Volume 3, Issue 5, May 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Effectiveness of FireFly Algorithm in solving Bin Packing Problem

R.yesodha*
Department of Computer Applications
Bharathiar University, Coimbatore, India.

T.Amudha
Department of Computer Applications
Bharathiar University, Coimbatore, India.

Abstract— Bin Packing Problem (BPP) is one of the NP-Hard optimization problems. Bin packing is an important task in solving many real world problems such as transportation planning, Container Loading, Resource Allocation, Scheduling, Cargo Airplanes and Ships. The foremost objective of the Bin Packing problem is to minimize the number of bins used for storing the set of all items exactly once with different weights. Bio-inspired algorithmic usage has been very promising in various fields ranging from computer science, electronics and mechanical engineering to chemical engineering and molecular biology. This field is much closely related to the domain of Artificial Intelligence. In this paper, FireFly Algorithm (FFA) and two traditional algorithms, First-Fit and Best-Fit were implemented to solve Bin packing problem. The performance of these algorithms was evaluated by testing on three classes of benchmark Data sets which was taken from OR Library. The results have shown that the Bio-inspired metaheuristics have performed well for most of the problem cases.

Keywords— Bio-inspired computing, Bin Packing Problem, FireFly Algorithm, First-Fit Algorithm, Best- Fit Algorithm

I. INTRODUCTION

Biologically-inspired methods are becoming increasingly important in face of the complexity of today's demanding applications [3]. The Bin Packing problem (BPP) was introduced by Garey & Johnson in 1979 [17]. It is a combinatorial NP-hard problem. In this paper one dimensional Bin Packing problem (1-BPP) with fixed sized bins was used. Bin Packing problems variants which includes one dimensional, two dimensional, three dimensional Bin Packing problems, High Multiplicity Bin Packing Problems, linear packing, packing by weight, packing by cost etc., [1, 3, 6]. Bin packing most commonly it involves packing smaller items into defined spaces. In the real world, the critical issue is to make efficient use of time and space. [11, 15]. Bin packing problem has wide opportunity to solve with the new era of Bio-inspired computing. Very few Bio-inspired metaheuristics were applied in recent years such as Ant Colony Optimization algorithm (ACO), Cuckoo search algorithm (CS) and Genetic algorithm (GA) to Bin Packing problem and the results were proven to be highly efficient. In this research work BPP was solved by using two traditional heuristics and one metaheuristic which was inspired by biology [1, 3, 6].

II. RELATED WORK

Aphirak Khadwilard., et al., (2011) have presented the successful application of a recent developed metaheuristic called FireFly Algorithm for solving JSSP, explore the parameters of the proposed algorithm and investigated the performance of the FireFly Algorithm with different parameter setting and compare with best known solution. The computational experiment was designed and conducted using five benchmarking JSSP data sets from a classical OR-Library. The analysis was aimed to study the effect of the FireFly parameter setting on its performance before comparing the FireFly Algorithm results between using and not using optimized parameter settings. The FireFly Algorithm performance with optimized parameter setting could produce the operation schedule better than the other parameter setting [2].

Sh. M. Farahani., et al., (2011) have stabilized the firefly's movement and proposed a new behavior to direct firefly's movement to a global best. In their proposed algorithm, if a firefly can't find any better firefly that is brighter in its neighborhood, it will move towards global best in that iteration, so firefly's movement will direct to better solution and algorithm can guide them to better state, so they can get near to optimum solution at the end of iteration. Moving fireflies by a Gaussian distribution as a social behavior causes a better position for each of them for next iteration and fireflies with worse cost have more chance to move to global best with a longer step length. Simulation results have shown a better performance than standard Firefly algorithm. Proposed algorithm was tested on five standard functions that were commonly used for testing the static optimization algorithms [7].

Abdesslem Layeb and Seriel Rayene Boussalia (2012) have proposed a new stochastic method based on the Quantum Inspired Cuckoo Search algorithm (QICSA) to deal with the 1-BPP problem. A new hybrid measure operation integrates the FF heuristic in the core of the standard measure operation. The great feature of this hybrid measure is its great ability to generate different good random solutions. The proposed algorithm reduces the population size and the number of

© 2013, IJARCSSE All Rights Reserved

iterations to have the optimal solution. The obtained results are very encouraging and show the feasibility and effectiveness of the proposed approach [1].

J.Y. Lin., et al., (2011) considered the NP-Hard problem of online Bin Packing while requiring that larger items be placed below smaller items such a version is called as LIB version of problems. Bin sizes can be uniform or variable. They provided the analytical upper bounds as well as computational results on the asymptotic approximation ratio for the First-Fit algorithm. This paper deals with problem instances of variable bin sizes and the LIB constraint. An upper bound "BFF" for Variable Sized Bin Packing with LIB is derived [8].

Petar Maymounkov (2001) in this paper the author has extended Kenyon and Mitzenmacher's technique for proving diveregnce of the online approximation algorithm Best Fit to Random Fit another approximation algorithm for the well-known NP-hard problem of bin packing [10].

Mohamed Maiza and Mohammed Said Radjef (2011) have aimed to solve one dimensional Bin Packing problem with conflicts. The conflicts are represented by a graph whose nodes are the items, and adjacent items cannot be packed into the same bin. They have proposed an adaptation of Minimum Bin Slack heuristic with a combination of heuristics based on the uses of the classical bin-packing methods to packing items of maximal-stable-subsets (MSS). Computational results based on a benchmark test showed good performance of proposed heuristics both on quality of solution and execution time [9].

Ruslan Sadykov and Francois Vanderbeck (2012) presented a branch-and-price algorithm for the bin packing problem. Implementation was made by using the software platform BaPCod. They developed a dynamic programming algorithm for pricing when the conflict graph is an interval graph and a depth-first-search and branch-and-bound approach for pricing when the conflict graph has no special structure. Their computational experiment report sets new benchmark results for this problem. The algorithm was tested on instances from the literature and newly generated ones. Branch-and-price algorithm gave comparatively better results [12].

III. BIN PACKING PROBLEM SOLUTIONS

This paper aimed to solve the NP-Hard Bin Packing Problem by using FireFly Algorithm, First-Fit and Best-Fit algorithm and to examine the efficiency of these algorithms in solving benchmark instances of BPP. The Bin Packing problem can be formulated as follows:

$$\text{Minimize } \sum_{j=1}^{n} y_j \qquad (1)$$

subject to constraints

$$\sum_{i=1}^{m} w_i x_{ij} \le c y_j \qquad \forall j \in \{1, \dots n\} \qquad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall i \in \{1, \dots n\} \qquad (3)$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in \{1, \dots n\}, \forall j \in \{1, \dots n\} \qquad (4)$$

$$y_i \in \{0, 1\} \qquad \forall j \in \{1, .. n\} \qquad (5)$$

With, yi= 1 if the bin i is used; else 0

xij= 1 if the item j is stocked in bin i.

Notations

c - Bin capacity

i,j - Index of items and bins respectively

wi - Weight or size of item i

m - minimal number of bins

Equation (1) states the objective function which is to minimize the number of bins assigned. Equation (2) ensures the constraint that the bin capacity is not exceeded by the sum of item sizes assigned to this bin. This is often called the capacity constraint. Equation (3) ensures that every item is packed to exactly one bin. Equation (4) and (5) determine that all variables are binary [1, 5].

A. Firefly Algorithm

The FireFly Algorithm (FFA) is a metaheuristic algorithm, inspired by the flashing behaviour of fireflies. The primary purpose for a firefly's flash is to act as a signal system to attract other fireflies. The flashing characteristics of fireflies consequently develop FireFly-inspired algorithms. There are three idealized rules. On the first rule, each firefly attracts all the other fireflies with weaker flashes. All fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their sex. Secondly, attractiveness is proportional to their brightness which is reverse proportional to their distances. For any two flashing fireflies, the less bright one will move towards the brighter one [4]. The attractiveness is proportional to the brightness and they both decrease as their distance increases. If there is no brighter one than a particular firefly, it will move randomly. Finally, no firefly can attract the brightest firefly and it moves randomly. Based on these three rules, the basic steps of the FireFly Algorithm (FFA) can be summarized as the pseudo code shown below. In the FireFly algorithm there are two important issues of the variation of light intensity and the formulation of the attractiveness. The light intensity I(r) varies with distance r monotonically and exponentially

$$I = I_0 e^{-\gamma r} \longrightarrow \qquad (6)$$

© 2013, IJARCSSE All Rights Reserved

where I0 is the original light intensity and γ is the light absorption coefficient. For simplicity, it is assumed that the attractiveness of a firefly is determined by its brightness which in turn is associated with the encoded objective function of the optimization problems. On the attractiveness of the FFA the main form of attractiveness function or β(r) can be any monotonically decreasing functions such as the following generalized form of

$$\beta(r) = \beta_0 e^{-\gamma r^2} \longrightarrow \quad (7)$$

where r or rij is the distance between the ith and jth of two fireflies. β0 is the attractiveness at r = 0 and γ is a fixed light absorption coefficient. The distance between any two fireflies i and j at xi and xj is the Cartesian distance as follows:

$$r_{ij} = \|x_i - x_j\|_2 = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \longrightarrow \quad (8)$$

where xi,k is the k-th component of the i-th firefly (xi). The movement of a firefly, i is attracted to another more attractive (brighter) firefly j, is determined by

$$x_i = x_i + \beta_0 e^{-\gamma r2_{ij}} (x_j - x_i) + \alpha*(rand-0.5) \longrightarrow \quad (9)$$

where the second term is due to the attraction while the third term is the randomization with α being the randomization parameter. Rand is a random number generator uniformly distributed in the range of [0, 1]. In this paper, the initial population of firefly is 5 and the iterations are 20. In the each iteration 5 different fireflies pack the number of items with different weights and its corresponding capacity from different positions and produce different solutions. Each firefly calculates the light intensity based on the objective function of the BPP and also based on the distance. The attractiveness of a firefly is determined by its brightness and varies according to the distance. The distance of each firefly is calculated based on the starting position which it starts packing the items. In this research work the distance between fireflies is calculated by using Cartesian distance. The movement of one firefly is attracted to another more attractive (brighter) firefly [7, 2].

PSEUDO CODE OF FIREFLY ALGORITHM

Procedure FFA Meta-heuristic()
Begin
Initialise algorithm parameters
MaxGen: the maximal number of generations
γ: the light absorption coefficient
r: the particular distance from the light source
d: the domain space
Define the objective function of f(x), where x=(x_1,.....,x_d)T
Generate the initial population of fireflies or x_i (i=1, 2 ,..., n)
Determine the light intensity of I_i at x_i via f(xi)
***While** (t<MaxGen)*
***For** i = 1 to n (all n fireflies)*
***For** j=1 to n (n fireflies)*
***if** (I_j > I_i), move firefly i towards j*
end if
Attractiveness varies with distance r via Exp [-γr2]
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 visualization
End procedure

TABLE I
PARAMETER SETTINGS FOR FIREFLY ALGORITHM

Parameter	Values	Description
α	0.2	Alpha
β_0	0.3	$Beta_0$
γ	0.2	Gamma
Firefly	5	Population of Fireflies
Iterations	20	Generations

B. First-Fit Algorithm

First-Fit (FF) keeps all empty bins open. It places the next item in the lowest numbered bin in which the item fits. If it does not fit in any bin, a new bin is opened. First-Fit could not achieve better execution time as it keeps all non-empty bins active and tries to pack every item in these bins before opening a new one. In this algorithm the rule followed is:

© 2013, IJARCSSE All Rights Reserved

First place an item in the first bin, called lowest indexed bin into which it will fit, i.e., if there is any partially filled bin then place the item in the lowest indexed bin otherwise, start a new bin [14].

PSEUDO CODE OF FIRST-FIT ALGORITHM

Procedure First-Fit ()
Begin
1: for *All objects i = 1, 2, . . . , n do*
2: for *All bins j = 1, 2, . . . do*
3: if *Object i fits in bin j then*
4*: Pack objects i in bin j.*
5*: Break the loop and pack the next object*
6: end if
7: end for
8: if *Object i did not fit in any available bin then*
9: *Create new bin and pack object i*
10: end if
11: end for
End procedure

C. Best-Fit Algorithm

Best-Fit (BF) is the best known algorithm for Bin Packing problem. It is simple and behaves well in practice. Best Fit picks (among the possible bins for the item) the one where the amount of free space is minimal. It picks the bin with the least amount of free space in which it can still hold the current element. This algorithm tries to choose the fullest bin possible with enough space each time an item is assigned. All unoccupied bins are kept open until the end. It places the next item in the bin whose current contents is the largest, but should not exceed its capacity. If it does not fit in any bin, new bin is opened [13].

PSEUDO CODE OF BEST-FIT ALGORITHM

Procedure Best-Fit ()
Begin
1: for *All objects i = 1, 2, . . . , n do*
2: for *All bins j = 1, 2, . . . do*
3: if *Object i fits in bin j then*
4: *Calculate remaining capacity after the object has been added*
5: end if
6: end for
7: *Pack object i in bin j, where j is the bin with minimum remaining capacity after adding the object (i.e. the object "fits best")*
8: If *no such bin exists, open a new one and add the object*
9: end for
End procedure

IV. IMPLEMENTATION RESULTS AND DISCUSSION

The results obtained from FireFly Algorithm were compared with First-Fit and Best-Fit algorithm. Experimental results and comparative study of the algorithms are discussed in this section.

A. Bin Packing Test Problems

In this research work three classes of benchmark Data set 1, Data set 2 and Data set 3 for BPP-1 are classified as Easy, Medium and Hard classes instances. The benchmark in the easy class instance, n (number of items) ranges from 50 – 500 and c (bin capacity) ranges from 100-150. In the medium class instance n ranges from 50 – 500 and c equals 1000 and in hard class instances n equals 1000 and c equals 100000 [17].

B. Implementation Results of Easy Class Instances

In this research work, 15 instances of easy class were solved by using First-Fit, Best-Fit and FireFly Algorithms. The results of three algorithms were compared and analyzed. FireFly Algorithm gave optimal solutions for 6 instances out of 15 with minimum bin space wastage. Best-Fit algorithm gave optimal solutions for 12 instances, with minimum bin space wastage and for remaining 3 instances, it obtained near optimal solutions when compared to First-Fit and FireFly Algorithm. Whereas, out of 15 instances First-Fit algorithm was found to be capable of producing 8 near optimal solutions and incase of 7 instances, it could not achieve optimal solutions. While comparing to the three techniques, FireFly technique have shown good performance for 8 instances out of 15 instances. It was found from the results that both FireFly and Best-Fit algorithm are efficient in terms of execution time when compare to First-Fit algorithm. The computational results are given in Table 2 and the Figure 1 and Figure 2 shows the graphical representation of Table 2.

© 2013, IJARCSSE All Rights Reserved

TABLE 2
COMPARATIVE RESULTS OF EASY CLASS BPP

Ins Name	N	C	Opt	FIRST- FIT TECHNIQUE			BEST- FIT TECHNIQUE			FIREFLY TECHNIQUE		
				Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)
N1C1W1_B	50	100	**31**	35	718	46	**31**	**318**	**12**	**31**	**318**	**12**
N1C1W1_C	50	100	**20**	26	616	31	**21**	**116**	**10**	**21**	**116**	**10**
N1C1W1_N	50	100	**25**	30	579	31	**26**	179	12	28	269	13
N1C2W1_H	50	120	**23**	30	977	31	**23**	**137**	**10**	**23**	**137**	**10**
N1C2W1_I	50	120	**27**	35	1280	31	**27**	**320**	**12**	**27**	760	13
N1C2W1_K	50	120	**24**	32	1144	31	**24**	**184**	**10**	**24**	**184**	**10**
N1C2W1_L	50	120	**25**	32	1093	32	**25**	**253**	**10**	**25**	**253**	**10**
N1C3W1_F	50	150	**20**	29	1435	32	**20**	**85**	**10**	**20**	**85**	**10**
N2C1W1_A	100	100	**48**	61	1428	32	**48**	**128**	**15**	49	228	**15**
N2C2W1_J	100	120	**42**	58	6960	31	**42**	**54**	**15**	51	538	16
N2C3W1_A	100	150	**35**	50	3862	31	**35**	**144**	**13**	36	174	15
N3C1W4_S	200	100	**145**	169	4025	47	**145**	**1625**	**20**	146	3750	20
N3C1W1_A	200	100	**105**	131	3012	31	**106**	**512**	**16**	**106**	**512**	**16**
N4C1W1_L	500	100	**258**	329	7557	78	**258**	**457**	**25**	262	6685	29
N4C1W1_M	500	100	**246**	310	6578	78	**246**	**178**	**25**	249	478	25

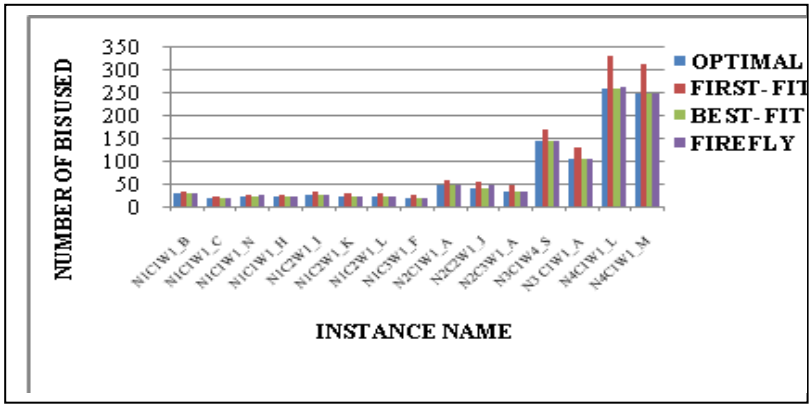

Fig. 1. Number of bins used for Easy class

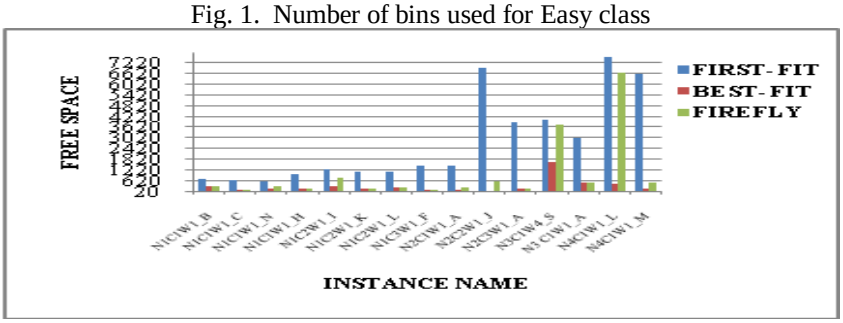

Fig. 2. Bin space wastage for Easy class

© 2013, IJARCSSE All Rights Reserved

C. Implementation Results of Medium Class Instances

In medium class, 20 instances were solved. FireFly Algorithm gave optimal solutions for 10 instances with minimum bin space wastage and for remaining 10 instances it obtained near optimal. Best-Fit algorithm gave optimal solutions for 9 instances, with minimum bin space wastage and for remaining 11 instances; it produced near optimal solutions when compared to First-Fit and FireFly Algorithm. Whereas, out of 20 instances First-Fit algorithm was found to be capable of producing 3 optimal solutions and for remaining 17 instances, it found near optimal solutions. It was found from the results that FireFly technique have shown good performance for 19 instances out of 20 instances when compare to other two techniques. Whereas, Best-Fit technique have achieved 10 instances out of 20 instances. Both FireFly and Best-Fit algorithm are efficient in terms of execution time when compare to First-Fit algorithm. The computational results are given in Table 3. Figure 3 and Figure 4 and shows the graphical representation of Table 3.

TABLE 3
COMPARATIVE RESULTS OF MEDIUM CLASS BPP

Ins Name	N	C	Opt	FIRST- FIT TECHNIQUE			BEST- FIT TECHNIQUE			FIREFLY TECHNIQUE		
				Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)
N1W1B1R0	50	1000	**18**	21	4066	31	20	3066	15	**19**	**2066**	**13**
N1W1B1R1	50	1000	**18**	22	5054	31	20	3054	15	**19**	**2054**	**13**
N1W1B1R3	50	1000	**18**	21	4155	31	20	3155	15	**19**	**2155**	**13**
N1W1B1R4	50	1000	**17**	21	4590	31	19	2590	15	**18**	**1590**	**13**
N1W1B1R5	50	1000	**17**	21	4534	31	19	2534	15	**17**	**534**	**13**
N1W1B1R6	50	1000	**17**	21	4239	31	19	2239	15	**18**	**1239**	**13**
N1W1B1R7	50	1000	**17**	21	4411	31	19	2411	15	**18**	**1411**	**13**
N1W1B1R8	50	1000	**17**	21	4184	31	19	2590	15	**18**	**1590**	**13**
N1W4B1R8	50	1000	**6**	**6**	**624**	31	**6**	**624**	**10**	**6**	**624**	**10**
N1W1B3R0	50	1000	**17**	26	1578	31	**17**	**866**	**15**	**17**	**866**	**15**
N1W1B3R7	50	1000	**18**	26	8069	31	**19**	**1069**	**15**	**19**	**1069**	**15**
N1W3B2R0	50	1000	**8**	**8**	**842**	16	**8**	**842**	**10**	**8**	**842**	**10**
N1W3B3R4	50	1000	**8**	**8**	**715**	16	**8**	**715**	**10**	**8**	**715**	**10**
N2W1B1R0	100	1000	**34**	41	8106	32	37	4106	18	**34**	**4212**	**15**
N2W1B1R6	100	1000	**34**	43	9511	32	38	4511	18	**36**	**2511**	**15**
N2W4B2R1	100	1000	**11**	12	1331	16	**11**	**331**	**10**	**11**	**331**	**10**
N3W4B3R0	200	1000	**24**	26	2068	31	**24**	**68**	**15**	25	1068	18
N4W4B3R4	500	1000	**55**	59	4543	47	**55**	**543**	**23**	**55**	**543**	**21**
N4W4B3R6	500	1000	**58**	62	4595	47	**58**	**595**	**23**	**58**	**595**	**21**
N4W4B3R9	500	1000	**56**	60	4066	42	**56**	**696**	**23**	**56**	**696**	**21**

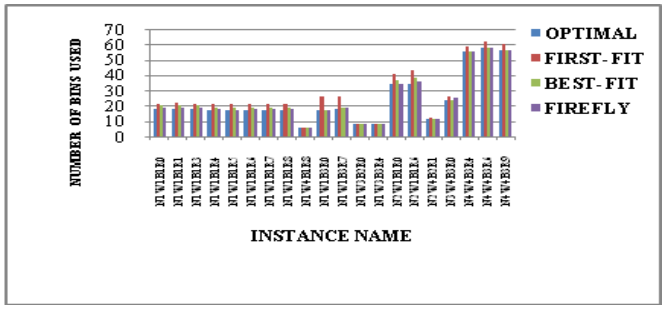

Fig. 3. Number of bins used for Medium class

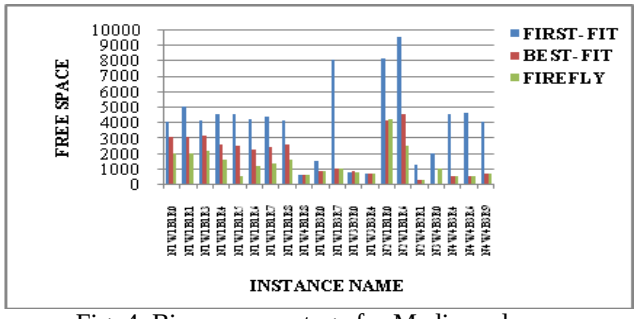

Fig. 4. Bin space wastage for Medium class

© 2013, IJARCSSE All Rights Reserved

D. Implementation Results of Hard Class Instances

Hard class instances of Bin Packing problem finds more difficult and challenging to solve. In hard class, 10 instances were solved. It was found from the results that all the three algorithms have obtained near optimal solutions. FireFly Algorithm shows good performance for 3 instances out of 10 instances when compared to First-Fit and Best-Fit techniques. Whereas, Best-Fit algorithm shown 8 instances out of 10. The computational results and the graphical representation of Table 4 are shown below.

TABLE 4
COMPARATIVE RESULTS OF HARD CLASS BPP

Ins Name	N	C	Opt	FIRST- FIT TECHNIQUE			BEST- FIT TECHNIQUE			FIREFLY TECHNIQUE		
				Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)	Used bins	Free Space	Exe Time (ms)
HARD1	200	100000	**57**	66	1061159	31	**60**	**461159**	**15**	**60**	**461159**	**15**
HARD2	200	100000	**56**	66	1044214	47	**60**	**444214**	**15**	61	544214	15
HARD3	200	100000	**55**	64	958397	31	**59**	**458397**	**15**	60	558397	15
HARD4	200	100000	**57**	65	949303	32	**60**	**449303**	**15**	61	549303	15
HARD5	200	100000	**56**	65	1019247	31	**59**	**419247**	**14**	60	519247	15
HARD6	200	100000	**57**	65	955534	31	**60**	**455534**	**14**	61	555534	15
HARD7	200	100000	**55**	63	900807	31	59	500807	15	**58**	**400807**	**13**
HARD8	200	100000	**57**	66	1046029	47	**60**	**446029**	**15**	61	546029	17
HARD9	200	100000	**56**	65	999542	32	**60**	**499542**	**15**	**60**	**499542**	**15**
HARD0	200	100000	**56**	65	1059718	31	**59**	**459718**	**15**	60	559718	15

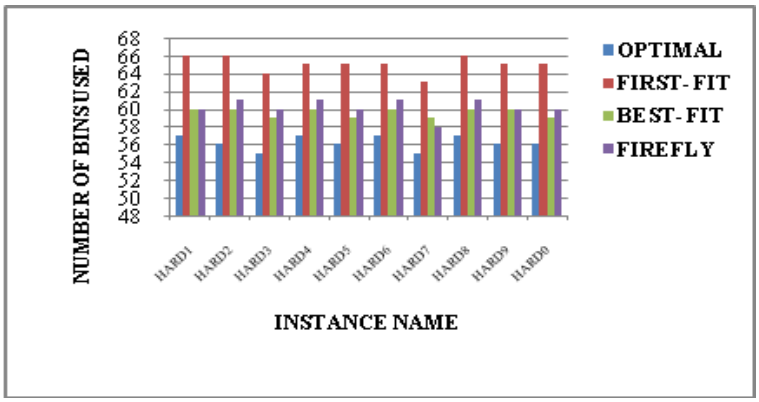

Fig. 5. Number of bins used for Hard class

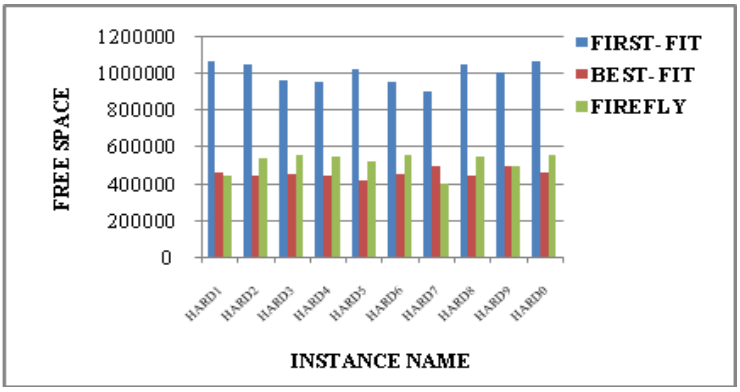

Fig. 6. Bin space wastage for Hard class

© 2013, IJARCSSE All Rights Reserved

V. Conclusion And Future Work

In this paper, a combinatorial optimization problem, the one dimensional Bin Packing problem with fixed sized bins was solved by using Bio-inspired algorithm, FireFly Algorithm (FFA) and two traditional algorithms, First-Fit and Best-Fit. The main purpose of the Bin Packing Problem is to pack the items with different weight into finite number of bins without exceeding its capacity and minimizing bin space wastage and execution time. These algorithms were tested on three classes of standard benchmark problem instances which were taken from OR Library. The results obtained by the FireFly Algorithm shows good performance for most of the problem cases when compared to First-Fit and Best-Fit algorithms. This work can be extended by modifying the parameter settings for FireFly Algorithm to improve the results. FireFly Algorithm can be effectively modified and be made suitable for other types of combinatorial optimization problems. FireFly Algorithm can also be applied to solve various other types of Bin Packing problem such as two dimensional, three dimensional etc.

References

[1] Abdesslem Layeb & Seriel Rayene Boussalia, “A Novel Quantum Inspired Cuckoo Search Algorithm for Bin Packing Problem”, I.J. Information Technology and Computer Science, 5, 58-67 2012.

[2] Aphirak Khadwilard, Sirikarn Chansombat, Thatchai Thepphakorn, Peeraya Thapatsuwan, Warattapop Chainate and Pupong Pongcharoen, “Investigation of Firefly Algorithm Parameter Setting for Solving Job Shop Scheduling Problems”, 2011.

[3] Bijaya Kumar Nanda & Gyanesh Das, “Ant Colony Optimization. A Computational Intelligence Technique”, International Journal of Computer & Communication Technology (IJCCT), Volume-2, Issue-VI 2011.

[4] Chai-ead N, P. Aungkulanon and P. Luangpaiboon, "Bees and Firefly Algorithms for Noisy Non-Linear Optimisation Problems", Proceedings of the International MultiConference of Enginers and Computer Scientists, Vol II, 2011.

[5] Doina Bein, Wolfgang Bein and Swathi Venigella, "Cloud Storage and Online Bin Packing", accepted at Intelligent Distributed Computing Conference (IDC 2011), The Delft, Netherlands, October 5-7, 2011.

[6] E.Falkenauer, & A. Delchambre, “A Genetic Algorithm for Bin Packing and Line Balancing”, Proceedings of the IEEE International Conference on Robotics and Automation 1992.

[7] Farahani Sh. M., A. A. Abshouri, B. Nasiri, and M. R. Meybodi, "A Gaussian Firefly Algorithm", International Journal of Machine Learning and Computing, Vol. 1, No. 5, December 2011.

[8] J.Y. Lin, P. Manyem and R.L. Sheu, "Performance Estimations of First Fit Algorithm for Online Bin Packing with Variable Bin Sizes and LIB constraints", 2011.

[9] Mohamed Maiza and Mohammed Said Radjef "Heuristics for Solving the Bin-Packing Problem with Conflicts", Applied Mathematical Sciences, Vol. 5, no. 35, 1739 – 1752, 2011.

[10] Petar Maymounkov, "Divergence-proving Techniques for Best Fit Bin Packing and Random Fit", April 2, 2001.

[11] Prof. Dr. Karsten Weihe and Prof. Dr. Matthias Müller-Hannemann, "Solution Techniques for specific Bin Packing Problems with Applications to Assembly Line Optimization", 2008.

[12] Ruslan Sadykov and Francois Vanderbeck, "Bin Packing with Conicts: a Generic Branch-and-Price Algorithm", 2012.

[13] Swathi Venigella, "Cloud Storage and Online Bin Packing", May 2008.

[14] Ullas Mohan, “Bio Inspired Computing”, Division Of computer science, SOE, CUSAT, July 2008

[15] Zhihua Cui, Zhongzhi Shi and Janos Abonyi, "New trends on bio-inspired computation", Transactions of the Institute of Measurement and Control 34(6) 665– 667, 2011.

[16] http://www.wiwi.uni-jena.de/Entscheidung/binpp/

[17] http://www2.informatik.hu-berlin.de/alkox/lehre/lvws1011/coalg/bin_packing.pdf

© 2013, IJARCSSE All Rights Reserved