IMPROVEMENTS ON THE BEES ALGORITHM FOR CONTINUOUS OPTIMISATION PROBLEMS

by

SILAH HAYATI KAMSANI

A thesis submitted to the

University of Birmingham

for the degree of

DOCTOR OF PHILOSOPHY

Department of Mechanical Engineering

School of Engineering

College of Engineering and Physical Sciences

University of Birmingham

April 2016

ABSTRACT

This work focuses on the improvements of the Bees Algorithm in order to enhance the algorithm's performance especially in terms of convergence rate. For the first enhancement, a pseudo-gradient Bees Algorithm (PG-BA) compares the fitness as well as the position of previous and current bees so that the best bees in each patch are appropriately guided towards a better search direction after each consecutive cycle. This method eliminates the need to differentiate the objective function which is unlike the typical gradient search method. The improved algorithm is subjected to several numerical benchmark test functions as well as the training of neural network. The results from the experiments are then compared to the standard variant of the Bees Algorithm and other swarm intelligence procedures. The data analysis generally confirmed that the PG-BA is effective at speeding up the convergence time to optimum.

Next, an approach to avoid the formation of overlapping patches is proposed. The Patch Overlap Avoidance Bees Algorithm (POA-BA) is designed to avoid redundancy in search area especially if the site is deemed unprofitable. This method is quite similar to Tabu Search (TS) with the POA-BA forbids the exact exploitation of previously visited solutions along with their corresponding neighbourhood. Patches are not allowed to intersect not just in the next generation but also in the current cycle. This reduces the number of patches materialise in the same peak (maximisation) or valley (minimisation) which ensures a thorough search of the

problem landscape as bees are distributed around the scaled down area. The same benchmark problems as PG-BA were applied against this modified strategy to a reasonable success.

Finally, the Bees Algorithm is revised to have the capability of locating all of the global optimum as well as the substantial local peaks in a single run. These multi-solutions of comparable fitness offers some alternatives for the decision makers to choose from. The patches are formed only if the bees are the fittest from different peaks by using a hill-valley mechanism in this so called Extended Bees Algorithm (EBA). This permits the maintenance of diversified solutions throughout the search process in addition to minimising the chances of getting trap. This version is proven beneficial when tested with numerous multimodal optimisation problems.

ACKNOWLEDGEMENTS

First and foremost I am grateful to Allah S.W.T. for giving me the strength to finish this study.

An utmost thanks to my supervisor, Professor Duc Truong Pham for all his guidance and patience in supervising me until I am able to complete this PhD journey. Your countless advises will stay with me forever.

To Dr. Marco Castellani, Dr. Ang Mei Choo and Dr. Zhang Zhi-cheng for giving me useful suggestions, my sincerest thanks from the bottom of my heart. To my research colleagues, Stephan Xie, Shafie Kamaruddin, Muhammad Syahril Bahari, and other closest friends, thank you for your continuous encouragements.

To the Malaysian Ministry of Higher Education, and the Universiti Teknikal Malaysia Melaka (UTeM), without your sponsorship, none of these are possible. Thank you very much.

My heartfelt gratitude to my parents and in-laws for believing in me and their prayers. There is no word to express how much I appreciate the sacrifices that you have made in helping me to take care of my children when I needed the most.

Last but not least, this goes to my husband Nik Mohd Farid who is also my research partner as well as my two daughters, Nik Farzana and Nik Alya. Thank you for all your understanding whenever I could not fulfil my duty as a wife and a mother first until this study is over.

This thesis was copy edited for conventions of language, spelling, and grammar by FH Proofreading Solution.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABC	Artificial Bee Colony
ABC-M	Multimodal Artificial Bee Colony
ACO	Ant Colony Optimisation
ACOR	Ant Colony Optimisation for Real Continuous Optimisation
ACS	Ant Colony System
AI	Artificial Intelligence
aiNet	Artificial Immune Network
AIS	Artificial Immune System
ANFIS	Adaptive Neuro-Fuzzy Inference System
ANNs	Artificial Neural Networks
AS	Ant System
BBO	Biogeography-based Optimisation
BCO	Bee Colony Optimisation
BFO	Bacterial Foraging Optimisation
BNSO	Bee Nest Site Optimisation
BP	Back-propagation
BS	Bee System
CLONALG	Clonal Selection Algorithm
CMA-ES	Covariance Matrix Adaptation-Evolutionary Strategy
CPU	Central Processing Unit
DE	Differential Evolution
E.coli	Escherichia coli
EAs	Evolutionary Algorithms
EBA	Extended Bees Algorithm
EED	Environmental/Economic Dispatch

EP	Evolutionary Programming
ES	Evolutionary Strategy
FA	Firefly Algorithm
FNN	Feedforward Neural Network
FNN	Feedforward Neural Network
FPAB	Flower Pollination by Artificial Bees
FS	Fuzzy System
FSA	Fish Swarm Algorithm
Func.	Function
GA	Genetic Algorithm
GD	Gradient Descent
GP	Genetic Programming
GRASP	Greedy Randomised Search Procedure
GSO	Glow-worm Swarm Optimisation
HBMO	Honey Bees Mating Optimisation
HIS	Habitat Suitability Index
HS	Harmony Search
IWO	Invasive Weed Optimisation
KF	Kalman Filter
lhc	Local Hill Climbing
LVQ	Learning Vector Quantisation
MBO	Marriage in Honey Bees Optimisation
MLP	Multi-layer Perceptron
MMO	Multimodal Optimisation
MSE	Mean Square Error
NFE	Number of Function Evaluation
NP	Non-deterministic Polynomial-time

NS	Newton Search
OPF	Optimal Power Flow
opt-aiNet	Artificial Immune Network for Optimisation
OR	Operation Research
PBO	Paired Bacteria Optimisation
PG-BA	Pseudo-gradient Bees Algorithm
PID	Proportional-Integral-Derivative
POA-BA	Patch Overlap Avoidance-Bees Algorithm
PSO	Particle Swarm Optimisation
qABC	Quick Artificial Bee Colony
RBF	Radial Basis Function
rpso	Ring Particle Swarm Optimisation
SA	Simulated Annealing
SAT	Propositional Satisfiability
SCM	Supply Chain Management
SD	Steepest Descent
SI	Swarm Intelligence
SIV	Suitability Index Variables
SPSO	Standard Particle Swarm Optimisation
SPSO	Standard Particle Swarm Optimisation
Std. Dev.	Standard Deviation
SVM	Support Vector Machine
TS	Tabu Search
TSP	Travelling Salesman Problem
VRP	Vehicle Routing Problem
XOR	Exclusive-OR

LIST OF SYMBOLS

f	Objective or cost function(s)
x	Parameter to be optimised/design or decision variable(s), can be continuous, discrete or a mixture of both
R^N	Design/search/solution space in real value
N	Number of decision variables
k	Archive/population size of ACOR
d	Number of dimension
ns	Number of scout bees in the Bees Algorithm
nb	Number of best sites in the Bees Algorithm
ne	Number of elite sites in the Bees Algorithm
nre	Number of bees recruited for e sites in the Bees Algorithm
nrb	Number of bees recruited for $(b - e)$ sites in the Bees Algorithm
ngh	Size of patches including site and its neighbourhood in the Bees Algorithm
p	Size of the bees' population in the Bees Algorithm
x_{rand}	Random position of bee in the Bees Algorithm
$rand$	Random vector element between 0 to 1 following the uniform distribution
x_{max}	Upper bound to the solution vector
x_{min}	Lower bound to the solution vector
Index i	Bee number $(i = 1, 2 \ldots n)$
x_{i+1}	Position of bee $i + 1$
x_i	Position bee i
$rand_i$	Random vector element between 0 to 1 following the uniform distribution for bee i
x_{max}^i	Upper bound to the solution vector for bee i

x_{min}^{i}	Lower bound to the solution vector for bee i
$s_i(t)$	Initial neighbourhood size for bee i at iteration t
$ngh(t)$	Neighbourhood size at iteration t
$ngh(t+1)$	Neighbourhood size at iteration $(t+1)$
$stlim$	Stagnation limit for the Bees Algorithm
ins	Initial number of scout bees in the Extended Bees Algorithm
nr	Number of recruited bees for selected sites in the Extended Bees algorithm
x_p, x_q	Two points to compare in the hill-valley mechanism
m	Number of selected sites in the Extended Bees Algorithm
sp	Sample points for the hill-valley mechanism
ts	Number of total scout bees in the Extended Bees Algorithm
x_k	Position of scout bee in the pseudo-gradient the Bees Algorithm
x_l	Position of scout bee in the pseudo-gradient the Bees Algorithm
g_p	Pseudo-gradient
dir	Direction
T	Matrix transpose
α	Confidence level
O	Number of optimum in function Shekel
CS	Colony size of Quick Artificial Bee Colony
r	Neighbourhood radius for Quick Artificial Bee Colony
l	Limit for abandonment in Quick Artificial Bee Colony
S	Swarm size for Standard Particle Swarm Optimisation 2011
c_1	Cognitive coefficient for Standard Particle Swarm Optimisation 2011
c_2	Social coefficient for Standard Particle Swarm Optimisation 2011
w	Inertia weight for Standard Particle Swarm Optimisation 2011

K	Number of informants for Standard Particle Swarm Optimisation 2011
n	Number of input nodes
h	Number of hidden nodes
o	Number of output nodes
W_{ij}	Connection weight from the *i*th node in the input layer to the *j*th node in the hidden layer
θ_j	Bias of the *j*th hidden node,
X_i	The *i*th input
Y_k	Final output of feedforward neural network
W_{kj}	Connection weight from *j*th hidden node to the *k*th output node
θ_k	Bias of the *k*th output node
E	Learning error of feedforward neural network
q	Number of training samples
D_i^k	Desired output of the *i*th input unit when the kth training sample is used
Y_i^k	The actual output of the *i*th input unit when the *k*th training sample is used
CF	Crowding factor

CHAPTER 1

INTRODUCTION

1.1 Background

In today's agile manufacturing environment, industry needs to be able to cope with rapidly changing markets. Optimised manufacturing systems and processes can help satisfy customers' demands of low price, high quality, and customised products. In the manufacturing sector, optimisation problems abound in areas such as job scheduling, process planning, machine cell formation, and assembly line balancing. They can be categorised as discrete or combinatorial optimisation problems that are NP (non-deterministic polynomial-time) complete, where computational times would increase exponentially as the size of the problem increases. Decision variables in this type of instances are from a set of finite or countable infinite elements. Others such as optimal machining parameters, component dimensions design, and controller parameters, are classified in the continuous domain due to their real-number nature.

For both groups, traditional optimisation methods such as Linear or Integer Programming are no longer sufficient in providing swift optimum results due to the complexity of problems that involve many dimensions, high degrees of non-linearity and severe constraints. Thus, in order to compete in a volatile and global world, companies have turned to Artificial Intelligence (AI) techniques in their decision making. One of the subsets of techniques in AI, Swarm Intelligence (SI), has garnered much interest in the past two decades. The success of SI can be attributed to its population-based feature that imitates nature and leads to an emergent behaviour through the collective actions of individual agents in the swarm (Bonabeau et al., 1999).

Examples of SI algorithms are the Ant Colony Optimisation (ACO) algorithm which mimics the food foraging behaviour of ants, Particle Swarm Optimisation (PSO) algorithm which uses the analogy of birds flocking, and Artificial Bee Colony (ABC) algorithm emulating the foraging behaviour of honeybees while searching for nectar. Another algorithm that captures the essence of honeybees searching for food is the Bees Algorithm.

1.2 Motivations

The Bees Algorithm was developed in 2005 by a team of researchers from Cardiff University (Pham et al., 2005). The algorithm combines random global search led by scout bees, and exploitative neighbourhood search by recruited bees. Furthermore, it has seven parameters that are number of random scouts, number of elite bees, number of best bees, number of foragers in elite sites, number of foragers in best sites, size of neighbourhood, and stagnation limit. Users must configure the parameters beforehand. In each iteration of the algorithm, selected random scouts of higher fitness (i.e. elite bees and best bees) recruit forager bees to start searching around the neighbourhood of the higher fitness point. In nature, this neighbourhood is analogous to flower patch or site. Meanwhile, unselected scouts will execute random search again. Bee with the most profitable fitness in each patch become scout in the next generation and performs bees' recruitment. If no improvement of fitness occurs in the next cycle, the neighbourhood size is shrinked. The flower site is abandoned if there is still no yield in the solution quality once the stagnation limit is reached. Scout bee from the abandoned patch is sent for random scouting. These steps are repeated until a stopping criterion is met which can be the maximum number of generations reached or a satisfactory solution has been found.

The Bees Algorithm has been used to solve various optimisation problems such as design of mechanical structures, training of Artificial Neural Networks (ANNs) for pattern recognition, tuning of fuzzy logic controllers, insertion sequence planning in PCB assembly, and machine scheduling (Pham and Castellani, 2015). These attainments can be credited to the algorithm's excellent behaviour in exploring landscapes with multiple hills and valleys (multimodal) as observed by Pham and Castellani (2009), and Tsai (2014b). Thus, it can be beneficial for industry to implement the Bees Algorithm as the search space of many complex real-world problems with continuous variables has multimodal characteristics.

Considering that the algorithm is a relatively recent introduction to the optimisation area, there are many opportunities for further development. Although the "No Free Lunch Theorem" clearly indicates that that no algorithm performs better than any other when their performance is averaged uniformly over all possible problems (Wolpert and Macready, 1997), progress in this field can bring about a method that, although perhaps not the best for all applications, is good enough for a reasonable range of problems.

The main goal of improvement is a better convergence speed. Many stochastic population-based techniques, including the Bees Algorithm, require a long computation time when in the region of the global optimum due to the random search direction. Specifically, in the case of the Bees Algorithm, this randomness manifests itself in the arbitrary positioning of bees within their neighbourhood during each cycle. Thus, one way to enhance the search rate is through hybridisation with gradient-based algorithms (e.g. Gradient Descent, Newton Search) which can discover an excellent direction by quickly following the gradient line. However, a disadvantage of gradient search is that the objective function needs to be differentiable (Kaveh and Talatahari, 2010; Geem et al., 2001). Meanwhile, integration with other meta-heuristics can provide some relief but it will require users to set extra control parameters too.

Several studies have been carried out to automatically tune parameters of the Bees Algorithm. Nevertheless, most applications so far have typically adopted an exclusive parameter setting for each particular problem. To encourage the use of the algorithm by new and inexperienced persons, having a single parameter configuration that works across a range of problems would be desirable. In fact, an investigation performed by Crossley et al. (2013) suggested that there is only a slight improvement between untuned and tuned Bees Algorithm. The study recommended tuning the neighbourhood size if the problem space or dimensionality is large. However, it did not take into account the neighbourhood shrinking strategy. Besides, the algorithm do not memorise previous visited solutions. This contributes to the creation of patches at unprofitable site. Besides, if position of high quality bees is close to each other, overlapping patches can occur. This is a waste of resources since the bees on those patches can be redirected to other search area. Thus, a more effective neighbourhood search technique with the help of memory can enhance the algorithm’s performance.

In addition, for a multimodal landscape where there exists more than one definite optimum, locating additional solutions can be advantageous because they can serve as alternatives if the others are no longer feasible to be implemented. In particular, in engineering design involving shape or structural optimisation, some physical constraints such as reliability, ease of manufacture, and ease of maintenance are difficult to formulate in objective functions (Saveni et al., 1998; Qing et al., 2005; Dilettoso and Salerno, 2006). Having options enables engineers to select the most suitable solution based on their experiences as well as giving them a better understanding of the problem's search space. Multiple solutions are also useful, for example, in digital image analysis where many objects need to be detected at once (Yao et al., 2005; Cuevas et al., 2013), in seismology where multiple fault lines have to be detected (Koper et al., 1999), and in power distribution system where all possible leaking points need to be found (Delvecchio et al., 2005). Nonetheless, the standard Bees Algorithm, just like any other global optimisers, only converges to a single global optimum. In order to solve multimodal optimisation (MMO) problems, the algorithm needs to have a mechanism able to retain multiple solutions over generations.

1.3 Aim and Objectives

The general aim of this research is to further improve the Bees Algorithm's ability to handle continuous optimisation problems. For each new strategy introduced, no additional parameter is needed beside the current ones, with the improvements made mainly focussed on the neighbourhood search.

To achieve the aforementioned aim, the following objectives were set:

i. Develop an improved Bees Algorithm with the aid of a gradient-like method to provide search direction for the algorithm in order to achieve faster convergence.

ii. Develop a strategy to avoid the formation of overlapping patches in the Bees Algorithm so that recruited bees are managed and distributed more efficiently.

iii. Develop a version of the Bees Algorithm that has the ability to detect multiple global optimal solutions in multimodal optimisation problems.

1.4 Research Methods

The research methodology consists of:

i. Reviewing biologically-inspired population-based optimisation algorithms with particular attention to swarm intelligence, and behaviour of honeybees, to identify current trends, research gaps, and potential solutions.

ii. Developing the proposed algorithms using MATLAB, a readily available tool that is widely adopted for creating and executing software for mathematical problem solving.

iii. Evaluating the developed algorithms on continuous mathematical benchmark problems encompassing various landscapes, and on the problem of training Artificial Neural Network, as well as comparing the results with other swarm algorithms to verify and validate the effectiveness of the proposed methods. For multimodal algorithms, only multimodal numerical functions were used for test purposes.

iv. Analysing the statistical significance of the results using the Student's *t*-test, a well-known tool for statistical significance testing.

1.5 Thesis Outline

The remainder of this thesis is organised as follows:

Chapter 2 reviews the definition of optimisation along with conventional methods used to solve optimisation problems. It also highlights current population-based algorithms inspired by the field of biology. A taxonomy to classify the algorithms is given based on the subject of the inspiration. Specifically, this chapter deals with swarm-based algorithms and focusses more on those algorithms emulating the foraging behaviour of bees. The Bees Algorithm's operation, its applications, and variants are surveyed in detail.

Chapter 3 presents the pseudo-gradient method and the way it is implemented in the Bees Algorithm. Four versions of the Pseudo-Gradient Bees Algorithms are introduced, each differing in terms of the relationship between the neighbourhood and the pseudo-gradient direction as well as the distribution of recruited bees. The modified algorithms were tested on several numerical benchmark functions and their convergence speeds were compared with the Standard Bees Algorithm. Comparisons were also made with other swarm-based algorithms. In addition, the best version of the Pseudo-Gradient Bees Algorithm was utilised in the training of an Artificial Neural Network for modelling an Exclusive-OR gate. Statistical testing was carried out on the results obtained.

Chapter 4 introduces a strategy to avoid the formation of overlapping patches. Two versions of the Patch Overlap Avoidance Bees Algorithm were developed: (1) the Bees Algorithm with standalone Patch Overlap Avoidance strategy, (2) the Bees Algorithm with the Patch Overlap Avoidance and Pseudo-Gradient strategies. The performances were evaluated against the best version of Pseudo-Gradient Bees Algorithm, and the same swarm-based algorithms used in the previous chapter. Applications to the training of Artificial Neural Networks were also implemented together with statistical analysis of each experiment.

Chapter 5 proposes the use of the Hill-valley mechanism originally developed by Ursem (1999) to the Bees Algorithm. By doing so, the Bees Algorithm's ability is extended to locate multiple optimum solutions without the need for niching parameters. This Extended Bees Algorithm was also equipped with the Pseudo-gradient procedure. Another variant of the Bees Algorithm for multimodal application modified how the Patch Overlap Avoidance strategy is instigated in the Extended Bees Algorithm as a way to reduce the number of function evaluations. All variants were tested on continuous multimodal functions, and compared to other multimodal swarm-inspired algorithms that also do not have any niching parameters.

Chapter 6 summarises the contributions and conclusions of this research. Suggestions for further investigations are provided in the Chapter.

CHAPTER 2

SWARM INTELLIGENCE IN OPTIMISATION

2.1 Preliminaries

This chapter reviews the literature in the area of intelligent optimisation, focusing on optimisation methods based on swarm intelligence, in particular, the Bees Algorithm and its applications.

2.2 Optimisation

Optimisation can be defined as a process of searching for the best possible solution to a problem. Mathematically, it is a technique for finding a combination of parameters to minimise or maximise objective functions, i.e., a quantitative measure of a system's performance, subject to some constraints on the variables ranges:

$$\min_{\boldsymbol{x}\in\boldsymbol{R}^N}(\text{or}\max)\ f(\boldsymbol{x}), \boldsymbol{x} = (x_1, x_2, \cdots, x_N), \tag{2.1}$$

where

f = the objective or cost function(s),

$\boldsymbol{x}$ = the parameters to be optimised called design or decision variables, can be continuous, discrete or a mixture of both,

$\boldsymbol{R}^N$ = the design/search/solution space in real value,

N = number of decision variables.

Today, optimisation can be found in various disciplines ranging from mathematics, engineering, computer science, and finance. Real-life problems spawned from these areas are complex due to characteristics such as nonlinear objective function, large solution space size, multimodal, non-convex surface, high dimensional data, and noise. The theory of computational complexity categorises these problems as NP-hard or NP-complete (Consoli, 2006; Abidin et al., 2011) because computational times increase exponentially as the size of the problem increases, thereby limiting the capacity of the computer's memory (Bianchi et al., 2009; Alia and Mandava, 2011; Brownlee, 2011; Sadiq and Hamad, 2012).

Conventional optimisation algorithms such as Gradient Descent and Conjugate Gradient rely on derivative information for fast convergence to the global optimum and high accuracy solutions. Thus, objectives and constraint functions need to be differentiable to search the solution space near an initial starting point. Beside, they also require a good initial value, as well as continuous variables, and objective functions instead of discrete variables or a mix of continuous and discrete variables (Kaveh and Talatahari, 2010; Geem et al., 2001).

Complex optimisation problems are not easily differentiated thereby necessitating the use of assumptions and/or modifications optimisation models of its parameter by variables rounding or constraints softening. However, these will affect the quality of the solution as the model is not easily validated in real situations, but without them the algorithm will fail and be trapped in the local optima. Moreover, fast convergence using conventional methods can increase the probability of missing important points due to minimum number of calculations. Overall, classic optimisation algorithms are incapable to adapt flexibly and efficiently to the solution procedure of complex optimisation problems because there is no universal solution approach that can be applied to problem design (Alatas, 2010).

Consequently, heuristics were developed as basic approximate techniques that search the solution space to find good and feasible solutions (near optimal) within a reasonable computational time and with reasonable use of memory without any loss of subtle nonlinear characteristics of the problem's model and without any requirement of complex derivatives or careful choices of initial values. The term comes from the Greek verb *to find*. Discovering the optimum solution using these methods is usually by trial and error which uses a probabilistic rule instead of deterministic (Consoli, 2006; Bianchi et al., 2009). Greedy-based search is a type of heuristic (Blum et al., 2011; Brownlee, 2011). These algorithms work most but not all the time and there is no guarantee that a heuristic that works on one problem can work on another. However, this can be improved by adding structural information such as nearest neighbours and/or ordered graphs. In general, these techniques are suitable when it is not necessary to find the best solutions and finding good solutions that are easily reachable suffices (Bang et al., 2010; Yang, 2011).

Heuristics can be categorised into two types depending on how solutions are built. They are (Consoli, 2006; Shah-Hosseini, 2008; Bianchi et al. 2009):

i. Constructive

   This type of heuristic builds solutions from scratch by gradually joining together solutions' components/pieces to the initially empty solutions after the other solution is completed. However, the solution quality tends to be low.

ii. Local

   This heuristic starts from a complete, pre-existent/current solution and tries to improve it over time by modifying some of its component. It is slower than constructive heuristics.

2.3 Metaheuristics

In 1986, Glover coined the term 'Metaheuristics' which includes the Greek prefix "meta" meaning beyond or in an upper level to refer to heuristics that are general purpose in nature (Glover, 1986). These methods combine one or more heuristics (maybe high or low level procedures such as random or local search) in a higher level with strategies for exploring search space efficiently and effectively to provide balance between exploitation of the accumulated search experience (intensification) and the exploration of search space (diversification). The region with high quality solution can then be quickly identified and time is not wasted searching already explored regions of those that do not have high quality solution. It performs better than heuristics as it can avoid premature convergence and stagnation at suboptimal points.

Previously known as modern heuristics (Consoli, 2006), today the term heuristics and metaheuristics are sometime used interchangeably (Yang, 2011). Due to its flexibility and adaptability, this technique can be applied to different optimisation problems with few modifications to suit specific problems. It refrains from making any assumptions or simplifying the original form of the optimisation problem. Most of these methods also employ some form of memory (long term or short term) based on its search experience to guide future searches (Consoli, 2006; Alia and Mandava, 2011; Brownlee, 2011). Overall, metaheuristics is the concept applied where a good (but not necessary optimal), fast, and cheap solution is sought.

As mentioned earlier, a balance between exploration and exploitation is an important concept in metaheuristics (Consoli, 2006; Bianchi et al., 2009; Akbari et al., 2010; Yang, 2011):

i. Diversification/exploration (global search)

   It generates diverse solutions to explore the search space on the global scale. It will avoid trapping in the local optima while increasing the solutions' diversity. When the search process starts, it needs to compute the value of every different point in the search domain in order to find the promising areas.

ii. Intensification/exploitation (local search)

   It focuses on the search in a local region (neighbourhood) by exploiting information from the current good solution found in this region. The algorithm then needs to investigate promising zones to find the local optimum.

In addition of the above components, an appropriate selection mechanism is needed to choose the best solution to ensure convergence and speed up the process. With these steps, the best local optimums found in the different areas are the candidate solutions hoping to be as near as possible to the global optimum (Yang, 2011).

Moreover, typical in all metaheuristics algorithms are parameters that need to be tuned in order for the algorithm to achieve a good optimum solution. This can be accomplished by trial and error as well as depending on the user's skill. However, metaheuristics with fewer adjustable parameters are normally favoured (Abu-Mouti and El-Hawary, 2012).

Despite their advantages, metaheuristics suffer from slower convergence if compared to classical optimisation techniques due to huge iterations as a result of lack of derivative information used. Therefore, there is a need for a faster algorithm which leads to the hybridisation of metaheuristics with local search techniques.

More advanced metaheuristics is called hyper-heuristics. This strategy can modify their parameters online or offline to improve the efficacy of the solution or efficiency of the computation. They can employ machine learning method and adapt their search behaviour by modifying the application of the sub-procedures or the procedures used when operating on the space of heuristics which in turn operate within the problem domain (Brownlee, 2011).

Subsequently, metaheuristics can be categorised into two solution types to proceed at each iteration or searches (Consoli, 2006; Shah-Hosseini, 2008; Alia and Mandava, 2011; Brownlee, 2011; Yang, 2011; Binitha and Sathya, 2012):

i. Single-point/trajectory-based

   It generates single solution; at each time-step, describing a curve (trajectory) in the search space during the progress of the search. Example: Simulated Annealing (SA);

ii. Population-based

   It works with a set of solution and trying to improve them; compute simultaneously a set of points at each time-step of the search process, describing the evolution of an entire population in the search domain, while providing a natural and intrinsic way for the exploration of search space. However, it depends on how the population is manipulated such as the Genetic Algorithm (GA).

2.4 Biologically-inspired Population-based Metaheuristics

Between the two types of metaheuristics, population-based methods have generated more interest in the field of optimisation and a lot of attention has been given to nature-inspired algorithms, especially ones based on biology. Figure 2.1 illustrates the taxonomy of nature-inspired population-based metaheuristics. Biologically-inspired population-based algorithms can be categorised according to evolution-based, water-based, ecologically-based, immunologically-based, and swarm-based. Henceforth, this taxonomy can serve as a guideline to group similar algorithms in the future. A brief description on the algorithms is provided but more emphasis is given on the bee-based algorithms because the current trend seems to be in this area. Nonetheless, there are nature-based algorithms inspired by physics and those based on human behaviours such as their sociology and/or pschology. Table 2.1 lists the algorithms based on this taxonomy. This list is not exhaustive as more and more algorithms are being developed in this area.

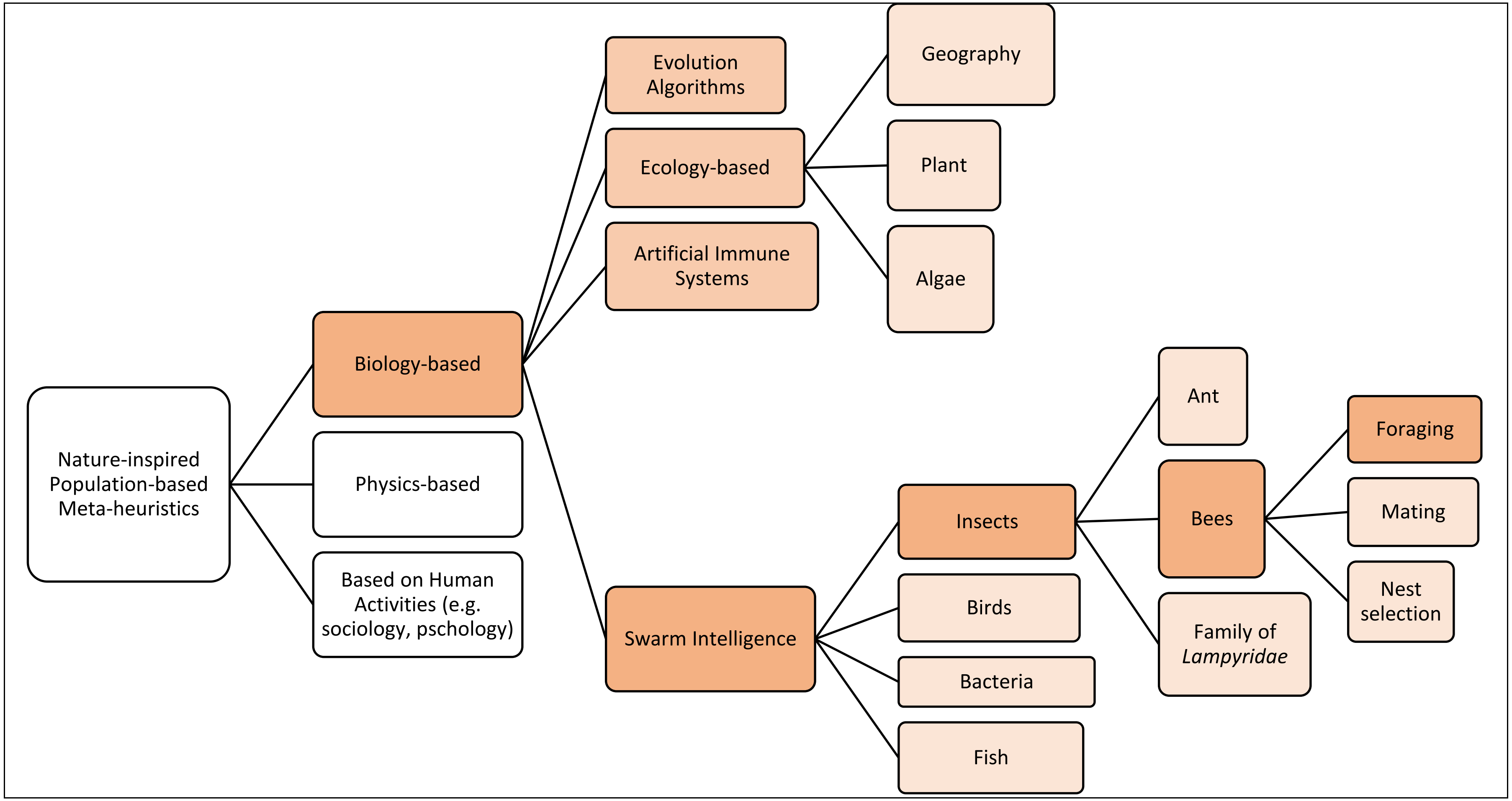

Figure 2.1: Taxonomy of nature-inspired population-based metaheuristics

Table 2.1: List of nature-inspired population-based metaheuristics

SOURCE OF INSPIRATION		ALGORITHM	ORIGINAL AUTHOR(S)
Physics-based			
		Big Bang-Big Crunch	Erol and Eksin (2006)
		Central Force Optimisation	Formato (2007)
		Gravitational Search	Rashedi et al. (2009)
		Charge System Search	Kaveh and Talatahari (2010)
Based on Human Activities			
		Harmony Search	Geem et al. (2001)
		Imperialist Competitive Algorithm	Atashpaz-Gargari and Lucas (2007)
		Teaching-Learning Based Optimisation	Rao et al. (2011)
		Brain Storm Optimisation	Shi (2011)
Bio-inspired			
Evolutionary Algorithms		Genetic Algorithm	Holland (1975)
		Genetic Programming	Koza (1992)
		Differential Evolution	Storn and Price (1995)
Ecology	**Geography**	Biogeography-based Optimisation	Simon (2008)
	Plant	Invasive Weed Optimisation	Mehrabian and Lucas (2006)
		Paddy Field Algorithm	Kong et al. (2012)
	Algae	Artificial Algae Algorithm	Uymaz et al. (2015)
Artificial Immune Systems		Negative Selection Algorithm	Forrest et al. (1994)
		Clonal Selection Algorithm	De Castro and Von Zuben (2000)
		Artificial Immune Network	De Castro and Von Zuben (2001)
		Dendritic Cells Algorithm	Greensmith et al. (2005)

Table 2.1: List of nature-inspired population-based metaheuristics (continued)

SOURCE OF INSPIRATION				ALGORITHM	ORIGINAL AUTHOR(S)
Bio-inspired					
Swarm Intelligence	**Insect**	**Ant**		Ant System	Dorigo et al. (1991)
				Ant Colony System	Dorigo and Gambardella (1997)
				Ant Colony Optimisation	Dorigo et al. (1999)
		Bees	**Foraging**	Bee Colony Optimisation	Teodorović and Dell'Orco (2005)
				Artificial Bee Colony	Karaboga (2005)
				Bees Algorithm	Pham et al. (2005)
				Bee Swarm Optimisation	Akbari et al. (2010)
			Mating	Marriage Bees Optimisation	Abbass (2001)
				Honey Bee Mating Optimisation	Haddad et al. (2005)
			Nest Selection	Bee Nest-Site Optimisation	Diwold (2011b)
		Family of *Lampyridae*		Glow-worm Swarm Optimisation	Krishnanand and Ghose (2005)
				Firefly Algorithm	Yang (2010a)
	Bird			Particle Swarm Optimisation	Eberhart and Kennedy (1995)
				Cuckoo Search	Yang and Deb (2009)
	Bacteria			Bacteria Foraging Optimisation	Passion (2002)
	Fish			Fish Swarm Algorithm	Li et al. (2012)
				Great Salmon Run	Mozaffari et al. (2012)

2.4.1 Evolutionary algorithms (EAs)

One of the earliest population-based optimisation algorithms inspired by biology is a group of algorithms dubbed Evolutionary Algorithms (EAs) due to their emulation of the Darwin's Theory of Evolution. The main algorithm process consists of iterative evolution through selection, reproduction, and survival of the fittest as the following steps:

1. Initialise population of candidate solutions
2. Evaluate individuals in population
3. While (stopping criterion is not met)
4. Select parents to generate offspring
5. Reproduction of offspring through genetic operators (i.e. crossover, mutation)
6. Evaluate new individuals
7. Select new generation based on fitness
8. End while

Typically, algorithms of this class will differ in terms of their variables' representation (i.e. real, binary), the use of the genetic operators, and the type of selection mechanism. In chronological order, the following are the algorithms belonging to this group (Bäck and Schwefel, 1993; Downing, 2010; Das and Suganthan, 2011; Binitha and Sathya, 2012):

1962	Evolutionary Programming (EP)
1965	Evolutionary Strategy (ES)
1975	Genetic Algorithm (GA)
1992	Genetic Programming (GP)
1995	Differential Evolution (DE)

Among the algorithms, GA is the most popular due to its broad range of applications especially in combinatiorial problems such as job shop scheduling, and process planning (Krishnanand et al., 2009; Raudenská, 2009; Zang et al. 2010). In GA, variables are represented in binary string called chromosome. The main parameters in GA are size of population, number of generations, crossover (sometimes called recombination) rate, and mutation rate. However, randomness in the algorithm leads to long computation time, and the parameters need to be tuned to each specific problem in order to obtain feasible solutions (Krishnanand et al., 2009; Zang et al. 2010).

Hence, recent development in EAs is DE as a faster algorithm has simpler and straightforward coding, less number of parameters, and low space complexity (ideal for large scale problems) (Das and Suganthan, 2011). They found that DE is an actively researched algorithm producing variants based on trigonometric mutation, arithmetic recombination, DE/rand/1/Either-Or, and opposition-based learning. In comparison with GA, DE uses real numbers, vector-based mutation, single vector trial creation through crossover, and equal chances for all individuals in the selection scheme. According to them, current studies in the algorithm involve self-adaptive parameters and convergence analysis.

As earlier mentioned, conventional optimisation techniques including heuristics and metaheuristics have inherent disadvantages. These can be rectified by hybridising some of these forms together to solve optimisation problems. GA has been integrated with other methods like the hill-climber, SA, and Tabu Search (TS) to solve various real world optimisation problems (Talbi, 2002). Other than merging with gradient-based methods to speed up the search process, some researchers developed a gradient-like procedure to be used together with EAs so that there is no need to differentiate the objective function. Pham and Jin (1995) devised a gradient-like reproduction operator to be incorporated with the conventional GA, and found that a feasible optimum solution can be discovered in less generation than the common GA. In their method, the best individual information with some test candidates gauged the direction that eventually led to the global gradient. This strategy has a slight similarity with PSO that uses its best particle to guide its search.

In contrast, Solomon (1998) utilised a weighted sum based on the difference between the test and the current position pointing towards the direction of the approximate global gradient. Together with the adaptive step size version of EP, it reduced the number of generations taken to reach the optimal solution. However, this procedure is unsuitable for large scale problems since all offspring are consumed in the calculation of the estimated gradient. This will ultimately increase the number of function evaluations if it is used as the measure of time instead of the number of generations.

Abbas et al. (2003) conceived a fairly similar approach known as discrete gradient where approximated sub-gradient (discrete gradient) was the difference between fitness values of the current and the previous position. It was mixed with ES, but here the discrete gradient was applied to all individuals in the population only at the initial generation. Afterwards, it was applied to the best so far solution.

Moreover, Lin et al. (2006) developed a rather similar strategy in an application of multi-objectives robust active suspension design of light rail devices. The estimated gradient direction was computed by dividing the difference between objective function of the current position and the previous position with their related position for each dimension.

In addition, Hewlett et al. (2007) subtracted the positions of the superior solution from the present and past population which resulted in individual errors. Newly generated population are located near the point of minimum of these errors which was the estimated gradient line. The author claimed that this feedforward system did not reach the solution asymptotically, unlike ordinary gradient-based technique.

EAs especially GA also pioneered the use of several niching techniques for diversity maintenance to arm the algorithm with the capacity to find multiple extremum in multimodal optimisation (MMO) problems. As regular optimisation algorithms only detect the global optimum, the ability to locate more than one distinct optimum solution is helpful as a back-up if the other solution is no longer feasible to implement in real world situations. Example of the niching methods are crowding (Thomsen, 2004; Qing et al., 2008), fitness sharing (Beasley et al., 1993; Miller and Shaw, 1995), and speciation (Li et al., 2010; Stoean et al., 2010; Shen and Xia, 2012).

2.4.2 Ecology-based algorithms

Ecology-based algorithms are centred on the natural ecosystem with the interaction of organism and the environment. One of such interaction is cooperation between species. In Biogeography-based Optimisation (BBO) algorithms (Simon, 2008), which are based on the immigration and emigration of species between habitats, information is shared between potential solutions. Each solution is an island with suitability index variables (SIV) that characterise their habitability. The fitness is called habitat suitability index (HIS). By sharing of SIV, High-HIS solutions will emigrate and the Low-HIS will immigrate. This will improve the solutions and facilitate the evolution process. Two main operators of this algorithm are migration and mutation, with the latter the means to provide exploration. Compared to other algorithms, BBO does not kill the old population but is modified through migration.

Another popular algorithm under this classification is Invasive Weed Colony Optimisation (IWO) proposed by Mehrabian and Lucas (2006). Through adaption with the environments, weeds invade the areas deserted due to inappropriate ploughing after the ground was filled with other vegetation. Changes in behaviour occur as the space becomes thick when the colony grows which makes the condition disadvantageous for unfit individuals. The algorithm begins with the initialisation of the population with random position. Then, fitness of each individual in the population is evaluated before being ranked according to their corresponding fitness. Weeds are allowed to produce seeds based on its fitness, and the highest and lowest fitness of the colony. The generated seeds are then placed randomly in the search space but with reduced standard deviation of the random function compared to the initial value. The maximum numbers of plants from the best plants reproduced are selected (Krishnanand et al., 2009). As surveyed by Binitha and Sathya (2012), IWO has been utilised in solving synthesis of linear antenna array, tuning of Proportional-Integral-Derivative (PID) controller, and tuning of Artificial Neural Networks (ANNs) to name a few. Additionally, IWO also has a variant that deals with MMO which uses the crowding strategy (Majumdar et al., 2012).

2.4.3 Immunology-based algorithms

Artificial Immune Systems (AIS) are algorithms based on immunology or stemmed from adaptive vertebrate immune system in order to protect itself from the invasion of pathogens. Two types of cells associated with this fighting mechanism are T-cells and B-cell with their place of maturation in the thymus and bone marrow, respectively. The four types of immune system models typical in developing an optimisation algorithm are clonal selection, immune network, negative selection, and danger theory (Timmis et al., 2008a; Timmis et al., 2008b).

The Clonal Selection Algorithm (CLONALG) is based on the cloning process in the human immune system that has the following characteristics: immune recognition, reinforcement learning, feature extraction, immune memory, diversity, and robustness. The only operator is mutation; thus it is the deciding factor of the algorithm's efficiency. The algorithm starts with the initialisation of antibodies with antigens representing the value of objective function to be optimised. Then, fitness evaluation is done with the best antibodies cloned. Hyper-mutation is employed to clones in inverse to their fitness where the best will be less mutated and the worst are mutated the most. The new and old clones are evaluated again with the best surviving to the next generation (De Castro and Von Zuben, 2000; 2002).

On the other hand, aiNet (artificial immune network) is an algorithm founded on the theory of immune network proposed by De Castro and Von Zuben (2001) for data mining before extending to MMO (De Castro and Timmis, 2002). This algorithm considers the analogy of the B-cells that is not only suppressed by non-self-antigens but also by other interacted B-cells thus creating two subpopulations. The first is to create initial immune network while the second is trained by non-self-antigens. Each cell in the search space is separated by affinity which is the Euclidean distance between two cells. After cloning, just like the process in CLONALG, elitist selection mechanism is enforced. Then, network suppression eliminates similar cells to avoid redundancy. With that, a number of random cells are generated to the current population. Termination criterion is determined by the population's size of memory. This algorithm can be run using a small number of population compared to CLONALG, and is more adaptive to the changing environment.

Nevertheless, through the incorporation of niching-like parameters such as affinity and suppression threshold aiNet is able to solve MMO problems due to the algorithm being devised as a clustering algorithm early in its development. This variant of aiNet is called opt-aiNet. Coelho and Von Zuben (2010), de Franca et al. (2010), Woldemariam and Yen (2010) tested opt-aiNet with slight modifications to further enhance its performance in several multimodal numerical benchmark functions. It has also been utilised to find alternative designs of electromagnetic devices (Campelo et al. 2006).

In contrast, algorithms based on negative selection model simulate the generation of T-cells where the receptors are made through a pseudo-random genetic rearrangement process. Then, the cells will undergo censoring (negative selection) in the thymus where cells that react towards self-protein are destroyed. Only cells that bind with self-protein are allowed to leave the thymus and circulate throughout the body. Henceforth, Forrest et al. (1994) developed an algorithm based on this self/non-self-discrimination as change detection method where in the generation process, candidates of detectors are generated randomly and censored by trying to match self-samples. Candidates that match are eliminated and the rest are kept. The set of detectors kept are then checked for whether an incoming data instance is self or non-self. Data that matches any detector are considered non-self and treated as an anomaly.

Conversely, danger theory offers an alternative to negative selection in which antigen presenting cells are activated by an alarm signal (danger signal) emitted by ordinary cells injured due to pathogens attack. This signal is necessary as a co-stimulator to T-cells reaction. This means that the adaptive immune response is controlled by the action of innate immune cells called dendritic cells. Greensmith et al. (2005) exploited this analogy in the context of a data signal at any time in computer system.

Overall, AIS-based algorithms are mainly applied in data mining, automation and design, bioinformatics, text processing, pattern recognition, clustering, and classification problems. Hybrids of these algorithms with Fuzzy System (FS), ANNs and EAs have also been done (De Castro and Timmis, 2003; Dasgupta et al., 2011; Mohamed Elsayed et al., 2012). In addition, Zhang and Yen (2013) modified the classic gradient-based mechanism into a quasi-gradient method to be used with the AIS optimisation process. The estimated gradient is the direction pointing from the worst clone to the best which will guide the antibody to the optimal rapidly. However, it was noted that the direction vector will not converge to zero thus requiring additional stopping criterion for the algorithm.

2.4.4 Swarm-based algorithms

Swarm refers to the collective behaviour of organisms such as social insects (ants, bees), school of fish, flock of birds, and bacteria. Complex tasks are performed through decentralised control and self-organising that lead to the emerging behaviour (Bonabeau et al., 1999). The intelligence of the swarm due to the interactions between individuals/agents inspired a lot of optimisation metaheuristics. Among the most prominent approaches are Ant Colony Optimisation (ACO) which is based on the food foraging behaviour of ants and Particle Swarm Optimisation (PSO) that imitates the flocking of birds searching for food. A few other algorithms in this class surfaced recently such as Bacterial Foraging Optimisation (BFO), Firefly Algorithm (FA), and Fish Search Algorithm (FSA). Inspired by the success of ACO, bee-based algorithms are one of the most researched topics as a solution to find optimal value in complex problems. The following provides descriptions of the mentioned algorithms.

2.4.4.1 Ant Colony Optimisation (ACO)

The core of the Ant Colony Optimisation (ACO) algorithm lies in the indirect communication based on the trail leaved by pheromone-laying ants which is call stigmergy that enables them to find the shortest path from nest to the food source. Thus, the original ACO, Ant System (AS) developed for combinatorial problem, specifically the Travelling Salesman Problem (TSP), attempted to exploit this pattern (Dorigo et al., 1991). However, it was less superior than algorithms for solving TSP at that time hence motivating research on creating better variants of the algorithm such as Elitist AS, Ant Colony System (ACS), Max-Min AS, Ranked-based AS, and Hyper-cube AS (Blum, 2005; Dorigo et al., 2006; Dorigo and Stützle, 2010). In 1999, Dorigo and colleagues formalised the ACO framework. Thus, any variants that used this structure belong to the group of ACO. As a whole, the steps in the algorithm are the iterative process of:

i. Constructing the candidate solutions via pheromone model which is a parameterised probability distribution;
ii. Modification of pheromone values by the candidate solutions with bias future sampling towards high quality solutions.

Since many practical optimisation problems can be formulated as continuous optimisation problems and the original works are in discrete domain, it sparks an interest in developing ACO for continuous optimisation with the best being ACOR (ACO for real-valued continuous optimisation) which utilises a continuous probability density function (Socha and Dorigo, 2008). This density function is produced for each solution construction from a population of solutions that the algorithm keeps at all times. Before starting the algorithm, the archive size or the population of *k* is filled with random solutions. Then, at each iteration the set of generated solutions is added to the population and the same number of the worst solutions are removed from it with the aim is to bias the search process towards the best solutions found during the search (Blum, 2005; El-Abd, 2012).

Like previous metaheuristics, besides the usual research on applications, the trend is towards hybridisation with either local searches or other classical AI or Operation Research (OR) methods. The success in hybridising an ACO is due to the constructive nature of the algorithm with, for example, incorporating Beam Search and/or Constraint Programming in ACO. Other studies on this algorithm include the definition of solution components and pheromone trails, and balancing the exploration and exploitation. With the advances in parallel computing hardware, investigation has already been conducted in implementing ACO in that subject.

Additionally, Angus (2006) implemented the niching methods of fitness sharing and crowding techniques separately with ACO to test numerical function and Travelling Salesman Problem modelled as MMO. He found that ACO with crowding performed better than ACO with fitness sharing. According to Dorigo and Stützle (2010), some researchers are working on other ant-based algorithms inspired by ant’s foraging and path marking, as well as other ant behaviour such as brood sorting, division of labour, and cooperative transport.

2.4.4.2 Particle Swarm Optimisation (PSO)

Eberhart and Kennedy (1995) simulated the social behaviour of bird flocks by initiating the Particle Swarm Optimisation (PSO). Each bird is a volume-less particle representing the potential solution determines by its position and the corresponding objective function value. This particle flies through a hyper-dimensional search space with dynamically adjusted velocity based on its own flying experiences and its members of the swarm. Unlike EAs, PSO does not use any selection operators which mean it does not apply the survival of the fittest concept (Shi and Eberhart, 1999). Instead, all the particles are kept throughout the algorithm’s run with the updating of velocity which consists of three terms. They are the previous particle’s velocity, cognition component, which is the particle’s best position so far, and social component determined by the swarm’s best position. The updated velocity revises the position of the particles which is akin to mutation operator in EAs. Fast convergence can occur as the particles can only fly in a limited number of directions towards an expected good area. However, this can also cause the algorithm to experience premature convergence at the local optima, and raise the issue of stability if the particles fly out of the solution space.

A lot of PSO variants have been developed to improve the performance of the algorithm. Inertia weight was first added to balance the exploration and exploitation phase. By linearly decreasing the value from large to small towards the end of the run, the algorithm will perform diversification earlier and intensification later. This method resembles the temperature parameter in SA or in the step size in EAs. It has also been found that without the first term of the velocity update equation, the algorithm will act as a local search, and with it, more global (Banks et al., 2007). Next, velocity clamping is added to avoid instability. There is also a variant with constriction similar to inertia weight but by using this, velocity clamping is no longer necessary. Other researchers also attempted to modify other parameters of PSO by making them more adaptive as well as increasing the population's diversity. Another active area of research in PSO is developing new topology which will affect the way particles interact with their neighbours. Hybridisations of PSO with other approaches are also a popular topic in the current research but most are done to suit a specific problem. PSO has been applied to many optimisation domains such as combinatorial, multi-objective, and dynamics. There is also PSO variant for parallel computing. Among the applications in real-parameter problems are electronics, training of ANNs, medicine, and system identification (Song and Gu, 2004; Eslami et al., 2012).

There are currently three versions of Standard Particle Swarm Optimisation (SPSO) used as a guideline for further improvements of the algorithm (Clerc, 2012). In SPSO2006, the swarm size is calculated based on the problem's dimensions. Meanwhile, the velocity updating is performed dimension by dimension in SPSO2007 as well as taking on an adaptive random topology. SPSO2011 ensures that the particles are bounded, i.e. not flying out of the search space. The update equation for velocity is also modified so as not to have any biases on the system of coordinates (Clerc, 2012; Zambrano-Bigiarini et al., 2013). A more through study on the stability, convergence and rotation invariance of SPSO2011 was performed by Bonyadi and Michalewicz (2014), and it was then applied to source seeking tasks for mobile robot (Zou et al., 2015).

Besides EAs, PSO is another algorithm actively involved in solving MMO problems where multi-solutions are sought. For this purpose, niching is commonly needed which requires extra parameters (i.e. niche radius) to be tuned. Nonetheless, most PSO variants that deal with this kind of problem do so by making it adaptive (Nickabadi et al., 2008) or not adding any niche parameter (Li, 2007; Liu et al., 2011). Li (2010) applied PSO with ring topology to MMO without the need to specify any niche parameters by employing the individual particles' local memory to form a stable network, retaining the best solutions found so far while these particles explore the search space more vastly. Other PSO-based niching algorithms are devised by techniques used to solve different domains of optimisation problems such as by using *k*-means clustering algorithm to create sub-swarm as cited by Liu et al. (2011) and fuzzy clustering as cited by Das et al. (2011).

2.4.4.3 Bacterial Foraging Optimisation (BFO)

In 2002, Passino developed a novel optimisation algorithm based on the movements of E. coli bacteria towards a higher concentration nutrients area to maximise energy obtained per unit time. The algorithm consists of four phases namely chemotaxis, swarming, reproduction, and elimination-dispersal. Chemotaxis describes the way bacteria moves by tumbling (random movement), and swimming (continuous movement in single direction) with both of these movements alternating throughout the bacteria life cycle. In this algorithm, swimming is only maintained if the solution keeps on improving. After chemotaxis, cell-to-cell communication between bacteria occurs in the presence of chemical attractant and repellent to form swarm. Next, in the reproduction phase, only half of the swarm with the best fitness are reproduced by splitting them into two to form new population. Finally, parts of the population are randomly selected to be diminished and dispersed into random positions in the environment, and the search space is re-initialised with low probability. Chemotaxis and reproduction are considered in the exploitation phase of the algorithm, while elimination-dispersal is the exploration phase as a way to increase diversity to prevent trapping, and improve the algorithm's global search capability (Parpinelli and Lopes, 2011; El-Abd, 2012).

Similar to other algorithms, researches in BFO focus on the performance improvement to get rid of the algorithm's drawback as much as it can especially in high dimensional and multimodal search space. One way to do this is through hybridisation with other population-based algorithms. Others have tried to make the step size more adaptive by using a large value at the beginning and small value at the end. This is a proven method that can balance exploration and exploitation of the algorithm as cited by Agrawal et al. (2011) in their review on BFO. Besides that, new phase and control parameters were inserted to the algorithm such as method based on quorum sensing; a regulating division mechanism in bacteria to improve communication and cooperation (Tang et al., 2007; Chen et al., 2009; Cho et al., 2009).

2.4.4.4 Algorithm based on family of insects, *Lampyridae*

A family of insects called *Lampyridae* is a group of insects able to emit natural light (bioluminescence) normally used to entice mates and prey. The light produced is due to a pigment called *luciferin*. More pigment means a brighter light (Parpinelli and Lopes, 2011). Two common types of insects in this family are fireflies and glow-worms, and both have attracted the attention of researchers to model optimisation algorithms based on them.

The Firefly Algorithm (FA) is a multi-agent algorithm developed by Yang (2010a) with three rules:

i. All fireflies are unisex. Thus, attraction between fireflies occurs regardless of sex. This is to make the exploitation phase more efficient.

ii. Attractiveness is proportional to brightness, which is inversely proportional to distance.

iii. Brightness correspond to the fitness of the objective function

By dynamically adjusting the algorithm's parameters, the position update equation of the algorithm is somewhere between the behaviour of total random search and PSO search. To ensure convergence, distance between fireflies is decreasing functionally. Later, Yang (2010b) combined FA with the Lévy flight concept, one of the earliest to do so in nature-inspired optimisation algorithms, to make the algorithm converge more rapidly, and provide a more natural way of doing global search. Instead of uniform or Gaussian distribution used in random search, a power-law distributed step length with a heavy-tail is used. Biologists have observed the behaviour of animals using the Lévy flight pattern for quite some time especially in animal food foraging behaviour in a large space. The success of this method has inspired other researcher to use the concept in their animal-based algorithm such as PSO (Gang et al., 2011).

On the other hand, Krishnanand and Ghose (2005) developed the Glow-worm Swarm Optimisation (GSO) to be used in collective robotics applications. It shares a few common behaviours of ACO and PSO especially in terms of position updating of the glow-worm. Probabilistic mechanism is used to select a neighbour that has higher luciferin value than itself and moves towards it. One attractive feature of this algorithm is the ability to divide swarm to disjointed subgroups which increases the chance of convergence to multiple optima in a multimodal domain. Subsequently, Krishnanand and Ghose (2009) integrated into GSO the variable neighbourhood radius to detect the presence of multiple peaks in MMO.

2.4.4.5 Fish Swarm Algorithm (FSA)

The Fish Swarm Algorithm (FSA) is based on the natural schooling behaviour of fish developed by Li et al. (2002). Five fish behaviours associated with this algorithm are:

i. Random – fish look for food and other members of the swarm randomly
ii. Searching – fish discover region with more food will go directly to it quickly
iii. Swarming – fish swim in swarm to avoid danger
iv. Chasing – when a fish in the swarm find food, others will find the food and chase after it
v. Leaping – when a region stagnated, fish leaps to look for food in other regions

The algorithm has three parameters to be set by the user. They are visual scope which defines the distance between two fishes, maximum step length the fish take, and crowd factor which determines the number of fish in an area. The algorithm's performance is said to be sensitive to the first two parameters.

The algorithm has been utilised in the following applications:

i. Training of feedforward neural network (FNN) (Wang et al., 2005)
ii. Optimal multi-user detection (Jiang et al., 2007)
iii. Image segmentation (Jiang et al., 2009)
iv. Bounded constrained problem (Fernandes et al., 2009)

2.4.5 Bee-based algorithms

Bees are social insects that live as colonies consist of queen, drones, and workers. They are able to dynamically allocate tasks between for example, tending of brood, and foraging of nectar, pollen, and water, while keeping track of the environmental change. All of these are performed with no centralised control.

The only function of the queen is to reproduce. The queen will perform mating flight, and mate with multiple drones before the sperms are stored in a spermathecal of the queen for brood production. Unfertilised eggs will become drones while fertilised eggs will become workers and queen (Abbas, 2001).

Workers do most of the tasks in a bee colony with the primary job being food foraging. Bees fly from the colony to more than 10 km to search for potential food sources. When they arrive back to the hive, they will start a ritual called "waggle dance". In the waggle dance, the quality, direction, and distance of the site are broadcasted to attract more bees to forage them. The more profitable the site, the livelier and longer the dance will be (Seeley, 1995).

When a young queen is being reared, the old queen together with a third of the workers' colony will leave the hive, and swarming into a cluster at a nearby temporary site, usually a tree branch. Several hundred scout bees will search for potential nest-site normally in the form of tree cavities or crevices in a building. Again, the waggle dance is performed to let other bees assess the potential site. However, after dancing scouts return to the site for re-evaluation, they fly back to the cluster to dance again. Each time the scout returns, dance circuit decays to approximately 15 dance circuits until the bee ceases dancing. A more profitable site will take longer to decay since the number of dance circuits is high at the beginning. Hence, more bees are recruited for a high quality site. A scout bee will estimate the number of bees found at the site, and if it exceeds a certain threshold, it will return to the swarm and begin making piping signals to prepare the swarm for lift-off to the new site (Passino and Seeley, 2006).

The three behaviours conferred above namely, mating, foraging, and nest selection have inspired a number of optimisation algorithms as will be discussed in the next sections.

2.4.5.1 Algorithms based on the mating behaviour of bees

Abbass (2001a) wrote the first paper that used bees' reproduction behaviour as an analogy in optimisation algorithm called the Marriage in Honey Bees Optimisation (MBO). The algorithm has four parameters that need to be set, namely number of queens, size of queen's spermathecal, number of broods, and number of workers. The mating flight starts with random initialisation of workers, drones, speed, energy, position, as well as genotype for the queens. Then, each queen flies in the space, and probabilistically mates with the drone met based on her speed. At the beginning when the queen's speed is high, it makes large steps in space thus increasing the probability of mating. However, this speed reduces after each transition due to the decrease in the neighbourhood covered by the queen, just like the annealing process in SA. For mating to be successful, the drone has to pass the probabilistic rule before its sperm is stored to the list of partial solutions inside the spermathecal. Breeding starts as soon as the mating flight is over with the production of broods by using crossover and mutation operator between the queen's genome, and the randomly selected sperm stored. After that, workers improve the broods by applying a set of different heuristics or local search. This is where the algorithm differs from the popular evolution method, GA. By using the local search method iteratively, the solution's improvement does not solely depend on crossover and mutation operators. Replacement happens when there are fitter broods than the queens. Broods that are not replaced with the queens are all killed, and a new mating flight begins.

The algorithm was first tested on propositional satisfiability (SAT) problems (Abbass, 2001a), with further improvements on the annealing function from the same author (Abbass, 2001b), and for 3-SAT (Abbass and Teo, 2001). Later, Haddad and Afshar (2004) applied the algorithm in discrete domain reservoir operation as well as benchmark mathematical test functions (Haddad et al., 2005) when they renamed the algorithm to the Honey Bees Mating Optimisation (HBMO) algorithm with both names used interchangeably ever since. Various improvements and applications soon followed with reservoir operation optimisation in continuous space by Afshar et al. (2007), and hybridisation with clustering algorithm *k*-means done by Fathian et al. (2007). Meanwhile, Yang et al. (2007a, b, and c) applied different local searches such as Nelder-Mead, and Wolf Pack Search and reduced the algorithm parameters. They also performed convergence analysis of the algorithm using Markov Chain. Marinakis et al. (2008) solved the location routing problem by using the algorithm with the Multiple Phase Neighbourhood Search-Greedy Randomised Search Procedure in the initialisation phase instead of random initialisation. They also used adaptive memory as the crossover operator, and the Expanding Neighbourhood Search in the local search phase. Next, they applied the same method on variants of TSP (Marinakis et al., 2009a and 2011), and vehicle routing problem (VRP) (Marinakis et al. 2010).

Other applications include forecasting (Pai et al., 2009), electrical power systems (Arefi et al., 2009), image processing (Horng, 2010), and feature selection (Marinakis et al., 2010). Vakil-

Baghmisheh and Salim (2010) developed a different selection method through the postponing of broods' replacement with the queen towards the end of the reproduction process while Bahamish et al. (2010) exercised the Metropolis selection criteria instead of the Greedy selection for predicting protein tertiary structure. Niknam (2011) tackled the multi-objective optimisation problem by using fuzzy logic to handle the conflicting objectives and the Chaotic Search for the workers' heuristics. Bernardino et al. (2010) used a single queen bee instead of multiple numbers for their application on communication network, and Poolsamran and Thammano (2011) modified the crossover and mutation operators together with the local search to suit application using the real string instead of binary numbers. The latest uses of the algorithm are on scheduling (Palominos et al., 2012; Yin et al., 2012).

Interestingly, another family of bees in the genus *Bombidae*, the bumblebees has also attracted the attention of optimisation researchers. Most algorithms discussed earlier are inspired by the European honeybee species, *Apis Mellifera*. The main difference between them is in terms of their colony size, where the bumblebees' population is smaller. Also, the bumblebees' queen is the only one that survives over winter.

Marinakis et al. (2009b) proposed the bumblebee algorithm which starts with random initialisation of the bee population. Then, the bees' fitness is evaluated and the best bees are selected as queens with the rest as drones. Next, drones' fitness is ranked to be mated with the queens. As the result, the sperms are stored in the queens' spermathecal just like MBO and HBMO. Reproduction of broods is through the crossover operator with their fitness later ranked with the fittest selected as the new queens, and the rest as workers. The new queens' genotypes are further improved through mutation of the old queens' genotype, and the fittest workers' genotype in the local search with momentum-like equation. The new drones are also generated with the old queens' genotype, and the fittest workers' genotype through random mutation. Drones then fly away from the nest to perform some kind of exploratory search to prevent suboptimal before their fitness being ranked to be mated with the new queens as previously explained. Only the queen with the best solution survives to the next generation while the rest of the population dies. Compared to MBO, workers in this algorithm are the solutions instead of a set of heuristics. While in both algorithms the fittest are the queens but in this method the rest are workers in contrast with MBO where the rest are drones.

To prove the efficiency of the algorithm, it was applied in clustering problems through hybridisation with Greedy Randomised Search Procedure (GRASP). Other applications also include unconstrained mathematical benchmark test functions (Marinakis et al., 2010), and VRP (Marinakis et al., 2011).

2.4.5.2 Algorithms based on the nest-site selection in bees

Much like foraging, communication between bees during hunting for their new nest-site is through waggle dance. However, in the nest-site selection there is dance attrition where scout bees reduce their dance strength over multiple visits to the prospective site. As mentioned in the earlier section, 15 waggle runs were reduced each time a scout bee returned to the swarm cluster thus increasing the rate of consensus building by hindering the broadcasting of less in quality sites (Passino and Seeley, 2006). In addition, there is the need to balance between speed and accuracy during site selection process due to the fact that bees that are looking for a new nest are hanging on a tree branch exposed to the environment. Hence, detection of quorum builds at the best site by scout bees launches the preparation of lift-off by the swarm to the new nest. Through some experiments, the same authors predicted the size of the quorum to be in the range of 15 to 20 bees which is in accordance with nature as witnessed by other researchers.

This decision-making process in honeybees is also perfect to model into an optimisation algorithm. Diwold et al. (2010) performed a biological simulation based on this behaviour in dynamic and noisy conditions, and the results are promising. They further investigated o function optimisation but instead of the typical *Apis Mellifera* behaviour, *Apis Florea* (Asian dwarf honeybees) are used because iteration process in optimisation are more suited to this open-nesting species where the choice of nest are abundant compared to the restricted requirements of the earlier mentioned species. Relocation is possible in *Apis Florea* if the initial solution was suboptimal. Later, the same authors formulated a scheme based on this paradigm with the bees' population placed at a random point in space at initialisation. Then, a fraction of bees that contains the scouts fly randomly to a certain distance from the home point before performing the local search to improve solution. Sites with the best fitness will have higher recruitment with the recruiters move to a random position with a predefined maximum distance from the scout's location. The recruiters will also perform the local search in that area for solution improvement. If there is no improvement, the site is abandoned and recruiters will become scouts. When the best solution is obtained, and there are a number of scouts that exceed the threshold value at that location, the whole swarm will lift-off to the current best solution from their home site otherwise the swarm is randomly placed at a new home point or at its current home. The process is repeated until a feasible solution is located (Diwold et al., 2011a).

In the same year, Diwold et al. (2011b) predicted the protein molecular docking using the scheme they proposed earlier with some modifications. They named the algorithm the Bee Nest-

Site Optimisation (BNSO) algorithm. In this problem, the distance between the scout's potential site and the home site, or the newly found point by recruiter and the scout's location is not fixed but there is a range factor that will reduce the distance over time to stimulate the intensification in the local search using the simple random walk. Moreover, a standard roulette wheel selection based on the fitness probability is employed in the recruitment process, and the site abandonment is induced once no improvement took place after a certain number of iteration. Even though the result is quite promising, it was inferior to PSO in terms of performance thus suggesting future hybridisation.

Furthermore, Paulovič (2011) specifically designed an algorithm for clustering inspired by bees' nest-site selection. The algorithm consists of four phases: proto-hives initialisation, swarming, natural selection, and cluster evaluation. Several hives are inserted in the data space during initialisation. Then, scout bees are positioned pseudo-randomly with the centre being the parent hive. The exhaustive or greedy search is performed around the area to find candidate solutions. Each candidate is ranked-based on the prospective hive fitness using a small radius to the fraction of the full hive radius area with the worst position being eliminated. A bigger radius is exploited at each iteration, and more of the less fit candidates are rejected to allow for the best site to become the new hive. A mechanism called the honey bee drift that occurs in a swarm with low probability will direct scouts to the whole data space to focus on the dataset elements in order to increase the chances of finding a good nest-site in an otherwise undiscovered area. New sites are created in the region of high data density if it was previously destroyed through natural selection. A clear distinction between cluster regions will form due to the cluster of high data density having a big number of hives, and sharper contours.

Kumar et al. (2012) suggested a novel algorithm based on the bees' decision making during house hunting in the multi-objective optimisation of electrical power system using Pareto-optimal, and multi-agent system. In this algorithm, the search area is divided into different fragments with agents sent to search the whole space. Around 30 to 50 agents are usually used for exploration to provide speed-accuracy trade-off. Midpoint of the area is selected as an initial point for the agents to start searching around the region of the best solution in each fragment due to the use of the PSO-like Nelder-Mead method. Every individual's (agent) best solution will be stored in a vector table before comparing them in the end to find the global best.

In general, it is found that algorithms derived from the behaviour of bees during nest selection are still scarce compared to mating and foraging behaviour. Thus, further works can be developed in this area to produce more efficient optimisation techniques.

2.4.5.3 Algorithms based on the foraging behaviour of bees

The algorithms based on the food foraging behaviour of bees are the most famous compared to the other activities of bees. According to Karaboga et al. (2012), three major players in this category are the Artificial Bee Colony (ABC) algorithm, the Bees Algorithm, and the Bee Colony Optimisation (BCO) algorithm with BCO among the first to use this analogy.

The earlier version of BCO is the Bee System (BS) developed for combinatorial problems by Lucic and Teodorović (2001). In this algorithm, scout bees do not have any guidance while foraging because the aim is to find all kinds of food. Thus, this approach is associated with low search cost and low average quality in food. Later, Lucic and Teodorović (2003) combined FS with this method to handle uncertainty in traffic, and transportation engineering problems as a Travel Demand Management system. The artificial bees (agents) in this system employed approximate reasoning, and rules of fuzzy logic in their communication, and action.

To combine both of the approaches above, Teodorović and Dell'Orco (2005) generalised the concept of BS so that it can be used in deterministic and combinatorial characterised by uncertainty. This multi-agent system uses the constructive concept which builds solutions from scratch within execution steps unlike the local-based metaheuristics that improves the current solution. The agents in this method live in a discrete time environment with each bee generating one solution to the problem. There are two phases in the steps of this algorithm; the forward, and backward pass. In the first phase, bees that visit solution's component in a predefined number of moves will create the partial solution, and return to the hive to start the second phase. Here, bees exchange information on the quality of the partial solution and form two types of bees. The first type is recruiters which are bees loyal to the solution while bees of a better solution will have a high probability to keep on advertising, and be chosen for further exploitation. The second type is called uncommitted bees whose food source has been abandoned, and will select one of the advertised bees to exploit. The numbers of both of these types of bees will change dynamically throughout the course of the algorithm's run. Moreover, the selection method for both type of bees is based on the probability using the roulette wheel. In addition, both of these phases will keep alternating until all solutions are completed with the best one to be updated as the global best. This completes one cycle of the algorithm, and the algorithm will repeat its cycle until a certain terminating condition is fulfilled.

Since then, this algorithm has been put into operation for vast hard combinatorial problems such as TSP, VRP, ride matching, optical network's assignment, constrained portfolio, highways' traffic sensors, optimal placement of distributed generation in electric network, optimal facilities location, and scheduling. It has also been hybridised with the rough set approach to solve problem in the supply chain management (SCM) (Teodorović et al., 2011). BCO has also been used together with the parallel computing to increase computation speed. There is also the local-based variant of this algorithm which begins with a complete feasible solution, generated randomly by heuristic, and then perturbed to improve it. This variant has been tested to nurse rostering problem as cited by Teodorović et al. (2011) in their review on applications of BCO.

Nonetheless, the most active algorithm based on the foraging nature of bees is ABC which was introduced by Karaboga (2005). There are three types of bees in this algorithm:

i. Employed bees, which are associated with the specific food source;
ii. Onlooker bees, which watch the dance of employed bees at the hive to choose food source from;
iii. Scout bee, which searches for food randomly.

In this algorithm, the position of food source represents solution to the problem, and nectar amount of a food source reflects the quality or fitness of the corresponding solution. Moreover, the number of food source is equal to the number of employed bees. The algorithm has three parameters that need to be set by the user. There are the colony size with 50-50 amount of employed bees, and onlooker bees; limit refers to the number of trials taken before a food source is abandoned; and maximum number of cycle. Initially, employed bees are placed at random food sources in the space. Then, they search the neighbourhood of the food source to try to improve the solution. Next, they return to the hive to exchange information with onlooker bees. Using the roulette wheel, food source with high probability proportional to the food source fitness is selected for further exploitation by onlooker bees. Once the food source has been exhausted and there is no improvement made after predetermined cycles, it is abandoned. The employed bee associated with the deserted food source becomes the scout bee, and starts searching for a new food source randomly. This abandonment of the food source serves as the negative feedback to the positive feedback of recruitment of onlooker bees in order to balance the search process.

Originally, ABC was designed to solve unconstrained continuous numerical problems consisting of multidimensional unimodal and multimodal functions. It produced better results when compared to GA, DE, and PSO. Then, it was extended to handle constrained problem with one of the method is by using dynamic control mechanism for equality constraints to facilitate the approach to the feasible region of the search space. Other variants of this algorithm cover other domains of optimisation problems such as multi-objective, discrete/combinatorial, and binary/integers. Among major applications of ABC are in the training of neural networks, data mining, wireless sensor network, image processing, variety of engineering problems of different fields, as well as in medical applications (Karaboga and Akay, 2009a). ABC has also been hybridised with local search methods such as the Nelder-Mead, GRASP, and Hooke-Jeeves, as well as other metaheuristics such as EAs, ACO, PSO, BFO, and AIS. Researches to combine this algorithm with new computing paradigms such as the chaotic, and quantum computing have also been conducted. Other modifications or improvements made on the algorithm were to prevent the suboptimal, and to increase the convergence speed. Studies have also been done on the effect of control parameters as well as on the selection, and position updating strategies. In contrast, one study has even suggested the use of Von Neumann topology of PSO in the algorithm.

Recently, the original inventor of ABC developed the quick-ABC (qABC) (Karaboga and Gorkemli, 2012). In this variant, a new position update equation for onlooker bees is introduced to reflect the nature more accurately which differ from employed bees' update positioning. Additionally, the limit to ascertain when a site needs to be abandoned can now be made dependent on the colony size and the problem's dimension. A new parameter, neighbourhood radius is also initiated. These changes were able to improve the local search which made convergence faster compared to the basic ABC on continuous benchmark functions. Later, Karaboga and Gorkemli (2014) studied the effect of the new added parameter on the algorithm's performance. It was found that tuning of the neighbourhood radius is necessary to obtain a good result. Besides continuous optimisation, qABC has also been applied to TSP problem with eight different neighbourhood operators borrowed from GA as well as the nearest neighbourhood tour constriction heuristic as the initialisation strategy (Gorkemli and Karaboga, 2013).

Likewise, ABC has also ventured into the finding of multiple optimal in MMO (Liu et al., 2012). As the basic ABC can only locate one global best solution, memorising the fitness of abandoned food sources has given ABC the aptitude to obtain all possible global and local optima. This multimodal variant of ABC has been applied to detect assortments of conic sections of multiple hyperboles, and circles (Cuevas et al., 2012; Rahkar-Farshi et al., 2014). Further explanations on other variants and applications of ABC can be found in the comprehensive review done by Karaboga et al. (2012).

There are several other algorithms based on the food foraging behaviour of bees. One is the Virtual Bee Algorithm designed by Yang (2005) to work on numerical function optimisation. In this method, a swarm of virtual bees is generated and move randomly in the search space. The nectar source corresponds to the encoded values of the function, and the interactions between agents begin when bees found the sources. Then, the solution is obtained due to the intensity of this interaction. In addition, Drias et al. (2005) developed the Bee Swarm Optimisation to work on the Maximum Weighted SAT problem before extending to a parallel version of the same problem to increase the computational speed. Meanwhile, Chong et al. (2006) used a similar principle to BCO, and solved the job shop scheduling problem, TSP, and the training of neural network for feature selection. Furthermore, Lemmens et al. (2008) compared the non-pheromone-based navigational of bees with the pheromone-based navigational of ants before applying the bee-based approach to mobile robot navigation problem. On the other hand, Lu and Zhou (2008) simulated the honeybees collecting pollen behaviour as a global optimisation algorithm to solve TSP.

By using movement based on probabilistic approach, Sundareswaran and Sreedevi (2008) tested their bee foraging-based algorithm on numerical test functions and compared them to ACOR. In their algorithm, random generated worker bees are forced to move in the direction of elite bees that represent the best possible solution. Bees' step distance of flight is made as a variable parameter in this method. Another algorithm designed to solve combinatorial problem is the Bee Colony-inspired Algorithm by Häckel and Dippold (2009) who used VRP as their test problem. The algorithm has a likeness to ABC. Moreover, McCaffrey (2009) proposed the Simulated Bee Colony to solve a range of combinatorial problems for example, generation of pairwise test vectors, extraction of rules set for clustered categorical data, and graph partitioning. Later, Akbari et al. (2010) used different updating equations for different types of bees that have a reminiscent of ABC and PSO in their algorithm. The algorithm has been extended to cooperative variant, and applied to Dynamic Economic Dispatch problem. While not long ago, Bitam (2012) combined food foraging, and mating behaviour of bees, and proposed the Bee Life Algorithm to be used in cloud computing.

On the matter of bee-based multimodal algorithms for MMO, two techniques are specifically invented for clustering problems. The first is the Bee Nest-site Selection Clustering algorithm (Paulovič, 2011) that has been described previously, and the second algorithm is the Flower Pollination by Artificial Bees (FPAB) developed by Kazemian et al. (2006). FPAB was used to form clusters before serving as the initial cluster centre for another clustering algorithm, the fuzzy *c*-means to reduce any misclassification errors. Just like other clustering algorithms, FPAB is easily trapped in the local optima, thus optimisation modules were used to find the optimal solutions. For examples, set-covering model has been employed with FPAB in the determination of machinable volumes in the production process planning (Houshmand et al., 2009), while hybridisation with BBO and BFO were used in the satellite image classification problem with the former done by Johal and Singh (2010) and the latter by Singh et al. (2011).

In contrast, Rashid et al. (2009) developed an algorithm based on PSO but using the bee's foraging behaviour for its particles. Sub-swarms were created after scouts found niches and foraging bees were recruited to exploit the area set by radius-based niching parameter using the position update equation of PSO. An overlapping of swarm is prohibited to eliminate redundancy by merging the swarm. If there is no improvement in any sub-swarm after several iterations, the swarm is considered converged. Then, the result is recorded in a blackboard similar to the method used in TS, and the swarm is disbanded with the scout sent to perform a random search.

A novel multimodal algorithm based on the collective decision making by bee colonies called OptBee was designed to generate, and maintain population's diversity in order to obtain multiple local optima without losing the ability of global optimisation (Maia et al., 2012). This capability is still lacking in many bee-based algorithms. The algorithm has a niche-like parameter, i.e. inhibition radius, and used probability for recruitment and scouting. Similar with other algorithms based on foraging behaviour of bees, scout bees are involved in random search while recruited bees exploited the sites found by scouts. Although the algorithm is suitable for multimodal domain, in some problems it is outperformed by concentration-based opt-aiNet.

2.5 The Bees Algorithm

As previously discussed, the Bees Algorithm is one of the key player in the foraging bees' algorithm category first developed by Pham et al. (2005). In its basic version, the algorithm performs a kind of neighbourhood search combined with random search. A bee in this algorithm is a *d*-dimensional vector containing the problem variables, and represents a feasible solution to the problem. Solution is represented by the visited site (food source) with a fitness associated with it. The fitness is calculated according to the objective function being optimised.

The algorithm requires a number of control parameters need to be determined by the user:

i. number of scout bees (*ns*),
ii. number of best sites (*nb*) out of sites visited by *ns*,
iii. number of elite sites out of *nb* selected sites (*ne*),
iv. number of bees recruited for *ne* sites (*nre*),
v. number of bees recruited for the other *nb-ne* selected sites (*nrb*),
vi. size of neighbourhood (*ngh*),
vii. stopping criterion (it can be the maximum function evaluation, the minimum error or/and others)

The steps for the Bees Algorithm in its basic form:

1. Initialise the scout population with random solutions.
2. Evaluate fitness of the population.
3. While (stopping criterion not met) //Forming the new population.
4. Select sites for the neighbourhood search.
5. Recruit bees for selected sites (more bees for elite sites), and evaluate fitness.
6. Select the fittest bee from each patch.
7. Assign remaining bees to search randomly, and evaluate their fitness.
8. End While.

The algorithm starts with scout bees placed randomly in the search space. Then, the fitness of the sites visited by scout bees is evaluated. Bees that have the highest fitness are chosen as "selected bees" and sites visited by them are chosen for neighbourhood search. Next, searches are made more detailed in the neighbourhood of the selected sites, assigning more bees to search near to the elite sites which represent the area where the promising solution lies. Normally, bees are chosen directly according to the fitness associated with the sites they are visiting. Subsequently, for each patch only bees with the highest fitness will be selected to form the next bee population. However, there is no such restriction in the nature. This restriction is introduced to reduce the number of points to be explored. Afterward, the remaining bees in the population are assigned randomly around the search space scouting for new potential solutions. These steps are repeated until a stopping criterion is met. The stopping criterion can be the maximum number of generations reached or a satisfactory solution has been found. At the end of each cycle, the colony will have two parts to its new population which are those that were the fittest representatives from every patch, and those that have been sent out randomly. Hence, random scouting and differential recruitment are the fundamental operations of the Bees Algorithm that balances between exploration and exploitation. With this explanation, it is the author's opinion that the Bees Algorithm is the metaheuristics that follows the bees' foraging nature more accurately due to the use of elitism and clear definition of food patches (neighbourhood) to translate the waggle dance performed by bees. However, it is done at the expense of using more tuneable parameters than others.

From the algorithm's control parameters, the size of the bees' population, p can be calculated as below:

$$p = ns + ne \cdot nre + (nb - ne) \cdot nrb \tag{2.2}$$

For random scouting in the initialisation phase as well as for the unselected bees, the following equation is used:

$$x_{rand} = x_{min} + rand \cdot (x_{max} - x_{min}) \tag{2.3}$$

where;

rand is a random vector element between 0 to 1,

x_{max}, x_{min} are the upper and lower bound to the solution vector respectively.

Meanwhile, in the neighbourhood search, the following equation is used for recruited bees:

$$x_{i+1} = (x_i - ngh) + 2 \cdot ngh \cdot rand_i \cdot (x_{max}^i - x_{min}^i) \tag{2.4}$$

To avoid misidentification between the Bees Algorithm and ABC, Table 2.2 lists the crucial disparities between both of these foraging bee-inspired techniques.

Table 2.2: Differences between the Bees Algorithm and the Artificial Bee Colony

<table>
<tr><th>Criteria</th><th>Bees Algorithm</th><th>Artificial Bee Colony</th></tr>
<tr><td rowspan="3">Position update</td><td>In the local search, the recruited bees' position was based on the random distribution guided by the selected bee of each patch with the patch size predetermined by the user.</td><td>The onlookers' position is based on the mutation position between the employed bees and another randomly selected employed bees.</td></tr>
<tr><td>No self-update strategy for the selected bees.</td><td>The employed bees can self-update their position.</td></tr>
<tr><td>Updates are done throughout the problem dimensions.</td><td>Only update the random index dimensions from the overall problem dimensions.</td></tr>
<tr><td>Selection strategy</td><td>The number of recruiters are user-defined for each site selected based on its fitness with the elite sites receiving more bees.</td><td>Bees are recruited using probabilistic approach where the better the solution, the larger the number of bees allocated to the site.</td></tr>
<tr><td rowspan="2">Scout bees</td><td>The remaining scout bees that are not selected for neighbourhood search, arbitrarily re-initialised their position independently in the global scale.</td><td rowspan="2">Only employed bees whose site is abandoned perform random scouting globally.</td></tr>
<tr><td>The same thing happens to selected bees after site abandonment.</td></tr>
</table>

2.5.1 Variants and applications of the Bees Algorithm

Since its inception, the Bees Algorithm has gained a lot of interest. Figure 2.2 that shows a steady increase of numbers of papers published using the Bees Algorithm each year. Initially, the algorithm was tested to various numerical benchmark function up to 10 dimensions (10D), and outperformed other optimisation methods such as deterministic simplex, ACO, GA, and SA in term of speed and accuracy (Pham et al., 2006). Soon after, Li et al. (2010) tested the Bees Algorithm to continuous functions higher than 10D, and compared with the performance of ABC and DE. They concluded that the Bees Algorithm is the better algorithm for multimodal functions while for unimodal, the superior performance was only achieved for higher dimension problems. A similar trait has also been found by Pham and Castellani (2009) and Tsai (2014b).

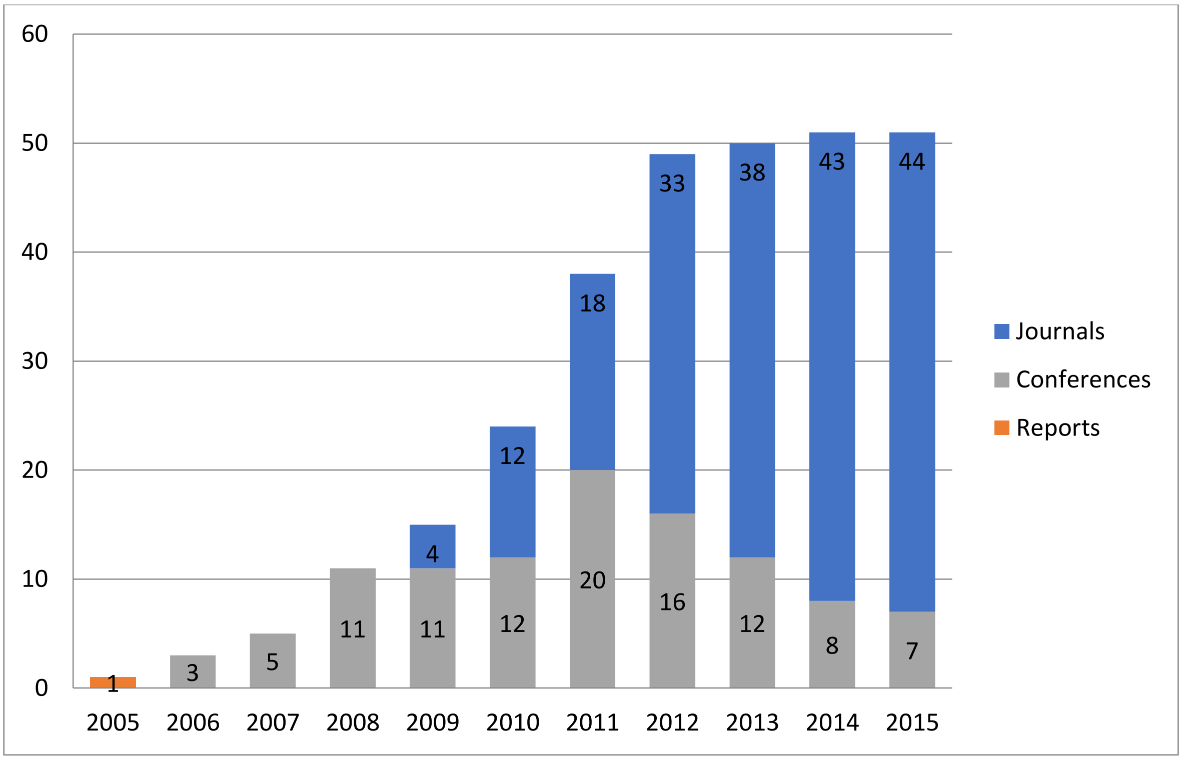

Figure 2.2: Number of papers based on the Bees Algorithm published per year

The Bees Algorithm was then extended to other domains such as combinatorial, multi-objective, and constrained optimisation problems. Typical in combinatorial problems, the Bees Algorithm together with new neighbourhood operators perturbed the solution in order to improve it. For example, Pham et al. (2007c, 2007e) introduced the 2-opt and insertion in scheduling, and sequencing problem respectively. Meanwhile, Ang et al. (2010) manipulated the TRIZ-inspired operator in printed circuit board assembly. In addition, Ozbakir et al. (2010) used the shift, double shift, and ejection chain in the Generalised Assignment Problem (GAP) while Dereli and Das (2011) utilised the 1-flip, swap, and *k*-flip to solve container loading problems. Jana et al. (2015), for their work involving protein structure, also attempted the use of some sort of mutation operator called the adaptive polynomial mutation for the scout re-initialisation after site abandonment. On top of that, most of the combinatorial problems solved using the Bees Algorithm employed the probability-selection instead of fitness-based such as research performed by Sadeghi et al. (2011) that implemented the selection of recruited bees based on the probability of the dancing rate for resource constrained project scheduling problem. The work by Bernardino et al. (2011) in load balancing problems in internet traffic did not just use the probability-based selection but also employed the deterministic Shortest Path algorithm in the initialisation phase. However, Abdullah and Alzaqebah (2013) made use of three different selection strategies; disruptive, tournament, and ranking selection besides hybridising the Bees Algorithm with SA and Hill Climbing. The application of choice for this hybrid algorithm was examination timetabling.

In multi-objective problems, solutions are found by locating the Pareto optimality such as in the works done by Anantasate et al. (2010), Anantasate and Bhasaputra (2011), and Sumpavakup et al. (2012) in the optimal power flow (OPF) problem; Ang et al. (2009) in motion planning of robotic arm, and Sayadi et al. (2009) in communication network design. Meanwhile, the algorithm has also been applied in the Environmental/Economic Dispatch (EED) problem by Pham et al. (2008d); Leeprechanon and Polratanasak (2010), as well as by Vennila and Prakash (2012). In the multi-objectives domain are studies seeking to solve various mechanical designs (Natchi et al, 2011) for the design of car suspension system (Kazemi et al., 2012), for optimisation in mechanical structures of composite laminates (Poor and Saz, 2012; Salamat and Ghanbarzadeh, 2012), in SCM (Mastrocinque et al., 2013), for the design of truss structure (Moradi et al. 2015), and in other electrical power applications (Lee and Kim, 2010; Rashtchi, 2010).

Some of them also incorporated FS and clustering method to reduce the Pareto front like studies performed by Bhasaputra et al. (2011) in solving OPF problem, and Phonrattanasak (2011) as well as Phonrattanasak et al. (2013) in the optimal sizing, and placement of wind farm. In the fuzzy logic methodology, Tolabi et al. (2013) transformed the multiple objectives functions into fuzzy membership functions while the improved analytical method simplified the candidate selection. Collectively, these strategies performed well in the distributed generators problem. However, not all multi-objective problems are solved through Pareto optimality. Lee and Darwish (2008) benefitted from the weighted sum, while Pham et al. (2010) and Marzi et al. (2010) exploited the parallel computing in EED problem.

Penalty function is the method normally used to handle constraints in optimisation problems. Examples of such researches are by Pham et al. (2009a), Pham et al. (2009b), Aydogdu and Akin (2011), Mirzakhani et al. (2011), Xu et al. (2011), and Alfi and Khosravi (2012) with most of the works involving the design problems in multiple disciplines.

One of the issues in the Bees Algorithm is the number of control parameters that need to be tuned by the user. Usually, this is done by conducting a small number of trials. However, Pham and Castellani (2009) proved that once a good setting has been found, the algorithm is able to consistently solve a large number of problems which eliminate the need to specifically set the control parameters for each different problem. Yet, Pham and Darwish (2008) employed the use of fuzzy inference system to reduce the number of parameters to be set. In this method, there is no need to set the number of selected sites, and the number of bees recruited to that sites. Instead, they are determined automatically by what they called the fuzzy greedy selection which disregards patches with low fitness or low rank. In addition, Otri (2011) used the Bees Algorithm to find the optimal value of the control parameter before using that value in the algorithm itself to solve optimisation problems in his PhD thesis. He called this method the Meta-Bees Algorithm. Meanwhile, to tune the Bees Algorithm parameters for the transportation management problem, Luangpaiboon (2011) tapped a mathematical programming optimisation method (weighted centroid modified simplex) together with linear constrained response surface whereas Mongkolkosol and Luangpaiboon (2011) employed the Steepest Descent. Other researchers also found that there is no need to differentiate between the elite sites, and the remaining best sites such as works done in the multi-objective Bees Algorithm (Ang et al., 2009; Pham et al., 2008d), and the design of antenna array by Guney and Onay (2011).

Beside the basic version of the Bees Algorithm, another variant includes two new procedures to further enhance the performance of the algorithm. They are the neighbourhood shrinking and the site abandonment (Pham et al., 2008c). In neighbourhood shrinking, the initial size of the neighbourhood is set at a very large value. For each site, the size is;

$$s_i(t) = ngh(t) \cdot \left(x_{max}^i - x_{min}^i\right) \tag{2.5}$$

$$ngh(0) = 1.0 \tag{2.6}$$

where t is the t^{th} iteration of the algorithm main loop.

The value is kept until no improvement is made on the fitness, then the size will be decreased to allow more detailed exploitation with the following formula:

$$ngh(t+1) = 0.8 \cdot ngh(t) \tag{2.7}$$

This procedure has a resonant of annealing temperature of SA. Furthermore, after applying the neighbourhood shrinking, and there is no improvement made after a certain duration, then the site is considered has converged to the local optima. The patch is abandoned and random scouting is re-initialised. With this procedure, a new control parameter is added which is the stagnation limit, *stlim*. Consequently, this modified version of the Bees Algorithm (from here onwards shall be called the Standard Bees Algorithm) has been applied successfully to the training of ANN (Pham et al., 2008c; Fahmy et al., 2011), numerical benchmark functions (Pham and Castellani, 2009; 2014; 2015), design of digital filter (Pham and Koç, 2010), service robot assignment (Xu et al., 2010 and 2011), software effort estimation (Azzeh, 2011), tuning of multi-objectives Proportional-Integral-Derivative (PID) controller (Ercin and Coban, 2011; Coban and Ercin, 2012), extraction of fuzzy measures for sample data (Wang et al., 2011), and optimal speed parameter of wind turbine generators (Fahmy, 2012). Loo et al. (2014) also converted it to be able to work on the parallel computing platform. Whereas Xie et al. (2015) applied the Bees Algorithm to the multi-user resource service composition in a cloud manufacturing.

Many attempts were made to improve the Bees Algorithm normally so that the algorithm depicts the bees' foraging nature more accurately. Muhammad et al. (2011) considered the possibilities that recruit bees are lost during flying by implementing the local search manoeuvres recruitment factor in the Bees Algorithm. This factor helps the algorithm to extend the neighbourhood size in a certain direction as well as mutate a part of the dimension of the search space. Accordingly, this enables the algorithm to reduce the number of iteration needed to achieve the optimum solution especially in high dimensional numerical problems.

Another variant of the Bees Algorithm introduced the concept of young bees to the most recent generated solutions. In order to protect this class of bees from the more evolved individuals, they were to compete only among themselves with the surviving individuals were given at least one evolution step to improve their solutions. In this version, the selected bees are not a fixed number set by the user but instead are a certain fraction of the population. Hence, there is no distinction between the best and elite bees as in the original form. There is also no value set for recruited bees aimed at the selected sites. In its place, operators such as mutation, creep, crossover, interpolation, and extrapolation are used to produce new individuals. Furthermore, a statistical-based method facilitated the tuning of the algorithm's parameters. With that, the algorithm has been tested to numerical benchmark test functions, and produced exceptional results when compared with EAs and PSO (Pham et al., 2012). Castellani et al. (2012) also applied it in dynamic optimal control in chemical engineering.

Hussein et al. (2013) incorporated the Lévy flight method to initialise the bees' position instead of using the uniform random distribution in the Bees Algorithm. Lévy distribution, as has been discussed in an earlier section, is a random heavy-tail distribution closer to the flying pattern of bees when foraging in their natural environment. Moreover, the size of the neighbourhood was determined by dividing the search space equally with the centre of the landscape as the hive location. This research was also applied in continuous numerical functions (Hussein et al., 2013 and 2014).

Subsequently, Shatnawi et al. (2013b) offered three versions of the memory based Bees Algorithm; local memory, global memory, and the combination of both. This approach compared the previous position stored in the memory with the new position of the best bee in the patch. If the previous position is better, recruited bees shall follow the memorised-position, otherwise the random search as in the normal Bees Algorithm is executed. A radius concept measured by Euclidean distance was also applied in this study so that different patches do not congregate on the same area.

Afterward, Yuce et al. (2013) presented the notion of adaptive neighbourhood search, and site abandonment into the Bees Algorithm. Through this strategy, the neighbourhood size can be shrunk and enlarged based on the fitness function throughout the search. After a certain limit, for each shrinking and enhancing purposes, unproductive sites will then be abandoned just like in the Standard Bees Algorithm. With this, two more new parameters are introduced; enhancing coefficient, and enhancing limit while another two are renamed as shrinking coefficient and shrinking limit. It was then tested against numerical benchmark functions with improved performance especially in 10D. Later, the upgraded algorithm was utilised in multi-objective SCM optimisation problem (Yuce et al., 2014).

A more recent study on the improvement of the Bees Algorithm is from Tsai (2014a) in which the neighbourhood size is made adaptive without using any input by the user. Instead it is based on the distance between two elite bees. This study verified its findings using the benchmark functions. It was also hybridised with ABC for constrained optimisation (Tsai, 2014b).

Another research that involved the adaptiveness of the patch size is by Ahmad et al. (2014). The size of the patch is augmented if improvements are made on the fitness value but is kept unchanged if there is none until a certain threshold. After this limit, much like the Standard Bees Algorithm, the size of the neighbourhood is shrunk. In addition, the neighbourhood size depends on the distance between the new best solution of the swarm, and the current best of each neighbourhood which make this magnitude bigger if the solution is far from each other. This study proved beneficial for mechanical design optimisation problems.

In addition, some of the Bees Algorithm enhancements were tailor-made to a specific application. For image classification purposes, Bradford Jr. and Hung (2012) modelled a pollen-based Bees Algorithm to include the environment interactions, and season changes to the foraging bees' behaviour. This version of clustering Bees Algorithm only used two parameters which are the pollen depletion rate, and the number of clusters that determine the value for other control parameters in the algorithm. Moreover, a bee in this variant is not the whole solution, but only a part of it, and the neighbourhood search was done in reverse where the size was set smaller initially then became bigger as it progressed. Later on, Leeprechanon and Phonrattanasak (2013) developed a two hives model of the Bees Algorithm where two populations of bees targeted different search spaces. They also utilised the Newton power flow to initialise the bees. Together this method was exploited in the OPF problem.

Another example is the study by Li et al. (2011) which engaged the use of controlled randomisation, and frequency memory based on polar coordinate in the initialisation phase of the algorithm to solve the circles packing problem. In the neighbourhood search, they divided the circles into three groups based on their radius. On the other hand, Bernardino et al. (2012) applied different local operators in the neighbourhood search of the Bees Algorithm that encompassed the exchange of one or two customers in the combinatorial problem of communication network design. Other researchers such as Guney and Onay (2013) also changed the way the neighbourhood search operates by using adaptive neighbourhood production mechanism in the linear antenna array application.

The trend in enhancing the capabilities of the Bees Algorithm usually involves the mechanism in the neighbourhood search as discussed in the previous paragraphs. One example of such an endeavour is the investigation by Ebrahimzadeh et al. (2013) where the neighbourhood size was not defined by the user but was based on the fitness value which means that each patch size differs with one another. It can be noted here that in the Standard Bees Algorithm the patch size will only vary once there is no yield whereas in here it changes throughout the search evolution. A new patch shrinking equation was also introduced based on the elite bees' fitness. Control chart pattern recognition was where this technique was justified. Next, it was also engaged in the classification of accuracy measurement of Global Positioning System (Azarbad et al., 2014).

The fittest bee of the swarm in the study from Masajed et al. (2013) influenced the scout bees' position after site abandonment. This is to ensure the bees do not fall on the unfruitful site, and are able to compete with the other bees much like the concept introduced by Pham et al. (2012). In this version, the algorithm was able to produce an optimised path plan for robot manipulator.

Whereas, Rambad and Rezaeian (2014) formulated probabilities of selecting and rejecting recruited bees based on the normalised fitness value. Two different parameter settings of the Bees Algorithm were then used in a machine scheduling application.

One of the latest undertakings of the Bees Algorithm is the work by Zhou et al. (2015) that manipulated the algorithm to enable the finding of multiple optimal solutions in MMO domain. By means of the radius estimation, variable colony size as well as the Hill-Valley mechanism, each patch converged to different peaks. The patch can also merge and split depending on their distance to each other. To speed up the local search, a balance technique guided the bees towards a better direction. Through this, the multiple optimums obtained can serve as an alternative especially if the fitness landscape has more than one global optimum.

Other variants of the algorithm include the hybrid versions with local search/ heuristics, or other metaheuristics. Most of these hybridised versions increase the performance of the Bees Algorithm in a specific application compared if just by using the basic form. Table A.1 in Appendix A lists the hybrid approaches used with the Bees Algorithm and their applications. Some even combined multiple methods such as Shafia et al. (2011) who used the Bees Algorithm together with GA and TS in the clustering problem. It was also merged with other metaheuristics as the local procedure to improve the algorithms such as in Gao et al. (2012, 2015) for the enhancement of the Harmony Search (HS), and Sagheer et al. (2012) to aid the Scatter Search.

Nonetheless, other approaches are not considered hybrid but rather inspired by some other optimisation technique. Pham et al. (2009b), motivated by PSO, updated the position of bees after the neighbourhood search towards the direction of the global best. Meanwhile, Packianather et al. (2009) eliminated the need to differentiate between best sites, and selected sites by using the Pheromone-based Bees Algorithm motivated by ACO. Long (2015) also developed his version of the Bees Algorithm with the update pheromone level equation for the design of hybrid vehicles. Besides using pheromone as the platform for information sharing between bees, Akpinar and Baykasoğlu (2014a) also manipulated multiple colonies of bees to explore different divisions of the search landscape. This variant was put into operation in the line balancing problem (Akpinar and Baykasoğlu, 2014a) as well as optimisation of numerical functions (Akpinar and Baykasoğlu, 2014b). Moreover, inspired by ABC, Alzaqebah et al. (2011) used probability-based selection in the application of examination timetabling.

One of the most popular usages of the Bees Algorithm from the literatures is in the field of machine learning, and data mining especially in pattern recognition purposes where the algorithm was applied to optimise various types of ANNs or a kernel-based method support vector machines (SVM). Multi-layer perceptron (MLP), a type of feedforward neural network has been used in wood defects identification (Pham et al., 2006b; Ghanbarzadeh, 2010), modelling of inverse kinematics for robot manipulators (Pham et al., 2008c; Fahmy et al., 2012), communication signal recognition (Sherme, 2011), breast cancer detection (Khosravi et al., 2011), fluid flow in porous media (Biglari et al., 2013), and recognition of communication modulation (Yang et al., 2015a, b). On the other hand, Pham and Darwish (2010) applied radial basis function network (RBF) in wood defects classification, Attaran et al. (2012) in machine fault diagnosis, Attaran and Ghanbarzadeh (2014) for fault detection in rotating machines, Ebrahimzadeh et al. (2014) for recognition of electrocardiogram signal, and Khajehzadeh (2015) for control chart pattern recognition. Conversely, Nebti and Boukerram (2010, 2012) utilised both MLP, and RBF in their effort to automatically detect Arabic numeral digits. Kalami (2014) also attempted the same strategy of using two different ANNs for fault recognition in electric power cable. Meanwhile, Pham et al. (2006c) used the learning vector quantisation network (LVQ) in control chart applications and Akkar (2010) used cellular neural network in opto-electronic circuit design.

In contrast, SVM was exploited in the breast cancer recognition problem (Addeh and Ebrahimzadeh, 2012), network anomaly detection (Alomari and Othman, 2012), image radar classification (Samadzadegan and Ferdosi, 2012), automatic modulation recognition (Sherme, 2012), and again in the wood defects identification (Pham et al., 2007d). Chen et al. (2014) to classify defects in welding process and implemented the combination of the Bees Algorithm and SVM.

ANNs alongside the Bees Algorithm are common for forecasting or prediction purposes. Khanmirzaei and Teshnehlab (2010) used recurrent neural network and the Bees Algorithm in weather forecasting while Şenyiğit et al. (2012) and Keskin et al. (2015) relied on the algorithm and MLP combo in lot size prediction, and the prediction of water pollution consecutively. Additionally, Azzeh (2011) estimated the software effort with the help of a regression based-model tree in conjunction with the Bees Algorithm, while Zarei et al. (2013) utilised a different combination involving the adaptive neuro-fuzzy inference system (ANFIS) which is another type of ANNs to predict heat from combustion of organic compounds. Using the same arrangement, they then abled to predict the removal rate of toxic material for treating waste water. In contrast, Ghaeni et al. (2015) employed a statistic-based response surface methodology with the Bees Algorithm for waste water treatment. Moreover, applying the linear regression model in cooperation with the Bees Algorithm helped Malekian et al. (2015) predict the energy consumption of a household.

In feature selection applications, Sadiq et al. (2012) engaged the Bees Algorithm together with the rough set theory for incomplete data problem. Rufai et al. (2014) that combined SVM with the Bees Algorithm for intrusion detection in cyber security, and Eesa et al. (2015) merged the algorithm with ID3; a decision tree learning algorithm for the same application. To boot, Packianather and Kapoor (2015) exploited a wrapper-based method for wood defect identification using the Bees Algorithm and the minimum distance classifier.

Another machine learning and data mining application that used the Bees Algorithm is clustering, where the algorithm optimises clustering algorithms such as fuzzy *c*-means (Pham et al., 2008a), and *k*-means (Pham et al., 2011; AbdelHamid et al., 2013). Saini and Kaur (2014) also incorporated *k*-means and Wards's clustering algorithm with the Bees Algorithm in the dataset of air pollution while Kataria and Rupal (2014) also employed the same combination for the typical benchmark clustering problems. Also using *k*-means but this time along with HS is the study from Bonab and Hashim (2014). Dhote et al. (2013) also applied a hybrid of the Bees Algorithm with PSO in benchmark clustering problems. Whereas Ananthara et al. (2013) modelled the Bees Algorithm as a clustering method for agricultural datasets, and Anaraki and Sadeghi (2015) for diseases data. In addition, Nebti (2013) exploited the Bees Algorithm for unsupervised image classification.

Figure 2.3 represents the percentage of researches using the Bees Algorithm according to the area of application. Based on the figure, applications in computer science and engineering constitute almost a quarter of the researches followed by electrical and electronics engineering. Other major areas include mathematics, mechanical engineering, and industrial engineering. A list of applications of the Bees Algorithm found through search engines IEEExplore, ScienceDirect, SpringerLink, and Google Scholar can be found in Appendix B. However, this list is not exhaustive as it does not include papers written in authors' native languages as well as papers that are behind pay-walls.

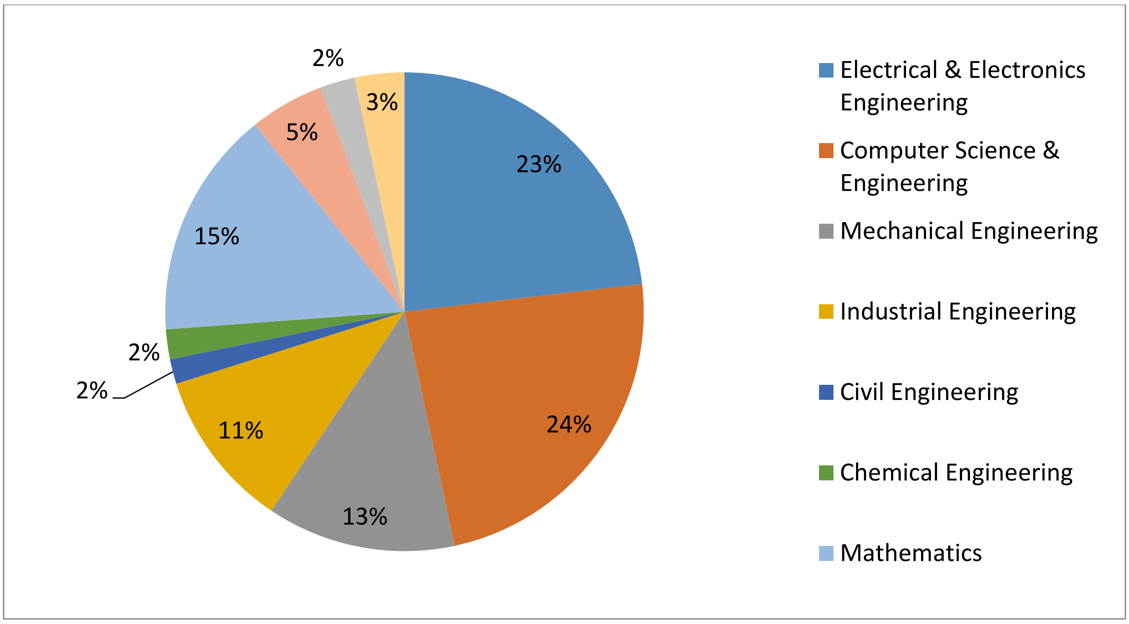

Figure 2.3 Percentages of applications using the Bees Algorithm per specialised area

2.6 Conclusions

This chapter has reviewed the area of intelligent optimisation and has focussed on techniques utilising metaheuristics inspired by the behaviours of swarms of animals. In particular, the chapter has examined the Bees Algorithm, the operation of which mimics the foraging behaviour of honey bees. Different variants of the algorithms are described together with their applications in various branches of engineering and science. From the review, there are still much can be done in improving the performance of the Bees Algorithm especially in term of its convergence speed. Thus, in the subsequent chapters this thesis introduces several improvements made to the algorithm in order to achieve faster convergence.

CHAPTER 3

A PSEUDO-GRADIENT BEES ALGORITHM (PG-BA)

3.1 Preliminaries

Most population-based schemes suffer from long computational time due to their stochastic nature as discussed in Chapter 2. Search directions are randomly selected which slow down the convergence to the global optimum especially in difficult problems. Literature shows that the Bees Algorithm also experiences similar problem (Khanmirzaei and Teshnehlab, 2010; Alfi and Khosravi, 2012; Pham and Darwish, 2010; Anantasate et al., 2010; Yuce et al., 2015). The convergence speed of the algorithm gradually decreases as it moves closer to the global optimal solution which is due to the random mutation operation in the neighbourhood search. A survey performed in Chapter 2 shows that the algorithm has been hybridised with many gradient-based techniques to rectify the problem. Using deterministic procedure, gradient-based optimisers are able to reach the optimal solution quickly with high accuracy by following the gradient direction obtained through differentiation of objectives function.

Nevertheless, without prior knowledge, derivative information in many real world problems is hard to get. Moreover, hybridisation with these gradient-based algorithms will also bring along additional parameters that need to be set.

Several researchers have entertained the idea of a gradient-like strategy to acquire gradient approximation, especially to work with Evolutionary Algorithms (EAs) (Pham and Jin, 1995; Solomon, 1998; Abbas et al., 2003; Lin et al., 2006; Hewlett et al., 2007). A quasi-gradient mechanism has also been employed in Artificial Immune Systems (AIS) (Zhang and Yen, 2013). Hence, this chapter investigates the use of gradient-like Bees Algorithm in improving the convergence speed when tested against selected numerical functions compared to the stand-alone version of the algorithm.

3.2 Pseudo-gradient Bees Algorithm (PG-BA)

During the waggle dance performed by returning bees, three types of information are conveyed: 1) quality, 2) distance and 3) direction. The recruiter bees know the direction of the site (deterministic) but their movement i.e., size of the step calculated is still a bit random (Soós, 2013). In nature, when bees find flowers packed with nectar, they will fly a short distance to the next flower in an attempt to find a better quality site but the direction of the flight is always maintained to avoid revisiting a depleted site. If the quality is poor, its trajectory is extended by increasing its rotation angle to move away from that area. However, this response is not immediate because it relates to the number of unrewarding visits made due to the variety of floral distribution. However, since rapid response is needed in this research, this fact will be neglected. Nonetheless, it is found that this is an effective strategy in a high density flora with clumped nectar distribution as studied by Banks et al. (2009).

On the other hand, the Standard Bees Algorithm can be grouped as a guided search. This is because the follower bees are positioned around the best bee, but the search direction is random. Randomness helps a system find new solutions especially in complex problems. However, this has also made the convergence process slower when it is nearing the optimal solution. Therefore, by employing the pseudo-gradient method in the neighbourhood search, it can also guide the follower bees towards a better direction in discovering new potential solution.

3.2.1 Pseudo-gradient method

The inspiration behind the pseudo-gradient method introduced by Wen et al. (2003) was based on the concept of use and disuse in the evolutionary field. This idea reflects a human organ in which the more it is used the stronger it will become; whereas organs that are not frequently used tend to be weaker. Contrary to other approaches, this technique compares fitness and its equivalent position of prior and current points in the solution space. If there is an improvement in the fitness, the pseudo-gradient is not equal to zero. Thus, a better fitness can be found in the next generation by following this pseudo-gradient direction. If there is no gain in fitness, the next search direction is according to random distribution. This procedure can be applied to any population-based algorithms to suit problem even with higher dimensions owing to its simple mechanism. In fact, it has been hybridised together with EP to solve a mixed optimisation problem where the objective functions are both continuous and discrete (Wen et al., 2004). Li et al. (2010) undertook a similar approach in Paired Bacteria Optimisation (PBO) to solve the optimal power flow problem. Additionally, a variant of PSO has been merged with the pseudo-gradient procedure for an application in economic dispatch with valve point loading effect (Dieu et al., 2011).

In contrast to the original method of Wen et al. (2003) where the pseudo-gradient is applied to the entire population, this research only applies this mechanism in each patch (subpopulation) that is bounded by the neighbourhood size. This means that the new points cannot lie outside this patch. Assume that the position of the scout bee is $x_k = x_{k1}, x_{k2}, \dots, x_{kn}$ and the forager's position is x_l, then pseudo-gradient is;

$$g_p(x_l) = [dir(x_{l1}), dir(x_{l2}), \dots, dir(x_{ln})]^T \tag{3.1}$$

i. if $-f(x_l) < -f(x_k)$, x_k is moving in the right direction where $g_p(x_l) \neq 0$ then,

$$dir(x_l) = \begin{cases} 1, if\ x_l > x_k \\ 0, if x_l = x_k \\ -1, if\ x_l < x_k \end{cases} \tag{3.2}$$

ii. if $-f(x_l) \geq -f(x_k)$, x_k is moving in the wrong direction then,

$$g_p(x_l) = 0 \tag{3.3}$$

Similar to the gradient-based method, this approach can provide strong evidence of search direction based on the latest two points in the search space. From the equations, it indicates that if pseudo-gradient is not equal to 0, then a better solution will be found in the next. Otherwise, the direction should be changed. Thus, the neighbourhood search formula in Equation 2.4 is updated as follows:

$$x_l = \begin{cases} (x_k - ngh) + g_p(x_l)\left|2 \cdot ngh \cdot rand_i \cdot \left(x^i_{max} - x^i_{min}\right)\right|, g_p(x_l) \neq 0 \\ (x_k - ngh) + 2 \cdot ngh \cdot rand_i \cdot (x^i_{max} - x^i_{min}), otherwise \end{cases} \tag{3.4}$$

It means that the placement of the forager bees depends on the scout bee of the past generation and the current. This can also be seen as cooperation between bees (share information), and the bees' memory (knowledge from previous experience).

Since the new position guided by the pseudo-gradient is limited by the boundaries of each patch (neighbourhood), a few other strategies are tested to determine the impact of this element. Figure 3.1 illustrates the position of the scout and followers inside the neighbourhood for each case. The square is the scout bee while circles are follower bees. Meanwhile, Figure 3.2 shows an example of the position of scout and the neighbourhood in 1 dimension problem. Blue square denotes PG-BA1 neighbourhood, red square denotes PG-BA2 neighbourhood, and green square denotes PG-BA3 and PG-BA4 neighbourhood. Each variant is described as follows:

i. PG-BA1

   This is the simplest form of PG-BA where the scout bee's position in the patch is the same as the Standard Bees Algorithm, which is at the centre of the neighbourhood. This give a fair chance to the follower bees to discover a new area initially as the size of the patch is set to a large value before shrinking if no improvement is found. Recruiters follow the direction of the pseudo-gradient set by the scout bees.

ii. PG-BA2

   As the neighbourhood size is the same for all functions, one way to intensify the local search is by shifting the position of the scout bees in the neighbourhood. In PG-BA2, the neighbourhood is totally shifted (100%) towards the direction of the pseudo-gradient thus making the position of the scout bee at the lower limit of the patch.

iii. PG-BA3

PG-BA2 can be regarded as a highly exploitative version of PG-BA. Thus, PG-BA3 tries to minimise the effect if any by slightly shifting (50%) the patch towards the direction of the pseudo-gradient. This means that the location of the scout bee is halfway between the centre of the patch and the lower limit.

iv. PG-BA4

The position of the scout bee in this version is the same as PG-BA3. However, in order to increase the chance of finding new promising location at the beginning of the optimisation process, only 90% of foragers inside each patch are placed in the direction of the gradient while the other 10% are at random.

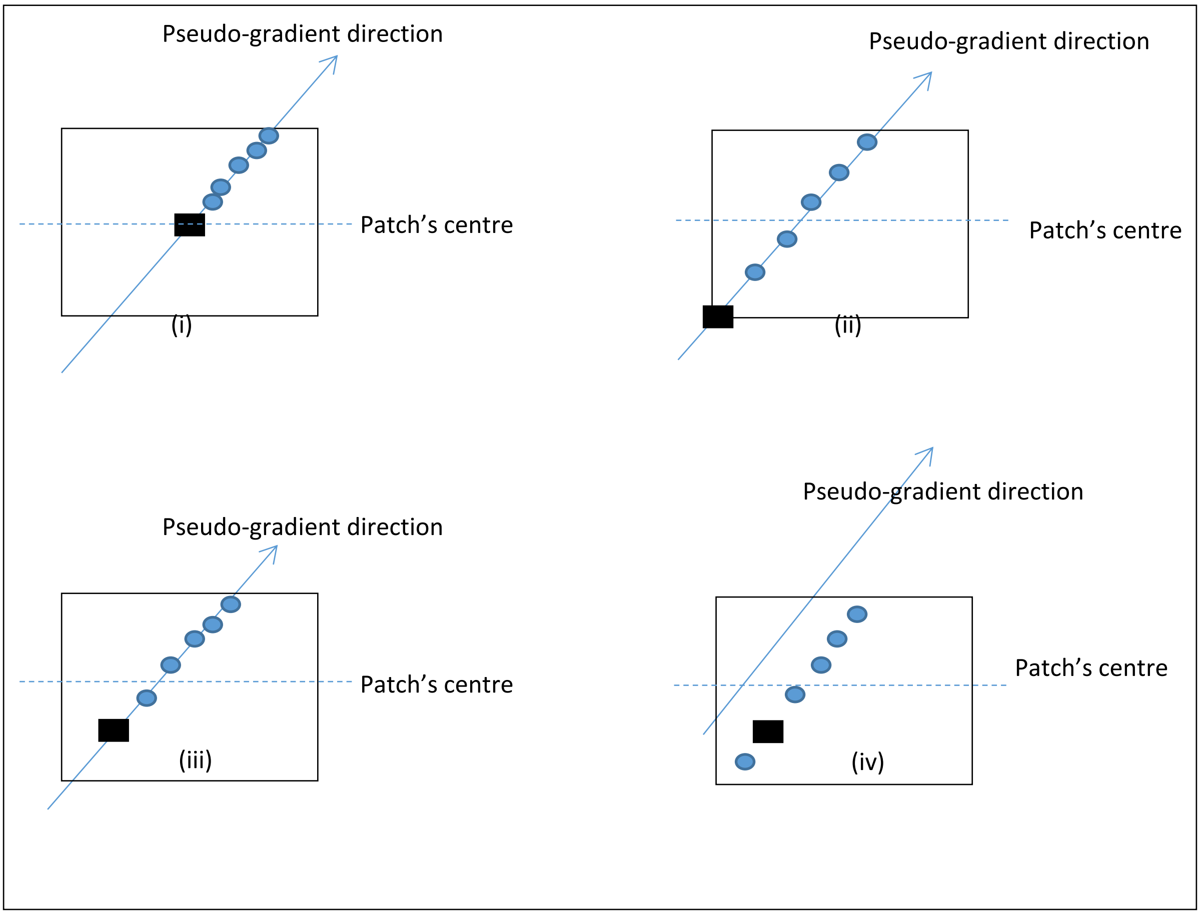

Figure 3.1 Position of bees in neighbourhood; i) PG-BA1, ii) PG-BA2, iii) PG-BA3, and iv) PG-BA4

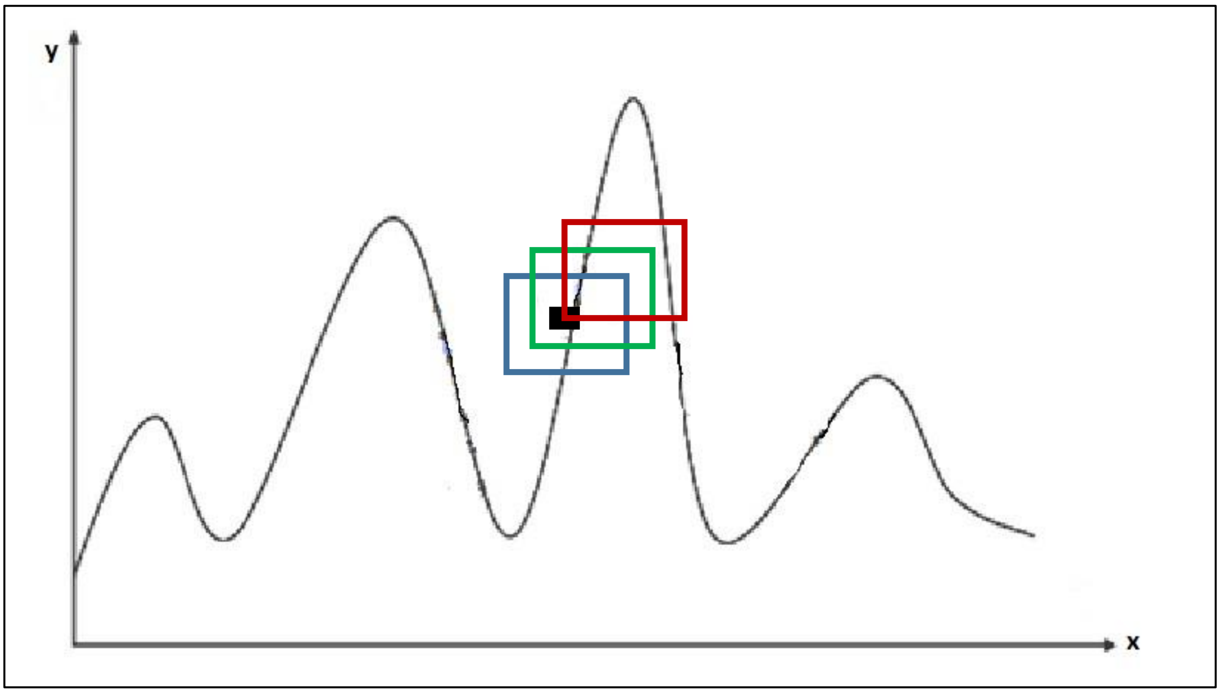

Figure 3.2: Example of PG-BA variants neighbourhood in 1D problem

3.3 Experimental Set-up

In manufacturing, optimisation problems can be modelled into numerical functions such as in manufacturing process where the optimiser is used to find the suitable values of global features such as cutting/milling speed, and feed rate. Another example is in finite element analysis for the product design process where the decision variable is the product's length or size, and the objectives are maximising the loads in relation to stress and strain or/and minimising consumption of material. These involve certain characteristics such as large solution boundary range, wide or narrow feasible solution regions, variables with high dimensionality, and multimodal exterior (Tao et al., 2015). Thus, benchmark test functions of varying topological search space may represent some of the key aspects of real world problems that are useful in the evaluation of optimisation techniques. The performance is then validated by comparing with other methods using the same test problems. Through this, the strengths and weaknesses of an algorithm can be identified so that further improvements can be made.

Fifteen continuous functions were selected to be implemented in this chapter. The full equation for each function can be found in Table C.1 from Appendix C as compiled by Jamil and Yang (2013). These test functions are chosen because they display different topological search surface. Some are unimodal which mean there is only one global optimum, and some are multimodal where there exists multiple global and local optima. They also come with different complexity in term of their regularity separability, and dimensionality. A regular function is differentiable (analytical) at each point of its domain (Simon, 2008). Separability refers to the relationship between variables in the function. Separable functions are easier to solve than non-separable functions because their variables are independent of each other (Mesa et al., 2011; Jamil and Yang, 2013). Meanwhile, dimensionality reflects the number of parameters to be optimised. Normally, functions with high dimensions are more difficult to solve than functions with low dimensions (Ali et al., 2005; Dietrich and Hartke, 2012).

Sphere function also known as De Jong's function 1 is a harmonic, convex, nonlinear, and symmetrical function. This unimodal function has smooth objective function and usually used to demonstrate general efficiency of any algorithm. As cited by Imanguliyev (2013), this function can be used to represent cost curve in engineering economy. Meanwhile, *Axis Parallel Hyper-ellipsoid* is an extension of *Sphere* function with weighted model. Just like its predecessor, it has a convex and unimodal surface. On the other hand, *Moved Axis Parallel Hyper-ellipsoid* is more elliptic than *Axis Parallel Hyper-ellipsoid* but it still has the same general convexity and singular modality. Another derivation from *Axis Parallel Hyper-ellipsoid* is *Schwefel 1.2* or sometimes known as Rotated Hyper-ellipsoid or Sum Squares or Double Sum. It is also a convex and unimodal function. These extended functions are good to investigate whether certain algorithms have any biases towards for example global optimum lying in the centre of the search range or local optima lying along the coordinate axes (Liang et al., 2005).

Schwefel 2.21 is also a unimodal function similarly called MaxMod. Other classic example of unimodal function commonly used to test optimiser's capability is *Sum of Different Power*. *Trid* or otherwise named Neumaier's function 3 is convex, quadratic, and unimodal. The boundary constraint of this function can be scaled to dimension which is important in testing the efficiency of an algorithm to locate the optimum once the basin of attraction has been identified. Due to its strong separability, this function can be difficult to some algorithm. Another unimodal function is *Powell*. It is convex and nonlinear. The region where the global optimum lies is smaller than the rest of the search landscape which can also prove hard to solve to some optimisation algorithm. In contrast, even though *Quartic* or Modified 4th De Jong function has an overall unimodal surface but it is padded with Gaussian noise. Thus, no similar point can have the same value. If an algorithm fails to tackle this function, it will not perform well in real world problems with noisy data.

Conversely, *Ackley* is a multimodal function that has multiple local minima covering its solution space due to the use of exponential term (Akay and Karaboga, 2012). According to Imanguliyev (2013), this function can be used to characterise the material surface of atomic and molecular level. *Griewank* also is a multimodal function with the location of the numerous optimum regularly distributed across the search domain. *Six Hump Camel Back* has two global optima and four local optima. *Shekel* has *O* optimum, with *O* can be 5, 7, or 10. Another example of multimodal function is *Schwefel 2.22*. In the meantime, *Alpine* is an asymmetrical multimodal function. Algorithms that do not have a good balance between exploration and exploitation can easily become trapped in these multimodal functions.

Table 3.1 summarises the characteristics of the above mentioned functions. It can be seen that slightly more unimodal functions are used. Previous researchs have shown that the Bees Algorithm is good in handling multimodal problems but quite slow to converge in unimodal such as the one found by Pham et al (2009). Thus, this thesis investigates whether the proposed algorithm displays similar trait. To the author's knowledge, besides *Sphere*, *Ackley* and *Griewank*, the rest of the functions have never even been tested with any variant of the Bees Algorithm. In fact, in this research some of the functions were tested to up to 50 dimensions.

Success rate was used to demonstrate efficiency and trustworthiness of the proposed method as one of the performance indicators. In addition, as one the objective of this research is to speed up the convergence of the Bees Algorithm, another performance measure is by using time complexity. For that reason, number of function evaluations (NFE) was used instead of central processing unit (CPU) time or number of generation. This will provide a fair comparison if there are differences in terms of the language code, compiler, and computer's processor. The function evaluations in this research also took into account the additional evaluations performed every time a scout is re-initialised as consequences of site abandonment procedure.

Table 3.1: Summary characteristics of test functions used

Functions	Differentiability	Separability	Scalability	Modality
f1, Six Hump Camel Back	Yes	Yes	No	Multimodal
f2, Shekel	Yes	Yes	No	Multimodal
f3, Trid	Yes	Yes	No	Unimodal
f4, Moved Axis Parallel Hyper-ellipsoid	Yes	Yes	Yes	Unimodal
f5, Schwefel 1.2	Yes	Yes	Yes	Unimodal
f6, Powell	Yes	Yes	Yes	Unimodal
f7, Sum of Different Power	Yes	No	Yes	Unimodal
f8, Sphere	Yes	No	Yes	Unimodal
f9, Griewank	Yes	Yes	Yes	Multimodal
f10, Axis Parallel Hyper-ellipsoid	Yes	Yes	Yes	Unimodal
f11, Ackley	Yes	Yes	Yes	Multimodal
f12, Schwefel 2.21	No	Yes	Yes	Unimodal
f13, Schwefel 2.22	No	Yes	Yes	Multimodal
f14, Quartic	Yes	No	Yes	Unimodal with noise
f15, Alpine	No	No	Yes	Multimodal

One of the main elements of this thesis is to investigate whether the same parameter setting of the Bees Algorithm can be used for all test functions to achieve acceptable results within the required tolerance without careful tuning. A rule of thumb can serve as a guide in selecting a feasible value for the parameters of this algorithm.

i. The number of selected bees must be less than or equal to the number of initial scout bees but more than 0
ii. The number of recruiters for elite sites must be more than the number of recruiters for the other selected sites

iii. The number of elite sites can be less or equal to the number of the other selected sites

iv. The neighbourhood size can be set at a large value in the beginning because by using the neighbourhood shrinking, this value will get smaller to adapt to the search so that more exploitation can be made

v. For the stagnation limit, it should not be too high so as to increase the function evaluations and not too low to allow the algorithm some time to achieve a better result. From review done by Pham and Castellani (2014), the typical value is between 5 and 10.

All the methods in this research shall use the same control parameters depicted in Table 3.2. Parameters selected in this research can be described as a balance between depth and breadth search. In the first category, the length of the evolution period is seen as the key to the success of the optimisation process. In the latter, the main search property is the population size that support explorative strategy especially in handling multimodal functions (Pham and Castellani, 2013). The parameters were selected so that the bees' population size is 100. As none of the algorithm's control parameter is finely tuned to each problem, the default size of neighbourhood is the search range of each function divided by two. The values of the parameters are also chosen by considering the typical values used by other researchers using the Bees Algorithm.

As for the stopping criterion, the optimisation process stopped when it has achieved a reasonable sensitivity value of 0.0001 to the optimal solution or the maximum NFE has reached 1,000,000 for all test functions except for *Alpine* and *Quartic*. These functions are considered problematic by many optimisation algorithms, thus the precision to the optimum for both functions was set at 0.1 which provides for a feasible solution to be obtained within the specified maximum NFE. Furthermore, to take into account the randomness of the algorithm, all methods were executed and averaged for 100 independent runs.

Table 3.2: Parameter setting for all Bees Algorithm's variants

Parameters	**Value**
Number of elite sites	2
Number of recruiters on elite site	29
Number of best sites	6
Number of recruiters on remaining best sites	9
Number of random scouts	6
Stagnation limit	10
Neighbourhood size	(Search range)/2

3.4 Results and Discussion

Table 3.3 shows the success rate achieved by each algorithm. Only PG-BA1 obtained a 100% success in all functions with PG-BA2 having the least number of success rate. The rest of methods including the standard BA managed to accomplish 100% success rate in the same seven functions out of 15. It appears that PG-BA1 can solve different types of problem landscape, whereas PG-BA2 can only solve unimodal functions and multimodal functions in low dimensions. On the other hand, PG-BA3, PG-BA4 and the Standard Bees Algorithm are adept at fairly all unimodal as well as simple multimodal in high dimensions (i.e. *Griewank*).

Table 3.3 Success rate over 100 runs for PG-BA experiment

Function	Standard BA (%)	PG-BA1 (%)	PG-BA2 (%)	PG-BA3 (%)	PG-BA4 (%)
f1	**100**	**100**	**100**	**100**	**100**
f2	97	**100**	96	97	97
f3	**100**	**100**	100	100	100
f4	100	**100**	100	**100**	**100**
f5	**100**	**100**	0	**100**	**100**
f6	1	**100**	0	1	0
f7	**100**	**100**	**100**	**100**	**100**
f8	**100**	**100**	**100**	**100**	**100**
f9	**100**	**100**	0	**100**	**100**
f10	**100**	**100**	0	**100**	**100**
f11	31	**100**	0	0	0
f12	0	**100**	0	0	0
f13	0	**100**	0	0	0
f14	1	**100**	0	0	0
f15	0	**100**	0	0	0

Table 3.4 displays the means and standard deviation of NFE generated by each algorithm after 100 runs. The bold values show the best performance for each function. Meanwhile, Table 3.5 depicts the percentage of improvements in term of reducing NFE when comparing with the Standard Bees Algorithm. The null value is due to no yield being realised. Both tables show that the better performance in term of speed was achieved by PG-BA1 and PG-BA3. Even though PG-BA2 and PG-BA4 achieved 100% success rate in some functions, their NFE was not lower than the Standard Bees Algorithm. PG-BA3 was capable to reduce NFE in three out of 15 functions. The functions were *Six Hump Camel Back* and *Trid* that are of low dimensions, as well as *Schwefel 1.2*. In the meantime, PG-BA1 excelled in the rest of the test functions. Based on these results, PG-BA1 is the most efficient procedures in the overall benchmark suite tested. If NFE of all the functions were totalled, PG-BA1 is capable to minimise the total NFE to 63.2% compared to the Standard Bees Algorithm.

Table 3.4 Mean and standard deviation of function evaluations over 100 runs for PG-BA experiment

Func.	Standard BA		PG-BA1		PG-BA2		PG-BA3		PG-BA4	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
f1	7,969.6	3,741.5	6,700.4	2,956.7	2,753.4	438.5	**2,693.6**	**426.5**	2,736.6	380.1
f2	36,720.9	2,925.2	**5,389.6**	**1,529.0**	45,888.5	194,909.7	35,597.2	169,686.9	35,628.4	169,686.4
f3	6,527.0	348.5	6,431.0	332.8	7,637.1	8,106.0	**6,347.0**	**395.8**	6,398.0	348.1
f4	8,557.0	463.3	**7,242.0**	**325.1**	10,822.0	3,164.3	8,442.0	529.1	9,217.0	1,005.6
f5	48,364.2	3,547.9	41,069.8	4,012.2	1,000,000.0	0.0	**35,887.0**	**2,832.7**	110,411.0	17,137.9
f6	998,884.0	11,223.5	**48,042.5**	**76,482.9**	1,000,000.0	0.0	989,821.0	30,900.9	1,000,000.0	0.0
f7	3,286.3	454.8	**1,982.7**	**228.3**	3,772.0	721.2	3,413.0	437.6	2,967.0	544.1
f8	14,460.4	628.2	**7,419.3**	**272.6**	58,488.0	18,952.0	15,871.0	670.8	20,515.0	1,456.1
f9	28,085.7	11,878.6	**11,252.5**	**365.2**	1,000,000.0	0.0	38,006.5	24,775.9	115,320.7	105,426.4
f10	43,797.6	9,749.4	**9,316.5**	**404.6**	1,000,000.0	0.0	50,982.0	13,387.6	347,848.0	163,998.6
f11	822,726.9	326,531.3	**13,778.2**	**351.6**	1,000,000.0	0.0	1,000,000.0	0.0	1,000,000.0	0.0
f12	1,000,000.0	0.0	**14,197.5**	**474.7**	1,000,000.0	0.0	1,000,000.0	0.0	1,000,000.0	0.0
f13	1,000,000.0	0.0	**16,031.1**	**5,523.4**	1,000,000.0	0.0	1,000,000.0	0.0	1,000,000.0	0.0
f14	292,293.1	452,737.3	**2,106.0**	**259.5**	1,000,000.0	0.0	1,000,000.0	0.0	1,000,000.0	0.0
f15	1,000,000.0	0.0	**22,933.2**	**42,985.1**	1,000,000.0	0.0	1,000,000.0	0.0	1,000,000.0	0.0
Total Eval.	5,311,672.7		**213,892.4**		9,129,361.0		6,187,060.2		6,651,041.7	

Table 3.5: Percentage of improvements of PG-BA compared to the Standard Bees Algorithm

Function	PG-BA1 (%)	PG-BA2 (%)	PG-BA3 (%)	PG-BA4 (%)
f1	15.9	65.5	**66.2**	65.7
f2	**85.3**	-	3.1	3.0
f3	1.5	-	**2.8**	2.0
f4	**15.4**	-	1.3	-
f5	15.1	-	**25.8**	-
f6	**95.2**	-	0.9	-
f7	**39.7**	-	-	9.7
f8	**48.7**	-	-	-
f9	**59.9**	-	-	-
f10	**78.7**	-	-	-
f11	**98.3**	-	-	-
f12	**98.6**	-	-	-
f13	**98.4**	-	-	-
f14	**99.3**	-	-	-
f15	**97.7**	-	-	-

A paired *t*-test was also performed to determine whether the difference between the results of two algorithms does not happen by chance. Table 3.6 shows the significant different between the variants of PG-BA and Standard Bees Algorithm with a confidence level of 95% ($\alpha = 0.05$). The *p*-values less than the significance level signal that the superior results accomplished by the top algorithm in each case are statistically significant and non-random. These findings demonstrate that the entire enhancement attained by PG-BA1 is statistically significant compared with only a few in the other variants. Despite the fact that PG-BA1 did not achieve the highest percentage of improvement in *Six Hump Camel Back* and *Trid*, the *p*-value of 0.0084 and 0.0477 correspondingly were still below the acceptance value of 0.05 that render these results significant. Even though PG-BA3 and PG-BA4 were able to slightly reduce the convergence speed of *Shekel 10* function by 3.1% and 3.0% respectively, the *p*-value showed that the result is not significant. Similar trait can be observed in function *Moved Axis Parallel*

Hyper-ellipsoid with PG-BA3. The minor reduction of 1.3% accomplished was also considered insignificant. This further supports that PG-BA1 is the preferred strategy in speeding up the optimisation process for all the benchmarks problems used.

Table 3.6: *p*-values using t-test (α = *0.05*) comparing PG-BA with the Standard Bees Algorithm

Function	PG-BA1	PG-BA2	PG-BA3	PG-BA4
f1	0.0084	**0.0001**	**0.0001**	**0.0001**
f2	**0.0001**	-	0.9631	0.9641
f3	0.0477	-	**0.0008**	0.0095
f4	**0.0001**	-	0.1036	-
f5	**0.0001**	-	**0.0001**	-
f6	**0.0001**	-	0.0064	-
f7	**0.0001**	-	-	**0.0001**
f8	**0.0001**	-	-	-
f9	**0.0001**	-	-	-
f10	**0.0001**	-	-	-
f11	**0.0001**	-	-	-
f12	**0.0001**	-	-	-
f13	**0.0001**	-	-	-
f14	**0.0001**	-	-	-
f15	**0.0001**	-	-	-

Based on the findings, all PG-BA variants are able to outperform the Standard Bees Algorithm in the *Six Hump Camel Back* function. Even though this is a multimodal function, the low dimensionality could help the new variants to attain excellent results. In fact, the Bees Algorithm is initially well-known in solving multimodal functions (Pham and Castellani, 2009). On the other hand, all of the PG-BA variants except PG-BA2 had only minor improvements in the *Trid* function if compared with other functions. Jamil and Yang (2013) observed that this function has many orders of magnitude different between domain and the function hypersurface which could prove difficult to handle by these variants. For the rest of the functions, the other variants except for PG-BA1 performed poorly on highly multimodal functions in high dimensions. This could be due to the location of the scout bee which is not at the centre of the neighbourhood i.e., shifting to the direction of the pseudo-gradient. This implies that fully investing at the best so far solution can make the algorithm over exploitative. Reducing the number of foragers and shorter stagnation limit can help in reducing the convergence speed of these algorithms by allowing the site to become abandon quickly. In contrast, directed search with some randomness in PG-BA1 using the scout bee at the centre of the patch is a better strategy that balances between exploitation, and exploration phase of the Bees Algorithm. This eventually makes the algorithm faster. Based on the different variants used in the experimentation, the neighbourhood size is the most likely parameter that can affect the optimisation result. A proper tuning of this parameter could result in a better performance. Crossley et al. (2013) confirmed these findings. Nonetheless, PG-BA1 proves that by using a single parameter set for all functions, it is able to achieve successful results.

Additionally, investigating the ability of PG-BA1 to scaling was conducted to ascertain the reaction of the algorithm towards functions with scalable dimension. PG-BA1 was selected for extra study because it is the clear winner in this research. Figure 3.3 exhibits that in general PG-BA1 has almost similar performance across the dimension. As a comparison, scalability test is also performed on the Standard Bees Algorithm with the result shown in Figure 3.4. In PG-BA1, the effect of higher dimensionality only slightly increases NFE. However, the Standard Bees Algorithm displays tremendous different in function evaluation in most of the scaling functions. This shows that PG-BA1 has the potential to be applied in large scale problem maybe even up to 1000D. This is with the exception of function Schwefel 1.2 and Schwefel 2.22 where

NFE rise tremendously as the dimension increases. Meanwhile, *Griewank* shows a unique behaviour that uses a high NFE at a lower dimension but decrease and stabilise and higher dimensions.

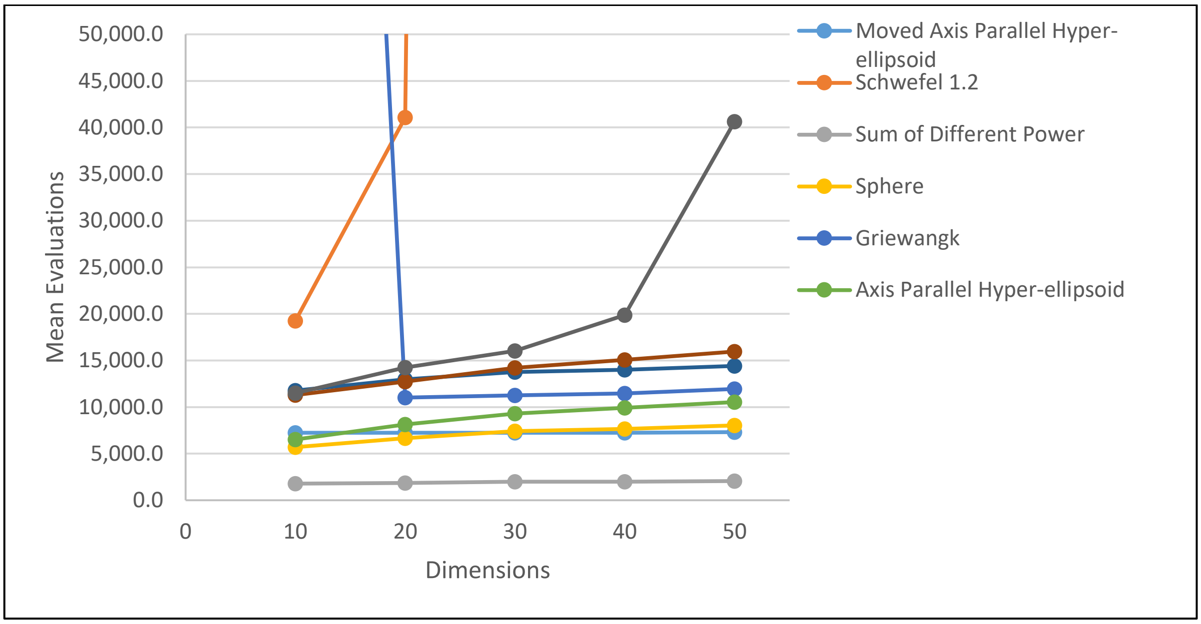

Figure 3.3: Scalability test of PG-BA1

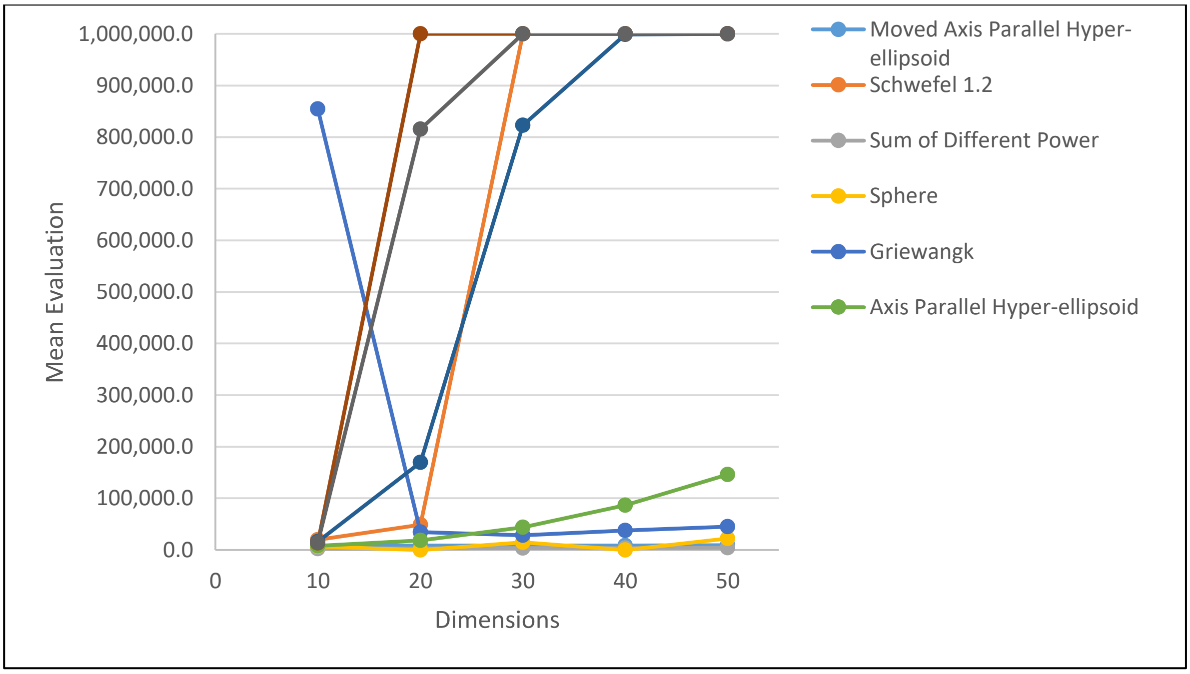

Figure 3.4: Scalability test of the Standard Bees Algorithm

3.4.1 Comparison of PG-BA with other swarm optimisers

To further validate the efficiency of PG-BA in achieving faster convergence rate, it is compared with other well-known swarm optimisers. The chosen algorithms are the Quick Artificial Bee Colony (qABC) and the Stardard Particle Swarm Optimisation 2011 (SPSO2011). qABC was chosen because it is also based on the food foraging behaviour of bees and one of the latest variant of ABC co-developed by the original inventor (Karaboga and Gorkemli, 2012 and 2014). PSO is one of the earliest and most popular algorithm based on the swarm principle. SPSO2011 is the state-of-the-art standard variant of PSO (Clerc, 2012, Zambrano-Bigiarini et al., 2013). The operation for both algorithms have been summarised in Chapter 2. Table 3.7 lists the parameter setting for both algorithms as suggested by their authors and are matched as close as possible with PG-BA parameters in term of population size for fair assessment.

Table 3.7: qABC and SPSO2011 parameter setting

qABC		SPSO2011
Colony size, CS = 100	Onlooker bees = 50, Employed bees = 50	Swarm size, S = 100
Neighbourhood radius, r = 3		Cognitive coefficient, $c_1 = 0.5 + ln(2)$
		Social coefficient, $c_2 = 0.5 + ln(2)$
Limit for abandonment, l = 100		Inertia weight, $w = 1/(2*ln(2))$
		Number of informants, K = 3
		Velocity clamping = $[-\vec{X}_{max}, \vec{X}_{max}]$

For this experiment, other settings such as stopping criterion and accuracy to achieve follow the previous values used to compare PG-BA variants with the Standard Bees Algorithm. Only PG-BA1 is used as comparison as it is the most efficient version of PG-BA. From this point onwards, PG-BA1 shall be called just as PG-BA.

Table 3.8 presents the mean and standard deviation of function evaluations for PG-BA, qABC, and SPSO2011. The *p*-value comparing PG-BA with either qABC or SPSO2011 for each function is also shown. The confidence interval used is the same as previous experiment, which is 95%.

Table 3.8: Performance comparison between PG-BA with qABC and SPSO2011

Functions	PG-BA		qABC			SPSO2011		
	Mean	**Std. Dev.**	**Mean**	**Std. Dev.**	***p*-value**	**Mean**	**Std. Dev.**	***p*-value**
f1	6,700.4	2,956.7	**704.1**	**885.8**	**0.0000**	3,397.0	918.1	**0.0000**
f2	**5,389.6**	**1,529.0**	45,888.5	194,909.7	**0.0390**	27,016.0	139,013.1	0.1214
f3	**6,431.0**	**332.8**	529,982.7	142,468.2	**0.0000**	9,609.0	504.0	**0.0000**
f4	**7,242.0**	**325.1**	32,673.9	10,321.5	**0.0000**	11,822.0	509.2	**0.0000**
f5	**41,069.8**	**4,012.2**	1,000,000.0	0.0	**0.0000**	72,838.0	4,652.8	**0.0000**
f6	**48,042.5**	**76,482.9**	1,000,000.0	0.0	**0.0000**	562,361.0	50,380.6	**0.0000**
f7	**1,982.7**	**228.3**	73,234.6	31,258.2	**0.0000**	2,666.0	473.8	**0.0000**
f8	**7,419.3**	**272.6**	230,306.1	26,437.1	**0.0000**	17,043.0	489.1	**0.0000**
f9	**11,252.5**	**365.2**	764,340.0	299,686.8	**0.0000**	381,521.0	463,935.7	**0.0000**
f10	**9,316.5**	**404.6**	220,787.1	36,420.0	**0.0000**	64,521.0	12,180.4	**0.0000**
f11	**13,778.2**	**351.6**	1,000,000.0	0.0	**0.0000**	102,831.0	246,180.4	**0.0004**
f12	**14,197.5**	**474.7**	1,000,000.0	0.0	**0.0000**	424,432.0	266,159.8	**0.0000**
f13	**16,031.1**	**5,523.4**	1,000,000.0	0.0	**0.0000**	1,000,000.0	0.0	**0.0000**
f14	**2,106.0**	**259.5**	60,088.3	14,282.8	**0.0000**	231,947.0	419,823.4	**0.0000**
f15	22,933.2	42,985.1	178,373.6	23,584.0	**0.0000**	**11,035.0**	**1,321.4**	0.0062

For function 1, *Six Hump Camel Back*, PG-BA did not perform as fast as qABC or SPSO2011. This function is of low dimension (i.e. 2D). This could be due to random exploration since the pseudo-gradient calculation is only performed on the local search. PG-BA also performed moderately on function 15, *Alpine* which is an asymmetrical function. It shows that even though no direct gradient is computed, this method still inherits a slight behaviour of its gradient-based counterparts. This means that a good starting point will certainly benefits this algorithm. Nonetheless, this effect is balance out by the multiple patches build across the solution landscape. For the rest of the functions, PG-BA performed considerably well with most of the *p*-values less than 0.05 when compared to qABC and SPSO2011 which demonstrates the significance of the result. A lower number of evaluations of PG-BA exhibits a substantial good speed of search much so on multivariate unimodal functions where previous variant of the Bees Algorithm is lacking. PG-BA is also efficient in handling noise functions such as *Quartic*. The performance of PG-BA on symmetrical multimodal function of high dimensionality is also excellent. Statistically insignificant results especially when compared with SPSO2011 in function 2, *Shekel* and function 15, *Alpine* indicates that the two algorithms performed the same. qABC did not perform so well on multiple functions and this could be attributed to the neighbourhood radius as well as the recommendation value for limit of abandonment. Fine tuning for both parameters could potentially lead to a better result. Furthermore, the position of bees in qABC only takes into account the random index dimensions instead of the overall problem dimensions, which proved problematic for this algorithm in handling large scale problems.

Figures 3.5 until Figure 3.19 display the convergence curve for each function between PG-BA, qABC, and SPSO2011. Since qABC took quite a high number of function evaluations, these figures are scaled down so that the behaviour for each optimiser can be clearly observed. These figures show that qABC is quite fast in the beginning of the search but spends a lot of evaluation when nearing the optima. This is the exact behaviour of the Standard Bees Algorithm that prompted this research on PG-BA. PG-BA even though it starts at a low quality fitness, the pseudo-gradient direction clearly helps the algorithm progresses quickly. The same can be said of SPSO2011 when its added rotational invariance and velocity constraints that enable the algorithm to achieve a modest convergence rate.

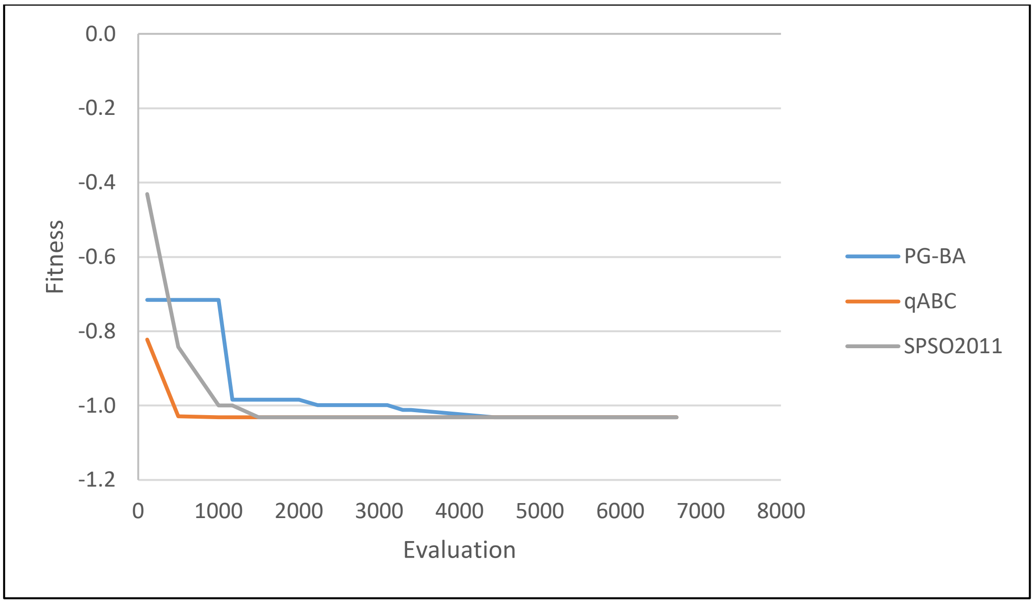

Figure 3.5: Convergence curve between PG-BA, qABC, and SPSO2011 for *f1*

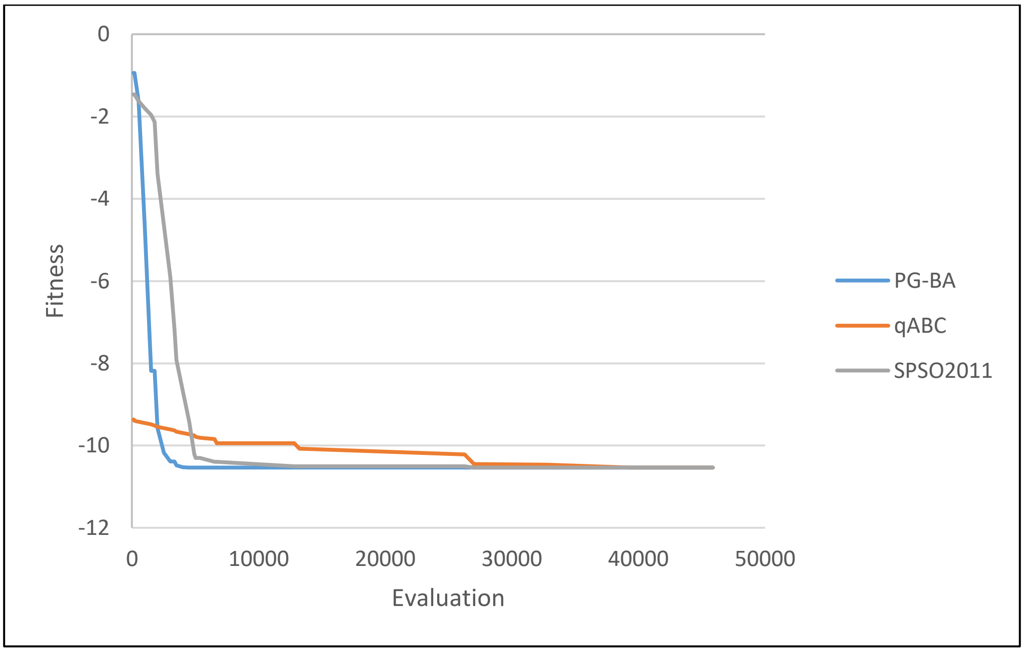

Figure 3.6: Convergence curve between PG-BA, qABC, and SPSO2011 for *f2*

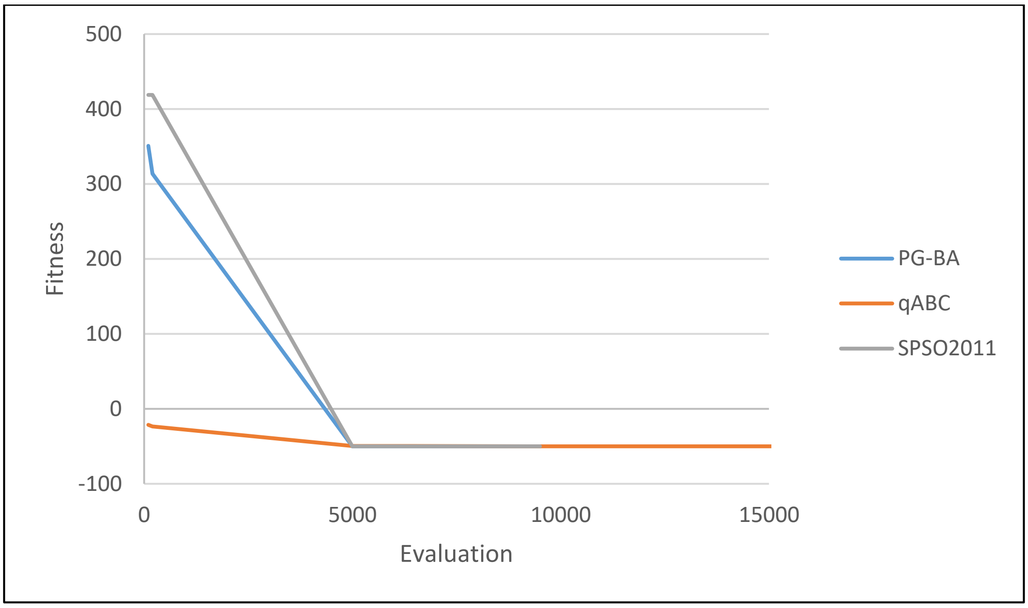

Figure 3.7: Convergence curve between PG-BA, qABC, and SPSO2011 for *f3*

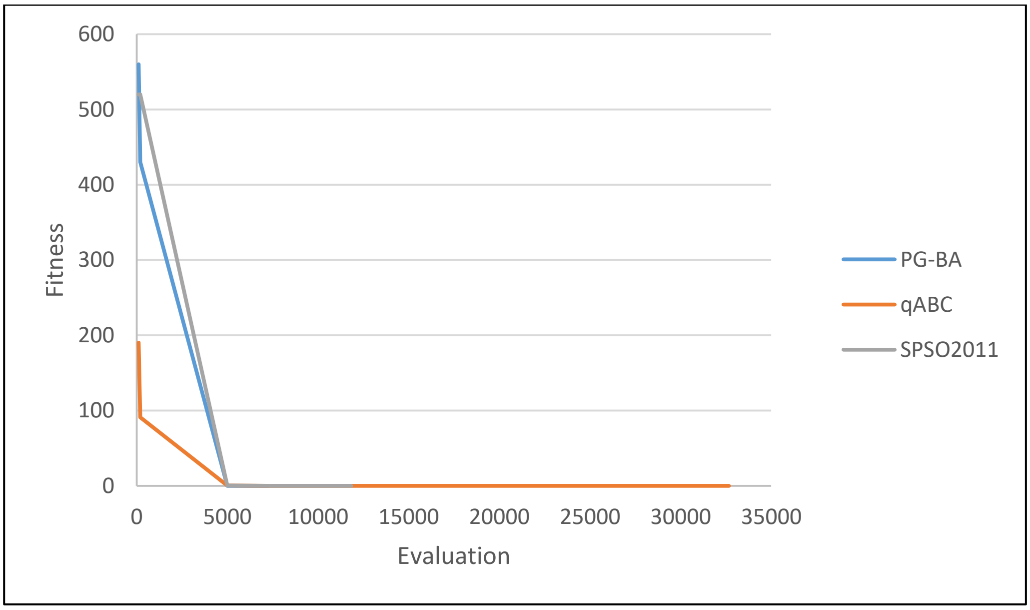

Figure 3.8: Convergence curve between PG-BA, qABC, and SPSO2011 for *f4*

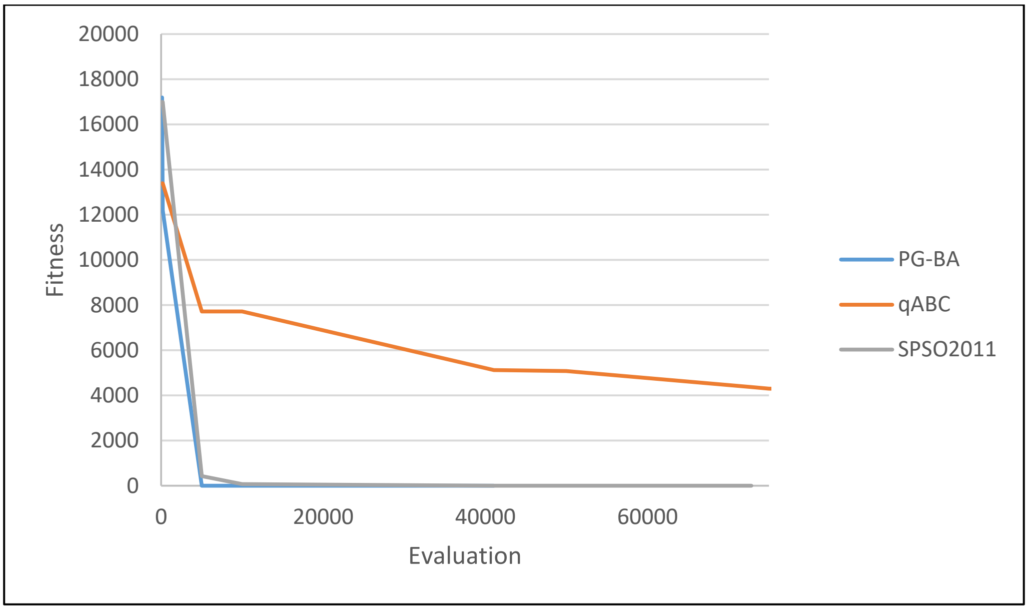

Figure 3.9: Convergence curve between PG-BA, qABC, and SPSO2011 for *f5*

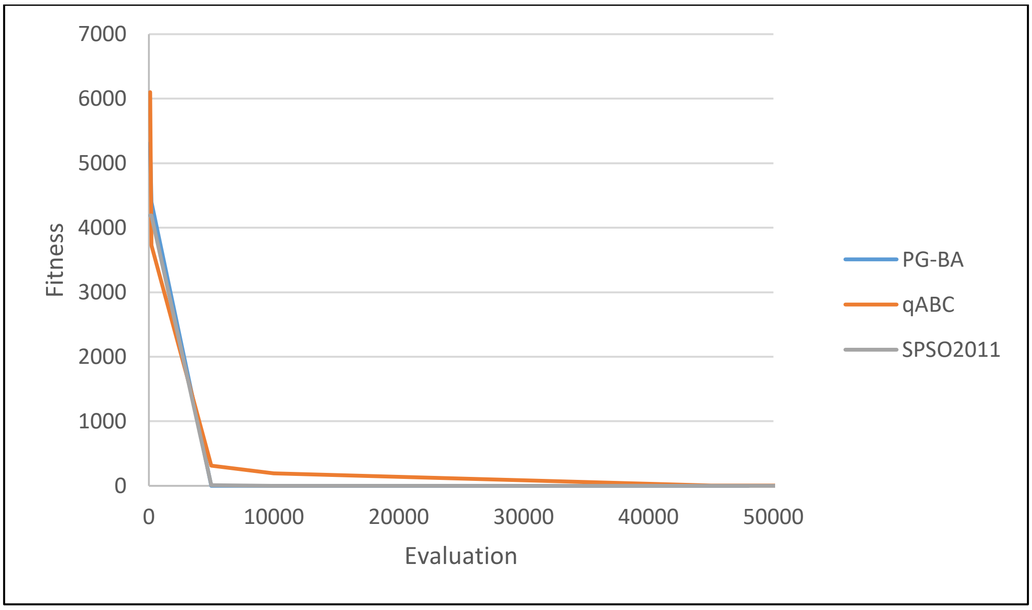

Figure 3.10: Convergence curve between PG-BA, qABC, and SPSO2011 for *f6*

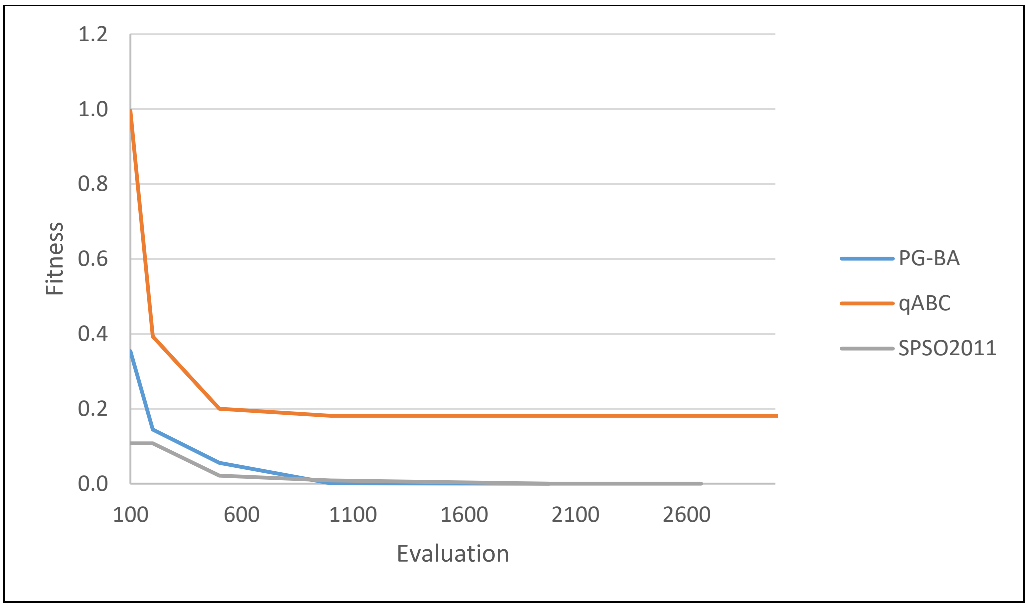

Figure 3.11: Convergence curve between PG-BA, qABC, and SPSO2011 for *f7*

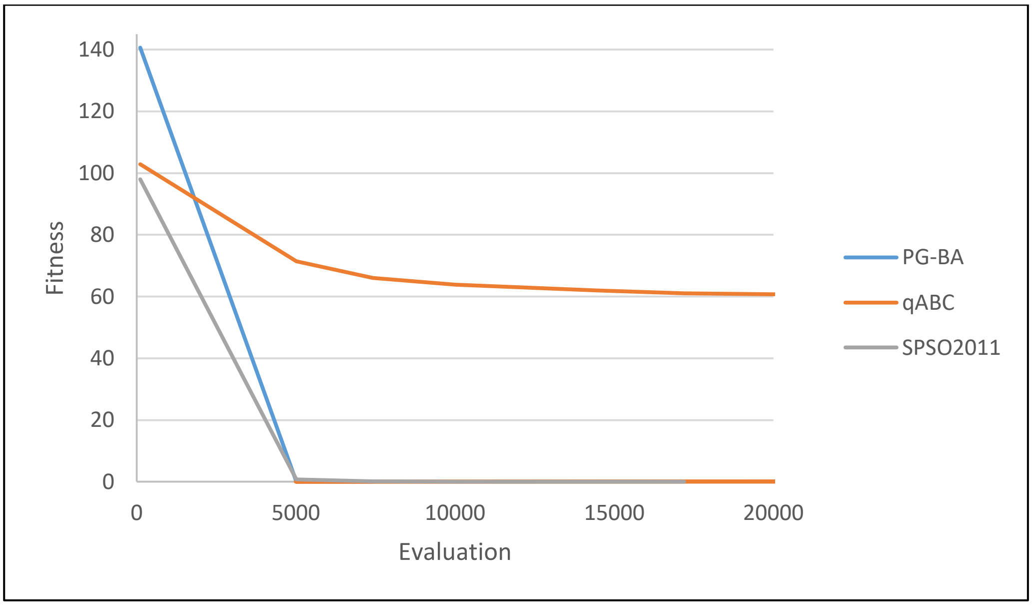

Figure 3.12: Convergence curve between PG-BA, qABC, and SPSO2011 for *f8*

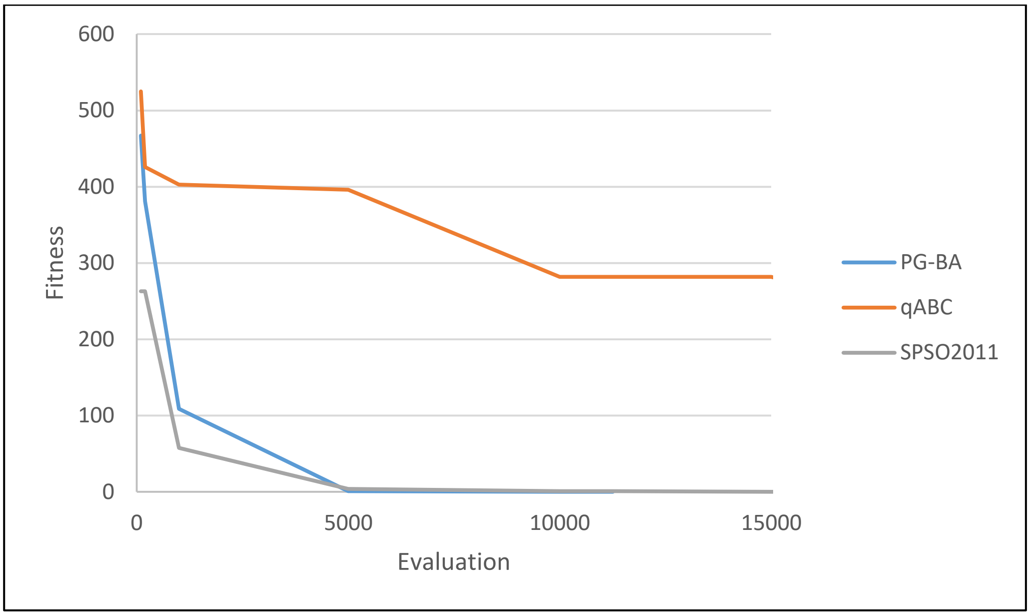

Figure 3.13: Convergence curve between PG-BA, qABC, and SPSO2011 for *f9*

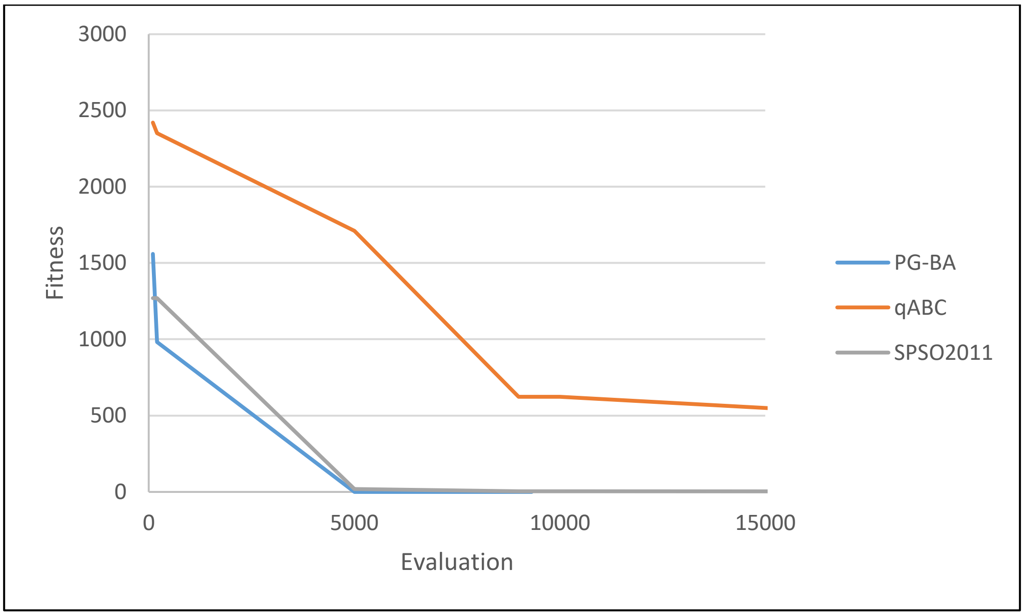

Figure 3.14: Convergence curve between PG-BA, qABC, and SPSO2011 for *f10*

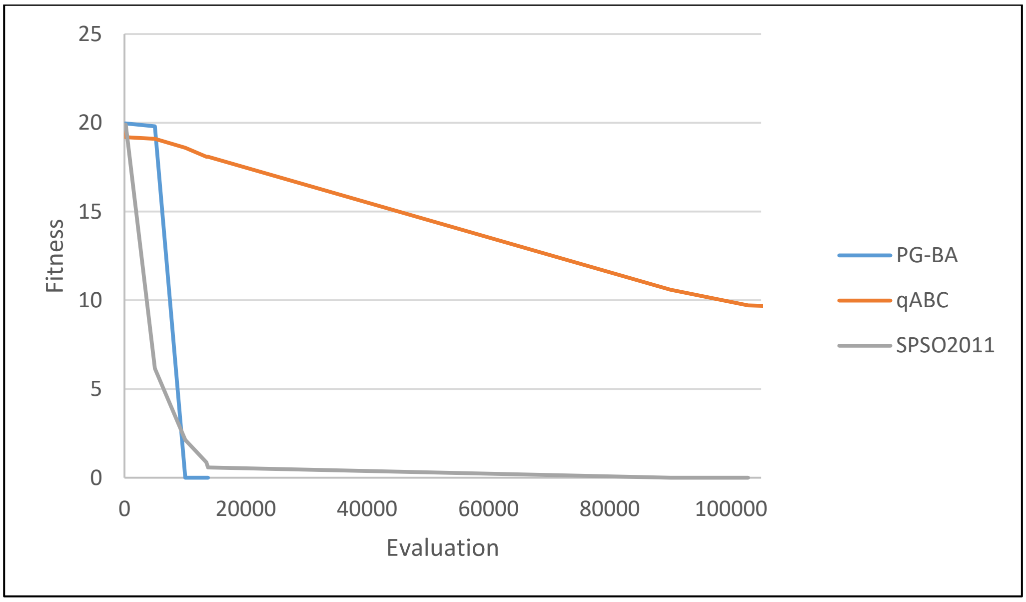

Figure 3.15: Convergence curve between PG-BA, qABC, and SPSO2011 for *f11*

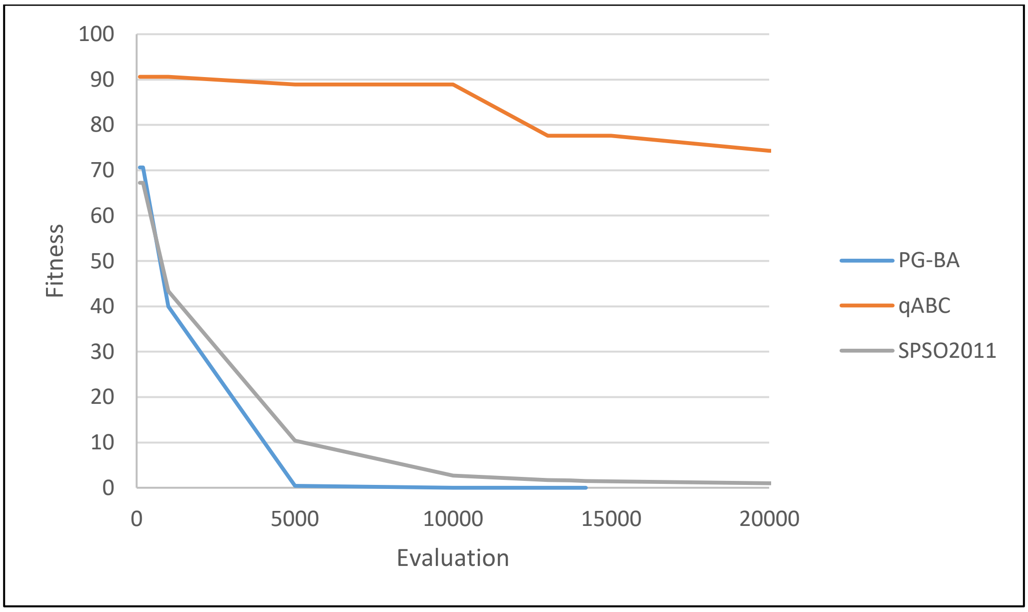

Figure 3.16: Convergence curve between PG-BA, qABC, and SPSO2011 for *f12*

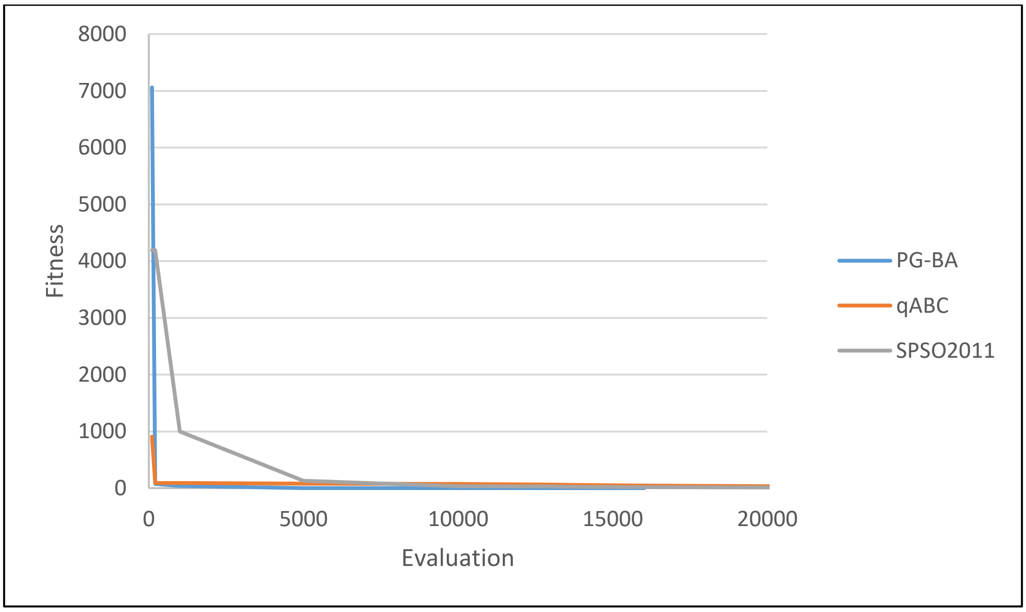

Figure 3.17: Convergence curve between PG-BA, qABC, and SPSO2011 for *f13*

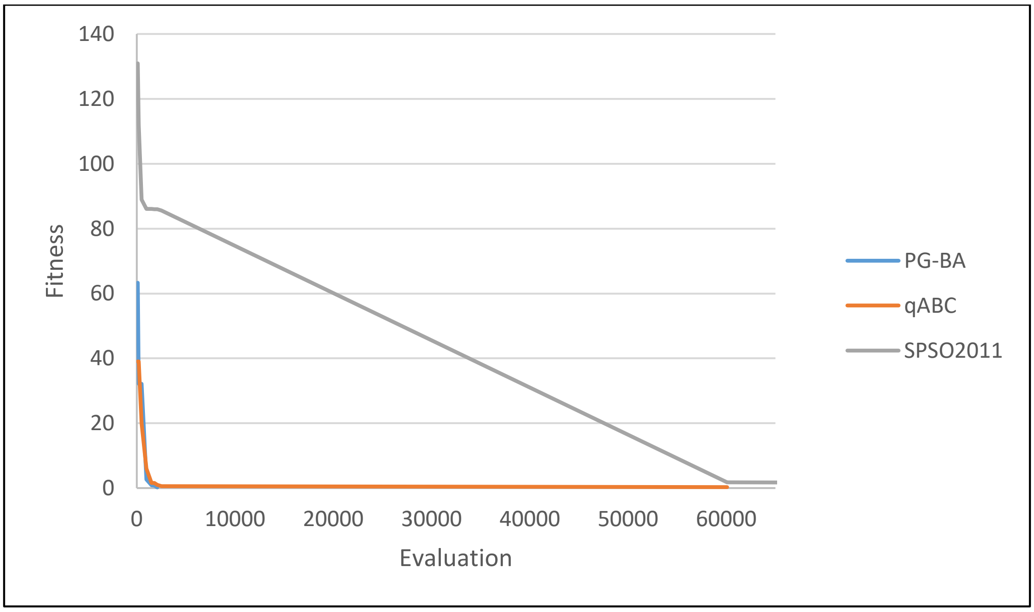

Figure 3.18: Convergence curve between PG-BA, qABC, and SPSO2011 for *f14*

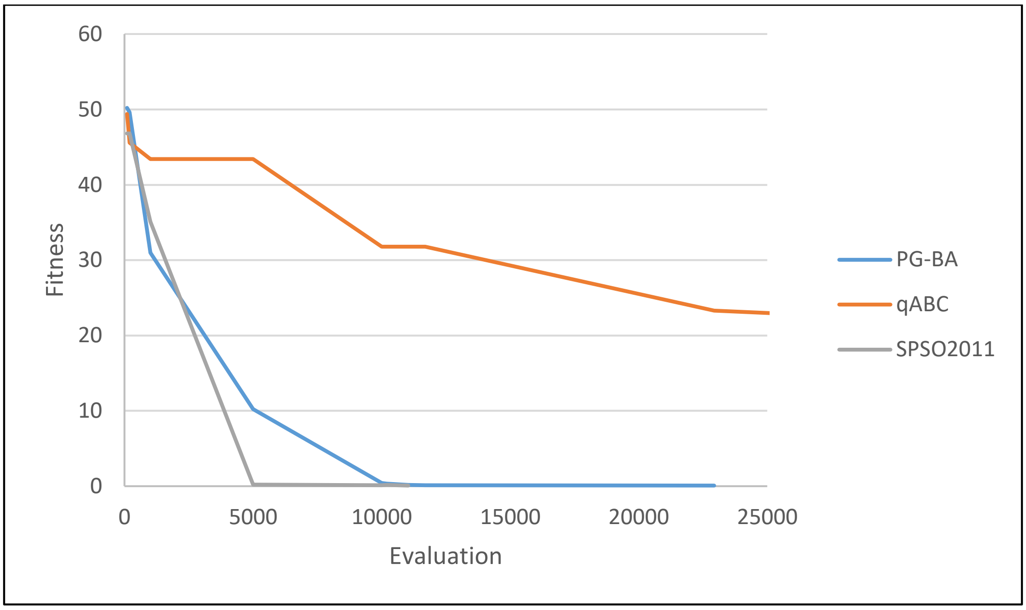

Figure 3.19: Convergence curve between PG-BA, qABC, and SPSO2011 for *f15*

3.5 Training of Feedforward Neural Network

The effectiveness of PG-BA is further verified by applying the algorithm to the training of Artificial Neural Network (ANN). The Artificial Neural Network (ANN) is a commonly used tool in pattern recognition and has the underlying principle of attempting to emulate the human nervous system in term of process information in the brain (Sivanandam and Deepa, 2006). They are nonlinear information processing devices, which consist of interconnected basic elements called neurons existing in various layers of the system. Every system has an input layer, hidden layer, and output layer. The input layer has input neurons which transmit data via synapses to the hidden layer, and similarly the hidden layer transfers this data to the output layer via more synapses. These synapses store values called weights which help them to manipulate the input and output to various layers.

There are many types of ANN but the most widely used is the feedforward neural network (FNN). FNN with multiple hidden (perceptron) layers is called Multi-Layer Perceptron (MLP). General structure of FNN is as illustrated in Figure 3.20 with *n* number of input nodes, *h* number of hidden nodes, *o* number of output nodes.

Assume that hidden transfer function is sigmoid function, and output transfer function is linear activation function, output at each hidden nodes is;

$$f(Hj) = 1/(1 + \exp(-(\sum_{i=1}^{n} W_{ij} \cdot X_i - \theta_j))), j = 1,2, \dots, h, \qquad (3.5)$$

where;

W_{ij} = connection weight from the *i*th node in the input layer to the *j*th node in the hidden layer,

θ_j = bias of the *j*th hidden node,

X_i = the *i*th input.

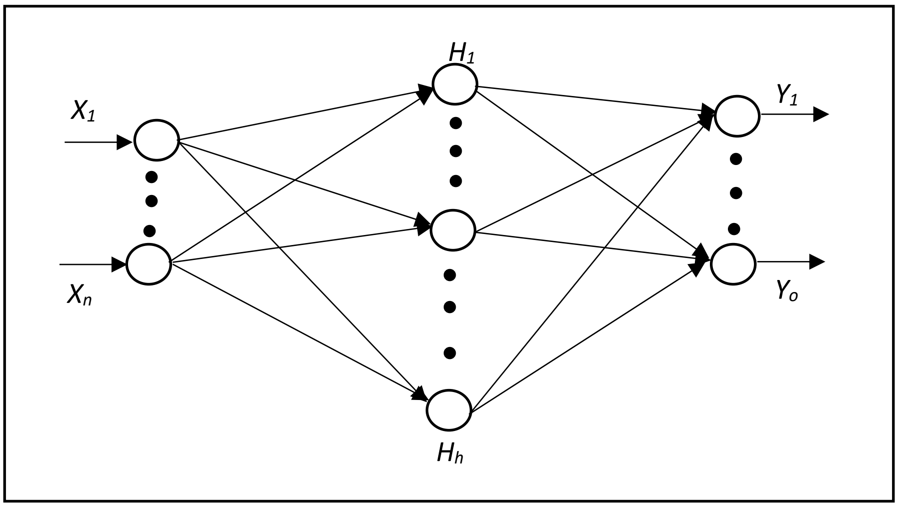

Figure 3.20: General structure of FNN

Thus, final output is;

$$Y_k = \sum_{j=1}^{h} W_{kj} \cdot f(S_j) - \theta_k, k = 1, 2, \dots, o \tag{3.6}$$

where;

W_{kj} = connection weight from jth hidden node to the kth output node,

θ_k = bias of the kth output node.

Therefore, learning error is;

$$E = \sum_{k=1}^{q} (\sum_{i=1}^{o} (Y_i^k - D_i^k)^2)/q \tag{3.7}$$

where;

q = number of training samples,

D_i^k = desired output of the ith input unit when the kth training sample is used,

Y_i^k = the actual output of the ith input unit when the kth training sample is used.

The success of FNN generally depends on its learning algorithm and network's structure. For many applications, the network's size is fixed, i.e. the number of hidden layers and their corresponding nodes is predetermined. Thus, a learning/training algorithm is used to find the optimum value of weights and biases (thresholds) that can produce minimum error. Traditionally, FNN training was done using the back-propagation (BP) algorithm. BP is a gradient-based technique; therefore, it also experiences common problems related to gradient-based methods. These problems are such easily getting trapped in the local minima especially for those non-linearly separable pattern classification problems, and dependency on the choices of its initial values of the network connection weights as well as the parameters in the algorithm such as learning rate, and momentum coefficient (Zhang et al., 2007; Mirjalili et al., 2012).

In the literature, many population-based optimisation algorithms have been used instead to train FNN. In fact, for the Bees Algorithm, the majority of its application is through the use of ANN as reviewed in Chapter 2. Hence, PG-BA is applied as a learning algorithm for FNN in this research for benchmark problem of Exclusive-OR.

3.5.1 Experimental set-up for FNN training

Exclusive-OR (XOR) is a classification benchmark problem where in this research FNN is used to recognise the number '1' in the input vector. If the input vector contains an odd number of '1', then the output is '1'. If the input vector has an even number of '1' or none at all, then the output is '0'. Table 3.9 shows the inputs and desirable outputs for this two bits problem.

Table 3.9: Exclusive-OR problem

Input		Output
0	0	0
0	1	1
1	0	1
1	1	0

This problem has a nonlinear separable pattern which required the use of hidden layers. This experiment also used biases to move the threshold of transfer functions. As it has two inputs and one output, the structure of FNN for this research is 2-H-1 where H is the number of hidden nodes. Figure 3.21 is an example of 2-2-1 network. In this research, H = 2 until 15, 20, 25, 30 so that the performance of the algorithm can be compared when subjected to a high number of hidden nodes. As the number of hidden nodes increase, the dimensionality of the problem also escalates.

In general, the number of weights and biases to be optimised by FNN is calculated as;

Number of weights and biases (Problem dimension) = $(n * h) + (h * o) + h + o$ (3.8)

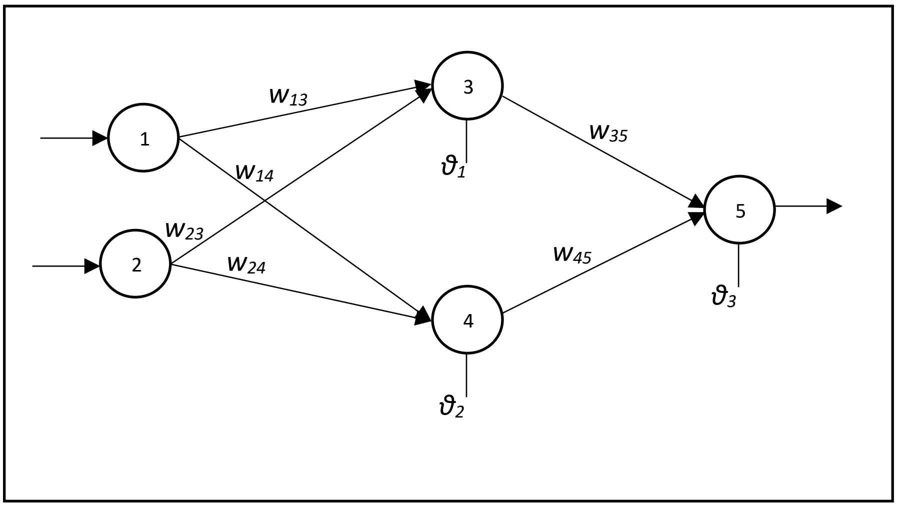

Figure 3.21: FNN with 2-2-1 structure

The range of the problem is set at [-10, 10]. Meanwhile, the same parameter setting as previous experiment is used for all algorithms. The performance measure is in term of mean square error (MSE) as well as the number of evaluation. The fitness function for this problem is derived from Equation 3.7;

$$Fitness\ (X_i) = E(X_i) \qquad (3.9)$$

All algorithms shall terminate when MSE<0.001 or when it reached 1000 epochs (cycles). Average performance is taken for 25 algorithms.

3.5.2 Results and discussion for FNN training

Tables 3.10 and 3.11 compare performance of FNN training for XOR problem between PG-BA, Standard Bees Algorithm, qABC, and SPSO2011. All algorithms were able to recognise the XOR pattern below the allowable MSE. As the number of hidden nodes increase, the accuracy of recognition also improves (Figure 3.22). Even though the number of variables (i.e. weights and biases) to be optimised soared, the number of function evaluation (NFE) for all algorithms decreased (Figure 3.23). It means for this particular problem; scalability is not an issue for all algorithms involved.

The performance for PG-BA and Standard Bees Algorithm is almost the same when $H = 3$ until 30, for MSE. While for NFE, PG-BA's improvement is not that much with the exception for $H = 2$. Statistics analysis using *t*-test ($p < 0.05$) shows that for XOR problem, there no significance difference between PG-BA and the Standard Bees Algorithm for most cases. From the result, it can be deduced that the suitable number of hidden nodes is equal or more than 13 as MSE is nearly stagnant at this point until $H = 30$.

Both variants of the Bees Algorithm converge faster than SPSO2011 and qABC with qABC has the highest NFE. MSE remains stable from $H = 14$ for SPSO2011 where for qABC the smallest value of MSE attained is when $H = 30$. In term of MSE, from statistically point of view, all algorithms' performance except qABC can be considered the same for most of the number of hidden nodes when compared to PG-BA. In contrast, the faster convergence due to low NFE for PG-BA is statistically significant compared to qABC and SPSO2011. This echoes the performance in previous experiments of numerical functions.

Table 3.10: MSE comparison of PG-BA, Standard Bees Algorithm, qABC, and SPSO2011 for XOR problem

H	**Dim.**	**Standard Bees Algorithm**				**PG-BA**			**qABC**				**SPSO2011**			
		Success rate (%)	**MSE**			**Success rate (%)**	**MSE**		**Success rate (%)**	**MSE**			**Success rate (%)**	**MSE**		
			Mean	**Std. Dev**	***p*-value**		**Mean**	**Std. Dev**		**Mean**	**Std. Dev**	***p*-value**		**Mean**	**Std. Dev**	***p*-value**
2	9	96	0.0008	0.0002	0.0835	100	0.0007	0.0002	68	0.0265	0.0499	**0.0128**	100	0.0008	0.0002	0.0835
3	13	100	0.0007	0.0002	-	100	0.0007	0.0002	68	0.0098	0.0268	0.0960	100	0.0006	0.0003	0.1719
4	17	100	0.0007	0.0002	-	100	0.0007	0.0002	92	0.0060	0.0250	0.2945	100	0.0007	0.0002	-
5	21	100	0.0006	0.0003	1.0000	100	0.0006	0.0002	92	0.0009	0.0007	**0.0448**	100	0.0008	0.0001	**0.0000**
6	25	100	0.0005	0.0002	0.0835	100	0.0006	0.0002	96	0.0007	0.0003	0.1719	100	0.0006	0.0003	1.0000
7	29	100	0.0006	0.0003	-	100	0.0006	0.0003	92	0.0009	0.0007	0.0547	100	0.0007	0.0003	0.2444
8	33	100	0.0005	0.0003	1.0000	100	0.0005	0.0002	96	0.0006	0.0004	0.2691	100	0.0006	0.0003	0.1719
9	37	100	0.0005	0.0003	-	100	0.0005	0.0003	100	0.0007	0.0002	**0.0079**	100	0.0005	0.0003	-
10	41	100	0.0004	0.0003	0.1719	100	0.0005	0.0002	92	0.0011	0.0015	0.0532	100	0.0005	0.0003	1.0000
11	45	100	0.0005	0.0003	0.2444	100	0.0006	0.0003	96	0.0007	0.0003	0.2444	100	0.0005	0.0003	0.2444
12	49	100	0.0005	0.0003	0.2444	100	0.0004	0.0003	100	0.0006	0.0003	**0.0225**	100	0.0004	0.0003	-
13	53	100	0.0004	0.0003	-	100	0.0004	0.0003	100	0.0008	0.0002	**0.0000**	100	0.0005	0.0003	0.2444
14	57	100	0.0004	0.0003	-	100	0.0004	0.0003	100	0.0006	0.0003	**0.0225**	100	0.0004	0.0003	-
15	61	100	0.0004	0.0003	0.1719	100	0.0003	0.0002	100	0.0007	0.0003	**0.0000**	100	0.0004	0.0003	0.1719
20	81	100	0.0004	0.0003	-	100	0.0004	0.0003	100	0.0006	0.0003	**0.0225**	100	0.0003	0.0003	0.2444
25	101	100	0.0003	0.0003	0.1719	100	0.0002	0.0002	100	0.0005	0.0003	**0.0000**	100	0.0004	0.0003	**0.0079**
30	121	100	0.0003	0.0003	0.1719	100	0.0002	0.0002	100	0.0004	0.0003	**0.0079**	100	0.0003	0.0003	0.1719

Table 3.11: NFE comparison of PG-BA, Standard Bees Algorithm, qABC, and SPSO2011 for XOR problem

H	Dim.	Standard Bees Algorithm			PG-BA			qABC			SPSO2011		
		Evaluation			Evaluation			Evaluation			Evaluation		
		Mean	Std. Dev	*p*-value	Mean	Std. Dev	% Improv.	Mean	Std. Dev	*p*-value	Mean	Std. Dev	*p*-value
2	9	7,928.24	22,446.88	0.4500	4,506.28	9,291.27	43.16	62,949.80	32,724.17	**0.0000**	5,816.00	2,667.99	0.5014
3	13	2,230.00	1,439.17	**0.0289**	1,568.04	294.08	29.68	61,305.32	35,577.97	**0.0000**	4,628.00	2,008.29	**0.0000**
4	17	1,758.00	529.09	**0.0109**	1,440.00	283.58	18.09	55,925.60	30,331.83	**0.0000**	4,264.00	2,021.07	**0.0000**
5	21	1,517.00	308.18	0.0627	1,344.04	333.15	11.40	44,378.20	27,275.18	**0.0000**	3,500.00	1,517.89	**0.0000**
6	25	1,430.00	312.41	**0.0320**	1,252.00	254.56	12.45	40,299.72	31,511.42	**0.0000**	2,920.00	1,233.21	**0.0000**
7	29	1,482.00	408.63	**0.0051**	1,200.00	251.90	19.03	31,539.88	30,306.12	**0.0000**	2,760.00	1,705.99	**0.0000**
8	33	1,264.00	269.99	0.5648	1,210.00	379.47	4.27	28,600.00	32,437.25	**0.0000**	2,416.00	1,476.94	**0.0003**
9	37	1,114.00	274.05	0.8153	1,096.00	267.85	1.62	30,021.44	21,110.29	**0.0000**	1,920.00	794.98	**0.0000**
10	41	1,146.00	289.11	0.5341	1,096.00	275.22	4.36	22,256.48	26,053.59	**0.0002**	1,660.00	1,141.23	**0.0202**
11	45	1,106.00	321.35	0.1795	996.00	244.43	9.95	24,123.28	23,121.98	**0.0000**	1,424.00	840.13	**0.0182**
12	49	1,214.04	347.46	**0.0054**	948.00	295.13	21.91	19,819.24	22,093.00	**0.0000**	1,592.00	758.38	**0.0002**
13	53	1,246.00	457.37	**0.0075**	972.00	176.64	21.99	23,866.12	22,380.14	**0.0000**	1,520.00	656.05	**0.0002**
14	57	1,040.00	287.78	**0.0273**	842.00	326.09	19.04	20,579.56	21,023.73	**0.0000**	1,420.00	809.44	**0.0018**
15	61	1,126.04	228.62	**0.0000**	860.00	213.77	23.63	12,402.20	11,687.24	**0.0000**	1,384.00	791.29	**0.0025**
20	81	918.00	234.47	0.2378	832.00	272.76	9.37	18,928.48	17,054.82	**0.0000**	1,012.00	503.05	0.1223
25	101	860.00	256.31	0.0784	714.00	314.81	16.98	12,723.00	14,484.31	**0.0000**	1,036.00	616.36	**0.0243**
30	121	798.00	348.85	0.7305	768.00	256.25	3.76	12,480.12	15,929.33	**0.0006**	616.00	371.68	0.0988

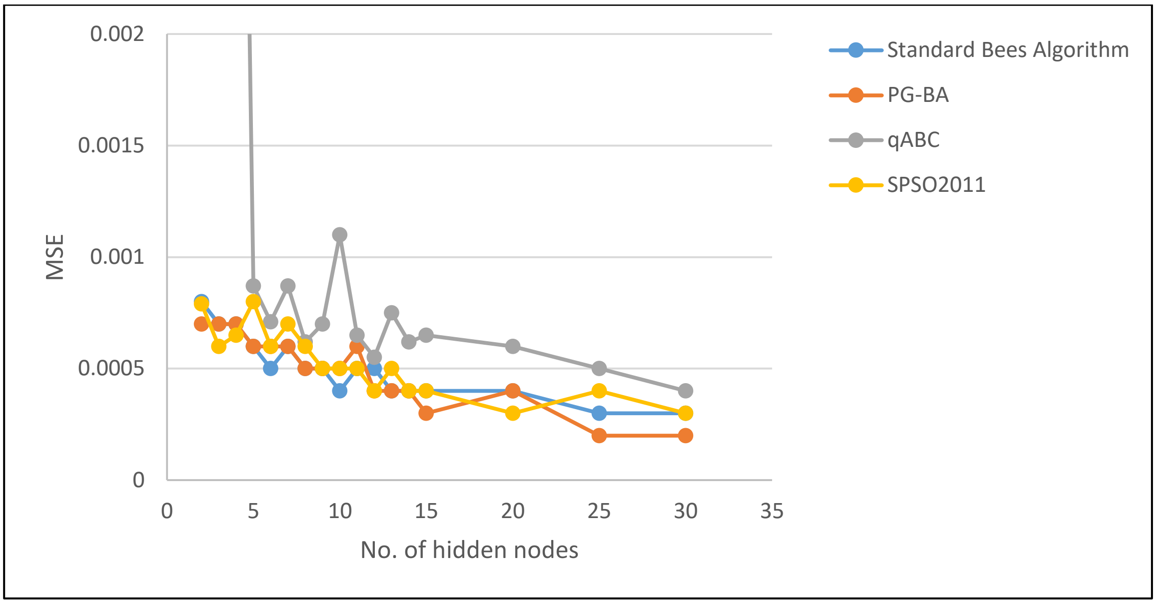

Figure 3.22: Effect of number of hidden nodes to MSE for PG-BA, Standard Bees Algorithm, qABC, and SPSO2011

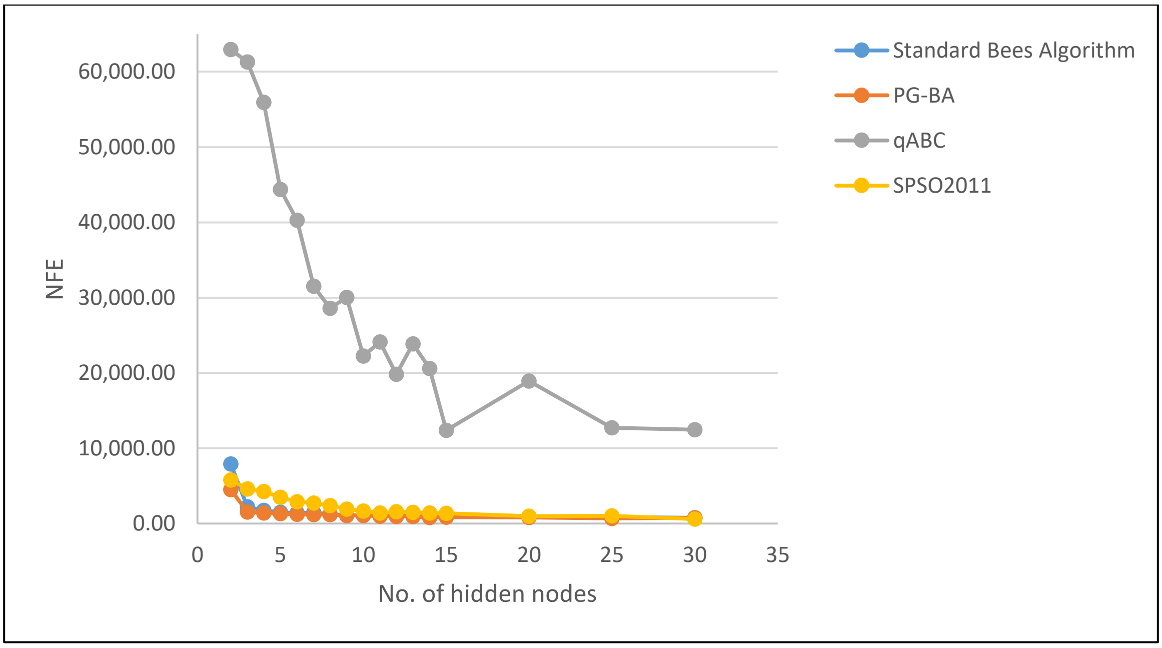

Figure 3.23: Effect of number of hidden nodes to NFE for PG-BA, Standard Bees Algorithm, qABC, and SPSO2011

3.6 Conclusions

This chapter introduced four new variants of hybrid Bees Algorithm with the pseudo-gradient method. Each variant differs in term of the position of the scout bee inside of the neighbourhood. It is aimed at enhancing the convergence speed of the algorithm by providing better directed guidance towards the optimal solution without the need for the objective function to be differentiable. Experimental results demonstrated that PG-BA1, which is the simplest form of them all, significantly outperforms the Standard Bees Algorithm on various numerical benchmark functions. This is because the other variants made the optimisation process too exploitative which could backfire in complex search space.

It should also be noted that when the scalability test was performed, it showed that the performance of PG-BA is almost the same across the dimension. Still, NFE reached more than 40,000 in function *Schwefel 1.2* and *Schwefel 2.22* at higher dimensions. Therefore, in the future, an advanced strategy is needed to reduce the rate of convergence even more especially when dealing with large scale problems of those particular functions.

In addition, PG-BA were utilised in the training of Artificial Neural Network for recognition of Exclusive-OR logic for successful results. A statistical test was also carried out on the results obtained.

CHAPTER 4

A PATCH OVERLAP AVOIDANCE BEES ALGORITHM (POA-BA)

4.1 Preliminaries

Through the use of global random search, there are likelihoods that scout bees in the Bees Algorithm will land on the exact or in the surrounding of previously visited area or abandoned sites. If they were chosen for further exploitation in the neighbourhood search, it will cause a wastage of resources (recruiters) to forage what should have been known as unpromising patches. In return, a higher number of function evaluations are incurred.

Therefore, this research strives to implement a temporary memory to the Bees Algorithm. The memory records previously visited positions, along with the fitness and patch boundary to avoid overlapping patches from forming in the current and subsequent iterations.

4.2 Patch Overlap Avoidance Bees Algorithm (POA-BA)

A sophisticated algorithm normally employs memory, whether it is a short term or long term, to guide the search into new promising area (Consoli, 2006; Alia and Mandava, 2011; Brownlee, 2011). One of the most well-known memory-embedded algorithms is Tabu Search

(TS) invented by Glover (1986). By using tabu or forbidden lists, earlier visited solutions or a set of rules determined by the user are memorised. If the new solution violates a rule or falls on the previously visited solution, it is stored in the tabu list. Thus in TS, there is no possibility of repeated solutions.

The Bees Algorithm has been integrated with TS as in Shafia et al. (2011) and in the doctoral thesis of Imanguliyev (2013) but with different rules on the condition of entering and exiting the tabu list. In this project, a short term memory akin to tabu list, archives previously visited solutions (including abandoned sites), their fitness, and corresponding neighbourhood to reduce the number of overlapping patches subjected to neighbourhood search. The fixed-length short term memory updated every generation is chosen so as not to put too much burden on the computing cost. Besides lessening redundant search of unprofitable area, it also minimises the number of patches founded in the same region of attraction. This is to ensure thorough search of the fitness landscape as recruiters are well disseminated and are not wasted on an already foraged zone. However, different patches can still exist on similar hills (for maximisation problem) or valleys (for minimisation problem) if they do not overlap each other. The idea is by decreasing the number of repeated site shrinks the search area which in turn lowering the number of function evaluation needed to achieve optima solution. In addition, flower patches are not allowed to intersect, not just in the next generation but also in the current cycle except on certain conditions which will be described in the next few paragraphs. Figure 4.1 illustrates the operation of the Bees Algorithm integrated with such memory dubbed as the Patch Overlap Avoidance Bees Algorithm (POA-BA).

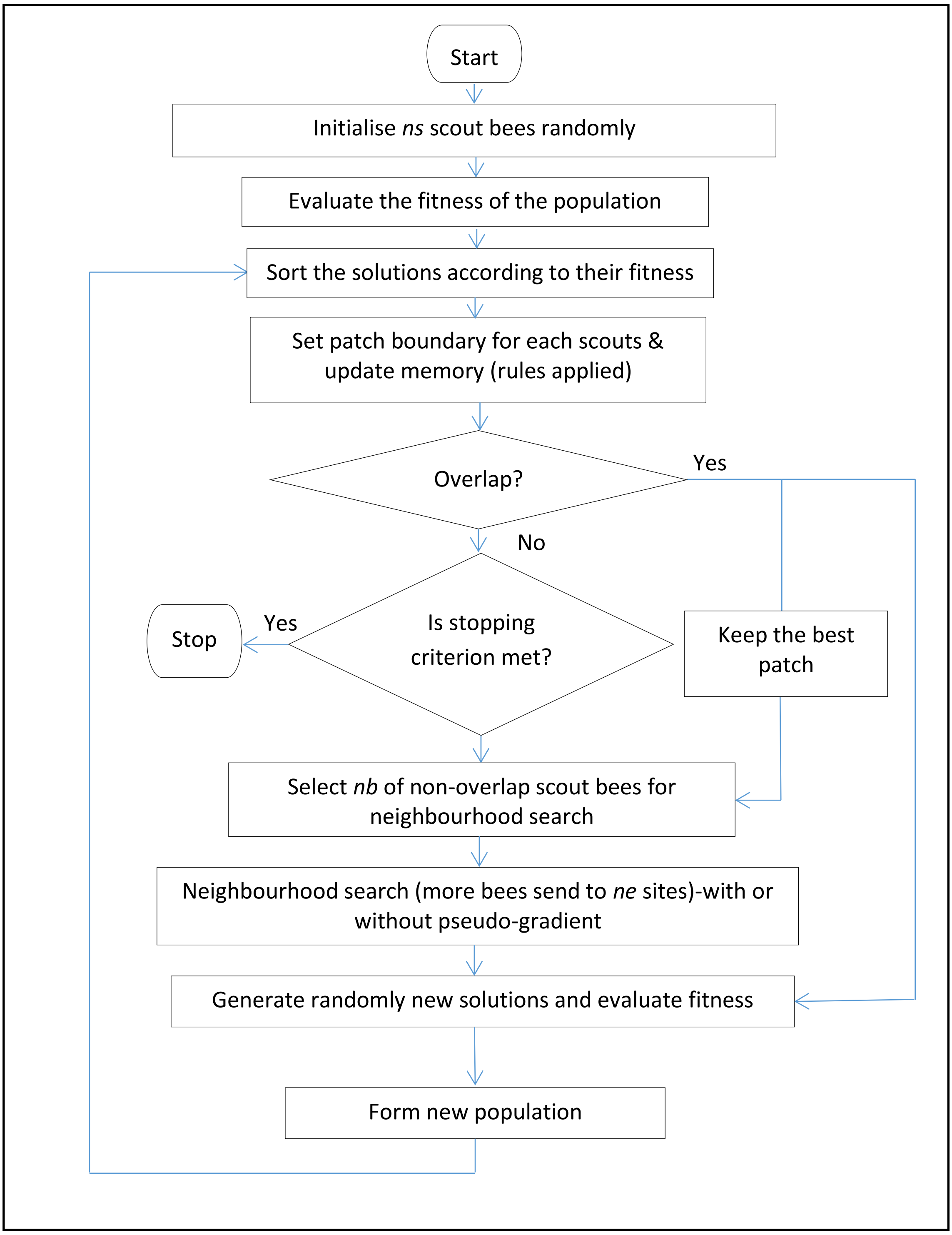

Figure 4.1: Flow chart of POA-BA

The proposed algorithm starts by placing *ns* number of scout bees randomly in the search space. Then, the fitness of scout bees is evaluated and sorts accordingly. The patch boundary for each scout which involved their position and initial neighbourhood size (Equation 4.1 and 4.2) is set and recorded in a short term memory along with their corresponding fitness.

$$\textit{Minimum boundary} = \textit{scout position} + \textit{neighbourhood size}, \tag{4.1}$$

$$\textit{Maximum boundary} = \textit{scout position} - \textit{neighbourhood size}. \tag{4.2}$$

The size of the memory is equal to *ns*. Scouts are checked if their patch limits overlap each other using the following rules;

If scout position (2) > scout position (1)

If minimum boundary.scout position (1) > maximum boundary.scout position (2)

Flag=overlap

Elseif minimum boundary.scout position (2) > maximum boundary.scout position (1)

Flag=overlap

End

End

nb number of scout bees of non-overlapping patch with highest fitness recruits more bees in the neighbourhood search. However, if the number of non-overlapping scouts is less than the value of *nb*, then the fittest bees of overlapping boundary will fill this gap. Neighbourhood search is performed as in the Standard Bees Algorithm or the pseudo-gradient method. More bees are recruited for *ne* sites. The remaining unselected bees are sent arbitrarily for global search and their fitness is evaluated.

In the next cycle, the fittest bees from each patch are discovered during neighbourhood search and the global scouts, form the new scout bees' population. Again, the boundary of the new population of scout bees is set, but here the neighbourhood size might change if neighbourhood shrinking is applied during neighbourhood search. Neighbourhood shrinking is initialised if there is no improvement made on the fitness. The boundary of the current scouts is checked against each other as well as the ones in the memory. If the number of *nb* required is more than the available non-overlapping scouts, and if any members of the memory have higher fitness than the current overlap scouts, memory's member(s) of higher fitness will trade places with the current overlap scout(s) of low fitness to satisfy the *nb* value. This is how the temporary memory list is updated and the size remains fixed throughout the evolution process. The memory always stored the patch boundary information and fitness of low promising area.

Another way of freeing and entering the memory is when site abandonment occurs. If no yield is found after *stlim* of stagnation limit, the site is abandoned as it is deemed stuck in the local optima. The patch boundary and fitness of abandoned site are recorded and members inside the memory with higher fitness are discarded. Furthermore, if local and global basin of attraction are close to each other and hence overlap, the region where the local optimum lies might be selected for further exploitation if the fitness found at that time is higher. Hence, allowing stored memory of higher fitness to be reconsidered in the search area can improve odds of finding the missing point. The rest of the search process follows the previously explained steps until a stopping criterion is met.

4.3 Numerical Functions Experiment on POA-BA

Two version of POA-BA are introduced. In the first version, standard neighbourhood search is implemented while the other employs the use of pseudo-gradient. The latter will be called POA-PG-BA. Both versions are tested using the same benchmark functions as described in Chapter 3. This experiment also used the same control parameters, as well as stopping criterion i.e. accuracy to achieved and maximum evaluation. Similar performance measures are utilised which is the average number of function evaluations (NFE) of 100 runs. Statistical *t*-test is also performed on results obtained. Outcomes of the experiment are compared to results of the Standard Bees Algorithm, Pseudo-Gradient Bees Algorithm (PG-BA), Quick Artificial Bee Colony (qABC), and Standard Particle Swarm Optimisation 2011 (SPSO2011) as in Chapter 3.

4.3.1 Results and discussion

Table 4.1 shows the performance of POA-BA and POA-PG-BA in terms of NFE in mean and standard deviation taken for 100 runs. Overall, POA-BA and POA-PG-BA are able to reach the global optimum solution in fewer function evaluations compared to the Standard Bees Algorithm, PG-BA, as well as qABC, and SPSO2011. The percentage reduction of NFE for POA-BA compared to the Standard Bees Algorithm and PG-BA is 81.48% and 46.93% respectively as shown in Table 4.2. This demonstrates the efficiency of using memory to avoid site repetition.

Table 4.1 Mean and standard deviation of function evaluations over 100 runs for POA-BA experiment

Func.	POA-BA		POA-PG-BA		Standard BA		PG-BA		qABC		SPSO2011	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
f1	495.8	624.1	**467.3**	**594.8**	7,969.6	3,741.5	6,700.4	2,956.7	704.1	885.8	3,397.0	918.1
f2	1,586.7	722.7	**1,464.0**	**583.5**	36,720.9	2,925.2	5,389.6	1,529.0	45,888.5	194,909.7	27,016.0	139,013.1
f3	3,404.6	571.4	**2,965.0**	**483.0**	6,527.0	348.5	6,431.0	332.8	529,982.7	142,468.2	9,609.0	504.0
f4	4,931.8	485.3	**4,322.7**	**557.8**	8,557.0	463.3	7,242.0	325.1	32,673.9	10,321.5	11,822.0	509.2
f5	23,919.1	4,805.5	**19,474.9**	**4,576.7**	48,364.2	3,547.9	41,069.8	4,012.2	1,000,000.0	0.0	72,838.0	4,652.8
f6	18,222.5	15,840.0	**13,710.6**	**7,157.7**	998,884.0	11,223.5	48,042.5	76,482.9	1,000,000.0	0.0	562,361.0	50,380.6
f7	1,280.4	395.7	**1,117.8**	**211.9**	3,286.3	454.8	1,982.7	228.3	73,234.6	31,258.2	2,666.0	473.8
f8	4,464.2	404.6	**3,425.4**	**356.5**	14,460.4	628.2	7,419.3	272.6	230,306.1	26,437.1	17,043.0	489.1
f9	6,128.1	309.7	**4,846.1**	**381.8**	28,085.7	11,878.6	11,252.5	365.2	764,340.0	299,686.8	381,521.0	463,935.7
f10	4,297.7	463.5	**3,857.6**	**330.0**	43,797.6	9,749.4	9,316.5	404.6	220,787.1	36,420.0	64,521.0	12,180.4
f11	7,237.7	366.0	**6,390.2**	**308.4**	822,726.9	326,531.3	13,778.2	351.6	1,000,000.0	0.0	102,831.0	246,180.4
f12	9,221.3	326.6	**7,430.5**	**349.8**	1,000,000.0	0.0	14,197.5	474.7	1,000,000.0	0.0	424,432.0	266,159.8
f13	14,302.9	3,817.2	**13,645.0**	**4,142.1**	1,000,000.0	0.0	16,031.1	5,523.4	1,000,000.0	0.0	1,000,000.0	0.0
f14	1,317.1	281.4	**1,223.2**	**264.6**	292,293.1	452,737.3	2,106.0	259.5	60,088.3	14,282.8	231,947.0	419,823.4
f15	10,920.8	1,386.3	**10,159.6**	**1,496.8**	1,000,000.0	0.0	22,933.2	42,985.1	178,373.6	23,584.0	11,035.0	1,321.4
Total Eval.	111,730.7		**94,500.0**		5,311,672.7		213,892.4		7,136,378.9		2,923,039.0	

Table 4.2: Percentage of improvements of POA-BA and POA-PG-BA in comparison to the Standard Bees Algorithm, PG-BA, and each other

Func.	POA-BA		POA-PG-BA		
	Standard BA	PG-BA	Standard BA	PG-BA	POA-BA
f1	93.78	92.60	94.14	93.03	5.76
f2	95.68	70.56	96.01	72.84	7.73
f3	47.84	47.06	54.57	53.89	12.91
f4	42.37	31.90	49.48	40.31	12.35
f5	50.54	41.76	59.73	52.58	18.58
f6	98.18	62.07	98.63	71.46	24.76
f7	61.04	35.42	65.99	43.62	12.70
f8	69.13	39.83	76.31	53.83	23.27
f9	78.18	45.54	82.75	56.93	20.92
f10	90.19	53.87	91.19	58.59	10.24
f11	99.12	47.47	99.22	53.62	11.71
f12	99.08	35.05	99.26	47.66	19.42
f13	98.57	10.78	98.64	14.88	4.60
f14	99.55	37.60	99.58	41.92	7.14
f15	98.91	52.38	98.98	55.70	6.97
Avg. Improv.	81.48	46.93	84.30	54.06	13.27

The performance of POA-PG-BA is slightly better than POA-BA due to the use of pseudo-gradient method in the neighbourhood search. On average, the percentage of improvement for POA-PG-BA compared to POA-BA is 13.27. Despite that, statistical analysis (Table 4.3) shows that the performances of both POA-BA variants are significantly the same for function *f1 Six Hump Camel Back*, *f2 Shekel*, and *f13 Schwefel 2.22*. *f1* and *f2* are multimodal functions of low dimensions while *f13* is a 30D complex multimodal function. The same can be said for POA-BA and qABC execution on *f1*, both POA-BA version and SPSO 2011 on *f2*, and POA-BA with SPSO2011 on asymmetrical *f15 Alpine*. For the rest of the functions, all improvements made are statistically justified.

Figures 4.2 until 4.16 display the convergence curve for all the Bees Algorithm variants for global optimisation developed in this research as well as the standard version. All figures clearly show the faster rate of convergence for both POA-BA variants compared to the Standard Bees Algorithm and PG-BA. This further validates the effectiveness of using memory to reduce the occurrences of overlapping patches in maximising the speed of search compared to the non-memory versions.

Table 4.3: *p*-values using t-test ($p = 0.05$) comparing POA-BA and POA-PG-BA with the Standard Bees Algorithm, PG-BA, qABC, SPSO2011, and each other

Func.	**POA-PG-BA**	**Standard BA**		**PG-BA**		**qABC**		**SPSO2011**	
	POA-BA	**POA-BA**	**POA-PG-BA**	**POA-BA**	**POA-PG-BA**	**POA-BA**	**POA-PG-BA**	**POA-BA**	**POA-PG-BA**
f1	0.7408	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.0560	**0.0276**	**0.0000**	**0.0000**
f2	0.1882	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0241**	**0.0237**	0.0689	0.0675
f3	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f4	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.0000	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f5	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f6	**0.0101**	**0.0000**	**0.0000**	**0.0002**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f7	**0.0004**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f8	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f9	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f10	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f11	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f12	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f13	0.2442	**0.0000**	**0.0000**	**0.0108**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f14	**0.0159**	**0.0000**	**0.0000**	0.0000	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
f15	**0.0002**	**0.0000**	**0.0000**	**0.0057**	**0.0033**	**0.0000**	**0.0000**	0.5517	**0.0000**

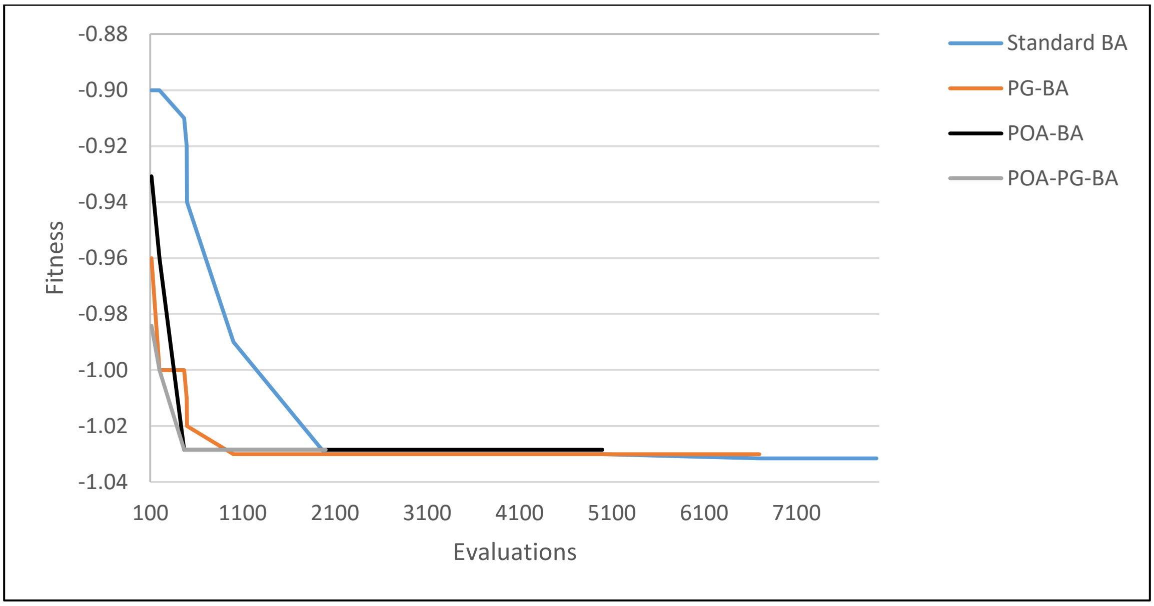

Figure 4.2: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f1*

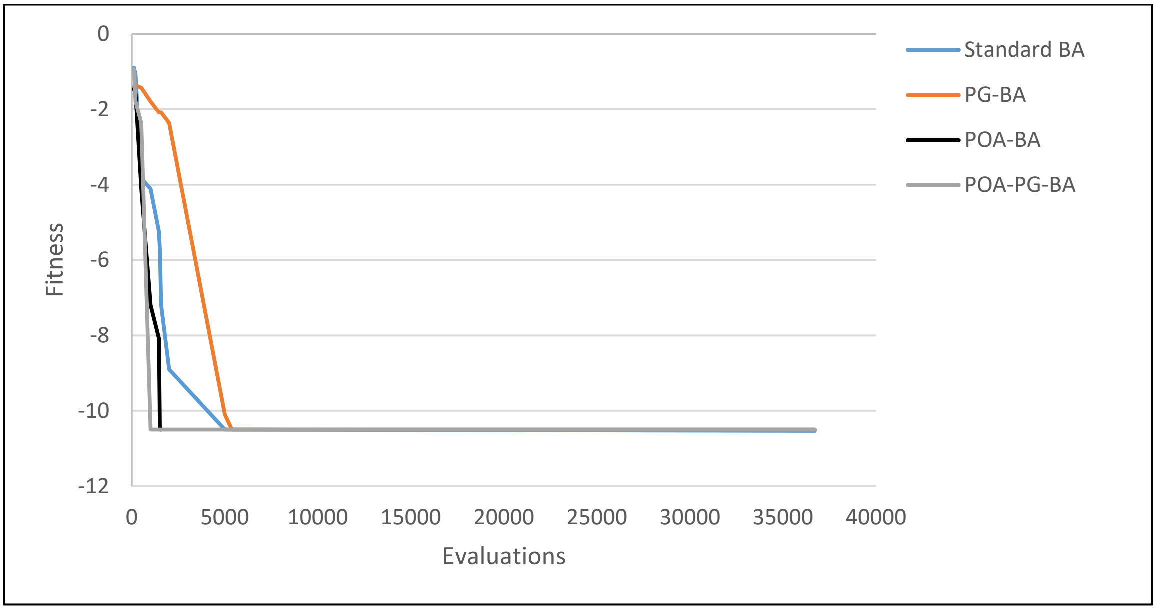

Figure 4.3: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f2*

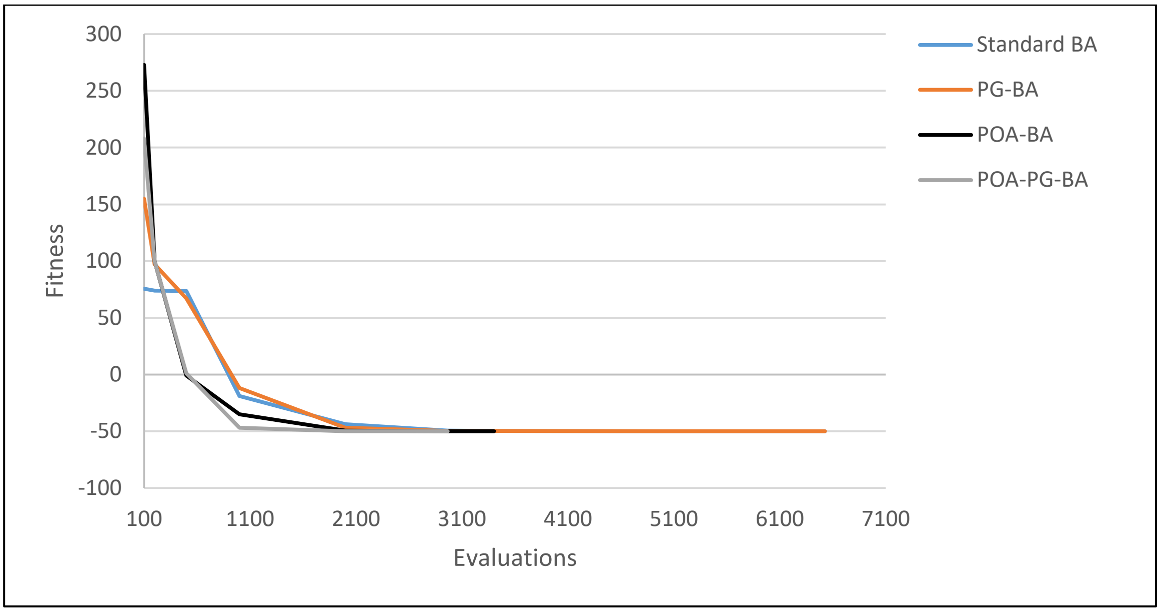

Figure 4.4: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f3*

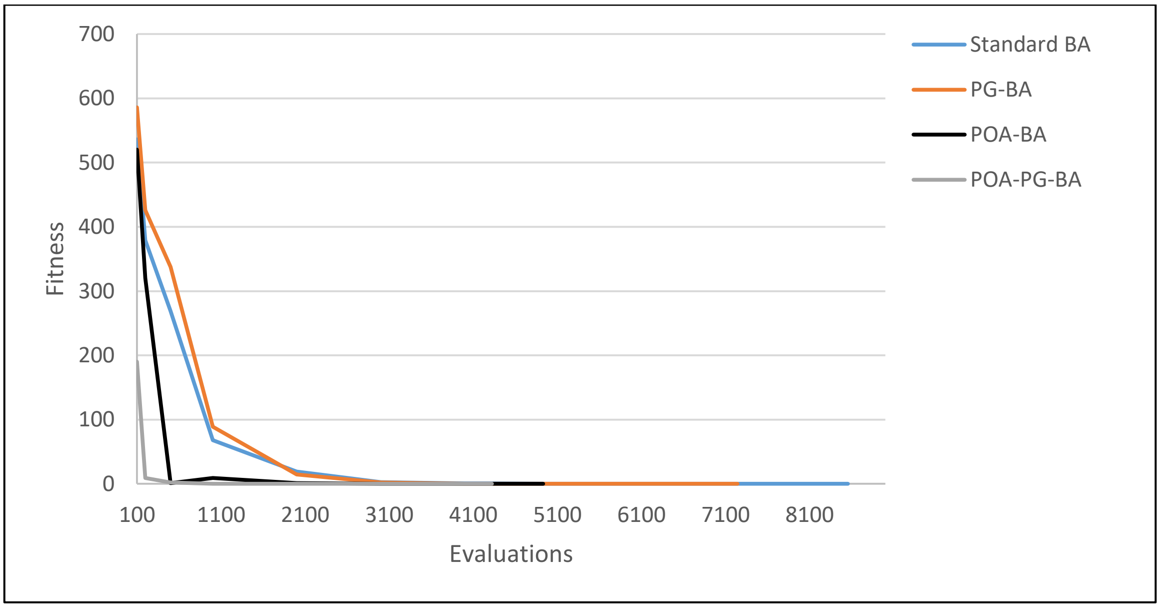

Figure 4.5: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f4*

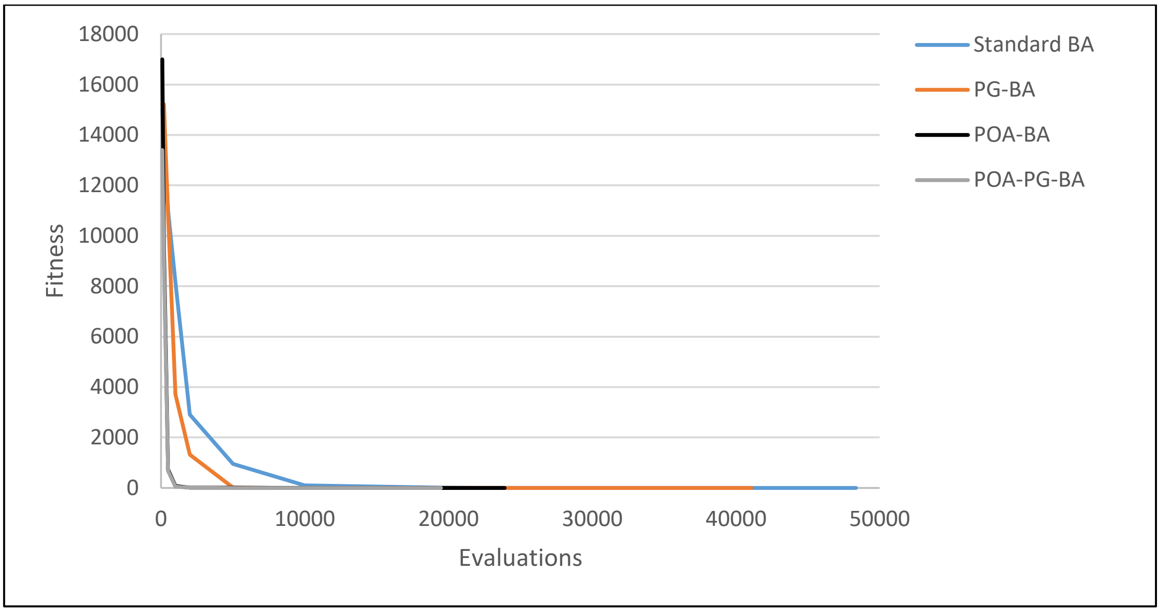

Figure 4.6: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f5*

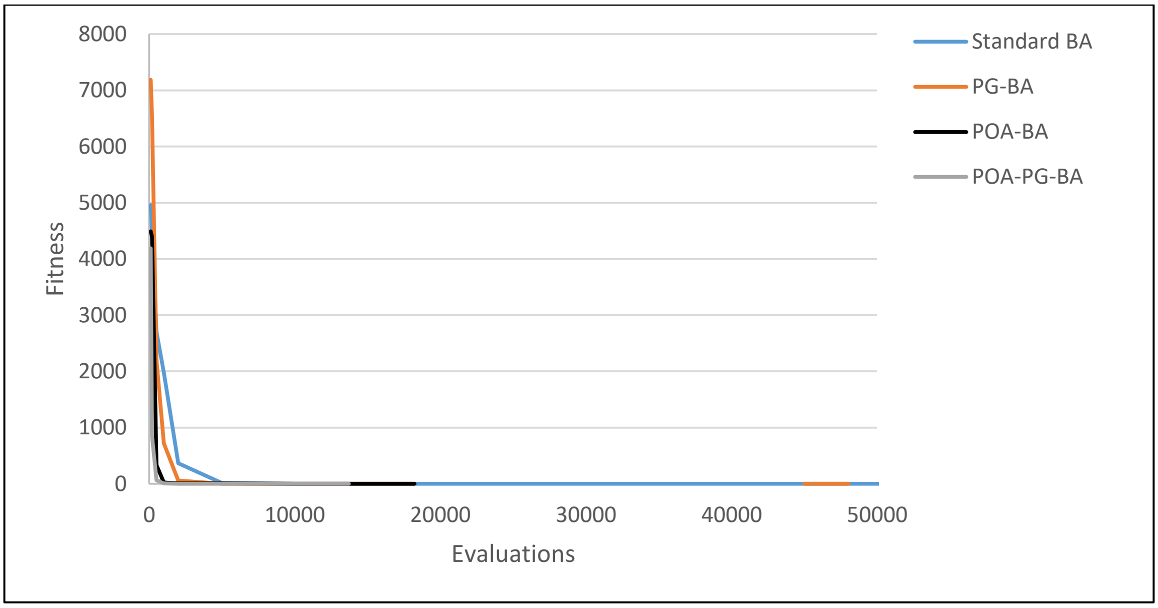

Figure 4.7: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f6*

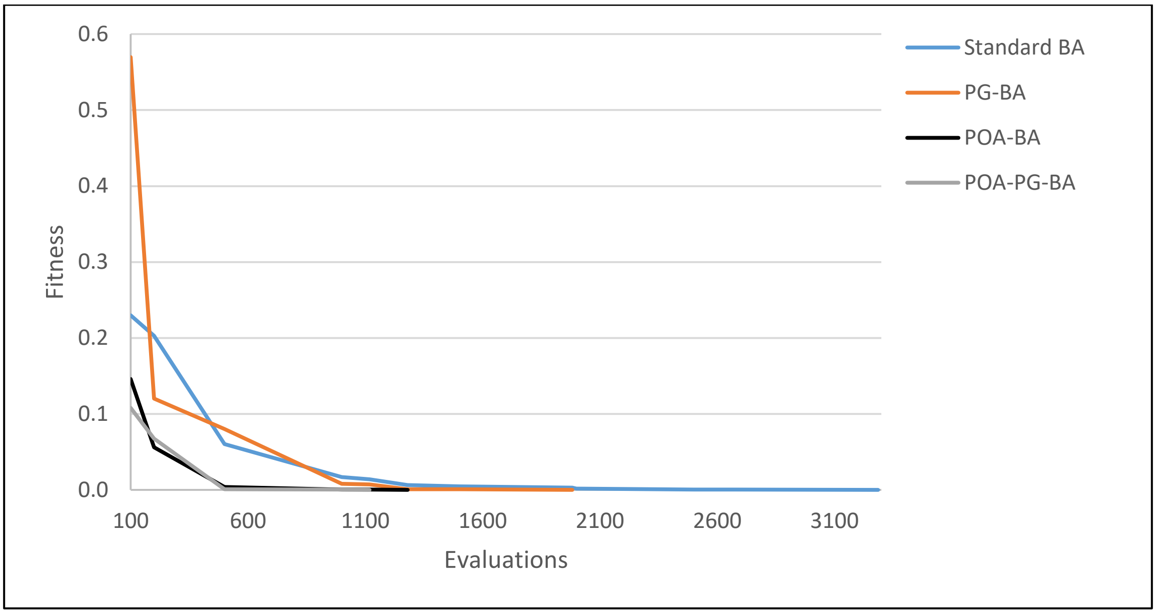

Figure 4.8: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f7*

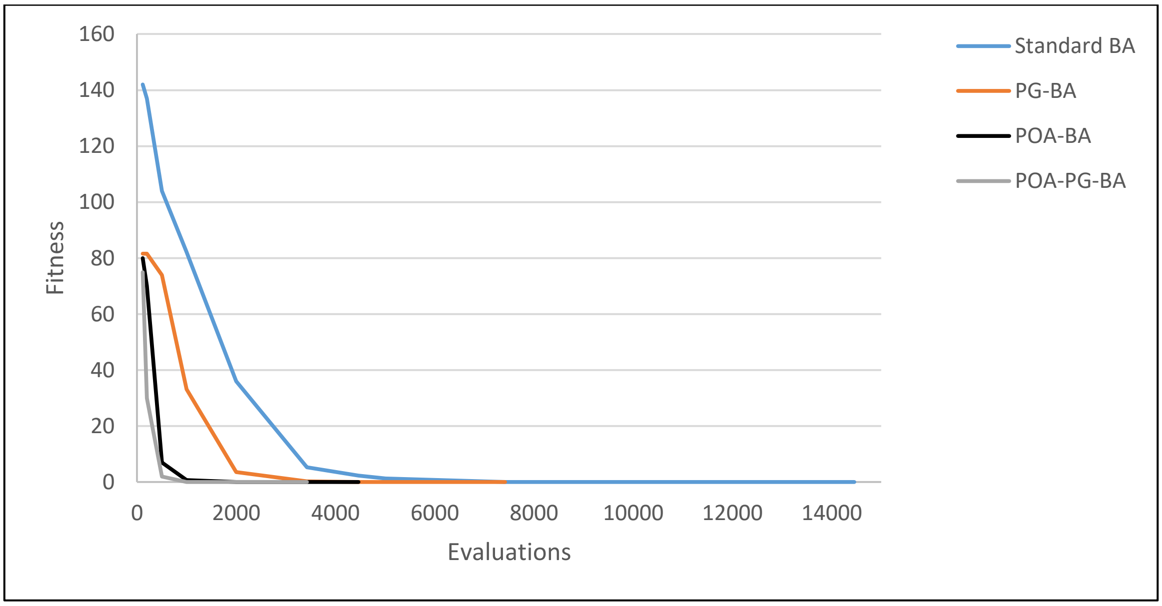

Figure 4.9: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f8*

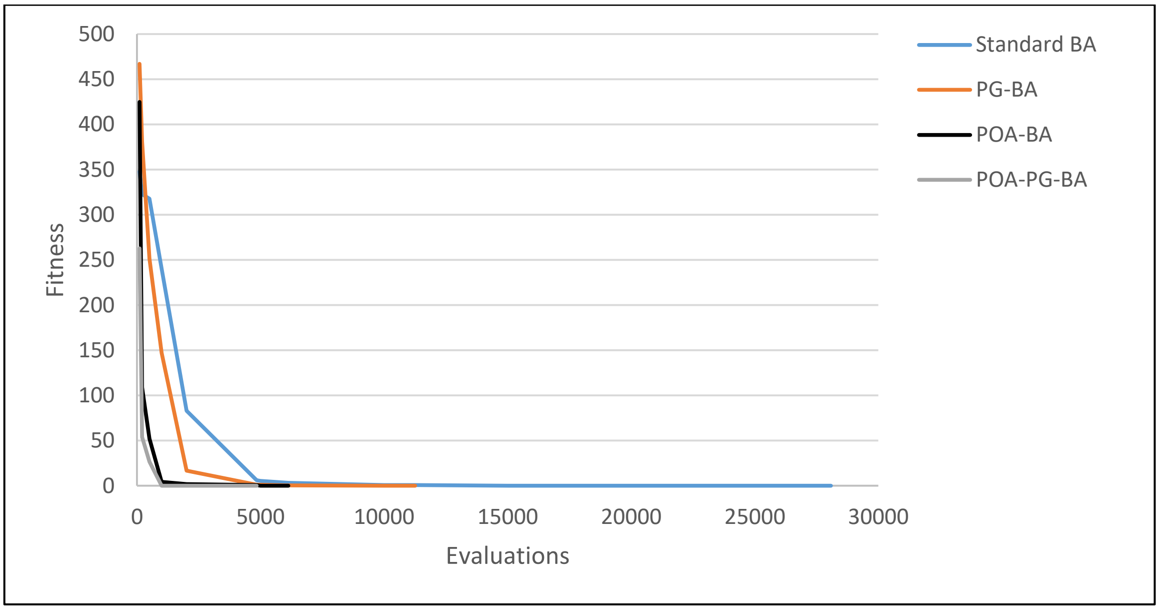

Figure 4.10: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f9*

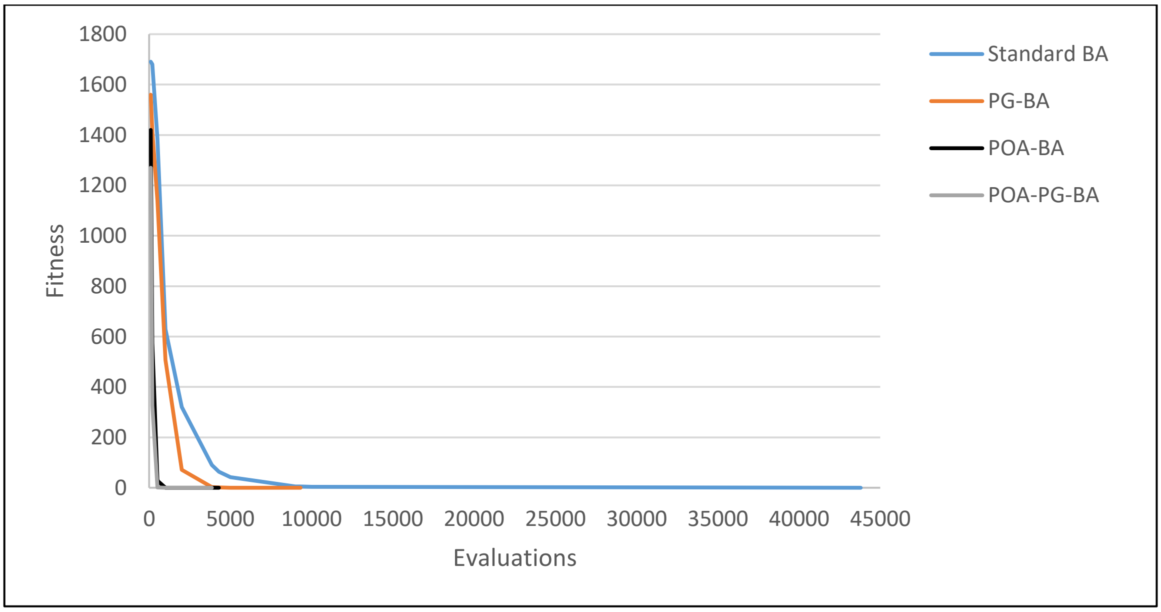

Figure 4.11: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f10*

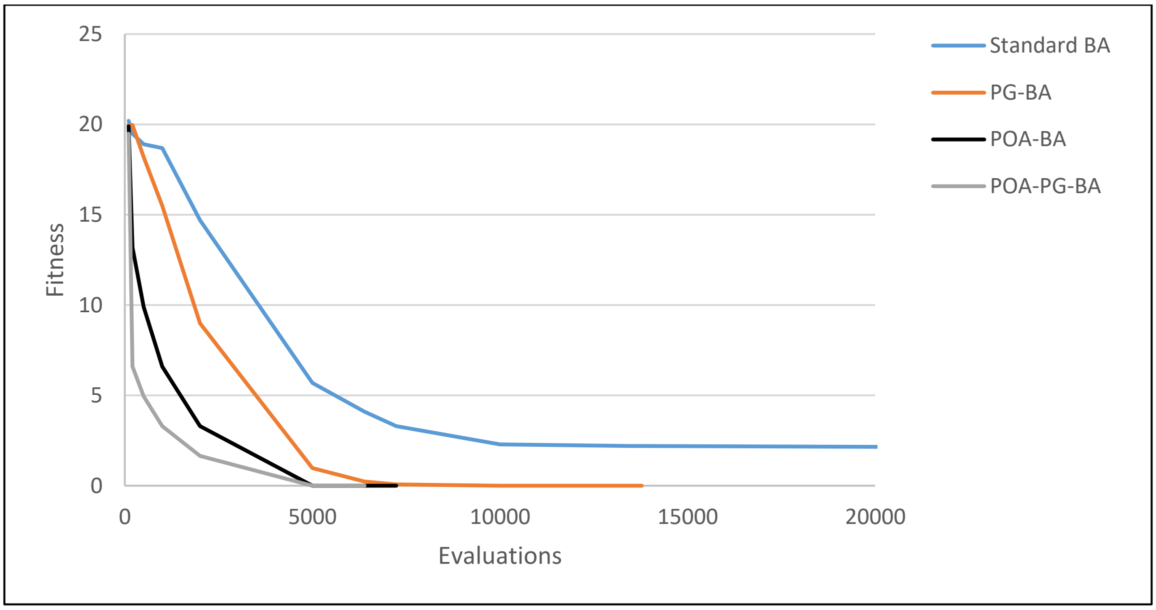

Figure 4.12: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f11*

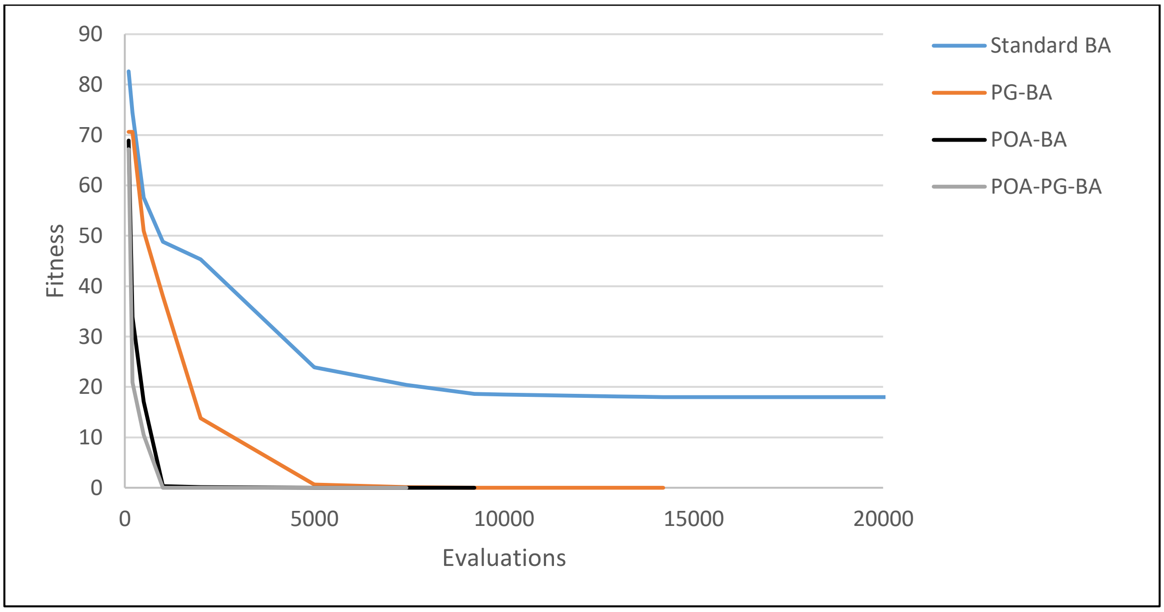

Figure 4.13: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f12*

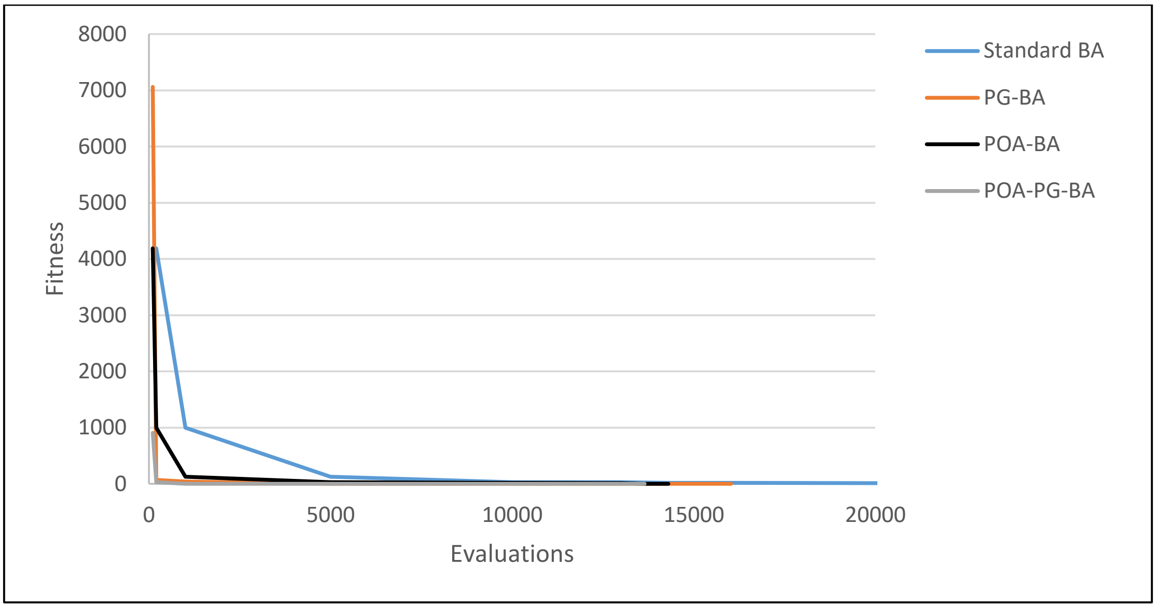

Figure 4.14: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f13*

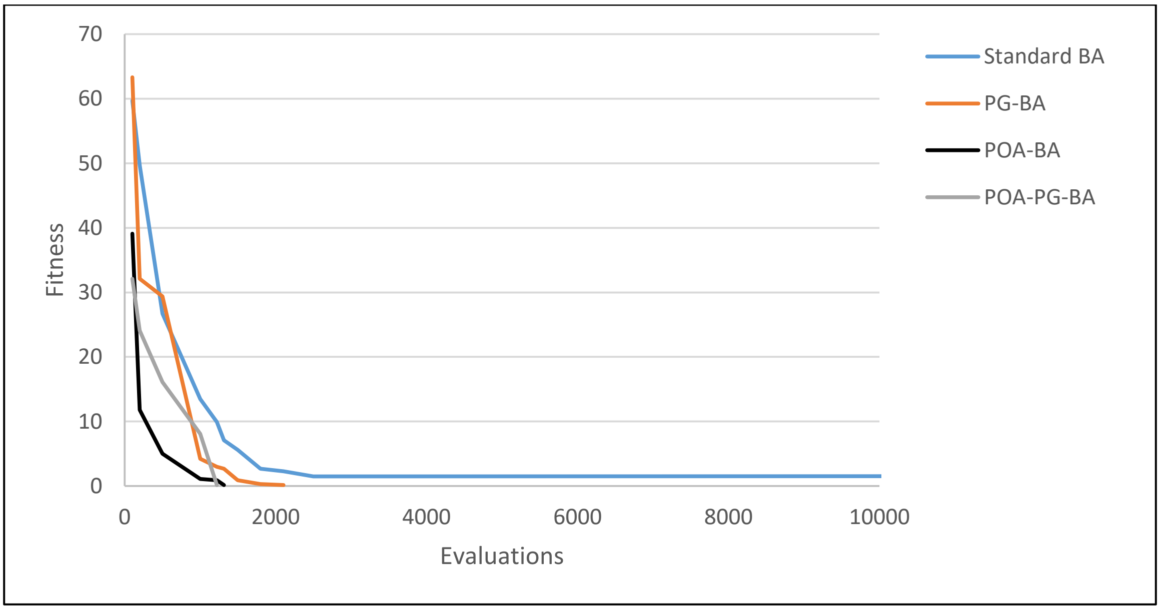

Figure 4.15: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f14*

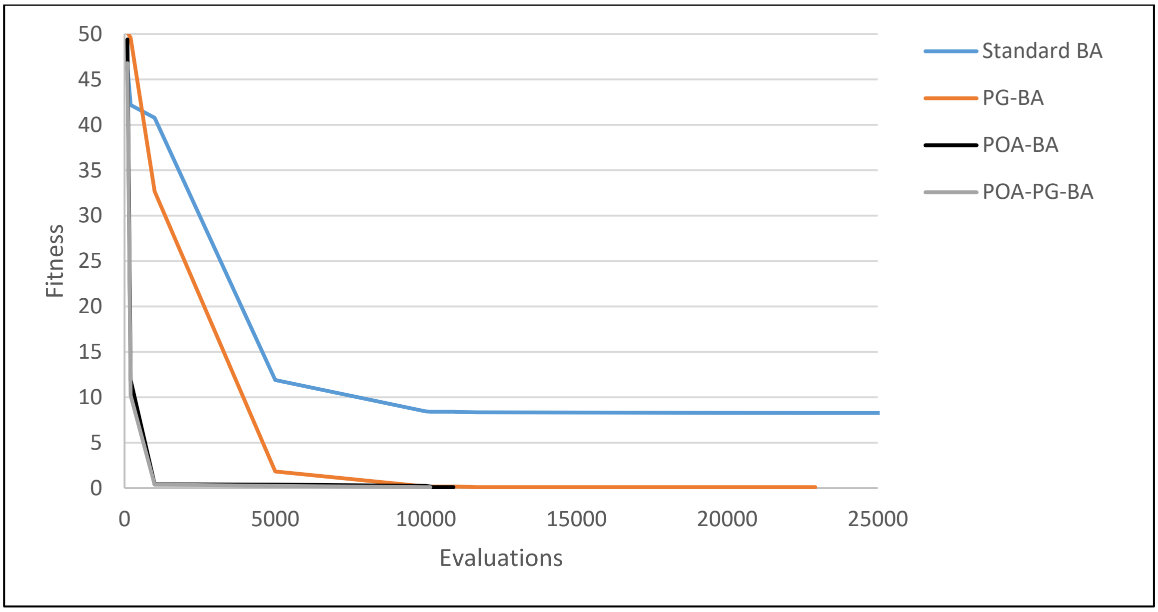

Figure 4.16: Convergence curve between POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for *f15*

4.3.2 Scalability test on POA-BA

Scalability test is also performed on both POA-BA and POA-PG-BA with the same dimensional setting as in Chapter 3. The same scalable behaviour is observed for both variants. The performance is almost the same for functions *f4*, *f7*, *f8*, *f10*, *f11*, and *f12* from 10D until 50D. *f9 Griewank* consumes a high NFE at 10D but nearly flat line after that. Meanwhile, NFE is linearly increasing with the augmented number of dimensions for *f13*. In contrast, NFE performance worsens for *f5* after 30D. *f5* is a rotated version of function Axis Parallel Hyper-ellipsoid. This means for this particular function, as the dimension increased, both algorithms have difficulty finding the global optimum as the basin of attraction is not lying along the coordinate axes.

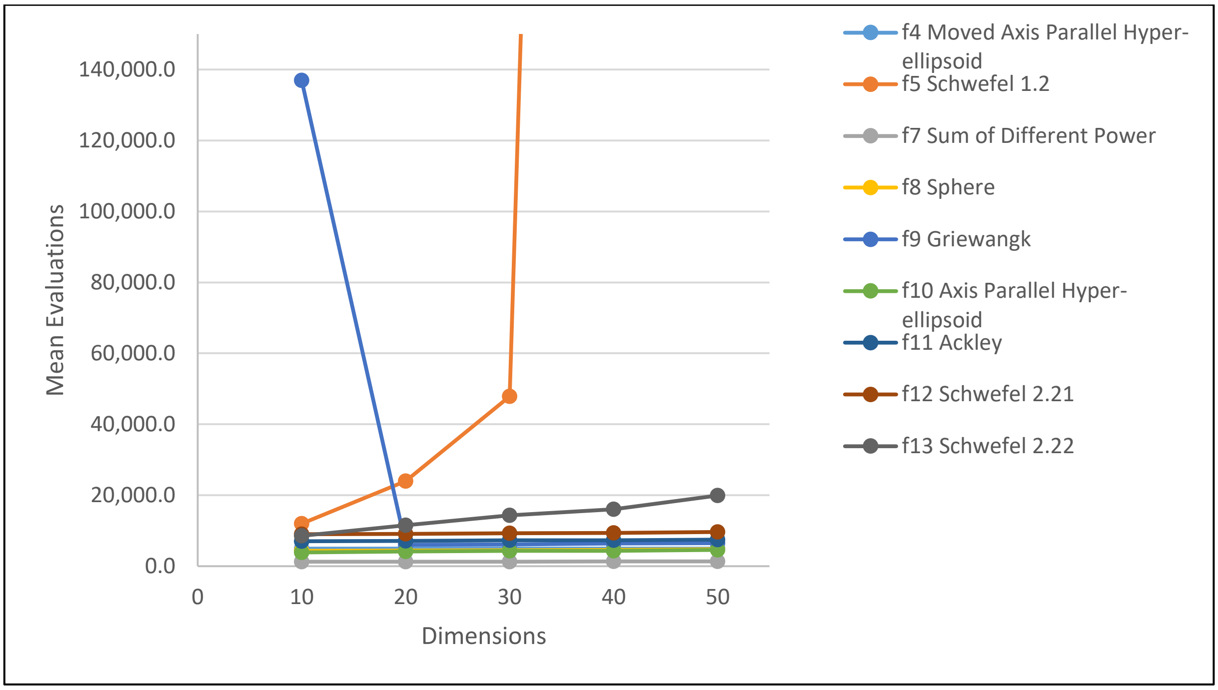

Figure 4.17: Scalability test of POA-BA

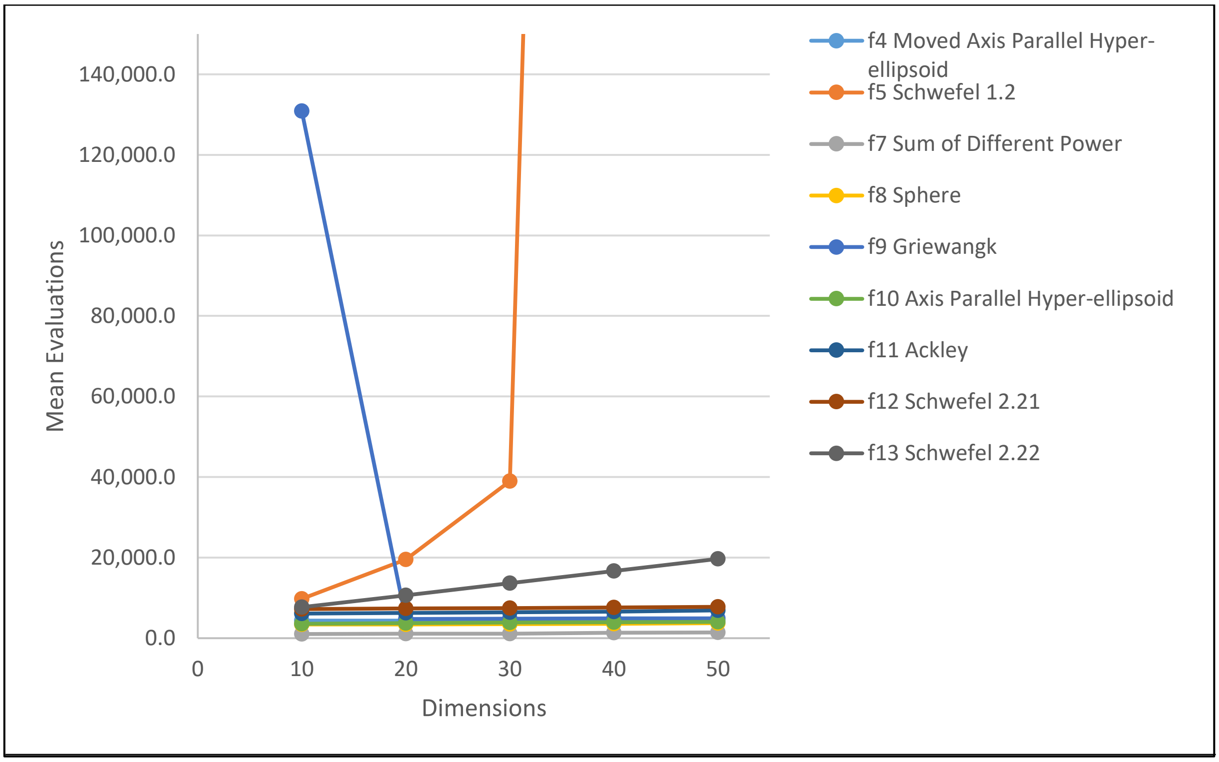

Figure 4.18: Scalability test of POA-PG-BA

4.4 Training of Feedforward Neural Network Using POA-BA

POA-BA and POA-PG-BA are also applied in the training of feedforward neural network (FNN). The same problem in Chapter 3 is used which is to find feasible weights in order to achieve minimum error in recognising pattern of Exclusive-OR (XOR) logic. Results in previous chapter show that the performance of PG-BA is nearly similar to the Standard Bees Algorithm. This experiment is performed to investigate whether comparable behaviour is displayed by both POA-BA variants.

Table 4.4 states the percentage of success out of 25 runs for POA-BA, POA-PG-BA, the Standard Bees Algorithm, and PG-BA. Both versions of POA-BA attain 100% success of finding mean squared error (MSE) less than 0.001 in all runs. Generally, all the Bees Algorithm variants achieved similar MSE as listed in Table 4.5. Hence, no statistic test is performed on this result. However, Table 4.6 shows that there are reductions on the number of evaluation taken to obtain this MSE for POA-BA and POA-PG-BA in comparison to the Standard Bees Algorithm and PG-BA due to the patch overlap avoidance mechanism.

Table 4.4: Success rate (%) comparison between POA-BA, POA-PG-BA, the Standard Bees Algorithm, and PG-BA for XOR problem

H	**Dim.**	**Standard Bees Algorithm**	**PG-BA**	**POA-BA**	**POA-PG-BA**
2	9	96	**100**	**100**	**100**
3	13	**100**	**100**	**100**	**100**
4	17	**100**	**100**	**100**	**100**
5	21	**100**	**100**	**100**	**100**
6	25	**100**	**100**	**100**	**100**
7	29	**100**	**100**	**100**	**100**
8	33	**100**	**100**	**100**	**100**
9	37	**100**	**100**	**100**	**100**
10	41	**100**	**100**	**100**	**100**
11	45	**100**	**100**	**100**	**100**
12	49	**100**	**100**	**100**	**100**
13	53	**100**	**100**	**100**	**100**
14	57	**100**	**100**	**100**	**100**
15	61	**100**	**100**	**100**	**100**
20	81	**100**	**100**	**100**	**100**
25	101	**100**	**100**	**100**	**100**
30	121	**100**	**100**	**100**	**100**

Table 4.5: MSE comparison of POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for XOR problem

H	**Dim.**	**Standard Bees Algorithm**		**PG-BA**		**POA-BA**		**POA-PG-BA**	
		Mean	**Std. Dev**	**Mean**	**Std. Dev**	**Mean**	**Std. Dev**	**Mean**	**Std. Dev**
2	9	0.0008	0.0002	**0.0007**	**0.0002**	0.0007	0.0003	**0.0007**	**0.0002**
3	13	**0.0007**	**0.0002**	**0.0007**	**0.0002**	**0.0007**	**0.0002**	**0.0007**	**0.0002**
4	17	0.0007	0.0002	0.0007	0.0002	**0.0006**	**0.0003**	**0.0006**	**0.0003**
5	21	0.0006	0.0003	**0.0006**	**0.0002**	**0.0006**	**0.0002**	**0.0006**	**0.0002**
6	25	**0.0005**	**0.0002**	0.0006	0.0002	0.0006	0.0002	0.0006	0.0002
7	29	0.0006	0.0003	0.0006	0.0003	**0.0006**	**0.0002**	**0.0006**	**0.0002**
8	33	0.0005	0.0003	**0.0005**	**0.0002**	0.0005	0.0003	0.0005	0.0003
9	37	**0.0005**	**0.0003**	**0.0005**	**0.0003**	**0.0005**	**0.0003**	**0.0005**	**0.0003**
10	41	**0.0004**	**0.0003**	0.0005	0.0002	0.0005	0.0002	0.0005	0.0002
11	45	0.0005	0.0003	0.0006	0.0003	0.0005	0.0003	**0.0005**	**0.0002**
12	49	0.0005	0.0003	**0.0004**	**0.0003**	**0.0004**	**0.0003**	**0.0004**	**0.0003**
13	53	0.0004	0.0003	0.0004	0.0003	0.0004	0.0003	**0.0004**	**0.0002**
14	57	**0.0004**	**0.0003**	**0.0004**	**0.0003**	**0.0004**	**0.0003**	**0.0004**	**0.0003**
15	61	0.0004	0.0003	**0.0003**	**0.0002**	**0.0003**	**0.0002**	0.0004	0.0002
20	81	0.0004	0.0003	0.0004	0.0003	0.0004	0.0002	**0.0003**	**0.0003**
25	101	0.0003	0.0003	**0.0002**	**0.0002**	**0.0002**	**0.0002**	**0.0002**	**0.0002**
30	121	0.0003	0.0003	**0.0002**	**0.0002**	**0.0002**	**0.0002**	**0.0002**	**0.0002**

Table 4.6: NFE comparison of POA-BA, POA-PG-BA, Standard Bees Algorithm, and PG-BA for XOR problem

H	Dim.	Standard Bees Algorithm				PG-BA				POA-BA			POA-PG-BA	
		Mean	Std. Dev	POA-BA	POA-PG-BA	Mean	Std. Dev	POA-BA	POA-PG-BA	Mean	Std. Dev	POA-PG-BA	Mean	Std. Dev
				p -value	*p* -value			*p* -value	*p* -value			*p* -value		
2	9	7,928.2	22,446.9	0.4350	0.3651	4,506.3	9,291.3	0.9132	0.7143	4,261.6	6,189.7	0.7304	**3,794.5**	**2,659.6**
3	13	2,230.0	1,439.2	**0.0178**	**0.0045**	1,568.0	294.1	0.4804	**0.0192**	1,505.3	328.3	0.1049	**1,347.3**	**347.4**
4	17	1,758.0	529.1	**0.0004**	**0.0000**	1,440.0	283.6	0.0785	**0.0031**	1,292.4	296.7	0.1919	**1,178.9**	**309.5**
5	21	1,517.0	308.2	**0.0025**	**0.0000**	1,344.0	333.2	0.3297	**0.0111**	1,261.2	256.4	0.0613	**1,123.5**	**251.6**
6	25	1,430.0	312.4	**0.0000**	**0.0000**	1,252.0	254.6	**0.0213**	**0.0093**	1,089.1	228.6	0.5414	**1,043.9**	**287.6**
7	29	1,482.0	408.6	**0.0002**	**0.0000**	1,200.0	251.9	0.1256	**0.0014**	1,086.2	264.2	0.1093	**974.8**	**216.2**
8	33	1,264.0	270.0	**0.0000**	**0.0000**	1,210.0	379.5	**0.0033**	**0.0008**	921.3	272.3	0.4086	**849.2**	**335.9**
9	37	1,114.0	274.1	**0.0013**	**0.0026**	1,096.0	267.9	**0.0023**	**0.0043**	873.9	218.1	0.6769	**840.6**	**331.9**
10	41	1,146.0	289.1	**0.0013**	**0.0000**	1,096.0	275.2	**0.0058**	**0.0006**	866.0	288.4	0.3436	**783.2**	**322.7**
11	45	1,106.0	321.4	**0.0034**	**0.0002**	996.0	244.4	0.0514	**0.0031**	860.8	233.9	0.2482	**782.6**	**239.1**
12	49	1,214.0	347.5	**0.0000**	**0.0000**	948.0	295.1	0.1845	0.0110	842.5	257.7	0.1512	**729.5**	**289.2**
13	53	1,246.0	457.4	**0.0000**	**0.0000**	972.0	176.6	**0.0030**	**0.0000**	760.8	287.4	0.3300	**691.5**	**203.4**
14	57	1,040.0	287.8	**0.0000**	**0.0000**	842.0	326.1	0.1401	0.0605	720.9	237.8	0.5741	**680.7**	**263.8**
15	61	1,126.0	228.6	**0.0000**	**0.0000**	860.0	213.8	**0.0029**	**0.0005**	656.6	243.6	0.5430	**614.1**	**246.9**
20	81	918.0	234.5	**0.0005**	**0.0000**	832.0	272.8	**0.0194**	**0.0012**	638.4	292.6	0.5069	**589.1**	**224.2**
25	101	860.0	256.3	0.0027	0.0002	714.0	314.8	0.3405	0.0916	637.8	239.8	0.3889	**580.4**	**226.8**
30	121	798.0	348.9	**0.0450**	**0.0055**	768.0	256.3	0.0502	**0.0037**	602.1	323.7	0.4875	**543.7**	**263.4**

As in the previous experiment, POA-PG-BA has fewer NFE compared to POA-BA. However, the percentage difference of NFE between the algorithms is on average a mere 8.59%. Table 4.7 show the percentage of NFE reduction for all number of hidden nodes considered. *T-test* also shows that POA-BA and POA-PG-BA perform statistically the same for this experiment. In comparison to PG-BA, the performance of POA-BA is only statistically different mostly on *H=8* and beyond. Whereas performance comparison of POA-PG-BA is more consistently significant.

Table 4.7: Percentage improvement of mean evaluation for POA-BA and POA-PG-BA for XOR problem

H	Dim.	POA-BA		POA-PG-BA		
		Standard BA	PG-BA	Standard BA	PG-BA	POA-BA
2	9	46.25	5.43	52.14	15.79	10.96
3	13	32.50	4.00	39.58	14.08	10.50
4	17	26.48	10.25	32.94	18.13	8.78
5	21	16.86	6.16	25.94	16.41	10.92
6	25	23.84	13.01	27.00	16.62	10.50
7	29	26.70	9.48	34.23	18.77	3.90
8	33	27.11	23.86	32.82	29.82	14.99
9	37	21.56	20.27	24.54	23.31	3.81
10	41	24.44	20.99	31.66	28.54	1.94
11	45	22.17	13.57	29.24	21.42	9.08
12	49	30.60	11.13	39.91	23.05	13.41
13	53	38.94	21.73	44.50	28.86	10.53
14	57	30.68	14.38	34.55	19.16	4.08
15	61	41.69	23.65	45.47	28.60	6.48
20	81	30.46	23.27	35.83	29.20	7.72
25	101	25.83	10.67	32.52	18.72	14.75
30	121	24.55	21.60	31.86	29.20	3.61

Figures 4.19 and 4.20 illustrate the relationship between the number of hidden nodes with MSE and NFE. For MSE, the value is linearly declining at approximately *H=12* before stabilising. In the meantime, at *H=3* NFE is slowly decreasing before almost plateauing at *H=15*.

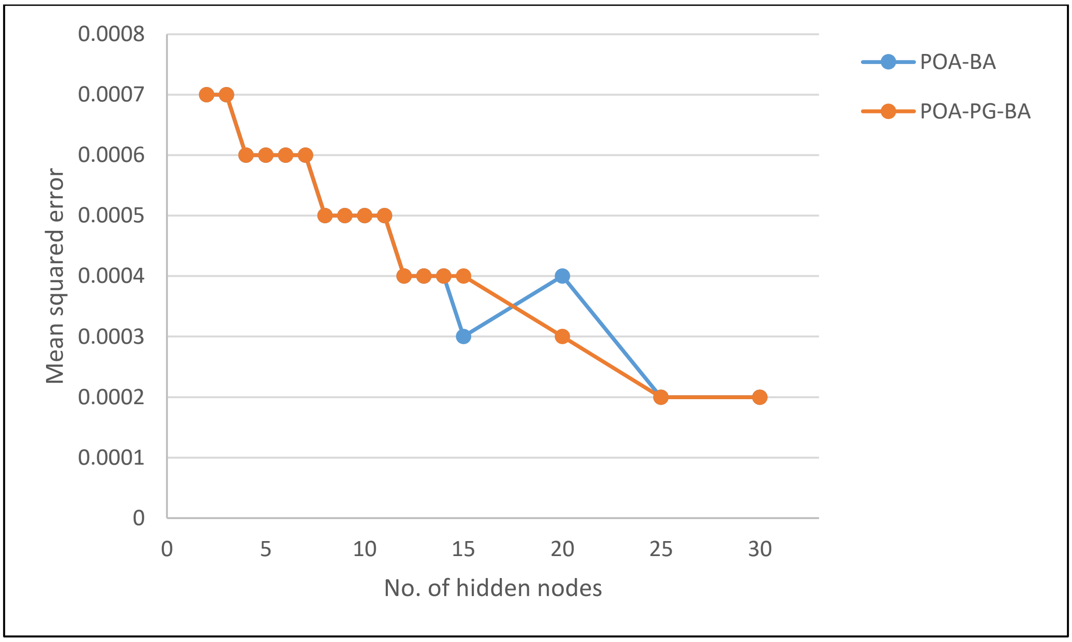

Figure 4.19: Effect of number of hidden nodes to MSE for POA-BA and POA-PG-BA

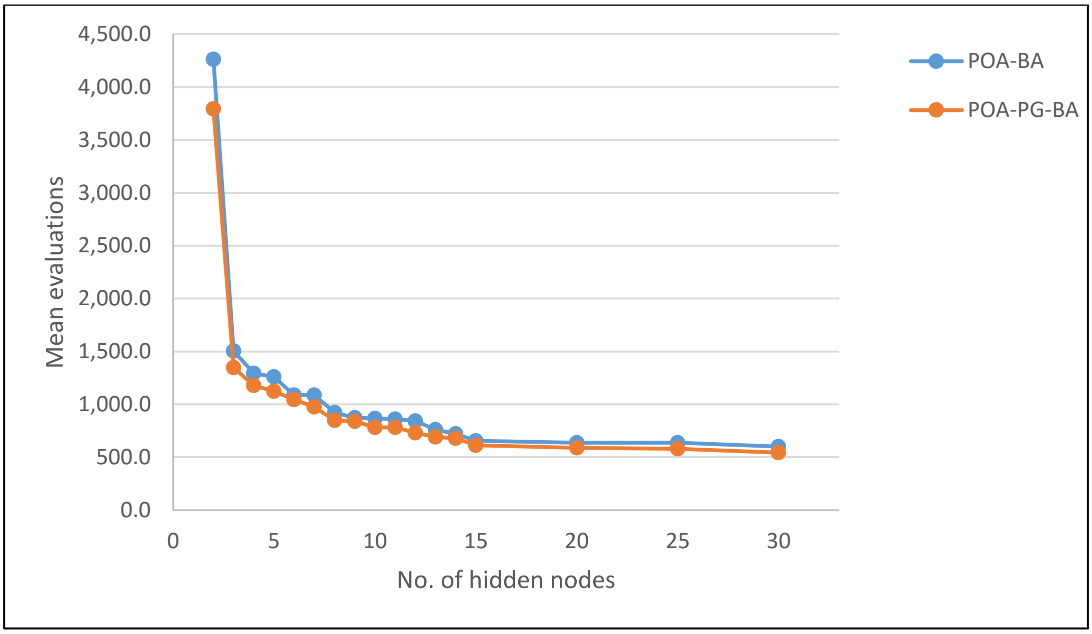

Figure 4.20: Effect of number of hidden nodes to NFE for POA-BA and POA-PG-BA

4.5 Conclusions

This chapter introduced the strategy of reducing the formation of overlapping patches in the Bees Algorithm by the used of fixed-size temporary memory akin to Tabu Search (TS). The memory is updated every cycle by recording previously visited and abandoned solutions (plus their neighbourhood boundary) with the priority on the low performers. New solutions are compared to the ones in the memory. If their boundary (patch) does not overlap, they are allowed to survive. If the opposite, their solution's information entered memory and traded places with the memory's member of higher fitness. This minimises redundant search on an already sought area especially if they are on the similar region of attraction. Consequently, the search space contracts in size permitting fewer function evaluations required to obtain the global optimal.

Two versions of the Patch Overlap Avoidance Bees Algorithms were developed each with a different neighbourhood search strategy. POA-BA used standard neighbourhood search of the Bees Algorithm, while POA-PG-BA utilised the pseudo-gradient method. Their performances were evaluated against the Standard Bees Algorithm, PG-BA, and the same swarm-based algorithms used in previous chapter of similar benchmark functions. Results show that both variants of POA-BA boosted the global search as the next scout bees cannot inhibit the same spot despite the idea behind this scheme is initially to reduce the chances of repeated exploitation at the neighbourhood level, thus improving performance especially on the low dimension. Exclusive-OR pattern recognition using feedforward neural network trained by the modified algorithm were also executed together with statistical analysis on each experiment.

CHAPTER 5

AN EXTENDED BEES ALGORITHM (EBA) TO FIND MULTIPLE OPTIMAL SOLUTIONS

5.1 Preliminaries

There is a growing trend for optimisation algorithms that are able to detect multiple global solutions as well as a number of useful local extremum in one run. Most population-based optimisers use niching techniques for diversity maintenance of feasible solutions in order to identify the entire optima. These methods introduce a niching parameter that need to be tuned alongside the algorithms' own parameters.

The development of POA-BA in the previous chapter has prompted the researcher to investigate a new strategy that only allows the formation of a single patch for every peak in the search landscape. With this extension, the Bees Algorithm will be able to maintain a set of diverse and multiple optimal throughout the optimising process which counteract the pressure of global selection scheme (i.e. the algorithm can only converge to one global optimum) without additional niching parameter.

This chapter introduces the Hill-Valley mechanism (Ursem, 1999) and how incorporating this approach has changed the way the Standard Bees Algorithm operates. Experimentation takes place to investigate whether this variant for Multimodal Optimisation (MMO), the Extended

Bees Algorithm (EBA) is capable to locate multiple peaks in multimodal numerical functions. Before that, definition of niching and its various techniques shall be presented.

5.2 Niching: Definition and Techniques

The idea of creating niches or species (these terms are used interchangeably) in optimisation algorithms stems from the field of ecology. In the natural ecosystem, distinct species exist with different roles that share and compete for the same resources (Thomsen, 2004; Brits et al., 2007; Engelbrecht, 2007; Li, 2007; Qing et al., 2008; Liang and Leung, 2011). Analogous to optimisation problem, niche refers to the location of each optimum in the search space while resources of the niche correspond to the fitness function. Niching can also be used to find Pareto optimality in multi-objective problem, and tracking changes in dynamic optimisation.

Niching-based algorithms can be classified depending on how the solutions are attained:

i) Sequential/iterative/temporal

   The same algorithm is run several times avoiding the area where convergence has occurred until the satisfied number of optimal has been achieved.

ii) Parallel/spatial

   Multiple solutions can be found in a single run as the population is divided into subpopulations that evolve in parallel and search different regions of the solution space.

Since sequential-type algorithms consume more time and have relatively limited performance, most algorithms that attempt to find multiple optimal points used the latter (Li et al., 2010). Several niching techniques were formulated primarily for Evolutionary Algorithms (EAs) especially Genetic Algorithm (GA) since interest in this research area grew. The earliest and simplest being crowding where a fixed percentage of the population, defined by niche parameter, crowding factor, *CF* is reproduced and killed each generation to replace the most similar individuals (the same niche). This is based on Hamming distance if binary numbers are used or Euclidean distance for real value. *CF* needs to be tuned to achieve a desirable result, and Thomsen (2004) said that by using *CF* equal to the size of the population has eliminated selection error associated with this technique. Nevertheless, Qing et al. (2008) found that genetic drifts still exist which reduce the chance of finding all optima. It is worth mentioning that there are two types of crowding; deterministic and probabilistic based on the selection rule used (Das et al. 2011; Liang and Leung, 2011).

Another popular niching method is fitness sharing in which sharing function adjusts the fitness of individuals in the same subpopulation. It degrades an individual's objective function due to the presence of nearby similar individuals. This technique is better at diversity preservation than crowding but due to sharing scheme enforcement, good solutions can be lost as individuals move away from peaks thus reducing the chances of local convergence (Thomsen, 2004; Liang and Leung, 2011). This technique can also be applied to iterative-based multimodal algorithms in which the stretching of fitness can prevent individuals from exploring the area where solutions have already been found (Beasley et al., 1993).

Both techniques reduce the effect of elitism employed in EAs in favour of diversity maintenance. Even so, elitism is needed in order to find the global optima quickly. One niching method that tried to balance between elitist strategy and preservation of diversity is clearing. Unlike sharing, resources are shared between highly fit individuals of each subpopulation while the other individuals' fitness is set to 0 thus reducing the computational complexity. Still, dissimilarity threshold based on distance need to be tuned by the user (Singh and Deb, 2006; Das et al., 2011; Liu et al., 2011).

In addition, recently developed species conservation used dominant individuals as species seed, and species containing similar individuals is built within the user-defined radius with species seed at the centre. Species seeds found in the current generation are conserved by moving them to the next generation enabling their survival after reproduction took place (Singh and Deb, 2006; Li et al. 2010; Das et al., 2011; Liang and Leung, 2011; Liu et al., 2011).

Nonetheless, all these methods need some niche parameters to be set by the user thus reducing the robustness of the algorithms especially in higher dimension problems. Few authors attempted to make the niche parameters adaptive such as efforts by Miller and Shaw (1995), Nickabadi et al. (2008), and Shir et al. (2010). By making certain parameters adaptive or more dynamic, a few less-sensitive parameters are introduced.

State-of-the-art niching methods avoid the utilisation of any niche parameters. By adapting the Fitness-distance-Ratio based Particle Swarm Optimisation (PSO), Li (2007) used the Fitness Euclidean-distance Ratio to create distinctive niches. On the other hand, Liu et al. (2011) engaged the near-neighbour effect as attractor and repellent to pull and push particles into locating multiple optima rapidly and accurately. They based their work on the Force-imitated PSO variant. Other methods used different PSO's topologies to create niche without extra parameters such as work by Li (2010) that employed ring topology.

Additionally, there are also algorithms devised by techniques to solve different domain of optimisation problems. Deb and Saha (2010) converted single objective multimodal into a bi-objective optimisation problem so that all optimal solutions became members of the resulting non-dominant Pareto-optimal set. They were the first to attempt constrained multimodal problem. Clustering algorithms have also been employed to solve MMO (Liu et al., 2011; Das et al., 2011).

Moreover, just like hybridisation between two or more algorithms was done to take each algorithm advantages for improving their performance, hybridisation can also occur between niching techniques. One example is Qing et al. (2008) which employed the hybrid of clustering technique to eliminate genetic drift in GA with crowding to form multiple niches. Meanwhile, Yu and Suganthan (2010) used an ensemble niching consist of clearing and the special-version of crowding, restricted tournament selection.

5.3 The Hill-Valley Mechanism

Ursem (1999) is one of the earliest MMO algorithms that did not implement any niche parameter. Instead he used topology-based scheme to divide population into species. He named the method Hill-Valley detection mechanism. This mechanism works by generating a line between two points (let's say x_p and x_q) that needs to be compared. A number of points are chosen in between these two points, and their fitness is calculated. The samples points used in this method are {0.25, 0.5, 0.75}. The first interpolated point is one quarter of the way along the line bisecting x_p and x_q, and so on. If any of the sample points has a fitness value smaller than the minimum of two compared points, thus there exists a valley between the two tested points (different peaks). Otherwise, they belong to the same hill (peak). The following is the steps for the mechanism:

1. $i = 1$;
2. found = FALSE;
3. while i < samples.length and not *found* do
4. for $j = 1$ to d do
5. $x_{interior}[j] = x_p + (x_q. x_p)$. sample[$j$];
6. end for
7. if $f(x_{interior}) < \min\{f(x_p), f(x_q)\}$ then
8. *found* = TRUE;
9. end if
10. $i = i + 1$;
11. end while
12. return *found*;

Figure 5.1 explains this operation by an example of a one-dimensional space. Between positions b_1 and b_2, sample points i_1 until i_3 all have greater fitness than the fitness of the comparing points. Thus inferring that b_1 and b_2 belong to the same peak. In contrast, a sample point i_6 in between b_3 and b_4 has the smallest fitness value in comparison which surmising the existence of a valley somewhere in the middle of these locations. Hence, b_3 and b_4 are located at different hill.

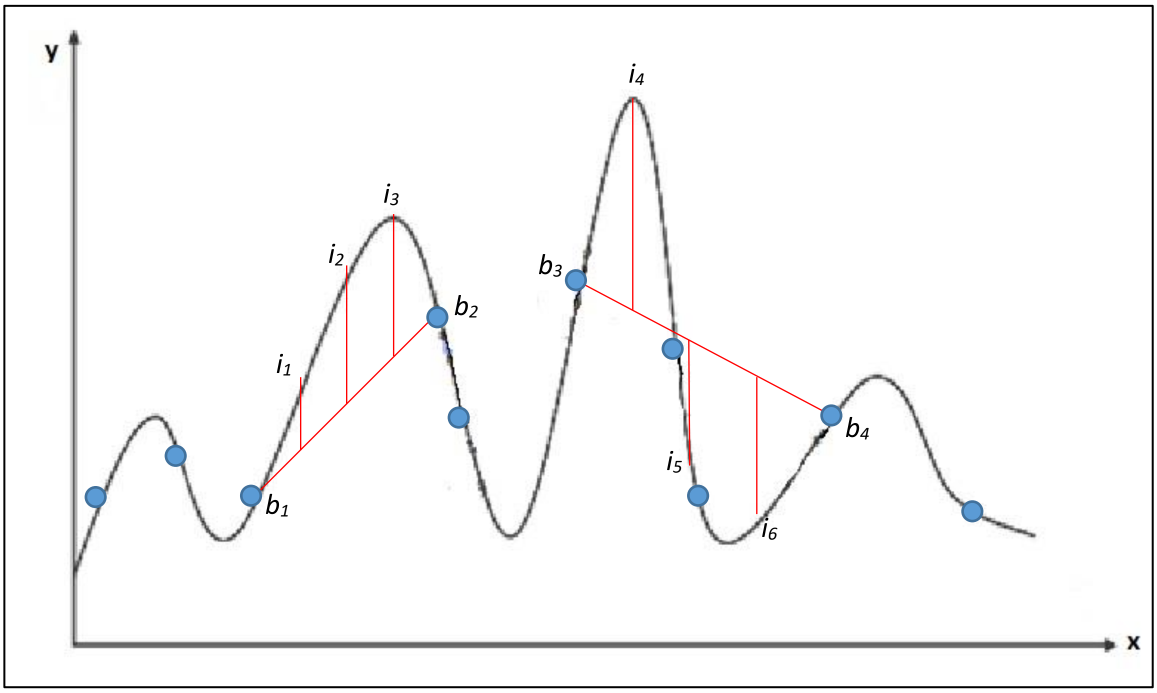

Figure 5.1: Example of Hill-Valley detection in 1D

This approach eliminates the redundancy between individuals approaching the same peak. The initial study the involved simulation process of migration between nations in multinational EAs is done by using this mechanism in MMO. Even though the original author suggested that three sample points are already adequate to successfully differentiate between peaks and valleys, users can increase these numbers to maximise the potential of locating more peaks but it is at the cause of increasing the number of function evaluations.

Inspired by this, other researchers used the Hill-Valley detection mechanism instead of distance-based niche parameters with crowding (Thomsen, 2004), and species conservation (Stoean et al., 2010; Shen and Xia, 2012). In a paper co-written by the author of this thesis, the Bees Algorithm also attempted to deploy this scheme in MMO so that each patch converged to different peaks (Zhou et al., 2015). Still, adaptive field radius, a niche parameter is used to merge and split patches depending on their distance with each other.

5.4 Extended Bees Algorithm (EBA)

Intrinsic mechanism in the Bees Algorithm when forming several patches can be exploited to locate multiple global and local optimal in a single run. This work seeks to extend the algorithm capability in terms of preservation mechanism so that the found solutions over generations can be maintained until the algorithm terminates. This is done without the need to apply any niche parameter.

In this work, there will be no distinction assumed between elite and best sites. Thus, the population of bees and the number of selected sites will be made adaptive. Therefore, the remaining parameters need to be tuned by the users are as follows:

i. Initial number of scout bees, *ins*

ii. Number of recruited bees for selected sites, *nr*

iii. Initial size of neighbourhood, *ngh*

iv. Stagnation limit, *stlim*

v. Termination criteria

The proposed algorithm starts with initial *ins* scout bees placed randomly in the search space as shows in Figure 5.2. Assuming maximisation problem, bees are ranked in decreasing order according to their fitness after evaluation. The site corresponding to the bee with the highest fitness is automatically selected as one of the selected sites. The rest of the selected sites will be determined by using the Hill-Valley detection mechanism.

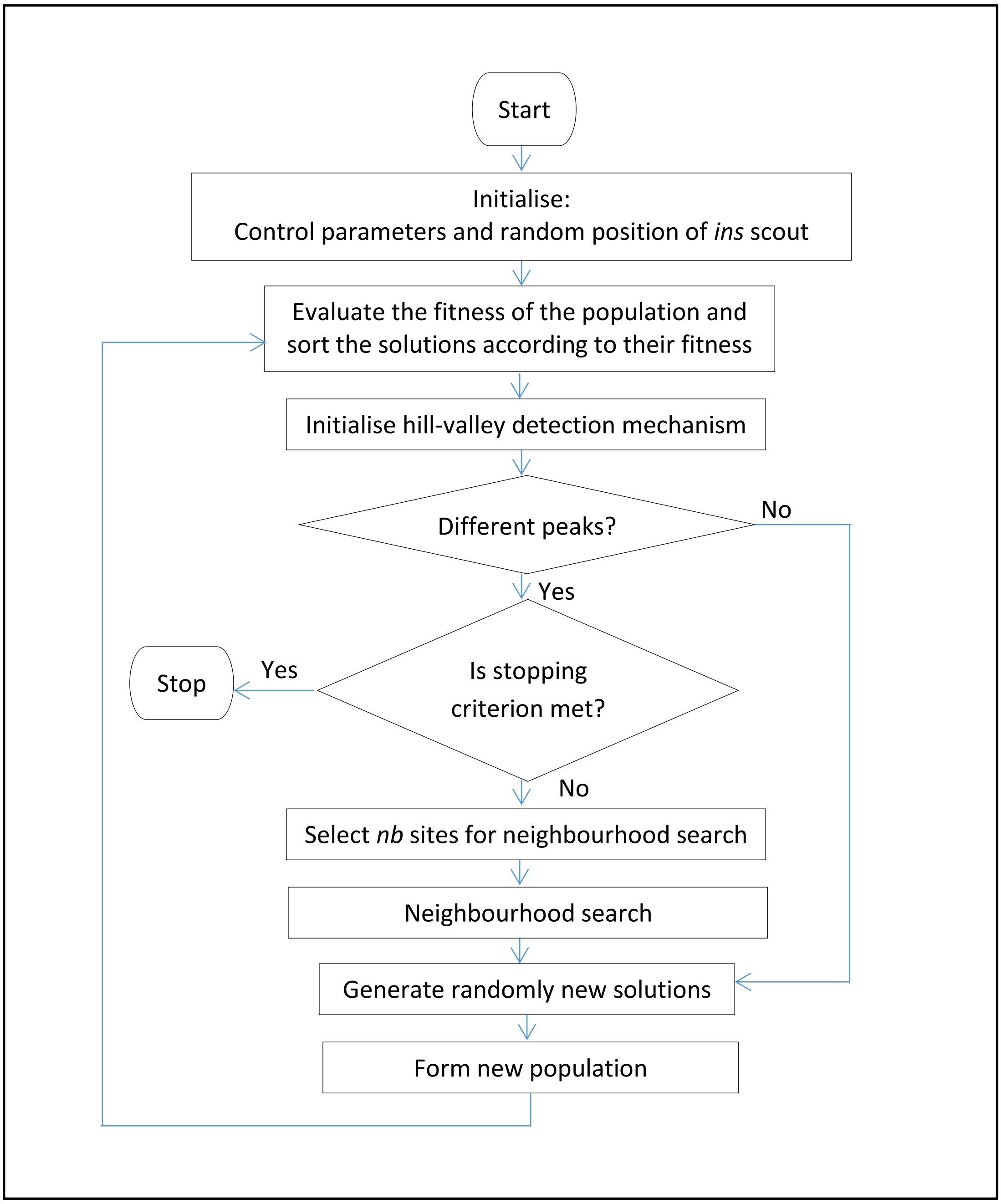

Figure 5.2: Flow chart of the main phase of EBA

Using this technique, the highest fitness point is compared one by one with the rest of the points found by scout bees to evade any points that are in the same peak. Next, it will be marked in the memory as evaluated to avoid being evaluated again. Then, the next highest surviving point is compared with the rest of the outstanding points again to avoid similarity. The process continues until all remaining points have been tested. Sites of the survivors are then selected and thus form patches (niches) by recruiting bees to their site for further exploitation. Initially, two versions of EBA are developed with each one different in terms of the neighbourhood strategy employed. EBA uses the standard neighbourhood search while EBA-PG utilises the pseudo-gradient neighbourhood search. This is done to investigate whether the Bees Algorithm with the Hill-Valley is able to stand on its own or needing help in the neighbourhood search. In general, the algorithm only allows the survival of the fittest in each peak with non-overlapping patches following different peaks. This will ensure global and local convergence in multiple peaks in a case where the numbers of peaks are not known a priori.

After that, each patch is evaluated and the bees are sorted according to their fitness as shows in Figure 5.3. Then, the Hill-Valley detection mechanism is re-initialised for every patch to determine whether there are bees on different peaks. This is important if the peaks are closed to each other in the search landscape. Bees with the highest fitness of each peak then will be conserved to the next generation. Meanwhile, bees that are not previously chosen as selected bees are assigned random positions to scout for potential new solutions.

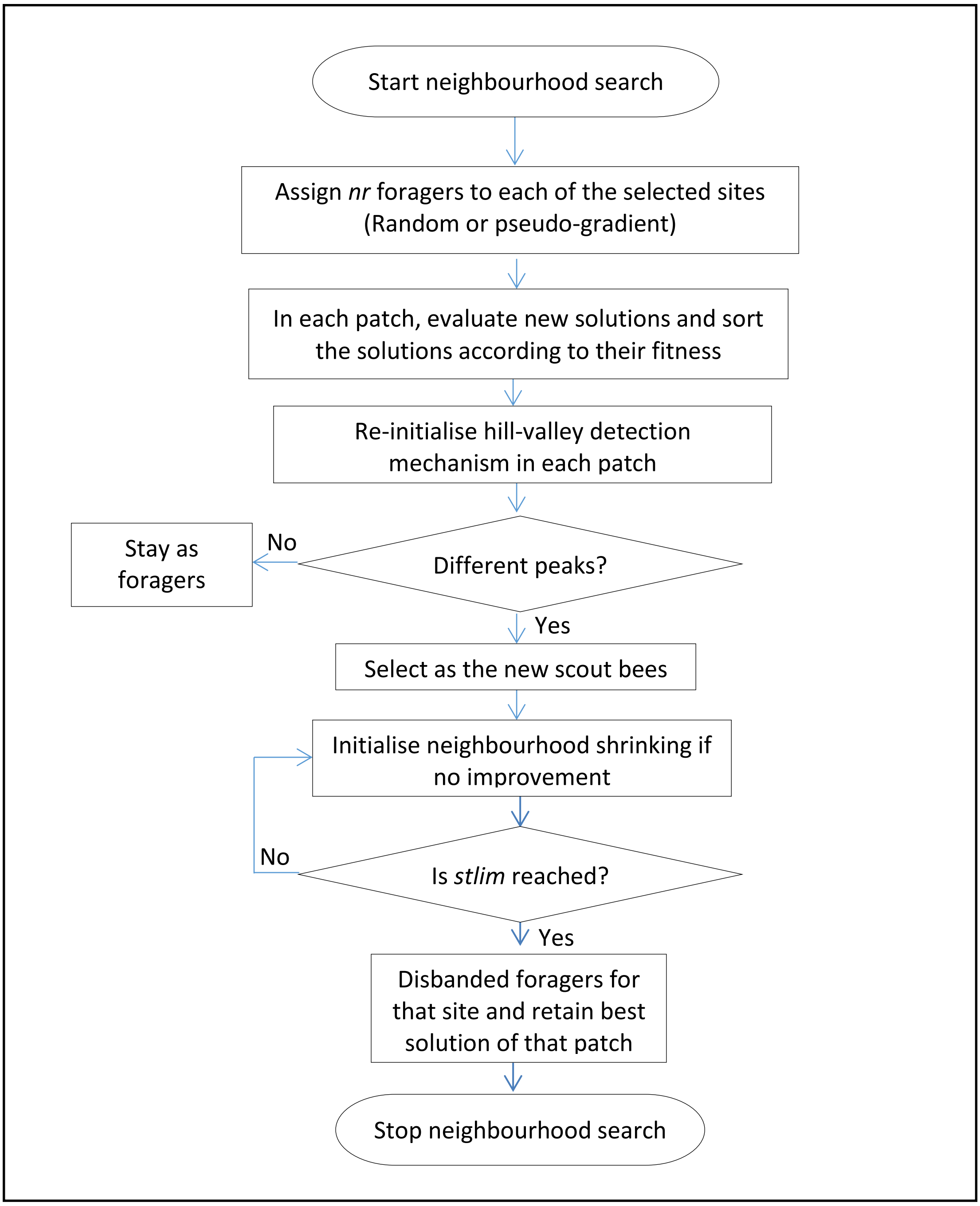

Figure 5.3: Flow chart of the neighbourhood search in EBA

In addition, the size of the neighbourhood will be shrunk if no further improvements are found. Later, after *stlim* and still no improvement, the site is considered to have converged to the local optima. The site will be abandoned and recruited bees of that patch disbanded but the bee with the highest fitness in that patch will retain its position as an inspector bee to monitor any changes to the site. This phenomenon is in accordance with nature where bees that have retired from foraging at a food source will make occasional visits to that site as observed by Granovsky et al. (2012). Foraging can be resumed if its quality improved. With this, the maintenance of the local solutions found so far is ensured until the algorithm's termination without the need to maintain them in a separate list (population). Furthermore, in case of dynamic optimisation problems, these bees can detect any changes. When this happens, inferior bees of the same peak are no longer sent for global scouting as in the earlier progress of the algorithm. Instead, they are killed in order to control the population size from growing out of proportion.

For the next iteration, the new generation of bees will again be subjected to Hill-Valley detection mechanism before selection begins. These plus the role of inspector bees are how the bee population and the number of selected sites can change according to solution landscape. The aforementioned steps are repeated until terminating condition is met.

5.5 Experimental Set-up

Ten commonly used multimodal functions as reviewed by Das et al. (2011) were selected to demonstrate the ability of EBA and its counterpart EBA-PG in finding multiple optimum. These functions have different characteristics such as deceptiveness, equal spread of optima and vice versa, and the existence of multiple local optima among multiple global optima. Table 5.1 lists the dimensionality and the number of both global and local peaks for each function while Figures 5.4 until 5.6 show the surface of this test suite. Complete functions' equation can be refered in Appendix D.

Table 5.1: Multimodal benchmark functions with their corresponding dimensions and number of peaks

Name of functions	Dimensions	Number of peaks [Global (Local)]
f_1 Two peaks trap	1	1 (1)
f_2 Central two peaks trap	1	1 (1)
f_3 Five uneven peaks trap	1	2 (3)
f_4 Equal maxima	1	5 (-)
f_5 Decreasing maxima	1	1 (4)
f_6 Uneven maxima	1	5 (-)
f_7 Uneven decreasing maxima	1	1 (4)
f_8 Himmelblau	2	4 (-)
f_9 Camelback	2	2 (2)
f_{10} Shekel's foxholes	2	1 (24)

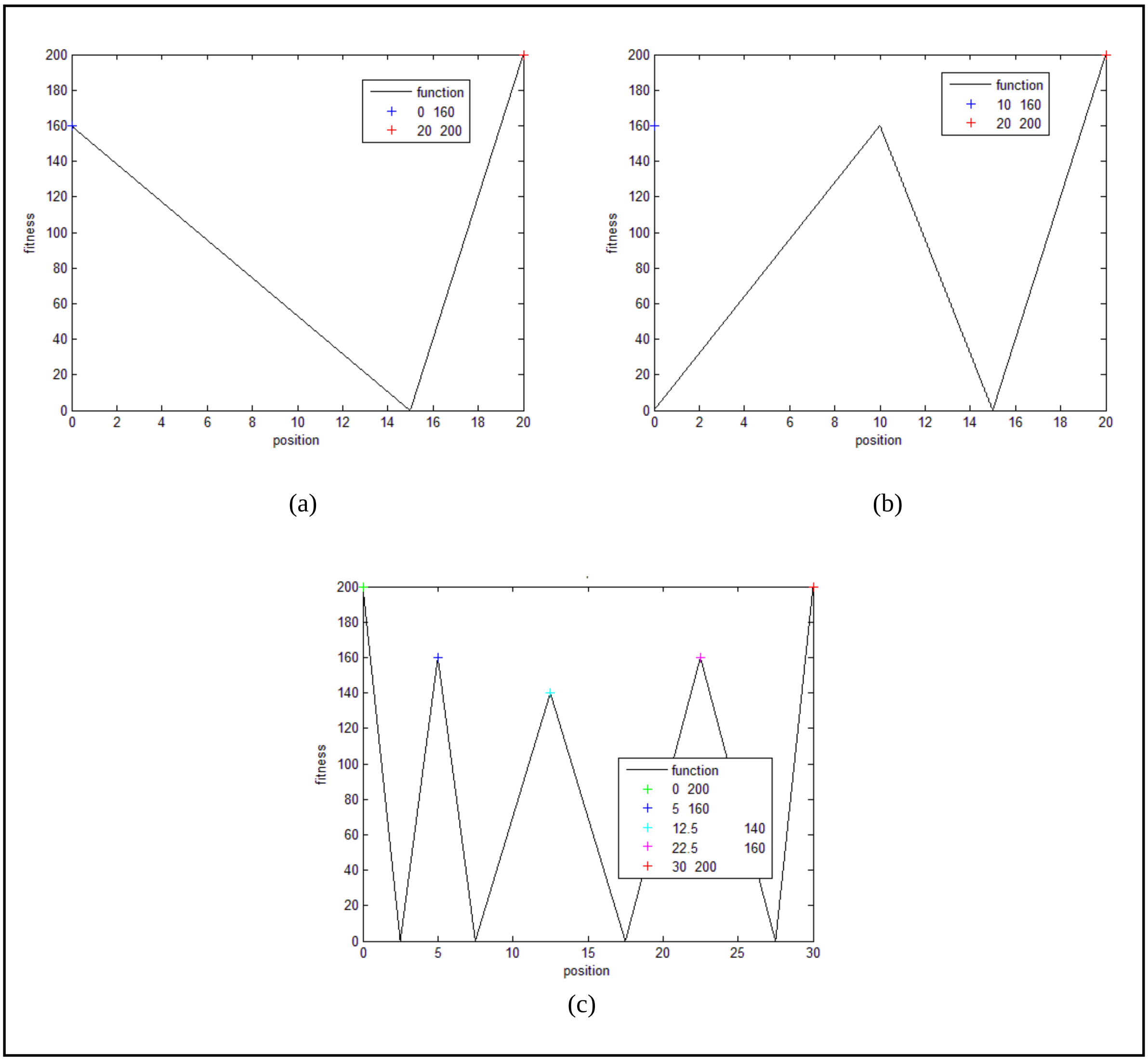

Figure 5.4: Surface of one dimension deceptive functions: (a) f_1 Two peaks trap; (b) f_2 Central two peaks trap; (c) f_3 Five uneven peaks trap

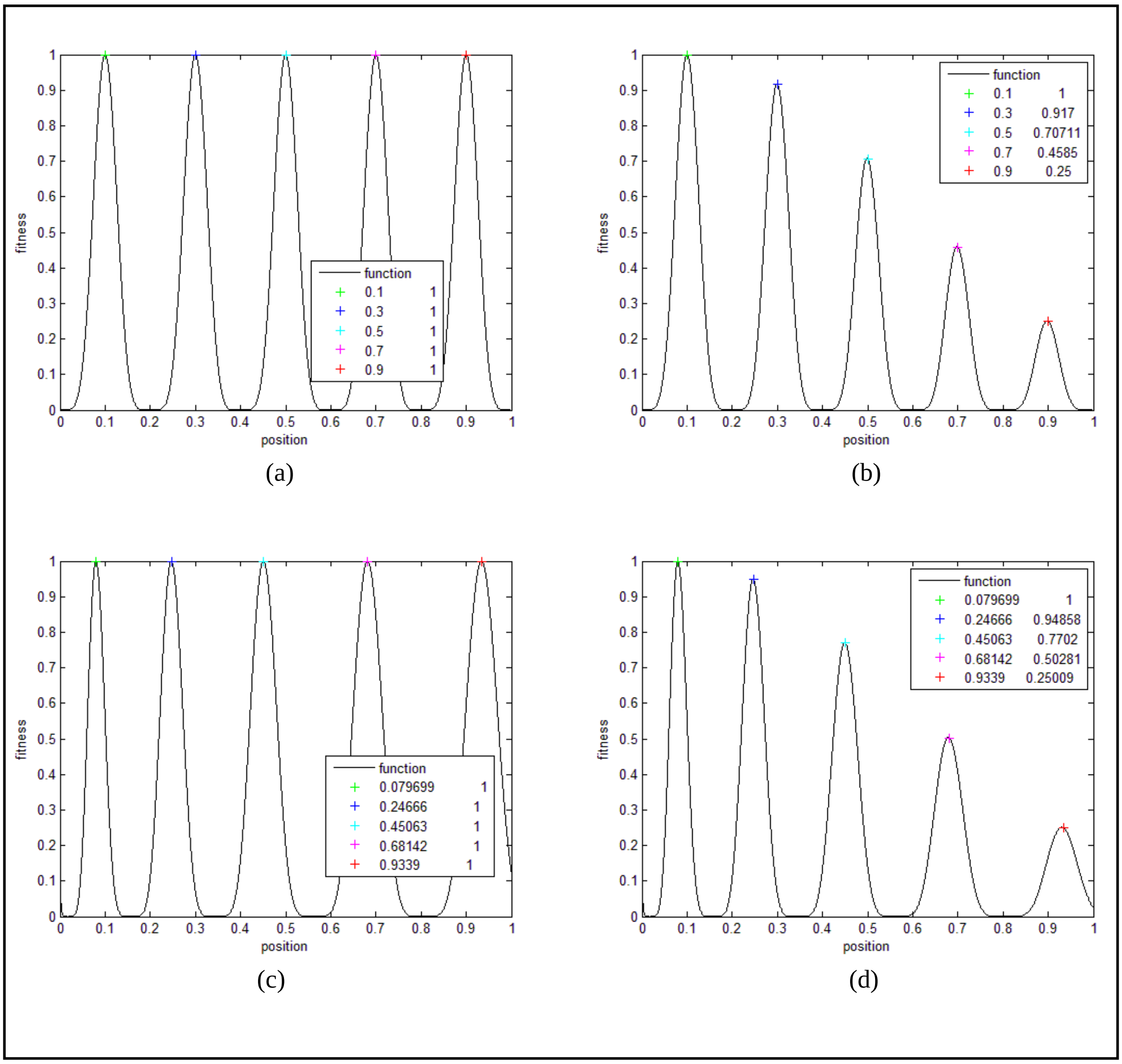

Figure 5.5: Surface of one dimension multimodal functions: (a) f_4 Equal maxima; (b) f_5 Decreasing maxima; (c) f_6 Uneven maxima; (d) f_7 Uneven decreasing maxima

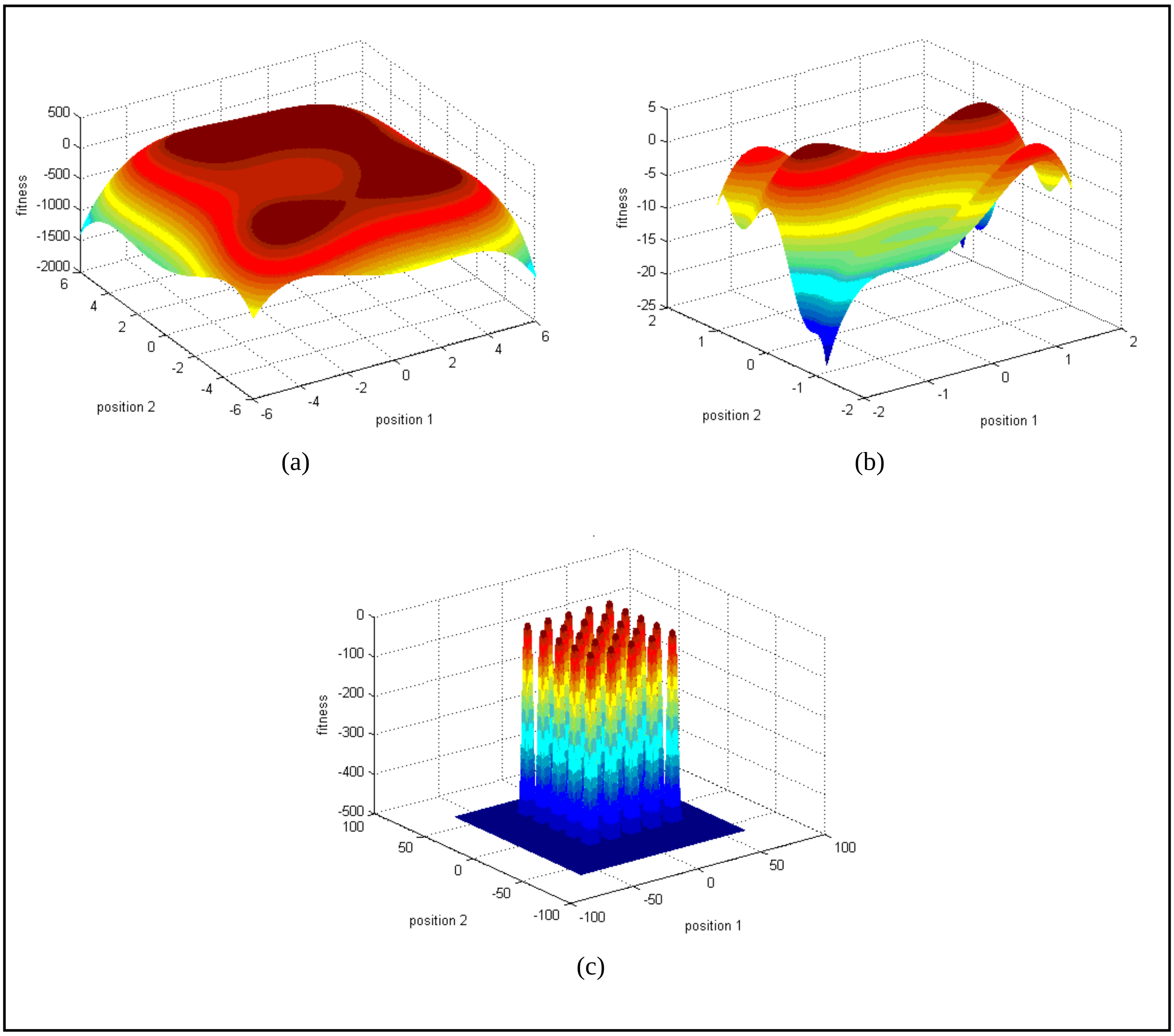

Figure 5.6: Surface of two dimensions multimodal functions: (a) f_8 Himmelblau; (b) f_9 Camelback; (c) f_{10} Shekel's foxholes

Functions f_1 until f_3 are deceptive functions because the area of the local peak(s) is wider than the global optimum, more individuals from the population can be misguided towards this region. These functions are good to evaluate the performance of an algorithm in tackling deceptiveness. Even though, functions f_4 and f_5 are evenly distributed functions but peaks in f_5 decrease exponentially leaving it with only one global optima. The peaks in f_6 and f_7 are irregularly spaced with the peaks in f_7 declining like in f_5. Functions f_4 until f_7 are useful to test optimiser ability to form multiple niches either globally or locally. Meanwhile, f_8 until f_{10} are 2-dimensional multimodal functions. f_8 has no local optima with four global solutions. The surface almost likes a plateau with two peaks that are closer to each other than the other two. f_9 has two global optima as well as two local optima that are distinguishingly apart. f_{10} has 25 peaks with one of them is the global optimum. This function can be challenging if the aim is to locate not only the global peak but all the local ones as well.

Multimodal Artificial Bee Colony (ABC-M) and four different variants of *local best* PSO with ring topology (r2pso, r3pso, r2pso-lhc and, r3pso-lhc) are employed for benchmarking since these swarm optimisers are known multimodal algorithms that do not require the use of niching parameters. ABC-M memorised all the abandoned solutions since an abandoned food source is considered has reach the optimum. At the end of the algorithm's run, the memorised set is retrieved together with the best global solution to reveal the multiple global and local optimal (Liu et al., 2012). Meanwhile, each member in r2pso only interacts with the member of its immediate right whereas members in r3pso communicate with its right and left members. The same mechanism of r2pso is applied in r2pso-lhc, but it has no overlapping neighbourhood thus acts more like multiple local hill climbers (i.e. multiple PSO search in parallel). The same applies to r3pso-lhc (Li, 2010).

To determine whether a peak has been located, a measure of accuracy ε is utilised. If the distance between the found solution with the known optimum is below ε, then the optimiser has successfully found the peak. The number of evaluations needed to locate all the peaks is recorded for comparison. All experiments were run 50 times and the average is taken. Success rate which is the percentage of runs where all peaks are located is also used as a performance criterion.

From their respective papers, both ABC-M and all variants of PSO employed the population size of 50. However, since population size of EBA and EBA-PG varies, the parameter setting for both follows closely the parameters set in the previous chapter as shown in Table 5.2. This table shows that only three parameters need to be set by the user while the initial neighbourhood size depends on the initial number of scouts. However, so that a fair comparison is made, the stopping criterion is the same for all optimisers as suggested by Li (2010) and Liu et al. (2012) for when all peaks are found or 100,000 evaluations for all functions.

Table 5.2: Parameter setting for EBA and EBA-PG

Parameters	**Value**
Initial number of scout bees, *ins*	6
Number of recruiters on selected sites, *nr*	9
Stagnation limit, *stlim*	10
Initial neighbourhood size, *ngh*	Search range/*ins*

5.6 Results and Discussion

The success rate obtained by EBA, EBA-PG, and benchmarked algorithms are presented in Table 5.3. All results for ABC-M and variants of PSO are extracted from Liu et al. (2012). This table also shows the level of accuracy used to detect the present of peak, ε for each functions.

Table 5.3: Comparison of success rate (%) for f_1-f_{10} between EBA and EBA-PG with other multimodal swarm optimisers

Functions	ε	EBA	EBA-PG	ABC-M	r2pso	r3pso	r2pso-lhc	r3pso-lhc
f_1	0.1	**100**	**100**	**100**	98	**100**	94	78
f_2	0.1	**100**	**100**	**100**	**100**	96	98	88
f_3	5	**100**	**100**	**100**	**100**	96	96	96
f_4	0.01	**100**	**100**	**100**	**100**	**100**	**100**	**100**
f_5	0.01	**100**	**100**	**100**	98	**100**	**100**	**100**
f_6	0.01	**100**	**100**	**100**	98	98	**100**	**100**
f_7	0.01	**100**	**100**	**100**	**100**	**100**	**100**	**100**
f_8	0.1	80	98	**100**	92	74	**100**	98
f_9	0.01	**100**	**100**	**100**	**100**	**100**	**100**	**100**
f_{10}	0.01	98	**100**	**100**	**100**	**100**	72	78

From this table, EBA was able to achieve 100% success in all functions except for f_8 and f_{10}. In f_8 the almost indistinguishable peaks due to the quite flat terrain made it harder for the Hill-Valley in EBA to differentiate between two summits. Increasing the number of sample points taken between hills can rectify the situation but at the risk of increasing the number of evaluation as well. EBA-PG also only managed to achieve 98% of success rate in f_8. The increase in percentage is due to the faster neighbourhood search which allows the algorithm to discover more peaks before reaching the maximum evaluation. As there are 25 peaks in f_{10}, more peaks are being selected for neighbourhood search after Hill-Valley. This also reduces the chance for the algorithm to reach near each optimal before the allowable number of evaluation as overall number of recruiters linearly rise with the number of selected sites. Again, the pseudo-gradient helps in EBA-PG because it manages to attain the 100% success rate in f_{10}. Nevertheless, both EBA and EBA-PG are capable to handle deceptive functions f_1-f_3 compared to some of the PSO variants. The same applies for functions that are evenly or irregularly spaced whether in one or two dimensions. This demonstrates that the Hill-Valley mechanism in EBA and EBA-PG is effective at creating multiple niches across the search landscape. The result also shows that only ABC-M managed to obtain 100% success rate for all functions which suggests that a separate memory to maintain candidate solutions is the better option in tackling MMO.

Tables 5.4 and 5.5 exhibit the number of evaluations taken for all optimisers to locate the multiple optima with bold typeface is the best result. Even though EBA and EBA-PG are capable to identify all the substantial peaks, they require a higher number of evaluations. For f_8 and f_{10}, EBA reached on average the maximum allowable evaluation. The pseudo-gradient in EBA-PG improved EBA's results but still some of the other optimisers performed better. The higher evaluation is attributed to the extra evaluation of the three sample points needed to distinguish between all feasible solutions whether they belong to the same peak. Although, with modern high speed CPU this can be made negligible.

Table 5.4: Comparison of mean, standard deviation, and standard error of number of evaluations over 50 runs for f_1-f_5 between EBA and EBA-PG with other multimodal swarm optimisers

Functions		f_1	f_2	f_3	f_4	f_5
EBA	**Mean**	3,350.00	3,330.00	3,490.00	53,000.00	13,000.00
	Std. Dev.	13,827.72	13,827.53	902.24	37,338.42	31,220.78
	Std. Error	1,955.53	1,955.51	127.60	5,280.45	4,415.29
EBA-PG	**Mean**	1,360.00	1,370.00	1,380.00	1,520.00	1,710.00
	Std. Dev.	463.82	541.18	548.58	232.05	264.30
	Std. Error	65.60	76.54	77.58	32.82	37.38
ABC-M	**Mean**	**76.70**	**78.30**	**80.00**	719.00	197.00
	Std. Dev.	**63.50**	**88.18**	**139.94**	7,755.55	947.31
	Std. Error	**8.98**	**12.47**	**19.79**	1,096.80	133.97
r2pso	**Mean**	3,460.00	2,960.00	978.00	**376.00**	2,120.00
	Std. Dev.	13,965.01	10,748.45	1,319.81	**216.02**	14,138.81
	Std. Error	1,974.95	1,520.06	186.65	**30.55**	1,999.53
r3pso	**Mean**	2,620.00	5,340.00	4,650.00	443.00	**141.00**
	Std. Dev.	6,180.61	19,549.59	19,687.34	366.28	**79.34**
	Std. Error	874.07	2,764.73	2,784.21	51.80	**11.22**
r2pso-lhc	**Mean**	7,390.00	4,340.00	4,710.00	396.00	143.00
	Std. Dev.	23,674.29	15,765.94	19,681.68	360.70	103.52
	Std. Error	3,348.05	2,229.64	2,783.41	51.01	14.64
r3pso-lhc	**Mean**	23,200.00	13,100.00	6,730.00	447.00	144.00
	Std. Dev.	41,257.28	32,448.00	21,835.53	373.00	96.73
	Std. Error	5,834.66	4,588.84	3,088.01	52.75	13.68

Table 5.5: Comparison of mean, standard deviation, and standard error of number of evaluations over 50 runs for f_6-f_{10} between EBA and EBA-PG with other swarm optimisers

Functions		f_6	f_7	f_8	f_9	f_{10}
EBA	**Mean**	60,600.00	5,700.00	100,000	24,100.00	100,000
	Std. Dev.	38,760.14	19,701.06	60,011.49	40,310.34	13.49
	Std. Error	5,481.51	2,786.15	8,486.91	5,700.74	1.91
EBA-PG	**Mean**	2,050.00	1,560.00	11,700.00	1,180.00	51,000.00
	Std. Dev.	357.01	318.11	3,129.77	631.85	11,653.77
	Std. Error	50.49	45.00	442.62	89.36	1,648.09
ABC-M	**Mean**	2,340.00	223.00	**1,340.00**	**360.00**	**1,130.00**
	Std. Dev.	22,785.10	1,461.94	**13,657.06**	**909.00**	**4,013.61**
	Std. Error	3,222.30	206.75	**1,931.40**	**128.55**	**567.61**
r2pso	**Mean**	2,400.00	175.00	7,870.00	619.00	4,360.00
	Std. Dev.	14,101.34	126.64	20,447.34	170.48	3,959.66
	Std. Error	1,994.23	17.91	2,891.69	24.11	559.98
r3pso	**Mean**	2,440.00	**160.00**	21,400.00	684.00	3,510.00
	Std. Dev.	14,104.87	**142.84**	38,658.87	212.27	3,207.08
	Std. Error	1,994.73	**20.20**	5,467.19	30.02	453.55
r2pso-lhc	**Mean**	**456.00**	178.00	1,490.00	680.00	29,700.00
	Std. Dev.	**238.51**	128.27	978.07	213.97	44,385.59
	Std. Error	**33.73**	18.14	138.32	30.26	6,277.07
r3pso-lhc	**Mean**	623.00	162.00	7,380.00	650.00	24,800.00
	Std. Dev.	1,931.32	119.36	23,667.78	177.00	40,576.97
	Std. Error	273.13	16.88	3,347.13	25.03	5,738.45

Table 5.6 displays the percentage of improvement in term of mean evaluation between EBA and EBA-PG. On average, EBA-PG improves by 76.43% which is fairly consistent with the results obtained in Chapter 3. However, based on previous tables, a better strategy in forming niches is still required so as not to consume a lot of function evaluation.

Table 5.6: Percentage improvement for mean evaluation between EBA-PG and EBA

Functions	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
% improvement of mean evaluation	59.4	58.9	60.5	97.1	86.8	96.6	72.6	88.3	95.1	49.0
Average										**76.43%**

Nonetheless, statistical analysis performed based on the average evaluation indicates that though EBA and EBA-PG have moderately higher evaluation than the other algorithms, they are not statistically significant in some cases. The *p*-value by using *t*-test with 95% confidence interval is shown in Table 5.7. Boldface indicates *p*-value lower than 0.05 meaning the result is statistically significant. For cases where it is not statistically distinguishable, the algorithms are considered to perform statistically the same.

Table 5.7: Comparison of *p*-value using *t*-test ($p < 0.05$) between EBA and EBA-PG with other multimodal swarm optimisers

Func.	**EBA-PG**	**ABC-M**		**r2pso**		**r3pso**		**r2pso-lhc**		**r3pso-lhc**	
	EBA	**EBA**	**EBA-PG**	**EBA**	**EBA-PG**	**EBA**	**EBA-PG**	**EBA**	**EBA-PG**	**EBA**	**EBA-PG**
f_1	+ 0.3116	+ 0.0973	+ **0.0000**	- 0.9685	- 0.2905	+ 0.7340	- 0.1538	- 0.3000	- 0.0748	- **0.0017**	- **0.0003**
f_2	+ 0.3190	+ 0.0995	+ **0.0000**	+ 0.9628	- 0.2987	- 0.5542	- 0.1544	- 0.7342	- 0.1862	- 0.0530	- **0.0003**
f_3	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ 0.0617	+ **0.0000**	- 0.2404	- 0.2432	- 0.2318	- 0.2346	- 0.0853	- 0.0864
f_4	+ **0.0000**	+ **0.0000**	+ 0.1900	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**
f_5	+ **0.0121**	+ **0.0446**	+ **0.0000**	+ **0.0270**	- 0.8380	+ **0.0044**	+ **0.0000**	+ **0.0044**	+ **0.0000**	+ **0.0045**	+ **0.0000**
f_6	+ **0.0000**	+ **0.0000**	- 0.9285	+ **0.0000**	- 0.8611	+ **0.0000**	- 0.8454	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**
f_7	+ 0.1406	+ 0.0528	+ **0.0000**	+ 0.0502	+ **0.0000**	+ **0.0496**	+ **0.0000**	+ 0.0503	+ **0.0000**	+ **0.0496**	+ **0.0000**
f_8	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ 0.1935	+ **0.0000**	- 0.0801	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ 0.2037
f_9	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**
f_{10}	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0000**	+ **0.0014**	+ **0.0000**	+ **0.0000**

Note: (+ signifies the upper column optimiser as the better performer in mean evaluation)

5.6.1 Modification to EBA-PG

Results attained in the previous section clearly show that even though EBA and EBA-PG are able to identify all the necessary peaks, however the function evaluations taken are quite large compare to the other multimodal swarm optimisers. Thus, there is the need for a different method able to create niche and at the same time require a smaller number of function evaluation. Earlier in this chapter, the idea to use the Hill-Valley to form patches stemmed from the Patch Overlap Avoidance (POA) strategy in Chapter 4. To recall, POA is able to build patch with minimum chances of overlapping with others by using short term memory but two or more patches can still be formed in the same peak. Initially, this method was thought not suitable to be applied as a multimodal algorithm because recruiters of more than one patches competing on the same peak might be useful if distributed to a different hill. However, since this strategy also avoids the positioning of bees on otherwise visited locations in subsequent cycles, adapting POA in EBA-PG could reduce the occurrence of the same point to be evaluated by the Hill-Valley.

This variant dubbed as EBA-PG-POA works as the steps follows:

1. Initialise the position of initial number of scout bees and set their corresponding boundary
2. Evaluate the fitness and sort
3. Select non-overlap scout bees of high fitness to be evaluated by Hill-Valley
4. Perform neighbourhood search for selected sites after Hill-Valley
5. Recruit bees for selected sites by using pseudo-gradient and evaluate fitness
6. Select the fittest bee from each patch

7. Assign unselected scout bees to search randomly avoiding boundary set by the fittest bees and previous memory, and evaluate their fitness
8. Stop algorithm when stopping criterion is fulfilled.

In EBA-PG-POA, Hill-Valley is still utilised but only on scout bees with non-overlapping corresponding boundary. This is to avoid non-overlapping patch of the same peak to be formed especially in the case of a peak in an almost plateau surface. Recruiters are positioned inside the selected sites after Hill-Valley using the pseudo-gradient method. No Hill-Valley is performed in the neighbourhood search. In EBA, Hill-Valley is re-initialised in neighbourhood search to detect a presence of new peak if they are quite close to each other. Here, there is no need for that since it is assumed that if they are near then they will overlap. The remaining scout bees re-assigned randomly just like the ones from the POA procedure. Unselected scouts after Hill-Valley are killed instead. The rest of the algorithm functions just like EBA with the deployment of inspector bees to maintain the multiple optimums found.

Table 5.8 presents the result of EBA-PG-POA as well as the percentage of improvement and *p*-value for mean evaluation in comparison to EBA-PG. EBA-PG-POA able to attain 100% success rate for all functions. The percentage of improvement for mean evaluation on average is 68.09% across all functions. The improve performance is due to the small number of points need to be evaluated by the Hill-Valley. The population size is also controls early on with the unselected bees after Hill-Valley are killed instead after site abandonment occurs. The *p*-value shows that these results are statistically significant. However, the number of evaluation is still slightly higher than the best performer of each function from Tables 5.4 and 5.5. Yet this does not undermine the ability of this variant to locate all the substantial peaks. Tuning the number of recruiters or stagnation limit could potentially lead to a better performance.

Figures 5.7 until 5.16 depict the convergence curve of all three variants of EBA. These figures prove the faster rate of EBA-PG-POA in detecting multiple peaks.

Table 5.8: Average evaluation of EBA-PG-POA and performance comparison with EBA-PG

Functions	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
Success rate (%)	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**
Mean	405.00	570.00	581.00	*432.00*	*730.00*	*545.00*	*737.00*	*2,030.00*	478.00	*1,450.00*
Std. Dev.	184.41	201.81	195.48	295.35	190.19	248.29	190.47	801.40	137.23	994.95
Std. Error	26.08	28.54	27.65	41.77	26.90	35.11	26.94	113.34	19.41	140.71
% improvement	70.22	58.39	57.90	71.58	57.31	73.41	52.76	82.65	59.49	97.16
p-value	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**

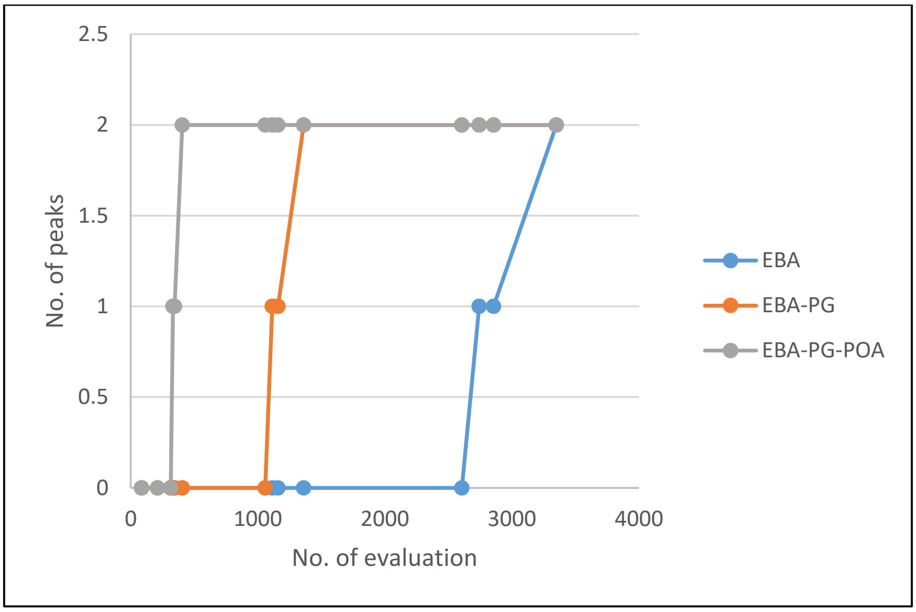

Figure 5.7: Convergence curve for f_1

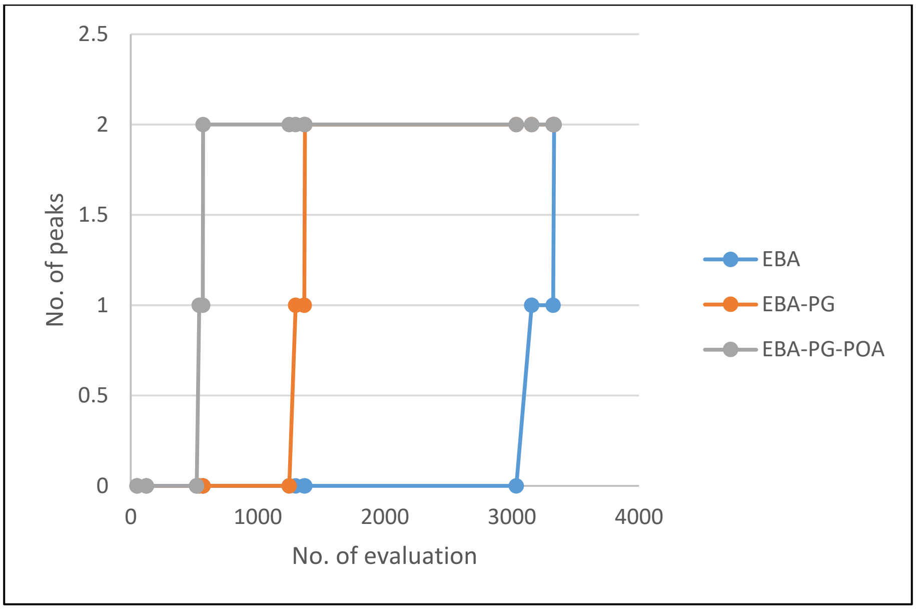

Figure 5.8: Convergence curve for f_2

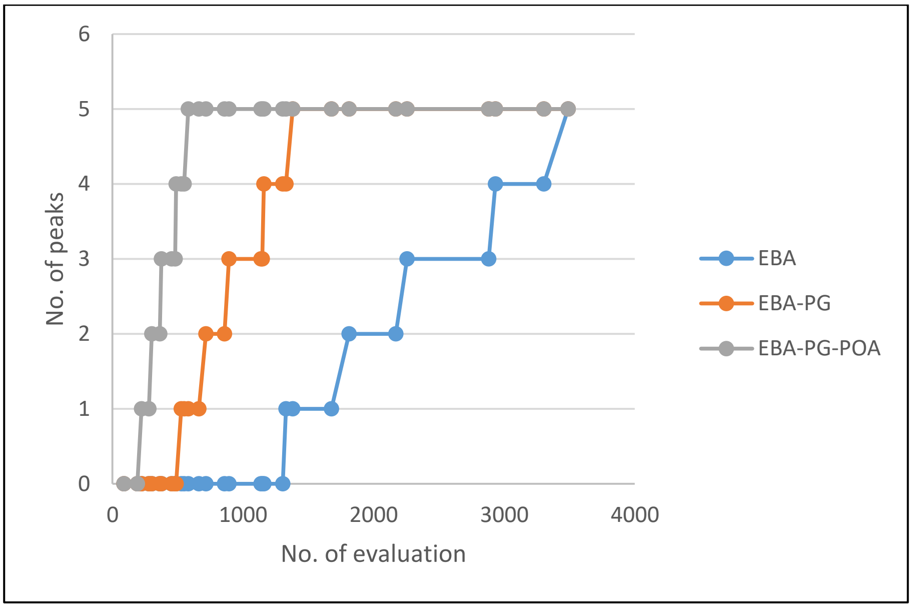

Figure 5.9: Convergence curve for f_3

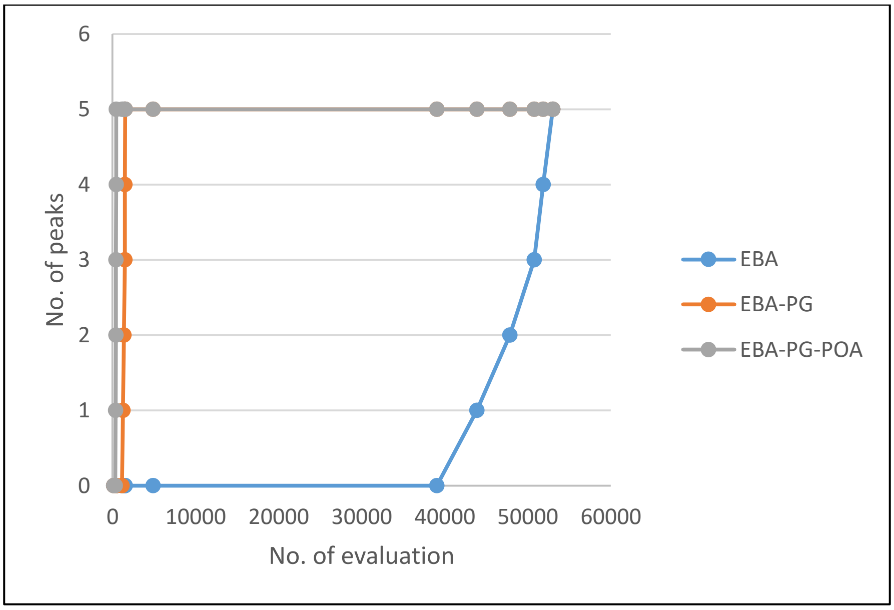

Figure 5.10: Convergence curve for f_4

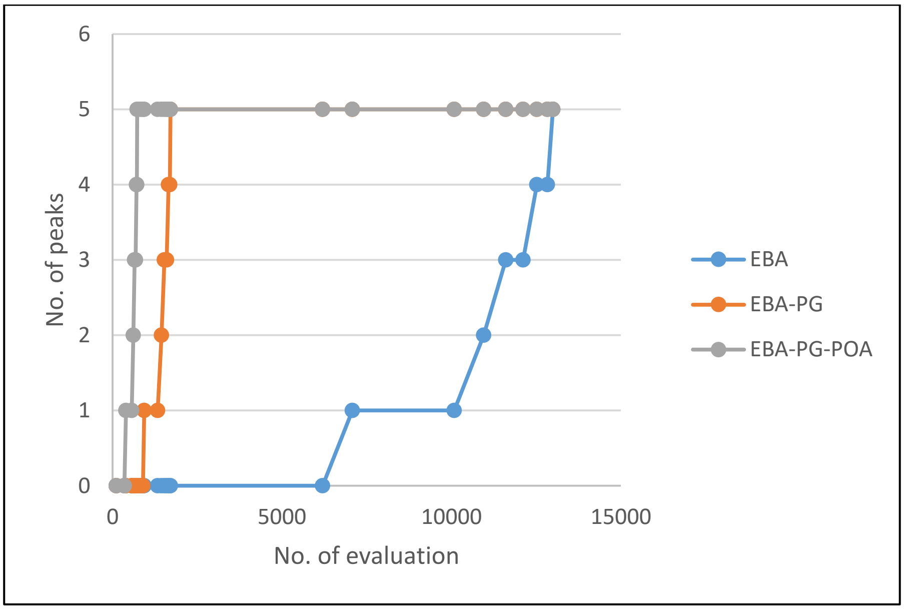

Figure 5.11: Convergence curve for f_5

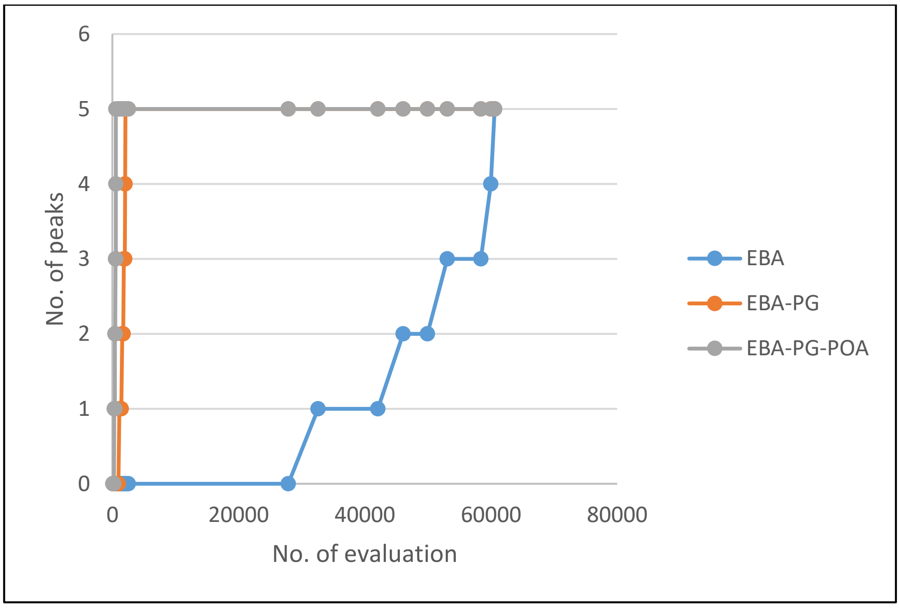

Figure 5.12: Convergence curve for f_6

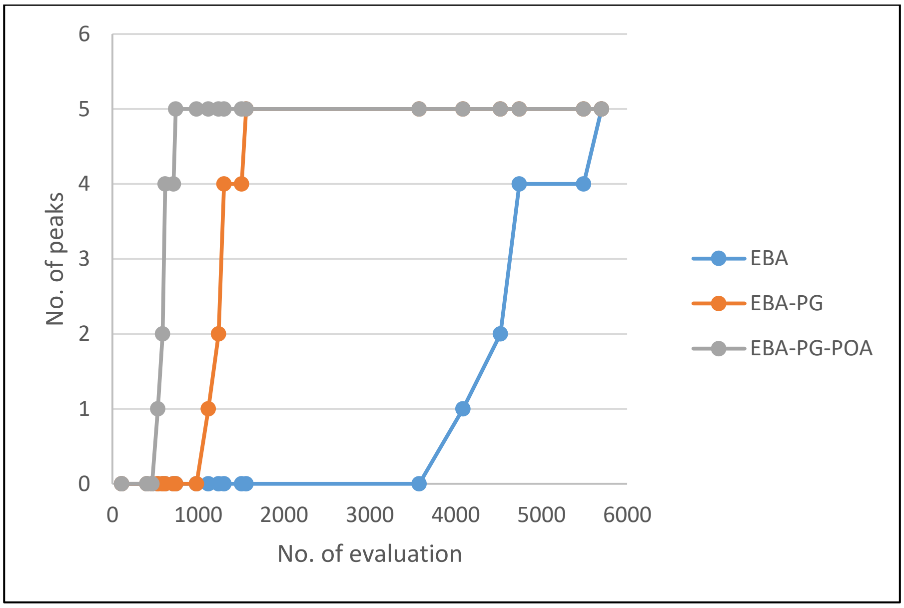

Figure 5.13: Convergence curve for f_7

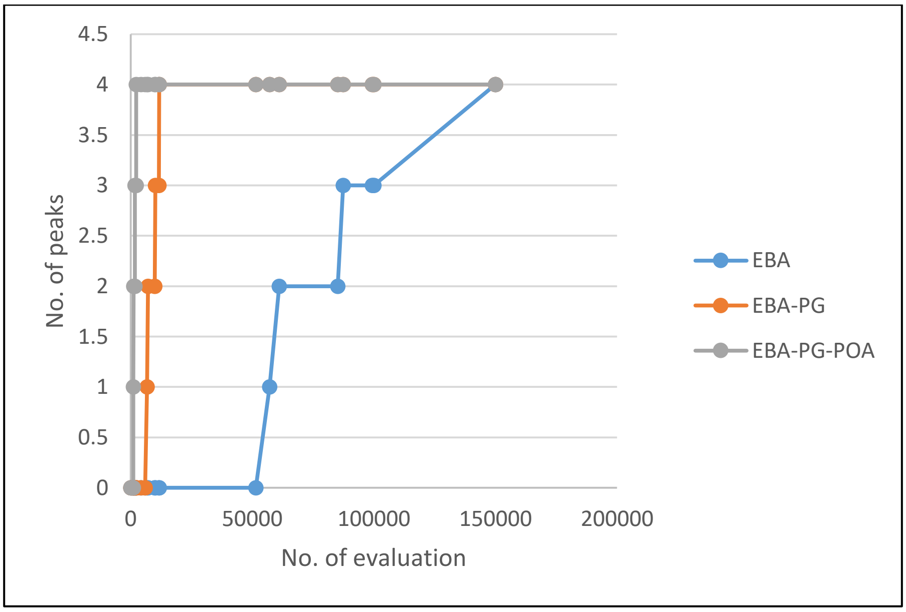

Figure 5.14: Convergence curve for f_8

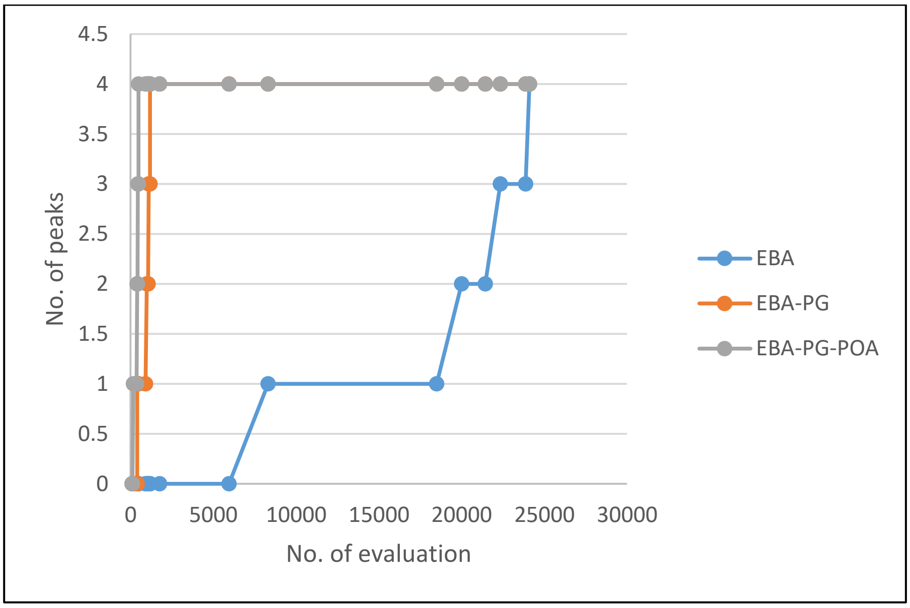

Figure 5.15: Convergence curve for f_9

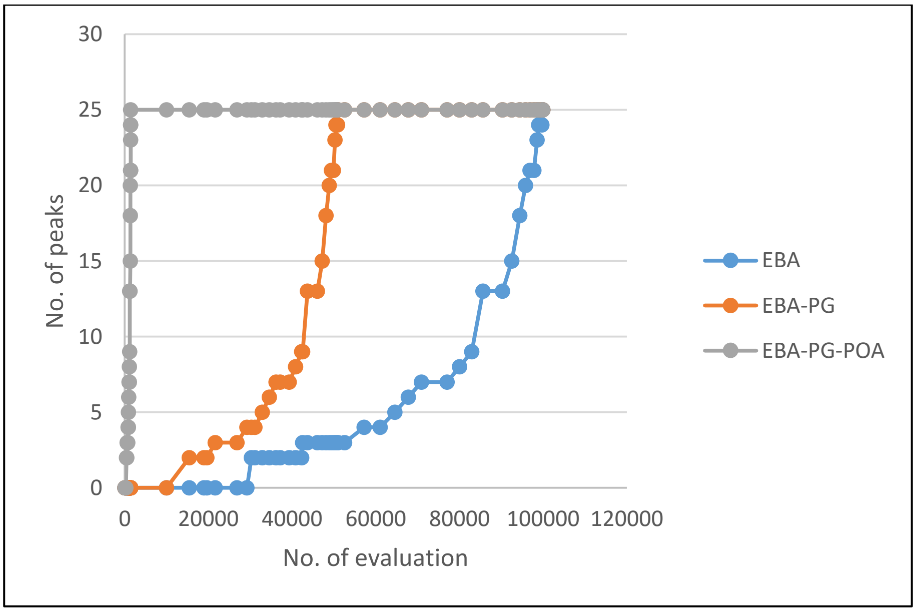

Figure 5.16: Convergence curve for f_{10}

5.6.2 Effect of varying number of initial scout bees

The population size of this variant of Bees Algorithm is not constant as explained in the previous section. In order to exhibit the robustness of this algorithm, a simple experiment is done using the one-dimensional f_7 to get a faster result and this is the most difficult 1D problem. Two-dimension problems were not chosen because it will consume a high computational cost for a simple test. Figure 5.17 shows the effect of varying the value of initial number scouts (*ins*) which is one of the algorithm's parameter. Similar control parameters were used for this test except for the *ins* value which is in the range of 5, 10, 15, 20, 25. Only EBA is employed for this test without the influence of PG and POA methods. Although a different initial number of scouts were used, the number of mean optima found remained constant which means that the dynamically changing bees population do not affect the ability of the proposed algorithm to uncovered all peaks. As suspected, the number of function evaluation increases with the increase of initial number scouts but this is still within the allowable maximum evaluation. This also shows that even with small initial number of scout bees, EBA is still an effective multimodal algorithm.

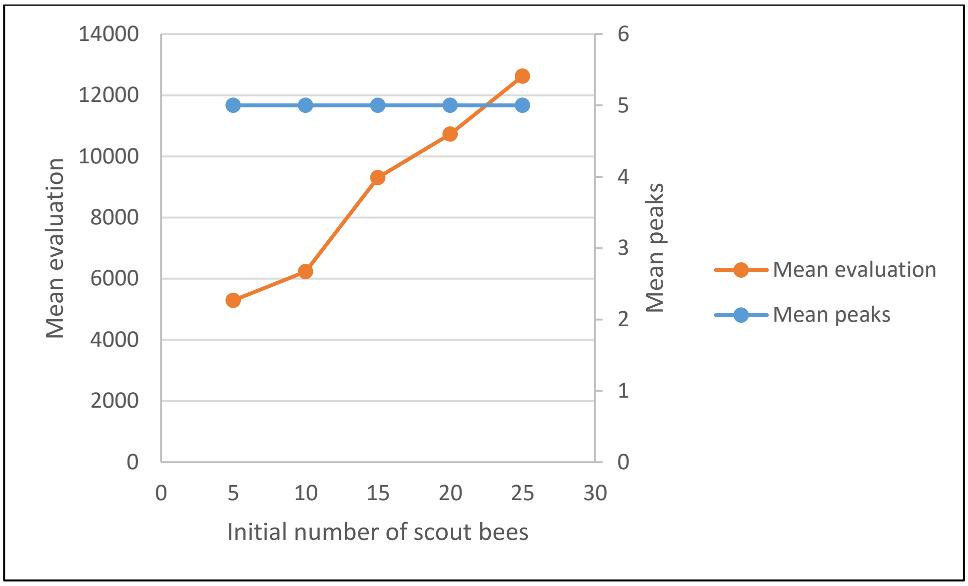

Figure 5.17: Effect of varying initial scout population

5.7 Conclusions

This chapter introduced three variants of the Bees Algorithm that use the Hill-Valley mechanism to maintain the diversity of population in order to locate multiple optimal solutions. This mechanism provides bees with the topological knowledge of the fitness landscape by having the ability to differentiate whether two solutions belong to the same optima. With the assistance of this technique, no niching parameter is needed and the population size was kept dynamic according to the search space. Still, the extra evaluation of sample points used to distinguish peaks increased the number of function evaluation required to locate all significant optima. A higher number of peaks means more patches are maintained throughout evolutionary process which resulting in more points to be evaluated in subsequent iteration.

Pseudo-gradient (PG) method is also employed to increase the speed of neighbourhood search. Faster neighbourhood search cause faster abandonment which means an optimum has been discovered. Thus, fewer points are subjected to Hill-Valley in the next cycle and the killing of bees to control the population size also starts sooner. However, there are still chances the remaining bees will fall on discovered solutions. Hence, Patch Overlap Avoidance (POA) is utilised which further reduces the number of evaluations taken.

Results show that all EBA variants have the capability to locate multiple optima albeit they are not the best performer for each function used in benchmarking. However, this study is not concerned with proving the superiority of any algorithm but to investigate the ability of the Bees Algorithm extended with the use of Hill-Valley to tackle multimodal optimisation problems. Furthermore, if the Hill-Valley is to be applied in global optimisation, it could reduce the chances of getting trapped, because by locating local optimal points, the odd of finding the global peaks increases.

CHAPTER 6

CONCLUSIONS

This chapter summarises the the conclusions reached in this study and the contributions made. It also provides suggestions for future work.

6.1 Conclusions

Three objectives have been set for this research. There are to; (i) develop an improved Bees Algorithm with the aid of a gradient-like method to provide search direction for the algorithm in order to achieve faster convergence, (ii) develop a strategy to avoid the formation of overlapping patches in the Bees Algorithm so that recruited bees are managed and distributed more efficiently, and (iii) develop a version of the Bees Algorithm that has the ability to detect multiple global optimal solutions in multimodal optimisation problems. Based on the findings of this study, those objectives have all been realised.

This thesis has presented three modifications to the Bees Algorithm in order to improve its performance, especially in terms of convergence speed in handling continuous optimisation problems. The first improvement made to the algorithm is by integrating the pseudo-gradient method into the neighbourhood search. This procedure is able to provide a good search direction to the best bee of each patch by contrasting the fitness of the current point with the previous point. Hence, no differentiation of the objective function is required. The pseudo-direction ultimately speeds up the search process compared to the Standard Bees Algorithm based on the reduction in the number of function evaluations ranging from 2.8% to 99.3% with various test functions as shown in Chapter 3. Evaluation against other swam-inspired algorithms also suggests the effectiveness of this method except on the low dimension problem (i.e. 2D) which could be due to random exploration since the pseudo-gradient calculation is only performed on the local scale. The Pseudo-gradient Bees Algorithm (PG-BA) also performed moderately on asymmetrical function which shows that, even though no direct gradient is computed, this method still inherits some behaviour of its gradient-based counterparts. This means that a good starting point will certainly benefit the algorithm. Nonetheless, the multiple patches built across the problem landscape balance this effect out. The algorithm also demonstrates that it is able to train Feedforward Neural Networks to a satisfactory result. Thus, the development of PG-BA has fulfilled Objective 1.

The second enhancement was an approach to reduce the creation of overlapping patches. The Patch Overlap Avoidance Bees Algorithm (POA-BA) avoids redundancy in the search area by forbidding the exact intensification of previously visited solutions along with their corresponding neighbourhood using a short-term memory. With this strategy, the bees are no longer allowed to land on abandoned sites or near other scout bees. This results in the minimisation of the number of patches formed on an identical hill or basin. The scaled down region ensures a thorough search of the problem landscape as resources (bees) are well disseminated and are not wasted on an already foraged area. The same benchmark problems as PG-BA were solved using this modified strategy. Overall, POA-BA enhances the global search as the next scout bees cannot inhabit the same spot despite the idea behind this scheme being initially to reduce the chances of repeated exploitation at the neighbourhood level. Thus, performance on low dimension problems also improves. As a result, Objective 2 has been met.

Finally, as there is a demand for obtaining multiple optimal solutions, the competency of the Bees Algorithm is extended to preserve diverse solutions during the evolution process. This warrants distinct multi-solutions to be located in one execution runtime. Since the Bees Algorithm is a population-type optimiser, there are already a population of candidate solutions. To make sure that all possible optimum points within the fitness landscape are found, a better technique than Patch Overlap Avoidance is required. This is because as long as the patches do not cross one another, they are permitted to exist at the same peak, which can make the algorithm miss a more substantial hill. Therefore, the Extended Bees Algorithm (EBA) uses the hill-valley mechanism to select the fittest between two points of indistinguishable summit to survive to the next generation. EBA do not utilise any niching parameters unlike other multimodal algorithms. In fact it uses fewer control parameters as no discrepancy is made between elite and best bees. By doing so, the population size as well as the number of selected sites vary to adapt to the number of optima found. Sites that do not improve after a certain cycle, are considered to have reached their maximum. An inspector bee is tasked to guard the neighbourhood to avoid being revisited by other scout bees. The role of inspector bees reduces the dependency on memory which can limit computing resources. At this point, some bees are 'killed' to control the population size. This extended variant is tested against continuous multimodal numerical functions and the outcomes are compared with other multimodal Swarm Intelligence algorithms that also do not employ niching parameters. Although the new Bees Algorithm was able to locate all the significant optima, the number of function evaluations is high due to the sampling made by the hill-valley method. The assistance by the pseudo-gradient technique only improves the convergence speed to some extent. It is not until the incorporation of Patch Overlap Avoidance before the hill-valley technique is executed that the algorithm manages to thrive. Consequently, EBA has satisfied Objective 3.

6.2 Contributions

This research introduces new improvements to the Bees Algorithm in solving continuous optimisation problems. The specific contributions are as follows:

- The development of PG-BA provides a guided search direction to the algorithm without the need to differentiate the objective functions. This consequently helps the algorithm to converge faster to the optimum compared to the standard Bees Algorithm.
- The development of POA-BA enables the algorithm to reduce the number of patches that overlap with others. Dissimilar neighbourhoods can be formed on the same peak (for maximisation problems) or valley (for minimisation problems) as long as they do not intersect one another. This guarantees a more comprehensive investigation as the search range is narrowing and the bees can be redirected to a different area.
- Both PG-BA and POA-BA are developed without adding any new parameter to the already established set of parameters of the Bees Algorithm which makes it attractive to users as there are only limited parameters to tune but the performance enhancement is still significant.
- In addition, both of these improved versions are tested using the same value of parameters across all the benchmark tests to a degree of success which shows the robustness of the algorithm.

- Most of the benchmark tests used in this study have never been employed on other versions of the Bees Algorithm which proves that the algorithm may not be the best for certain functions but it can work relatively well for a wide range of problems.
- The dimensions of the problems for the mathematical functions are also much higher than those used with other variants of the Bees Algorithm in the literature. This indicates the potential of the algorithm to be implemented for large scale global optimisation problems.
- The development of EBA extends the ability of the Bees Algorithm to locate multiple optima of equal or comparable fitness without the use of any niching parameters.

6.3 Future work

There are a number of promising new directions for further research that can enrich the Bees Algorithm and expand its prospective applications.

- Even though all the three improvements made in this thesis require no additional control parameters, reducing the current algorithm's parameters or even making them self-adapting will attract more users of the Bees Algorithm.
- There is potential for the Bees Algorithm to solve optimisation problems of a larger scale as the improved algorithms performed well on functions with 50 dimensions. Future research should involve scaling the dimensions up to 1000D, for example, in continuous numerical benchmark problems. It would be interesting to see the behaviour of the Bees Algorithm when dealing with this kind of magnitude.

- PG-BA accomplished moderate success on an asymmetrical function. Other functions without biases such as rotated or shifted functions should also be considered in the future to understand more how the algorithm behaves.
- Further investigation is also needed to study the possibility to apply all the three new variants to other real-world problems especially for finding multiple optimum solutions.
- Since EBA has the ability to locate multiple solutions, there lay some prospects to expand its use to multi-objective or dynamic optimisation problems.

REFERENCES

Ab Khalid, N.S., Mustafa, M.W. and Mohamed Idris, R., 2015, August. Optimal Parameters Tuning of Power System Stabilizer via Bees Algorithm. In *Applied Mechanics and Materials* (Vol. 781, pp.397-401). Trans Tech Publications.

Abbass, H. A.; Bagirov, A.M. and Zhang, J., 2003. The Discrete Gradient Evolutionary Strategy Method for Global Optimisation. In *IEEE Congress on Evolutionary Computation (CEC'03)*, December 2003, pp. 435-442.

Abbass, H. A. and Teo, Jason, 2001. A True Annealing Approach to the Marriage in Honey-Bees Optimisation Algorithm. In *The inaugural Workshop on Artificial Life. Adelaide, 2001,* pp.1-14.

Abbass, H. A., 2001a. MBO: Marriage In Honey Bees Optimisation A Haplometrosis Polygynous Swarming Approach. In *The Congress on Evolutionary Computation (CEC2001). Seoul, May 2001,* pp.207-214.

Abbass, H. A., 2001b. A Single Queen Single Worker Honey-Bees Approach to 3-SAT. In *The Genetic and Evolutionary Computation Conference (GECCO 2001). San Francisco, 2001.*

Abdelhakim, A.M., Saleh, H.I. and Nassar, A.M., 2015. Quality Metric-Based Fitness Function for Robust Watermarking Optimisation with Bees Algorithm. *IET Image Processing*; pp.1-6.

AbdelHamid, N.M., Halim, M.A. and Fakhr, M.W., 2013, May. Bees Algorithm-Based Document Clustering. In *ICIT 2013 The 6th International Conference on information Technology*.

Abdullah, S. and Alzaqebah, M., 2013. A Hybrid Self-Adaptive Bees Algorithm for Examination Timetabling Problems. *Applied Soft Computing*, *13*(8), pp.3608-3620.

Abdul-Razaq, T.S. and Ali, F.H., 2014. Modification of Some Solution Techniques of Combinatorial Optimization Problems to Analyze the Transposition Cipher. *Mathematical Theory and Modeling*, *4*(9), pp.120-141.

Abidin, Zulkifli Z.; Arshad, Mohd R. and Ngah, Umi K., 2011. A Simulation Based Fly Optimization Algorithm for Swarms of Mini Autonomous Surface Vehicles Application. *indian Journal of Geo-Marine Sciences*, 40(2): 250-266.

Abu-Mouti, Fahad S. and El-Hawary, Mohamed E., 2012. Overview of Artificial Bee Colony (ABC) Algorithm and Its Applications. In *IEEE International System Conference (SysCon). 19-22 March 2012*, pp.1-6.

Addeh, A. and Ebrahimzadeh, A., 2012. Breast Cancer Recognition Using a Novel Hybrid Intelligent Method. *Journal of Medical Signals and Sensors*, 2(2): 22-30.

Afshar, A; Haddad, O. B.; Mariño, M. A. and Adam, B. J., 2007. Honey-Bee Mating Optimisation (HBMO) Algorithm for Optimal Reservoir Operation. *Journal of the Franklin Institute*, 344: 452-462.

Agrawal, Vivek; Sharma, Harish and Bansal, Jagdish C., 2012. Bacterial Foraging Optimization: A Survey. In *Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011), 20-22 December 2011, Vol.130*. pp. 227–242, India: Springer Engineering.

Ahangarpoura, A., Moradib, A., Nafchib, A.M., Ghanbarzadehb, A. and Farboda, M., 2012. Optimization of Continual Production of CNTs by CVD Method by Using the Bees Algorithm and Neural Network. *Proceedings of the 4th International Conference on Nanostructures (ICNS4), 12-14 March 2012*, Kish Island, I.R. Iran, pp.1279-1281.

Ahmad, S. A.; Pham, D. T.; Ng, K. W. and Ang, M. C., 2012. TRIZ-inspired Asymmetrical Search Neighborhood in the Bees Algorithm. In *IEEE 2012 Sixth Asia Modelling Symposium (AMS), May 2012*, pp.29-33.

Ahmad, S.A., Pham, D.T. and Faieza, A.A., 2014, July. Combination of Adaptive Enlargement and Reduction in the Search Neighbourhood in the Bees Algorithm. In *Applied Mechanics and Materials* (Vol. 564, pp.614-618).

Ahmed, J.A. and Brifcani, A.M.A., 2015. A New internal Architecture Based on Feature Selection for Holonic Manufacturing System. *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, industrial, Mechatronic and Manufacturing Engineering*, *9*(8), pp.1477-1480.

Akbari, Reza; Mohammadi, Alireza and Ziarati, Koorush, 2010. A Novel Bee Swarm Optimization Algorithm for Numerical Function Optimization. *Journal of Communication in Nonlinear Science and Numerical Simulation*, 5: 3142–3155.

Akkar, H. A., 2010. Optimizing Opto-Electronic Cellular Neural Networks Using Bees Swarm intelligent. *International Journal of information Processing and Management*, 1(1): 114-125.

Akpinar, Ş. and Baykasoğlu, A., 2014a. Modeling and Solving Mixed-Model Assembly Line Balancing Problem with Setups. Part II: A Multiple Colony Hybrid Bees Algorithm. *Journal of Manufacturing Systems*, *33*(4), pp.445-461.

Akpinar, Ş. and Baykasoğlu, A., 2014b. Multiple Colony Bees Algorithm for Continuous Spaces. *Applied Soft Computing*, *24*, pp.829-841.

Alatas, B., 2010. Chaotic Bee Colony Algorithms for Global Numerical Optimization. *Expert Systems with Applications, 37*: 5682–5687

Alfi, A. and Khosravi, A., 2012. Constrained Nonlinear Optimal Control via a Hybrid BA-SD. *International Journal of Engineering-Transactions C: Aspects* [Online], 25(3): 197-204. Available from http://www.ije.ir [Accessed 1 October 2012].

Alfi, A.; Khosravi, A. and Razavi, S. E., 2011. Bee Algorithm-Based Nonlinear Optimal Control Applied to a Continuous Stirred-Tank Chemical Reactor. *Global Journal of Pure and Applied Science and Technology* [Online], 1: 73-79. Available from http://www.gjpast.com [Accessed 1 March 2012].

Ali, G.A., Enhancing intrusion Detection System (IDS) by Using Honeybee Concepts and Framework. *ICIT 2015 The 7th International Conference on information Technology*, pp.297-302.

Ali, I.K. and Mahmod, A.G., 2015. Hybrid Bees Algorithm with Simulated Annealing for Cryptanalysis of Simple Substitution Cipher. *Journal of Babylon University, 2(23)*, pp.565-574.

Ali, M. M.; Khompatraporn, C. and Zabinsky, Z. B., 2005. A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimisation Test Problems. *Journal of Global Optimisation*, 31: 635-672

Alia, Osama M. and Mandava, Rajeswari, 2011. The Variants of the Harmony Search Algorithm: An Overview. *Artificial intelligent Review*, 36:49–68.

Alomari, O. and Othman, Z. A., 2012. Bees Algorithm for Feature Selection in Network Anomaly Detection. *Journal of Applied Sciences Research*, 8(3): 1748-1756.

Alzaqebah, M. and Abdullah, S., 2011. The Bees Algorithm for Examination Timetabling Problems. *International Journal of Soft Computing* [Online], 1: 105-110. Available from http://www.ijcse.org [Accessed 1 March 2012].

Amirinejad, M. and Ali Noori, A., 2014. Automatic PID Controller Parameter Tuning Using Bees Algorithm. *International Journal of Scientific & Engineering Research*, *5*(8), pp.24-28.

Anantasate, S. and Bhasaputra, P., 2011. A Multi-Objective Bees Algorithm for Multi-Objective Optimal Power Flow Problem. In *IEEE 2011 8th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and information Technology (ECTI-CON), May 2011*. pp. 852-856.

Anantasate, S.; Chokpanyasuwan, C. and Bhasaputra, P., 2010. Optimal Power Flow by Using Bees Algorithm. In *IEEE 2010 International Conference on Electrical Engineering/Electronics Computer Telecommunications and information Technology (ECTI-CON), May 2010*. pp. 430-434.

Ananthara, M.G., Arunkumar, T. and Hemavathy, R., 2013, February. CRY—An Improved Crop Yield Prediction Model Using Bee Hive Clustering Approach for Agricultural Data Sets. In *Pattern Recognition, informatics and Mobile Engineering (PRIME), 2013 International Conference on* (pp.473-478). IEEE.

Ang, M. C.; Pham, D. T. and Ng, K. W., 2009. Minimum-Time Motion Planning for a Robot Arm Using the Bees Algorithm. In *7th IEEE International Conference on industrial informatics (INDIN 2009), June 2009*, pp. 487-492.

Ang, M. C.; Pham, D. T.; Soroka, A. J. and Ng, K. W., 2010. PCB Assembly Optimisation Using the Bees Algorithm Enhanced with TRIZ Operators. In *IECON 2010-36th Annual Conference on IEEE industrial Electronics Society, November 2010*, pp. 2708-2713.

Ang, M.C., Ng, K.W. and Pham, D.T., 2013. Combining the Bees Algorithm and Shape Grammar to Generate Branded Product Concepts. *Proceedings of the institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, p.0954405413494922.

Angus, D., 2006. Niching for Population-Based Ant Colony Optimization. In *Second IEEE International Conference on e-Science and Grid Computing (e-Science'06), December 2006*, pp. 115-115.

Archana, C. and Rejith, K.N., 2014a, December. The Use of Bees Algorithm for RA and MA Based Resource Allocation in OFDMA. In *Computational Systems and Communications (ICCSC), 2014 First International Conference on* (pp. 339-343). IEEE.

Archana, C. and Rejith, K.N., 2014b. Rate Adaptive Resource Allocation in OFDMA Using Bees Algorithm. *IJRET: International Journal of Research in Engineering and Technology, Volume: 3, Special Issue: 15*, pp.14-18.

Arefi, A.; Haghifam, M. R.; Fathi, S. H.; Niknam, T. and Olamei, J., 2009. A Novel Algorithm Based on Honey Bee Mating Optimisation for Distribution Harmonic State Estimation including Distributed Generators. In *Proceedings of IEEE Bucharest Power/ Tech Conference. Bucharest, 2009*, pp.1-7.

Arzeha, N.A., Mustafa, M.W. and Mohamed Idris, R., 2015, August. Lead Lag Controller of TCSC Optimized by Bees Algorithm for Damping Low Frequency Oscillation Enhancement in SMIB. In *Applied Mechanics and Materials* (Vol. 781, pp.374-378). Trans Tech Publications.

Assareh, E. and Biglari, M., 2015. A Novel Approach to Capture the Maximum Power Generation from Wind Turbines Using Hybrid MLP Neural Network and Bees Algorithm (HNNBA). *IETE Journal of Research*, pp.1-11.

Assareh, E., Behrang, M.A., Ghalambaz, M., Noghrehabadi, A.R. and Ghanbarzadeh, A., 2011. A New Approach to Solve Blasius Equation using Parameter Identification of Nonlinear Functions based on the Bees Algorithm (BA). *World Academy of Science, Engineering and Technology*, *49*, pp.1109-1111.

Atashpaz-Gargari, E. and Lucas, C., 2007, September. Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on* (pp. 4661-4667). IEEE.

Attaran, B. and Ghanbarzadeh, A., 2014. Bearing Fault Detection Based on Maximum Likelihood Estimation and Optimized ANN Using the Bees Algorithm. *Journal of Applied and Computational Mechanics*, *1*(1), pp.35-43.

Attaran, B., Ghanbarzadeh, A. and Ansari-Asl, K., 2012. Two New Feature Extraction Methods for Fault Diagnosis Based on Optimized Neural Networks and the Bees Algorithm. In *6th Condition Monitoring & Fault Diagnosis Conference, February 2012*, Sharif University of Technology, Tehran, Iran.

Aungkulanon, P. and Luangpaiboon, P., 2012. Stochastic Search Mechanisms on the Bee Algorithm for Optimising Noisy Multi-Response Surfaces. *Lecture Notes in Management Science*, 4: 154-164.

Aydogdu, I. and Akin, A., 2011. Bees Algorithm Based Optimum Design of Open Canal Sections. *International Journal of Engineering and Applied Sciences (IJEAS) Vol.3, Issue 4*, pp.21-31.

Azarbad, M., Azami, H., Sanei, S. and Ebrahimzadeh, A., 2014. New Neural Network-based Approaches for GPS GDOP Classification based on Neuro-Fuzzy inference System, Radial Basis Function, and Improved Bee Algorithm. *Applied Soft Computing*, *25*, pp.285-292.

Azarbad, M.; Ebrahimzade, A., and Izadian, V., 2011. Segmentation of infrared Images and Objectives Detection Using Maximum Entropy Method Based on the Bee Algorithm. *International Journal of Computer information Systems and industrial Management Applications (IJCISIM)*, 3: 026-033.

Azzeh, M., 2011. Software Effort Estimation Based on Optimized Model Tree. In *ACM Proceedings of the 7th International Conference on Predictive Models In Software Engineering, Canada, 20-21 September 2011*. [Online] Available from http://dl.acm.org [Accessed 1 March 2012].

Azzeh, M., Elsheikh, Y. and Alseid, M., 2014. An Optimized Analogy-Based Project Effort Estimation. *(IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 5, No. 4*, pp.6-11.

Bahamish, H. A. A.; Abdullah, R. and Abu-Hashem, Muhammad A., 2010. A Modified Marriage in Honey Bee Optimisation (MBO) Algorithm for Protein Structure Prediction. In *Proceedings of IEEE 2nd International Conference on Computer Technology and Development*, pp.65-69.

Bahamish, H. A. A.; Abdullah, R. and Salam, R. A., 2008. Protein Conformational Search Using Bees Algorithm. In *IEEE Second Asia International Conference on Modeling & Simulation (AICMS 08), May 2008,* pp.911-916.

Bahrainian, S.S., Mehrdoost, Z. and Ghanbarzadeh, A., 2013. The Application of Bees Algorithm in Finding the Neutral Stability Curve for Plane Poiseuille Flow. *Meccanica*, *48*(9), pp.2255-2261.

Bang, J.; Yoo, S.; Lim, J.; Ryu, J.; Lee, C. and Lee, J., 2010. Quantum Heuristic Algorithm for Traveling Salesman Problem. *arVix* [Online]. Available from: http://arxiv.org/abs/1004.4124 [Accessed 5 June 2012].

Banks, A.; Vincent, J. and Anyakoha, C., 2007. A Review of Particle Swarm Optimization. Part I: Background and Development. *Journal of Natural Computing*, 6(4): 467-484

Banks, A.; Vincent, J. and Phalp, K., 2009. Natural Strategies for Search. *Journal of Natural Computing*, 8: 547-570.

Banooni, S., Zarea, H. and Molana, M., 2014. Thermodynamic and Economic Optimization of Plate Fin Heat Exchangers Using the Bees Algorithm. *Heat Transfer—Asian Research*, *43*(5), pp.427-446.

Baykasoglu, A.; Ozbakir, L. and Tapkan, P., 2007. Artificial bee colony algorithm and its application to generalized assignment problem. *Swarm intelligence: Focus on ant and particle swarm optimization* [Online], Available from http://www.intechopen.com [Accessed 1 March 2012].

Beasley, D.; Bull, D. R. and Martin, R. R., 1993. A Sequential Niche Technique for Multimodal Function Optimization. *Journal of Evolutionary Computation*, 1(2): 101-125.

Belaid, I., Ouni, B., Muller, F. and Benjemaa, M., 2013. Complete and Approximate Methods for off-line Placement of Hardware Tasks on Reconfigurable Devices. *Journal of Circuits, Systems, and Computers*, *22*(02), p.1250080.

Bera, S.; Barman, G. and Mukherjee, I., 2011. Effect of Seemingly Unrelated Regression-Based Modeling Approach on Solution Quality for Correlated Multiple Response Optimization Problems. In *2011 IEEE International Conference on industrial Engineering and Engineering Management (IEEM), December 2011*, pp.1490-1494.

Bernardino, A.M., Bernardino, E.M., Sánchez-Pérez, J.M., Gómez-Pulido, J.A. and Vega-Rodríguez, M.A., 2011. Efficient Load Balancing Using the Bees Algorithm. In *Modern Approaches in Applied intelligence* (pp. 469-479). Springer Berlin Heidelberg.

Bernardino, E.M., Bernardino, A.M., Sánchez-Pérez, J.M., Gómez-Pulido, J.A. and Vega-Rodríguez, M.A., 2012. Solving Large-Scale SONET Network Design Problems Using Bee-inspired Algorithms. *Optical Switching and Networking*, *9*(2), pp.97-117.

Bernardino, Eugénia M.; Bernardino, Anabela M.; Sánches-Pérez, Juan Mannuel; Gómez-Pulido, Juan Antonio and Vega-Rodríguez, Miguel Angel, 2010. Hybrid Honey Bees Mating Optimisation Algorithms to Assign Terminal to Concentrators. In *Proceedings of IEEE 3rd International Symposium on Applied Sciences in Biomedical and Communication Technologies,* pp.1-7.

Bhasaputra, P., Sumeth Anantasate, and Woraratana Pattaraprakorn, 2011. Multiobjective Bees Algorithm for Optimal Power Flow Problem. *ECTI Transactions on Electrical Eng., Electronics, and Communications, 9(1)*, pp.56-64.

Bianchi, Leonora; Dorigo, Marco; Gambardella, Luca M. and Gutjahr, Walter J., 2009. A Survey on Metaheuristics for Stochastic Combinatorial Optimization. *Journal of Natural Computing*, 8:239-287.

Biglari, M., Valipour, M.S., Assareh, E., Nedaei, M., Poultangari, I., Branch, D. and Dezful, I., 2013. A New Solution for Natural Convection of Darcian Fluid About A Vertical Full Cone Embedded In Porous Media Prescribed Wall Temperature. *Global Journal of Science, Engineering and Technology, Issue 11*, pp.10-18.

Binitha, S. and Sathya, S. Siva, 2012. A Survey of Bio inspired Optimization Algorithms. *International Journal of Soft Computing and Engineering*, 2: 2231-2307.

Blum, Christian; Puchinger, Jakob; Raidl, Günther R. and Roli, andrea, 2011. Hybrid Metaheuristics in Combinatorial Optimization: A Survey. *Journal of Applied Soft Computing*, 11: 4135–4151.

Blum, Cristian, 2005. Ant Colony Optimization: Introduction and Recent Trends. *Physics of Life Reviews*, 2: 353–373.

Bonab, M.B. and Mohd Hashim, S.Z., 2014, December. Improved K-Means Clustering with Harmonic-Bee Algorithms. In *information and Communication Technologies (WICT), 2014 Fourth World Congress on* (pp. 332-337). IEEE.

Bonab, M.B., Hashim, S.Z.M., Bazin, N.E.N. and Alsaedi, A.K.Z., 2015. An Effective Hybrid of Bees Algorithm and Differential Evolution Algorithm in Data Clustering. *Mathematical Problems in Engineering, 501*, P.240419.

Bonabeau, E., Dorigo, M. And Theraulaz, G., 1999. *Swarm Intelligence: From Natural to Artificial Systems* (No. 1). Oxford University Press.

Bonyadi, M.R. and Michalewicz, Z., 2014, July. SPSO 2011: Analysis Of Stability; Local Convergence and Rotation Sensitivity. In *Proceedings of the 2014 conference on Genetic and evolutionary computation* (pp. 9-16). ACM.

Boumazouza, D., Sefouane, Y., Djeddi, M., Khelouat, B. and Benatchba, K., 2013, November. Bees for Block Matching. In *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE* (pp. 2390-2394). IEEE.

Bradford Jr., David and Hung, Chih-Cheng, 2012. Pollen-Based Bee Algorithm for Data Clustering - A Computational Model, *PICA: Progress In intelligent Computing and Applications, 1(1),* pp.16-36.

Braiwish, N.Y., Anayi, F.J., Fahmy, A. and Eldukhri, E.E., 2014, September. Design Optimisation of Permanent Magnet Synchronous Motor for Electric Vehicles Traction Using the Bees Algorithm. In *Power Engineering Conference (UPEC), 2014 49th International Universities* (pp. 1-5). IEEE.

Braiwish, N.Y., Anayi, F.J., Fahmy, A.A. and Eldukhri, E.E., 2015, March. Design Optimization Comparison of BLPM Traction Motor Using Bees and Genetic Algorithms. In *Industrial Technology (ICIT), 2015 IEEE International Conference on* (pp. 702-707). IEEE.

Brits, R.; Engelbrecht, A. P. and Van Den Bergh, F., 2007. Locating Multiple Optima Using Particle Swarm Optimization. *Journal of Applied Mathematics and Computation*, 189(2): 1859-1883.

Brownlee, Jason, 2011. *Clever Algorithms: Nature-inspired Programming Recipes* [Online]. 1st Ed. Australia: Lulu Enterprise. Available from: http://www.cleveralgorithms.com [Accessed 31 January 2012].

Bäck, Thomas, Schwefel, Hans-Paul, 1993. An Overview of Evolutionary Algorithms for Parameter Optimization. *Journal of Evolutionary Computation*, L(1): 1-23.

Cabrera G, G., Cabrera, E., Soto, R., Rubio, L., Crawford, B. and Paredes, F., 2012. A Hybrid Approach Using an Artificial Bee Algorithm with Mixed Integer Programming Applied to a Large-Scale Capacitated Facility Location Problem. *Mathematical Problems in Engineering*.

Campelo, F.; Guimarães, F. G.; Igarashi, H.; Ramírez, J. A. and Noguchi, S., 2006. A Modified Immune Network Algorithm for Multimodal Electromagnetic Problems. *IEEE Transactions on Magnetics*, 42(4): 1111-1114.

Castellani, M.; Pham, Q. T. and Pham, D. T., 2012. Dynamic Optimisation by a Modified Bees Algorithm. *Proceedings of the institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 226(7): 956-971.

Chai-Ead, N.; Aungkulanon, P. and Luangpaiboon, P., 2011. Bees and Firefly Algorithms for Noisy Non-Linear Optimization Problems. In *Proceedings of The International Multi Conference of Engineering and Computer Scientists (IMECS 2011), Vol.2, Hong Kong, 16-18 March 2011*. [Online] Available from http://iaeng.org [Accessed 1 March 2012].

Chen, H.; Zhu, Y. and Hu, K., 2009. Cooperative Bacterial foraging Optimization. *Journal of Discrete Dynamics in Nature and Society* [Online], Available from: http://www.hindawi.com [Accessed 9 September 2012].

Chen, Y., Ma, H.W. and Zhang, G.M., 2014. A Support Vector Machine Approach for Classification of Welding Defects from Ultrasonic Signals. *Nondestructive Testing and Evaluation*, *29*(3), pp.243-254.

Cheng, M.Y. and Lien, L.C., 2012. Hybrid Artificial intelligence–Based PBA for Benchmark Functions and Facility Layout Design Optimization. *Journal of Computing in Civil Engineering*, *26*(5), pp.612-624.

Chmiel, W. and Szwed, P., 2016. Bees Algorithm for the Quadratic Assignment Problem on CUDA Platform. In *Man–Machine interactions 4* (pp. 615-625). Springer International Publishing.

Cho, J. H.; Park, J.; Jeong, J. S. and Chun, M. G., 2009. Bacterial foraging with Quorum Sensing Based Optimization Algorithm. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2009), August 2009,* pp.29-34.

Chong, C. S.; Low, M. Y. H.; Sivakumar, A. I. and Gay, K. L., 2006. A Bee Colony Optimization Algorithm to Job Shop Scheduling. In *IEEE Proceedings of the Winter Simulation Conference (WSC 06), December 2006,* pp.1954-1961.

Choon, Y.W., Mohamad, M.S., Deris, S., Chong, C.K., Chai, L.E., Ibrahim, Z. and Omatu, S., 2012. Identifying Gene Knockout Strategies Using a Hybrid of Bees Algorithm and Flux Balance Analysis for in Silico Optimization of Microbial Strains. In *Distributed Computing and Artificial intelligence* (pp. 371-378). Springer Berlin Heidelberg.

Choon, Y.W., Mohamad, M.S., Deris, S., Chong, C.K., Omatu, S. and Corchado, J.M., 2015. Gene Knockout Identification Using an Extension of Bees Hill Flux Balance Analysis. *Biomed Research International*.

Choon, Y.W., Mohamad, M.S., Deris, S., Illias, R.M., Chai, L.E. and Chong, C.K., 2013a. Identifying Gene Knockout Strategy Using Bees Hill Flux Balance Analysis (BHFBA) for Improving the Production of Ethanol in Bacillus Subtilis. In *Advances in Biomedical infrastructure 2013* (pp. 117-126). Springer Berlin Heidelberg.

Choon, Y.W., Mohamad, M.S., Deris, S., Illias, R.M., Chong, C.K. and Chai, L.E., 2014a. A Hybrid of Bees Algorithm and Flux Balance Analysis with OptKnock as a Platform for In Silico Optimization of Microbial Strains. *Bioprocess and Biosystems Engineering*, *37*(3), pp.521-532.

Choon, Y.W., Mohamad, M.S., Deris, S., Illias, R.M., Chong, C.K., Chai, L.E., Omatu, S. and Corchado, J.M., 2014b. Differential Bees Flux Balance Analysis with OptKnock for In Silico Microbial Strains Optimization. *Plos ONE 9(7): E102744.* Doi:10.1371/Journal.Pone.0102744.

Choon, Y.W., Mohamad, M.S.B., Deris, S., Illias, R.M., Chai, L.E. and Chong, C.K., 2013b. Using Bees Hill Flux Balance Analysis (BHFBA) for In Silico Microbial Strain Optimization. In *intelligent information and Database Systems* (pp. 375-384). Springer Berlin Heidelberg.

Chrysostomou, D. and Gasteratos, A., 2012, July. Optimum Multi-Camera Arrangement Using A Bee Colony Algorithm. In *Imaging Systems and Techniques (IST), 2012 IEEE International Conference on* (pp. 387-392). IEEE.

Clerc, Maurice, 2012. *Standard Particle Swarm Optimisation*: Particle Swarm Central, Technical Report, 2012, [Online]. Available from http://clerc.maurice.free.fr/pso/SPSO descriptions.pdf [Accessed 19 August 2014].

Çoban, R. and Erçđn, Ö. 2012. Multi-Objective Bees Algorithm To Optimal Tuning of PID Controller. *Cukurova University Journal of the Faculty of Engineering and Architecture*, *27*(2), pp.13-26.

Coelho, G. P. and Von Zuben, F. J., 2010. A Concentration-Based Artificial Immune Network for Continuous Optimization. In *2010 IEEE Congress on Evolutionary Computation (CEC)*, July 2010, pp.1-8.

Consoli, S., 2006. *Combinatorial Optimization and Metaheuristics*: Operational Research Report. School of information Systems, Computing and Mathematics, Brunel University, UK.

Crossley, M.; Nisbet, A. and Amos, M., 2013. Quantifying the Impact of Parameter Tuning on Nature-Inspired Algorithms. In *European Conference on Artificial Life (ECAL)*, September 2013, pp. 925-932.

Cuevas, E., Zaldívar, D. and Pérez-Cisneros, M., 2013. A Swarm Optimization Algorithm for Multimodal Functions and Its Application in Multicircle Detection. *Mathematical Problems In Engineering*, *2013*.

Danaei, H. and Khajezadeh, A., 2015. Optimal Design of PID Controller Using New Version of Bee's Algorithm for Quarter-Car Active Suspension System. *Academie Royale Des Sciences D Outre-Mer Bulletin Des Seances*, *4*(4), pp.119-125.

Darwish, R.R., 2013. Autonomic Power Aware Cloud Resources Orchestration Architecture for Web Applications. *International Journal of Grid and Distributed Computing*, *6*(6), pp.63-82.

Das, S. and Suganthan, P. N., 2011. Differential Evolution: A Survey of the State-of-The-Art. *IEEE Transactions on Evolutionary Computation*, 15(1): 4-31.

Das, S.; Maity, S.; Qu, B. Y. and Suganthan, P. N., 2011. Real-Parameter Evolutionary Multimodal Optimization—A Survey of the State-of-The-Art. *Journal of Swarm and Evolutionary Computation,* 1(2): 71-88.

Dasgupta, D.; Yu, S. and Nino, F., 2011. Recent Advances in Artificial Immune Systems: Models and Applications. *Journal of Applied Soft Computing*, 11(2): 1574-1587.

De Castro, L. and Timmis, J. I., 2003. Artificial Immune Systems as a Novel Soft Computing Paradigm. *Journal of Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 7(8): 526-544.

De Castro, L. N. and Timmis, J., 2002. An Artificial Immune Network for Multimodal Function Optimization." In *IEEE Proceedings of the Evolutionary Computation (CEC'02). May 2002*, pp.699-704.

De Castro, L. N. and Von Zuben, F. J., 2002. Learning and Optimization Using the Clonal Selection Principle. *IEEE Transactions on Evolutionary Computation*, 6(3): 239-251.

De Castro, L.N. and Von Zuben, F.J., 2000, July. The Clonal Selection Algorithm with Engineering Applications. In *Proceedings of GECCO* (Vol. 2000, pp. 36-39).

De Castro, L.N. and Von Zuben, F.J., 2001. AiNet: An Artificial Immune Network for Data Analysis. *Data Mining: A Heuristic Approach*, *1*, pp.231-259.

De França, F. O.; Coelho, G. P. and Von Zuben, F. J., 2010. On the Diversity Mechanisms of Opt-AiNet: A Comparative Study with Fitness Sharing. In *2010 IEEE Congress on Evolutionary Computation (CEC), July 2010*, pp.1-8.

Deb, K. and Saha, A., 2010. Finding Multiple Solutions for Multimodal Optimization Problems Using a Multi-Objective Evolutionary Approach. In *ACM Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, July 2010*, pp.447-454.

Dehuri, S.; Cho, S. B. and Jagadev, A. K., 2008. Honey Bee Behavior: A Multi-Agent Approach for Multiple Campaigns Assignment Problem. In *IEEE International Conference on information Technology (ICIT'08), December 2008*, pp.24-29.

Delvecchio, G., Lofrumento, C., Neri, F. and Sylos Labini, M., 2006. A Fast Evolutionary-Deterministic Algorithm to Study Multimodal Current Fields under Safety Level Constraints. *COMPEL-The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, *25*(3), pp.599-608.

Derakhshan, M. and Shirazi, K.H., 2014. Optimized Fuzzy Controller for A Power–Torque Distribution In A Hybrid Vehicle with A Parallel Configuration. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, P.0954407013496183.

Dereli, T. and Das, G. S., 2011. A Hybrid 'Bee (S) Algorithm for Solving Container Loading Problems. *Journal of Applied Soft Computing*, 11(2): 2854-2862.

Dhote, C.A., Thakare, A.D. and Chaudhari, S.M., 2013, July. Data Clustering Using Particle Swarm Optimization and Bee Algorithm. In *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on* (pp. 1-5). IEEE.

Dhurandher, S. K.; Singhal, S.; Aggarwal, S.; Pruthi, P.; Misra, S. and Woungang, I., 2009. A Swarm intelligence-Based P2P File Sharing Protocol Using Bee Algorithm. In *IEEE/ACS International Conference on Computer Systems and Applications (AICCSA 2009), May 2009,* pp.690-696.

Dietrich, J. M. and Hartke, B., 2012. Empirical Review of Standard Benchmark Functions Using Evolutionary Global Optimisation. *Journal of Applied Mathematics*, 3(10A): 1552-1564.

Dieu, V. N.; Schegner, P. And Ongsakul, W., 2011. A Newly Improved Particle Swarm Optimization for Economic Dispatch with Valve Point Loading Effects. In *IEEE Power and Energy Society General Meeting*, July 2011, Pp. 1-8.

Dilettoso, E. and Salerno, N., 2006. A Self-Adaptive Niching Genetic Algorithm for Multimodal Optimization of Electromagnetic Devices. *Magnetics, IEEE Transactions on*, *42*(4), pp.1203-1206.

Dimopoulos, Christos and Zalzala, Ali M. S., 2000. Recent Developments in Evolutionary Computation for Manufacturing Optimization: Problems, Solutions, and Comparisons. *IEEE Transactions on Evolutionary Computation,* 4(2): 93-113.

Diwold, Konrad; Beekman, Madeleine and Middendorf, Martin Middendorf, 2010. Bee Nest Site Selection as an Optimization Process. In *Proceedings of the 12th Alife Conference, Odense, 2010*, pp.626-633.

Diwold, Konrad; Beekman, Madeleine and Middendorf, Martin Middendorf, 2011a. Honeybee Optimisation – An Overview and a New Bee inspired Optimisation Scheme. In *Handbook of Swarm intelligence.* Berlin Heidelberg: Springer-Verlag, pp.295-327.

Diwold, Konrad; Himmelbach, Daniel; Meier, René; Baldauf, Carsten and Middendorf, Martin, 2011b. Bonding As A Swarm: Applying Bee Nest-Site Selection Behaviour To Protein Docking" In *The Genetic and Evolutionary Computation Conference (GECCO'11). Dublin, July 2011,* pp.93-100.

Dorigo, M. and Gambardella, L.M., 1997. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *Evolutionary Computation, IEEE Transactions on*, *1*(1), pp.53-66.

Dorigo, M. and Stützle, T., 2010. Ant Colony Optimization: Overview and Recent Advances. In *Handbook of Metaheuristics, International Series in Operations Research & Management Science Vol.146*, US: Springer Business and Economics.

Dorigo, M., Di Caro, G. and Gambardella, L.M., 1999. Ant Algorithms for Discrete Optimization. *Artificial Life*, *5*(2), pp.137-172.

Dorigo, M.; Birattari, M. and Stützle, T., 2006. Ant Colony Optimization: Artificial Ants as a Computational intelligence Technique. In *IEEE Computational intelligence Magazine, November 2006,* pp.28-39.

Dorigo, M.; Maniezzo, V. and Colorni, A., 1991. *The Ant System: An Autocatalytic Optimizing Process*: Technical Report 91-016. Dipartimento Di Elettronica, Politecnico Di Milano, Italy.

Downing, Keith L., 2010. Introduction to Evolutionary Algorithms [Online]. Available From: http://citeseerx.ist.psu.edu [Accessed 19 August 2012].

Drias, H.; Sadeg, S. and Yahi, S., 2005. Cooperative Bees Swarm for Solving the Maximum Weighted Satisfiability Problem. In *Computational intelligence and Bioinspired Systems, Lecture Notes in Computer Science*, Vol.3512, Berlin Heidelberg: Springer-Verlag, pp.417-448.

Düğenci, M., Aydemir, A., Esen, İ. and Aydın, M.E., 2015. Creep Modelling of Polypropylenes Using Artificial Neural Networks Trained with Bee Algorithms. *Engineering Applications of Artificial intelligence*, *45*, pp.71-79.

Eberhart, R. and Kennedy, J., (1995. A New Optimizer Using Particle Swarm Theory. *In IEEE Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS'95), October 1995*, pp.39-43.

Ebrahimzadeh, A. and Mavaddati, S., 2014. A Novel Technique for Blind Source Separation Using Bees Colony Algorithm and Efficient Cost Functions. *Swarm and Evolutionary Computation*, *14*, pp.15-20.

Ebrahimzadeh, A., Addeh, J. and Ranaee, V., 2013. Recognition of Control Chart Patterns Using an intelligent Technique. *Applied Soft Computing*, *13*(5), pp.2970-2980.

Ebrahimzadeh, A., Shakiba, B. and Khazaee, A., 2014. Detection of Electrocardiogram Signals Using an Efficient Method. *Applied Soft Computing*, *22*, pp.108-117.

Eesa, A.S., Orman, Z. and Brifcani, A.M.A., 2015. A New Feature Selection Model Based on ID3 and Bees Algorithm for intrusion Detection System. *Turkish Journal of Electrical Engineering and Computer Science*, *23*(2), pp.615-622.

El-Abd, Mohammed, 2012. Performance Assessment of Foraging Algorithms *vs*. Evolutionary Algorithms. *Journal of information Sciences,* 182: 243–263.

Eldukhri, E.E. and Kamil, H.G., 2013. Optimisation of Swing-Up Control Parameters for a Robot Gymnast Using the Bees Algorithm. *Journal of intelligent Manufacturing*, pp.1-9.

Eldukhri, E.E. and Pham, D.T., 2010. Autonomous Swing-Up Control of a Three-Link Robot Gymnast. *Proceedings of The institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *224*(7), pp.825-832.

Engelbrecht, A. P., 2007. *Computational intelligence: An introduction*. New Jersey: John Wiley & Sons

Ercin, O. and Coban, R., 2011, June. Comparison of the Artificial Bee Colony and the Bees Algorithm for PID Controller Tuning. In *2011 International Symposium on Innovations in intelligent Systems and Applications*.

Erol, O.K. and Eksin, I., 2006. A New Optimization Method: Big Bang–Big Crunch. *Advances in Engineering Software*, *37*(2), pp.106-111.

Eslami, Mahdiyeh; Shareef, Hussain; Khajehzadeh, Mohammad and Mohamed, Azah, 2012. A Survey of the State of The Art in Particle Swarm Optimization. *Research Journal of Applied Sciences, Engineering and Technology*, 4(9): 1181-1197.

Eslami, M.; Shareef, H.; Mohamed, A. And Khajehzadeh, M., 2013. Gradient-Based Artificial Bee Colony Algorithm for Damping Controllers Design. In *IEEE 7th International Power Engineering and Optimisation Conference (PEO2013)*, pp. 293-297.

Fahmy, A. A., 2012. Using the Bees Algorithm to Select the Optimal Speed Parameters for Wind Turbine Generators. *Journal of King Saud University-Computer and information Sciences*, 24: 17-26.

Fahmy, A. A.; Kalyoncu, M. and Castellani, M., 2012. Automatic Design of Control Systems for Robot Manipulators Using the Bees Algorithm. *Proceedings of the institution of Mechanical Engineers*, *Part I: Journal of Systems and Control Engineering*, 226(4): 497-508.

Fahmy, A.A. and Ghany, A.A., 2015. Adaptive Functional-Based Neuro-Fuzzy PID incremental Controller Structure. *Neural Computing and Applications*, pp.1-16.

Far, Hussein Khosravy and Aghazadeh, F., 2015. Dynamic Cellular Learning Bees Algorithm. *International Journal of Research in Management & Technology, 5(1),* pp.180-185.

Farhan, A. A. and Bilal, S., 2011. A Novel Fast and Robust Digital Image Watermarking Using Bee Algorithm. In *2011 IEEE 14th International Multitopic Conference (INMIC), December 2011*, pp.82-86.

Farhang, P. and Mazlumi, K., 2014. Low-Frequency Power System Oscillation Damping Using HBA-Based Coordinated Design of IPFC and PSS Output Feedback Controllers. *Transactions of the institute of Measurement and Control*, *36*(2), pp.184-195.

Fathallahnezhad, A. and Eslami, M., 2015. Optimization of Location and Size Value of DGs in Power Network Using Bee's Algorithm. *Cumhuriyet Science Journal*, *36*(6), pp.51-57.

Fathian, Mohammad; Amiri, Babak and Maroosi, Ali, 2007. Application of Honey-Bee Mating Optimisation Algorithm on Clustering. *Journal of Applied Mathematics and Computation*, 190: 1502-1513.

Fernandes, E. M. D. G., Martins, T. F., & Rocha, A. M. A., 2009. Fish Swarm intelligent Algorithm for Bound Constrained Global Optimization. In *Proceedings of International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE 2009)* [Online], Gijón, Spain. Available from http://hdl.handle.net/1822/9673 [Accessed 1 October 2012].

Fon, C. W. and Wong, K. Y., 2010. Investigating the Performance of Bees Algorithm in Solving Quadratic Assignment Problems. *International Journal of Operational Research*, 9(3): 241-257.

Formato, R.A., 2007. Central Force Optimization: A New Metaheuristic with Applications in Applied Electromagnetics. *Progress In Electromagnetics Research*, *77*, pp.425-491.

Forrest, S.; Perelson, A. S.; Allen, L. and Cherukuri, R., 1994. Self-Nonself Discrimination in a Computer. In *IEEE Computer Society Symposium on Research in Security and Privacy, May 1994*. pp. 202-212.

Furlan, M.M. and Santos, M.O., 2015. BFO: A Hybrid Bees Algorithm for The Multi-Level Capacitated Lot-Sizing Problem. *Journal of intelligent Manufacturing*, pp.1-16.

Gang, H.; Yuanming, L. and Jinhang, L., 2011. Lévy Flight Search Patterns in Particle Swarm Optimization. In *IEEE 7th International Conference on Natural Computation (ICNC), July 2011. Vol.2*. pp. 1185-1189.

Gao, X.Z., Wang, X. and Zenger, K., 2015. A Memetic-inspired Harmony Search Method in Optimal Wind Generator Design. *International Journal of Machine Learning and Cybernetics*, *6*(1), pp.43-58.

Gao, X.Z., Wang, X., Zenger, K. and Wang, X., 2012, October. A Bee foraging-Based Memetic Harmony Search Method. In *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on* (pp. 184-189). IEEE.

Geem, Zong W.; Kim, Joong H. and Loganathan, G.V., 2001. A New Heuristic Optimization Algorithm: Harmony Search. *Simulation*, 76(2): 60-68.

Ghaedi, A.M., Ghaedi, M., Vafaei, A., Iravani, N., Keshavarz, M., Rad, M., Tyagi, I., Agarwal, S. and Gupta, V.K., 2015a. Adsorption of Copper (II) Using Modified Activated Carbon Prepared From Pomegranate Wood: Optimization by Bee Algorithm and Response Surface Methodology. *Journal of Molecular Liquids*, *206*, pp.195-206.

Ghaedi, M., Ansari, A., Assefi Nejad, P., Ghaedi, A., Vafaei, A. and Habibi, M.H., 2015b. Artificial Neural Network and Bees Algorithm for Removal of Eosin B Using Cobalt Oxide Nanoparticle-Activated Carbon: Isotherm and Kinetics Study. *Environmental Progress & Sustainable Energy*, *34*(1), pp.155-168.

Ghanbarzadeh, A., 2010, January. Neural Network Weight Optimisation Using the Bees Algorithm. In *ASME 2010 10th Biennial Conference on Engineering Systems Design and Analysis* (pp. 849-854). American Society of Mechanical Engineers.

Ghasemi, B., Sadeghi, A., Roghani, M., Researchers, Y., Club, E. and Branch, S.T., 2015. The Solution of Multi-Objective Multimode Resource-Constrained Project Scheduling Problem (RCPSP) with Partial Precedence Relations by Multi-Objective Bees Algorithm. *Silvae Genetica*, *57*(1), pp.20-38.

Gholipour, R., Khosravi, A. and Mojallali, H., 2012. Bees Algorithm Based Intelligent Backstepping Controller Tuning for Gyro System. *The Journal of Mathematics and Computer Science*, *5(3)*, pp.205-211.

Gholipour, R., Khosravi, A. and Mojallali, H., 2015. Multi-Objective Optimal Backstepping Controller Design for Chaos Control in a Rod-Type Plasma Torch System Using Bees Algorithm. *Applied Mathematical Modelling*, 39, pp.4432-4444.

Gholipour, R., Khosravia, A. and Mojallali, H., 2012. Parameter Estimation of Loranz Chaotic Dynamic System Using Bees Algorithm. *International Journal of Engineering-Transactions C: Aspects*, *26*(3), pp.257-262.

Ghosha, Tamal; Senguptaa, Sourav; Chattopadhyayb, Manojit and Dana, Pranab K., 2011. Meta-Heuristics in Cellular Manufacturing: A State-of-The-Art Review. *International Journal of industrial Engineering Computations*, 2: 87–122

Glover, F., 1986. Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers & Operations Research*, *13*(5), pp.533-549.

Gorkemli, B. and Karaboga, D., 2013, September. Quick Combinatorial Artificial Bee Colony –qCABC Optimization Algorithm for TSP. In *2nd International Symposium on Computing in Informatics and Mathematics (ISCIM)*, Epoka University, Albania, pp.97-101.

Granovskiy, B.; Latty, T.; Duncan, M.; Sumpter, D. J. and Beekman, M., 2012. How Dancing Honey Bees Keep Track of Changes: The Role of inspector Bees. *Journal of Behavioral Ecology*, 23(3): 588-596.

Greensmith, J.; Aickelin, U. and Cayzer, S., 2005. Introducing Dendritic Cells as a Novel Immune-inspired Algorithm for Anomaly Detection. *In Artificial Immune Systems Lecture Notes in Computer Science*. Berlin Heidelberg: Springer-Verlag, pp.153-167.

Grega, M., Bryk, D. and Napora, M., 2014. INACT—INDECT Advanced Image Cataloguing Tool. *Multimedia Tools and Applications*, *68*(1), pp.95-110.

Guney, K. and Onay, M., 2007. Amplitude-Only Pattern Nulling of Linear Antenna Arrays with the Use of Bees Algorithm. *Progress in Electromagnetics Research*, 70: 21-36.

Guney, K. and Onay, M., 2010. Bees Algorithm for interference Suppression of Linear Antenna Arrays by Controlling the Phase-Only and Both the Amplitude and Phase. *Journal of Expert Systems with Applications*, 37(4): 3129-3135.

Guney, K. and Onay, M., 2011. Synthesis of Thinned Linear Antenna Arrays Using Bees Algorithm. *Microwave and Optical Technology Letters*, 53(4): 795-799.

Guney, K. and Onay, M., 2013. Bees Algorithm for interference Suppression of Linear Antenna Arrays by Controlling the Positions of Selected Elements. *Journal of Communications Technology and Electronics*, *58*(12), pp.1147-1156.

Häckel, S. and Dippold, P., 2009. The Bee Colony-inspired Algorithm (BCiA): A Two-Stage Approach for Solving the Vehicle Routing Problem with Time Windows. In *ACM Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, July 2009*. pp. 25-32.

Haddad, O. B. and Afshar, A., 2004. MBO (Marriage Bees Optimisation) A New Heuristic Approach In Hydrosystem Design and Operation. In *Proceedings of 1st International Conference on Managing Rivers In The 21st Century: Issues and Challenges. Penang, 2004,* pp.499-504.

Haddad, O. B.; Afshar, A. and Mariño, M. A., 2005. HBMO in Engineering Optimisation. In *Proceedings of 9th International Water Technology Conference, Sham El-Sheikh, 2005,* pp.1053-1063.

Hadi, B., Khosravi, A., Ranjbar, N. and Sarhadi, P., 2014. RISE Feedback Control Design for RLED Robot Manipulator Using Bees Algorithm. *Journal of Advances in Computer Research*, *5*(4), pp.53-65.

Hewlett, J.; Wilamowski, B. and Dundar, G., 2007, June. Merge of Evolutionary Computation with a Gradient-based Method for Optimization Problems. In *IEEE International Symposium on Industrial Electronics (ISIE 2007)*, June *2007,* pp. 3304-3309.

Holland, J.H., 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. U Michigan Press.

Hoorfar, A., 2007. Evolutionary Programming in Electromagnetic Optimisation: A Review. *IEEE Transactions on Antennas and Propagation*, 55(3): 523-537

Horng, Ming-Huwi and Jiang, Ting-Wei, 2010. A Multilevel Image Thresholding Using the Honey Bee Mating Optimisation. *Journal of Applied Mathematics and Computation*, 215: 3302-3310.

Houshmand, M.; Imani, D. M. and Niaki, S. T., 2009. Using Flower Pollinating with Artificial Bees (FPAB) Technique to Determine Machinable Volumes in Process Planning for Prismatic Parts. *The International Journal of Advanced Manufacturing Technology*, 45(9): 944-957.

Hussein, W.A., Sahran, S. and Abdullah, S.N.H.S., 2013. A New initialization Algorithm for Bees Algorithm. In *Soft Computing Applications and Intelligent Systems* (pp. 39-52). Springer Berlin Heidelberg.

Hussein, W.A., Sahran, S. and Abdullah, S.N.H.S., 2014. Patch-Levy-Based initialization Algorithm for Bees Algorithm. *Applied Soft Computing*, *23*, pp.104-121.

Idris, R. M.; Khairuddin, A. and Mustafa, M. W., 2009a Optimal Allocation of FACTS Devices for ATC Enhancement Using Bees Algorithm. *Journal of World Academy of Science, Engineering and Technology*, 54: 318-325.

Idris, R. M.; Kharuddin, A. and Mustafa, M. W., 2009b. Optimal Choice of FACTS Devices for ATC Enhancement Using Bees Algorithm. In *IEEE Australasian Universities Power Engineering Conference (AUPEC 2009), September 2009,* pp.1-6.

Idris, R.M., Khairuddin, A. and Mustafa, M.W., 2010a, May. A Parallel Bees Algorithm for ATC Enhancement in Modern Electrical Network. In *Mathematical/Analytical Modelling and Computer Simulation (AMS), 2010 Fourth Asia International Conference on* (pp. 450-455). IEEE.

Idris, R.M., Khairuddin, A. and Mustafa, M.W., 2010b. Optimal Allocation of FACTS Devices in Deregulated Electricity Market Using Bees Algorithm. *WSEAS Transactions on Power Systems*, *5*(2), pp.108-119.

Imanguliyev, A., 2013. *Enhancements for the Bees Algorithm* (Doctoral dissertation, Cardiff University).

Jamil, M. And Yang, X., 2013. A Literature Survey of Benchmark Functions for Global Optimisation Problems. *Journal of Numerical Optimisation*, 4(2): 150-194

Jana, N.D., Sil, J. and Das, S., 2015. Improved Bees Algorithm for Protein Structure Prediction Using AB off-Lattice Model. In *Mendel 2015* (pp. 39-52). Springer International Publishing.

Jiang, M.; Mastorakis, N. E.; Yuan, D. and Lagunas, M. A., 2009. Image Segmentation with Improved Artificial Fish Swarm Algorithm. In *Proceedings of The European Computing Conference*, pp.133-138. US: Springer.

Jiang, M.; Wang, Y.; Pfletschinger, S.; Lagunas, M. and Yuan, D., 2007. Optimal Multiuser Detection with Artificial Fish Swarm Algorithm. In *Advanced Intelligent Computing Theories and Applications with Aspects of Contemporary Intelligent Computing Techniques*, pp.1084-1093. Berlin Heidelberg: Springer.

Johal, N. K. and Singh, S., 2010. A Hybrid FPAB/BBO Algorithm for Satellite Image Classification. *International Journal of Computer Applications IJCA*, 6(5): 31-36.

Jones, K. O. and Bouffet, A., 2008. Comparison of Bees Algorithm, Ant Colony Optimisation and Particle Swarm Optimisation for PID Controller Tuning. In *ACM Proceedings of The 9th International Conference on Computer Systems and Technologies and Workshop for Phd Students In Computing [Online], June 2008*. Available from http://dl.acm.org [Accessed 1 March 2012].

Kalami, M.S., 2014. Electric Power Cable Fault Recognition via Combination of Wavelet Transform and Optimized Artificial Neural Network by Using Bees Algorithm. *International Journal of Mechatronics, Electrical and Computer Technology, 4(10)*, Special Number, pp.1112-1132.

Karaboga, D., 2005. *An Idea Based on Honey Bee Swarm for Numerical Optimization*: Technical Report TR06, Erciyes Univ. Press, Erciyes.

Karaboga, D. and Akay, B., 2009a. A Survey: Algorithms Simulating Bee Swarm intelligence. *Artificial Intelligence Review*, 31(1): 61-85.

Karaboga, D. and Akay, B., 2009b. Artificial Bee Colony (ABC), Harmony Search and Bees Algorithms on Numerical Optimization. In *Proceedings of Innovative Production Machines and Systems Virtual Conference (IPROMS)* [Online], Available From http://www.iproms.org [Accessed 1 March 2012].

Karaboga, D. and Gorkemli, B., 2012, July. A Quick Artificial Bee Colony-qABC-Algorithm for Optimization Problems. In *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on* (pp. 1-5). IEEE.

Karaboga, D. and Gorkemli, B., 2014. A Quick Artificial Bee Colony (qABC) Algorithm and Its Performance on Optimization Problems. *Applied Soft Computing*, *23*, pp.227-238.

Karaboga, D.; Gorkemli, B.; Ozturk, C. and Karaboga, N., 2012. A Comprehensive Survey: Artificial Bee Colony (ABC) Algorithm and Applications. Artificial intelligence Review [Online], Available From http://www.springerlink.com [Accessed 30 April 2012].

Karray, A., Teyeb, R. and Jemaa, M.B., 2013. A Heuristic Approach for Web-Service Discovery and Selection. *Arxiv Preprint Arxiv: 1305.2684*. *International Journal of Computer Science & information Technology (IJCSIT), 5(2),* pp.107-119.

Kataria, P. and Rupal, N., 2014. Mining Spatial Data and Enhancing Classification Using Bio-inspired Approaches. *International Journal of Science and Research (IJSR), 3(10)*, pp.1473-1479.
Kaveh A. and Talatahari, S., 2010. A Novel Heuristic Optimization Method: Charged System Search. *Journal of Acta Mechanica*, 213: 267–289.

Kavousi, A.; Vahidi, B.; Salehi, R.; Bakhshizadeh, M.; Farokhnia, N. and Fathi, S. S., 2012. Application of the Bee Algorithm for Selective Harmonic Elimination Strategy in Multilevel inverters. *IEEE Transactions on Power Electronics*, 27(4): 1689-1696.

Kazemi, M., Shirazi, K.H. and Ghanbarzadeh, A., 2012. Optimization of Semi-Trailing Arm Suspension for Improving Handling and Stability of Passenger Car. *Proceedings of the institution of Mechanical Engineers, Part K: Journal of Multi-Body Dynamics*, *226*(2), pp.108-121.

Kazemian, M.; Ramezani, Y.; Lucas, C. and Moshiri, B., 2006. Swarm Clustering Based on Flowers Pollination by Artificial Bees. In *Swarm Intelligence in Data Mining, Studies in Computational Intelligence*, Vol.34. Berlin Heidelberg: Springer-Verlag, pp.191-202.

Keshayeh, M.S. and Gholamian, S.A., 2013. Optimum Design of a Three-Phase Permanent Magnet Synchronous Motor for Industrial Applications. *International Journal of Applied Operational Research*, *2*(4), pp.67-86.

Keskin, T.E., Düğenci, M. and Kaçaroğlu, F., 2014. Prediction of Water Pollution Sources Using Artificial Neural Networks in the Study Areas of Sivas, Karabük and Bartın (Turkey). *Environmental Earth Sciences*, *73*(9), pp.5333-5347.

Khajehzadeh, A., 2015. Control Chart Patterns Detection by Optimized RBF and Frequency Features. *Academie Royale Des Sciences D Outre-Mer Bulletin Des Seances*, *4*(4), pp.47-54.

Khang, N.T.T.M., Phuc, N.B. and Nuong, T.T.H., 2011, June. The Bees Algorithm for a Practical University Timetabling Problem in Vietnam. In *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on* (Vol. 4, pp. 42-47). IEEE.

Khanmirzaei, Z. and Teshnehlab, M., 2010. Prediction Using Recurrent Neural Network Based Fuzzy inference System by The Modified Bees Algorithm. *IJACT: International Journal of Advancements In Computing Technology*, 2(2): 42-55.

Khazaei, S., Tahani, A., Yazdani-Asrami, M. and Gholamian, S.A., 2015. Optimal Design of Three Phase Surface Mounted Permanent Magnet Synchronous Motor by Particle Swarm Optimization and Bees Algorithm for Minimum Volume and Maximum Torque. *Journal of Advances in Computer Research*, *6*(2), pp.83-98.

Kong, X., Chen, Y.L., Xie, W. and Wu, X., 2012, June. A Novel Paddy Field Algorithm Based on Pattern Search Method. In *Information and Automation (ICIA), 2012 International Conference on* (pp. 686-690). IEEE.

Koo, C.L., Salleh, A.H.M., Mohamad, M.S., Deris, S., Omatu, S. and Yoshioka, M., 2014, October. A Gene Knockout Strategy for Succinate Production Using a Hybrid Algorithm of Bees Algorithm and Minimization of Metabolic Adjustment. In *Granular Computing (GRC), 2014 IEEE International Conference on* (pp. 131-136). IEEE.

Koper, K.D., Wysession, M.E. and Wiens, D.A., 1999. Multimodal Function Optimization with a Niching Genetic Algorithm: A Seismological Example. *Bulletin of the Seismological Society of America*, *89*(4), pp.978-988.

Korns, M. F., 2012. Predicting Corporate Forward 2 Month Earnings [Online], Available From http://www.intechopen.com [Accessed 1 October 2012].

Koza, J.R., 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (Vol. 1). MIT Press.

Krainyukov, A., Kutev, V. and Andreeva, E., 2014. Using Bees Algorithms for Solution of Radar Pavement Monitoring Inverse Problem. *Transport and Telecommunication*, *15*(1), pp.53-66.

Krishnanand, K. N. and Ghose, D., 2005. Detection of Multiple Source Locations Using a Glowworm Metaphor with Applications to Collective Robotics. In *IEEE Proceedings of Swarm intelligence Symposium (SIS 2005), June 2005*. pp. 84-91.

Krishnanand, K. N. and Ghose, D., 2009. Glowworm Swarm Optimization for Simultaneous Capture of Multiple Local Optima of Multimodal Functions. *Journal of Swarm intelligence*, 3(2): 87-124.

Krishnanand, K.R.; Nayak, Santanu K.; Panigrahi, B.K. and Rout, P.K., 2009. Comparative Study of Five Bio-inspired Evolutionary Optimization Techniques. In *IEEE World Congress on Nature and Biologically inspired Computing (Nabic 2009)*. *9-11 December 2009,* pp.1231-1236.

Kumar, Rajesh; Sadu, Abhinav; Kumar, Rudesh and Panda, S.K., 2012. A Novel Multi-Objective Directed Bee Colony Optimization Algorithm for Multi-Objective Emission Constrained Economic Power Dispatch. *Journal of Electrical Power and Energy Systems*, 43: 1241–1250.

Lara, C.; Flores, J. and Calderón, F., 2008. Solving a School Timetabling Problem Using a Bee Algorithm. In *MICAI 2008: Advances In Artificial intelligence, Lecture Notes in Computer Science*, Vol.5317. Berlin Heidelberg: Springer-Verlag, pp.664-674.

Lee, C. G.; Cho, D. H. and Jung, H. K. (1999). Niching Genetic Algorithm with Restricted Competition Selection for Multimodal Function Optimization. *IEEE Transactions on Magnetics*, 35(3): 1722-1725.
Lee, J. Y. and Darwish, A. H., 2008. Multi-Objective Environmental/Economic Dispatch Using the Bees Algorithm with Weighted Sum. In *EKC2008 Proceedings of the EU-Korea Conference on Science and Technology*, pp.267-274. Berlin Heidelberg: Springer.

Lee, J.S., Wang, J.W. and Giang, K.Y., 2014. A New Image Watermarking Scheme Using Multi-Objective Bees Algorithm. *Appl. Math, 8*(6), pp.2945-2953.

Leeprechanon, N. and Phonrattanasak, P., 2013, June. Bees Two-Hive Algorithm for Optimal Power Flow. In *Applied Mechanics and Materials* (Vol. 313, pp. 870-875).

Leeprechanon, N. and Polratanasak, P., 2010. "Multiobjective Bees Algorithm with Clustering Technique for Environmental/Economic Dispatch." In *IEEE 2010 International Conference on Electrical Engineering/Electronics Computer Telecommunications and information Technology (ECTI-CON), May 2010,* pp.621-625.

Lemmens, N.; De Jong, S.; Tuyls, K. and Nowé, A., 2008. "Bee Behaviour in Multi-Agent Systems." In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning, Lecture Notes in Computer Science*, Vol.4865, and Berlin Heidelberg: Springer-Verlag, pp.145-156.

Lenin, K., Reddy, B. Ravindranath and Kalavathi, M. Surya, 2014. Dwindling of Real Power Loss by Using Improved Bees Algorithm. *International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE), 1(1)*, pp.34-42.

Leong, Y.Y., Chong, C.K., Chai, L.E., Deris, S., Illias, R., Omatu, S. and Mohamad, M.S., 2013. Simulation of Fermentation Pathway Using Bees Algorithm. *ADCAIJ: Advances In Distributed Computing and Artificial intelligence Journal*, *1*(2), pp.13-19.

Li, H., Liu, K. and Li, N., 2010. Improved Bees Algorithm for The Large-Scale Layout Optimization without Performance Constraints. In *2010 IEEE Fifth International Conference on Bio-inspired Computing: Theories and Applications (BIC-TA)*, pp.459-463.

Li, H., Liu, K. and Li, X., 2010. A Comparative Study of Artificial Bee Colony, Bees Algorithms and Differential Evolution on Numerical Benchmark Problems. In *Computational intelligence and intelligent Systems* (pp. 198-207). Springer Berlin Heidelberg.

Li, J. P.; Li, X. D. and Wood, A., 2010. Species Based Evolutionary Algorithms for Multimodal Optimization: A Brief Review. In *2010 IEEE Congress on Evolutionary Computation (CEC), July 2010*, pp.1-8.

Li, M. S.; Ji, T. Y.; Wu, Q. H. and Xue, Y. S., 2010. Stochastic Optimal Power Flow Using a Paired-Bacteria Optimizer. In *IEEE Power and Energy Society General Meeting*, July 2010, pp. 1–7.

Li, X., 2007. A Multimodal Particle Swarm Optimizer Based on Fitness Euclidean-Distance Ratio. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, *July 2007. Vol.7, No.11*, pp.78-85.

Li, X., 2010. Niching without Niching Parameters: Particle Swarm Optimization Using a Ring Topology. *IEEE Transactions on Evolutionary Computation*, 14(1): 150-169.

Li, X. L.; Shao, Z. J. and Qian, J. X., 2002. An Optimizing Method Based on Autonomous Animals: Fish-Swarm Algorithm. *Journal of System Engineering Theory and Practice*, 11:32-38.

Liang, J.J., Suganthan, P.N. and Deb, K., 2005, June. Novel Composition Test Functions for Numerical Global Optimization. In *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE* (pp. 68-75). IEEE.

Liang, Y. and Leung, K. S., 2011. Genetic Algorithm with Adaptive Elitist-Population Strategies for Multimodal Function Optimization. *Journal of Applied Soft Computing*, 11(2): 2017-2034.

Lien, L. C. and Cheng, M. Y., 2012. A Hybrid Swarm Intelligence Based Particle-Bee Algorithm for Construction Site Layout Optimization. *Journal of Expert Systems with Applications*, 39(10): 9642-9650.

Lien, L.C. and Cheng, M.Y., 2014. Particle Bee Algorithm for Tower Crane Layout with Material Quantity Supply and Demand Optimization. *Automation in Construction*, *45*, pp.25-32.

Lin, Y.; Lin, C. And Shieh, N. Shieh, 2006. A Hybrid Evolutionary Approach for Robust Active Suspension Design of Light Rail Vehicles. *IEEE Transactions on Control Systems Technology*, 14(4): 695-706.

Liu, L.; Yang, S. and Wang, D., 2012. Force-imitated Particle Swarm Optimization Using the Near-Neighbour Effect for Locating Multiple Optima. *Journal of information Sciences*, 182(1): 139-155.

Liu, Y., Ling, X., Liang, Y., Lv, M. and Liu, G., 2012. Artificial Bee Colony (ABC) Algorithm for Multimodal Function Optimization. *Advanced Science Letters*, *11*(1), pp.503-506.

Liu, Y.; Ling, X.; Shi, Z.; Lu, M.; Fang, J. and Zhang, L., 2011. A Survey on Particle Swarm Optimization Algorithms for Multimodal Function Optimization. *Journal of Software*, 6(12): 2449-2455.

Long, V.T., 2015. Application of a Pheromone-Based Bees Algorithm for Simultaneous Optimisation of Key Component Sizes and Control Strategy for Hybrid Electric Vehicles. *International Journal of Swarm intelligence and Evolutionary Computation*, *4(1).*

Lu, W., Quan, Z., Liu, Q., Zhang, D. and Xu, W., 2015. QoE Based Spectrum Allocation Optimization Using Bees Algorithm In Cognitive Radio Networks In *Algorithms and Architectures for Parallel Processing* (pp. 327-338). Springer International Publishing.

Lu, X. and Zhou, Y., 2008. A Novel Global Convergence Algorithm: Bee Collecting Pollen Algorithm. In *Advanced intelligent Computing Theories and Applications with Aspects of Artificial intelligence*, *Lecture Notes In Computer Science*, Vol.5227, Berlin Heidelberg: Springer-Verlag, pp.518-525.

Luangpaiboon, P., 2011. Bee Parameter Determination via Weighted Centriod Modified Simplex and Constrained Response Surface Optimisation Methods. *World Academy of Science, Engineering and Technology*, *56*, pp.434-440.

Lucic, P. and Teodorović, D., 2001. Bee System: Modeling Combinatorial Optimization Transportation Engineering Problems by Swarm intelligence. In *Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis, June 2001,* pp.441-445.

Lucic, P. and Teodorović, D., 2003. Vehicle Routing Problem with Uncertain Demand at Nodes: The Bee System and Fuzzy Logic Approach. *Studies in Fuzziness and Soft Computing*, 126: 67-82.

Luo, G.H., Huang, S.K., Chang, Y.S. and Yuan, S.M., 2014. A Parallel Bees Algorithm Implementation on GPU. *Journal of Systems Architecture*, *60*(3), pp.271-279.

Mahmuddin, M. and Yusof, Y., 2009. A Hybrid Simplex Search and Bio-inspired Algorithm for Faster Convergence. In *2009 International Conference on Machine Learning and Computing*, pp.203-207.

Maia, R. D.; De Castro, L. N. and Caminhas, W. M., 2012. Bee Colonies as Model for Multimodal Continuous Optimization: The OptBees Algorithm. In *2012 IEEE Congress on Evolutionary Computation (CEC), June 2012*. pp. 1-8.

Majumdar, R.; Ghosh, A.; Das, A.; Raha, S.; Laha, K.; Das, S. and Abraham, A., 2012. Artificial Weed Colonies with Neighbourhood Crowding Scheme for Multimodal Optimization. In *Proceedings of The International Conference on Soft Computing for Problem Solving (Socpros 2011), 20-22 December 2011*. pp. 779-787. Berlin Heidelberg: Springer.

Malekian, R., Bogatinoska, D.C., Karadimce, A., Ye, N., Trengoska, J. and Nyako, W.A., 2015. A Novel Smart ECO Model for Energy Consumption Optimization. *Elektronika Ir Elektrotechnika*, *21*(6), pp.75-80.

Malekzadeh, M., Khosravi, A., Alighale, S. and Azami, H., 2012. Optimization of Orthogonal Poly Phase Coding Waveform Based on Bees Algorithm and Artificial Bee Colony for MIMO Radar. In *intelligent Computing Technology* (pp. 95-102). Springer Berlin Heidelberg.

Marghaki, R.S. and Zayandehroodi, H., 2015. Application of Bees' Algorithm for Optimal Distribution Generation Placement in Power Network. *Academie Royale Des Sciences D Outre-Mer Bulletin Des Seances*, *4*(3), pp.1-8.

Marinakis, Yannis and Marinaki, Magdalene, 2011. Bumble Bees Mating Optimisation Algorithm for the Vehicle Routing Problem. *Handbook of Swarm intelligence.* Berlin Heidelberg: Springer-Verlag, pp.347-369.

Marinakis, Yannis; Marinaki, Magdalene and Dourias, Georgios, 2010. Honey Bee Mating Optimisation for Vehicle Routing Problem. *Journal of Natural Computing*, 9 (1): 5-27.

Marinakis, Yannis; Marinaki, Magdalene and Dourias, Georgios, 2011. Honey Bee Mating Optimisation for Euclidean Traveling Salesman Problem. *Journal of information Science,* 181 (20): 4684-4698.

Marinakis, Yannis; Marinaki, Magdalene and Matsatsinis, Nikolaus, 2008. Honey Bee Mating Optimisation for Location Routing Problem. In *Proceedings of IEEE International Engineering Management Conference*, pp.1-5.

Marinakis, Yannis; Marinaki, Magdalene and Matsatsinis, Nikolaus, 2009a. Honey Bee Mating Optimisation for Probabilistic Traveling Salesman Problem. In *Proceedings of IEEE Congress on Evolutionary Computation*, pp.1762-1769.

Marinakis, Yannis; Marinaki, Magdalene and Matsatsinis, Nikolaus, 2009b. A Hybrid Bumble Bees Mating Optimisation-GRASP Algorithm for Clustering. In *Hybrid Artificial intelligence Systems Lecture Notes in Computer Science.* Berlin Heidelberg: Springer-Verlag, pp.549-556.

Marinakis, Yannis; Marinaki, Magdalene and Matsatsinis, Nikolaus, 2010. A Bumble Bees Mating Optimisation Algorithm for Global Unconstrained Optimisation Problems. In *Nature inspired Cooperation Strategies for Optimisation (NISCO 2010) Studies in Computational intelligence.* Berlin Heidelberg: Springer-Verlag, pp.305-318.

Marinakis, Yannis; Marinaki, Magdalene and Zopounidis, Constantin, 2010. HBMO Algorithm for Financial Classification Problems. *Journal of Applied Soft Computing*, 10 (3): 806-812.

Marzi, A. and Marzi, H., 2015, May. Effects of Data Complexity on The intelligent Diagnostic Reasoning. In *Humanitarian Technology Conference (IHTC2015), 2015 IEEE Canada International* (pp. 1-4). IEEE.

Marzi, A.; Marzi, H. and Marzi, E., 2010. Bio-inspired Solution to Economic Dispatch Problem Using Distributed Computing. In *2010 IEEE Electric Power and Energy Conference (EPEC), August 2010*, pp.1-6.

Masajedi, P., Shirazi, K.H. and Ghanbarzadeh, A., 2013. Verification of Bee Algorithm Based Path Planning for A 6DOF Manipulator Using ADAMS. *Journal of Vibroengineering*, *15*(2), pp.805-815.

Massah, A., Zamani, A., Salehinia, Y., Sh, M.A. and Teshnehlab, M., 2013. A Hybrid Controller Based on CPG and ZMP for Biped Locomotion. *Journal of Mechanical Science and Technology*, *27*(11), pp.3473-3486.

Mastrocinque, E., Yuce, B., Lambiase, A. and Packianather, M.S., 2013. A Multi-Objective Optimisation for Supply Chain Network Using The Bees Algorithm. I*nt. J. Eng. Bus. Manage*, *5*, pp.1-11.

Mayteekrieangkrai, N. and Wongthatsanekorn, W., 2015. Optimized Ready Mixed Concrete Truck Scheduling for Uncertain Factors Using Bee Algorithm. *Songklanakarin Journal of Science & Technology*, *37*(2).

Mccaffrey, J. D., 2009. Generation of Pairwise Test Sets Using A Simulated Bee Colony Algorithm. In *IEEE International Conference on information Reuse & integration (IRI'09), August 2009*. pp. 115-11.

Mccaffrey, J. D., 2011. Graph Partitioning Using a Simulated Bee Colony Algorithm. In *2011 IEEE International Conference on Information Reuse and integration (IRI), August 2011*. pp. 400-405.

Mesa, E.; Velásquez, J. D. and Jaramillo, P., 2011. Generation of Complex Non-Linear Benchmark Functions for Optimisation Using Fuzzy Sets and Classical Test Functions. *Revista Ingenierias Universidal de Medellin*, 10(19): 171-177.

Mechter, A., Kemih, K. and Ghanes, M., 2015. Sliding Mode Control of a Wind Turbine with Exponential Reaching Law. *Acta Polytechnica Hungarica*, *12*(3), pp.167-183.

Mehdinejadiani, B., Naseri, A.A., Jafari, H., Ghanbarzadeh, A. and Baleanu, D., 2013. A Mathematical Model for Simulation of a Water Table Profile between Two Parallel Subsurface Drains Using Fractional Derivatives. *Computers & Mathematics with Applications*, *66*(5), pp.785-794.

Mehrabian, A. R. and Lucas, C., 2006. A Novel Numerical Optimization Algorithm Inspired from Weed Colonization. *Journal of Ecological Informatics*, 1(4): 355-366.

Metni, N. and Lahoud, J., 2013, September. Neuro-Control Robustness Analysis of An inverted Pendulum Using the Bee Algorithm. In *Applied Mechanics and Materials* (Vol. 339, pp. 3-9).

Miller, B. L. and Shaw, M. J., 1996. Genetic Algorithms with Dynamic Niche Sharing for Multimodal Function Optimization. In *Proceedings of IEEE International Conference on Evolutionary Computation, May 1996,* pp.786-791.

Mirjalili, S., Hashim, S.Z.M. and Sardroudi, H.M., 2012. Training Feedforward Neural Networks Using Hybrid Particle Swarm Optimization and Gravitational Search Algorithm. *Applied Mathematics and Computation*, *218*(22), pp.11125-11137.

MizanAdl, S. M. M. and Ardakani, M. D., 2012. Using the Bees Algorithm with the Boundary Elements Method to Solve the Inverse Problem of Transient Heat Conduction. *Majlesi Journal of Mechanical Engineering* [Online], Available From http://journals.iaumajlesi.ac.ir/me/index.php/me [Accessed 1 October 2012].

Mohamed Elsayed, S. A.; Ammar, R. A. and Rajasekaran, S., 2012. Artificial Immune Systems: Models, Applications, and Challenges. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. *March 2012*, pp.256-258.

Mohammadi, N. and Nasirshoaibi, M., 2015. Optimization of Vibrational Absorber Rested on Linear Structures Under Arbitrary Vibrations. *International Journal of Engineering & Technology*, *4*(3), pp.446-450.

Mollabakhshi, N. and Eshghi, M., 2013, January. Combinational Circuit Design Using Bees Algorithm. In *Conference Anthology, IEEE* (pp. 1-4). IEEE.

Möller, T., Bernst, I., Panoglou, D., Muders, D., Ossenkopf, V., Röllig, M. and Schilke, P., 2013. Modeling and Analysis Generic interface for External Numerical Codes (MAGIX). *Astronomy & Astrophysics*, *549*, P.A21.

Mongkolkosol, P. and Luangpaiboon, P., 2011. Steepest Descent Algorithm To Determine The Proper Levels of Bees Parameters on Dynamic Multi-Zone Dispatching Problems. In *Lecture Notes In Engineering and Computer Science: Proceedings of The International Multiconference of Engineers and Computer Scientists 2011, IMECS 2011, 16-18 March, 2011, Hong Kong, Pp1417* (Vol. 1422).

Moradi, A., Ghanbarzadeh, A. and Soodmand, E., 2011a, December. Optimization of Linear and Nonlinear Full Vehicle Model for Improving Ride Comfort vs. Road Holding with the Bees Algorithm. In *Humanities, Science and Engineering (CHUSER), 2011 IEEE Colloquium on* (pp. 17-22). IEEE.

Moradi, A., Ghanbarzadeh, A., Rezazadeh, A. and Soodmand, E., 2011b, December. Solving Engineering Optimization Problems Using the Bees Algorithm. In *Humanities, Science and Engineering (CHUSER), 2011 IEEE Colloquium on* (pp. 162-166). IEEE.

Moradi, A., Nafchi, A.M. and Ghanbarzadeh, A., 2015. Multi-Objective Optimization of Truss Structures Using The Bee Algorithm. *Scientia Iranica. Transaction B, Mechanical Engineering*, *22*(5), pp.1789-1800.

Moradi, A., Shirazi, K.H., Keshavarz, M., Falehi, A.D. and Moradi, M., 2014. Smart Piezoelectric Patch in Non-Linear Beam: Design, Vibration Control and Optimal Location. *Transactions of The institute of Measurement and Control*, *36*(1), pp.131-144.

Moradi, S. and Alimouri, P., 2012. Crack Detection of Plate Structures Using Differential Quadrature Method. *Proceedings of The institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, P.0954406212461881.

Moradi, S. and Kargozarfard, M.H., 2013. On Multiple Crack Detection in Beam Structures. *Journal of Mechanical Science and Technology*, *27*(1), pp.47-55.

Moradi, S. and Tavaf, V., 2013. Crack Detection in Circular Cylindrical Shells Using Differential Quadrature Method. *International Journal of Pressure Vessels and Piping*, *111*, pp.209-216.

Moradi, S.; Fatahi, L. and Razi, P., 2010. Finite Element Model Updating Using Bees Algorithm. *Journal of Structural and Multidisciplinary Optimization*, 42(2): 283-291

Moussa, A. and El-Sheimy, N., 2010. Localization of Wireless Sensor Network Using Bees Optimization Algorithm. In *2010 IEEE International Symposium on Signal Processing and information Technology (ISSPIT), December 2010*, pp.478-481.

Mozaffari, A., Fathi, A. and Behzadipour, S., 2012. The Great Salmon Run: A Novel Bio-Inspired Algorithm for Artificial System Design and Optimisation. *International Journal of Bio-Inspired Computation*, *4*(5), pp.286-301.

Muhamad, Z., Mahmuddin, M., Nasrudin, M.F. and Sahran, S., Local Search Manoeuvres Recruitment In The Bees Algorithm. *Proceedings of The 3rd International Conference on Computing and informatics, ICOCI 2011, 8-9 June, 2011 Bandung, indonesia, pp.*43-48.

Naderian, M., 2014. Environmental Pollution Assessment Using Different intelligent Techniques: A Case Study for Iran. *Bull. Env. Pharmacol. Life Sci*, *3*, pp.126-132.

Nafchi, A.M., Moradi, A., Ghanbarzadeh, A., Yaghoubi, S. and Moradi, M., 2012. An Improved Bees Algorithm for Solving Optimization Mechanical Problems. *20th Annual International Conference on Mechanical Engineering-ISME2012 16-18 May, 2012, School of Mechanical Eng., Shiraz University, Shiraz, Iran.*

Nebti, S. and Boukerram, A., 2012. Handwritten Characters Recognition Based on Nature-inspired Computing and Neuro-Evolution. *Journal of Applied intelligence* [Online], Available from http://www.springer.com [Accessed 1 July 2012].

Nebti, S. and Boukerram, A., 2010. Handwritten Digits Recognition Based on Swarm Optimization Methods. In *Networked Digital Technologies* (pp. 45-54). Springer Berlin Heidelberg.

Nebti, S., 2013. Bio-inspired Algorithms for Colour Image Segmentation. *International Journal of Computer Applications*, *73*(18), pp.11-16.

Nguyen, K.; Nguyen, P. and Tran, N., 2012. A Hybrid Algorithm of Harmony Search and Bees Algorithm for a University Course Timetabling Problem. *International Journal of Computer Science Issue* [Online], Available From http://ijcsi.org [Accessed 1 October 2012].

Nickabadi, A.; Ebadzadeh, M. M. and Safabakhsh, R., 2008. DNPSO: A Dynamic Niching Particle Swarm Optimizer for Multi-Modal Optimization. In *IEEE Congress on Evolutionary Computation (CEC 2008), June 2008,* pp.26-32.

Niknam, Taher, 2011. A New HBMO Algorithm for Multiobjective Daily Volt/Var Control In Distribution Systems Considering Distributed Generators. *Journal of Applied Energy*, 88: 778-788.

Otri, S., 2011. *Improving the Bees Algorithm for Complex Optimisation Problems.* PhD. Thesis, Cardiff University.

Özbakır, L. and Tapkan, P., 2010. Balancing Fuzzy Multi-Objective Two-Sided Assembly Lines via Bees Algorithm. *Journal of intelligent and Fuzzy Systems*, 21(5): 317-329.

Özbakır, L. and Tapkan, P., 2011. Bee Colony intelligence In Zone Constrained Two-Sided Assembly Line Balancing Problem. *Journal of Expert Systems with Applications*, 38(9); 11947-11957.

Özbakir, L.; Baykasoğlu, A. and Tapkan, P., 2010. Bees Algorithm for Generalized Assignment Problem. *Journal of Applied Mathematics and Computation*, 215(11): 782-3795.

Packianather, M. S.; Landy, M. and Pham, D. T., 2009. Enhancing the Speed of the Bees Algorithm Using Pheromone-Based Recruitment. In *7th IEEE International Conference on industrial informatics (INDIN 2009), June 2009*, pp.789-794.

Packianather, M.S. and Kapoor, B., 2015, May. A Wrapper-Based Feature Selection Approach Using Bees Algorithm for a Wood Defect Classification System. In *System of Systems Engineering Conference (SOSE), 2015 10th* (pp. 498-503). IEEE.

Packianather, M.S., Yuce, B., Mastrocinque, E., Fruggiero, F., Pham, D.T. and Lambiase, A., 2014, August. Novel Genetic Bees Algorithm Applied To Single Machine Scheduling Problem. In *World Automation Congress (WAC), 2014* (pp. 906-911). IEEE.

Pai, Ping-Feng; Yang, Shun-Ling and Chang, Ping-Teng, 2009. Forecasting Output of integrated Circuit industry by Support Vector Regression Models with Marriage Honey-Bees Optimisation Algorithms. *Journal of Expert Systems with Applications*, 36: 10746-10751.

Palominos, Pedro; Toledo, Francisco; Véjar, Andŕes and Alfaro, Miguel, 2012. Marriage in Honey Bees Optimisation Algorithm for Flow-Shop Problems. *Journal of informatica Economică*, 16 (2): 27-34.

Parpinelli, R.S. and Lopes, H.S., 2011. New inspirations In Swarm intelligence: A Survey. *International Journal of Bio-inspired Computation*, 3(1): 1-16.

Parsa, H.R., Asghargholamian, S. and Abbasi, M., 2013. Design and Optimization of Eddy Current Testing Probe Using Bees Algorithm and Finite Element Analysis. *International Journal of Modern Education and Computer Science (IJMECS)*, *5*(12), pp.40-46.

Passino, K. M., 2002. Biomimicry of Bacterial foraging for Distributed Optimization and Control. In *IEEE Control Systems Magazine, 22(3),* pp.52-67.

Passino, K. M. and Seeley, Thomas D., 2006. Modeling and Analysis of Nest-Site Selection by Honeybee Swarms: The Speed and Accuracy Trade-off. *Journal of Behavioural Ecology and Sociobiology*, 59: 427–442.

Paul, S., Müller, H., Preiser, R., De Lima Neto, F.B., Marwala, T. and De Wilde, P., 2014. Developing A Management Decision-Making Model Based Upon A Complexity Perspective with Reference To The Bee Algorithm. *Emergence: Complexity and Organization*, *16*(4), P.D1.

Paulovič, Aurel, 2011. Bee Nest-Site Selection Clustering. *information Sciences and Technologies Bulletin of The ACM Slovakia, Special Section on Student Research In informatics and information Technologies*, 3 (2): 100-103.

Pham D. T.; Ghanbarzadeh A.; Koç E.; Otri S.; Rahim S. and Zaidi M., 2005. *The Bees Algorithm*: Technical Note, Manufacturing Engineering Centre, Cardiff University, UK.

Pham, D. T. and Castellani, M., 2009. The Bees Algorithm: Modelling foraging Behaviour to Solve Continuous Optimization Problems. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 223(12): 2919-2938.

Pham, D. T. and Darwish, H. A., 2008. Fuzzy Selection of Local Search Sites in the Bees Algorithm. In *4th International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2007)* [Online], Available from http://conference.iproms.org [Accessed 1 March 2012].

Pham, D. T. and Darwish, H. A., 2010. Using the Bees Algorithm with Kalman Filtering To Train an Artificial Neural Network for Pattern Classification. *Proceedings of The institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 224(7): 885-892.

Pham, D. T. and Kalyoncu, M., 2009. Optimisation of a Fuzzy Logic Controller for a Flexible Single-Link Robot Arm Using the Bees Algorithm." In *7th IEEE International Conference on industrial informatics (INDIN 2009), June 2009*, pp.475-480.

Pham, D. T. and Koç, E., 2010. Design of a Two-Dimensional Recursive Filter Using the Bees Algorithm. *International Journal of Automation and Computing*, 7(3): 399-402.

Pham, D. T.; Afify, A. A. and Koç, E. (2007a). "Manufacturing Cell Formation Using the Bees Algorithm." In *Proceedings of Innovative Production Machines and Systems Virtual Conference* [Online], Available From From http://conference.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Al-Jabbouli, H.; Mahmuddin, M.; Otri, S. and Darwish, A., 2008a. Application of the Bees Algorithm to Fuzzy Clustering. In *Proceedings of 4th International Virtual Conference on intelligent Production Machines* *and Systems (IPROMS 2008)* [Online], Scotland, Available From http://conference.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Ang, M.; Ng, K.; Otri, S. and Darwish, H. A., 2008b. Generating Branded Product Concepts: Comparing The Bees Algorithm and An Evolutionary Algorithm. In *Proceedings of innovative Production Machines and Systems Virtual Conference* [Online], Available From From http://conference.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Castellani, M. and Fahmy, A. A., 2008c. Learning the inverse Kinematics of a Robot Manipulator Using the Bees Algorithm. In *6th IEEE International Conference on industrial informatics (INDIN 2008), July 2008,* pp.493-498.

Pham, D. T.; Darwish, H. A.; Eldukhri, E. and Otri, S., 2007b. Using the Bees Algorithm to Tune a Fuzzy Logic Controller for a Robot Gymnast. In *Proceedings of innovative Production Machines and Systems Virtual Conference (IPROMS) [Online], May 2007*. Available From http://conference.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Ghanbarzadeh, A.; Koc, E.; Otri, S.; Rahim, S. and Zaidi, M., 2006a. The Bees Algorithm – A Novel Tool for Complex Optimisation Problems. In *Proceedings of innovative Production Machines and System Conference (IPROMS) [Online], July 2006,* pp.454-461. Available From http://www.iproms.org [Accessed 17 January 2012].

Pham, D. T.; Ghanbarzadeh, A.; Otri, S. and Koç, E., 2009a. Optimal Design of Mechanical Components Using the Bees Algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 223(5): 1051-1056.

Pham, D. T.; Koç, E.; Ghanbarzadeh, A. and Otri, S., 2006b. Optimisation of the Weights of Multi-Layered Perceptrons Using the Bees Algorithm. In *Proceedings of 5th International Symposium on intelligent Manufacturing Systems, May 2006,* pp.38-46.

Pham, D. T.; Koç, E.; Lee, J. Y. and Phrueksanant, J., 2007c. Using the Bees Algorithm to Schedule Jobs for a Machine. In *Proceedings of Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, June 2007,* pp.430-439.

Pham, D. T.; Lee, J. Y.; Haj Darwish, A. and Soroka, A. J., 2008d. Multi-Objective Environmental/Economic Power Dispatch Using the Bees Algorithm with Pareto Optimality. In *4th International Virtual Conference on intelligent Production Machines and Systems (IPROMS 2008)* [Online], Available from http://www.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Marzi, H.; Marzi, A.; Marzi, E.; Darwish, A. H. and Lee, J. Y., 2010. Using Grid Computing To Accelerate Optimization Solution: A System of Systems Approach. In *IEEE 2010 5th International Conference on System of Systems Engineering (SOSE), June 2010*, pp.1-6.

Pham, D. T.; Muhamad, Z.; Mahmuddin, M.; Ghanbarzadeh, A.; Koç, E. and Otri, S., 2007d. Using the Bees Algorithm to Optimise a Support Vector Machine for Wood Defect Classification. In *International Virtual Conference on intelligent Production Machines and Systems (IPROMS 2007)* [Online], Available from http://www.iproms.org [Accessed 1 March 2012].

Pham, D. T.; Otri, S. and Darwish, A. H., 2007e. Application of the Bees Algorithm to PCB Assembly Optimisation. In *IPROMS, 3rd International Virtual Conference on intelligent Production Machines and Systems [Online]*, pp. 511-516. Available From http://www.iproms.org [Accessed 17 January 2012].

Pham, D. T.; Otri, S.; Ghanbarzadeh, A. and Koc, E., 2006c. Application of the Bees Algorithm to the Training of Learning Vector Quantisation Networks for Control Chart Pattern Recognition. In *IEEE 2nd International Conference Information and Communication Technologies: From Theory to Applications (ICTTA'06), April 2006. Vol.1*, pp.1624-1629.

Pham, D. T.; Pham, Q. T.; Ghanbarzadeh, A. and Castellani, M., 2008e. Dynamic Optimisation of Chemical Engineering Processes Using the Bees Algorithm. In *17th IFAC World Congress, South Korea, July 2008*, pp.6100-6105.

Pham, D. T.; Sholedolu, M. and Packianather, M., 2009b. The Bees Algorithm with Attraction to Global Best Solutions. In *Proceedings of Innovative Production Machines and Systems Virtual Conference (IPROMS 2009)* [Online], Cardiff, UK, Available From http://conference.iproms.org [Accessed 1 November 2012].

Pham, D. T.; Soroka, A. J.; Ghanbarzadeh, A.; Koc, E.; Otri, S. and Packianather, M., 2006d. Optimising Neural Networks for Identification of Wood Defects Using the Bees Algorithm. In *2006 IEEE International Conference on Industrial Informatics, August 2006*, pp.1346-1351.

Pham, D. T.; Suarez-Alvarez, M. M. and Prostov, Y. I., 2011. Random Search with K-Prototypes Algorithm for Clustering Mixed Datasets. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 467(2132): 2387-2403.

Pham, D.T. and Castellani, M., 2014. Benchmarking and Comparison of Nature-inspired Population-Based Continuous Optimisation Algorithms. *Soft Computing*, *18*(5), pp.871-903.

Pham, D.T. and Castellani, M., 2015. A Comparative Study of the Bees Algorithm as a Tool for Function Optimisation. *Cogent Engineering*, *2*(1), P.1091540.

Pham, Q. T.; Pham, D. T. and Castellani, M., 2012. A Modified Bees Algorithm and a Statistics-Based Method for Tuning Its Parameters. *Proceedings of The institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 226(3): 287-301.

Phonrattanasak, P., 2011. Optimal Placement of Wind Farm on the Power System Using Multiobjective Bees Algorithm. In *Proceedings of The World Congress on Engineering*, 2, pp.1414-1418.

Phonrattanasak, P., Miyatake, M. and Sakamoto, O., 2013, May. Optimal Location and Sizing of Solar Farm on Japan East Power System Using Multiobjective Bees Algorithm. In *Energytech, 2013 IEEE* (pp.1-6). IEEE.

Phuc, N. B.; Khang, N. T. T. M. and Nuong, T. T. H., 2011. A New Hybrid GA-Bees Algorithm for a Real-World University Timetabling Problem. In *IEEE 2011 International Conference on intelligent Computation and Bio-Medical instrumentation (ICBMI), December 2011*, pp.321-326.

Poolsamran, Patcharawadee and Thammano, Arit, 2011. A Modified Marriage in Honey-Bees Optimisation for Function Algorithm. *Journal of Procedia Computer Science*, 6: 335-342.

Poonam, E. and Dhaiya, R., 2015. Artificial Intelligence Based Cluster Optimization for Text Data Mining. *International Journal of Computer Science and Mobile Computing, 4(9),* pp.8-15.

Poor, M.M. and Saz, M.S., 2012. Multi-Objective Optimization of Laminates with Straight Free Edges and Curved Free Edges by Using Bees Algorithm. *American Journal of Advanced Scientific Research (AJASR)*, *1*(4), pp.130-136.

Pourkamalianaraki, M. and Sadeghi, M., 2015. Honey Bee-inspired Algorithms for SNP Haplotype Reconstruction Problem. *Journal of Experimental & Theoretical Artificial intelligence*, pp.1-14.

Qing, L., Gang, W. and Qiuping, W., 2005, September. Restricted Evolution Based Multimodal Function Optimization in Holographic Grating Design. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on* (Vol. 1, Pp. 789-794). IEEE.

Qing, L.; Gang, W.; Zaiyue, Y. and Qiuping, W., 2008. Crowding Clustering Genetic Algorithm for Multimodal Function Optimization. *Journal of Applied Soft Computing*, 8(1): 88-95.

Rahkar-Farshi, T., Kesemen, O. and Behjat-Jamal, S., 2014, April. Multi Hyperbole Detection on Images Using Modified Artificial Bee Colony (ABC) for Multimodal Function Optimization. In *Signal Processing and Communications Applications Conference (SIU), 2014 22nd* (pp. 894-898). IEEE.

Rahnamayan S.; Tizhoosh, H. R. And Salama, M. M. A., 2007. A Novel Population Initialisation Method for Accelerating Evolutionary Algorithms. *International Journal of Computers and Mathematics*, 53: 1605-1614.

Rambod, M. and Rezaeian, J., 2014. Robust Meta-Heuristics Implementation for Unrelated Parallel Machines Scheduling Problem with Rework Processes and Machine Eligibility Restrictions. *Computers & Industrial Engineering*, *77*, pp.15-28.

Rao, R.V., Savsani, V.J. and Vakharia, D.P., 2011. Teaching–Learning-Based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems. *Computer-Aided Design*, *43*(3), pp.303-315.

Rashedi, E., Nezamabadi-Pour, H. and Saryazdi, S., 2009. GSA: A Gravitational Search Algorithm. *Information Sciences*, *179*(13), pp.2232-2248.

Rashid, M.; Baig, A. R. and Zafar, K., 2009. Niching with Sub-Swarm Based Particle Swarm Optimization. In *International Conference on Computer Technology and Development (ICCTD'09), November 2009. Vol.2*. pp. 181-183.

Rashtchi, V., Gholinezhad, J. and Farhang, P., 2010, October. Optimal Coordination of Overcurrent Relays Using Honey Bee Algorithm. In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress on* (pp. 401-405). IEEE.

Raudenská, Lenka, 2009. Swarm-Based Optimisation. Journal of Quality Innovation Prosperity [Online], 13: 45-52. Available from: http://qip-journal.eu [Accessed 19 August 2012].

Rönkkönen, J.; Li, X., Kyrki, V. and Lampinen, J., 2008. A Generator for Multimodal Test Functions with Multiple Global Optima. In *Simulated Evolution and Learning*, *Lecture Notes In Computer Science*, Vol.5361, Berlin Heidelberg: Springer-Verlag, pp.239-248.

Rufai, K.I., Muniyandi, R.C. and Othman, Z.A., 2014. Improving Bee Algorithm Based Feature Selection In intrusion Detection System Using Membrane Computing. *Journal of Networks*, *9*(3), pp.523-529.

Ruiz-Vanoye, Jorge A. and Díaz-Parra, Ocotlán, 2011. Similarities between Meta-Heuristics Algorithms and the Science of Life. *Central European Journal of Operations Research*, 19:445–466.

Ruz, G. A. and Goles, E., 2011. Learning Gene Regulatory Networks Using the Bees Algorithm. *Journal of Neural Computing and Applications* [Online], Available From http://www.springerlink.com [Accessed 1 March 2012].

Ruz, G. A. and Goles, E., 2012. Reconstruction and Update Robustness of the Mammalian Cell Cycle Network. In *2012 IEEE Symposium on Computational intelligence In Bioinformatics and Computational Biology (CIBCB), May 2012*, pp.397-403.

Ruz, G.A., Goles, E., Montalva, M. and Fogel, G.B., 2014. Dynamical and Topological Robustness of the Mammalian Cell Cycle Network: A Reverse Engineering Approach. *Biosystems*, *115*, pp.23-32.

Saad, E. M.; Awadalla, M. H. and Darwish, R. R., 2008. A Data Gathering Algorithm for a Mobile Sink in Large-Scale Sensor Networks. In *IEEE The Fourth International Conference on Wireless and Mobile Communications (ICWMC'08), July 2008*, pp.207-213.

Sadeghi, A. and Alahyari, A., 2013. Bi-Objective MRCPCP Problem with Bees Metaheuristic Algorithm. *International Research Journal of Applied and Basic Sciences, 4(8)*, pp.165-2170.

Sadeghi, A., Kalanaki, A., Noktehdan, A., Samghabadi, A.S. and Barzinpour, F., 2011. Using Bees Algorithm To Solve The Resource Constrained Project Scheduling Problem In PSPLIB. In *Theoretical and Mathematical Foundations of Computer Science* (pp. 486-494). Springer Berlin Heidelberg.

Sadiq, A.T., Duaimi, M.G. and Shaker, S.A., 2012, November. Data Missing Solution Using Rough Set Theory and Swarm intelligence. In *Advanced Computer Science Applications and Technologies (ACSAT), 2012 International Conference on* (pp. 173-180). IEEE.

Sadiq, Ahmed and Hamad, Amaal G., 2010. BSA: A Hybrid Bees' Simulated Annealing Algorithm To Solve Optimization & NP-Complete Problems. *Engineering and Technology Journal* [Online], 28(2): 271-281. Available from http://www.iasj.net [Accessed 1 April 2012].

Sadiq, Ahmed and Hamad, Amaal G., 2012. Exploration-Balanced Bees Algorithms to Solve Optimization & NP-Complete Problems. *International Journal of Research and Reviews in Soft and intelligent Computing (IJRRSIC)* [Online], 2(1): 108-113. Available From http://sciacademypublisher.com [Accessed 1 April 2012].

Sagayam, R. and Akilandeswari, K., 2012. Comparison of Ant Colony and Bee Colony Optimization for Spam Host Detection. *International Journal of Engineering Research and Development*, *4*(8), pp.26-32.

Sagheer, A.M., Sadiq, A.T. and Ibrahim, M.S., 2012, December. Improvement of Scatter Search Using Bees Algorithm. In *Signal Processing and Communication Systems (ICSPCS), 2012 6th International Conference on* (pp. 1-7). IEEE.

Saini, G. and Kaur, K., 2014. Comparative Study of Regionalization Based on Hybrid K-Mean and Ward's Clustering Algorithm Using Different Optimization Techniques. *International Journal of Computer Engineering and Applications, 8(2)*, pp.33-44.

Salamat, A.R. and Ghanbarzadeh, A., 2012. Multi Objective Optimization of Antisymmetric Angle-Ply Laminate under Transverse Loads with Bees Algorithm. *American Journal of Advanced Scientific Research (AJASR)*, *1*(3), pp.93-98.

Salamat, A.R. and Raiesinezhad, M., 2012. Optimum Design of Antisymmetric Cross-Ply and Angle-Ply Laminate with Bees Algorithm. *American Journal of Advanced Scientific Research (AJASR)*, *1*(4), pp.202-207.

Samadzadegan, Farhad and Ferdosi, Elahe, 2012. Classification of Polarimetric SAR Images Based on Optimum Svms Classifier Using Bees Algorithm. In *Proceedings of International Conference on intelligent Computational Systems (ICICS'2012), Dubai, 7-8 January 2012*, pp.106-111.

Sarailoo, M., Rahmani, Z. and Rezaie, B., 2015. A Novel Model Predictive Control Scheme Based on Bees Algorithm in a Class of Nonlinear Systems: Application to a Three Tank System. *Neurocomputing*, *152*, pp.294-304.

Sareni, B., Krähenbühl, L. and Nicolas, A., 1998. Niching Genetic Algorithms for Optimization in Electromagnetics. I. Fundamentals. *IEEE Transactions on Magnetics*, *34*(5), pp.2984-2987.

Satheesh, A., and Manigandan, T., 2013. Maintaining Power System Stability with FACTS Controller Using Bees Algorithm and NN. *Journal of Theoretical and Applied information Technology, 49(1)*, pp.38-47.

Sayadi, F.; Ismail, M.; Misran, N. and Jumari, K., 2009. Multi-Objective Optimization Using The Bees Algorithm In Time-Varying Channel for MIMO MC-CDMA Systems. *European Journal of Scientific Research*, 33(3): 411-428.

Sayarshad, H. R., 2010. Using Bees Algorithm for Material Handling Equipment Planning In Manufacturing Systems. *The International Journal of Advanced Manufacturing Technology*, 48(9): 1009-1018.

Seeley, T. D. (1995). *The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies*. Cambridge: Harvard University Press.

Seleem, Y.Z., Mohamed, M.H. and Hussain, K.F., 2013, September. Improving Genetic Process Mining Using Honey Bee Algorithm. In *informatics and Applications (ICIA), 2013 Second International Conference on* (pp. 59-65). IEEE.

Sen, M.A. and Kalyoncu, M., 2015. Optimisation of A PID Controller for An inverted Pendulum Using the Bees Algorithm. *Applied Mechanics & Materials*, *789-790*, pp.1039-1044.

Şenyiğit, E.; Düğenci, M.; Aydin, M. E. and Zeydan, M., 2012. Heuristic-Based Neural Networks for Stochastic Dynamic Lot Sizing Problem. *Journal of Applied Soft Computing* [Online], Available From http://www.sciencedirect.com [Accessed 1 April 2012].

Shafia, M. A.; Rahimi Moghaddam, M. and Tavakolian, R., 2011. A Hybrid Algorithm for Data Clustering Using Honey Bee Algorithm, Genetic Algorithm and K-Means Method. *Journal of Advanced Computer Science and Technology Research* [Online], 1(2: 110-125. Available From http://www.sign-ific-ance.co.uk [Accessed 18 September 2012].

Shatnawi, N., Faidzul, M. and Sahran, S., 2013. Optimization of Multilevel Image Thresholding Using the Bees Algorithm. *Journal of Applied Sciences*, *13*(3), pp.458-464.

Shatnawi, N., Sahran, S. and Faidzul, M., 2013. A Memory-Based Bees Algorithm: An Enhancement. *Journal of Applied Sciences*, *13*(3), pp.497-502.

Shen, D. and Xia, X., 2012. An Improved Species Conserving Genetic Algorithm for Multimodal Optimization. In *IEEE 2012 Eighth International Conference on Natural Computation (ICNC), May 2012*, pp.1156-1160.

Sherme, A. E., 2011. Hybrid Intelligent Technique for Automatic Communication Signals Recognition Using Bees Algorithm and MLP Neural Networks Based on The Efficient Features. *Journal of Expert Systems with Applications*, 38(5): 6000-6006.

Sherme, A. E., 2012. A Novel Method for Automatic Modulation Recognition. *Applied Soft Computing*, *12*(1), pp.453-461.

Shi, Y., 2011. Brain Storm Optimization Algorithm. In *Advances in Swarm Intelligence* (pp. 303-309). Springer Berlin Heidelberg.

Shi, Y. and Eberhart, R. C., 1999. Empirical Study of Particle Swarm Optimization. In *IEEE Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99). Washington D.C. 6-9 July 2009. Vol.3*. pp.1945-1950.

Shir, O. M.; Emmerich, M. and Bäck, T., 2010. Adaptive Niche Radii and Niche Shapes Approaches for Niching with the CMA-ES. *Journal of Evolutionary Computation*, 18(1): 97-126.

Simon, D., 2008. Biogeography-Based Optimization. *IEEE Transactions on Evolutionary Computation*, 12(6): 702-713.

Singh, G. and Deb, K., 2006. Comparison of Multi-Modal Optimization Algorithms Based on Evolutionary Algorithms. In *Proceedings of The Genetic and Evolutionary Computation Conference (GECCO-2006), New York*, pp.1305-1312.

Singh, P.; Kaur, N. and Kaur, L., 2011. Satellite Image Classification by Hybridization of FPAB Algorithm and Bacterial Chemotaxis. *International Journal of Computer Technology and Electronics Engineering* [Online], 1: 21-27. Available From http://www.ijctee.org [Accessed 1 November 2012].

Sivanandam, S.N. and Deepa, S.N., 2006. *Introduction to Neural Networks Using Matlab 6.0*. Tata McGraw-Hill Education.

Socha, K. and Dorigo, M., 2008. Ant Colony Optimization for Continuous Domains. *European Journal of Operational Research*, *185*(3), pp.1155-1173.

Solomon, Ralf, 1998. Evolutionary Algorithms and Gradient Search: Similarities and Differences. *IEEE Transactions on Evolutionary Computation*, 2(2): 45-55

Song, Mei-Ping and Gu, Guo-Chang, 2004. Research on Particle Swarm Optimization: A Review. In *IEEE Proceedings of The Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004*, pp.2236-2241.

Songmuang, P. and Ueno, M., 2011. Bees Algorithm for Construction of Multiple Test forms In E-Testing. *IEEE Transactions on Learning Technologies*, 4(3): 209-221.

Songmuang, P., Ueno, M. and Nagaoka, K., 2012. Development of Automated E-Testing Construction System with Redundant Item Detection. In *The 8th International Conference on Elearning for Knowledge-Based Society, Keynote Address* (Vol. 104).

Soós, D., 2013. Optimisation Algorithm Inspired By Social Insect Behaviour In Comparison With Hill Climbing. *Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on Student Research in Informatics and Information Technologies*, 5(2): 48-52

Stoean, C.; Preuss, M.; Stoean, R. and Dumitrescu, D., 2010. Multimodal Optimization by Means of a Topological Species Conservation Algorithm. *IEEE Transactions on Evolutionary Computation*, 14(6): 842-864.

Storn, R. and Price, K., 1995. *Differential Evolution-A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces* (Vol. 3). Berkeley: ICSI.

Sumpavakup, C., Chusanapiputt, S. and Srikun, I., 2011, August. A Hybrid Cultural-Based Bee Colony Algorithm for Solving the Optimal Power Flow. In *Circuits and Systems (MWSCAS), 2011 IEEE 54th International Midwest Symposium on* (pp. 1-4). IEEE.

Sumpavakup, C.; Srikun, I. and Chusanapiputt, S., 2012. A Solution To Multi-Objective Optimal Power Flow Using Hybrid Cultural-Based Bees Algorithm. In *IEEE 2012 Asia-Pacific Power and Energy Engineering Conference (APPEEC), March 2012*. pp. 1-4.

Sundareswaran, K. and Sreedevi, V. T., 2008. Development of Novel Optimization Procedure Based on Honey Bee foraging Behavior. In *IEEE International Conference on Systems, Man and Cybernetics (SMC 2008), October 2008*. pp. 1220-1225.

Taher, M.T. and Masoudrahmani, A., 2012. Multicast Routing In Computer Networks Considering Quality of Service (Qos) Based on Honey Bee Algorithm. *International Journal of Computer Applications, 58*(2), pp.20-24.

Talbi, E.G., 2002. A Taxonomy of Hybrid Metaheuristics. *Journal of Heuristics*, *8*(5), pp.541-564.

Tang, W. J.; Wu, Q. H. and Saunders, J. R., 2007. A Bacterial Swarming Algorithm for Global Optimization. In *IEEE Congress on Evolutionary Computation (CEC 2007), September 2007*. pp. 1207-1212.

Tao, F., Laili, Y. and Zhang, L., 2015. Recent Advances of Intelligent Optimization Algorithm in Manufacturing. In *Configurable Intelligent Optimization Algorithm* (pp. 35-80). Springer International Publishing.

Tapkan, P.; Özbakır, L. and Baykasoğlu, A., 2012a. Bees Algorithm for Constrained Fuzzy Multi-Objective Two-Sided Assembly Line Balancing Problem. *Optimization Letters*, 6(6): 1039-1049.

Tapkan, P.; Ozbakir, L. and Baykasoglu, A., 2012b. Modeling and Solving Constrained Two-Sided Assembly Line Balancing Problem via Bee Algorithms. *Journal of Applied Soft Computing*, 12: 3343-3355.

Tapkan, P.; Özbakır, L. and Baykasoğlu, A., 2013. Solving Fuzzy Multiple Objective Generalized Assignment Problems Directly Via Bees Algorithm and Fuzzy Ranking. *Journal of Expert Systems with Applications*, 40: 892-898.

Tariq, S. Abdul-Razaq and Faez, H. Ali, 2015. Hybrid Bees Algorithm to Solve Aircraft Landing Problem. *Journal of Zankoi Sulaimani, 17(1), Part A*, pp.71-89.

Teimoury, E. and Haddad, H., 2013. A Bee Algorithm for Parallel Batch Production Scheduling. *International Journal*, *2*(2), pp.169-171.

Teodorović, D. and Dell'Orco, M., 2005. Bee Colony Optimization–A Cooperative Learning Approach to Complex Transportation Problems. In *Proceedings of 16th Mini–EURO Conference and 10th Meeting of EWGT Advanced OR and AI Methods in Transportation, 13-16 September 2005,* Poznan: Publishing House of the Polish Operational and System Research. pp. 51-60.

Teodorović, D.; Davidovic, T. and Selmic, M., 2011. Bee Colony Optimization: The Applications Survey. *ACM Transactions on Computational Logic* [Online], Available From http://www.sanu.ac.rs [Accessed 1 March 2012].

Thomsen, R., 2004. Multimodal Optimization Using Crowding-Based Differential Evolution. *In IEEE Congress on Evolutionary Computation (CEC2004), June 2004. Vol.2*, pp.1382-1389.

Timmis, J.; Andrews, P.; Owens, N. and Clark, E., 2008a. An Interdisciplinary Perspective on Artificial Immune Systems. *Journal of Evolutionary intelligence*, 1(1): 5-26.

Timmis, J.; Hone, A.; Stibor, T. and Clark, E., 2008b. Theoretical Advances in Artificial Immune Systems. *Journal of Theoretical Computer Science*, 403(1): 11-32.

Tolabi, H.B., Ali, M.H., Ayob, S.B.M. and Rizwan, M., 2014a. Novel Hybrid Fuzzy-Bees Algorithm for Optimal Feeder Multi-Objective Reconfiguration by Considering Multiple-Distributed Generation. *Energy*, *71*, pp.507-515.

Tolabi, H.B., Ayob, S.B.M., Moradi, M.H. and Shakarmi, M., 2014b. New Technique for Estimating the Monthly Average Daily Global Solar Radiation Using Bees Algorithm and Empirical Equations. *Environmental Progress & Sustainable Energy*, *33*(3), pp.1042-1050.

Tolabi, H.B., Moradib, M.H. and Tolabia, F.B., 2013. New Technique for Global Solar Radiation forecast Using Bees Algorithm. *International Journal of Engineering*, *26*(11), pp.1385-1392.

Toloei, A.R., Zarchi, M. and Attaran, B., 2014. Application of Active Suspension System to Reduce Aircraft Vibration Using PID Technique and Bees Algorithm. *International Journal of Computer Applications*, *98*(6), pp.17-24.

Tran, Q. D.; Liatsis, P.; Zhu, B. and He, C., 2011. An Approach for Multimodal Biometric Fusion under the Missing Data Scenario. In *IEEE 2011 International Conference on Uncertainty Reasoning and Knowledge Engineering (URKE), August 2011. Vol.1*, pp.185-188.

Triwate, P. and Luangpaiboon, P., 2010. Bees Algorithm for Dynamic Multi-Zone Dispatching In Truck Load Trucking. In *2010 IEEE International Conference on industrial Engineering and Engineering Management (IEEM), December 2010*, pp.1165-1169.

Tsai, H.C., 2014a. Novel Bees Algorithm: Stochastic Self-Adaptive Neighbourhood. *Applied Mathematics and Computation*, *247*, pp.1161-1172.

Tsai, H.C., 2014b. Integrating The Artificial Bee Colony and Bees Algorithm To Face Constrained Optimization Problems. *Information Sciences*, *258*, pp.80-93.

Tudu, B., Majumder, S., Mandal, K. K., & Chakraborty, N., 2011. Optimal Unit Sizing of Stand-Alone Renewable Hybrid Energy System Using Bees Algorithm. In *IEEE 2011 International Conference on Energy, Automation, and Signal (ICEAS), December 2011*, pp.1-6.

Tudu, B., Mandal, K.K. and Chakraborty, N., 2014, January. Optimal Design and Performance Evaluation of A Grid independent Hybrid Micro Hydro-Solar-Wind-Fuel Cell Energy System Using Meta-Heuristic Techniques. In *Non Conventional Energy (ICONCE), 2014 1st International Conference on* (pp. 89-93). IEEE.

Ursem, R.K., 1999. Multinational Evolutionary Algorithms. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99)*. Pp1633-1640.

Uymaz, S.A., Tezel, G. and Yel, E., 2015. Artificial Algae Algorithm (AAA) for Nonlinear Global Optimization. *Applied Soft Computing*, *31*, pp.153-171.

Vakil-Baghmisheh, M. T. and Salim, M., 2010. A Modified Fast Marriage in Honey Bee Optimisation Algorithm. In *Proceedings of IEEE 5th International Symposium on Telecommunications,* pp.950-955.

Vennila, H. and Prakash, T. R. D., 2012. A Solution for Environmental Constrained Economic Dispatch Problems Using Honey Bee Algorithm. *International Journal of Computer Applications*, 47(22): 13-17.

Wang, C. R.; Zhou, C. L. and Ma, J. W., 2005. An Improved Artificial Fish-Swarm Algorithm and Its Application in Feed-forward Neural Networks. In *IEEE Proceedings of 2005 International Conference on Machine Learning and Cybernetics, August 2005. Vol.5*. pp.2890-2894.

Wang, X., Ceberio, M., Virani, S., Garcia, A. and Cummins, J., 2013. A Hybrid Algorithm to Extract Fuzzy Measures for Software Quality Assessment. *Journal of Uncertain Systems*, *7*(3), pp.219-237.

Wang, X., Contreras, A.F.G., Ceberio, M., Del Hoyo, C., Gutierrez, L.C. and Virane, S., 2012, August. Interval-Based Algorithms to Extract Fuzzy Measures for Software Quality Assessment. In *Fuzzy information Processing Society (NAFIPS), 2012 Annual Meeting of The North American* (pp. 1-6). IEEE.

Wang, X.; Cummins, J. and Ceberio, M., 2011. The Bees Algorithm to Extract Fuzzy Measures for Sample Data. In *IEEE 2011 Annual Meeting of The North American Fuzzy information Processing Society (NAFIPS), March 2011*, pp.1-6.

Wen, J. Y.; Wu, Q. H.; Turner, D. R.; Cheng, S. J. and Fitch, J., 2004. Optimal Coordinated Voltage Control for Power System Voltage Stability. *IEEE Transaction on Power Systems*, 19(2): 1115-1122.

Wen, J. Y.; Wu, Q. H.; Jiang, L. and Cheng, S. J., 2003. Pseudo-Gradient Based Evolutionary Programming. *Electronics Letters*, 39(7): 631-632.

Woldemariam, K. M. and Yen, G. G., 2010. Vaccine-Enhanced Artificial Immune System for Multimodal Function Optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 40(1): 218-228.

Wolpert, D. H. and Macready, W. G., 1997. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1): 67-82.

Wongthatsanekorn, W. and Matheekrieangkrai, N., 2014. A Case Study of Bee Algorithm for Ready Mixed Concrete Problem. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering, 8(7),* pp.1253-1258.

Xie, Y., Zhou, Z., Pham, D.T., Xu, W. and Ji, C., 2015. A Multi-User Manufacturing Resource Service Composition Method Based on the Bees Algorithm. *Hindawi Publishing Corporation Computational intelligence and Neuroscience Volume 2015, Article ID 780352,* 13 Pages http://dx.doi.org/10.1155/2015/780352

Xu, S., Ji, Z., Pham, D.T. and Yu, F., 2010. Bio-inspired Binary Bees Algorithm for A Two-Level Distribution Optimisation Problem. *Journal of Bionic Engineering*, *7*(2), pp.161-167.

Xu, S., Ji, Z., Pham, D.T. and Yu, F., 2011. Binary Bees Algorithm–Bioinspiration From The foraging Mechanism of Honeybees to Optimize a Multiobjective Multidimensional Assignment Problem. *Engineering Optimization*, *43*(11), pp.1141-1159.

Xu, S.; Yu, F.; Luo, Z.; Ji, Z.; Pham, D. T. and Qiu, R., 2011. Adaptive Bees Algorithm—Bioinspiration from Honeybee foraging To Optimize Fuel Economy of a Semi-Track Air-Cushion Vehicle. *The Computer Journal*, 54(9): 1416-1426.

Xu, W., Tian, S., Liu, Q., Xie, Y., Zhou, Z. and Pham, D.T., 2015. An Improved Discrete Bees Algorithm for Correlation-Aware Service Aggregation Optimization in Cloud Manufacturing. *The International Journal of Advanced Manufacturing Technology*, pp.1-12.

Xu, W.; Zhou, Z.; Pham, D. T.; Liu, Q.; Ji, C. and Meng, W., 2012. Quality of Service in Manufacturing Networks: A Service Framework and Its Implementation. *The International Journal of Advanced Manufacturing Technology*, 63: 1227-1237.

Yang, Chenguang; Chen, Jie and Tu, Xuyan, 2007a. Algorithm of Fast Marriage in Honey Bees Optimisation and Convergence Analysis. In *Proceedings of the IEEE International Conference on Automation and Logistics. Jinan, 2007,* pp.1794-1799.

Yang, Chenguang; Chen, Jie and Tu, Xuyan, 2007b. Algorithm of Marriage in Honey Bees Optimisation Based on the Nelder-Mead Method. In *International Conference on intelligent Systems and Knowledge Engineering (ISKE 2007), Advances In intelligent Systems Research.*

Yang, Chenguang; Chen, Jie and Tu, Xuyan, 2007c. Algorithm of Marriage in Honey Bees Optimisation Based on the Wolf Pack Search. In *Proceedings of IEEE International Conference on Intelligent Pervasive Computing*, pp.462-467.

Yang, F., Li, Z. and Fan, Y., 2015a, April. A Specific Combination Scheme for Communication Modulation Recognition Based on the Bees Algorithm. In *2015 International Conference on Mechatronics, Electronic, Industrial and Control Engineering (MEIC-15)*. Atlantis Press, pp.185-188.

Yang, F., Tan, H. and Fan, Y., 2015b. A Specific Combination Scheme for Communication Modulation Recognition Based on the Bees Algorithm and Neural Network. *Journal of Communications*, *10*(10), pp.797-803.

Yang, X. S. (2010a) Firefly Algorithm, Stochastic Test Functions and Design Optimisation. *International Journal of Bio-inspired Computation*, 2(2): 78-84.

Yang, X. S. and Deb, S., 2009, December. Cuckoo Search via Lévy Flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on* (pp. 210-214). IEEE.

Yang, X. S., 2005. Engineering Optimizations via Nature-inspired Virtual Bee Algorithms. In *Artificial intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, *Lecture Notes in Computer Science*, Vol.3562. Berlin Heidelberg: Springer-Verlag, pp.317-323.

Yang, X. S., 2010b. Firefly Algorithm, Levy Flights and Global Optimization. In *Research and Development in Intelligent Systems XXVI*, pp.209-218. London: Springer Computer Science.

Yang, X. S., 2011. Review of Metaheuristics and Generalized Evolutionary Walk Algorithm. *International Journal of Bio-inspired Computation*, 3(2): 77-84.

Yao, J., Kharma, N. and Grogono, P., 2005. A Multi-Population Genetic Algorithm for Robust and Fast Ellipse Detection. *Pattern Analysis and Applications*, *8*(1-2), pp.149-162.

Yazdi, E., Azizi, V. and Haghighat, A.T., 2010, October. Evolution of Biped Locomotion Using Bees Algorithm, Based on Truncated Fourier Series. In *Proceedings of the World Congress on Engineering and Computer Science* (Vol. 1).

Yazdi, E., Azizi, V. and Haghighat, A.T., 2011. Biped Locomotion with Arm Swing, Based on Truncated Fourier Series and Bees Algorithm Optimizer. In I*ntelligent Automation and Systems Engineering* (pp.15-26). Springer New York.

Yin, Yunqiang; Wu, Wen-Hsiang; Cheng, Shuenn-Ren and Wu, Chin-Chia, 2012. An investigation on a 2-Agent Single Machine Scheduling Problem with Unequal Release Dates. *Journal of Computers and Operations Research*, 39 (12): 3062-3073.

Yu, E. L. and Suganthan, P. N., 2010. Ensemble of Niching Algorithms. *Journal of Information Sciences*, 180(15): 2815-2833.

Yuce, B., Mastrocinque, E., Lambiase, A., Packianather, M.S. and Pham, D.T., 2014. A Multi-Objective Supply Chain Optimisation Using Enhanced Bees Algorithm with Adaptive Neighbourhood Search and Site Abandonment Strategy. *Swarm and Evolutionary Computation*, *18*, pp.71-82.

Yuce, B., Packianather, M.S., Mastrocinque, E., Pham, D.T. and Lambiase, A., 2013. Honey Bees inspired Optimization Method: The Bees Algorithm. *Insects*, *4*(4), pp.646-662.

Yuce, B., Pham, D.T., Packianather, M.S. and Mastrocinque, E., 2015. An Enhancement to the Bees Algorithm with Slope Angle Computation and Hill Climbing Algorithm and Its Applications on Scheduling and Continuous-Type Optimisation Problem. *Production & Manufacturing Research*, *3*(1), pp.3-19.

Zabil, M. H. M.; Zamli, K. Z. and Othman, R. R., 2012. Sequence-Based Interaction Testing Implementation Using Bees Algorithm. In *2012 IEEE Symposium on Computers & informatics (ISCI), March 2012*, pp.81-85.

Zabil, M.H.M. and Zamli, K.Z., 2013. Implementing a T-Way Test Generation Strategy Using Bees Algorithm. *International Journal of Advances in Soft Computing & Its Applications*, *5*(3), pp.116-126.

Zambrano-Bigiarini, M., Clerc, M. and Rojas, R., 2013, June. Standard Particle Swarm Optimisation 2011 At CEC-2013: A Baseline For Future PSO Improvements. In *Evolutionary Computation (CEC), 2013 IEEE Congress on* (pp. 2337-2344). IEEE.

Zang, Hongnian; Zhang, Shujun and Hapeshi, Kevin, 2010. A Review of Nature-inspired Algorithms. *Journal of Bionic Engineering*, 7(Supplementary): S232–S237.

Zarea, H., Kashkooli, F.M., Mehryan, A.M., Saffarian, M.R. and Beherghani, E.N., 2014. Optimal Design of Plate-Fin Heat Exchangers by a Bees Algorithm. *Applied Thermal Engineering*, *69*(1), pp.267-277.

Zarei, K., Atabati, M. and Kor, K., 2014. Bee Algorithm and Adaptive Neuro-Fuzzy inference System as Tools for QSAR Study Toxicity of Substituted Benzenes to Tetrahymena Pyriformis. *Bulletin of Environmental Contamination and Toxicology*, *92*(6), pp.642-649.

Zarei, K., Atabati, M. and Moghaddary, S., 2013. Predicting The Heats of Combustion of Polynitro Arene, Polynitro Heteroarene, Acyclic and Cyclic Nitramine, Nitrate Ester and Nitroaliphatic Compounds Using Bee Algorithm and Adaptive Neuro-Fuzzy inference System. *Chemometrics and intelligent Laboratory Systems*, *128*, pp.37-48.

Zargartalebi, H., Attaran, B., Noghrehabadi, A.R. and Ghanbarzadeh, A., 2012. Simulating Flow in Partly Porous Region Using RBF Neural Network and the Bees Algorithm. *20th Annual International Conference on Mechanical Engineering-ISME2012 16-18 May, 2012, School of Mechanical Eng., Shiraz University, Shiraz, Iran.*

Zhang, J.R., Zhang, J., Lok, T.M. and Lyu, M.R., 2007. A Hybrid Particle Swarm Optimization–Back-Propagation Algorithm for Feedforward Neural Network Training. *Applied Mathematics and Computation*, *185*(2), pp.1026-1037.

Zhang, W. and Yen, G. G., 2013, June. A Quasi-Gradient and Cluster-Based Artificial Immune System for Dynamic Optimization. In *IEEE Congress on Evolutionary Computation (CEC'13),* June 2013, pp. 2306-2313.

Zhou, Z.D., Xie, Y.Q., Pham, D.T., Kamsani, S. and Castellani, M., 2015. Bees Algorithm for Multimodal Function Optimisation. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, P.0954406215576063.

Ziarati, K., Akbari, R. and Zeighami, V., 2011. On The Performance of Bee Algorithms for Resource-Constrained Project Scheduling Problem. *Applied Soft Computing*, *11*(4), pp.3720-3733.

Zou, R., Kalivarapu, V., Bhattacharya, S., Winer, E. and Oliver, J., 2015. Standard Particle Swarm Optimization on Source Seeking Using Mobile Robots. *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 5-9 January 2015, Kissimmee, Florida,* American Institute of Aeronautics and Astronautics, Inc.

APPENDICES

APPENDIX A

LIST OF HYBRID BEES ALGORITHM

Table A.1 Hybrid methods used with the Bees Algorithm and their applications

Hybrid methods	Applications
Heuristic:	
1. Heuristic Filling Procedure	Supply chain management (Dereli and Das, 2011)
2. Forward-backward Interchange Scheduling Heuristic	Resource constrained project scheduling (Ziarati et al. (2011)
3. Quick-and-Dirty	Sequence-based iteration software testing (Zabil et al. 2012; Zabil and Zamli, 2013)
4. Fix-and-Optimise	Multi-level capacitated lot sizing (Furlan and Santos, 2015)
Gradient-based local search:	
1. Gradient Descent	Weather forecasting (Khanmirzaei and Teshnehlab, 2010)
2. Steepest Descent	Constrained nonlinear optimal control (Alfi and Khosravi, 2012)
	Pattern classification (Pham and Darwish, 2010)
3. Kalman Filter	Electricity generation and power systems (Anantasate et al., 2010)
4. Newton Search	Numerical benchmark functions and single maschine scheduling (Yuce et al., 2015)
5. Hill Climbing	
Quasi- or Non-gradient local search:	
1. Nelder-Mead	Numerical benchmark functions (Mahmuddin and Yusof, 2009)
2. Variable Neighbourhood Search	Noisy multi-response surfaces (Aungkulanon and Luangpaiboon (2012)
Meta-heuristics:	
1. Simulated Annealing	Four Colur Map problem (Sadiq and Hamad, 2010); Substitution ciphers (Ali and Mahmod, 2015)
2. Particle Swarm Optimisation	Antenna design (Guney and Onay, 2007); Design of construction layout (Lien and Chen, 2012 and 2014); Numerical benchmark functions (Chen and Lien, 2012); Benchmark clustering problems (Dhote et al., 2013)
3. Cultural Algorithm	Electricity generation and power systems (Anantasate and Bhasaputra, 2011)
4. Genetic Algorithm	University timetabling (Phuc et al., 2011); Process mining (Saleem et al., 2013); Single machine scheduling (Packianather et al., 2014)
5. Harmony Search	University timetabling (Nguyen et al., 2012)

Table A.1 Continued

Hybrid methods	Applications
Meta-heuristics (continued):	
6. Firefly Algorithm	Noisy multi-response surfaces (Aungkulanon and Luangpaiboon (2012)
7. Differential Evolution	Control of biped robot (Massah B. et al., 2013)
8. Artificial Bee Colony	Constrained numerical functions (Tsai, 2014a)
Other algorithms:	
1. RealPaver (interval-based)	Software quality assessment (Wang et al. (2011, 2013)
2. Mixed Integer Programming	Capacitated facility location problem (Cabrera G. et al. (2012)
3. Otsu's Method	Multilevel image thresholding (Shatnawi et al., 2013a)
4. *k*-means Clustering Algorithm	Benchmark clustering problems (Pham et al., 2011); Document clustering (AbdelHamid et al., 2013)
5. Fuzzy *c*-means Clustering Algorithm	Benchmark clustering problems (Pham et al., 2008a)
6. Flux Balance Analysis	Optimisation of metabolite production (Koo et al., 2014)
7. ID3 (decision tree learning algorithm)	Intrusion detection (Eesa et al., 2015)
Multiple algorithms:	
1. Genetic Algorithm + Tabu Search	Benchmark clustering problems (Shafia et al., 2011)
2. Hill Climbing + Flux Balance Analysis	Gene knock-out strategy (Choon et al., 2012, 2013b)
3. Hill Climbing + Flux Balance Analysis + OptKnock	Gene knock-out strategy (Choon et al., 2013a, 2014a, 2015)
4. Hill Climbing + Flux Balance Analysis + OptKnock + Differential Evolution	Gene knock-out strategy (Choon et al., 2014b)
5. *k*-means + Ward's clustering algorithm	Clustering of data air pollution (Saini and Kaur, 2014); Benchmark clustering problems (Kataria and Rupal, 2014)
6. *k*-means + Harmony Search	Benchmark clustering problems (Bonab and Hashim, 2014)
7. Differential Evolution + *k*-means + Cluster centre initialisation algorithm	Benchmark clustering problems (Bonab et al., 2015)
8. Simulated Annealing + Hill Climbing	Examination timetable (Abdullah and Alzaqebah, 2013)

APPENDIX B

LIST OF THE BEES ALGORITHM APPLICATIONS

Table B.1 List of the Bees Algorithm applications

AREAS	APPLICATIONS	REFERENCES
Electrical & Electronic Engineering	Optimal power flow	Anantasate & Bhasaputra (2010); Bhasaputra et al. (2011); Sumpavakup et al. (2011); Sumpavakup et al. (2012); Leeprechanon & Phonrattanasak (2013)
	Placements of FACTS devices	Idris et al. (2009a); Idris et al. (2009b); Idris et al. (2010a); Idris et al. (2010b); Satheesh & Manigandan (2013)
	Power system applications	Rashtchi et al. (2010); Tudu et al. (2011); Kavousi et al. (2012); Lenin et al. (2014); Marghaki & Zayandehroodi (2015)
	Controller tuning	Jones & Bouffe (2008); Ercin & Coban (2011); Fahmy et al. (2011); Ghalipour et al. (2012); Coban & Ercin (2012); Metni & Lahoud (2013); Amirinejad et al. (2014); Farhang & Mazlumi (2014); Hadi et al. (2014)Sarailoo et al. (2014); Toloei et al. (2014); Arzeha et al. (2015); Assareh & Biglari (2015); Danaei & Khajezadeh (2015); Fahmy & Ghany (2015); Gholipour et al. (2015); Khalid et al. (2015); Sen & Kalyoncu (2015)
	Antenna design	Guney & Onay (2007); Guney & Onay (2010); Guney & Onay (2011); Malekzadeh et al. (2012); Guney & Onay (2013); Derakhshan & Shirazi (2014)
	Multi-objectives economic load dispatching	Pham et al. (2008d); Lee & Darwish (2008); Leeprechanon & Polratanasak (2010); Marzi et al. (2010); Pham et al. 2010); Vennila & Prakash (2012)
	Filter design	Pham & Koç (2010)
	Biped robot control	Yazdi et al. (2010); Yazdi et al. (2011); Massah et al. (2013)
	Renewable energy system	Lee & Kim (2010); Phonrattanasak (2011); Fahmy (2012); Phonrattanasak et al. (2013); Tolabi et al. (2013); Tolabi et al. (2014b); Tudu et al. (2014); Gao et al. (2015); Mechter et al. (2015)
	Motion estimation	Boumazouza et al. (2013)
	Circuits Design	Mollabakhshi & Eshghi (2013); Belaid et al. (2013)
	Motor design	Keshageh & Gholamian (2013); Braiwish et al. (2014); Braiwish et al. (2015); Khazaei et al. (2015)
	Optimal sizing & location	Tolabi et al. (2014a); Fathallahnezhad & Eslami (2015)
	Signal processing	Ebrahimzadeh & Mavaddati (2014)
	Sensor networking	Saad et al. (2008); Moussa & El-Sheimy (2011)

Table B.1 Continued

AREAS	APPLICATIONS	REFERENCES
Computer Science & Engineering	Image analysis	Azarbad et al. (2011); Bradford Jr. & Hung (2012); Nebti (2013); Shatnawi et al. (2013); Grega et al. (2014)
	Software testing	Zabil et al. (2012); Wang et al. (2012); Wang et al. (2013); Zabil & Zamli (2013)
	Pattern recognition	Pham et al. (2006b); Pham et al. (2006c); Pham et al. (2006d); Pham et al. (2007d); Pham et al. (2008c); Akkar (2010); Ghanbarzadeh (2010); Nebti & Boukerram (2010); Pham & Darwish (2010); Khosravi et al. (2011); Sherme (2011); Addeh & Ebrahimzadeh (2012); Alomari & Othman (2012); Attaran et al. (2012); Fahmy et al. (2012); Samadzadegan & Ferdosi (2012); Nebti & Boukerram (2012); Sherme (2012); Ebrahimzadeh et al. (2013); Attaran & Ghanbarzadeh (2014); Azarbad et al. (2014); Chen et al. (2014); Ebrahimzadeh et al. (2014); Kalami (2014); Khajehzadeh (2015); Marzi & Marzi (2015); Yang et al. (2015a); Yang et al. (2015b)
	Design of communication networks	Dhurandher et al. (2009); Sayadi et al. (2009), Bernardino et al. (2012); Taher & Masoudrahmani (2012)
	Internet traffic load balancing	Bernardino et al. (2011)
	Spam host detection	Sagayam & Akilandeswari (2012)
	Data mining	AbdelHamid et al. (2013); Ananthara et al. (2013); Saleem et al. (2013); Kataria & Rupal (2014); Poonam & Dhaiya (2015)
	Feature selection	Sadiq et al. (2012); Rufai et al. (2014)
	Forecasting/prediction	Khanmirzaei and Teshnehlab (2010); Azzeh (2011); Şenyiğit et al. (2012)
	Cloud computing	Darwish (2013)
Mechanical Engineering	Mechanical design	Pham et al. (2009a); Pham et al. (2009b); Mirzakhani et al. (2011); Moradi et al. (2011b); Natchi et al. (2011); Ahmad et al. (2012); Nafchi et al. (2012); Ahmad et al. (2014); Banooni et al. (2014); Zarea et al. (2014); Moradi et al. (2015)
	Robotics	Pham et al. (2007b); Ang et al. (2009); Pham and Kalyoncu (2009) Pham et al. (2009); Eldukhri & Pham (2010); Eldukhri & Kamil (2013); Masajed et al. (2013)
	Thermo-fluids	Assareh et al. (2011); MizanAdl and Ardakani (2012); Zargartalebi et al. (2012); Bahrainian et al. (2013); Biglari et al. (2013)
	Mechanics of structure	Moradi et al. (2010); Moradi &Alimouri (2012); Salamat & Raiesinezhad (2012); Salamat & Ghanbarzadeh (2012), Poor & Saz (2012); Moradi & Kargozarfard (2013); Moradi & Tavaf (2013); Moradi et al. (2014); Düğenci et al. (2015)
	Nano structures	Ahangarpour et al. (2012)
	Non-destructive testing	Parsa et al. (2013)
	Dynamic systems	Pham and Darwish (2009); Moradi et al. (2011a); Xu et al. (2011); Kazemi et al. (2012); Long (2015); Mohammadi & Nasirshoaibi (2015);

Table B.1 Continued

AREAS	APPLICATIONS	REFERENCES
Industrial Engineering	Production Scheduling	Pham et al. (2007c); Packianather et al. (2014); Rambod & Rezaeian (2014)
	Sequencing	Pham et al. (2007e); Ang et al. (2010)
	Facility layout design	Fon and Wong (2010)
	Product design & conceptual	Pham et al. (2008b); Ang et al. (2013)
	Material handling	Sayarshad (2010)
	Supply chain management	Triwate and Luangpaiboon (2010); Dereli and Das (2011); Luangpaiboon (2011); Mongkolkosol & Luangpaiboon (2011); Mastrocinque et al. (2013); Teimoeny & Haddad (2013); Mayteekrieangkrai & Wongthatsanekorn (2015); Wongthatsanekorn & Mayteekrieangkrai (2014); Yuce et al. (2014)
	Line balancing	Ozbakir and Tapkan (2010); Ozbakir and Tapkan (2011); Tapkan et al. (2012a); Tapkan et al. (2012b); Akpinar & Baykasoglu (2014a)
	Manufacturing cell formation	Pham et al. (2007a)
	Hospital service mobile robots assignment	Xu et al. (2010); Xu et al. (2011)
	Project Scheduling	Sadeghi et al. (2011); Ziarati et al. (2011); Sadeghi & Alahyari (2013); Ghasemi et al. (2015)
	Optimum placements of multi-camera	Chrysostomou & Gasteratos (2012)
	Lot-sizing	Furlan & Santos (2015)
	Manufacturing resource service management	Xu et al. (2012); Xu et al. (2015); Xie et al. (2015)
Civil Engineering	Construction site layout	Lien and Chen (2012); Chen & Lien (2012); Lien & Cheng (2014)
	Design open canal	Aydogdu & Akin (2011)
	Road maintenance	Krainyukov et al. (2014)
Chemical Engineering	Chemical process	Pham et al. (2008e); Alfi et al. (2011); Castellani et al. (2012)
	Organic compounds	Zarei et al. (2013)
	Waste water treatment	Zarei et al. (2014); Ghaedi et al. (2015a); Ghaedi et al. (2015b)
Education	Timetabling	Lara et al. (2008); Alzaqebah & Abdullah (2011); Khang et al. (2011); Phuc et al. (2011); Nguyen et al. (2012); Abdullah & Alzaqebah (2013)
	Test form construction	Songmuang & Ueno (2011); Songmuang et al. (2012)
Biological and medical	Learning gene regulatory network	Ruz & Goles (2011); Ruz & Goles (2012); Ruz et al. (2014)
	Protein structure prediction	Bahamish et al. (2008); Jana et al. (2015)
	Metabolic pathway	Leong et al. (2015)
	Haplotype reconstruction	PourkamaliAnaraki & Sadeghi (2015)
	Identification of gene knockout strategies	Choon et al. (2012); Choon et al. (2013a); Choon et al. (2013b); Choon et al. (2013c); Choon et al. (2014a); Choon et al. (2014b); Koo et al. (2014); Choon et al. (2015)

Table B.1 Continued

AREAS	APPLICATIONS	REFERENCES
Mathematics	Numerical functions	Pham et al. (2006a); Pham & Darwish (2008); Karaboga & Akay (2009b); Mahmuddin & Yusof (2009); Packianather et al. (2009); Pham & Castellani (2009); Li et al. (2010); Aghazadeh & Meybodi (2011); Muhamad et al. (2011); Chen & Lien (2012); Gao et al. (2012); Pham et al. (2012); Hussein et al. (2013); Shatnawi et al. (2013); Yuce et al. (2013); Akpinar & Baykasoglu (2014b); Hussein et al. (2014); Luo et al. (2014); Pham & Castellani (2014); Tsai (2014b); Far & Aghazadeh (2015); Pham & Castellani (2015); Yuce et al. (2015); Zhou et al (2015)
	Clustering	Pham et al. (2008a); Pham et al. (2011); Dhote et al. (2013); Bonab & Hashim (2014); Saini & Kaur (2014); Bonab et al. (2015)
	Constrained optimisation	Alfi & Khosravi (2012); Tsai (2014a)
	Combinatorial optimisation	Sadiq & Hamad (2010); Sadiq & Hamad (2012); Cabrera G. et al. (2012); Sagheer et al. (2012); Chmiel & Szwed (2015)
	Generalised assignment problems	Ozbakir et al. (2010); Tapkan et al. (2013)
	Circle packing problem	Li et al. (2010)
	Multi-response surface	Bera et al. (2011); Aungkulanon & Luangpaiboon (2012)
	Chaotic system	Gholipour et al. (2012)
	Noisy functions	Chai-ead et al. (2011)
Others	Financial	Korns (2012)
	Biometric	Tran et al. (2011)
	Marketing	Dehuri et al. (2008)
	Agricultural	Mehdinejadiani et al. (2013)
	Environmental	Nademian (2014)
	Cryptanalysis	Abdul-Razaq & Ali (2014); Ali & Mahmod (2015)
	Aircraft landing problem	Abdul-Razaq & Ali (2015)
	Business management model	Paul et al. (2014)
	Astronomy	Möller et al. (2013)

APPENDIX C

BENCHMARK TEST FUNCTIONS FOR GLOBAL OPTIMISATION

Table C.1 Benchmark test functions used for Chapter 3 and Chapter 4

Function	Equation	Search range	Minimum
Six Hump Camel Back (2D)	$f_1(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (4x_2^2 - 4)x_2^2$	$-5 \le x_i \le 5$	$x = (-0.0898, 0.7126) / (0.0898, -0.7126)$ $f(x) = -1.0316$
Shekel 10* (4D)	$f_2(x) = -\sum_{i=1}^{10} \frac{1}{\sum_{j=1}^{4}(x_j - A_{ij})^2 + c_i}$	$0 \le x_j \le 10$	$x = (4,4,4,4)$ $f(x) \approx -10.5319$
Trid (6D)	$f_3(x) = \sum_{i=1}^{6}(x_i - 1)^2 - \sum_{i=1}^{6} x_i x_{i-1}$	$36 \le x_i \le 36$	$f(x) = -50$
Moved Axis Parallel Hyper-ellipsoid (10D)	$f_4(x) = \sum_{i=1}^{D} 5ix_i^2$	$-5.12 \le x_i \le 5.12$	$x(i) = 5i$ $f(x) = 0$
Schwefel 1.2 (20D)	$f_5(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$-100 \le x_i \le 100$	$x = (0, \cdots, 0)$ $f(x) = 0$
Powell (24D)	$f_6(x) = \sum_{i=1}^{D/4}(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4$	$-4 \le x_i \le 5$	$x = (3, -1, 0, 1, \cdots, 3, -1, 0, 1)$ $f(x) = 0$
Sum of Different Power (30D)	$f_7(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	$-1 \le x_i \le 1$	$f(x) = 0$
Sphere (30D)	$f_8(x) = \sum_{i=1}^{D} x_i^2$	$0 \le x_i \le 10$	$x = (0, \cdots, 0)$ $f(x) = 0$
Griewank (30D)	$f_9(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-100 \le x_i \le 100$	$x = (0, \cdots, 0)$ $f(x) = 0$
Axis Parallel Hyper-ellipsoid (30D)	$f_{10}(x) = \sum_{i=1}^{D} ix_i^2$	$-5.12 \le x_i \le 5.12$	$x = (0, \cdots, 0)$ $f(x) = 0$
Ackley (30D)	$f_{11}(x) = -20e^{-0.02\sqrt{D^{-1}\sum_{i=1}^{D} x_i^2}} - e^{D^{-1}\sum_{i=1}^{D}\cos(2\pi x_i)} + 20$	$-35 \le x_i \le 35$	$x = (0, \cdots, 0)$ $f(x) = 0$
Schwefel 2.21 (30D)	$f_{12}(x) = \max_{1 \le i \le D} \lvert x_i \rvert$	$-100 \le x_i \le 100$	$x = (0, \cdots, 0)$ $f(x) = 0$

Table C.1 Continued

Function	**Equation**	**Search range**	**Minimum**
Schwefel 2.22 (30D)	$f_{13}(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$-100 \le x_i \le 100$	$x = (0, \cdots, 0)$ $f(x) = 0$
Quartic (30D)	$f_{14}(x) = \sum_{i=1}^{D} i x_i^4 + random(0,1)$	$-1.28 \le x_i \le 1.28$	$x = (0, \cdots, 0)$ $f(x) = 0$
Alpine (30D)	$f_{13}(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1 x_i \rvert$	$-10 \le x_i \le 10$	$x = (0, \cdots, 0)$ $f(x) = 0$

*Coefficients A_{ij} and c_j are as defined in Jamil and Yang (2013).

`

APPENDIX D

BENCHMARK TEST FUNCTIONS FOR MULTIMODAL OPTIMISATION

Table D.1 Benchmark test functions for Chapter 5

No.	Name	Functions	Range
1	Two-peak trap	$f(x) = \begin{cases} \frac{160}{15}(15-x), for\ 0 \le x \le 15 \\ \frac{200}{5}(x-15), for\ 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$
2	Central two-peak trap	$f(x) = \begin{cases} \frac{160}{10}x, & for\ 0 \le x \le 10 \\ \frac{160}{5}(15-x), & for\ 10 \le x \le 15 \\ \frac{200}{5}(x-15), & for\ 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$
3	Five-uneven-peak trap	$f(x) = \begin{cases} 80(2.5-x), & for\ 0 \le x \le 2.5 \\ 64(x-2.5), & for\ 2.5 \le x \le 5 \\ 64(7.5-x), & for\ 5 \le x \le 7.5 \\ 28(x-7.5), & for\ 7.5 \le x \le 12.5 \\ 28(17.5-x), & for\ 12.5 \le x \le 17.5 \\ 32(x-17.5), & for\ 17.5 \le x \le 22.5 \\ 32(27.5-x), & for\ 22.5 \le x \le 27.5 \\ 80(x-27.5), & for\ 27.5 \le x \le 30 \end{cases}$	$0 \le x \le 30$
4	Equal maxima	$f(x) = sin^6(5\pi x)$	$0 \le x \le 1$
5	Decreasing maxima	$f(x) = \exp[-2\log(2) \cdot (\frac{x-0.1}{0.8})^2] \cdot sin^6(5\pi x)$	$0 \le x \le 1$
6	Uneven maxima	$f(x) = sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$
7	Uneven decreasing maxima	$f(x) = \exp\left(-2\log(2) \cdot \left(\frac{x-0.08}{0.854}\right)^2\right) \cdot sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$
8	Himmelblau	$f(x_1, x_2) = 200 - ({x_1}^2 + x_2 - 11)^2 - (x_1 + {x_2}^2 - 7)^2$	$-6 \le x_1, x_2 \le 6$
9	Camel back	$f(x_1, x_2) = -4\left[\left(4 - 2.1{x_1}^2 + \frac{{x_1}^4}{3}\right){x_1}^2 + x_1x_2 + (-4 + 4{x_2}^2){x_2}^2\right]$	$-1.9 \le x_1 \le 1.9$ $-1.1 \le x_2 \le 1.1$
10	Shekel's foxholes	$f(x_1, x_2) = 500 - \frac{1}{0.002 + \sum_{i=0}^{24} \frac{1}{1 + i + (x_1 - a(i))^6 + (x_2 - b(i))^6}}$ where $a(i) = 16((i\ mod\ 5) - 2)$, and $b(i) = 16(\left\lfloor\left(\frac{i}{5}\right)\right\rfloor - 2)$	$-65.536 \le x_1, x_2 \le 65.535$