Solving Optimization of a Function Problem Using Bees Algorithm

Lecturer. Aseel G. Mahmoud
College of Nursing-University of Baghdad

Abstract

Bees Algorithm (BA) which is an optimization algorithm that mimics the food foraging behavior of swarms of honey bees have proven to be very powerful computational techniques due to their search capabilities.

This paper attempts to utilize and investigate the use of BA in optimization of functions problem. The aim of this paper was to use the BA to solve some kinds of two variables nonlinear function which it submits to optimization of a function filed

We investigate a comparison study between BA and GA to this kind of problems. The experimental results reported will shed more light into which algorithm is best in solving optimization problems.

1. Introduction

Genetic algorithms (GAs) are a part of evolutionary computing, which is a rapidly growing area of artificial intelligence. GA's are inspired by Darwin's theory about evolution. Simply said, solution to a problem solved by genetic algorithms is evolved.

GAs were first suggested by John Holland and developed by him and his students and colleagues in the seventies. This lead to Holland's book "Adoption in Natural and Artificial Systems" published in 1975 [1]. Over the last twenty years, it has been used to solve a wide range of search, optimization and machine learning problems. Thus, the GA is an iteration procedure, which maintains a constant size population of candidate solution [1]. In 1992 John Koza has used GA to evolve programs to perform certain tasks. He called his method "**genetic programming**" (GP) [2].

In 1995, Kennedy J. and Eberhart R. [3], introduces a concept for the optimization of nonlinear functions using particle swarm methodology. The evolution of several paradigms outlined, and an implementation of one of the paradigms had been discussed.

Swarm Intelligence (SI) is an Artificial Intelligence (AI) technique that focuses on studying the collective behavior of a decentralized system made up by a population of simple agents interacting locally with each other and with the environment [4].

The main goal of this paper is to explore and illustrate the abilities of swarm intelligence especially the usage of the Bees algorithm for developing intelligent optimization tool colonies for the purpose of providing a "good" solution of optimization of a function problem.

Also, work presented in this work studies and utilizes the use of Bees algorithm to solve this problem within acceptable amount of time with a faster convergence and time reduction and compares it with the conventional algorithms.

A comparison study has been made to compare the results obtained from applying GA and BA to find an optimal solution for optimization of a function problem to explore the best algorithm to apply in such problems.

2. Genetic Algorithm

Genetic Algorithms (GA's) are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old; an occasional new part is tried for good measure. While randomized, GA's are no simple random walk. They efficiently exploit historical information to speculate on new search point with expected improved performance [5].

3. Optimization of a Function Problem

There is a large class of interesting problems for which no reasonably fast algorithms have been developed. Given such a hard optimization problem it is often possible to find an efficient algorithm whose solution is approximately optimal. We discuss the basic features of a GA for optimization of a simple function.

$$f(x_1,x_2)= 21.5 + x_1 \,.\, \sin(4\pi x_1) + x_2 \,.\, \sin(20\pi x_2) \qquad \ldots(1)$$

where $-3.0 \le x_1 \le 12.1$ and $4.1 \le x_2 \le 5.8$.

Since x_1, x_2 are real numbers, this implies that the search space can be huge and that traditional methods can fail to find the optimal solution [6].

4. Implementation of GA in Optimization of a Function Problem

Problem Representation

To apply the GA for maximizing $f(x_1,x_2)$, a genetic representation of solution to the problem must be appropriately chosen first. The Simple GA uses the binary representation in which each point is described by a chromosome vector coded as a binary string. We use a binary vector as a chromosome to represent real values of the variable x, the length of the vector depends on the required precision, which in this example, is six places after the decimal point.

The domain of the variable x_1 has length 15.1; the precision requirement implies that the range [-3.0,12.1] should divided into at least 15.1×10000 equal size ranges. This means that 18 bits are required as the first part of the chromosome:

$2^{17} < 151000 < 2^{18}$

The domain of the variable x_2 has length 1.7; the precision requirement implies that the range [4.1,5.8] should divided into at least 1.7×10000 equal size ranges. This means that 15 bits are required as the second part of the chromosome:

$2^{14} < 17000 < 2^{15}$

The total length of a chromosome (solution vector) is then m=18+15=33 bits; the first 18 bits code x_1 and remaining 15 bits (19–33) code x_2 [7].

Let us consider an example chromosome: (010001001011010000111110010100010) corresponds to (x_1,x_2) = (-2.334465,4.699438). The fitness value for this chromosome is:

$f(-2.334465,4.699438) = 26.566770.$

Initial Population and Evaluation Function

To optimize the function f using GA, we create a population of pop_size chromosomes. All 33 bits in all chromosomes are initialized randomly.

Evaluation function for binary vector v is equivalent to, the function f:

$$eval(v) = f(x_1,x_2) \qquad \ldots(2)$$

where the chromosome v represents the real value x.

During the evaluation phase we decode each chromosome and calculate the fitness s function from(x_1,x_2) values just decode.

For example, three chromosomes:

v_1 = (100110100000001111111010011011111),

v_2 = (111000100100110111001010100011010),

correspond to values x_1 and x_2 respectively. Consequently, the evaluation function would rate them as follows:

$eval(v_1) = f(11.161431,4.954643) = 34.237697$

$eval(v_2) = f(10.953948,7.767766) = 33.832967$

Clearly, the chromosome v_1 is the best of the three chromosomes, since its evaluation returns the highest value.

Genetic Operators

1. **Selection Operator**

Roulette wheel is chosen to sums up the fitness' of all individuals and calculates each individual percentage of the total fitness. The percentage of the total fitness is then used as the probabilities to select N individuals from the set population and copy them into the set selected-parents.

2. **The Mating Crossover Operator**

Individuals from the set selected-parents are mated to generate offsprings for the next generation. The two parents generate two offsprings using a crossover operation. For this example, to illustrate the crossover operator on chromosome with a crossover with probability Pc, we generate a random integer number pos from the range 1..32. The number *pos* indicate the position of the crossing point. The pair of chromosomes is:

v_1 = (10011010000000111111110100110111111),

v_2 = (00001000011001000000101011101101),

and the generated number *pos*=9. These chromosomes are cut after the 9th bit and replaced by a pair of the offspring:

the two resulting offspring are:

v_1'= (10011010001100100000101011101101),

v_2'= (000010000000001111111101001101111111),

3. **Mutation Operator**

Mutation is a random change of one or more genes (positions in a chromosome) with a probability equal to the mutation rate P_m a gene is changed/swapped, i.e 0→1 and 1→0. The probability for a mutation is usually kept small. Assume that the fifth gene from the v_2 chromosome was selected for a mutation. Since the fifth gene in this chromosome is 1, it would be flipped into 0. So the chromosome v_3 after this mutation would be:

v_2'= (000000000000001111111101001101111111).

4. **Genetic Parameters**

For this particular problem, Michalewicz [7] used the following parameters: population size pop_size=20, probability of crossover Pc=0.25, probability of mutation Pm =0.01.

Experimental Results

In table (1) we provide the generation number for which we noted improvement in the evaluation function, together with the value of the function.

Table (1) Results of 1000 generations for optimization of a function problem.

Chromosome	Gen. No.	Evolution Function	Variables x_j	
			x_1	x_2
010001001011010000010001001011010	0	26.566769	2.334464	4.699438
011100101001111110011100101001111	1	28.068086	4.451298	5.711127
110101010111100111101010101111100	3	33.236484	10.664784	4.507321
111101100001110111111101100001110	7	34.012620	11.044728	4.849531
000110011100101111000110011100101	10	34.755134	11.445236	5.210263
010100100110101111010100100110101	13	34.823983	11.484981	5.246061
111110111110101111111110111110101	17	34.863961	11.508310	5.267073
110000101111011111110000101111011	20	35.415673	11.853232	5.577739
000110101111011111000110101111011	23	35.417473	11.854442	5.578829
011111001101111111011111001101111	24	35.667943	12.029898	5.736860
000010111011111111000010111011111	29	35.719240	12.067800	5.770998
011101111011111111011101111011111	35	35.721555	12.069528	5.772554
101101010111111111101101010111111	54	35.736241	12.080530	5.782464
001111011111111111001111011111111	100	35.756931	12.096140	5.796523
110000111111111111110000111111111	296	35.757463	12.096543	5.796887
001100111111111111001100111111111	530	35.758147	12.097062	5.797354
101100111111111111101100111111111	637	35.758223	12.097119	5.797405
101110111111111111101110111111111	638	35.759438	12.098041	5.798236
011001111111111111011001111111111	642	35.760122	12.098559	5.798702
010011111111111111010011111111111	660	35.761033	12.099251	5.799325

For this problem, a simulation has been constructed in order to apply the GA, using the crossover parameters mentioned above, the following results are being obtained: v_{max} = (010011111111111111010011111111111)
Which corresponds to a value (x_1,x_2)=(12.099251,5.799325), and $f(v_{max})$=35.761003.

5. Artificial Swarm Intelligence

The artificial intelligence is used to explore distributed problem solving without having a centralized control structure. This is seen to be a better alternative to centralized, rigid and preprogrammed control. Real life swarm intelligence can be observed in ant colonies, beehives, bird flocks and animal herds [8].

The most common examples of swarm intelligence systems: **Ant Colony Optimization** (ACO), **Particle Swarm Optimization** (PSO) and **Marriage in Honey Bees Optimization** (MBO).

The most important applications of artificial swarm intelligence:

1. Swarm intelligence has applications in decentralized controls of unmanned vehicles for the military so single operators can control more unmanned vehicles [8].
2. The use of swarm intelligence in medical nanobots may also help combat cancer [8].
3. The European Space Agency is thinking about an orbital swarm for self-assembly and interferometer [9].
4. NASA is investigating the use of swarm technology for planetary mapping [9].
5. Swarm intelligence was used in the creation of the video sequence in the movie "Battle of Helm's Deep", "Lord of the Rings".
6. Swarm intelligence can also be used in communication networks optimization [10].

6. Bees Algorithm

MBO is a new development which is based on the haploid-diploid genetic breeding of honeybees and is used for a special group of propositional satisfiability problems. The main processes in MBO are: the mating flight of the queen bee with drones, the creation of new broods by the queen bee, the improvement of the broods' fitness by workers, the adaptation of the workers' fitness, and the replacement of the least fit queen with the fittest brood [11].

The challenge is to adapt the self-organization behavior of the colony for solving the problems. The **Bees Algorithm (BA)** is an optimization algorithm inspired by the natural foraging behavior of honey bees to find the optimal solution. Figure (1) shows the pseudo code for the algorithm in its simplest form[12].

The algorithm requires a number of parameters to be set, namely:

a. Number of scout bees (n).
b. Number of sites selected out of n visited sites (m).
c. Number of best sites out of m selected sites (e).
d. Number of bees recruited for best e sites (nep).
e. Number of bees recruited for the other (m-e) selected sites (nsp).
f. Initial size of patches (ngh) which includes site and its neighborhood and stopping criterion.

Bees Algorithm

Input: Number of (scout bees (*n*), sites selected out of n visited sites (m).
best sites out of m selected sites (e).
bees recruited for best e sites (nep).
bees recruited for the other (m-e) selected sites (nsp). Initial size of patches (ngh) which includes site and its neighborhood and stopping criterion. Maximum of iterations).

Output: Optimal solutions.

step1. Initialize population with random solutions.
step2. Evaluate fitness of the population.
step3. Repeat.
step4. Select sites for neighborhood search.
step5. Recruit bees for selected sites (more bees for best e sites) and evaluate fitness's.
step6. Select the fittest bee from each patch.
step7. Assign remaining bees to search randomly and evaluate their fitness's.
step8. Until stopping criterion is met.

Figure (1) Pseudo code of the basic bee's algorithm

The algorithm starts with the "n" scout bees being placed randomly in the search space. The fitness's of the sites visited by the scout bees are evaluated in step 2. In step 4, bees that have the highest fitness's are chosen as "selected bees" and sites visited by them are chosen for neighborhood search. Then, in steps 5 and 6, the algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best "e" sites. The bees can be chosen directly according to the fitness's associated with the sites they are visiting. Alternatively, the fitness values are used to determine the probability of the bees being selected.

Searches in the neighborhood of the best "e" sites which represent more promising solutions are made more detailed by recruiting more bees to follow them than the other selected bees. Together with scouting, this differential recruitment is a key operation of the Bees Algorithm. However, in step 6, for each patch, only the bee with the highest fitness will be selected to form the next bee population. In nature, there is no such a restriction. This restriction is introduced here to reduce the number of points to be explored.

In step 7, 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. At the end of search iteration, the colony will have two parts to its new population representatives from each selected patch and other scout bees assigned to conduct random searches [13].

The advantages of Bees algorithm [14]:

- ❖ Bee's algorithm is more efficient when finding and collecting food that is it takes less number of steps.
- ❖ Bee's algorithm is more scalable, it requires less computation time to complete the task.

7. Applications of the Bees Algorithms

The Bee Algorithm, which is inspired by the food foraging behavior of honey bees, has found many applications in engineering field such as [15]:

1. Training neural network for pattern recognition.
2. Forming manufacturing cells.
3. Scheduling jobs for a production machine.
4. Finding multiple feasible solutions to preliminary design problem.
5. Data clustering.
6. Optimizing the design of mechanical components.
7. Multi-Objective optimization.
8. Tuning a fuzzy logic controller for a robot gymnast.
9. Computer vision and image analysis.

8. Implementation of BA

In this paper we will try to apply BA algorithm in such as optimization of a function. This problem was chosen according to different factors such as representation of the problem (which has a great influence on BA algorithm) can be applied more efficiently. Furthermore, this problem chosen since they own a high complexity (the size and the shape of the search space), which its, cannot be solved using traditional known searches, like exhaustive search method.

Problem Representation

To apply the BA for maximizing $f(x)$, a genetic representation of solution to the problem must be appropriately chosen first. BA can uses no more than (2) real variable numbers these values are x_1 and x_2 will considered to be the component of the mentioned solution vectors.

Initial Population

To optimize the function f using BA, we create a population of pop_size=10 or 20 bees each represented with (2) random real values $x_1 \in [-3.0, 12.1]$ and $x_2 \in [4.1, 5.8]$.

Experimental Results

For this particular problem, we use population size pop_size=10, m = 5, e = 2, nep = 7 and nsp = 5. It's important to mentioned that the accuracy of any real number taken within (6) digits.

For this problem, a simulation has been constructed in order to apply BA, using the parameters mentioned above; the optimal solution to maximize f (the limit point of generated sequence of x_i) when:

$f(v_{max}) = 35.762019651$ at $(x_1, x_2) = (12.1, 5.8)$.

When execute the simulation the result of the generation number, evolution function, the variable values and the absolute error are been showed, for example the following results are being obtained after 20 generation:

v = (0110111111111111110110111111111)

which corresponds to a the variable values (x_1,x_2) = (11.990843,5.292858), and $f(v)$ = 35.2658236708275794,with absolute error E=0.2631456342459779,where:

E=|exact – approximation|.

Table (2) shows the results of applying BA of optimization of function for 600 generations.

Table (2) the results of applying BA of optimization of function.

Gen. No.	**Evolution Function**	**Variables x_j**		**Absolute Error**
		x_1	x_2	
0	26.4755307198027339	2.108364	4.675113	9.2864889311972661
2	27.7049859135519760	3.796075	5.772001	8.057033737448024
5	29.1626256536019390	6.193844	5.241760	6.599393997398061
11	33.0019951959638860	9.890708	5.657963	2.760024455036114
19	34.0813961292307113	10.789018	5.652406	1.6806235217692887
20	35.2658236708275794	11.719070	5.757113	0.4961959801724206
42	35.4988740167540221	11.990843	5.292858	0.2631456342459779
77	35.6600160304442468	12.022043	5.791223	0.1020036205557532
101	35.6911182560626420	11.999926	5.788733	0.0709013949373580
150	35.7184962220350073	12.066758	5.796257	0.0435234289649927
296	35.7476142577482392	12.089001	5.798761	0.0144053932517608
332	35.7596094867498664	12.098160	5.799792	0.0024101642501336
455	35.7615533291011332	12.099644	5.799960	0.0004663218988668
501	35.7618716297629646	12.099887	5.799987	0.0001480212370354
579	35.7619305745692486	12.099932	5.799992	0.0000890764307514

Figure (2) shows the graph of comparison results of the fitness average (mathematical mean) of all solutions of the population in one generation when applying BA and GA on optimization of function problem.

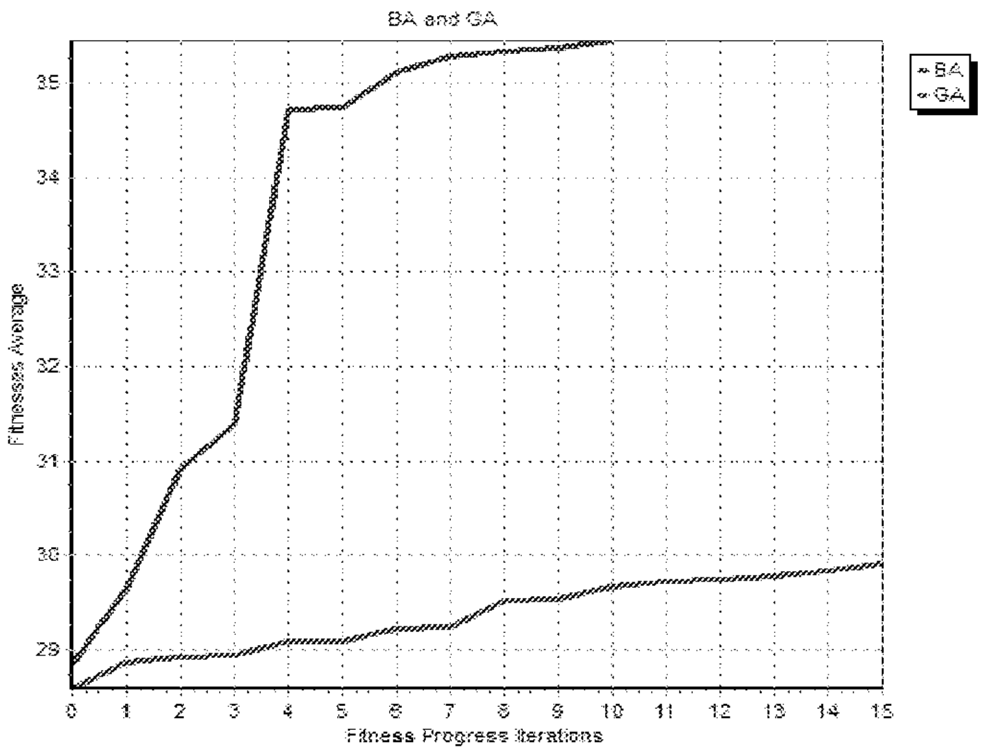

Figure (2) graph of comparison results of applying BA and GA.

9. Conclusions and Future Works

1. Form figure (2), notice that for GA, the average of fitness of solutions of the population is approach 29.91232715 in 1389 iterations with absolute error 5.849692501 which it's a big error. While for BA, the average is about 35.4500599385783 in 1191 iterations with absolute error 0.3119597124217.
2. BA approaches the exact solution in less number of iterations and less value in absolute error from GA.
3. The relative error for GA is about 0.163573, while the relative error for BA is about 0.008723211.
4. The achievement BA is better from GA to approach the good fitness result in less process time, less absolute error, high speed approach and in less number of generations.
5. We suggest making a hybrid between GA and BA in solving optimization of function to improve approaching solution in less time and number of generations.
6. We suggest to use more complicated function includes more than two variables to get solutions to these complicated functions.

References

[1]. Holland, J., "**Adaptation in Natural and Artificial Systems**", University of Michigan Press, 1975.

[2]. Koza J. R., "**Genetic Programming**", 2nd Edition, 1992.

[3]. Kennedy J. and Eberhart R. C. "**Particle Swarm Optimization**", Proceedings of IEEE International Conference on NN, Piscataway, pp. 1942-1948, 1995.

[4]. Xiaodong L., Gao L., and Gao H., "**Swarm Intelligence and its Applications**", Congress on Evolutionary Computation, Canberra, Australia, 8th- 12th December 2003.

[5]. Mitchell M., "**An Introduction of Genetic Algorithms**", Abroad Book 1998.

[6]. Grefenstette J. J., **"Optimization of Control Parameters of Genetic Algorithm**" Genetic Algorithm; IEEE Computer Society, 1992.

[7]. Michalewicz, Zbigniew, "**Genetic Algorithms + Data Structures = Evolution Programs**", Springer-Verlag, 1999.

[8]. Web Site http:// www.tech-faq.com/swarm-intelligence.html, downloaded in 04/04/2011.

[9]. Web Site http:// www.en.wikipedia.org/wiki/Swarm-Intelligence # References-in-popular-culture, download in 15/04/2011.

[10]. Jonas P.,"**Swarm Intelligence**", Communication and Operating Systems Group, Berlin University of Technology, 2007.

[11]. Chong C., Low M. Yoke H., Sivakumar A., and Gay K., "A Bee Colony Optimization Algorithm to Job Shop Scheduling", Proceedings of the 2006 Winter Simulation Conference, N.J., USA. WSC, PP. 1954-196, 2006.

[12]. Ashraf A., Michael P.,and Marco C., "**Bees Algorithm**", Manufacturing Engineering Center, Cardiff University, Wales,UK,2009.

[13]. Pham D. T., Ghanbarzadeh A., Koc E., Otri S., and Zaidi M. **"The Bee's Algorithm – a Novel Tool for Complex Optimization Problems"**. In: Pham D.T., Eldukhri E., Soroka A. J. ed(s) 2nd Virtual International Conference on Intelligence Production Machines and Systems (IPROMS 2006). Elsevier, Oxford, 2006, pp 454-459.

[14]. Steven de J. and Nyree L., "**Bee Behavior in Multi agent Systems**", February, 2008.

[15]. Web Site http://www.en.wikipedia.org/wiki/Ant_colony_optimization, downloa-ded in 25/03/2011.

حل مسألة أمثلية الدوال باستخدام خوارزمية النحل

م. أسيل غازي محمود
جامعة بغداد – كلية التمريض

مستخلص

ان خوارزمية النحل Bees Algorithm (BA) هي خوارزمية مثالية والتي تحاكي سلوك عملية جمع الطعام من قبل سرب نحل العسل والتي اثبتت كونها تمثل تقنيات حسابية عالية الانجاز بسبب قابليتها للبحث.

البحث يحاول تحقيق وبيان فائدة استخدام (BA) في حل مسائل امثلية الدوال. هدف البحث هو استخدام خوارزمية النحل (BA) لحل بعض انواع الدوال غير الخطية التي تعتمد متغيرين والتي تخضع الى حقل أمثلية الدوال.

وقد تم تحقيق دراسة مقارنة بين خوارزمية (BA) والخوارزمية (GA) الجينية لحل هذا النوع من المسائل. ان النتائج العملية المستخصلة من البحث تسلط الضوء على أي الخوارزميتين افضل في حل مسائل أمثلية الدوال.