An Alternative Heuristics for Bin Packing Problem

Nurul Afza Hashim, Faridah Zulkipli, Siti Sarah Januri and S.Sarifah Radiah Shariff
Centre for Statistical and Decision Science Studies,
Faculty of Computer and Mathematical Sciences,
Universiti Teknologi MARA,
40450 Shah Alam, Selangor, MALAYSIA

Abstract

This study describes an alternative development of metaheuristic approaches to automate a one dimensional problem. Extensive computational testing is done to demonstrate the effectiveness of the proposed heuristic, a Variable Neighbourhood Search (VNS)-based algorithm. Several heuristics algorithms that have been used for solving the bin packing problem, Exact algorithm, Random Algorithm, First Fit Algorithm, Best Fit Algorithm, First Fit Decreasing Algorithm and Best Fit Decreasing Algorithm are incorporated into VNS and re-run for the results. The procedures have been coded in Matlab 2010 version 7.11 and the statistic was calculated with SPSS version 17.0. This study used two classes of bin packing problem instances (uniform and triplets) available in the OR Library. Results are compared to the reference solutions or the best known lower bounds where the optimum is not known. The results of the analysis showed that the combination with Best Fit Decreasing and First Fit Decreasing are remarkably effective tools for solving the bin packing problem. However, First Fit Decreasing found the existing best known or optimal solution to 8 instances with the least processing time. The success of VNS with the basic algorithms indicates that the results of this study can provide an alternative heuristic for one dimensional bin packing problem.

Keywords
Bin Packing Problem (BPP), Variable Neighbourhood Search (VNS)

1. Introduction

Bin packing problem (BPP) is a combinatorial optimization problem that has some similarities with classical knapsack problems. In the traditional knapsack problem, a set of objects of given sizes must be accommodated into a set of containers of given capacity so that the available capacity is utilized as efficiently as possible, with no additional constraints exist. The BPP was exploited in many fields such as computer science and engineering, transportation, logistics and communications. Due to its theoretical and practical relevance, several variants and richer setting were proposed with the most advanced variants of BPP are the Variable Cost and Size Bin Packing Problem (VCSBPP) (Crainic et al., 2011) and the Generalized Bin Packing Problem (GBPP) (Baldi et al., 2012). Despite that, the interest in the simpler BPP continues as many techniques have been found applied not only to solve simple but also higher dimensional problems (Martello et al., 2000; Lodi et al., 2002) on three-dimensional problems.

One-dimensional bin-packing problem is the simplest bin-packing problem. In bin packing problem, objects of different volumes must be packed into a finite number of bins of capacity C in way that minimizes the number of bins used. Even though this is a simple problem, but it is NP hard, so it is unlikely that there exists an algorithm that could solve every instance of it in polynomial time. Solution for more general realistic packing problem also depends on the effectiveness and efficiency of solution procedures for the basic problem. Choosing an objective function which is to minimize the number of bins is pointless because many different configurations, in terms of the assignment of items to bins, correspond to the same number of bins. Naturally, an objective function seeks a good solution that will always have nearly full bins. Along with maximizing bin loads, the objective function also seeks to reduce the number of bins. This study describes an alternative development of metaheuristic approaches to automate the bin packing process in one dimensional. This work has been motivated by an interest in developing modern automated algorithms that tackle this problem in more effective way as an alternative for than current existing methods. This alternative may also benefit the development of optimization techniques that can be applied to other such problems.

Among the study on solving the one-dimensional BPP are Falkenauer (1996) described a hybrid grouping genetic algorithm (HGGA), the branch-and-bound procedure of Martello and Toth (Alvim et al., 2004) used as the basic reference, Scholl et al. (1997) proposed an exact method (BISON) which makes use of several bounds, reduction procedures, heuristics, and de Carvalho (1999) and Vanderbeck (1999) presented exact algorithms based on column generation and branch-and-bound. More recently, Fleszar and Hindi (2002) proposed a new heuristics to BPP, based on the VNS metaheuristic (Hansen and Mladenovic, 1999) and using new lower bounds proposed by Fekete and Schepers (2001). A simple one dimensional BPP can be solved using the basic greedy algorithms Best Fit and First Fit algorithms, hence combining it will VNS can generate better and faster results. Hence, in this study we will be using two of the fastest heuristics for the approximate solution of BPP, the well-known First-Fit Decreasing (FFD) and Best-Fit Decreasing (BFD) greedy algorithms (Li and Chen, 2006).

Variable Neighbourhood Search (VNS) heuristic (Hansen and Mladenovic,1997), is rapidly developed in its methods and its applications. VNS is a recent metaheuristic for solving combinatorial and global optimization problems whose basic idea is systematic change of neighbourhood within a local search (Hansen and Mladenovic,1997; 1999; 2001a; 2001b; 2001c; 2003; 2008, Hansen et.al.,2010). The aim of Variable Neighbourhood Search approaches is to avoid poor local optima by systematically changing neighbourhood in order to explore an increasingly larger region of the solution space at solving combinatorial and global optimization problems. It strives to obtain the eight qualities properties of metaheuristics that state by Hansen and Mladenovic(2003) and three more items added in Hansen et.al.(2010). VNS is based on simple principle namely systematic change of neighbourhood during the search. Its principle is simple and all steps of the basic and extended schemes rely upon it. This idea has had some antecedents. It allows a change of the neighbourhood structures within this search. Besides, they are state in precise mathematical terms.

Hansen and Mladenovic(2001) claimed that VNS has proved efficient in solving the problems of several benchmarks with optimal or very close to optimal results and within moderate (or at least reasonable) computing times. Unlike many other metaheuristics, the basic schemes of VNS and its extensions are simple and require few, and sometimes no parameters. Therefore, to provide good solutions in simpler ways, VNS can lead to more efficient and sophisticated implementations. Moreover, its performance appears to be robust and the basic principles are easy to apply and very easy to use. The use of VNS gets good or better results than most other metaheuristics on many problems in much simpler way indeed.

2. Methodology

The VNS algorithm for the BPP is based on *moves*, the transfers of an item from its current bin to another or the swap of a pair of items across their respective current bins. Moves are considered valid if it does not violate the bin capacity constraint (Scholl et al., 1997; Fleszar and Hindi, 2002). VNS chooses the minimum free space weight in the current bin that allows the routine to identify the minimum free space so that it is easier to examine the items list in term of finding an item with its weight that can fit the current gap in the current bin completely. In addition, if there is no way item that can fit completely; the first item in the list that finds the gap without overlap is selected.

The overlapping test in VNS is not needed. This is because the selected item will be packed at the current gap in the current bin. The remainder (free space) will be updated after an item is packed or there is no way item in the list that can be packed at the current remain. The remainder will be updated to the next available gap in the current bin. Then, the available gap weight related to the differences between loads of bin and bin capacity. These actions give us dimension of the available gap. Hence, the current best fitting will not overlap with other items that already packed in the current bin. Hence, this it will reduce the processing time. VNS consists of two stages: Shaking stage and local search stage.

2.1 Model

We adapt the objective function from Fleszar & Hindi (2002) in which it observes that a good solution will always have nearly full bins leads naturally to an objective function that seeks configurations having this feature.

$$\max f(x) = \sum_{m=1}^{m} (l(\alpha))^2 \quad (1)$$

Where m is a number of bins in x and $l(\alpha)$ denotes a sum of sizes of items A_α assigned to bin α, i.e.,

$$l(\alpha) = \sum_{i \in A_\alpha} t_i$$

The objective function also seeks to reduce the number of bins when it maximizing bin loads. It is also worth observing that the value of the function will not change if an empty bin is added or removed.

2.2 Procedures

In developing the VNS procedure, there are three basic steps which are: shaking, local search and move. During the *shaking* phase, a random solution is generated from the current solution using the *k*th neighbourhood structure. In *local search* stage, the change in the objective function will result in the improvement, in which all possible improving moves will be specified in order to further improve he objective function values.The *k*th neighbourhood of solution $x(N_k(x))$ is defined as the set of solutions that can be obtained from x by successively performing k moves of some distinct items. k random moves are performed on x in order to find a random solution in $N_k(x)$, A random move is generated as follows (Fleszar & Hindi,2002):

1. A list of Z', is created by copying from Z the items that have not been moved so far, preserving the non-increasing item size order.
2. A random item i is selected from list Z'.
3. All possible moves involving i are determined and saved:
   a) Transfers are determined by considering all bins $\alpha=1,\ldots,m$ where m is the number of bins in x. The transfer of item i to bin α is saved only if $t_i \leq s(\alpha)$ i.e., if there is enough space in bin α for item i ($s(\alpha) = c - l(\alpha)$).
   b) Swaps are determined by considering the items in the list Z' other than i. Processing starts from the end of the list and finishes in one of two cases: either the beginning of the list is reached or item $j = Z'_q$ is too big to be exchanged with item i, i.e. $t_j > t_i + s(b_i)$. In the latter case, all items in front of q in Z' are not considered since they are not smaller than j and cannot, therefore, be swapped with i.

      Swap $i \leftrightarrow j$ is saved only if two conditions are satisfied: $t_i \neq t_j$ (swapping items of the same size does not change the solution) and $t_i \leq t_j + s(b_j)$ i.e., there is enough space for i to place it in bin b_j after removing j (the opposite condition is ensured by the stopping condition).
4. If there is no possible move for item i, then i is removed from Z' and processing is restarted from step 2. Otherwise, from all possible moves involving i, a random move is selected and performed. Item i and the counterpart item in the case of a swap are marked as moved, in order to exclude their participation in further moves in the current shaking.

This process is repeated until k moves are performed or until it is not possible to perform any move using items not marked as moved. Note that if there are no possible moves for an item, i, it is removed from Z' in step 4 above only temporarily, since after some transfers or swaps are performed, moves involving i can be found. This shaking procedure is always preserves the number of bins. Hence, a solution with a larger number of bins will not be created. However, it is not certain that every optimal solution can be obtained starting from a given solution by shaking in this manner (Fleszar and Hindi, 2002)

A local optimum is determined by a steepest descent algorithm. At each step, the procedure effectively enumerates all possible improving moves (transfers and swaps) and performs one that maximises the improvement of the objective function (1), if any. Since moves involving items from full bins do not increase the objective function, these are excluded from consideration. Thus at the beginning of each pass of the local search procedure, a list Z' of all items i for which $l(b_i)$ (c is created in a linear time, by copying from Z while preserving order. Thereafter, the following is carried out:

***Transfers*:** For each non-full bin α, items are taken from Z' starting from the end of the list. For each item i, the value of transferring it from b_i to α ($b_i \neq \alpha$) is computed and the best transfer is saved. The evaluation is stopped either when the beginning of the list is reached or when the item $j = Z'_q$ is too big to fit into bin α, i.e., when $t_j > s(\alpha)$.

In the latter case all items in Z' in front of q are not considered, since they are not smaller than j and cannot, therefore, be put in α.

***Swaps*:** Items from list Z' are considered. For each item $i = Z'_q$ the following is carried out:

1) $r = q - 1$.
2) While item $j = Z'_r$ has the same size as i, i.e., while $t_j = t_i$, decrease r (swapping items of the same size does not change the solution).
3) For each item in Z' starting from r backwards, the value of swapping i with $j = Z'_r$ is computed and the best swap is saved. Processing is terminated either when the beginning of the list is reached or when j is too big to be exchanged with i, i.e., when $t_j > t_i + s(b_i)$. Note that there is always space for item i in bin b_j , since i is always follows j in Z' ensuring that $t_i \leq t_j$.

For each of the transfers and swaps necessary, we vary the methods as per the algorithms:

(1) Exact method
(2) Random method
(3) First Fit Algorithm
(4) Best Fit Algorithm
(5) First Fit Decreasing Algorithm and,
(6) Best Fit Decreasing Algorithm

Observe that in a near-optimal solution, many bins are full. Consequently, the number of items in Z' is usually small and there are very few bins to which items can be transferred. Naturally, since the overarching goal of the local search is to reduce the number of bins, once a bin becomes empty, it is immediately removed from the solution.

For example for the First-fit algorithm, the item is processed in arbitrary order. For each item, it attempts to place the item in the first bin that can accommodate the item. If no bin is found, it opens a new bin and puts the item within the new bin. Another example is the *First Fit Decreasing* (FFD) strategy, operates by first sorting the items to be inserted in decreasing order by their sizes, and then inserting each item into the first bin in the list with sufficient remaining space.

3. Analysis and Results

In order to validate the performance of the algorithm, the computational results and discussion of VNS are discussed. This study used two classes of benchmark BPP instances available in the OR library (tpps://www.ms.ic.ac.uk/info.html). The first class (U) is uniformly distributed size items in (20, 100) to be packed into bins of size 150, while the second class (T) consists of triplets of items from (25, 50) to be packed into bins of size 100. All the instances in the T class are constructed with a known global optimal solution whereas each instance is equal to the number items divided by 3. Results are compared to the reference solutions, which are the optimal solutions or the best known lower bounds where the optimum is not known. The absolute deviation is represented by the number of bins and the relative deviations computed as the absolute deviation divided by the number of bins in the optimal solutions (or in the best-known lower bounds). The time processing is measured in seconds.

N indicates the number of instances in each test set. The "hits" column shows the number of the reference solutions achieved. The next four columns show the absolute deviation (abs dev.) and the relative deviation (rel. dev.) from the reference solution. For both, the average (av.) and the maximum (max.) values over all members of each set are displayed.

3.1 Result using Exact Algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	0	2.75	4	0.06	0.083	0.04	0.187
U250	20	0	6	8	0.06	0.08	0.03	0.047
U500	20	0	10.8	13	0.05	0.066	0.04	0.062
U1000	20	0	19.8	23	0.05	0.056	0.07	0.094
T60	20	0	1.7	3	0.09	0.15	0.02	0.047
T120	20	0	2.95	4	0.07	0.1	0.02	0.047
T249	20	0	5.3	7	0.06	0.084	0.02	0.062
T501	20	0	9.85	11	0.06	0.066	0.04	0.047
ALL	160	0	7.39	23	0.06	0.15	0.04	0.187

3.2 Using Random Algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	0	3.65	8	0.07	0.154	0.02	0.031
U250	20	0	5.25	7	0.05	0.069	0.02	0.031
U500	20	0	10.2	12	0.05	0.059	0.03	0.062
U1000	20	0	18.9	22	0.05	0.055	0.07	0.094
T60	20	0	2.65	3	0.13	0.15	0.03	0.031
T120	20	0	5.2	6	0.13	0.15	0.02	0.031
T249	20	0	10.3	12	0.12	0.145	0.03	0.062
T501	20	0	20.1	23	0.12	0.138	0.04	0.109
ALL	160	0	9.52	23	0.09	0.154	0.03	0.109

3.3 Using First Fit Algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	0	3.05	5	0.06	0.104	0.02	0.031
U250	20	0	6.4	9	0.06	0.086	0.02	0.031
U500	20	0	11.6	15	0.06	0.074	0.03	0.047
U1000	20	0	20.9	24	0.05	0.058	0.05	0.078
T60	20	0	1.65	3	0.08	0.15	0.02	0.031
T120	20	0	2.95	4	0.07	0.1	0.02	0.031
T249	20	0	5.35	7	0.06	0.084	0.02	0.031
T501	20	0	9.8	11	0.06	0.066	0.02	0.062
ALL	160	0	7.7	24	0.06	0.15	0.03	0.078

3.4 using Best Fit Algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	0	2.75	4	0.06	0.083	0.02	0.031
U250	20	0	6	8	0.06	0.08	0.02	0.031
U500	20	0	10.8	13	0.05	0.066	0.04	0.062
U1000	20	0	19.8	23	0.05	0.056	0.08	0.109
T60	20	0	1.7	3	0.09	0.15	0.02	0.031
T120	20	0	2.95	4	0.07	0.1	0.02	0.031
T249	20	0	5.35	7	0.06	0.084	0.02	0.031
T501	20	0	9.85	11	0.06	0.066	0.04	0.062
ALL	160	0	7.4	23	0.06	0.15	0.03	0.109

3.5 Using First Fit Decreasing Algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	8	0.6	1	0.01	0.022	0.02	0.031
U250	20	0	1.4	3	0.01	0.03	0.02	0.031
U500	20	0	2.65	3	0.01	0.015	0.03	0.047
U1000	20	0	4.9	7	0.01	0.018	0.06	0.078
T60	20	0	3.1	4	0.16	0.2	0.02	0.031
T120	20	0	5.8	7	0.15	0.175	0.02	0.031
T249	20	0	12	13	0.15	0.157	0.02	0.031
T501	20	0	23.1	24	0.14	0.144	0.02	0.047
ALL	160	8	6.69	24	0.08	0.2	0.03	0.078

3.6 Using Best Fit Decreasing algorithm

SET	*N*	HITS	ABS. DEV.		REL. DEV.		TIME	
			AV.	MAX.	AV.	MAX.	AV.	MAX.
U120	20	8	0.6	1	0.01	0.022	0.02	0.047
U250	20	0	1.4	3	0.01	0.03	0.03	0.047
U500	20	0	2.95	8	0.02	0.041	0.04	0.062
U1000	20	0	4.85	7	0.01	0.018	0.07	0.078
T60	20	0	3.2	4	0.16	0.2	0.02	0.047
T120	20	0	5.8	7	0.15	0.175	0.02	0.047
T249	20	0	12	13	0.15	0.157	0.03	0.047
T501	20	0	23.1	24	0.14	0.144	0.04	0.062
ALL	160	8	5.98	24	0.07	0.2	0.03	0.078

3.7 Comparison of all methods

Algorithm	HITS	ABS. DEV.		REL. DEV.		TIME	
		AV.	MAX.	AV.	MAX.	AV.	MAX.
Exact	0	7.39	23	0.06	0.15	0.04	0.19
Random	0	9.52	23	0.09	0.15	0.03	0.11
First Fit	0	7.7	24	0.06	0.15	0.03	0.08
Best Fit	0	7.4	23	0.06	0.15	0.03	0.11
First Fit Decreasing	8	6.69	24	0.08	0.2	0.03	0.08
Best Fit Decreasing	8	6.73	24	0.08	0.2	0.03	0.08

3.8 Comparison of processing time

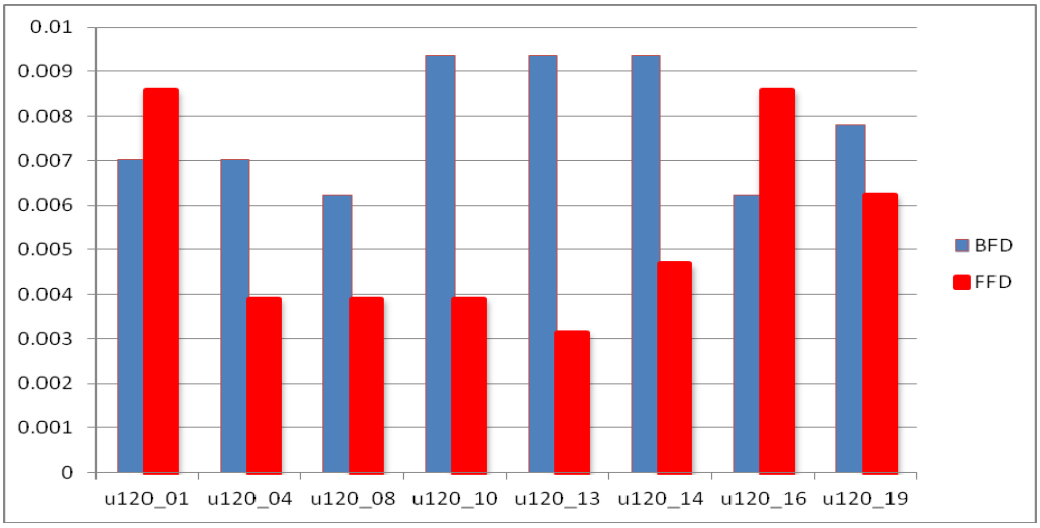

Figure 1: Comparison of Processing time between BFD and FFD optimum solutions

Computational experiments show that all the algorithms give results in reasonably short processing times (Figure 1). The BFD and FFD are remarkably effective tools for solving the bin packing problem. When applied to 160 benchmark instances, both of it found the existing best known or optimal solutions to 8 instances. Our computational experiments showed that FFD performed slightly better than BFD in term of processing time.

4. Conclusion and Recommendation

Several methods which are exact algorithm, Random Algorithm, First Fit Algorithm, Best Fit Algorithm, First Fit Decreasing Algorithm and Best Fit Decreasing Algorithm are used for the transfers and moves necessary in order to improve the objective function values. Computational experiments also show that all five algorithms give results in reasonably short processing times. The BFD and FFD are remarkably effective tools for solving the bin packing problem. However, FFD found the optimal solutions to 8 instances with the least processing time. The success of VNS algorithm indicates that the VNS metaheuristic does provide a useful framework for tackling many operational research problems.

There are still more works to be done in order to improve the performance of the algorithm, by extending the usage of the algorithm procedure for two dimensional bin packing problem. Similarly, future work could explore the possibility of designing more sophisticated VNS-based heuristics that combine creatively several ideas developed in this work. The solution strategies presented in this study could conceivably be used to solve effectively similar problem like simple assembly line balancing.

Acknowledgements

This project has been funded by Research Management Institute (RMI) Universiti Teknologi MARA, through its Research Intensive Fund (RIF) No: 600-RMI/DANA 5/3/RIF(619/2012) under the Research Interest Group, "Logistics Modelling".

References

Alvim, A.C.F., Ribeiro, C.C., Glover, F. and Aloise, D.J., A Hybrid Improvement Heuristic for the One-Dimensional Bin Packing Problem, *Journal of Heuristics,* no. 10: pp. 205 – 229, 2004.

Baldi, M.M. Crainic, T.G. Perboli, G. Tadei, R. The generalized bin packing problem, *Transportation Research Part E*, vol. 48, no. 6, pp. 1205-1220. 2012

Carvelho, J.M.V. Exact Solutions Of Bin-Packing Problems Using Column Generation And Branch And Bound, *Annals of Operations Research*, no. 86, pp. 629 – 659, 1999.

Chung-Lun Li, Zhi-Long Chen. Bin-packing problem with concave costs of bin utilization, *Naval Research Logistic,* vol. 53, no. 4, pp. 298-308, 2006.

Crainic, T.G. Perboli, G. Rei, W. Tadei, R. Efficient lower bounds and heuristics for the variable cost and size bin packing problem, *Computer & Operations Research,* vol. 38, pp. 1474-1482, 2011.

Falkenauer, E. A Hybrid Grouping Genetic Algorithm for Bin Packing, *Journal of Heuristics,* no. 2, pp. 5 – 30, 1996.

Fekete, S.P. and Schepers, J. New Classes of Fast Lower Bounds for the Bin Packing Problems, *Mathematical Programming,* no. 91, pp. 11 – 31, 2001.

Fleszar, K. and Hindi, K.S. New Heuristics for One-Dimensional Bin-Packing, *Computers and Operations Research,* no. 29, pp. 821 - 839, 2002.

Hansen, P. and Mladenović, N., Complement to a comparative analysis of heuristics for the p-median problem, *Statistics and Computing* 18 (1) : 41- 46, 2008.

Hansen, P. and Mladenovic, N., Developments of variable neighborhood search, in: *Essays and Surveys in Metaheuristics,* C. Ribeiro and P. Hansen, eds, Kluwer, Dordrecht, pp. 415-440, 2001c.

Hansen, P. and Mladenovic, N., Variable neighborhood search, in: *Handbook of Metaheuristics,* F. Glover and G. Kochenberger, eds, Kluwer, Dordrecht, pp. 145-184, 2003.

Hansen, P., and Mladenovic, N., An introduction to variable neighborhood search, in: *Metaheuristics, Advances and Trends in Local Search Paradigms for Optimization,* S. Voss et al., eds, Kluwer, Dordrecht, pp. 433-458, 1999.

Hansen, P., and Mladenovic, N., J-Means: A new local search heuristic for minimum sum-of-squares clustering. *Pattern Recognition* 34:405-413, 2001b.

Hansen, P., and Mladenovic, N., Variable neighborhood search for the p-median. *Location Sci.* 5: 207-226, 1997.

Hansen, P., and Mladenovic, N., Variable neighborhood search: Principles and applications, *Eur. J. Open Res.* 130:449-467, 2001a.
Hansen, P., Mladenović, N. and Moreno Pérez, JA., Variable neighbourhood search: Methods and applications, *Annals of Operations Research* 175 (1) : 367- 407, 2010.
Lodi, A. Martello, S. Vigo, D. Heuristic algorithms for the three-dimensional bin packing problem, *European Journal of Operational Research*, vol. 141, no. 2, pp. 410-420, 2002.
Martello, S., Lodi, A. and Vigo, D. Heuristic algorithm for the three dimensional bin packing problem, *Management Science,* vol. 141, no. 2, pp. 410 – 420, 2002.
OR Library. Retrieved on July 2013 from *http://www.ms.ic.ac.uk/info.html.*
Scholl, A. Klien, R. and Jurgens, C. BISON: A Fast Hybrid Procedure For Exactly Solving The One-Dimensional Bin Packing Problem, *Computers and Operations Research,* vol. 24, no. 7, pp. 627-45, 1997.
Vanderbeck, F. Computational Study Of A Column Generation Algorithm For Bin Packing And Cutting Stock Problems, *Maths Programming*, no. 86, pp. 565 – 594, 1999.

Biography

Nurul Afza Hashim is a final year student at UiTM, pursuing a Master degree in Quantitative Sciences. Upon graduation, she plans to pursue a PhD in the area of Financial Logistics modeling.

Faridah Zulkipli is currently a fulltime lecturer at the Centre for Statistical and Decision Science Studies at the Faculty of Computer and Mathematical Sciences of Universiti Teknologi MARA (UiTM), Shah Alam, Malaysia. She earned a Bachelor of Science degree in Decision Science from Universiti Utara Malaysia (UUM), Sintok, Malaysia, and a Master in Decision Sciences from Universiti Utara Malaysia (UUM), Sintok, Malaysia. She is a member of Research Interest Group for Logistics Modeling and has published and reviewed international conference papers. Her research interests include system dynamics, computer simulation, heuristic techniques, logistics process modeling and optimization. She is a member for Management Science/Operations Research Society of Malaysia (2008-2015), and a member of Malaysia Institute of Statistics (ISM).

Siti Sarah Januri is currently a fulltime lecturer at the Centre for Statistical and Decision Science Studies at the Faculty of Computer and Mathematical Sciences of Universiti Teknologi MARA (UiTM), Shah Alam, Malaysia. She earned a Diploma and Bachelor of Science degree in Statistics from Universiti Teknologi MARA (UiTM), Shah Alam, Malaysia, and a Master in Quantitative Sciences from UiTM, Shah Alam. She is a member of Research Interest Group for Logistics Modeling and has published and reviewed international conference papers. Her research interests include logistics process modeling, warehousing, vehicle routing problem, and optimization. She is a member for Management Science/Operations Research Society of Malaysia (2009-2015), and a member of Malaysia Institute of Statistics (ISM).

S.Sarifah Radiah Shariff is currently a fulltime senior lecturer at the Centre for Statistical and Decision Science Studies at the Faculty of Computer and Mathematical Sciences of Universiti Teknologi MARA (UiTM), Shah Alam, Malaysia. Dr Shariff earned a Bachelor of Science degree in Statistics and Mathematics from Purdue University, West Lafayette, Indiana, USA, a Master in Information Technology from UiTM, Shah Alam and PhD in Operational Research from University of Malaya, Kuala Lumpur. She is the Head of Research Interest Group for Logistics Modelling and has published and reviewed journal and conference papers. Her research interests include facility location modeling, logistics process modeling, warehousing, inventory routing, optimization and performance measurement. She is the Secretary for Management Science/Operations Research Society of Malaysia (2013-2015), a member of INFORMS, and a Lifetime Member of Mathematical Society of Malaysia (PERSAMA).