IMPLEMENTATION OF PARALLEL ARTIFICIAL BEE COLONY ALGORITHM ON VEHICLE ROUTING PROBLEM

Akancha Tiwari[1], Prof. M.Afshar Alam[2]

[1]*Department of Computer Science, Jamia Hamdard, New Delhi*

[2] *Faculty of Management and Information Technology, Jamia Hamdard,New Delhi*

ABSTRACT

Artificial Bee Colony algorithm has proved to be useful in various fields of problems such as assignment problems, scheduling problems, solving transportation problems etc. In this paper we will particularly apply Artificial Bee Colony algorithm to solve vehicle routing problem efficiently. We have also tried to obtain better quality of results by opting for very large search spaces Moreover we will speed up execution of ABC algorithm. For this purpose we have opted for the parallelization of ABC algorithm.

Different swarms are assigned separate threads and different types of communications among these swarms are proposed and examined. Standard benchmark functions were tested and results proved this approach to be superior.

Keywords: Artificial Bee Colony Algorithm, Vehicle Routing Problem, Travelling Salesman Problem, D'Jong's Optimization Functions, Parallelization of Algorithms.

I INTRODUCTION

The Artificial Bee Colony algorithm was introduced by Dervis Karaboga in 2005.Since then many important researches have taken place in the same field. Artficial Bee Colony Algorithm is an efficient algorithm to solve the problem of Vehicle Routing which is a kind of Travelling Salesman Problem. It was found that ABC gave better results for VRP than other algorithms.

So here we propose a method known as parallelization of algorithm which would reduce execution time of ABC algorithm by a large factor. This paper has been organized from the literature review of Travelling salesman problem and ABC algorithm to D'jong's test functions and methods for parallelization of ABC algorithm.

II TRAVELLING SALESMAN PROBLEM

2.1 Literature Review

VRP is an m-TSP problem which is NP-hard, so it cannot be solved in polynomial time of execution. This problem has been solved by algorithms like Genetic algorithm, nearest neighbour algorithm and swarm optimization techniques. It is found that best results have been obtained by SO algorithms. Comparative studies carried out prove that ABC algorithm achieves best optimal results among all swarm intelligence algorithms.

TSP is a problem in which a sales person has to visit certain cities following some path, such that each city is visited only once and then reach back to the place he started from. He should travel in such a way that his travelling cost or we can say travelled distance is minimum.

The formal language for the corresponding decision problem is

TSP = {<G,c,k> : G=(V,E) is a complete graph,

c is a function from V X V -> Z and k ϵ Z,

G has a travelling salesman tour with cost at most k}

Finding an optimized route in various fields is the main application of TSP.[16]

III ARTIFICIAL BEE COLONY ALGORITHM

3.1 Literature Review

Artificial Bee Colony algorithm (ABC) was initially published by Karaboga in 2005 as a technical report for numerical optimization problems. Its development was inspired by simulating the intelligent foraging behaviour of honey bees in their colony and its performance was initially measured using benchmark optimization function.

Pseudo code of the basic bees algorithm:

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.

In step 4, bees that have the highest fitness are chosen as “selected bees” and sites visited by them are chosen for neighbourhood search. Then, in steps 5 and 6, the algorithm conducts searches in the neighbourhood 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 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 neighbourhood 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 each 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.

IV D'JONG'S TEST FUNCTIONS

There are following four test functions against which we have obtained optimized results.

4.1 Sphere function

It is called the first *function of De Jong's* .It is one of the simplest test benchmark. It has the following general definition

$$f1(x) = \sum_{i=1}^{n} x_i^2$$

Function *f1(x)* is Sphere function that is continuous, convex and unimodal. *x* is in the interval of [-100, 100]. Global minimum value for this function is 0 and optimum solution is *xopt =(x1, x2, . . . , xn)* = (0, 0, . . . , 0)..

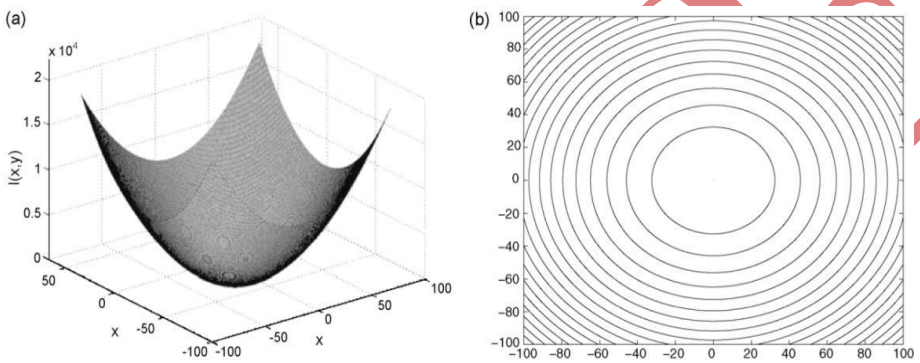

Fig. 4.1 Surface Plot (a) and Contour Lines (b)

4.2 Griewank's function

Griewank's function has many widespread local minima regularly distributed. Function has the following definition

$$f2(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\frac{x_i}{\sqrt{i}} + 1$$

Function *f2(x)* is Griewank function. *x* is in the interval of [-600, 600]. The global minimum value for this function is 0 and the corresponding global optimum solution is *xopt =(x1, x2. . . xn)* = (100, 100, . . . , 100).

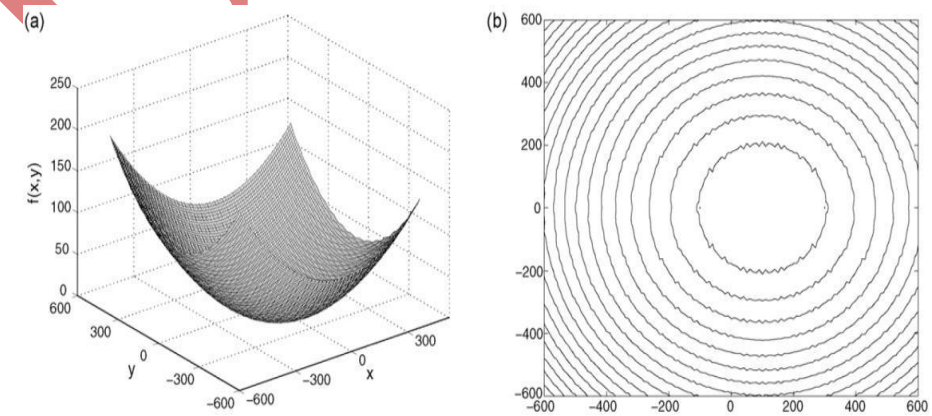

Fig. 4.2 Surface Plot (a) and Contour Lines (b)

4.3 Rastrigin's function

This test function is highly multimodal. However, the locations of the minima are regularly distributed. Function has the following definition

$$f3(x) = 10x + \sum_{i=1}^{n} (x_i^2 - 10\cos 2\pi x_i)$$

Function *f3(x)* is Rastrigin function. *x* is in the interval of [-5.12, 5.12]. The global minimum value for this function is 0 and the corresponding global optimum solution is *xopt =(x1, x2. xn)* = (0, 0 . . . 0).

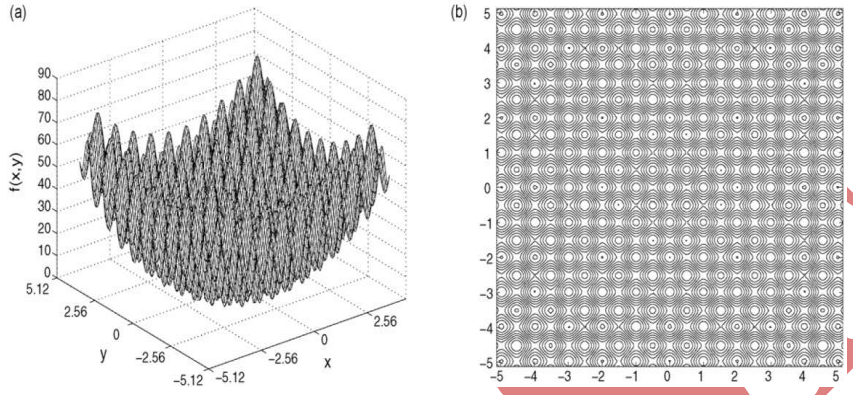

Fig. 4.3 Surface Plot (a) and Contour Lines (b)

4.4 Rosenbrock's function

Rosenbrock's valley is a classic optimization problem, also known as *banana function* or the *second function of De Jong*. Function has the following definition

$$f4(x) = \sum_{i=1}^{n-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$$

Function *f4(x)* is well-known classic optimization problem: Rosenbrock valley. The global optimum is inside a long, narrow, parabolic shaped flat valley *x* is in the interval of [-50, 50]. Global minimum value for this function is 0 and optimum solution is *xopt =(x1, x2, . . . , xn)* = (1, 1, . . . , 1). Global optimum is the only optimum, function is unimodal.

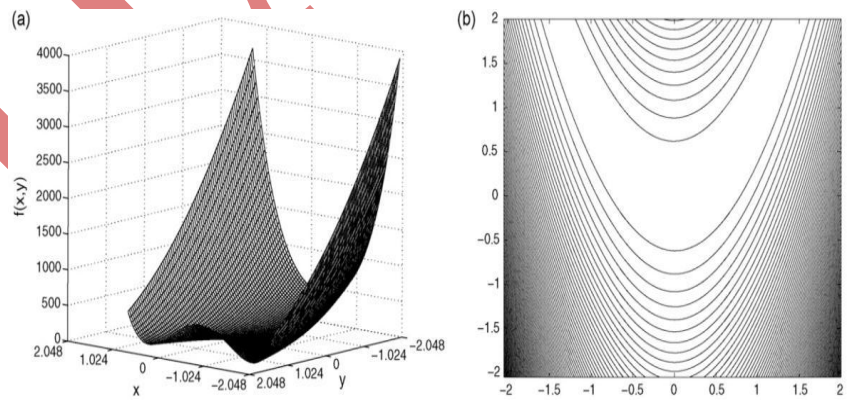

Fig. 4.4 Surface Plot (a) and Contour Lines (b)

Parameter ranges, formulations and global optimum values of these functions are given in Table 1.

Table 3.1 Parameter ranges, formulations and global optimum values [14]

Functions	Ranges	Min Value
Sphere : $f1(x) = \sum_{i=1}^{n} x_i^2$	$-100 \le x_i \le 100$	$f1(x) = 0$
Griewank: $f2(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\frac{x_i}{\sqrt{i}} + 1$	$-600 \le x_i \le 600$	$f2(x) = 0$
Rastrigin: $f3(x) = 10x + \sum_{i=1}^{n}(x_i^2 - 10\cos 2\pi x_i)$	$-5.12 \le x_i \le 5.12$	$f3(x) = 0$
Rosenbrock: $f4(x) = \sum_{i=1}^{n-1}[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$	$-50 \le x_i \le 50$	$f4(x) = 0$

V PARALLELIZATION APPROACHES

We are witnessing a dramatic change in computer architecture due to the multicore paradigm shift, as every electronic device from cell phones to supercomputers confronts parallelism of unprecedented scale . Majority of processors today have multiple cores and even for a single core multiple treads can be implemented. In general, a system of *n* parallel processors, each of speed *k*, is less efficient than one processor of speed *n* * *k*. However, the parallel system is usually much cheaper to build and its power consumption is significantly smaller. [2][10]

5.1 Independent parallel runs approach

It is desirable to run population based heuristics many times, because they do not provide exact result but rather give approximation as final result. It is quite useful to run all iterations simultaneously in order to save time. In this approach threads have no communication between themselves at all. Every thread runs the same sequential ABC algorithm with different random seeds. The final solution is the best one of all the independent runs. The speed increases almost as many folds as there are execution cores in system.

Independent parallel runs approach is too coursed grained and there are no speed gains for one single runtime. On single execution core system this implementation can be slower than serial execution of all runs. This can be explained by high cost of switching CPU between threads. But for today's modern CPU's that is not an issue, hence almost every PC has at least processor with two cores.

VI METHODOLOGY OF VALIDATION

The essential control parameters in the Artificial bee colony algorithm are, the number of food sources which is equal to the number of employed/onlooker bees (CS-Colony Size), the working to onlooker bee rate, the value of the limit (L) and the number of cycles or the number of iterations (MCN) that are required to terminate the program.

Table 5.1 Parameters of ABC corresponding to the VRP[5]

Features of VRP	**Options to ABC**
Number of vehicles	Colony size (number of bees)

Input to vehicles	Data (map consisting locations in the given areas)
Number of depots	Number of food sources
Kind of demand	Non Deterministic
Activities of a vehicle	Working, Onlooker and Scout bees
Maximum time on route	Working time (time constraints)

VII TEST RESULTS

Test parameters for all benchmark function are given in table 2. Limits are calculated by formula:

Limit = 0.25 x NP x D. (D=50 for all functions)

Table 7.1: Parameters for ABC Algorithm

Functions	Max Cycles	NP	Runs
Sphere(f_1)	2500	40	30
Griewank(f_2)	2500	40	30
Rastrigin(f_3)	2500	40	30
Rosenbrock(f_4)	2500	40	30

Test results for various large search spaces for sphere function:

Table 7.2 Test results for various large search spaces

Function Name	**Serial runs(mean of 30 runs)**
Sphere:	
100	1.231925E+003
200	1.102502E+002
300	1.107611E+001
400	1.339968E+000

As we can see that there is quite an improvement in results of ABC algorithm when used for very large search spaces, this is because with large number of locations or search spaces there are more opportunities of exploring and exploiting the search spaces. So there is higher probability of getting better optimal results. And with lesser number of locations we have to restrict ourselves to some confined paths only, which does not utilizes ABC algorithm for VRP efficiently.

Speed test results are shown in the following table. Times are given in seconds.

Table 7.3. Speed tests for benchmark functions for 250 parameters

Functions	**Serial Runs**	**Parallel Ind. Run**
Sphere	18.52	08.55
Griewank	03:08.66	09.66

Rastrigin	02:35.59	07.01
Rosenbrock	05:12.14	08.18

Since the Sphere function is simple function, it requires small amount of CPU time, obtained when serial runs are used. More CPU time is used for creating and synchronizing threads then for calculating Sphere function. Computational time can be prolonged by increasing the number of parameters.

Independent parallel runs approach has no influence on quality of results. The results are shown in Table 3 for all functions.

Table 7.4 Results for VRP implementing Serial and Parallel ABC (only for Rastrigin):

Function Name	**Serial runs**	**Parallel Independent Runs**
Rastrigin:		
Mean of 30 runs:	2.155016E+004	4.368154E+004
Standard Deviation:	4.404791E+003	1.422403E+003
Best Solution:	1.036080E+004	8.716261E+004
Worst Solution:	3.440006E+004	1.376442E+005

VIII TOOLS FOR EMPIRICAL EVALUATION

All of the parallelization approaches have been implemented using Visual C++ programming language. The .Net platform is designed from the ground up to support concurrent programming, with basic concurrency support in the Visual C++ programming language and the C++ class libraries. In the Visual C++ programming language, concurrent programming can be achieved with the help of threads. Threads are sometimes called lightweight processes[15]. Both processes and threads provide an execution environment, but creating a new thread requires fewer resources than creating a new process. A thread is a unit of processing in a program. The Visual C++ allows an application to have multiple threads of execution running concurrently. For test purposes, we created test application in Visual C++ programming language based on Karaboga`s and Bastuk`s software in C programming language. All of our tests have been performed on an Intel(R) Core(TM) i3 @ 2.53 GHz with 3 GB of RAM with Microsoft Windows 7 Home Basic Service Pack 1. We used Microsoft Visual Studio 2010 for the purpose.

IX CONCLUSION

In this paper three different approaches in parallelization of artificial bee colony algorithm ware implemented. The aim was to achieve speed gains by using independent parallel runs approach and to obtain better results by using large number of location approaches. Independent parallel runs implementation is significantly faster than serial runs method, especially if objective functions is complicated and demands a lot of CPU time or /and has a great number of parameters. In the future, as the number of execution cores increases, the time difference between parallel and serial runs will be even greater. And we also observed that using ABC algorithm for larger search spaces gave better results than for lesser search spaces.

REFERENCES

[1] Bee Colony Optimization Overview , Dusan Teodorivic, Milica Selmic [University Of Belgrade, Faculty Of Transport And Traffic Engineering, Vojvode Step 305,11000 Belgrade ,Serbia {Dusan , M.Selmic}@Sf.Bg.Ac.Rs] And Tatjana Davidovic [Mathematical Institute, Serbian Academy Of Sciences And Arts, Kneza Mihaila 36, P.O.Box 367,11000 Belgrade, Serbia{ Tanjad@Mi.Sanu.Ac.Rs}

[2] Different Approaches In Parallelization Of The Artificial Bee Colony Algorithm Milos Subotic, Milan Tuba And Nadezda Stanarevic,International Journal Of Mathematical Models And Methods In Applied Sciences, Issue 4, Volume 5, 2011

[3] Artificial Bee Colony (Abc), Harmony Search And Bees Algorithms On Numerical Optimization, D. Karaboga, B. Akay,Erciyes University, The Dept. Of Computer Engineering, 38039, Melikgazi, Kayseri, Turkiye

[4] A Powerful And Efficient Algorithm For Numerical Function Optimization: Artificial Bee Colony (Abc) Algorithm, Dervis Karaboga · Bahriye Basturk, Received: 31 May 2006 / Accepted: 12 February 2007 / Published Online: 13 April 2007

[5] Artificial Bee Colony (Abc) Algorithm For Vehicle Routing Optimization Problem Ashita S. Bhagade, Parag. V. Puranik, International Journal Of Soft Computing And Engineering (Ijsce) Issn: 2231-2307, Volume-2, Issue-2, May 2012

[6] Improving The Bees Algorithm For Complex Optimisation Problems ,By Sameh Otri, B.Eng., M.Sc. Manufacturing Engineering Centre ,School Of Engineering Cardiff University United Kingdom 2011

[7] Travelling Salesman Problem Using Bee Colony With SPV ,Nishant Pathak, Sudhanshu Prakash Tiwari, International Journal Of Soft Computing And Engineering (Ijsce) Issn: 2231-2307, Volume-2, Issue-3, July 2012

[8] Designing Parallel Meta-Heuristic Methods, Teodor Gabriel Crainic, Tatjana Davidovic, Dusan Ramljak, June 2012, Cirrelt-2012-28

[9] A Comparative Study Of Artificial Bee Colony Algorithm,Dervis Karaboga *, Bahriye Akay,Erciyes University, The Department Of Computer Engineering, Melikgazi, 38039 Kayseri, Turkey, Applied Mathematics And Computation 214 (2009) 108–132

[10] Parallelization Of The Artificial Bee Colony (ABC) Algorithm,Milos Subotic, Milan Tuba, Nadezda Stanarevic,Faculty Of Computer Science Megatrend University Belgrade Bulevar Umetnosti 29 Serbia,Milos.Subotic@Gmail.Com, Tubamilan@Ptt.Rs, Srna@Stanarevic.Com, Recent Advances In Neural Networks, Fuzzy Systems & Evolutionary Computing

[11] Parallel Approaches For Artificial Bee Colony Algorithm,Rafael Stubs Parpinelli, Academia.Edu

[12] An Idea Based On Honey Bee Swarm For Numerical Optimization,(Technical Report-Tr06, October, 2005) Dervis Karaboga Karaboga@Erciyes.Edu.Tr Erciyes University, Engineering Faculty Computer Engineering Department Kayseri/Türkiye.

[13] Code.Msdn.Microsoft.Com/Mag201104beecolony.

[14] Test Functions For Optimization Needs Marcin Molga, Czesław Smutnicki 3 Kwietnia 2005 Streszczenie.

[15]Http://En.Wikipedia.Org/Wiki/Multithreading_(Computer_Architecture), 15/04/2013

[16] Introduction To Algorithms (3rd.Edition) - Thomas H.Cormen,Charles E.Leiserson,Ronald L.Rivest,Clifford Stein

[17] The Bees Algorithm, and Its Applications Dr. Ziad Salem, HDD, PhD, BsC. Spezs1@hotmail.com Computer Engineering Department Electrical and Electronic Engineering Faculty Aleppo University, Aleppo, Syrian Arab Republic Masaryk University, Brno, Czech Republic , Wed 08 Apr 2009