Impact Factor (SJIF): 3.632

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 3, Issue 11, November-2016

SURVEY ON BIN PACKING PROBLEM

Suman Sahu[1], Asst.Prof. Prabhakar Sharma[2],

Department of Computer Science and Engineering, RITEE Raipur (C.G.), sumansahu19@gmail.com
Department of Computer Science and Engineering, RITEE Raipur (C.G.),sharmaprabhakar@rediffmail.com

Abstract— Bin Packing Problem (BPP) is one of the NP-Hard optimization problems. The bin packing problem aims to pack a set of items in a minimum number of bins, with respect to the size of the items and capacity of the bins.BPP has many important applications such as multiprocessor scheduling, resource allocation,transportation planning, Container Loading, Cargo Airplanes and Ships ,real-world planning, and packing and scheduling optimization problems.In this paper, best fit algorithms, first fit algorithm, best fit decresing algorithms, first fit decresing algorithm ,map reduce algorithms,Martello and Toth algorithms ,particle swarm optimization algorithms and Genetic Algorithm are analyzed deeply. This research paper presents the thorough survey of algorithms for optimize Bin Packing problem .

Keywords- best fit algorithms, first fit algorithm, best fit decresing algorithms, first fit decresing algorithm ,map reduce algorithms,Martello and Toth algorithms ,particle swarm optimization algorithms and Genetic Algorithm

I. INTRODUCTION

In bin packing problem, main objective is pack every one of the articles in minimum number of bins or maximize the profit associated with bins. The expression "bin" here is in fact a generic name which could stand for a "container", as in the "transportation" context, "work stations " in industrial assembly lines (line balancing), "a space in time" in scheduling, or a "surface area", as in metal working, for example.The various applications of BPP in our day to day life can include Loading trucks with weight limit limitations ,Creating document backups in removable media ,Filling up containers, Storing an expansive gathering of music onto tapes/CD's, Computer processor selection with job assignment, In metal working where steel sheets of various sizes must be cut from "master" sheets[1]. Different techniques have been proposed to solve a variety of packing problems (exactly or approximately) and these techniques might be gathered into the three wide classes. heuristics meta heuristics and exact methods Heuristics are techniques taking into account natural and/or conceivable arguments that give good solutions under certain conditions, but do not guarantee optimality. Meta heuristics are approximate procedures that efficiently and effectively search through the arrangement space (or a subspace thereof) heuristically, by iteratively employing a collection of (possibly greedy) heuristics. At last, correct strategies are guaranteed to find optimal solutions[2]. Bin packing problem is a combinatorial NP-hard issue. Various algorithms have been worked out for obtaining an ideal arrangement by constraining its upper limits . The offline versions of algorithms are best ones and provide an optimal solution in relatively less time. The online versions are more practical but are far away from the optimal solution. So a various algorithms have been proposed to accomplish optimality in less time[3].Epstein et al have discussed also, looked at the online algorithms of the Bin Packing Issue . Different improvements in the online version has also been discussed. Gyorgy et al propose one by presenting On-line Sequential Bin Packing.In online version of this problem, the input items are taken as they come, without prior knowledge of the next one. The work is just to take every data item as it come and allocate it the best possible bin, before considering the next input, and move towards an ideal arrangement. Though in offline version, complete information about all the items that are to be balanced in the unit measured containers is now provided[4]. It is comparatively easier to reach towards an ideal arrangement in disconnected from the net form than on the web, as earlier knowledge of all the item sizes is given. Next fit, First fit and Best fit are the three famous online version algorithms used to derive solutions of this issue. To start with Fit Decreasing and Best Fit Decreasing are the two offline versions.

II. RELATED WORK

One-dimensional BPP has been tackled by various heuristics and exact techniques. A classical reference for the BPP is the book by Martello and Toth (1990), in which theydepict a number of simple heuristics and lower bounds, introduce a reduction procedure (MTRP), and an exact algorithm (MTP). Best-Fit-Decreasing (BFD) and the First-Fit-Decreasing (FFD) calculations are the two best known heuristics. Both heuristics sort items in decreasing order and place the largest item in either the first bin it fits (FFD) alternately the container with the smallest but sufficient remaining capacity (BFD). A new bin is added whenever no suitable bin can be found.. Petar Maymounkov (2001) in this paper the author has amplified Kenyon and Mitzenmacher's technique for proving diveregnce of the online estimation calculation Best Fit to Random Fit another approximation algorithm for the wellknown NP-hard problem of bin packing [5].Sh. M. Farahani., et al., (2011) have stabilized the firefly's movement and proposed a new behavior to direct firefly's development 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 development will direct to better arrangement and calculation can guide them to better state, so they can get near to optimum arrangement toward the end of emphasis. Moving fireflies by a Gaussian distribution as a social behavior causes a better position for each of them for next cycle and fireflies with worse cost have more chance to move to global best with a more drawn out step length. Recreation results have appeared better performance than standard Firefly algorithm. Proposed algorithm was tried on five standard capacities that were commonly used for testing the static optimization algorithms. J.Y. Lin., et al., (2011) considered the NP-Hard problem of online Bin Packing while requiring that bigger things be put underneath littler things such a variant 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 proportion for the To start with Fit calculation. This paper manages issue instances of variable bin sizes and the LIB constraint. An upper bound "BFF" for Variable Sized Bin Packing with LIB is derived [6]. Fleszar and Charalambous (2011) proposed a strategy of controlling the average weight of items packed by bin-oriented heuristicsin which valuable heuristics and a change heuristic are introduced.Memetic algorithms are successfully used for the one dimensional BPP and one of these methodologies used separate individual learning or local improvement procedures (Le, Ong, Jin, & Sendhoff, 2009; Ong, Lim, Zhu, & Wong, 2006). Segura, Segredo, and Leon(2011) described a multi-objectivized memetic algorithm for the two-dimensional BPP which performs faster than traditional genetic algorithms. Coleman and Wang (1992) presented a bin packing heuristic that is well-suited for implementation on massively parallel SIMD on the other hand MIMD registering frameworks. The normal case conduct of the technique was predictable when the input data has a symmetric dispersion[7]. The strategy,. The method is asymptotically ideal, yields perfect packing and achieves the best possible average case behavior with high probability**.**

III. METHODOLOGY

1.Martello &Toth Algorithms

The best existing algorithm for finding ideal arrangements to bin-packing problems is due to Martello and Toth (Martello &Tooth 1990a; 1990b). Their branch-and-bound calculation is complex, and we describe here only the main features. Their fundamental issue space takes the components in decreasing order of size, and places each element in turn into each partially filled bin that it fits in, and into another bin, branching on these different alternatives. This results in a problem space bounded by n!, where n is the number of elements, but this is a very pessimistic upper bound, since many elements won't fit in the same bins as other elements. At each node of the search tree, Martello and Tooth compute the first-fit, best-fit, and worst-fit decreasing completion of the corresponding partial arrangement[8]. A partial solution to a bin-packing problem is one where some but not all elements have already been assigned to bins. A fulfillment of an incomplete solution takes the current partially-filled bins, and assigns the staying unassigned components to bins. The worst-fit decreasing algorithm places each successive element in the partially-filled bin with the largest residual capacity that will accommodate that value. Each of these approximate arrangements is contrasted with a lower bound on the remaining solution that they call L3. The L3 bound is computed by successively relaxing the remaining subproblem by removing the smallest element, and then applying the L2 bound to each of the relaxed instances, returning the largest such lower bound. Martello and Tooth's L2 bound equals the estimated wasted-space bound described above, but they use a more complex algorithm to compute it. If the number of container used by any of the approximate completions equals the lower bound for completing the corresponding partial solution, no further find is performed below that node. If the number of bins in other approximate solution equals the lower bound on the original problem, the algorithm exit, returning that solution as optimal. The main source of efficiency of the Martello and Tooth algorithm is a method to reduce the size of the remaining subproblems

2.Online Versions

2.1First-Fit (FF)

First fit keeps all empty bins open. It places the next item in the lowest numbered bin in which the item fits. In case it doesn't fit in any bin, another bin is opened. First-Fit could not achieve better execution time as it keeps keeps all non-vacant bin active and tries to pack each item in these bins before opening a new one. In this algorithm the rule followed is: 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 canister at that point 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

2.2 Best Fit Algorithm:

Best-Fit (BF) is the best known calculation for Bin Packing issue. It is simple and behaves well in practice. Best Fit picks (among the possible bins for the Best-Fit (BF) is the best known algorithm for Bin Packing problem item) the one where the amount of free space is minimal. It picks the bin with minimal measure of free space in which it can still still hold the present component. This calculation tries to pick the fullest bin posiible with enough space every time an item is appointed. Every single vacant bin are kept open until the end. It puts the following thing in the bin whose current contents is the biggest, but should not exceed its ability.. If it does not fit in any bin, new bin is opened [9].

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

3.Offline Versions

3.1 First Fit Decreasing Algorithm:

Initially sorts the given things in non-decreasing request and then allocate bins to these items, in which they first fit, same as in first fit online algorithm

3.2 Best Fit Decreasing Algorithm:

Initially sorts the given things in non-decreasing request and then allocate bins to these items, in which they best fit, same as in best fit online algorithm[10].a list of items and unit sized bin are taken and output results,after applying these algorithms one by one

4. MapReduce Algo

The idea behind the MapReduce is that whole of the computer architecture is divided into 2 types of computers or nodes. One is the expert node and rest all are the laborer nodes. However laborer nodes are further divided into two types according to the tasks assigned to them. Some are the mapper nodes and others are the reducer nodes. First of all, the expert node is initiated/ started. The master node does the work of efficiently distributing the larger problem into smaller sub problems, within the laborer nodes. The worker nodes which take in raw data sets are the mapper nodes[11]. They take in these sets and discharge out intermediate key value pairs, which consist of data and assigned keys nodes. They take in these sets and discharge out intermediate The response of intermediate files emitted out by the mapper nodes are further locally rearranged, sorted, grouped and assed on to the reducer nodes . The reducer node collects the intermediate record and binds a particular answer for the given problem, without any redundancy. This phase is also known as the Reduce phase.

The proposed algorithm undergoes the following basic stages:

4.1 Collection of Inputs: The buffer or window collects the required items along with the bin capacity and count in which these weights are to be balanced and when the window size is full, the inputs are submitted as first parameter being the bin limit, then number of bins with the same capacity that are available i.e. bin's count and at last the list of things containing weights that are to be balanced in those bins, to the Master node of the MapReduce Framework.

4.2Mapping of Items: As the inputs are sustained into the mapper nodes, each mapper node starts its processing independently.The list of items submitted for a specific sized bin limit are extracted out and sorted in non-decreasing order. The output of this stage is the bin limit as the key, then bin's count and finally the non-decreasing ordered list of items that
are to be accommodated in those bins.

4.3 Combining The Mapped items lists: Now the above mapper phase outputs are locally combined, shuffled and sorted as indicated by their key qualities. And again they are partitioned further depending upon the number of reducers. These lists go about as inputs for the reducer nodes.

4.4 Optimal allocation phase: The reducer nodes are given these mapped things list as inputs. Reducer nodes are given the task of applying first fit decreasing algorithm. As the input items for a specific container are now sorted, this gives a straightforwardness to the reducer. The reducer nodes directly act on these sorted documents applies first fit calculation in order to obtain order to obtain nearly optimal solutions. The output of this stage is the ideal list of items obliged in the corresponding bin against which those were submitted.

4.5 Analyzing the Outputs: The output list now finally contains the following information. First is the bin capacity and then the list of optimally allocated items to that bin

5.Particle Swarm Optimization:

PSO is a population-based enhancement calculation, which could be implemented and applied easily to solve various limit improvements issue, or the issues that can be changed to the function minimization or maximization problem. The utilization of PSO for taking care of packing problem has been presented by some researchers. Liu et al. displayed developmental PSO for taking care of bin packing problem. Zhao et al. applied the discrete PSO for solving rectangular packing problem. Thapatsuwan et al. analyzed GA and PSO for solving multiple container packing problems. They concentrate on the pressing issue of compartment in the capacity or the boat lodge. Since the sizes of the storage what's more, the boat lodge are composed in adjustment to the holder sizes, the optimization problem can be defined as the discrete-valued problem and therefore, the special PSOs such as discrete PSO are applied. In this study, arbitrarily polygon-shaped pressing regions are accepted. in this way, the issues are defined as the continuous-valued optimization problems[12]. The configuration objective function is to maximize the total number of the items packed into the packing region without their cover. The aggregate number of things and the position vectors .the thing focuses are utilized as the configuration variables[12] The issue is illuminated by the first and the enhanced PSOs. In the PSO, the potential arrangements of the streamlining problem to be solved are defined as the particle position vectors. At that point, the molecule positions are redesigned by PSO upgrade rules. In the first PSO, the molecule position vector upgraded by the best position of all particles (global best position) and the local best position in previous positions of each particle (personal best position). The enhanced PSO uses, in addition to them, the second best of all particles (global second best position). The use of the global second best molecule position has been as of now introduced in. Its numerical dialogs and applications were most certainly not in this reference. Also,the stochastic use of the global second-best particle position xg2 is presented in this study.

PSO Algorithm
Total number of the particles in the swarm is fixed as N. The original PSO algorithm is shown in
Fig. 1 and summarized as follows.

1. Set $t = 0$.
2. For $i = 1, \cdots, N$, set $x_{pi}(t) = 0$.

3. Initialize the position vector $x_i(t)$ and the velocity vector $v_i(t)$ of each particle with random numbers.

4. Set $t = t + 1$.

5. For $i = 1 \cdots N$,

(a) Evaluate fitness functions $f(x_i(t))$ for each particle.

(b) If $f(x_i(t)) > f(x_i(t - 1))$, set $x_i(t) = x_i(t)$.

(c) If $f(x_i(t)) \leq f(x_i(t - 1))$, set $x_i(t) = x_i(t - 1)$.

6. Find the best particle position $x_1(t)$ among $x_i(t)$ ($i = 1, 2, \cdots, N$).

7. If $f(x_1(t)) > f(x(t - 1))$, set $x(t) = x_1(t)$.

8. If $f(x_1(t)) \leq f(x(t - 1))$, set $x(t) = x(t - 1)$.

9. For $i = 1 \cdots N$, update the velocity vector $v_i(t)$ and position vector $x_i(t)$ of each particle
according to eqs. (2) and (1), respectively.

10. Go to step 5 if $t \leq t_{max}$.

6.Grouping genetic algorithms

Grouping genetic algorithms (GGAs) are vigorous tools to optimize this well-known problem when the conventional exact arrangement strategies have a tendency to come up short[13]. Despite the fact that GGAs function admirably in combination with the issue particular heuristics, they don't make best utilization of the capacities of the multi-center processors

6.1Falkenauer's chromosome representation

GGAs work better with an exceptional encoding plot that was proposed by Falkenauer (1996) to make relevant structures of collection issues get to be qualities in chromosomes(Radcliffe & Surry,1995). With this encoding scheme, more efficient crossover, mutation, and inversion operators are portrayed to maintain the information gained by the chromosomes. The new chromosome is enlarged with a gathering part. Moreover to the genes representing the groups of each item,the new chromosome structure speaks to the gatherings[14]. To avoid the problems of Holland-style chromosomes the representation of Falkenauer is employed in our proposed algorithms

6.2 Crossover, mutation, and inversion operators of Falkenauer's chromosome

Falkenauer's chromosome administrators work with the grouping part of a chromosome.. The grouping part has more meaning than the other part of the chromosome. So as to explore the search space more effectively and find the more promising solution regions, administrators deal with the groups rather than items. If chromosomes do not allow the subsets to be exploited, at that point the GGA misses the mark and the calculation performs just a little better than a random search. Under these constraints, GGA administrators need to handle variable length chromosomes of whose group orders are irrelevant. The crossover operator of Falkenauer's chromosome works with the variable length chromosomes that represent the groups (bins).The mutation operator for Falkenauer's chromosome embeds new characteristics into the population to enhance the search space by diversification Inversion operator proposes the same solution with a different representation. In order to increase the chance for fitter containers to be selected together in crossover and mutation operations, an inversion operator should be used

6.3 Exon shuffling crossover

In addition to Falkenauer's crossover operator, we have used exon shuffling crossover (Kolkman & Stemmer,2001Rohlfshagen & Bullinaria, 2007),
In the first phase, all mutually exclusive segments are combined. In the second stage, the remaining things are utilized to assemble another container.

6.4 Fitness function

GAs require a fitness function which apportions a score to each chromosome in the current Population[15]. Thus, it can calculate how well the arrangements are coded and how well they solve the problem

IV. CONCLUSION

In this paper related to Bin Packing Problem has been carried out. Various research works have already been done on Bin Packing Problem. Here, the survey is done on five method(online method, offline method, martello and tooth ,map reduce and grouping genetic algorithms) .On this survey, it is found that the GA is good to apply for the Bin Packing Problem as it has various advantages over the other algorithm. Further ,direction include more robust strategies for bin packing problem by evolutionary algorithms and suggesting newer fitness function.

REFERENCES

1.N. Ntene, J.H. van Vuuren , "A survey and comparison of guillotine heuristics for the 2D oriented offline strip packing problem'' Discrete Optimization 6 (2009) 174_188

2.Khushbu Patel1, Mahesh Panchal2, , "One-Dimension Multi-Objective Bin Packing Problem using Memetic Algorithm'' ,*IJCSMC, Vol. 3, Issue. 2, February 2014, pg.761 – 766*

3.Nelishia Pillay1,A Study of Evolutionary Algorithm Selection Hyper-Heuristics for the One-Dimensional Bin-PackinProblem'',SACJ No. 48, June 2012

4.Anika, Dr. Deepak Garg , "Parallelizing Generalized One-Dimensional Bin Packing Problem using MapReduce" 978-1-4799-2572-8/14/$31.00_c 2014 IEEE International Advance Computing Conference (IACC)

5.Tansel Dokeroglu , Ahmet Cosar, , "Optimization of one-dimensional Bin Packing Problem with island parallel grouping genetic algorithms'',science direct,Computers & Industrial Engineering 75 (2014) 176–186

6.R.yesodha, T.Amudha, "Effectiveness of FireFly Algorithm in solving Bin Packing Problem",International Journal of Advanced Research in Computer Science and Software Engineering Volume 3, Issue 5, May 2013 ISSN: 2277 128X

7.Rina Panigrahy1, Kunal Talwar1, Lincoln Uyeda2, and Udi Wieder1
1 Microsoft Research Silicon Valley.
2 Microsoft's VMM product group

8.Richard E. Korf, "A New Algorithm for Optimal Bin Packing",2002,AAAI-02,Pg.731-736

9. Yoga Jaideep Darapuneni,"A Survey of Classical and Recent Results in Bin Packing problem"*University of Nevada, Las Vegas*

10. Brian Kroth "Bin Packing: A Survey and its Applications to Job Assignment and Machine Allocation"
2013-04-19, CS787 Project

11. Eunice L´opez-Camacho,Hugo Terashima-Mar´ın "Evolving Feature Selection for Characterizing and Solving the 1D and 2D Bin Packing Problem",2013 IEEE Congress on Evolutionary ComputationJune 20-23, Cancún, México, 978-1-4799-0454-9/13/$31.00 ©2013 IEEE

12. Anika & Dr. Deepak Garg"Parallelizing Generalized One-Dimensional Bin Packing Problem using MapReduce" 978-1-4799-2572-8/14/$31.00_c 2014 IEEE

13. Young-Bin Shin, Eisuke Kita,"Solving two-dimensional packing problem using particle swarm optimization" **19**: 241–255, 2012.

14. N. Mohamadi1"Application of Genetic Algorithm for the Bin Packing Problem with a New Representation Scheme"

15. G. S. Wang 1 F. K. Huang 2 and H. H. Lin "APPLICATION OF GENETIC ALGORITHM TO STRUCTURAL DYNAMIC PARAMETER IDENTIFICATION,August 1-6, 2004 ,Paper No. 3227