ZULKIFLI BIN ZAINAL ABIDIN

FRUIT-FLY BASED SEARCHING ALGORITHM FOR COOPERATIVE SWARMING ROBOTIC SYSTEM

ZULKIFLI BIN ZAINAL ABIDIN

UNIVERISITI SAINS MALAYSIA

2013

2013 P.hD

FRUIT-FLY BASED SEARCHING ALGORITHM FOR COOPERATIVE SWARMING ROBOTIC SYSTEM

by

ZULKIFLI BIN ZAINAL ABIDIN

Thesis submitted in fulfillment of the requirements
for the degree of
Doctor of Philosophy

July 2013

ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude to the Almighty; with His Blessing, I've been given the opportunity to complete this thesis and brought me to where I am today. Alhamdulillah.

This thesis will not be possible without endless support, advice and encouragement from my supervisor Assoc. Prof. Dr. Mohd Rizal Arshad. Furthermore, my co-supervisor Assoc. Prof. Dr. Umi Kalthum Ngah for her patience and continuous guidance, especially on the writing process which I'm greatly indebted by her assistance.

My heartfelt appreciation goes to the URRG lab members of Universiti Sains Malaysia, who has been providing great working environment throughout the four years, especially Khairul Izman Bin Abdul Rahim, Mohd Ikhwan Hadi Bin Yaakob and Nur Afande bin Ali Hussain, Ong Boon Ping and Kok Chee Hou whom always lend their hand whenever in need.

I would also like to thank my family and friends for their love and prayers.

Last but not least, to all the lecturers, staff and the financial support of the National Oceanography Department, Malaysia, under grants NOD-USM 6050124 and USM Research University, under grant 1001/PELECT/814059.

TABLE OF CONTENTS

CHAPTER 6 : RESULTS AND DISCUSSIONS

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

AB	Adaptive Behaviour
ACO	Ant Colony Optimisation
AI	Artificial Intelligence.
APF	Artificial Potential Field
ASV	Autonomous Surface Vehicle
BA	Bee Algorithm
CS	Cuckoo Search
FA	Firefly Algorithm
FOA	Fly Optimisation Algorithm
GSO	Glowworm Swarm Optimisation
MA	Metaheuristic Algorithm
MS	Monkey Search
PSO	Particle Swarm Optimisation
SI	Swarm Intelligence
SR	Swarming Robot
UAV	Unmanned Aerial Vehicle
VRP	Vehicle Routing Problem
BFOM	Bacteria Foraging Optimization Methods.

ALGORITMA CARIAN BERDASARKAN LALAT BUAH UNTUK KERJASAMA SEKUMPULAN SISTEM ROBOTIK

ABSTRAK

Kepintaran kumpulan boleh digambarkan sebagai tingkah laku yang kompleks, terhasil daripada sejumlah besar ejen individu, yang mana setiap ejen mematuhi peraturan yang amat mudah. Ia sebenarnya di ilhamkan dengan memahami mekanisme berpusat bagi organisasi haiwan semula jadi seperti burung, semut, lebah, ulat cahaya, dan kelip-kelip. Pemerhatian tingkah laku biologi ini menghasilkan sekumpulan robotik yang berupaya untuk bekerja antara satu sama lain dalam kumpulan bagi mencapai keselarian, keteguhan dan keupayaan kolektif. Tingkah laku kolektif adalah strategi pergerakan seperti "sumber carian" dan "kesatuan" yang biasa dipamerkan oleh haiwan semasa mencari sumber makanan. Walaubagaimanapun, keadaan untuk robot adalah pergerakan bagi mencari punca bau, cahaya, dan bunyi. Dalam pada itu, terdapat minat yang meningkat, terutamanya dalam mencari lokasi yang paling dalam di tasik dan di empangan untuk sistem kajian batimetri. Kaedah yang sedia ada seperti “lawnmower” melibatkan kos yang besar dari segi masa, ketepatan dan kebolehpercayaan. Oleh itu, penggunaan sistem robotik berkumpulan dicadangkan. Di dalam tesis ini, rangka kerja yang ringkas dan kaedah dalam membangunkan algoritma bio-inspirasi telah dibangunkan untuk aplikasi koperasi sekumpulan robot. Strategi pergerakan lalat buah atau *Drosophila Melanogaster* mempamerkan beberapa kelebihan seperti kitaran strategik carian berkumpulan, corak pergerakan dengan pengedaran rawak Levi, perkongsian maklumat dalam masa nyata dan pengurangan parameter pengawal semasa pergerakan. Beberapa proses fungsi penanda aras telah dijalankan untuk menilai prestasi yang dicadangkan iaitu FOA (*Fly Optimisation Algorithm*). Pembentukan berkumpulan

kemudiannya diselakukan dengan perisian alat Netlogo. Strategi carian bio-inspirasi optima telah di aplikasikan kedalam sekumpulan kenderaan permukaan berautonomi kecil yang dinamakan *Drosobots.* Untuk mengesahkan prestasi sebenar sistem yang dibangunkan, platfom robotik yang baru ini di uji pada sebuah kolam renang bersaiz Olimpik. Dalam keadaan permukaan air tenang, kesilapan pelayaran ASVs dapat dikurangkan. Keputusan menunjukkan bahawa tingkah laku yang terbentuk dalam pergerakan lalat buah adalah sangat baik dengan menggunakan bilangan ejen yang kecil.

FRUIT-FLY BASED SEARCHING ALGORITHM FOR COOPERATIVE SWARMING ROBOTIC SYSTEM

ABSTRACT

Swarm intelligence can be described as a complex behaviour generated from a large number of individual agents, where each agent follows very simple rules. It is actually inspired by understanding the decentralized mechanisms in the organization of natural swarms such as the birds, the ants, the bees, the glowworms, and the fireflies. Observation of these biological behaviour has given birth to swarm robotics whereby robots have the capability to work with one another in a group to achieve the same kind of parallelism, robustness and collective capabilities. A collective behaviour movement strategy such as a “source search” and “aggregation” are commonly exhibited by the animals while finding their source of food. However, the situation for the robots is to find the source of odour, light, and sound. Meanwhile, there has been mounting interest, particularly for finding the deepest location in lakes and dams for bathymetric survey systems. Using the existing lawnmower methods incur substantial costs in terms of time, accuracy and reliability. Therefore, the usage of a swarming robotic system is proposed. In this thesis, a simple framework and methodology in developing a bio-inspired algorithm for cooperative swarming robotic application has been developed. The fruit flies or *Drosophila Melanogaster* movement strategy offers some advantages such as strategic 'search-aggregation' cycle, distribution of moving patterns with Levy Random, information sharing in real-time, and reduction of controller parameters during movements. A number of benchmark function processes were conducted to assess the performance of proposed FOA (Fly Optimisation Algorithm). The swarming formation is then simulated with the Netlogo simulation tool software. This optimal bio-inspired

searching strategy has been incorporated into the application of a swarm of mini autonomous surface vehicles (ASVs) named Drosobots. In order to verify the actual performance on the system developed, this new robotic platform is tested on an Olympic sized swimming pool. Under calm water surface conditions, the navigational error of the ASVs can be minimized. The results show that the emergent behaviour of fruit flies is very well organized involving a small number of agents.

CHAPTER 1

INTRODUCTION

1.1 Overview

In earlier years, the creation of robots are generally cylindrical in shape, having static hands, wheels like a car, with the purpose of assisting in the construction industry, manufacturing, and search and rescue operations. However, the development of bio-mimetic (nature inspired) approaches in the early 1990s has caused researchers to shift their robotic design perception. In this perspective, each shape of the animals has its own specialty and role. In addition, the "thinking like an animal" has a place in the field of computer science and is used in programming or Soft Computing. In recent years, Swarm Intelligence (SI) and Adaptive Behaviour (AB) have garnered attention in the field of robotics and Artificial Intelligence (AI) applications. These are also known as Bio-inspired Algorithms.

Two branches of applications which are frequently utilising bio-inspired algorithms or SI are;1) Optimisation and 2) Prediction and Forecasting. Swarming Robot (SR) is the one of the topic which included under optimisation. The overview of the optimisation, SR, and prediction and forecasting are explained in this part. Optimisation applications are very broad, traversing from one specific area and spreading into engineering design such as;

i. Mechanical engineering design (Schwabacher *et al*., 1998; Coelho and Mariani, 2008; Kashan, 2011),
ii. Process optimization (Egea *et al*., 2010; Joshi and Pande, 2011; Kwak and Kim, 2012),

iii. Scheduling system (Andersson *et al*., 2007; Frantzen *et al*., 2011; Skobelev, 2011),

iv. Routing and flow control in networks and networking (Madan *et al*., 2007; Shakkottai and Srikant, 2007; Minoux, 2010),

v. Service oriented applications in finance *(*He *et al*., 2008*;* Leibfritz and Maruhn, 2009; Pennanen, 2011),

vi. Healthcare system design (Harrell and Lange, 2001; Bagirov and Churilov, 2003; Patriarca-Almeida *et al*., 2011),

vii. Bioinformatics (Hernandez and Kambhampati, 2004; Nebro *et al*., 2008)

For example, in the formulation of the scheduling system; optimisation algorithm can be used to determine the path of vehicle systems to the various destinations, determining the job schedules in factories, scheduling lectures at universities, creating multiple timetables, computer network design and planning strategies.

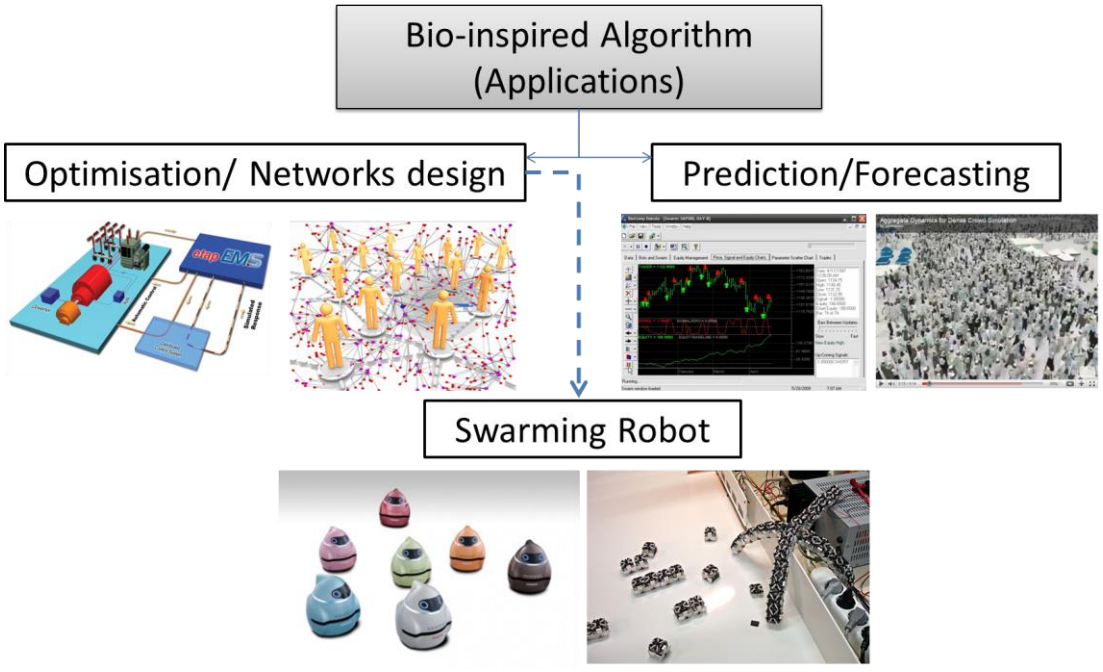


Figure 1.1: Example applications of Bio-inspired Algorithms.

The searching strategies are also designed to accommodate different situations, which are determined by historical information, (i.e. to predict or to forecast) (Lou *et al.,* 2011). Examples of such applications are;

i. Housing market fluctuations (Azadeh *et al*., 2012),
ii. Trend adjustment for electricity demand forecasting (Wang *et al*., 2011),
iii. Short-term food price forecasting in China (Haofei *et al*., 2007),
iv. Financial Forecasting (Kim, 2004),
v. Short-time weather forecasting (Kilifarev *et al*., 2008),
vi. The model of rainfall forecasting (Zhao and Wang, 2010),
vii. Urban traffic forecasting model (Hong *et al*., 2007),
viii. Forecasting output of integrated circuit industry (Pai *et al.*, 2009),
ix. Traffic safety forecasting (Gang and Zhuping, 2011),
x. Simulating believable crowd and group behaviors (Huerre *et al*., 2010),
xi. Tawaf simulation for Hajj training application (Rahim *et al*., 2011),
xii. Crowd modeling and traffic simulation (Lin and Manocha, 2010).

If viewed at random, natural life cycles that happen around people are actually closely aligned with each other. The movement of tens of thousands of birds and fish swarming produce shades and shapes of certain formations. Based on the natural life cycle, the weather conditions that occur prior to the moment in time could also be reenacted through computer simulations databases.

The concept of SR system is adopted from nature; from the appearance of flocking birds, the movement of a school of fish, the ant colonies and swarming bees among others. This "emergent behavior" is the aggregation result of many simple interactions occurring within the animals themselves (Corner and Lamont, 2004). It

is a broad field of computational swarm intelligence and is applied in swarming robotics (Jevtic and Andina, 2007). Today, swarm robotics is one of the most popular research areas in robotic technology. SR brings together experts in artificial intelligence, control theory, robotics, systems engineering and biology with the goal of understanding swarming behaviours in nature and applications of biologically-inspired models of swarm behaviors to large networked groups of autonomous vehicles.

The basic criteria in designing the SR's hardware system are; mechanical design, dynamic and kinematic design, communication, sensory system and power management (Brambilla *et al,*. 2012). The design of the components is dependent upon the mission and the biological behavior itself. SI is also related to Metaheuristic Algorithm (MA) development. For many years, SI has focused more on the virtual simulation, where MA in applied on the agents in order to observe the aggregation behaviours. As far as MA is concerned, each animal inspired algorithm has its own strength and capability based on its natural behaviour.

Some of the SR and bio-inspired project are working on a miniature platform where the mobile robots' agents are assigned to form aggregation behaviour within the field work with a camera based localisation technique (Arvin *et al*., 2009; Martínez *et al*., 2010; Hereford and Siebold, 2010; Meng *et al*., 2011; Schmickl, 2011). The information is used to emulate the minimum/ maximum searching area or virtually named Artificial Potential Fields (APF). For the real situation, the particular location might be the most hazardous zone, highest peak on the ground or deepest part of the water column (i.e., the potential site that the system is searching for).

In underwater surveying expedition, finding the deepest part of the aquatic area such as oceans, lakes, dams, ponds and rivers are also considered as one of the most difficult task. This kind of mission requires a lot of efforts, and proper planning is necessary. One of the prime expeditions carried out at this particular moment in time is the one which involves locating the actual GPS position of Mariana Trench. Most of the voyages used multi-beam sonar and their movement is simple "lawn mower" formations to find the specific coordinate (Nishimura, 2011). For biologists, the area serves as keen interest as it may contain various new organisms and microorganisms.

1.2 Problem Statement

Today, bio-inspired algorithm for optimization applications is no longer restricted only to the computational and simulation problem, but has also been used in other fields such as swarming robotics system. Most of the studies conducted are either focusing on the optimisation capability and robustness of the algorithm or the capability of the algorithm or robotic platforms mimicking the animal's formation. For example, the collective movement of flocking birds and shoaling fish were successfully implemented on Nissan EPORO project for accident avoidance system and traffic congestion management for automatic car driving of the future (Nissan, 2009). Another example is a Micro Air Vehicles (MAVs) that navigate through a dynamic grid composed of node-MAVs using pheromone based rules inspired from army ant foraging. MAVs maintain a virtual map of the pheromone, which is only possible with positioning information (Hauert *et al.,* 2010). Maintaining the positional accuracy in a collective manner is the main task for the existing cooperative SR system (Levi and Kernbach, 2010, Valenti, 2007, Oung, 2010, Rubenstein, 2011).

In some situations, the bio-inspired algorithms are not necessarily applicable to all SR applications. According to Yang (2010), the bio-inspired algorithm is still widely open for different animal species. Each algorithm is not necessarily suitable for every case or situation. However, the framework and methodology of realizing the idea from animal's perception to the SR is too extensive. In general, it is a multi-disciplinary in nature and it requires expertise from various fields such as biologists, system engineers, and experts on robotics, control system theory, and artificial intelligence (Kumar, 2010). Based on the existing SR development and applications, those strategies might not be suitable for other applications such as to locate the deepest part of the lakes and dam. This research area has triggered some interest and new challenges (Gutierrez *et al.*, 2010). The use of existing bathymetric techniques involves a lot of efforts, time and money for scientists to plot and identify the deepest area using a survey vessel. Thus, for this application, the usage of SR is proposed with an optimised bio-inspired algorithm. By analyzing the performance of the existing bio-inspired algorithm particularly in collective motion behaviour, this study is expected to find a novel approach which is suitable for the suggested SR application. Finally, in order to experiment on the capability of the proposed algorithm, the actual robotics platform or the ASVs would be tested comprehensively.

1.3 Objectives

The main goal of the research is to identify an appropriate bio-inspired fast searching algorithm whereby the secondary goal would be to use the algorithm for a suitable swarming robotics (SR) application. Specifically, in this study, the research objectives are:

1. To investigate and develop a simple framework and methodology in developing a bio-inspired fast searching algorithm.
2. To assess the performance of proposed FOA (Fly Optimisation Algorithm) on a number of benchmark functions.
3. To develop a new swarming robot platform.

This involves a large number of autonomous surface vessels systems made up of small robotic platforms, equipped with multiple sensors inspired by collective behaviour of animal species.

1.4 Research Scope

This research will focus on the related processes development that is necessary in finding suitable and appropriate bio-inspired algorithm which forages in a smart and efficient manner for an SR application. The research scopes are explained as below:

1. Details formulation of the algorithm will be not cover comprehensively. This study is focusing more on the real application which is to analyze the performance of the collective behavior of the proposed animal for cooperative swarming robotic system.
2. The performance and comparison are grouped into animal based algorithm only. Other bio-inspired algorithms such as Neural Networks, Genetic Algorithm, Intelligent Water Drops Algorithm and *etc* are not included.

3. Due to fieldwork accessibility, and minimizing the positioning error, an Olympic-size swimming pool is used as the test platform. The water surface is expected to be calm so that the disturbance of the ASV's prototype is tolerable.

1.5 Thesis Outline

Chapter One presents an overview about SI, SR system, suitable engineering application and problem statements. The main objectives of this research and research scope are also presented in this chapter.

Chapter Two will discuss the existing diversity of algorithm and their applications towards optimisation, networking design, forecasting, prediction and swarming robots. An overview on the challenges of ASVs development for lakes and river mapping usage is also presented.

Chapter Three will touch on the biological behaviour and the movement model of the fruit flies and its capability. An extensive study of this animal species is formulated for the algorithm to suit the swarming ASV application.

Chapter Four presents the major contribution of the study whereby the proposed algorithm is synthesized and tested using mathematical modelling and simulation. This includes the benchmark functions comparisons with Matlab™ and pattern behaviour analysis with NetLogo™. Chapter 4 also illustrate how simulation-based methods can be applied to analyse and evaluate the performance of the system under development.

Chapter Five is the experimental work towards realisation of the swarming robot application. It covers the ASV design requirements and all the development processes involved, the virtual robot simulator using Webots™, and the fieldwork trial.

Chapter Six discusses and analyses the results obtained, and shows the viability and feasibility of the FOA as an optimised searching algorithm.

Chapter Seven concludes the research and also provides some future research recommendations to enhance the proposed FOA performance.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Any natural system that congregate as a result of some form of collective intelligence of nature is known as Swarm Intelligence (SI). This metaphor inspires a variety of techniques to solve in most cases, the problem of calculating optimisation problems. It has sparked interest amongst numerous scientists (Chu *et al*., 2011; Martens *et al*., 2011; Panigrahi *et al.*, 2011; Sudholt, 2011). Optimisation is one of the techniques in seeking ideal values of variables that lead to the best possible value of the function where the perfection is not necessarily important for the particular problem (Yang, 2008).

Recent approaches in the optimisation techniques are mostly based on natural phenomenon and behavioral observation. Genetic algorithms (GA)(Glover, 1994) and evolutionary algorithms (EA) (Thomas, 1996), generate solutions to optimisation problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. Harmony Search (HS) (Geem *et al*., 2001) is inspired by the improvisation process of musicians. Intelligent Water Drops algorithm (IWD) (Duan *et al*., 2009) is a nature-inspired optimisation algorithm, which is based on the natural flow of rivers in finding almost optimal paths to their destination. Gravitational search algorithm (GSA) (Rashedi *et al*., 2009) is constructed based on the law of gravity and the motion of mass interactions. On the other hand, techniques related to behavioral observation include the Ant colony optimisation (ACO) (Dorigo *et al*., 2006), which is a class of optimisation algorithms modeled on the actions of an ant colony. Artificial immune systems (AIS) are

computational systems inspired by the principles and processes of the vertebrate immune system. Particle swarm optimisation (PSO) (Poli and Kennedy., 2007) is inspired by the social behavior of bird flocking or fish schooling. Photosynthetic algorithm (PA) (Murase, 2000) utilises the darkening reaction rules governing the transfer of carbon molecules from one substance into another in the Calvin–Benson cycle and photorespiration. Galaxy-based search algorithm (GbSA) (Shah-Hosseini, 2011) imitates the spiral arm of spiral galaxies in searching for its surroundings.

Most of Animal-based Metaheuristic Algorithms are also related to swarm intelligence. Swarm intelligence has gathered enormous research interest in related fields in recent years. Swarm intelligence is defined as ‘‘Any attempt to design algorithms or distributed problem-solving devices inspired by the collective behavior of social insect colonies and other animal societies.’’ (Bonabeau *et al.,* 1999). Swarming behavior is one of the sub chapters under biological based or bio-mimicry or bio-inspired algorithm concept. It is a concept adopted from collective activities shown by a group of people or animal aggregating and forming an emergent behavior naturally.

2.2 Animal Inspired Algorithm (Swarming AI Design and Approach)

Some examples of animal inspired metaheuristics adopted are: Ant Colony Optimisation (ACO)(Dorigo, 1992), Particle Swarm Optimisation (PSO) (Eberhart and Kennedy, 1995), Monkey Search (Mucherino and Seref, 2007), Bee Algorithm (BA) (Nakrani and Tovey, 2004), Firefly algorithm (FA) (Yang, 2008), Artificial Bee Colony Algorithm (ABC) (Karaboga and Basturk, 2008), Glowworm Swarm Optimisation (GSO)(Krishnanand and Ghose, 2008) and Cuckoo Search (CS)(Yang, 2010).

2.2.1 Ant Colony Optimisation (ACO)

The Ant Colony Optimisation algorithm (ACO) was introduced by Dorigo (1992) in his PhD thesis. ACO is a probabilistic technique for solving computational problems, which can be reduced by finding good paths through graphs. According to Dorigo *et al.*(1996), the collective behavior that emerges is a form of autocatalytic behavior where the more member of ants following a trail, the more attractive that trail becomes for other to follow. 20 years after the founding of the algorithm, as of today, the ACO has become well-established and complicated because of the hybridisation's evolvement. However, the basic concept and algorithm can be easily understood as follows (Figure 2.1):

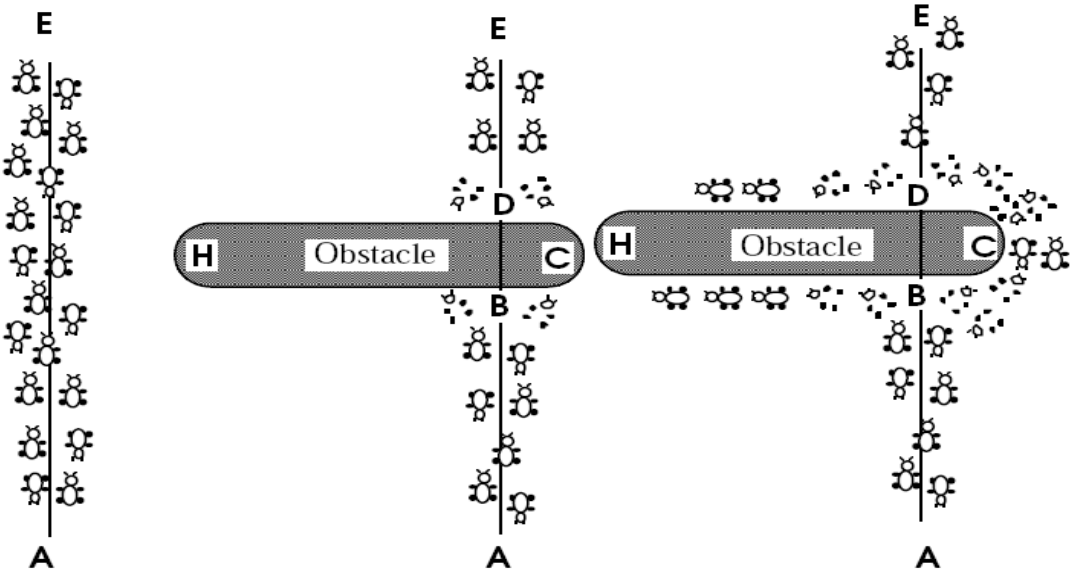


Figure 2.1: An example with real ants (Dorigo *et al.* 1996).

a) Ants follow a path between points A and E.

b) An obstacle (H – C) is interposed; ants can choose to go around it following one of the two different paths with equal probability.

c) After sometime, the shorter path is where there exist more pheromones laid down, which is at C.

Pseudo-code and formulas published by Marco Dorigo is as below:

```
Algorithm 1 The Ant Colony Optimisation Metaheuristic
Set parameters, initialise pheromone trails
   while termination condition not met do
        ConstructAntSolutions
        ApplyLocalSearch(optional)
        UpdatePheromones
   Endwhile
```

Figure 2.2: Pseudo-code of ACO (Dorigo *et al.*, 1996).

In general, the *k*th ant moves from state *x* to state *y* with probability

$$p_{xy}^{k} = \frac{(\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})}{\Sigma(\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})} \tag{2.1}$$

Where τ_{xy} is the amount of pheromone deposited for transition from state *x* to *y*, $0 \leq \alpha$ is a parameter to control the influence of τ_{xy}, η_{xy} is the desirability of state transition *xy* (*a priori* knowledge, typically 1/ d_{xy}, where *d* is the distance) and $\beta \geq 1$ is a parameter to control the influence of η_{xy}. When all the ants have completed a solution, the trails are updated by

$$\tau_{xy} = (1-\rho)\tau_{xy}^{k} + \Delta\tau_{xy}^{k} \tag{2.2}$$

Where$\tau_{xy}^{k}$is the amount of pheromone deposited for a state transition *xy*, ρ is the *pheromone evaporation coefficient* and $\Delta\tau_{xy}^{k}$is the amount of pheromone deposited, typically given for a TSP problem (with moves corresponding to arcs of the graph) by

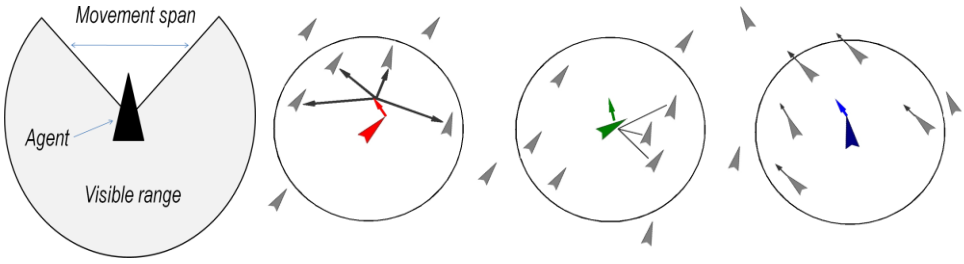
$$\Delta\tau_{xy}^{k} = \begin{cases} \frac{Q}{L_k}, & if ant\ k\ uses\ curve\ xy\ in\ its\ tour \\ 0, & otherwise \end{cases} \tag{2.3}$$

Where L_k is the cost of the *k*th ant's tour (typically length) and *Q* is a constant.

2.2.2 Particle Swarm Optimisation (PSO)

PSO is originally attributed to (Kennedy and Eberhart, 1995). A stylised representation of the movement of organisms in a bird flock or fish school is simplified based on the simulation done by Reynolds (1987). The algorithm was observed to demonstrate optimisation. He had proposed a behavioral model in which each agent follows three rules: *Separation*- Each agent tries to move away from its neighbours if they are too close. *Alignment*- Each agent steers towards the average heading of its neighbours. *Cohesion*- Each agent tries to more towards the average position of its neighbours. An individual agent should change its heading or direction and velocity based on the positions and velocities of its nearby neighbours.

Figure 2.3 (a) is an illustration of an individual agent where the visibility range is the variable angle where how far each agent can see or sense from its position is defined; while the movement span is a set of maximum angles that are available for the agent to change its direction either to the left or right. Figure 2.3 shows three basic strategies of Reynolds' flocking rules. The circles indicate sensing range for the boids in the centre. This means that the boid in the centre of the circle can see or sense other boids within the radius angle. Figure 2.3(a) representation of an individual agent and Figure 2.3 (i to iii) illustrates Reynolds's basic flocking steering strategies. The circle indicates the neighbourhood range of the agent's in the centre of the circle. (i) shows cohesion, (ii) shows separation, and (iii) shows alignment strategy respectively (Othman, 2009).



(a) (i) (ii) (iii)

Figure 2.3: Reynolds's basic flocking steering strategies.

i. The boid feels the urge to steer towards the average position of its flockmates in its vicinity, resulting in the boids staying close to one another.

ii. This strategy is to ensure that the boid is maintaining a safe distance from its flockmates and encourages the boid population to avoid crowding the neighbourhood.

iii. The alignment strategy which sometimes is referred to as the velocity matching strategy. This rule encourages the boid to move with a similar heading and velocity as its neighbours.

The motion is based on their position known in the search space and the position of swarm around the best-known position. When a better position is found it will then guide the movement of the swarm. This process is repeated and thus it is expected, but not guaranteed, that a satisfactory solution will eventually be found.

2.2.3 Bee Algorithm (BA)

From literature review, it seems that the Bee Algorithm was first formulated around 2001 by Abbass (2001) where a colony contains a single queen with multiple workers. The model is used to solve a special class of propositional satisfiability problems (SAT) known as 3-SAT, where each constraint contains exactly three variables. Then in 2004 Nakrani and Tovey (2004) from Oxford University studied a method to allocate computers among different clients and web-hosting servers (Bee Algorithm). In 2005, Haddad *et al.* (2005) presented a Honey Bee Mating Optimisation (HBMO) algorithm to solve their reservoir modeling and clustering. Yang (2008) also formulated the algorithm to solve numerical optimisation problem and named it as the Virtual Bee Algorithm (VBA). In 2008, Karaboga and Basturk (2008) reformulated the algorithm and named it as the Artificial Bee Algorithm. In the ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. The first half of the colony consists of the employed artificial bees while the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources (Karaboga and Basturk, 2008).

The main steps of the algorithm are given below (Figure 2.4):

```
Bee Algorithm
Initialise
REPEAT
-Move the employed bees onto their food sources and
determine their nectar amounts.
-Move the onlookers onto the food sources and
determine their nectar amounts.
-Move the scouts for searching new food sources.
-Memorise the best food source found so far.
UNTIL (requirements are met)
```

Figure 2.4: Pseudo-code of Bee Algorithm.

Each cycle of the search consists of three steps: moving the employed and onlooker bees onto the food sources and calculating their nectar amounts and determining the scout bees and then moving them randomly onto the possible food sources.

A detailed Pseudocode of the ABC Algorithm proposed by Karaboga and Basturk (2008) is as follows;

Initialise the population of solutions $x_{i,j}$
Evaluate the population
cycle=1
repeat
Produce new solutions (food source positions) $v_{i,j}$ in the neighborhood of $x_{i,j}$for the employed bees using the formula $v_{i,j} = x_{i,j} + \square_{i,j}(x_{i,j} - x_{k,j})$ (k is a solution in the neighborhood of i, $\square$ is a random number in the range [-1,1])and evaluate them.
Apply the greedy selection process between xi and ui
Calculate the probability values P_i for the solutions x_i by means of their fitness values using the equation (2.4):

$$P_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \qquad (2.4)$$

In order to calculate the fitness values of solutions we employed the following equation (2.5):

$$fit_i = \begin{cases} \frac{1}{1+f_i} & if f_i \geq 0 \\ 1 + abs_i & if f_i \leq 0 \end{cases} \qquad (2.5)$$

NormaliseP_i values into [0,1]
Produce the new solutions (new positions) v_i for the onlookers from the solutionsx_i, selected depending on P_i, and evaluate them
Apply the greedy selection process for the onlookers between xi and v_i
Determine the abandoned solution (source), if exists, and replace it with a new randomly produced solution x_i for the scout using the equation (2.6)

$$x_{ij} = min_j + rand(0,1) * (max_j - min_j) \qquad (2.6)$$

Memorise the best food source position (solution) achieved so far.
cycle=cycle+1
until cycle= Maximum Cycle Number (MCN)

2.2.4 Monkey Search (MS)

According to Mucherino and Seref (2007), when climbing up a tree for the first time, the monkey only chooses the branches of the tree in a random way. This is because it does not have any previous experience on that particular tree. However, when it climbs up the tree again, it tries to follow the paths that will lead it to good food. This practice allows the monkey to discover a set of connected branches of the tree in which there are good food resources. Like the chimpansee, memorisation is also very important for the monkey. Subsequently, the monkey leaves some kind of "marker" to the branches to be used later, while returning to the ground. The monkey chooses among the several branches based on the marks it left before. Based on this high probability in finding better solution, Mucherino and Seref (2007) proposed a Monkey Search (MS) Algorithm for global optimisation inspired by the behavior of a monkey climbing trees looking for food.

Figure 2.6 gives a graphic representation of the monkey behavior and its algorithm. In Figure 2.6(a), the monkey climbs a new tree for the first time. At each step, two new solutions are generated and placed on two nodes of the tree. The dashed arcs are the ones the monkey rejects, and all the others form a path on the tree. Every new path is considered as additional weights (*w)*. When the top of the tree is reached *(level = maxlevels),* the monkey climbs the chosen arcs in the opposite direction and marks them (the weights *w* are modified with the best improvement available in the direction of the corresponding arc). At a certain point, the monkey restarts climbing up (see the node marked by a blue square in Figure 2.6(b). Then, new solutions are generated and one of them is chosen, until the top of the tree is reached again.

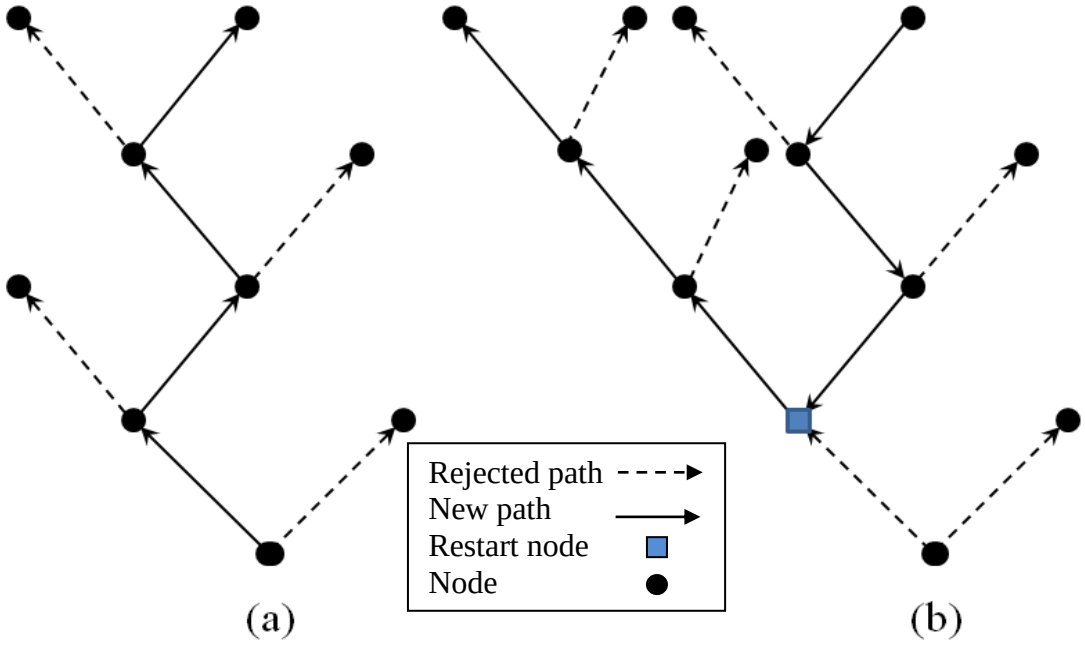


Figure 2.5: Two representations of the monkey behavior. The monkey climbs a tree for the first (a) and the second (b) time. (Mucherino *et al.*, 2009).

The MS procedure is based on a set of parameters, which influences the convergence of the algorithm. The height of the trees is the total number of branches that the monkey can climb from the root to the top. The number of paths the tree contains is represented by the number of times the monkey starts climbing up the same tree. Two other parameters deal with the memory of the Monkey Search heuristic. In order to avoid local minima, a predetermined number of best solutions found on each specific tree are kept in memory. In fact, every time the monkey stops climbing a tree because it has reached the allowed total number of paths, it starts climbing a new tree from a different root solution. The best solutions are kept in memory, so that the monkey can select either one of the best solutions or a combination of them as the root of a new tree. The Monkey Search procedure stops when all the solutions in memory are sufficiently close to one another. A detailed implementation of this algorithm is described in (Mucherino and Seref, 2007).

2.2.5 Firefly Algorithm (FA)

Firefly Algorithm (FA) was developed by Yang (2008) at Cambridge University in 2007. It uses the following three idealised rules:

1) All fireflies are unisex so that a firefly will be attracted to other fireflies regardless of their sex;
2) Attractiveness is proportional to their brightness; thus for any two flashing fireflies, the less bright will move towards the brighter one and they both decrease as their distance increases. If there is no brighter firefly than a particular one, it will move randomly;
3) The brightness of a firefly is affected or determined by the landscape of the objective function. For maximisation problem, the brightness can simply be proportional to the value of the objective function. Other forms of brightness can be defined in a similar way to the fitness function in genetic algorithm.

```
   Algorithm 3   Firefly Algorithm
begin
      Objective function f(x), x=(x1,....,xd)T
      Generate initial population of fireflies xi (i=1,2,...,n)
      Light intensity Ii at xi is determined by f(xi)
      Define light absorption coefficient γ
      while (t<MaxGeneration)
            for j=1: d loop over all d dimensions
            if (Ij > Ii), Move firefly i towards j;
            end if
            Attractiveness varies with distance r via exp[-γ r]
            Evaluate new solutions and update light intensity
            end for j
      end for i
      Rank the fireflies and find the current best
      end while
      Postprocess results and visualisation
end
```

Figure 2.6: Pseudo code of the Firefly Algorithm (FA).

By assuming the attraction of a firefly is determined by its brightness; the variation of light intensity and formulation of the attraction is associated with the encoded objective function.

In the simplest case for maximum optimisation problems, the brightness, *I* of a firefly at a particular location *x* can be chosen as $I(x) \propto f(x)$. However, the attractiveness β is relative; it should be seen in the eyes of the beholder or judged by other fireflies. Thus, it will vary with the distance *rij* between firefly *j*. A detailed implementation of this algorithm is described in (Xin-She Yang, 2008).

2.2.6 Glowworm Swarm Optimisation (GSO)

The Glowworm Swarm Optimisation (GSO) algorithm was developed by Krishnanand and Ghose (2006). Agents in the GSO are regarded as glow worms that carry the quantity called luciferin luminescence with them. Glowworms encode the suitability of their current location, which are evaluated using the objective function, into the luciferin that they broadcast to their neighbors. Glowworms identify their neighbors and calculate the movements to exploit an environment that can be modified, which is bounded above by the various sensors. Using a mechanism of probabilities, each glowworm chooses the neighbor which has a value higher than their own luciferin and moves toward it.

In GSO, a swarm is composed of *N* agents called glowworms. A state of a glowworm, *i* at time, *t* can be described by the following set of variables: a position in the search space $(x^i(t))$, a *luciferin* level $(l^i(t))$and a neighbourhood range $(r^i(t))$. GSO algorithm describes how these variables change over time. Initially, agents are randomly distributed in the search space. Other parameters are initialised

by predefined constants. Each time, the next iteration is composed of three phases: *luciferin* level update, glowworm movement and neighbourhood range update.

To encode the fitness of the current position of a glowworm *i* in the *luciferin* level, the following formula is used:

$$\left(l^i(t)\right) = (1-\rho)l^i(t-1) + \gamma J(l^i(t)) \tag{2.7}$$

Where ρ is the *luciferin* decay constant, γ is the *luciferin* enhancement constant and *J* is an objective function.Then, each glowworm tries to find its neighbours. In GSO, a glowworm *j* is a neighbour of a glowworm *i* only if the distance between glowworms *i* and *j* is shorter than the neighbourhood range $r^i(t)$ and additionally, glowworm *j* has to shine brighter than$i$ ($l^j(t) > l^i(t)$. If one glowworm has multiple neighbours, it chooses one at random with a probability that is proportional to the *luciferin* level of this neighbour. Finally, the glowworm moves one step in the direction of the chosen neighbour. The step size is constant and is equals *s*.

In the last phase, the neighbourhood range $r^i(t)$ is updated in order to limit the range of the communication in an ensemble of agents. The following formula is used:

$$r^i(t+1) = \min\{r_s, \max[0, r^i(t) + \beta(n_d - |n^i(t)|)]\} \tag{2.8}$$

where: r_s is a sensor range (a constant, which limits the size of the neighbourhood range), n_d is a desired number of neighbours, $|n^i(t)|$ is a number of neighbours of a glowworm *i* at time t, and β is a model constant.

The whole GSO group algorithm is shown in the form of flowchart in Figure 2.7. It is understood that the core of the algorithm is relatively simple and consists of boxes marked with thick lines.

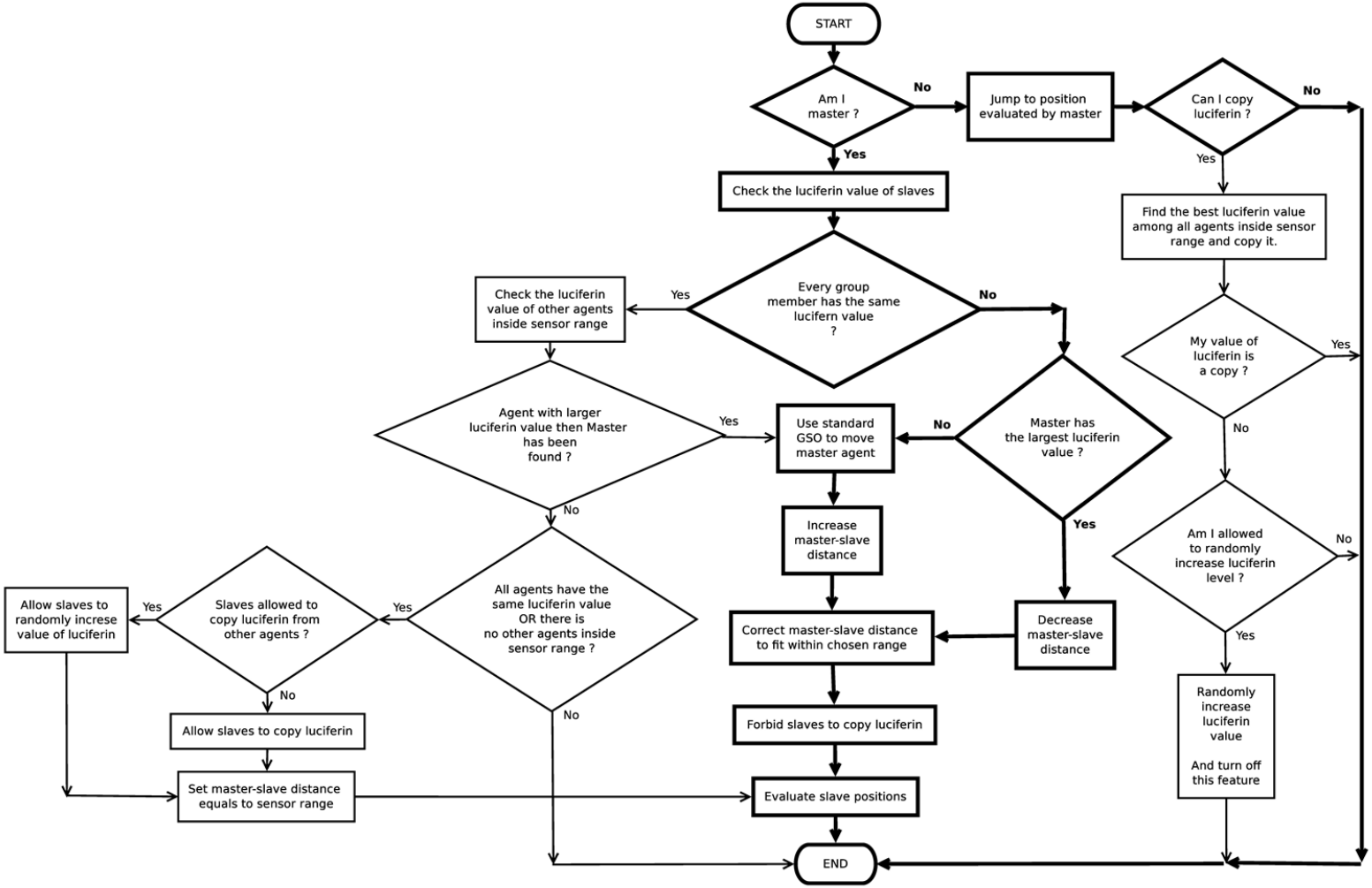


Figure 2.7: Flowchart of single iteration of GSO group algorithm (Krishnanand and Ghose, 2006).

2.2.7 Cuckoo Search (CS)

Cuckoo is one of the birds' species, which exhibit a unique behavior for survival. They have also known to be a solitary bird and engage in severe brood parasitism; by laying their eggs in the nests of other host birds (Payne, 2005). Cuckoo Search algorithm was proposed by Yang and Deb (2009), using the following three idealised rules:

1) Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;
2) The best nests with high quality of eggs (solutions) will carry over to the next generations;
3) The number of available host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

For simplicity, this last assumption can be approximated by a fraction p_a of the$n$ nests being replaced by new nests (with new random solutions at new locations). For maximisation problems, the quality or fitness of a solution can simply be proportional to the objective function. Other forms of fitness can be defined in a similar way to the fitness function in genetic algorithms.

Based on these three rules, the basic steps of the Cuckoo Search (CS) can be summarised as the pseudocode shown in Figure 2.8.

When generating new solutions $x^{(t+1)}$ for, say cuckoo i, a Levy flight is performed

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus Levy(\lambda) \tag{2.9}$$

where $\alpha > 0$ is the step size which should be related to the scales of the problem of interest. In most cases, use $\alpha = O(1)$ is used. The product $\oplus$ means entry-