Available online at www.sciencedirect.com
ScienceDirect
Journal of Bionic Engineering 7 Suppl. (2010) S232–S237

A Review of Nature-Inspired Algorithms

Hongnian Zang, Shujun Zhang, Kevin Hapeshi

Department of Computing, the University of Gloucestershire, Cheltenham, GL50 2RH, UK

Abstract

The study of bionics bridges the functions, biological structures and organizational principles found in nature with our modern technologies, and numerous mathematical and metaheuristic algorithms have been developed along with the knowledge transferring process from the lifeforms to the human technologies. Output of bionics study includes not only physical products, but also various computation methods that can be applied in different areas. People have learnt from biological systems and structures to design and develop a number of different kinds of optimisation algorithms that have been widely used in both theoretical study and practical applications. In this paper, a number of selected nature-inspired algorithms are systematically reviewed and analyzed. Though the paper is mainly focused on the original principle behind each of the algorithm, their applications are also discussed.

Keywords: bionic optimization algorithms review, Ant Colony Optimization, Bees Algorithm, Genetic Algorithm, Firefly Algorithm

doi: 10.1016/S1672-6529(09)60240-7

1 Introduction

In the last few decades, more and more researches suggest that nature is a great source for inspirations to both develop intelligent systems and provide solutions to complicated problems. Taking animals for example, evolutionary pressure forces them to develop highly optimized organs and skills to take advantages of fighting for food, territories and mates. Some of the organs and skills can be well refined as optimization algorithms, and the evolution is a process to fine-tune the parameter settings in the algorithms. In this paper, four nature-inspired algorithms are systematically reviewed, including Ant Colony Optimization (ACO), Bees Algorithm (BA), Genetic Algorithm (GA), Firefly Algorithm (FA).

2 Ant Colony Optimization

2.1 Introduction

The ACO is a metaheuristic approach to solve problems that has been inspired by the ants' social behaviours in finding shortest paths. Real ants walk randomly until they find food and return to their nest while depositing pheromone on the ground in order to mark their preferred path to attract other ants to follow[1]. If other ants travel along the path and find food, they will deposit more pheromone as to reinforce the path for more ants to follow[2].

In the past decades substantial amount of research has been done to both develop the ACO algorithm itself and practical applications to solve the relevant problems in the real world. The initial ACO algorithm, Ant System (AS), was proposed by Marco Dorigo in 1992 in his PhD thesis[3]. Dorigo and Gambardella introduced Ant Colony System (ACS) as a variant of AS in 1997[4]. In parallel, Stützle and Hoos invented the MAX-MIN Ant System in 1996[5]. In the early twenties, Iredi *et al*. published the first multi-objective algorithm which is a very popular extension to the original ACO algorithm[6]. Table 1 lists the main research work in ACO.

Table 1 A short list of well known ACO algorithms in chronology order

Year	Algorithm	Authors
1992	Ant System (AS)	Dorigo
1996	MAX-MIN Ant System	Stützle and Hoos
1997	Ant Colony System (ACS)	Dorigo and Gambardella
2001	Multi-Objective	Iredi *et al*.

Corresponding author: Shujun Zhang
E-mail: szhang@glos.ac.uk

2.2 Ant System (AS)

Since the pheromone evaporates over time, the longer iteration from node i to node j on a path, the faster the pheromone reduces its density. In the algorithm, the desirability of edge (i, j) is inversely proportional to its length $(\frac{Q}{d_{ij}})$.

The ants always join the path with higher pheromone, and then they provide feedback as to deposit their own pheromone to lead more ants to the shorter path[7]. The core concept of AS[3] is that the amount of pheromone (τ_{ij}) on each edge is updated by all the n ants. After all ants have finished a solution independently, pheromone on each trail is updated as described in Eq. (1):

$$\tau_{ij} = (1-\rho)^{\tau_{ij}} + \sum_{k=1}^{n} \Delta\tau_{ij}^{k}, \tag{1}$$

where ρ is the rate of the pheromone evaporation, n is the number of ants, τ_{ij} is the amount of pheromone on edge (i, j), and $\Delta\tau_{ij}^{k}$ is the amount of pheromone deposited by ant k, given by Eq. (2):

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{if ant} k \text{ travels on edge } (i,j) \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where L_k is the length of ant k's journey, and Q is a constant parameter.

The move probability from node i to node j (p_{ij}^{k}) will be 0 if node i is infeasible for ant k staring from node i (M_k), otherwise they are calculated by Eq. (3), where α is a parameter to control the influence of τ_{ij} and β is a parameter to control the influence of η_{ij} ($0 < a, b < 1$),

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\Sigma_{(l\psi)\not\subset M_k}\tau_l\psi^{\alpha}\eta_l\psi^{\beta}} & \text{if } (l\psi)\notin M_k \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

2.3 Ant Colony System (ACS)

The ACS is an extension of AS. It is mainly aimed to improve the performance of the AS algorithm and to reduce its complexity. Dorigo and Gambardella proposed ACS in 1997, and ACS works as described below[4]:

(1) m ants initially wonder among n nodes randomly.

(2) Each ant constructs a tour by repeatedly applying the state transition rule, and the ant also updates the amount of pheromone on the visited edges by applying the local updating rule.

(3) The amount of pheromone on each edge is updated again upon all ants have finished their tours (by applying the global updating rule).

2.3.1 The state transition rule

At any given time, an ant moves from one node to another while constructing a complete tour, and the state transition rule determines which is the next node to move to. In another word, the ant visits node s next by applying Eq. (4). The state provides a direct way to balance between exploration of new edges and exploitation of a priori and accumulated knowledge.

$$s = \begin{cases} \arg\max(ij) \notin M_k\{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}\} & \text{if } q \le q_0 \\ S & \text{otherwise} \end{cases}, \tag{4}$$

where q_0 is the probability of the best state being chosen, and it is a parameter $0 \le q_0 \le 1$ and usually is set to 0.9; τ_{ij}, η_{ij} as discussed in Section **2.2**, usually τ_{ij} is set to 1 and η_{ij} is set to 2.

2.3.2 Global updating rule

In AS all the ants deposit pheromone while in ACS only the globally best ant is allowed to lay down pheromone, and the best ant is the one that provides the best solutions as its trail is the shortest. The amount of pheromone is updated by applying Eq. (5):

$$\tau_{ij} = (1-\alpha)\tau_{ij} + \alpha\Delta\tau_{ij}. \tag{5}$$

If $i, j \in N_{\text{global-best-tour}}$, $\Delta\tau_{ij} = L_{gb}^{-1}$, otherwise, $\Delta\tau_{ij} = 0$. Also, α is the pheromone decay parameter that represents the trail evaporation, and it is usually set to 0.1.

2.3.3 Local updating rule

The pheromone associated to the edge is updated when an ant moves from one end to the other by applying the local updating rule of Eq. (6):

$$\tau_{ij}(k) = \rho\tau_{ij}(k-1) + (1-\rho)\tau_0, \tag{6}$$

where $\tau_0 = nL_{nn}^{-1}$, and n is the number of nodes and L_{nn} is the tour length produced by the nearest neighbour heuristic.

2.4 Applications

The ACO was initially introduced as a cooperative learning approach to the Travelling Salesman Problem (TSP), and it now has been applied to a wide range of combinatorial optimization problems, including scheduling (e.g. Job-shop scheduling problem[8], Single machine total tardiness problem[9]), vehicle routing (e.g. Capacitated vehicle routing problem[10], Stochastic vehicle routing problem[11]) and assignment problems (e.g. Frequency assignment problem[12]). This algorithm can be applied in the bionic management study such as the optimisation of the distribution and location of the new town during the integration of rural with city.

3 Bees Algorithm (BA)

The BA has been inspired from the food foraging behaviour of honey bees whose colony can typically extend itself over 10 km to 14 km and in multiple directions simultaneously to exploit a large number of food sources[13]. The colony tends to attain the optimal use of its members. Theoretically, the richer (more nectar or pollen) and closer a food source is, the more bees that would be sent to it[14]. The food foraging process begins with scout bees' searching activities. They fly randomly to explore all the food sources in all directions. When they return to the hive, they report the searching results to the others by performing waggle dance[15] which communicates these important pieces of information about the food source, including its direction, distance and quality rating. The waggle dance is essential in the food source evaluation and it helps the colony to deploy flower bees to the food sources precisely.

The BA is a population based optimisation algorithm first developed in 2005. The original pseudo code for the algorithm is defined as[16]:

(1) Initialise population with random solutions.

(2) Evaluate fitness of the population.

(3) While (stopping criterion not met)//Forming new population.

(4) Select sites for neighbourhood search.

(5) Recruit bees for selected sites (more bees for best e 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.

Scout bees are initially positioned randomly in the search area, and they move randomly and continuously from one site to another to exam the fitness of the sites and then report to the colony for other colony members to accurately evaluate the quality of the food sources and the effect needed to collect the food. As step 4 suggests, the bees having the highest fitness are chosen as "selected bees" and the sites visited by them are chosen for neighbourhood search. In steps 5 and 6, the algorithm deploys more bees to accompany the "selected bees" in order to perform detailed search around the neighbourhood of their visited sites. Difference scouting bees recruiting strategy is also an important operation of the BA. In step 7, the rest of the bees are sent out randomly to search for new potential solutions in order to balance the number of new sites against the discovered ones.

BA have been applied to many combinatorial optimization problems, ranging from manufacturing[17] to computer image analysis[18], and it is proved to have a very robust performance and a high success rate compared with other intelligent optimisation methods. BA can also be applied in the research of face detection 3D modelling of bio-robot design and development.

4 Genetic Algorithm (GA)

GA generates solutions to search, optimization and machine learning problems via applying techniques inspired by biological evolution. Genetic algorithms were initially developed in computer simulations in the early 1950s[19]. Later on, computer simulation of evolution became very popular, and Fraser and Burnell published a book to systematically describe the simulations which contains all the basic components of modern generic algorithms[20]. GA adopts some genetic terminologies, including:

(1) Chromosome is an encoding of a solution to an optimization problem. The solutions are typically represented in binary[21].

(2) Selection is a stage in the GA where individual genomes are chosen for breeding new generation.

(3) Crossover and Mutation both are genetic operators applied to a pair of parents when they reproduce to alter their genetic composition[22].

To perform a GA, the first step is to initialize the population that normally is composed of randomly generated individuals covering the entire range of possible solutions, and size of the population is determined by the nature of the problem itself. The next step is to evaluate the fitness of each member of the population. A fitness function is employed in this stage to provide fitness values for each individual, and the results are then normalized in order to sort the entire population by descending fitness values. Once the selection process has finished, it comes to the reproduction stage where this generation performs repeatedly until one of the termination criteria is met. Fig. 1 shows the interconnection of these stages.

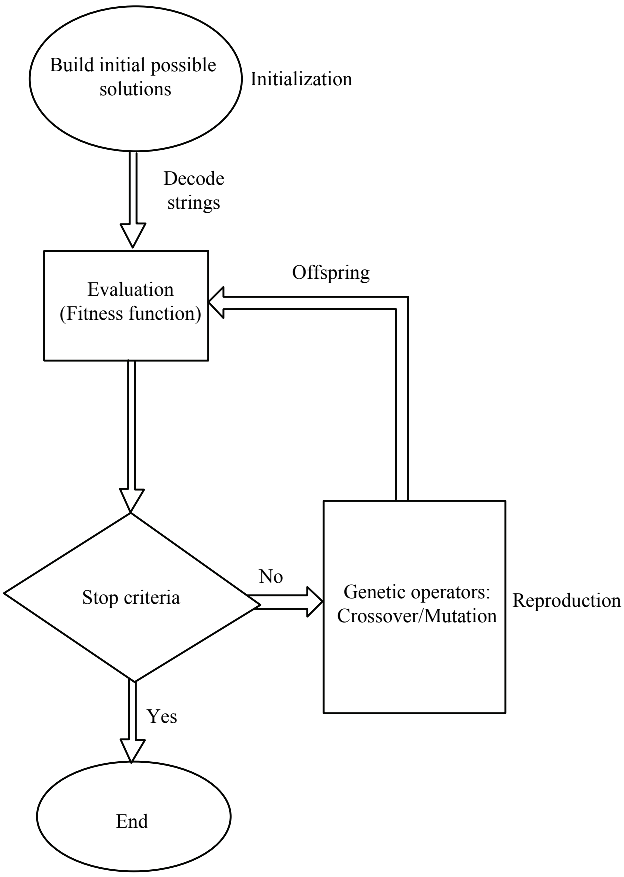

Fig. 1 GA stages.

GA has been widely adopted to solve complex problems, especially in the areas of scheduling, global optimization and control engineering. Table 2 lists some of the most successful applications.

The implementation of genetic algorithms is one of the easiest compared to alternative intelligent optimization algorithms, but the building block hypothesis has been criticized that it lacks of theoretical justification[31,32]. Moreover, in some cases, the termination criterion is not clear. If the fitness function only has 0/1 as results, GA normally cannot solve the problem.

Table 2 A short list of well known GA applications in alphabetical order

Applicaiton	Authors	Year
Bioinformatics multiple sequence alignment[23]	Gondro and Kinghorn	2007
Building phylogenetic trees[24]	Hill *et al.*	2005
Computer-automated design[25]	Li *et al.*	2004
Control engineering[26]	Ng	1995
Gene expression profiling analysis[27]	To and Vohradsky	2007
Multimodal optimization[28]	Wong *et al.*	2010
Power electronics design[29]	Zhang *et al.*	2006
Protein folding and protein/ligand docking[30]	Willett	1995

5 Firefly Algorithm (FA)

Fireflies produce luminescent flashes as a signal system to communicate with other fireflies, especially to prey attractions[33]. FA is inspired by the firefly's biochemical and social aspects and based upon the following three assumptions: (a) Each firefly attracts all the other fireflies with weaker flashes; (b) Attractiveness is proportional to their brightness which is reverse proportional to their distances; (c) No fireflies can attract the brightest firefly, and it moves randomly[34].

The brightness of firefly is associated with the objective function $f(x)$, $x\in M_d$, assume that there exists n fireflies x_i, $i = 1,2,\ldots,n$ initially positioned randomly in the space. Given the distance between firefly i and firefly j as $r_{ij} = d(x_i, x_j)$, the attractiveness between them should follow monotonically decreasing function, e.g. $= ke^{-r_{ij}}$, where k is the max attractiveness value. The flash intensity I is associated with the objective function $f(x)$, i.e. $I = \alpha f(x)$. Only firefly with higher flash intensity attracts the other one i.e. $I_i<I_j, j = 1, 2,\ldots, n, j\neq i$.

The algorithm can be summarized as the following pseudo code[35].

```
While (t<Max Generation)
 for i = 1:n (all n fireflies);
   for j = 1:n (n fireflies)
     if (Ij>Ii)
       move firefly i towards j;
     end if
   Vary attractiveness with distance rij via = 0e^(-rij);
   Evaluate new solutions and update light intensity;
```

end for j
end for i
Rank fireflies and find the current best;
end while

FA can be applied in the optimisation of traffic system design and development of a new town.

6 Conclusion

In conclusion, four nature-inspired algorithms are reviewed in this paper: Ant Colony Optimization (ACO), Bees Algorithm (BA), Genetic Algorithm (GA) and Firefly Algorithm (FA). ACO and BA have been inspired by the social behaviour within ants' and bees' food foraging process respectively while FA simulates the attractions system of the fireflies. GA, on the other hand, has inherited the principles of Darwin's Evolutionary Theory. Undoubtedly, a substantial amount of genius solutions and algorithms are gestated in the nature, and we just need to dig them out, and then employ them to solve our problems. There are several important stages involved in developing efficient algorithms, including: (a) observe and summarize the behaviour of the creatures in nature, (b) establish a raw model to represent the behaviour, (c) convert into mathematical module (make assumptions, setup initial parameters), (d) develop the pseudo code to simulate the behaviour, (e) test the algorithm both theoretically and experimentally, and refine the parameter settings to achieve better performance of the algorithm. Meanwhile, apparently behaviour is usually the actions in relation to its environment, so it should team up with other behaviours, event behaviours of others, in order to achieve better results. Therefore, the nature-inspired algorithms could hybridize together with other algorithms to enhance itself to be faster, more efficient, and more robust.

References

[1] Jackson D E, Ratnieks F L W. Communication in ants. *Current Biology*, 2006, **16**, R570–R574.

[2] Goss S, Aron S, Deneubourg J L, Pasteels J M. Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 1989, **76**, 579–581.

[3] Dorigo M. Optimization, learning and natural algorithms. Ph.D.Thesis, Politecnico di Milano, Italy, 1992. (In Italian)

[4] Dorigo M, Gambardella L M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1997, **1**, 53–66.

[5] Stützle T, Hoos H H. Max-Min ant system, future generation computer systems. *Future Generation Computer Systems*, 2000, **16**, 889–914,

[6] Iredi S, Merkle D, Middendorf M. Bi-criterion optimization with multi colony ant algorithms, evolutionary multi-criterion optimization. *First International Conference EMO* 2001, Zurich, Switzerland, 2001, 359–372.

[7] Beckers R, Deneubourg J L, Goss S. Trails and U-turns in the selection of the shortest path by the ant Lasius Niger. *Journal of Theoretical Biology*, 1992, **159**, 397–415.

[8] Martens D, Backer M De, Haesen R, Vanthienen J, Snoeck M, Baesens B. Classification with ant colony optimization. *IEEE Transactions on Evolutionary Computation*, 2007, **11**, 651–665.

[9] Baucer A, Bullnheimer B, Hartl R F, Strauss C. Minimizing total tardiness on a single machine using ant colony optimization. *Central European Journal for Operations Research*, 2000, **8**, 125–141,

[10] Toth P, Vigo D. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 2002, **123**, 487–512.

[11] Secomandi N. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 2000, 27, 1201–1225.

[12] Aardal K I, van Hoesel S P M, Koster A M C A, Mannino C, Sassano A. Models and solution techniques for the frequency assignment problem. *A Quarterly Journal of Operations Research*, 2001, **1**, 261–317.

[13] Pham D T, Ghanbarzadeh A, Koc E, Otri S, Rahim S, Zaidi M. *The Bees Algorithm*, Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005.

[14] Seeley T D. *The Wisdom of the Hive*: *The Social Physiology of Honey Bee Colonies*, Harvard University Press, Cambridge, 1996.

[15] Frisch K V. *Bees*: *Their Vision*, *Chemical Senses and Language*, revised edition, Cornell University Press, Ithaca, New York, 1976.

[16] Pham D T, Ghanbarzadeh A, Koç E, Otri S, Rahim S, Zaidi M. The bees algorithm – a novel tool for complex optimisation problems. *Proceedings of* 2*nd International Virtual Conference on Innovative Production Machines and Systems*, 2006, 454–461.

[17] Pham D T, Ghanbarzadeh A, Koç E, Otri S. Application of the bees algorithm to the training of radial basis function networks for control chart pattern recognition. *Proceeding of*

5*th CIRP International Seminar on Intelligent Computation in Manufacturing Engineering* (CIRP ICME'06), Ischia, Italy, 2006, 454–461.

[18] Olague G, Puente C. Honeybees as an intelligent based approach for 3D reconstruction. The 18*th International Conference on Pattern Recognition*, 2006, **1**, 1116–1119.

[19] Barricelli N A. Symbiogenetic evolution processes realized by artificial methods. *Methods*, 1957, **9**, 143–182.

[20] Fraser Alex, Burnell D. *Computer Models in Genetics*, McGraw-Hill, New York, 1970.

[21] Cha S H, Tappert Charles C. A genetic algorithm for constructing compact binary decision trees. *Journal of Pattern Recognition Research*, 2009, **4**, 1–13.

[22] Ting C K. On the mean convergence time of multi-parent genetic algorithms without selection. *Proceedings of the Eighth European Conference on Artificial Life*, Canterbury, UK, 2005, 403–412.

[23] Gondro C, Kinghorn B P. A simple genetic algorithm for multiple sequence alignment. *Genetics and Molecular Research*, 2007, **6**, 964–982.

[24] Hill T, Lundgren A, Fredriksson R, Schiöth H B. Genetic algorithm for large-scale maximum parsimony phylogenetic analysis of proteins. *Biochimica Biophysica Acta*, 2005, **1725**, 19–29.

[25] Li Y, Ang K H, Chong G C Y, Feng W, Tan K C, Kashiwagi H. CAutoCSD – Evolutionary search and optimisation enabled computer automated control system design. *International Journal of Automation and Computing*, 2004, **1**, 76–88.

[26] Ng K C. *Switching Control Systems and Their Design Automation via Genetic Algorithms*, PhD Thesis, University of Glasgow, 1995.

[27] To C C, Vohradsky J. A parallel genetic algorithm for single class pattern classification and its application for gene expression profiling in Streptomyces coelicolor. *BMC Genomics*, 2007, 8, 6–9.

[28] Wong K C, Leung K S, Wong M H. Effect of spatial locality on an evolutionary algorithm for multimodal optimization. *Applications of Evolutionary Computation*, 2010, **6024**, 481–490.

[29] Zhang J, Chung H S H, Lo W L. Pseudocoevolutionary genetic algorithms for power electronic circuits optimization. *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, *Part C*: *Applications and Reviews*, 2006, **36**, 590–598.

[30] Willett P. Genetic algorithms in molecular recognition and design. *Trends in Biotechnology*, 1995, **13**, 516–521.

[31] Goldberg David E. *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley Professional, 1st edition, 1989.

[32] Vose M D, Wright A H, Rowe J E. Implicit parallelism. *Proceedings of the Genetic and Evolutionary Computation Conference*, 2003, 7–11.

[33] Babu B G, Kannan M. Lightning bugs. *Resonance*, 2002, **7**, 49–55.

[34] Yang X S. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, Beckington, UK, 2008.

[35] Yang X S. Firefly algorithms for multimodal optimization. *Stochastic Algorithms*: *Foundations and Applications*, 2009, **5792**, 169–178.