Volume 3, Issue 6, June 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Approximations Algorithms with Simulated Annealing (SA)

***Diptam Dutta, Sandeep Kumar Jha, Seikh Basir Ahammad, Dhiraj Kumar Pal**
Computer Science & Engineering
Heritage Institute of Technology, India

Abstract- Simulated annealing is a method of finding optimal values numerically. It chooses a new point, and (for optimization) all uphill points are accepted while some downhill points are accepted depending on probabilistic criteria. For certain problems, simulated annealing may be more efficient than exhaustive enumeration — provided that the goal is to find an acceptably good solution in a fixed amount of time, rather than the best possible solution. Simulated Annealing is a local search method based on local optimization. In this method each trial solution in the solution space has a cost, and the objective is to find a feasible solution of least cost. The method is iterative. In each cycle we try to move from the current trial solution S to a neighboring point S' in the solution space in an effort to find a better trial solution. Let us assume that the problem is a minimization problem. If cost(S') < cost(S), S' becomes the new trial solution; the move from S to S' is then called a downhill move. If cost(S') > cost(S), S' becomes the new trial solution with probability p = exp (-Δ/temp), where temp is a parameter known as the temperature and Δ = cost(S') - cost(S); S is retained as the trial solution with probability (1-p). Thus S' can become the new trial solution even when its cost is higher than the cost of the current trial solution S; this kind of move from S to S' is called an uphill move. This deliberate choice of an inferior trial solution with a non-zero probability helps to ensure that the procedure does not get trapped in a local minimum. By slowly reducing the temperature, the probability p is reduced in the course of the iteration as better trial solutions are found. Bin packing problem [1], [2] solves the packing of objects of different volumes into a finite number of bins of capacity V in a way that minimizes the number of bins used. The approximation algorithm is applied on Multiple Bin Packing Problem in such a way that the algorithm produces the minimum number of bin used as a result.

Keyword- One bin packing, multiple bin packing, simulated annealing, best fit problem, first fit decreasing, meta-heuristics, constraints (parameters).

1. Introduction:

. This paper describes a complementary mechanism that attempts to learn the structure of the search space over multiple runs of SA on a given problem (Best fit Problem [6]) for one bin packing as well as Multiple Bin Packing. For this, we introduced different parameters for one bin packing and also for Multiple Bin Packing. Specifically, we also introduce a mechanism that attempts to predict how (un)-promising a SA run is likely to be, based on probability distributions that are "learned" over multiple runs. The distributions, which are built at different checkpoints, each corresponding to a different value of the temperature ('temperature' is a variable which decrements it's value at each step-as SA has a great relation with physics, the variable is termed in this manner) parameter used in the procedure, approximate the cost reductions that one can expect if the SA run is continued below these temperatures.
.

II. Literature Review

A. Bin Packing Problem-

The bin packing problem asks for the minimum number *k* of identical bins of capacity C needed to store a finite collection of weights w_1, w_2, w_3, ... , w_n so that no bin has weights stored in it whose sum exceeds the bin's capacity. Traditionally the capacity C is chosen to be 1 and the weights are real numbers which lie between 0 and 1, but here, for convenience of exposition, I will consider the situation where C is a positive integer and the weights are positive integers which are less than the capacity.

B. Simulated Annealing-

Simulated Annealing (SA) is a general-purpose search procedure that generalizes iterative improvement approaches to combinatorial optimization by sometimes accepting transitions to lower quality solutions to avoid getting trapped in local minima. SA procedures have been successfully applied to a variety of combinatorial optimization problems, including Traveling Salesman Problems ,Graph Partitioning Problems , Graph Coloring Problems[20], Vehicle Routing Problems[15] , Design of Integrated Circuits, Minimum Make-span Scheduling Problems as well as other complex scheduling problems, often producing near-optimal solutions, though at the expense of intensive computational efforts. The procedures, typically requiring that the procedure be rerun (iterate) a large number of times before a near optimal solution are found. Other names of Simulated Annealing are Monte Carlo Annealing[5], Statistical Cooling[6], Probabilistic Hill Climbing[7], Stochastic Relaxation[9], Probabilistic Exchange Algorithm[8] etc.

C. Problem Definition-

The problem is categorized into two phases i.e., Phase I & Phase II

1) Phase I: The goal is to fit the different weighted objects into a single bin with the least cost function.

2) Phase II: The goal is to fit the different weighted objects into multiple bins such that minimum number of bins used.

III. Proposed Work:

A. Proposed Algorithm For Simulated Annealing:

```
Procedure SA
{
        input a trial solution S; c = cost(S); c* = infinity; freezecount = 0; initialize temp;
        initialize frzlim, sizefactor, tempfactor, minpercent, tcent;
        while ( freezecount < frzlim )
{
                changes = trials = 0;
                while ( trials < sizefactor * N )
 {      /* N is determined by the size of the problem */
                        trials = trials + 1; generate a random neighbour S' of S;
                        c' = cost(S'); Δ = c'- c;
                        if (S' is feasible and cost(S') < c* )
                        {
                S* = S'; c* = cost(S');
                        }
                        /* save best feasible solution found so far */
                        if (Δ < 0)
                        {
                                changes = changes + 1;  c = c'; S = S';
 }      /* downhill move */
                        else
{       /* possible uphill move */
                                choose a random number r in [0,1];
                                if ( r <= exp(-Δ/temp) )
{
                                        changes = changes+1; c = c'; S = S';
                                }
        }
}
                if  (changes/trials > tcent ) temp = 0.5 * temp;        /* reduce temperature quickly */
                else temp = tempfactor * temp;      /* reduce temperature slowly */
                        if ( changes/trials < minpercent ) freezecount = freezecount+1;
                else freezecount = 0;
        }
        output the final solution S*;           /* S* is a feasible solution of minimum cost */
}
```

IV. Results Analysis:

A. For One Bin packing-

B. Table 1: Stress- Testing One Bin Packing NP hard Problem with Simulated Annealing

Target (T): To reach MaxBin Size	**Running-ONE-Bin-Packing-Approximation-Problem-utilizing-SimulatedAnnealing-with-Trials(frzlim*trialLimit)**										**Differential Error: Absolute Value of (T\`-T)/T**
	Round_1: Trials =25	**Round_2: Trials = 50**	**Round_3: Trials =100**	**Round_4: Trials =200**	**Round_5: Trials = 400**	**Round_6: Trials = 800**	**Round_7: Trials =1600**	**Round_8: Trials = 3200**	**Round_9: Trials = 6400**	**Average BinSize Reached (T)**	
	frzlim =5;TrialLimt =5	**frzlim =10; TrialLimit=5**	**frzlim =20; Trial Limit**	**frzlim =40; Trial Limit**	**frzlim =80; TrialLimit=5**	**frzlim =160; TrialLimit=5**	**frzlim =320; Trial Limit**	**frzlim =640; Trial Limit**	**frzlim =1200; TrialLimit=5**		

			=5	**=5**			**=5**	**=5**			
100	95	99	82	100	72	98	100	91	99	93	0.07
300	300	300	300	299	300	300	300	300	300	300	0
900	884	897	899	900	884	900	900	900	900	896	0.004444
2700	2684	2696	2576	2698	2694	2696	2699	2700	2700	2683	0.006296
8100	7927	8063	8024	8097	8093	8100	8100	8099	8100	8067	0.004074
24300	24012	24154	24219	24187	24199	24230	24259	24279	24178	24191	0.004485
72900	70912	71861	72454	72865	72756	72888	72565	72701	72876	72431	0.006433
218700	217772	216438	215792	216754	214778	217865	21720	216820	213820	216373	0.010640

C. Analysis of Approximation Algorithm:

D. Table 2: Analysis of Approximation Algorithm

FIRST-FIT Algorithm				Analysis of Approximation Algorithms					
Trials	**MaxBin Size**	**Max Object Number**	**Minimum Number of Bin Required(OPT)**	**BFD/FFD::Bound is :: 11/9 OPT + 1 bins**	**Speciality Case :: FFD bound is tight :: 11/9 OPT + 6/9 bins**	**Modified Bin Packing (MFFD) :: 71/60 OPT +1 bins**	**Bound 1 :: MFFD is bounded by 1.18 OPT**	**Bound2 :: 1.22 OPT for FFD**	**Tight Upper Bound for FF :: 17/10 OPT bins (Recent 2013)**
Round 1	10	6	4	5.888889	5.55555555	5.7333333	4.72	4.888	6.8
Round 2	20	12	6	8.333332	7.99999999	8.0999998	7.08	7.32	10.2
Round 3	40	24	12	15.66664	15.3333333	15.199996	14.16	14.64	20.4
Round 4	80	48	24	30.33338	29.9999994	29.399992	28.32	29.28	40.8
Round 5	160	96	48	59.66666	59.3333322	57.799998	56.64	58.56	81.6
Round 6	320	192	96	118.3331	117.999997	114.59999	113.28	117.12	163.2
Round 7	640	384	192	235.6662	235.33329	228.19993	226.56	234.24	326.4
Round 8	1280	768	384	470.3332	469.999984	455.39999	453.12	468.48	652.8

E. For Multiple Bin packing:

Table 3: Resultant Data

Trials	Max Bin Size	Max Object Number	Minimum number of Bin Required (OPT)
Round 1	10	6	4
Round 2	20	12	6

Round 3	40	24	12
Round 4	80	48	24
Round 5	160	96	48
Round 6	320	192	96
Round 7	640	384	192
Round 8	1280	768	384

Table3: Work in Jan 2013 [27]

Trials	Max Bin Size	Max Object Number	Minimum number of Bin Required (OPT)
Round 1	10	6	6.8
Round 2	20	12	10.2
Round 3	40	24	20.4
Round 4	80	48	40.8
Round 5	160	96	81.6
Round 6	320	192	163.2
Round 7	640	384	326.4
Round 8	1280	768	652.8

SERIES 2
Maximum Object Number & Minimum Number of Bin Required in this proposed work.

SERIES 1
Maximum Object Number & Minimum Number of Bin in 2013

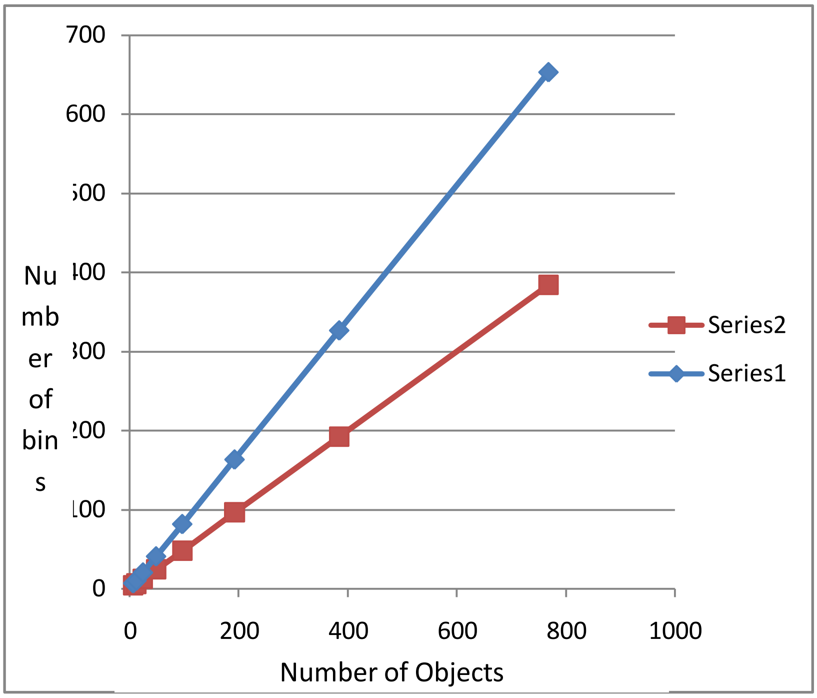

Figure 1: Comparison plot of Resultant data.

V. Conclusions:

Our work has been accomplished on a single bin of variable sizes with the implementation of simulated annealing on that particular bin with least runtime complexity. We have also accomplished our work on multiple bins of variable sizes with the implementation of simulated annealing with minimum number of bins used. A future aspect is to implement the above problems of 1bin packing as well as multiple bin packing in a 2-dimensional pattern.

References:

1. Assmann, S. and D. Johnson, D. Kleitman, J. Leung, On a dual version of the one-dimensional bin packing problem, J. Algorithms 5 (1984) 502-525.
2. Baker, B., A new proof for the first-fit decreasing bin-packing algorithm, J. Algorithms 6 (1985) 49-70.
3. Baker, B. and E. Coffman, Jr., A tight asymptotic bound for next-fit-decreasing bin packing, SIAM J. Alg. Disc. Math., 2 (1981) 147-152.

4. Bentley, J. and D. Johnson, F. Leighton, C. McGeoch, L. McGeoch, Some unexpected expected behavior results for bin packing., in Proceedings of the 16th Annual ACM Sym. on Theory of Computing, 1984, p. 279-288.
5. Brucker, P., Scheduling Algorithms, Springer-Verlag, New York, 1995.\
6. Coffman, Jr., and G. Galambos, S. Martello, and D. Vigo, Bin Packing Appoximation Algorithms: Combinatorial Analysis, in Handbook of Combinatorial Optimization, D. Du and P. Pardalos, (eds.), Kluwer, Amsterdam, 1998.
7. Coffman, Jr., and M. Garey, D. Johnson, Dynamic bin packing, SIAM J. Comput., 12 (1983) 227-258.
8. Coffman, Jr., and M. Garey, D. Johnson, Approximation Algorithms for Bin-Packing,: An updated survey, in Algorithm Design for Computer Systems Design, G. Ausiello, M. Lucertini, and P. Serafini, (eds.), Springer-Verlag, New York, 1984, 49-106.
9. Conway, R. and W. Maxwell, L. Miller, Theory of Scheduling, Addison-Wesley, Reading, 1967.
10. Courcoubetis, C. and R. Weber, Necessary and sufficient conditions for the stability of a bin packing system, J. Appl. Prob., 23 (1986) 989-999.
11. Csirik, J., The parametric behavior of the first-fit decreasing bin packing algorithm, J. Algorithms 15 (1993) 1-28.
12. Csirik, J. and J. Frenk, G. Galambos, A. Rinnooy Kan, Probabilistic analysis of algorithms for dual bin packing problems, J. Algorithms 12 (1991) 189-203.
13. Csirik, J. and D. Johnson, Bounded space on-line bin packing; best is better than first, In Proceedings, Second Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 1991, p. 309-319.
14. Fernandez del la Vega, W. and G. Lueker, Bin packing can be solved in $1 + \varepsilon$ in linear time, Combinatorica 1 (1981) 34-355.
15. Flexzar, k. and K. Hindi, New heuristics for one-dimensional bin packing, Computers and Operations Research 29 (1902) 821-839.
16. Floyd, S. and R. Karp, FFD bin packing for item sizes with distribution on [0, 1/2], Algorithmica, 6 (1991) 222-240.
17. French, S., Sequencing and Scheduling, Wiley, New York, 1982.
18. Garey, M. and R. Graham, D. Johnson, A. Yao, Resource constrained scheduling as generalized bin packing, J. Combinatorial Theory Ser. A, 21 (1976) 257-298.
19. Garey, M., and D. Johnson, Approximation algorithms for bin packing problems-A survey, in Analysis and Design of Algorithms in Combinatorial Optimization, G. Ausiello and M. Lucertini, (eds.)., Springer-Verlag, New York, 1981, p. 147-172.
20. Garey, M. and D. Johnson, A 71/60 theorem for bin packing, J. of Complexity, 1 (1985) 65-106.
21. Graham, R., Bounds for certain multiprocessing anomalies, Bell System Tech. J., 45 (1966) 1563-1581.
22. Graham, R., Bounds on multiprocessing anomalies, SIAM J. Applied Math., 17 (1969) 263-269.
23. Graham, R., Combinatorial Scheduling, in Mathematics Today, L. Steen, (Ed.), Springer-Verlag, New York, 1978, p. 183-211.
24. Hofri, M., Probabilistic Analysis of Algorithms, Springer-Verlag, New York, 1987
25. Johnson, D., Near-Optimal Bin Packing Algorithms, Doctoral Thesis, MIT, Cambridge, 1973.
26. Average-Case Analyses of First Fit and Random Fit Bin Packing, Susanne Albers, Michael Mitzenmacher,
27. Dósa G., Sgall J. (2013) First Fit bin packing: A tight analysis. To appear in STACS 2013