to investigate the effect of the number of parents in multi-parent genetic algorithm (MPGA), which is analyzed by solving ten high multi-model benchmark functions, as expressed in Section 3.2. The parameters for the multi-parent genetic algorithm is listed in Table 4.1.

Table 4.1: Parameter setting of Multi-parent genetic algorithm

Names	**Parameters**
Number of Parents	$2 - 20$
Number of Dimensions ($\boldsymbol{D}$)	$10 - 90$
Initial Population ($\boldsymbol{P}$)	$20(D \leq 50)\ and\ 40(D > 50)$
Chromosome Length	$3D\ bits$
Mutation Rate ($\boldsymbol{p_m}$)	0.01
Mating Pool Size	$P + 0.5P$
Number of Test Functions	10

All the experiments are independently run ten times for each test function. The multi-parent genetic algorithm with various numbers of parents (from 2 to 20) and dimensions (from 10 to 90) are used. We terminate the multi-parent algorithm when its best solution has not changed for more than e^{-8} so far, which unchanged up to 20 consecutive generations. The performance of multi-parent genetic algorithm is measured by three criteria: the best solution (fitness) found, the minimum number of generations to reach the solution and minimum running time.

The average results of ten run for the first test function, Rastringin's Function in Equation 3.1, are illustrated in Figure 4.1. The results are separated by the three criterions. The best result found for the Rastringin's function at 18 parents. The result shows that after using more than two parents in genetic algorithm minimizes the best fitness solution rather than the traditional genetic algorithm.

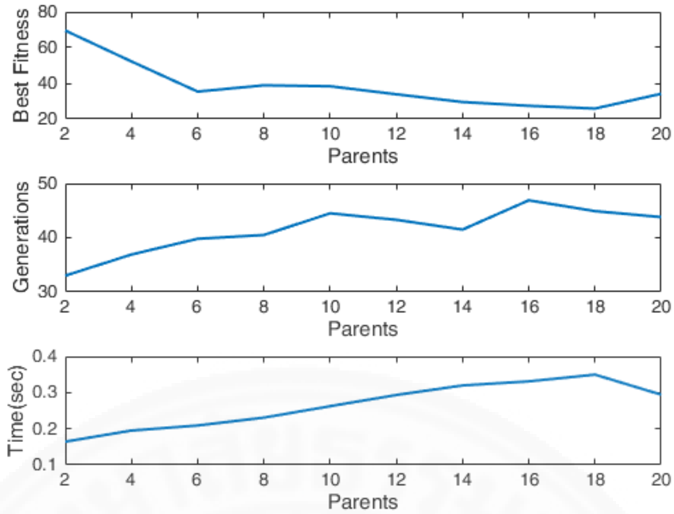

Figure 4.1 Average results of Rastringin’s Function in three criterions

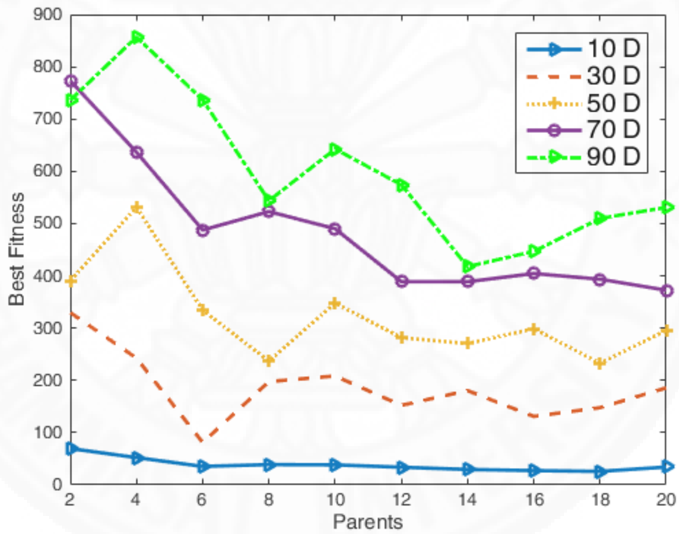

Figure 4.2 Results of best fitness in Rastringin’s function

When using more parents in the Rastringin’s function, the necessary of the number of generations took slightly higher than the traditional genetic algorithm. Minimum consuming time in traditional genetic algorithm is lower than the multi-parent genetic algorithm for not more than 0.2 second. In order to emphasize the result of best fitness values in all number of parents, the details of different dimensions (10 to 90) are shown in Figure 4.2. All fitness values are minimized in all dimensions against some fitness slightly increased after using two parents.

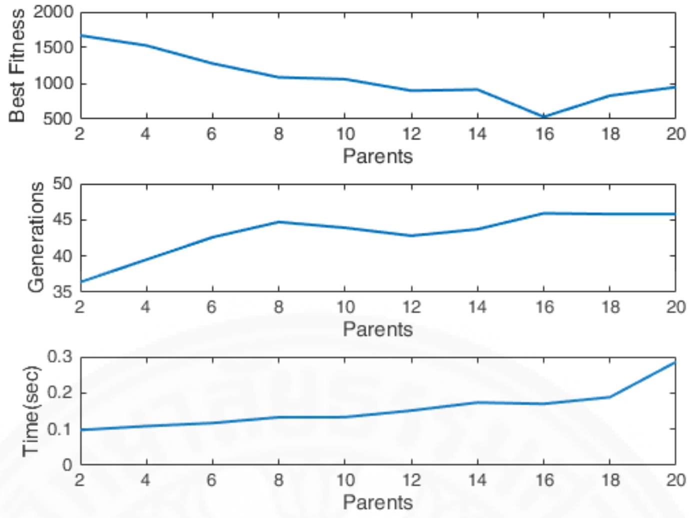

Figure 4.3 Average results of Schwefel's function in three criterions

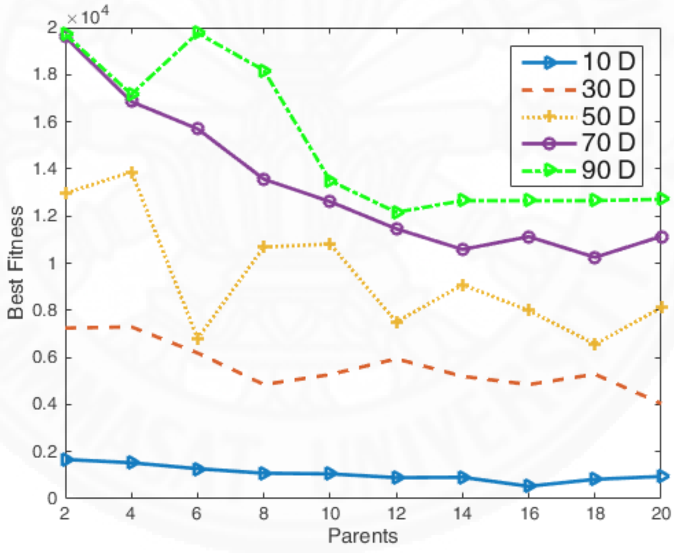

Figure 4.4 Results of best fitness in Schwefel's function

The results for Schwefel's function, Equation 3.2, with three criterions are shown in Figure 4.3. After using more than two parents, the fitness values are minimized from 1500 to 500. The details result for the fitness values in all different dimensions are shown in Figure 4.4. Only the fitness values with dimension 50 and 90 are slightly higher after two parents.

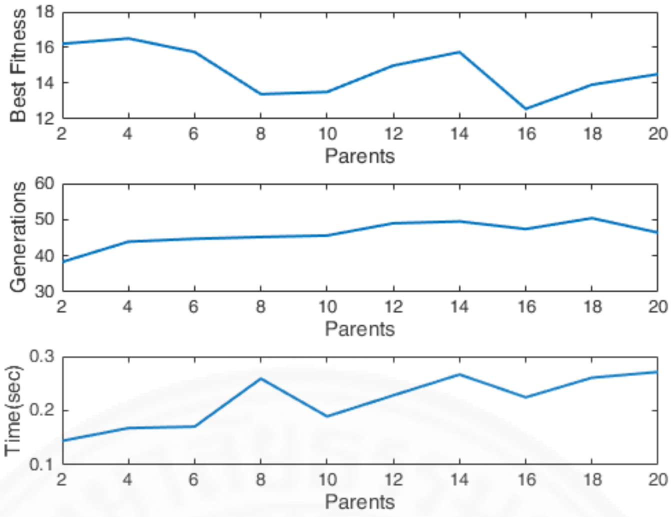

Figure 4.5 Average results of Ackeley's function in three criterions

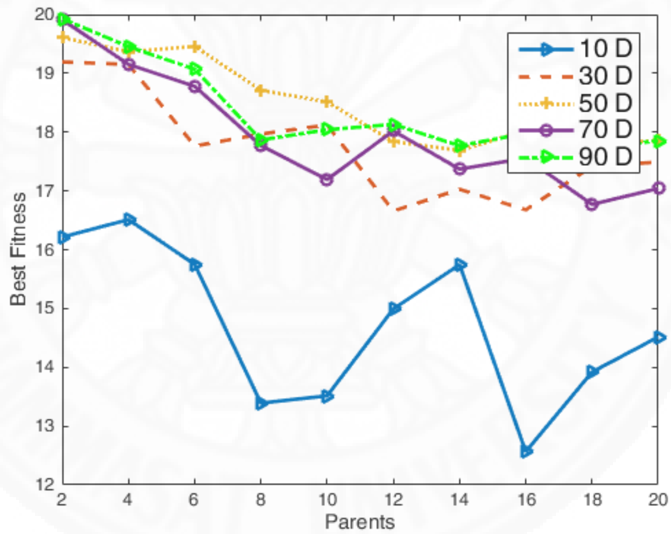

Figure 4.6 Results of best fitness in Ackley's function

The results for Ackley's function, Equation 3.3, with three criterions are shown in Figure 4.5. The result of time consumed at 10 parent slightly down for 0.1 second when the number of generations and the fitness value are higher than at 8 parent. The results of best fitness for each dimension are shown in Figure 4.6. The results shown that the fitness values are dramatically minimized when the function are using multi-parents with higher dimension.

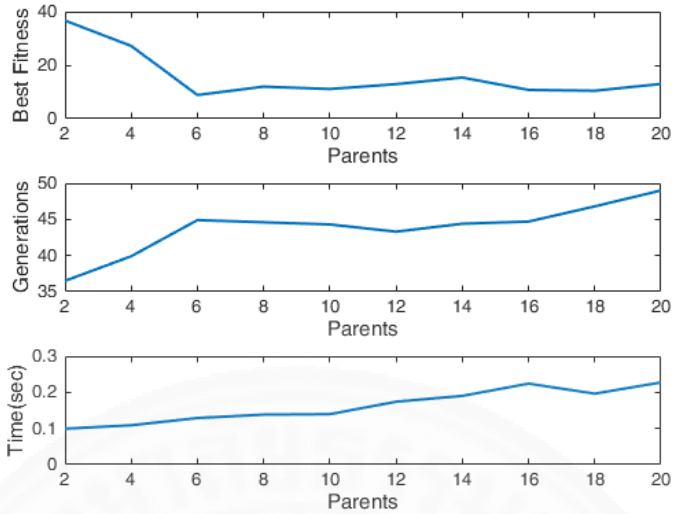

Figure 4.7 Average results of Griewangk's function in three criterions

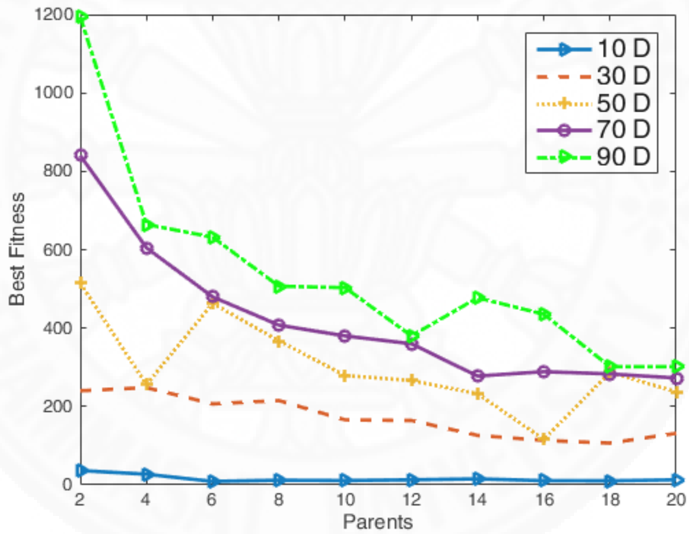

Figure 4.8 Results of best fitness in Griewangk's function

The results for Griewangk's function, Equation 3.4, with three criterions are shown in Figure 4.7. The results for the number of generations and time are slightly increased up to 20 parents when the best fitness values are minimized. In Figure 4.8, the best fitness values at 4 and 16 parents in 50 dimensions are not better than two parents traditional genetic algorithm.

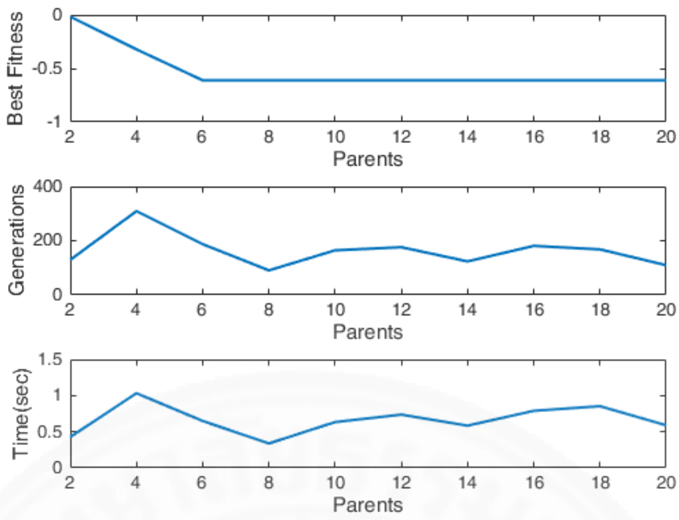

Figure 4.9 Average results of Langermann's function in three criterions

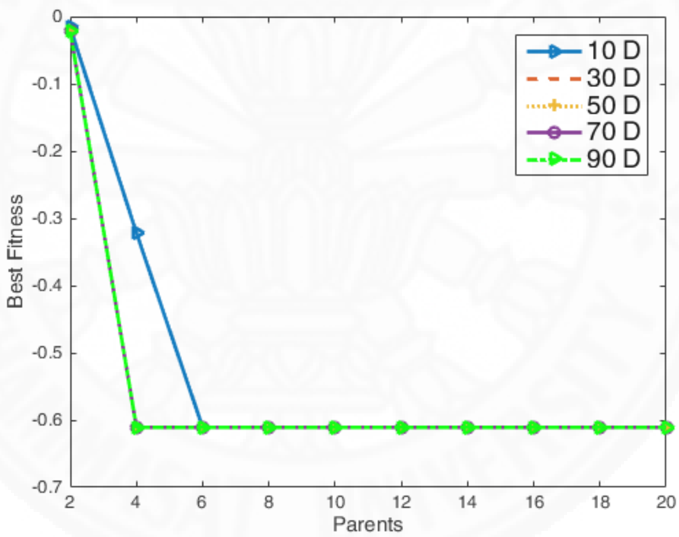

Figure 4.10 Results of best fitness in Langermann's function

The results for Langermann's function, Equation 3.5, with three criterions are shown in Figure 4.9 and the details of best fitness for different dimensions are illustrated in Figure 4.10. The best fitness values are clearly minimized after two parents in all dimensions.

The results for Michalewicz's function, Equation 3.6, with three criterions are shown in Figure 4.11 and the details of best fitness for different dimensions are illustrated in Figure 4.12. The results for all dimensions are minimized equally at 8 and 18 parents.

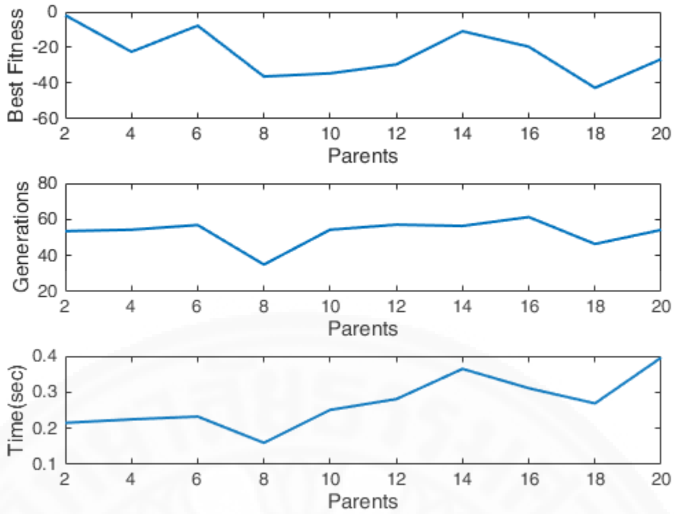

Figure 4.11 Average results of Michalewicz's function in three criterions

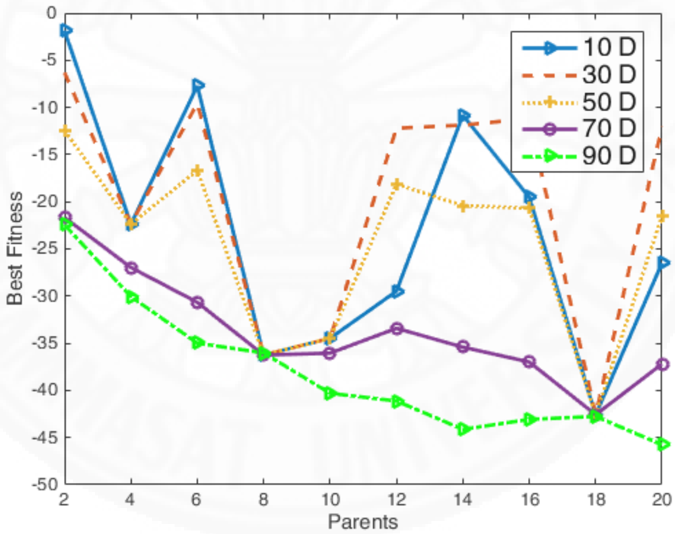

Figure 4.12 Results of best fitness in Michalewicz's function

The results for Easom's function, Equation 3.7, with three criterions are shown in Figure 4.13 and the details of best fitness for different dimensions are illustrated in Figure 4.14. Using 20 parents totally minimized the best fitness in all dimensions.

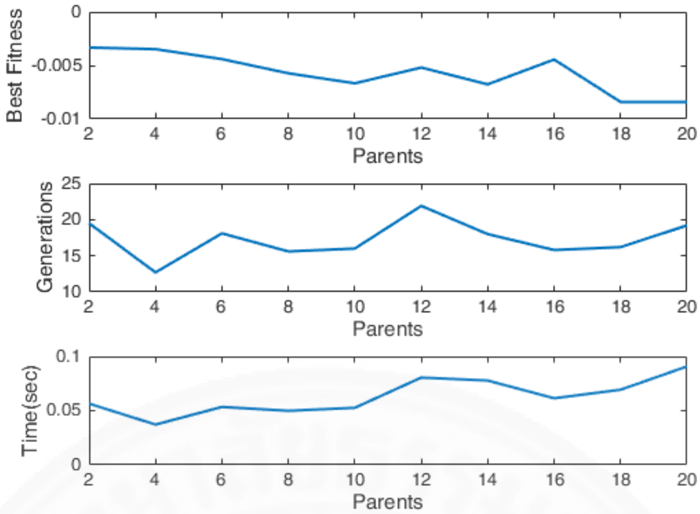

Figure 4.13 Average results of Easom's function in three criterions

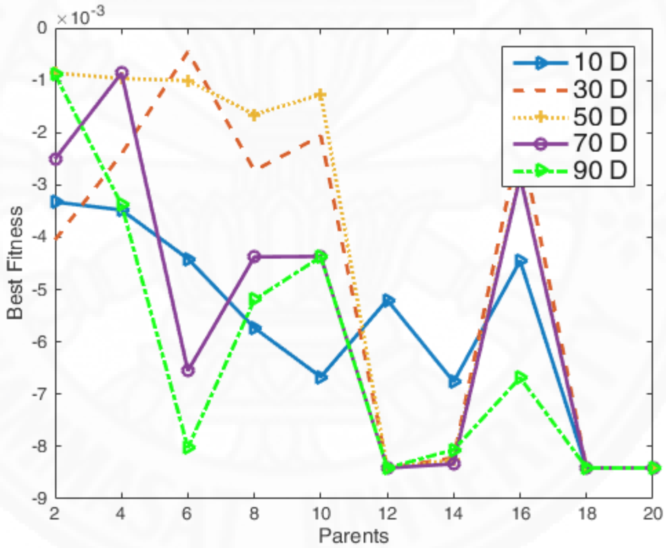

Figure 4.14 Results of best fitness in Easom's function

The results for Drop Wave function, Equation 3.8, with three criterions are shown in Figure 4.15 and the details of best fitness for different dimensions are illustrated in Figure 4.16. The best result of fitness values found at 14 parents in 10 dimensions when the rest of dimensions, 20 to 90, did not minimize the fitness well.

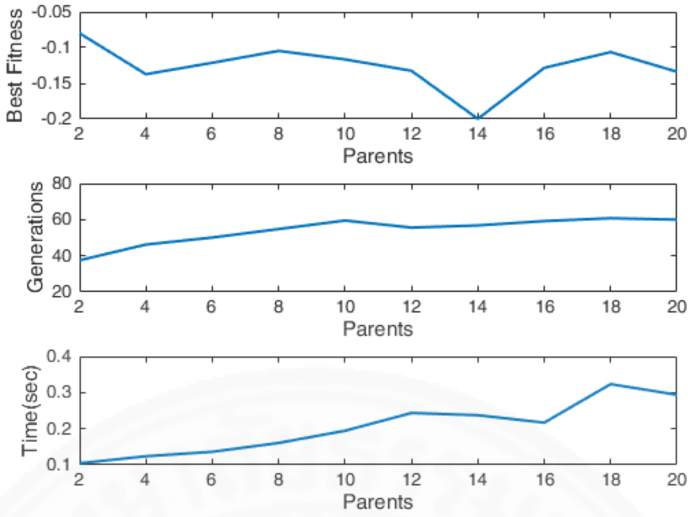

Figure 4.15 Average results of Drop Wave function in three criterions

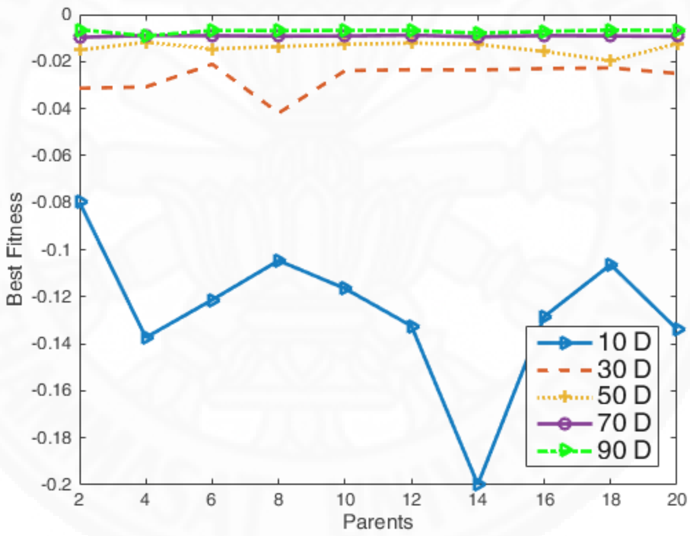

Figure 4.16 Results of best fitness in Drop Wave function

The results for Shubert's function, Equation 3.9, with three criterions are shown in Figure 4.17 and the details of best fitness for different dimensions are illustrated in Figure 4.18. The results are minimized in all dimensions but not steady.

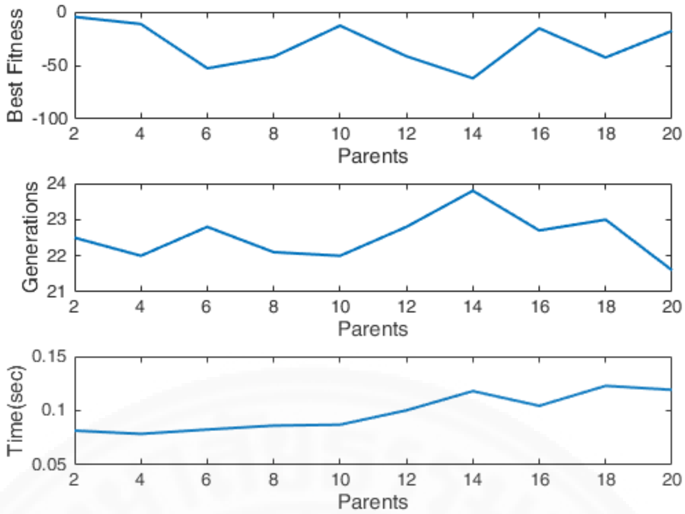

Figure 4.17 Average results of Shubert’s function in three criterions

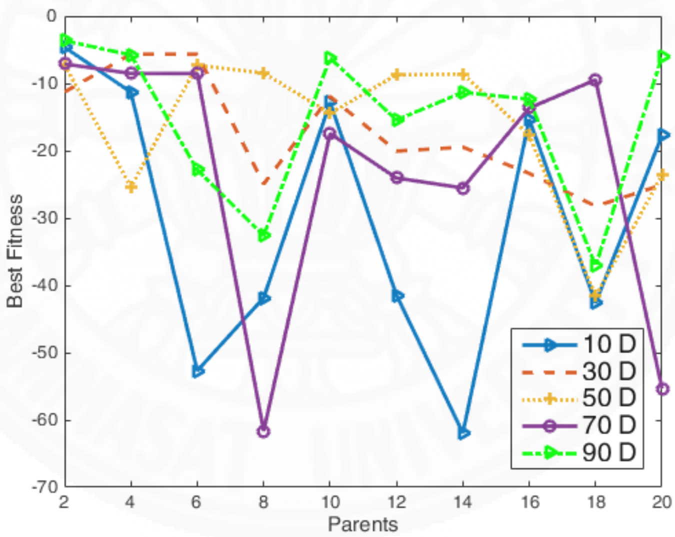

Figure 4.18 Results of best fitness in Shubert’s function

The results for Rosenbert’s function, Equation 3.10, with three criterions are shown in Figure 4.19 and the details of best fitness for different dimensions are illustrated in Figure 4.20. The results in all dimensions are obviously minimized after using more than two parents.

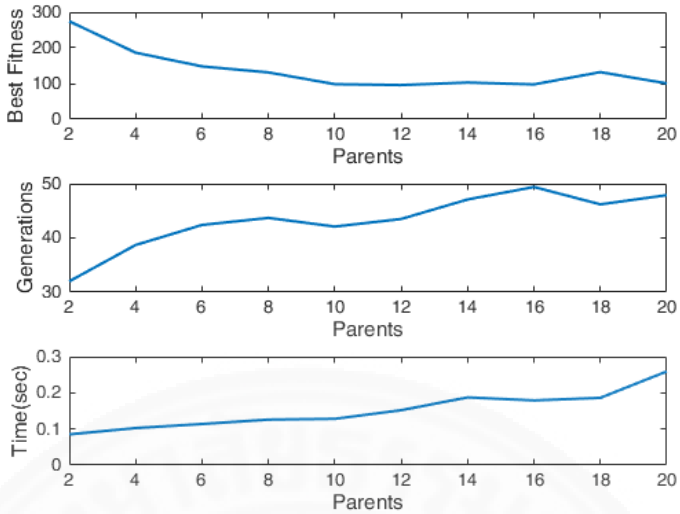

Figure 4.19 Average results of Rosenbrock's function in three criterions

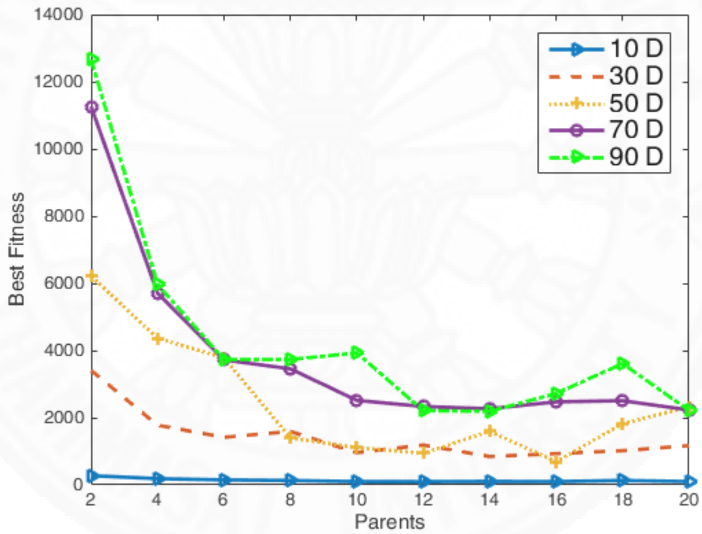

Figure 4.20 Results of best fitness in Rosenbrock's function

To sum up the experiment results in all test functions, the multi-parent genetic algorithm outperformed than the traditional genetic algorithm in most of the generalized test functions. The higher dimensions exploited the biggest achievement for the higher number of parents. Using high dimensions and large number of parents only effected within a tiny second of processing time. By ranking the probability of the best solutions found in each of the dimensions, the multi-parent genetic algorithm with 8 parents minimized more than other number of parents.

4.2 Multi-parent genetic algorithm with SBRS

The second fold of the experiment is to investigate not only the effect of the number of parents in multi-parent genetic algorithm (MPGA) but also the number of different crossover points in the diagonal crossover operator, by solving the real-world problem of the shuttle bus routing system (SBRS) from the Thammasat University (Rangsit Campus) in Thailand.

In this experiments, two type of technique in the diagonal crossover operator are applied to investigate the effect of the number of parents in GAs. The first technique, the crossover chooses n parents chromosomes from the mating pool to reproduce the new n child chromosomes by using the n-1 crossover points in the multi-parent genetic algorithm. The second technique, the crossover reproduces the new n child chromosomes by n parents chromosomes where the different number of multiple crossover points increased up to 19 in both GAs. The main reason of those two approaches is to compare the effect of the same crossover points in both traditional genetic algorithm and the multi-parent genetic algorithm as a result of the previous experiment improved the multi-parent genetic algorithm with more crossover points rather than in the traditional genetic algorithm.

The experiments run ten times to each of the different number of parents. The problem of SBRS is formulated as a minimization function as described in Equation 3.11. The parameters for solving SBRS are listed in Table 4.2.

Table 4.2: Parameter setting of MPGA for SBRS

Names	**Parameters**
Number of Parents	$2-20$
Initial Population ($\boldsymbol{P}$)	$20, 100\ and\ 1000$
Chromosome Length	$6P\ bits$
Mutation Rate ($\boldsymbol{p_m}$)	0.01
Mating Pool Size	$P + 0.5P$
Penalty Value, $\boldsymbol{k}$	100

For the termination of criterion for the problem, the genetic algorithms cease the routine of generations where the fitness value of the fittest solution found in the last 20 consecutive generations are exactly same or the different values are less than e^{-8} from the last generation. The performance of the multi-parent genetic algorithm measures according to the following three criteria:

- The value of the fittest solution
- The number of minimum missing stops.
- The total amount of minimum running time.

Table 4.3: Results of MPGA with different number of crossover points

No of Parents	Ave. Fitness	Ave. Missed Stops	Ave. Generations	Ave. Time (sec)
2	611.72	16.4	35.2	58.791
4	522.88	11.7	31.6	56.389
6	319.87	4.2	42.7	70.183
8	267.25	2.8	48.8	84.776
10	176.58	2.5	57.8	94.145
12	53.729	0	58.5	111.14
14	35.573	0	58.6	135.05
16	34.924	0	86.6	144.85
18	31.824	0	83.4	163.56
20	30.658	0	84.8	188.37

Table 4.3 shows that the experiment results in ten runs. The results are described as the average values from the results of ten runs. The fittest values of MPGA are dramatically optimized as long as the more number of parent chromosomes are extended from two parents. After using ten parents in MPGA, the algorithm covered all the missing bus. The result proof that using more parent in the multi-parent genetic algorithm covered every bus in all three routes. The rest of two criteria from the experiment shows that the diagonal crossover, the algorithms spend more on the running time for counting the generations. Nevertheless, the increasing amount of generations and processing time improved the genetic algorithms to have higher chance of delivering the better chromosomes into the mating pool for the next generation.

Table 4.4: Results of GA with different number of crossover points

No of Crossover Points	Ave. Fitness	Ave. Missed Stops	Ave. Generations	Ave. Time (sec)
1	640.86	16.1	36.6	65.104
3	576.58	10.4	36.5	71.022
5	578.32	14.7	36.3	72.911
7	599.23	13.9	35.8	66.991
9	611.89	9.5	35.8	57.273
11	678.65	12.1	36.7	58.095
13	619.33	14.6	33.3	52.806
15	687.77	13.8	37.6	59.95
17	699.32	16.9	32.5	51.763
19	641.28	17	32.9	52.502

The previous experiment shown the improvement of genetic algorithms by the first technique of the proposed diagonal crossover. In order to proof the better solution of SBRS in the multi-parent genetic algorithm rather than the traditional genetic algorithm, the same crossover points are used in both genetic algorithms. The result for the same number of crossover points are shown in Table 4.4. The fitness values for each number of parents show that the best fitness value of 19 parents did not minimize when comparing with best fitness value of two parents. After increasing the number of parents, the fitness values in 3, 5 and 7 parents are minimized but the others number of parents increased back to the minimum fitness value of two parents. As the effect of the results are unstable in the result of fitness, the number of missing bus stops for the traditional genetic algorithm are not stable until 19 parents. Several bus stops are missing in each number of parents in all genetic algorithms. The results of the last two criteria, the number of minimum generation and processing time, the experiment results show that the values are slightly decreased after using more parents. However, the values between the differences minimum number of generations did not significantly diverse in all genetic algorithms. The results confirmed that the more number of crossover points in the diagonal crossover operator did not improve the results in both GAs.

The detail improvement of fitness results is emphasized for both traditional genetic algorithm and multi-parent genetic algorithm. The comparison is as shown in

Figure 4.21. The results obviously shown that the best fitness values of multi-parent genetic algorithm are minimized in the problem of SBRS. The comparison of the number of missing bus stops is shown in the Figure 4.22. After using 12 parents GA, the missing bus stops is empty up to 19 parents. Apparently, the classic two parents GA did not cover the destinations.

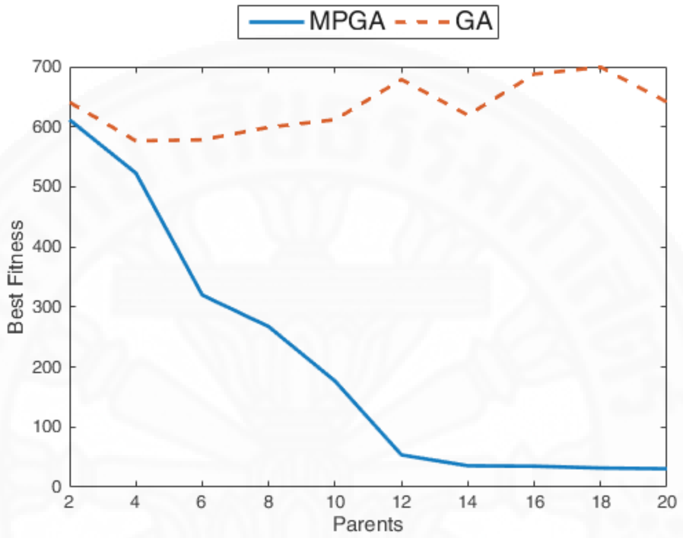

Figure 4.21 Comparison of the best fitness in MPGA and GA

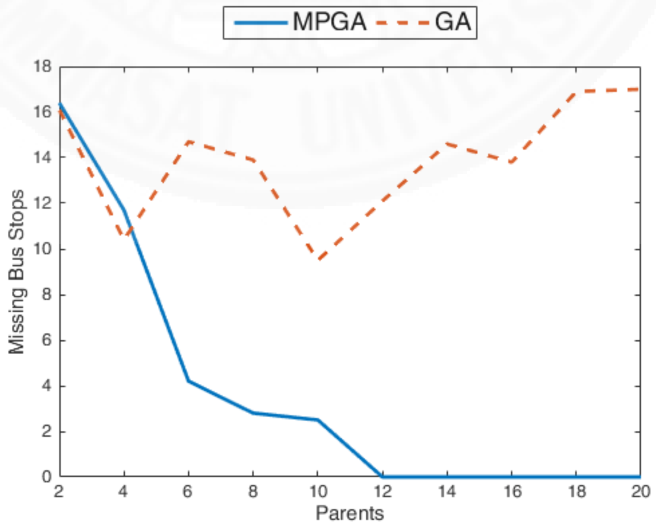

Figure 4.22 The missing stops discovered in MPGA and GA

To encapsulate from the experiment results in GAs, we discovered that the proposed multi-parent genetic algorithm surpassed the traditional genetic algorithm in all the benchmark test functions in the first fold. For the second fold, the multi-paernt genetic algorithm outperformed the classic genetic algorithm by solving the problem of SBRS from the Thammasat University (Rangsit Campus). The bus routes are successfully minimized the distances between all the bus stops without keeping the number of missing bus stops. Whether the number of crossover points in the diagonal crossover are increased or not, the multi-parent genetic algorithm improves the solution better than the classic two parent genetic algorithm.

Chapter 5
Conclusions

Genetic algorithm (GA) is a powerful search technique from the family of Evolutionary Algorithms (EAs). Many researchers have studied different type of genetic algorithms in the literature. Genetic algorithms are different from other optimization in variety of parameters, search by multiple search points, not derivatives and but stochastic. Genetic algorithms optimize the solutions by applying three major mechanisms: selection, crossover, and mutation. Since the crossover operator plays vital role in genetic algorithm, unlike the nature, a genetic algorithm with multi-parent reproduction method is proposed.

First, the experiment of multi-parent genetic algorithm (MPGA) for solving ten multimodal high dimension benchmark functions shows that using higher number of parents yielded better solution without taking much computation time in all of the test cases. The multi-parent genetic algorithm with higher number of dimensions improved better convergence with $n-1$ crossover points in diagonal crossover operator.

Second, for the performance of multi-parent genetic algorithm with the same crossover points in all number of parents, the multi-parent genetic algorithm with a real-world problem of the school bus routing system (SBRS) from Thammasat University (Rangsit Campus) is considered. The experiments confirmed that when comparing the same number of crossover points in the diagonal crossover operator for all number of parents, the proposed multi-parent genetic algorithm outperformed the traditional genetic algorithm.

Consequently, using more than two parents in genetic algorithm does improve the performance of genetic algorithm, but no optimal number of parents are not yet found.

References

1. Ahmed, Z.H. (2015). A Multi-parent Genetic Algorithm for the Quadratic Assignment Problems, OPSEARCH, vol. 52, No. 4, 2015, pp. 714-732.

2. Braysy, O. and Gendreau, M. (2005). Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms, Transportation Science, vol. 39, pp. 104-118.

3. Bently, P.J. (1999). Evolutionary Design by Computers, Morgan Kaufmann, San-Francisco.

4. Ben Sghaier, S., Ben Guedria, N. and Mraihi, R. (2013). Solving School Bus Routing Problem with genetic algorithm, Advanced Logistics and Transport (ICALT), 2013 International Conference, Sousse, pp. 7-12.

5. Back, T, Schwefel, H.-P. (1993). An Overview of Evolutionary Algorithms for Parameter Optimization, Evolutionary Computation 1 (1), 1-23.

6. Dantzig, G.B. and Ramser, J.H. (1959). The Truck Dispatching Problem, Management Science, Vol. 6, No. 1, pp. 80-91.

7. Das, S. and Suganthan, P.N. (2010). Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems, Technical Report, Jadavpur University, Nanyang Technology University.

8. Eiben, A.E., Raue, P-E., Ruttkay Zs. (1994). Genetic algorithms with multi-parent recombination, Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, LNCS 866, Springer-Verlab, 78-87.

9. Eiben, A.E., Van Kemenade C.H.M. and Kok, J.N. (1995). Orgy in the Computer: Multi-parent Reproduciton in Genetic Algorithms, Proceeding of the 3rd European Conference on Artificial Life, LNAI 929, Springer-Verlag, pp. 934-945.

10. Eiben, A.E., van Kemenade C.H.M. (1997). Diagonal Crossover in genetic algorithms for numerical optimization, Control and Cybernetics, vol 26, No. 3.

11. Elsayed, C.K., Saker, R.A., Essam, D.L. (2011). GA with a New Multi-parent Crossover for Solving Constrained Optimization, Evolutionary Computation (CEC), 2011 IEEE Congress, pp. 857-864.

12. Edmondson, L.V. (1993). Genetic Algorithm with 3-parent crossover. University of Missouri/Rolla.

13. Goldberg, D.E. (1989). Genetic Algorithm in Search, Optimization, and Machine Learning (Addison-Wesley, Reading, Massachusetts, 1989).

14. Holland, J.H. (1975). Adaption in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI.

15. James, I. (1835). Navigation and Nautical Astronomy, for the Use of British Seaman, C & J Rivington.

16. Kenneth A. De Jong. (1795). An Analysis of the Behavior of a Class of Genetic Adaptive Systems, PhD thesis, Ann Arbor, MI, USA.

17. Nazif, H. and Lee, L. (2010). Optimized Crossover Genetic Algorithm for Vehicle Routing Problem with Time Windows, American journal of applied sciences, vol. 7, pp. 95-101.

18. Molga, M., Smutnicki, C. (2005). Test functions for optimization needs, from http://www.rosenbertmarks.org/Classes/ENGR5358/Papers/functions.pdf

19. Malhotra, R., Singh, N. & Singh, Y. (2011). Genetic Algorithms: Concepts, Design for Optimization of Process Controllers, Computer and Information Science, 4, 39-54.

20. Yassen, E.T., Ayob, M., Nazri, M.Z.A., Sabar, N.R., (2012). Multi-parent insertion crossover for vehicle routing problem with time windows, 4th Conference on Data Mining and Optimization (DMO), Langkawi, Malaysia.

21. Ben Sghaier, S., Ben Guedria, N. and Mraihi, R. (2013). Solving School Bus Routing Problem with genetic algorithm, Advanced Logistics and Transport (ICALT), 2013 International Conference, Sousse, pp. 7-12.

Appendix
List of Publications

A.1 International Conference

1. Seng Pan That Pann Phyu, Gun Srijuntongsiri, "Effect of the number of parents on the performance of multi-parent genetic algorithm", *Proceedings of the 11th 2016 International Conference on Knowledge, Information and Creativity Support Systems* (KICSS2016), 2016, pp. 188-193.
2. Seng Pan That Pann Phyu, Gun Srijuntongsiri, "A Binary Coded Multi-Parent Genetic Algorithm for Shuttle Bus Routing System in a College Campus", *Proceedings – 4th IGNITE Conference and the 2016 International Conference on Advanced Informatics: Concepts, Theory and Application (ICAICTA2016)*, 201.