

**An Adaptive Tabu Search Approach
for 2-Dimensional Orthogonal Packing Problems**

J.M. Harwig, Major, US Army, and J.W. Barnes
Graduate Program in Operations Research and Industrial Engineering
The University of Texas at Austin

J. T. Moore
Department of Operational Sciences
Graduate School of Engineering & Management
Air Force Institute of Technology

Abstract

The research documented in this paper implements an adaptive tabu search procedure, ATS-BP, that controls the partitioning, ordering and orientation features of the solution to the general multibin two-dimensional orthogonal packing problem. The method uses a new highly effective fine-grained objective function to drive the neighborhood search procedure and embodies a very efficient dynamic move neighborhood strategy to very quickly find excellent near-optimal solutions. ATS-BP results meet or exceed all other techniques presented in the literature for a common test set of 500 problems. Relative to a defined lower bound, ATS-BP achieves a *twenty-five percent* average improvement over the previous best results.

The described problem has potential applications for shipment of military equipment, a vast majority of which are rectangular, by strategic airlift or sealift. The techniques described in the paper also have natural extensions both for two-dimensional packing problems with arbitrarily shaped items and for three-dimensional orthogonal packing problems.

An Adaptive Tabu Search Approach for 2-Dimensional Orthogonal Packing Problems

1. Introduction

The research documented in this paper implements an adaptive tabu search procedure, ATS-BP, that controls the partitioning, ordering and orientation features of the solution to the general multibin two-dimensional orthogonal packing problem. The motivation for this research originated with a desire to enhance the logistics capability of the US Armed Forces. Representative past efforts of a Research Consortium consisting of representatives of the USAF Air Mobility Command (AMC), the Air Force Institute of Technology and The University of Texas at Austin are documented in Barnes et al. (2003, 2004), Crino et al. (2004), and McKinzie and Barnes (2004). The logistics problems addressed in those past efforts are critical to the efficient movement of items, but a key piece of the logistics puzzle was still lacking. In addition to being concerned with the movement of vehicles (trucks, aircraft, and ships), practical logistics must consider the efficient and effective packing of equipment and supplies. From a military perspective, combat aircraft and warships ‘self-deploy’. Everything else must be packed. Two possible military goals could be to ship everything in the fewest possible containers (a multi-dimensional bin packing problem) or ship the highest priority materiel with the vehicle assets available (a geometric multi-dimensional multi-knapsack problem). These two problems are present both in the deployment phase of an operation and in everyday shipments at every level of the military transportation system.

The US Armed forces have automated the process for ordering and designating items for shipment. Thus, at any time, lists of items exist that must be shipped from one location to another. However, the automation that supports the loading of these items does not provide a standard strategy to obtain effective and efficient packing solutions. Current load planning is performed either by ad-hoc methods or by trial-and-error. An automated system that provides high quality packing solutions within very short implementation horizons is greatly needed. Since the tragic events of September 11th, 2001, AMC has flown on average 1000 airlift missions per month. Given the cost of each mission is nearly \$ 1 million, approaches to reduce the number of loads in a reasonably short amount of time merit further investigation. The methodology described in this paper is a first step in achieving such a system.

2. The Two-Dimensional Bin Packing Problem (2D-BPP)

The 2D-BPP is concerned with $j = 1, 2, \dots, n$ rectangular items of *width* w_j and *height* h_j . An unlimited number of identical rectangular bins of width W and height H are available. The objective is to pack all items into the minimum number of bins, in such a way that no two items overlap and the item edges are parallel to those of the bins. Items may be packed in either of their two allowable orientations. It is assumed without loss of generality that all input data are positive integers and that any item will fit into a bin in at least one of its orientations. Lodi, Martello, and Vigo (1999-1) use the notation 2D|R|F (two dimensions |rotation allowed | free packing allowed) for this problem. The 2D-BPP is known to be NP-hard. (Chu and Beasley, 1998).

3. Literature Review

When considering practical sized 2D-BPPs, classical exact methods are insufficient. Most recently published work has investigated heuristic and metaheuristic approaches. Lodi, Martello and Toth (2001, 2002) present surveys on 2D packing problems and advances in 2D packing. The first paper covers models, approximation algorithms, lower bounds and exact algorithms. The second paper discusses bounds, exact methods, heuristic approaches, and metaheuristic methods for the various classes of 2D problems. Lodi, Martello, and Vigo (1999-2) present approximation algorithms for the 2D bin packing problem (BPP).

There are several popular one-pass heuristic approaches for packing 2D rectangular items. The most basic of the 2D heuristics rely on a version of the bottom left (BL) heuristic (Baker et al. 1980). Chazelle (1983) provides the first $O(n^2)$ implementations of a one-pass Bottom-Left (BL) heuristic for the 2D-BPP, but offers no detail on how to keep the data structure updated. There are several different variations for selecting where an item is placed. Chazelle (1983) used the lowest possible BL stable location and broke ties by taking the leftmost, Jakobs (1996) began at the top right and then alternated movements, first moving as far as possible toward the bottom then as far as possible to the left until the item was BL stable. Liu and Teng (1999) offer a new method that allows for representation of *any* BL stable solution. Starting from the top right corner, each piece is pushed as far down as possible until it makes contact with another item and then as far to the left as possible. These bottom-left movements are repeated until the item can no longer be shifted further to the bottom or left. A packing pattern is represented by a permutation, π , of the items. For example $\pi = (3, 2, 4, 5, 1)$ first loads item 3, then items 2, 4, 5 and 1. This is illustrated in Figure 1a.

BL heuristics tend to create packing solutions with relatively large dead spaces, i.e., spaces not occupied by items. The $O(n^3)$ Bottom-left-fill (BLF) heuristic (Chazelle 1983) attempts to fill the gaps

created by BL. The strategy is to left-justify items at the lowest available position. Figure 1b shows the improved packing BLF obtained by using the identical π . BLF is capable of obtaining a denser packing, but requires more computation. Minor permutations of π can yield identical packings with either of the two heuristics. For example, both heuristics yield the pattern in Figure 2-1b when $\pi = (3,2,4,1,5)$.

Gilmore and Gomory (1965) implemented the first exact approach to the 2D-BPP, a column generation approach based on enumerating all subsets of items (patterns) that can fit into a single bin. Fekete and Schepers (1997-1, 1997-3) use a graph theoretic framework to enumerate feasible packing solutions with an exact tree search algorithm and present results for test problems for single bins with up to 97 items. Lodi, Martello, and Vigo (1998) analyze a simple lower bound, the sum of the item areas to be loaded divided by the container area, and determine its worst case performance. They also use new lower bounds in a branch-and-bound algorithm for problems of up to 120 items. Martello and Vigo (1998) present a two-level decomposition principle to obtain an exact solution to the 2D-BPP.

A limited number of metaheuristic approaches have been applied to the 2D-BPP. Jakobs (1996) presents a hybrid genetic algorithm (GA) that uses the BL heuristic. A weakness of many BL variants is that item sets can exist where no π will yield an optimal packing. Liu and Teng (1999) use an improved BL with a GA which apparently overcomes this problem. Lodi et al. (1998) present a tabu search (TS) algorithm for problems with *guillotine cuts* and fixed item orientation. Lodi et al. (1999-1) generalize their approach to include non-guillotine cuts and item rotations.

4. An Implementation of the Bottom Left Heuristic, pcf-BL

Prior to pcf-BL (perimeter contact function-Bottom Left) Heuristic (Harwig, 2003), there was no publicly available BL Heuristic code available. Fortunately, Chazelle (1983) offers a fairly robust representation of the data structure and includes pseudocode for finding BL stable positions. Harwig (2003) used Chazelle's method to find the set of feasible locations and Liu and Teng's technique to choose the packed location because of its capability to represent *any* physical packing solution as a permutation of letters.

Unfortunately, Chazelle's (1983) belief that there were only two key cases to be considered in the update of a BL heuristic's was incorrect. Harwig (2003) details effective and efficient ways to identify and implement the *several* required update cases, provides a method for updating the data structure so

that any permutation of items can be packed in $O(n^2)$, and presents modifications required for packing multiple bins with items that are allowed to rotate.

For the designated placement in each bin considered by the pcf-BL heuristic, a perimeter contact function, pcf, is evaluated and the placement with the maximal pcf is implemented. The pcf is motivated by Lodi et al.'s (1999-1) measure that considered the item's "touching perimeter" for each BL position in the entire bin. The pcf measure similarly rewards a newly placed item's boundary contact with the boundaries of other previously placed items or with the boundary of the container. However, the perimeter of all simple "trapped" unfilled spaces is penalized in a similar fashion. Figure 2 illustrates how the boundaries of such unfilled spaces are counted in the penalty. The newly placed item is the topmost item and the heavy lines on the boundary of the shaded empty space contribute to the penalty.

Harwig (2003) presents a performance comparison of his heuristic, the pcf-BL heuristic, with two others implemented by Lodi et al. (1999-1) on two different test sets. For details on the meticulous logic involved, readers are referred to Harwig (2003) and to the embedded documentation in the Java computer code, available upon request from the Graduate Program in Operations Research and Industrial Engineering at The University of Texas at Austin.

5. An Adaptive Tabu Search Approach to the 2D-BPP (ATS-BP)

This section presents the ATS-BP, an adaptive tabu search method for solving the 2D-BPP viewed as a partitioning and ordering (P|O) problem. An initial incumbent solution is provided by the pcf-BL heuristic. ATS-BP applies moves that transform the incumbent solution's P|O representation and the pcf-BL heuristic uses that new representation to construct the geometric packing associated with the new solution.

This section has five subsections. The first subsection presents an objective function that allows for differentiation of intermediate moves which may not change the total number of bins being used. The second subsection discusses the solution representation used by ATS-BP. The third subsection presents essential move concepts and neighborhoods implemented in this research. The fourth subsection explains a dynamic move strategy that integrates the different move sets into an ATS.

5.1 A Fine-Grained Objective Function for the BP Problem

In BP problems, the primary goal is to minimize the number of bins while ensuring that every item is packed. If a search method uses only the ‘number of bins’ as a relative measure for a set of moves, the great number of equal valued moves would thwart the search process. This subsection presents a fine-grained objective function that allows effective relative evaluation of competing moves in support of the primary goal.

Moves that increase the likelihood of reducing the number of bins should be favored. Three kinds of moves fit this definition:

- (1) Excess bin moves. An *excess bin* is a designated candidate bin for complete emptying by moving its items to other bins. Moves that reduce the total space occupied by items in an excess bin should have high relative value. When such moves yield the same amount of occupied space, the move that results in the greater number of items in the excess bin is preferred, i.e., more items with smaller average size will be easier to relocate to other bins in subsequent rearrangements.
- (2) Intra-bin moves. If two moves yield different packing arrangements of the same set of items in a bin, the one with a more compact arrangement is preferred. A “compact arrangement” is discussed below.
- (3) Inter-bin moves. Given two bins, a move yielding greater aggregation of the dead space should be preferred because the likelihood of relocating a large item from the excess bin (or other bins) increases.

The fine-grained objective function should integrate the effects of excess bin, intra-bin, and inter-bin moves. This is achieved by using a *potential energy* based approach. Li and Milenkovic (1995) used a potential energy based objective function for 2D strip packing of polygons with their Position Based Physics technique. We use a similar objective function tailored to produce the desired features of the moves described above.

In Newtonian physics, an item’s potential energy (PE) is mgd where m is the item’s mass, d is the distance of the item’s center of gravity (cg_i) relative to a specified reference plane and g is the gravitational constant. In a 2D domain, the reference planes are the x and y axes of the associated Cartesian system. All items possess equal, homogeneous density implying that item i ’s mass is $m_i = h_i w_i$ and its $cg_i = [cg_{ix}, cg_{iy}]$ is at its geometric center. In the current context, following Li and Milenkovic (1995), we define PE relative to *both* the x and y axes. Let μ and ρ be the vertical and horizontal gravitational constants, respectively. As pictured in Figure 3, $PE_x = m_i \rho d_x$ and $PE_y = m_i \mu d_y$.

As shown in Figure 4, $CG_k = [CG_{kx}, CG_{ky}]$ is the *composite* CG (center of gravity) of bin k and the items packed within bin k . CG_k is computed as

$$CG_{kx} = \frac{HW(\frac{1}{2}W) + \sum_i m_i d_{ix}}{HW + \sum_i m_i} \text{ and } CG_{ky} = \frac{HW(\frac{1}{2}H) + \sum_i m_i d_{iy}}{HW + \sum_i m_i}$$

The PE_k of a bin k is defined analogously to that of an item, with the exception that only *open bins* (bins containing one or more items) have $PE_k > 0$.

As discussed below, the fine-grained objective function of “minimizing the total PE of all bins” assures that the effects of the excess bin, intra-bin, and inter-bin moves will drive the total packing configuration toward minimizing the number of packed bins. The next three subsections discuss the interaction of objective function components associated with these moves.

5.1.1 Evaluation of Excess Bin Potential Energy

We penalize an excess bin by translating its bottom left corner to $(2.5W, 2.5H)$ before determining its PE . (The multipliers of value 2.5 assure the excess bin’s PE will be large relative to all other bins.) Since we do not attempt to improve the packing of an excess bin, each excess bin item is assigned a cg_i equal to the geometric center of the excess bin (as pictured in Figure 5). However, we do assure that the items in an excess bin may be feasibly packed in that bin.

If two moves yield the same total space occupied in the excess bin, the move that leaves more items in the excess bin is preferred because smaller items are easier to place in currently open bins. This preference is enforced by subtracting $\frac{|S_i| h_{min} w_{min}}{2HW}$ from both components, where h_{min} and w_{min} are the dimensions of the smallest item. (Lodi et al. (1999-1) used a similar term in their “filling function.”) The final equations that define the x - and y - values of the excess bin’s PE are

$$PE_{Excess_x} = [3\rho W_{Excess}](H_{Excess} W_{Excess} + \sum_{i \in Excess} h_i w_i - \frac{|S_i| h_{min} w_{min}}{2HW})$$

$$PE_{Excess_y} = [3\mu H_{Excess}](H_{Excess} W_{Excess} + \sum_{i \in Excess} h_i w_i - \frac{|S_i| h_{min} w_{min}}{2HW})$$

5.1.2 Setting the gravitational constants

Arrangement A is more “compact” than arrangement B if $PE_A < PE_B$. The values selected for μ and ρ should encourage layouts that are favorable for future placements associated with intra-bin rearrangements. It is intuitively appealing to set $\mu = \rho$. However, as illustrated in Figure 6, setting μ marginally greater than ρ can yield better dead space consolidation when packing in nearly full bins.

If $\mu = \rho$, then the bin A arrangement would be preferred over the bin B arrangement ($PE_A < PE_B$). However, the bin B arrangement is superior, i.e., any item packable in bin A is packable in bin B but the *converse* is false. Consider the PE mathematical relationship for Figure 6. The PE relationship required to favor bin B over bin A is

$$\rho(4.5(72) + 4.5(9)) + \mu(4(72) + 8.5(9)) > \rho(4.5(72) + 9.5(9)) + \mu(4(72) + 4.5(9))$$

which implies $\frac{\rho}{\mu} > 0.889$. Bins A and B are 81% full; bins that are more full would require a higher ratio. For this reason, we use $\mu=1.0$ and $\rho=0.9$.

5.1.3 Bin Dead Space Penalty

Inter-bin moves should be used to consolidate dead space and the PE is indicative of how compactly the items are placed. However, in some cases, a simple PE measure can encourage inferior configurations. With two or more partially filled bins, PE encourages equal distribution of the items. For example, in Figure 7, a simple PE favors moving an item from bin A to bin B (as shown in Figure 7b) which is contrary to our goal of dead space consolidation.

Modifying the PE measure by translating the lower left corner of a bin, X_k and Y_k , by a distance proportional to the square root of the dead space, in both the x - and y - directions, corrects this shortcoming. An additional scaling factor of 1.5 favorably weights inter-bin moves relative to intra-bin moves and ensures that the top right corner of a non-excess bin is always at lower PE than the bottom left corner of an excess bin. Thus the lower left corners of all bins (X_j, Y_j) are redefined as

$$X_j = 1.5W_j \sqrt{1 - \frac{\sum_{i \in j} h_i w_i}{H_j W_j}} \quad Y_j = 1.5H_j \sqrt{1 - \frac{\sum_{i \in j} h_i w_i}{H_j W_j}}$$

Figure 8 pictures the relative dead space translation distances of non-excess bins compared to that of excess bins. Full bins’ lower left corners are at the origin. *Near* empty bins’ lower left corners would be near $(1.5W, 1.5H)$.

After determining the new translation penalties, the new PE_k is calculated as

$$PE_k = [\rho(CG_{kx} + X_j) + \mu(CG_{ky} + Y_j)](1 + \frac{\sum_i h_i w_i}{HW})(HW)$$

Moves that reduce the contents of an excess bin also reduce the dead space penalty of the destination bin. The total PE of all bins comprises the fine-grained objective function.

Although this objective is tailored to the 2D bin packing problem, the use of PE has potential for many types of packing problems including arbitrary shaped items in both 2D and 3D for both bin packing and geometric knapsack variants.

5.2 Solution Representation for Bin Packing

In this section, we describe our order-based solution representation for the BP problem. Each bin is allocated a unique “letter.” We reserve the numbers 0 to $n-1$ for the maximal number of bins. A bin letter, fixed as the first letter in its bin-item set, is accompanied by the letters associated with the items in the bin. Each of the n items has two associated letters. One letter represents the orientation where the item’s longest axis is in the horizontal plane (“orientation h ”). The other orientation has the longest axis in the vertical plane (“orientation v ”). The letters n to $2n-1$ represent the orientation h for the n items in a specified order. Letters $2n$ through $3n-1$ represent orientation v for the items in the same order. One and only one orientation for each item must be in a bin-item set. Empty bins and absent item orientations are not shown.

For example, suppose there are 20 items, numbered 20 through 39, with items 20 through 30 packed in order in bin 0 in orientation h excepting item 25 and items 31-39 packed in order in bin 1 in orientation h excepting item 32. The corresponding solution representation would be

$$(0, 20, 21, 22, 23, 24, 45, 26, 27, 28, 29, 30) (1, 31, 52, 33, 34, 35, 36, 37, 38, 39)$$

5.3 ATS-BP moves

The four subsections of this section discuss the moves used by ATS-BP. The first subsection describes ejection chains, a fundamental concept for all move types. The second, third and fourth subsections detail the excess bin, intra-bin, and inter-bin moves. For all moves, we define a *packing position* to be a BL placement in a *non-excess* open bin.

5.3.1 Ejection Chains

Let us consider a *single* bin and the effects of a single move on that bin. All items that are not *directly relocated* as part of the move retain their current orientation and relative order regardless of any required absolute change in the packing sequence. However, a direct result of a move may be that all items no longer fit in the bin. The pure pcf-BL manages any bin overflow by opening a new bin. The ATS-BP utilizes an excess bin for this purpose. This is achieved through a slight modification of the pcf-BL.

Laguna and Glover (1997) state “An *ejection chain* is an imbedded neighborhood construction that compounds the neighborhoods of simple moves to create more complex and powerful moves.” An ejection chain begins with selecting certain elements of the current solution to change. This directed change results in at least one other element being forced to change or be “ejected” from its current state. Ejection chains have been used in tabu search techniques for several classes of problems with significant success. Approaches include Barnes and Chambers (1995) for the Job Shop Scheduling Problem, Dorndorf and Pesch (1994) for the Clustering Problem, Laguna et al. (1995) for the Multilevel Generalized Assignment Problem, Rego (1998) for Vehicle Routing Problems, Glover (1992) and Glover and Pesch (1997) for the Traveling Salesman Problem, and Gonzalez-Velarde and Laguna (2002) for the Coloring of Graphs. This research is the first to incorporate this technique to solve packing problems.

Whenever the ATS-BP moves an item for an insert move or a pair of items in a swap move, it *directly relocates* the selected items. However, the relocation may cause other items to no longer fit in their current bin. In this situation, the ATS-BP implements an ejection chain through the pcf-BL that feasibly places non-fitting items into an excess bin.

In terms of the solution representation, consider the following incumbent solution for the 20 item example of Section 5.2 (where bin 2 is the excess bin):

(0, 20, 21, 22, 23, 24, **45**, 26, 27, 28, 29, 30) (1, 31, **52**, 33, 34, 35, 36, 37) (2, 38, 39)

Swapping items 45 and 52 initially yields

(0, 20, 21, 22, 23, 24, **52**, 26, 27, 28, 29, 30) (1, 31, **45**, 33, 34, 35, 36, 37) (2, 38, 39)

However, item 37 no longer fits in bin 1 and is ejected to the excess bin 2 yielding

(0, 20, 21, 22, 23, 24, **52**, 26, 27, 28, 29, 30) (1, 31, **45**, 33, 34, 35, 36) (2, 37, 38, 39)

5.3.2 Excess Bin Moves

In general, larger items are harder to place. At each iteration, ATS-BP identifies the excess bin's *Big-Item*, i.e., the item with maximal $h_i w_i$. Excess bin moves seek to relocate the Big-Item and other relatively large items into packing positions by checking both orientations. No items smaller than a stated threshold (relative to the Big-Item) are moved. The excess bin threshold function is detailed below.

An *excess-bin-insert move* extracts a single item from the excess bin and places it in a packing position. This move is the primary move for emptying an excess bin. At each iteration, the ATS-BP attempts to insert large items into all possible packing positions.

Figure 9 shows an excess bin insert move where item 10 is inserted after item 12 in bin 0. Since item 10 fits in its current orientation, the move is feasible and items 15 and 16 are ejected into excess bin 1. In Figure 9 and later similar figures, for clarity of presentation, the item labels in the bin “pictures” will be orientation h letters regardless of the actual orientation. The solution representation will identify each item by its correct orientation letter.

Excess-bin-swap moves swap *only* the Big-Item with candidate items in packing positions and may cause ejections. (Both Big-Item orientations are considered.) Figure 10 shows an excess-bin-swap where item 15 is swapped with item 10 and item 16 is ejected.

Candidates to be swapped with the Big-Item are screened based on specified lower and upper values proportional to the Big-Item. Big-Items are not allowed to swap with items of identical size. These screening methods reduce the neighborhood size without sacrificing effectiveness. Excess-bin-swap moves consider only the Big-Item to enhance its likelihood of placement.

5.3.3 Intra-bin-insert Moves

Intra-bin insert moves relocate a single item from one packing position to another in the *same* bin. Since intra-bin insert moves make small or null changes in the fine-grained objective function and are prone to cycling, ATS-BP prohibits consecutive intra-bin-insert moves. Since the pcf-BL heuristic implements a placement in the first BL stable packing position encountered, moves that change the order of items to be placed can result in a new placement in a highly desired packing position. A special case of the intra-bin-insert move, the *trapped-hole-delay* move, attempts to make such packing positions available.

A trapped-hole-delay move directly relocates an item, that completely “seals” an unoccupied space, to a later position in the packing order. Trapped-hole-delay moves are less prone to cycling and are allowed to be performed up to 2 consecutive times. Only bins with more than 5 items are considered for this move.

Figure 11 illustrates how the *trapped-hole-delay* move works. Consider the configuration pictured in Figure 11a where the trapped hole of interest is indicated by the light cross-hash right adjacent to item 45 and below item 51. Item 51, the 27th item in the packing order, is the associated “hole sealer” that needs to be moved later in the packing order. Big-Item 99 is currently in the excess bin (pictured separately here for simplicity). A superior new configuration would place item 99 into the trapped hole sealed by item 51. Unfortunately, the dense configuration of Figure 11a causes all excess-bin-insert moves to be unfavorable. For example, the configuration pictured in Figure 11b is the result of inserting item 99 into packing position 27 thus shifting item 51 to position 28. In this case, the pcf-BL “first BL stable packing rule” causes item 99 to be placed left adjacent to item 78. This argues directly for the consideration of trapped-hole-delay moves which check all items that completely seal an unoccupied space.

Returning to the initial configuration of Figure 11a, the result of one such trapped-hole-delay move is shown in Figure 11c where item 51 is delayed one position to $k=28$ in the packing sequence, i.e., item 51 swaps with item 69. This trapped-hole-delay move, with a move value of zero, is implemented. On the next iteration, the excess-bin-insert neighborhood places item 99 at position $k=28$ as shown in Figure 11d. To complete the move, the remaining items, beginning with item 51 in position $k=29$, are placed by the pcf-BL heuristic. Previous tabu search applications have also seen the value of intermediate “zero valued moves” (Barnes and Chambers, 1995). Other trapped-hole-delay move combinations could have resulted in other appealing placements of item 99 such as in the trapped hole below item 68.

5.3.4 Inter-Bin Moves

The inter-bin moves, inter-bin-insert or inter-bin-swap, allow ATS-BP to consolidate dead space by moving at least one item from a position in one bin and placing it into another. The *inter-bin-insert move* inserts one item, considering both orientations, from one packing position to a packing position in another bin. The departure bin is repacked without the relocated item. The heuristic attempts to repack the destination bin with the inserted item. The move is not considered if the inserted item ejects

from the destination bin. However, items that follow the inserted item are allowed to eject. This move only considers items that either have $h_i < h_{Big}$ or $w_i < w_{Big}$. This is the most powerful of the non-excess bin moves since it can empty an open bin.

The *inter-bin-swap move* swaps the packing positions of two items in different bins. Ejection of either of the moved items is not allowed. Many items too large for inter-bin-insert moves are accessible to inter-bin-swap moves. Identical items are not considered for swap moves. Inter-bin-swap moves embody a search intensification context which preserves the bin orientation of the swapped objects. An example of this concept is given in Figure 12 where item 12 in bin A (in orientation h) and item 13 in bin B (in orientation v) are swapped. This yields item 13 in orientation h and item 12 in orientation v . The new solution representation replaces letters from 12 and 23 with letters 13 and 22.

5.4 The Architecture of ATS-BP

Tabu search is concerned with imposing *restrictions* to guide the search process and to allow the search to escape from local optima. *Tabu restrictions* are constraints imposed by *attributes* to prevent the reversal of recent moves. A *tabu tenure* stipulates the number of subsequent iterations that a tabu restriction remains in force. *Tabu attributes* are solution components that change with a move and are used to distinguish future solutions from recently visited solutions. ATS-BP imposes two types of tabu restrictions, Tabu Bin and Tabu Order.

Tabu Bin prohibits an item, in either orientation, from returning to an excess bin within tabu tenure iterations. The tabu bin attributes consist of a bin letter and an item's letters. *Tabu Order* prohibits an item-orientation from repeating a particular packing order associated with a bin, i.e., prohibits an item-orientation letter from returning to a specific position in a bin within tabu tenure iterations. Tabu order is used for all moves not involving excess bins. A triplet of tabu attributes is used for this restriction, (bin letter, item letter, item position). Moves satisfying an *aspiration criterion* can override a tabu restriction. ATS-BP uses a single aspiration criterion; only moves that achieve a new best global objective function value are allowed to override a tabu restriction.

The Tabu Bin and Tabu Order restrictions use the same *adaptive tabu tenure structure*. The initial tenure is set to a minimum value. If the selected move does not improve the objective function, the tabu tenure increases by 1. If the selected move improves the objective function, the tabu tenure decreases by one. The increase (decrease) is never allowed to violate a maximum (minimum) tenure. Any time the aspiration criteria is met, the tabu tenure is reset to the minimum.

Before packing any bins, a simple lower bound on the number of bins, SLB , and an *item-to-bin ratio*, IBR , are determined. Define

$$SLB = \frac{\sum_i^n h_i w_i}{HW}$$

where n is the total number of items, h_i is the height and w_i is the width of item i . H and W are the common height and width of the set of identical bins. The item to bin ratio, $IBR = n/SLB$, is an upper bound on the average number of items that can be packed in a bin and is used for bin selection.

ATS-BP has two minimum and maximum tenure settings associated with the item-to-bin ratio (IBR):

- (1) $IBR \leq 7$: minimum 7 and maximum 12.
- (2) $IBR > 7$: minimum 9 and maximum 14.

These limits reflect the fact that the higher IBR problems make greater use of cycle prone intra-bin moves.

We now discuss the dynamic move selection strategies and the logical flow of the heuristic. Key to implementing a successful move strategy is defining high and low influence moves. Influence "...measures the degree of change induced in solution structure..." (Laguna and Glover, 1997) High influence moves in ATS-BP are the excess bin moves. All other moves have low influence. If a non-ejecting excess-bin-insert move is found or if any excess bin move finds a new global best solution relative to the fine-grained objective, the best high influence move is implemented. ATS-BP allows a maximum of four consecutive low influence moves. ATS-BP rewards low influence moves that yield arrangements favorable to future high influence moves. It also limits, independently, both the number of consecutive inter-bin moves and intra-bin moves.

The dynamic move selection strategies described in this section vary the frequency of use and the size of both high and low influence move neighborhoods in order to achieve intensification/diversification effects. ATS-BP selects which move neighborhoods are implemented and changes the neighborhood sizes based on the following measures:

- (1) Number of items, n
- (2) Item-to-Bin ratio, IBR
- (3) Aspiration Criteria count (AC): number of iterations since the global best solution was found
- (4) Non-improving moves count (NIMC): number of iterations since last improving move
- (5) Non-excess bin moves count (NEBMC): number of consecutive low influence moves.
- (6) Inter-bin moves count (InterMC): number of consecutive inter-bin swaps or inserts.

- (7) Intra-bin moves count (IntraMC): number of consecutive hole-top inserts or intra-bin inserts.
- (8) For problems with $IBR < 7$, the search strategy oscillates (every 50 iterations) between the intensification and diversification modes, starting with the intensification mode. The ATS-BP uses *only* the intensification mode for problems with $IBR \geq 7$. These modes are discussed below.

Figure 13 presents the pseudo-code for ATS-BP. At initialization, the ATS-BP obtains the initial solution, identifies the initial excess bin and Big-Item, and sets counters and required parameters. There are three termination conditions: (1) achievement of a known lower bound on bins, (2) performance of a maximal number of iterations and (3) achievement of a maximum limit on computation time. The procedure then begins examining the neighborhoods presented in Section 5.3.

All excess-bin moves are evaluated at each iteration. The *excess-bin-insert neighborhood* incorporates a dynamic *functional* candidate list mechanism to screen items considered for extraction from the excess bin, i.e., only items that satisfy

$$(h_i > \varphi h_{Big} \cup w_i > \varphi w_{Big}) \cap (h_i w_i \geq f(\varphi) h_{Big} w_{Big})$$

are considered, where $f(\varphi)$ is a discrete function of φ as detailed below.

The Big-Item is *always* a candidate for excess bin extraction. The logic supporting this pair of relations is that a high value of φ will strongly restrict the candidate list of items available for insertion from the excess bin and will favor larger items *dissimilar* from the Big-Item. Small values of φ will have the opposite effect, admitting more items to this candidate list while still prohibiting the smallest items. Figure 14 details the four values of φ used in ATS-BP and the decision process that invokes each value. $\varphi \equiv 1$ is the default setting for ordinary search conditions. $\varphi \equiv 2$ is applied when the number items in the excess bin exceeds 12. This prevents the larger items from being overshadowed by numerous smaller items and hindering the progress of the search. $\varphi \equiv 0.8$ and $\varphi \equiv 0.6$ are increasingly liberal values which allow larger candidate lists. φ is set to 0.8 to combat moderate search stagnation evidenced by three simultaneous conditions: no recent aspiration criterion satisfaction, a nonimproving previous move, and a low influence previous move. φ is set to 0.6 when a larger number of iterations have seen no aspiration criterion satisfaction and the last two moves were low influence moves.

The discrete function, $f(\varphi)$, was created for the following reasons. When the search is not in the diversification mode and $\varphi \neq 2$, $f(\varphi) = 0.5\varphi$, which achieves the standard area threshold for candidate list membership. When the search is in the diversification mode and $\varphi \neq 2$, $f(\varphi)$ is set to 0.65φ to be more restrictive on small items becoming part of the candidate list for excess-bin-insert moves. When

$\varphi = 2$, $f(\varphi)$ is set to 0.375φ to relax the restriction on area, since $f(\varphi) = 0.5\varphi$ would force candidate items to be of identical area.

The *excess-bin-swap neighborhood* uses a single static equation to dynamically screen candidates for swapping positions with the Big-Item in the excess bin,

$$\frac{1}{16}h_{Big}w_{Big} \leq h_iw_i \leq 4h_{Big}w_{Big}$$

where all items, i , in any packing position are considered. The screening method is dynamic because the Big-Item may change with each iteration. Since this neighborhood is evaluated at every iteration, this restriction is present to lessen the associated computational effort without lessening the power of the search.

Having considered the high influence excess bin moves, the algorithm proceeds to the low influence moves. If the last 4 iterations have used low influence moves, the best high influence move is immediately implemented. Otherwise, if the last 2 moves have *not* been intra-bin moves (which include intra-bin-insert and trapped-hole-delay moves) and $IBR \geq 5$, the trapped-hole-delay neighborhood is evaluated. In this neighborhood, bins with more than 5 items are checked for trapped holes and the move is conducted on all items which seal trapped-holes. The IBR threshold prevents this type of move computation when associated improvements are unlikely.

Intra-bin-insert moves are examined only if the previous move was not an intra-bin move. In addition, when the search is in the diversification mode (which only occurs when $IBR < 7$), the additional restriction of the last 3 moves not being low influence moves is imposed. In evaluating the intra-bin neighborhood, candidate bins are limited to those bins with a specified amount of dead space in any candidate bin. If the best move from this neighborhood achieves the aspiration criterion, the remaining neighborhoods are not checked.

We now discuss the inter-bin neighborhoods. If the inter-bin move count (InterMC), which includes both inter-bin-insert and inter-bin-swap moves, is equal to 3, the inter-bin neighborhoods are not evaluated. This criterion controls an over application of inter-bin moves while allowing essential consolidation of dead space. Given that $InterMC < 3$, one additional condition can preclude evaluation of the inter-bin neighborhoods:

$$\text{diversification mode active} \wedge n \leq 60 \wedge InterMC = 2$$

This additional restriction is present because the smaller problems more quickly achieve over application of this move type.

When the inter-bin neighborhoods are evaluated, inter-bin-insert moves are considered first. Inter-bin-insert moves consolidate dead space possibly creating adequate space for relocation of the Big-Item in the excess bin. When $NEBMC < 3$, candidate moves are required to create dead space in the departure bin that is at least 50% of the area of the Big-Item. When $NEBMC = 3$, candidate moves are required to create dead space in the departure bin that are greater than or equal to the area of the Big-Item. When $AC > 10$ and $NEBMC < 2$, the dead space requirement is no longer enforced with the goal of enabling a short term diversification. If the inter-bin-insert neighborhood finds a new global best solution, the related move is implemented without further investigation. Failing a new global best solution, the algorithm continues with the evaluation of the inter-bin-swap neighborhood.

The number of candidate moves for the inter-bin-swap neighborhood depends on $NEBMC$, AC , and whether the search is in the diversification mode. The *combined dead space* of candidate bin pairs, (j,k) , must exceed a defined proportion of the Big-Item area. During intensification and when the $NEBMC \geq 1$, candidate moves must create a dead space area in one of the bins exceeding the size of the Big Item. For problems with high IBR , this is easily achieved. For lower IBR problems, this equation greatly reduces the candidates that must be examined.

If the search is in the diversification mode and $NEBMC = 0$:

- (1) the maximum dead space constraint is not applied
- (2) candidate moves are required to have combined dead space that is at least 50% of the area of the Big-Item.
- (3) When $AC > 10$, the combined dead space requirement is no longer enforced with the goal of enabling a short term diversification.

After consideration of all possible moves, the “best move” is implemented, the tabu structures are updated, and the tabu tenure and all counters are adjusted. If required, a new excess bin and/or Big-Item is determined. The ATS-BP program continues iterations until a stopping criterion is reached.

One final comment about the ATS-BP structure involves an efficient bin data structure that appears to have been overlooked in previous research. The pcf-BL heuristic is an efficient 2D packing heuristic requiring $O(m^2)$ operations for each move evaluated, where m is the number of items that are in bins to be repacked. Since the low influence moves have up to $O(n^2)$ combinations, the number of low influence moves per iteration grows rapidly with problem size. Techniques to reduce the number of moves considered will enhance computational efficiency. In addition to the basic screening criteria detailed in the previous section, this research implements two novel approaches to improve efficiency

that can be adapted to any direct search method using an order-based heuristic applied to multidimensional bin packing.

Dowland (1996) states that order based heuristics require a complete repacking of the container at each move. This is true only in a very rare worst case scenario. The ATS-BP saves the bin data structure as each item is packed so that future moves do not have to completely repack a bin. The bins are only repacked from the point of change. This is achieved by storing the “Bin History” for each item as it was packed (A numerical Item Data list is also stored that saves where each item is packed). The *Bin History* saves all crucial arrays and data needed to continue the pcf-BL from any point in the packing sequence. Whenever a new move is accepted, the bin history for all accepted items is updated. Whenever an item is inserted into a sequence, the ATS-BP identifies its immediate predecessor and retrieves the Bin History for the item. The expected computational savings associated with this technique is 50%.

6. Application of ATS-BP to Benchmark Test Sets

This section presents the ATS-BP results for the Berkey and Wang (1987) and Martello and Vigo (1998) test sets, compares those results with those of Lodi et al. (1999-1), and provides pictorial solution representation of selected instances. The test sets comprise ten different bin packing problem classes. The first six classes are from the Berkey and Wang (1987) test set. The last four are those proposed by Martello and Vigo (1998). Each class considers five values of n : 20, 40, 60, 80, and 100. Each n has 10 instances for a total of 500 problems.

The six classes of the Berkey and Wang test set are described below:

Class I: w_j and h_j uniformly random in $[1,10]$, $W=H=10$;

Class II: w_j and h_j uniformly random in $[1,10]$, $W=H=30$;

Class III: w_j and h_j uniformly random in $[1,35]$, $W=H=40$;

Class IV: w_j and h_j uniformly random in $[1,35]$, $W=H=100$;

Class V: w_j and h_j uniformly random in $[1,100]$, $W=H=100$;

Class VI: w_j and h_j uniformly random in $[1,100]$, $W=H=300$;

The Martello and Vigo test set (Lodi et al. 1999-1) was constructed to consider “... a more realistic situation ...” where not all items are generated on the same interval. Four *types* of items are combined in varying proportions as described below:

Type 1: w_j uniformly random in $[2/3W, W]$ and h_j uniformly random in $[1,1/2H]$

Type 2: w_j uniformly random in $[1, 1/2W]$ and h_j uniformly random in $[2/3H,H]$

Type 3: w_j uniformly random in $[1/2W, W]$ and h_j uniformly random in $[1/2H,H]$

Type 4 w_j uniformly random in $[1, 1/2W]$ and h_j uniformly random in $[1, 1/2H]$

These item types created the following four problem classes:

Class VII: Type 1 with probability 70%, Type 2, 3, 4 with probability 10% each.

Class VIII: Type 2 with probability 70%, Type 1, 3, 4 with probability 10% each.

Class IX: Type 3 with probability 70%, Type 1, 2, 4 with probability 10% each.

Class X: Type 4 with probability 70%, Type 1, 2, 3 with probability 10% each.

ATS-BP results detailing the number of bins used and solution times are provided in Harwig (2003) for each of the 500 instances. Unfortunately, the *only* comparable results available for the previous best technique are the average ratios for each 10 problem subset as reported for Lodi et al.'s (1999-1) TS-TP method.

Tables 1 and 2 detail the comparative average results for the ATS-BP and for Lodi et al.'s (1999-1) TS-TP method. The results in Table 1 and 2 are given in terms of an *average* ratio across the 10 problem instances in each subclass where the numerator is the number of bins used by the heuristic and the denominator is the improved lower bound (LB) on the required number of bins from Dell'Amico, Martello, and Vigo (2002). The consolidated results (where all problems are weighted equally) are presented for each test set in the row labeled "Test Set Average" in Tables 1 and 2. Overall, the ATS-BP achieved an average ratio of 1.045. This improvement of 0.015, compared to the TS-TP result of 1.060 reported by Lodi et al. (1999-1), is a *25% reduction* in the previous best average result.

The largest improvements occurred in the three test sets with the highest IBR. Classes II, IV, and VI had IBR's that ranged from a minimum of 20 to a maximum of 40. For these three classes, the ATS-BP found results within 2.1% of the lower bound, while TS-TP was within 5.7% of the lower bound. Part of this difference is explained by the TS-TP's inability to improve an initial solution to a single bin problem, and its limited search neighborhood when there are only a few bins.

ATS-BP also performed well on data sets with low IBR. ATP-BP was within 5.5% of the lower bound while TP-TS was within 6.1% of the lower bound. Classes I, III, V, VII, VIII, IX, X have IBR values ranging from 2 to 10. However, of those classes only Class X had problems where the IBR exceeded 5.

To illustrate the quality of the results obtained from the ATS-BP, three specific packing solutions are now provided. The boundaries of each container pictured are represented by a thick black line. Unfilled space *within* the container is also given in black. These figures were created with the software described in Rippert (2003). Figures 15 through 17 demonstrate the quality of solutions that may be

obtained when explicit control of partitioning, ordering and orientation is conducted within an adaptive tabu search methodology.

ATS-BP did take longer to find the answer to some of the problems that TS-TP discovered rather quickly. This is not surprising. Methods that sort by size tend to perform well in many cases. If a 'sorted' good solution exists, these methods quickly find it. In general there are $m!2^m$ orderings for a set of m items to be ordered for packing in a single bin. Of these, TS-TP considers only one ordering. However, in many instances that one ordering is all that is required.

7. Conclusions and Recommendations for Future Work

This paper detailed ATS-BP, an adaptive tabu search method for bin packing which makes extensive use of a *new* one-pass bottom-left packing heuristic, pcf-BL. The results of ATS-BP are *significantly* superior to the results provided by the previous best method. ATS-BP's results met or outperformed the previous best technique on all 50 subsets from the Berkey and Wang (1987) and Martello and Vigo (1998) test sets. Overall, ATS-BP reduced the gap with the lower bound by 25%.

Not apparent in Tables 1 and 2 is the ability of ATS-BP, with its fine grained objective function, to discriminate between two or more solutions that use the same number of bins. ATS-BP considers both how full the bins are packed and how compact the items are packed within the bins. Thus, ATS-BP is readily adaptable to practical considerations such as when bins are scarce, a characteristic of the 2D orthogonal knapsack version of this problem.

There are several insights for future developments that may be observed in the comparison summarized in Tables 1 and 2: (1) ATS-BP could possibly benefit from a move neighborhood similar to that of TS-TP that sorts and repacks, (2) ATS-BP might also benefit from moves that use more aggressive $O(n^3)$ heuristics in association with the pcf-BL heuristic and (3) a data structure which saves the results of moves that do not change, instead of recomputing such results, could greatly accelerate ATS-BP execution.

Adapting this approach to the U.S. Air Force's air loading problem (ALP) promises immediate rewards. If an ALP version of ATS-BP reduces the number of loads by 1%, it can save the U.S. Military nearly \$10 million per month. Based on the ATS-BP's performance on the 500 bench mark test problems presented in this research, the actual savings could be much greater.

References

- Baker, B.S., E.G. Coffman Jr., R.L. Rivest, "Orthogonal packing in two dimensions", *Siam Journal on Computing* 9, 1980, p846-855.
- Barnes, J.Wesley and John .B. Chambers. "Solving the job shop scheduling problem using tabu search". *IIETransactions*, 27, 1995, p257-263.
- Barnes, J.W., S. Dokov, B. Dimova, and A. Solomon, "A Theory of Elementary Landscapes," *Applied Mathematics Letters*, 16, 2003.
- Barnes, J.W., V. Wiley, J. Moore, and D. Ryer, "Solving the Aerial Fleet Refueling Problem using Group Theoretic Tabu Search," to appear *Mathematical and Computer Modeling* , 2004
- Berkey, J.O. and P.Y. Wang, "Two Dimensional Finite Bin Packing Algorithms" *Journal of the Operational Research Society* 38, 1987, p.423-429.
- Chazelle, B., "The Bottom-Left Bin-Packing Heuristic: An Efficient Implementation" *IEEE Transactions on Computers*. c-32: 8, 1983 p.697-707.
- Chu, P.C. and J.E. Beasley "A Genetic Algorithm for the Multidimensional Knapsack Problem", *Journal of Heuristics*, 1998, p.63-86.
- Crino, J., J. Moore, J.W. Barnes, and W. Nanry,"Solving The Military Theater Distribution Vehicle Routing and Scheduling Problem Using Group Theoretic Tabu Search", to appear *Mathematical and Computer Modeling* , 2004
- Dorndorf, Ul. and E. Pesch, "Fast Clustering Algorithms", *ORSA Journal on Computing* 6, 1994, p.141-153.
- Dowsland, K. "Simple Tabu Thresholding and the Pallet Loading Problem" *Meta-Heuristics Theory and Application* edited by Ibrahim H. Osman and James P. Kelly, 1996, p.381-405.
- Fekete, S., and J. Schepers "On more-dimensional packing I: Modeling", Technical paper ZPR97-288, Mathematisches Institut, Universität zu Köln, 1997.
- Fekete, S., and J. Schepers "On more-dimensional packing I: Exact Algorithms", Technical paper ZPR97-290, Mathematisches Institut, Universität zu Köln, 1997.
- Gilmore, P.C. and Gomory R.E. "Multistage cutting problems in two or more dimensions" *Operations Research* 12, 1965, p.94-119.
- Glover, F. and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.
- Glover, F., "New ejection chain and alternating path methods for traveling salesman problems". In O. Balci, R. Sharda, and S. Zenios, editors, *Computer Science and Operations Research: New Developments in Their Interfaces*, Pergamon, Oxford, 1992. p.491-507.
- González-Velarde, J. and M. Laguna 'Tabu Search with Simple Ejection Chains for Coloring Graphs' Technical Report Leeds School of Business, University of Colorado, Boulder, CO, 2002.
- Harwig, J. M.. *An Adaptive Tabu Search Approach to Cutting and Packing Problems*, Ph.D. Dissertation, The University of Texas at Austin, 2003.
- Jakobs, S., "On genetic algorithms for the packing of polygons", *European Journal of Operational Research* 88, 1996, p.165-181.

- Laguna, M., J. P. Kelly, J. L. González Velarde and F. Glover "Tabu Search for the Multilevel Generalized Assignment Problem" *European Journal of Operational Research*, vol. 82, 1995 pp. 176-189.
- Liu, D. and H. Teng "An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles", *European Journal of Operational Research* 112, 1999, p.413-420.
- Lodi, A., S. Martello, and D. Vigo, "Heuristic And Metaheuristic Approaches For A Class Of Two-Dimensional Bin Packing Problems", *INFORMS Journal On Computing* 11, 1999, p.345-357.
- Lodi, A., S. Martello and D. Vigo, "Approximation algorithms for the oriented two-dimensional bin packing problem", *European Journal of Operational Research* 112, 1999, p.158-166.
- Lodi, A., S. Martello, And D. Vigo, "Neighborhood Search Algorithm for the Guillotine Non-Oriented Two-dimensional Bin Packing Problem", in *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I.H. Osman, and C. Roucairol, (eds.) Kluwer Academic Publishers, Boston, 1998, p.125-139.
- Li, Z. and V. Milenkovic, "Compaction and Separation Algorithms for Non-Convex Polygons and Their Applications" *European Journal of Operational Research* 84, 1995, p.539-561.
- Martello, S. and D.Vigo, "Exact Solution of the Two-dimensional Finite Bin Packing Problem", *Management Science* 44, 1998 p.388-399.
- McKinzie, M. and J.W. Barnes, "A Review of Strategic Mobility Models Supporting the Defense Transportation System," to appear in *Mathematical and Computer Modeling*, 2004.
- Pesch, E. and F. Glover, "TSP Ejection Chains", *Discrete Applied Mathematics* 76, 1997, p.165-181.
- Rego, C., "A Subpath Ejection Method for the Vehicle Routing Problem", *Management Science* 44:10, 1998, p.1447-1459.
- Rippert, T Input Interface and Graphical Representation for 2D Multiple Bin Packing Problem, Master's Report, The University of Texas at Austin, 2003.
- "U.S. Transportation command tackles historic logistics feat - force rotations in and out of Southwest Asia." USATRANSCOM News Service, Scott Air Force Base Ill, Release Number 040127-1, January 27 2004.

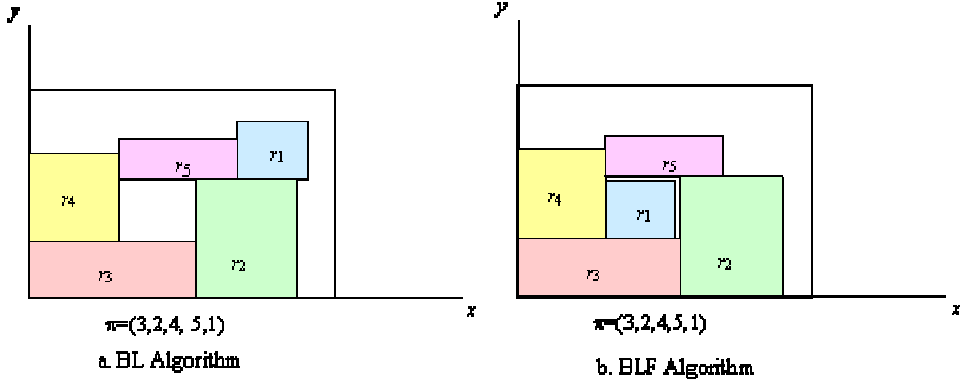


Figure 1 - Examples of the BL and BLF Algorithms

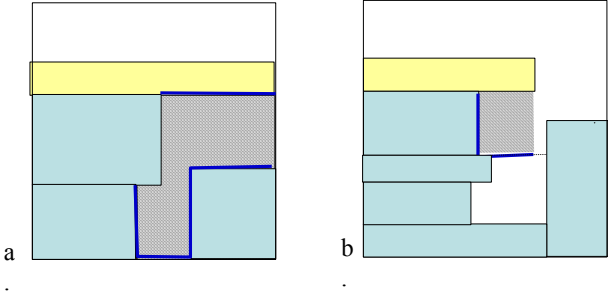


Figure 2a. trapped space penalty b. "shaded" space penalty

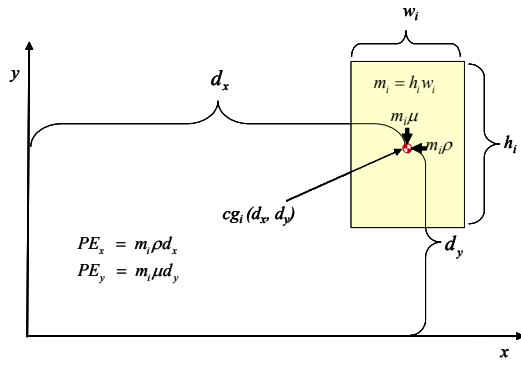


Figure 3 Determining the PE of item i

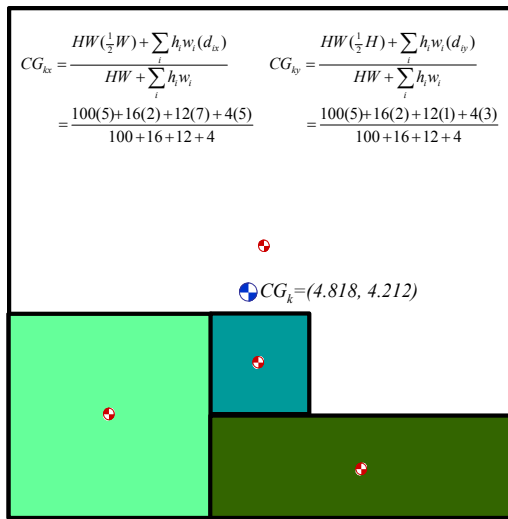


Figure 4 Determining CG_k

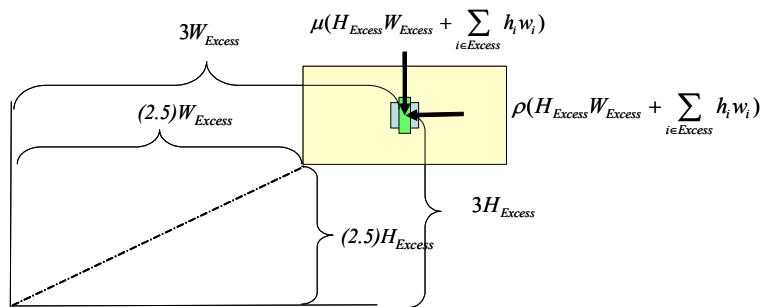


Figure 5 Excess bin PE evaluation

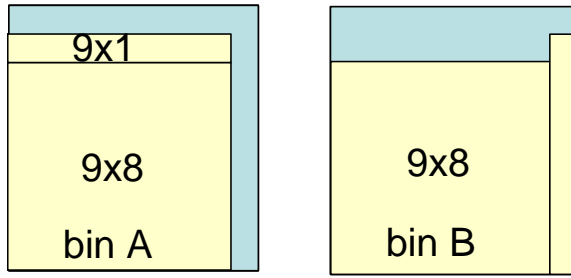


Figure 6 Adjusting μ and ρ .

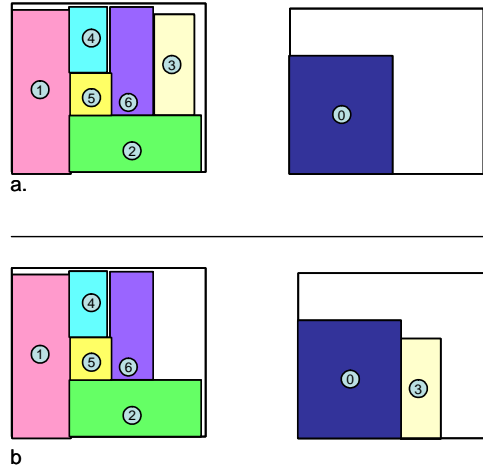


Figure 7 Simple *PE* equal distribution preference

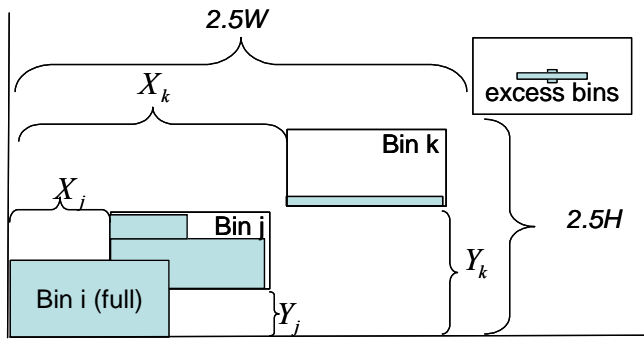


Figure 8 Translation of lower left corner based on dead space

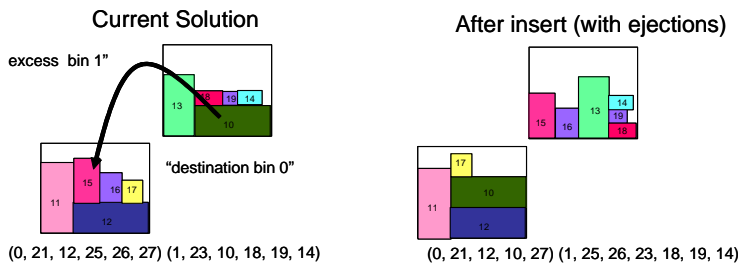


Figure 9 Excess-bin-insert with ejections

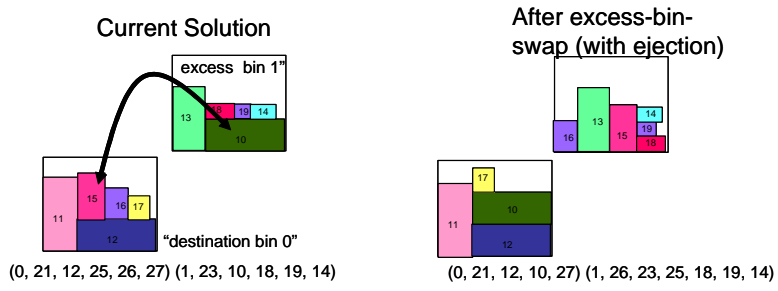


Figure 10 Excess-bin-swap with ejection.

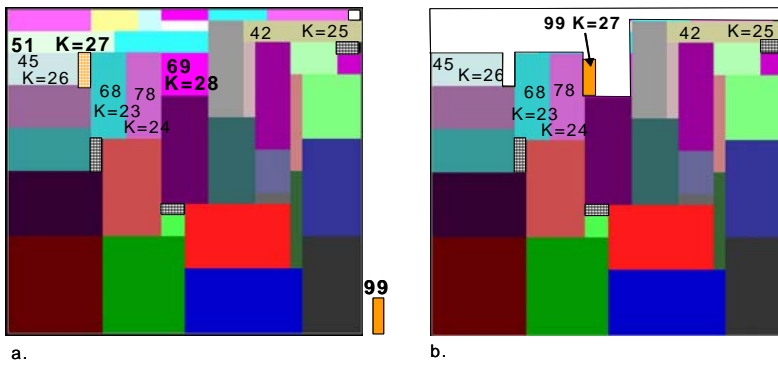


Figure 11a, b Trapped-hole-delay insert example

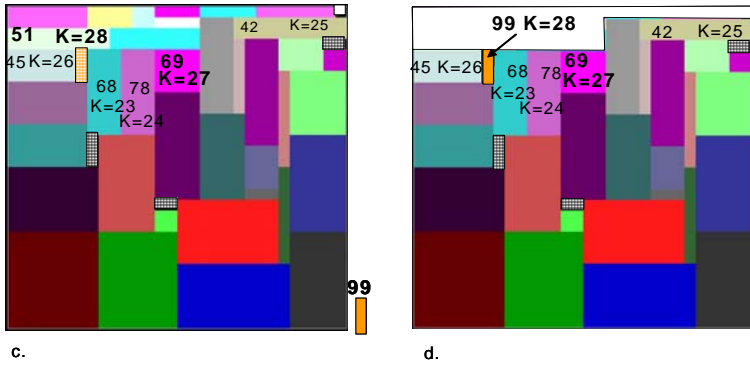


Figure 11c, d Trapped-hole-delay insert example

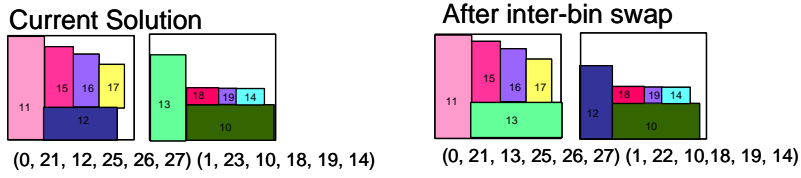


Figure 12 Inter-bin-swap move

```

Obtain initial solution, initialize counters, and other parameters
Identify excess bin & Big-Item
While termination conditions are not satisfied, do
{
  // Find best move
  Examine excess bin moves, record BEST_MOVE
  // (Excess-bin-insert neighborhood restrictions determined by AC, NIMC, NEBMC, |S|)
  // (Excess-bin-swap neighborhood restrictions are not dynamic )
  If insert non-ejection move found or aspiration satisfied, exit to FOUND BEST MOVE
  Else
  If NEBMC < 4
  { // Trapped-hole-delay neighborhood
    If IntraMC < 2  $\cap$  IBR  $\geq$  5
    { Examine trapped-hole-delay moves for bins that have > 5 items, update BEST_MOVE
      If aspiration satisfied take best trapped-hole-delay move, exit to FOUND BEST MOVE
    } Else
    // Intra-bin-insert neighborhood
    If IntraMC < 1
    { If (diversification  $\cap$  NEBMC < 3)  $\cup$  (In intensification mode)
      { Examine intra-bin moves, update BEST_MOVE
        //(Neighborhood definition restrictions determined by NIMC, AC, **)
        If aspiration satisfied take best intra-bin move, exit to FOUND BEST MOVE
      } Else
      }
    // Inter-bin neighborhoods
    If (InterMC = 3)  $\cup$  (diversification  $\cap$  n $\leq$ 60  $\cap$  InterMC = 2), exit to FOUND BEST MOVE
    Else
    { Examine inter-bin insert moves, update BEST_MOVE
      // (Inter-bin insert neighborhood definition restrictions determined by NEBMC, AC**)
      If aspiration satisfied take best inter-bin insert move, exit to FOUND BEST MOVE
      Examine inter-bin swap moves, update BEST_MOVE
      // (Inter-bin swap neighborhood definition restrictions determined by NEBMC, AC**)
      If aspire take best inter-bin swap move, exit to FOUND BEST MOVE
    } End Else
  } End If NEBMC < 4
  FOUND BEST MOVE: Implement BEST_MOVE
  Update solution, tabu tenure, cntrs, store best solution found, new excess bin and Big-Item
} End while termination conditions

** IBR < 7 varies neighborhood size for diversification setting: ITERS Mod (100)  $\geq$  50.
IBR  $\geq$  7 remain in the intensification settings at all times

```

Figure 13 ATS-BP pseudo code

```

 $\phi = 1$  // Default value
If ( $|S| > 12$ )  $\phi = 2$ 
Else If ( $(2 \leq AC \leq 7) \cap (1 \leq NIMC) \cap (1 \leq NEBMC)$ )  $\phi = 0.8$ 
Else If ( $(7 < AC \leq 35) \cap (2 \leq NEBMC)$ )  $\phi = 0.6$ 

```

Figure 14 Excess-Bin-Insert ϕ

BERKEY AND WANG TEST SET(1987) (2BP R F)						
Lodi, Martello, and Vigo (1999)						
Class	<i>n</i>	TP TS LB	average time	ATS-BP LB	average time	time to match
I	20	1.05	18.00	1.03	0.13	0.13
	40	1.04	30.02	1.04	4.11	4.11
	60	1.04	34.00	1.04	4.27	4.27
	80	1.06	48.08	1.06	0.97	0.97
	100	1.03	47.92	1.03	1.20	1.20
	Average		1.044	35.60	1.040	2.14
II	20	1.00	0.01	1.00	0.13	0.13
	40	1.10	0.01	1.00	1.46	0.21
	60	1.00	0.01	1.00	1.14	1.14
	80	1.03	6.10	1.00	22.61	2.76
	100	1.00	0.01	1.00	4.25	4.25
	Average		1.026	1.23	1.000	5.92
III	20	1.06	18.00	1.02	11.35	0.17
	40	1.09	42.17	1.09	0.38	0.38
	60	1.08	54.15	1.08	64.85	64.85
	80	1.07	60.07	1.07	9.94	9.94
	100	1.07	60.18	1.06	68.05	0.98
	Average		1.074	46.91	1.064	30.91
IV	20	1.00	0.01	1.00	0.14	0.14
	40	1.00	0.01	1.00	0.25	0.25
	60	1.10	0.09	1.05	28.80	0.31
	80	1.07	12.00	1.07	36.16	36.17
	100	1.03	6.00	1.03	9.19	9.19
	Average		1.040	3.62	1.030	14.91
V	20	1.04	12.01	1.04	1.31	1.31
	40	1.07	42.00	1.06	41.21	18.72
	60	1.06	45.23	1.06	74.59	74.59
	80	1.07	54.14	1.06	143.01	14.63
	100	1.07	60.12	1.05	196.23	1.58
	Average		1.062	42.70	1.054	91.27
VI	20	1.00	0.01	1.00	0.17	0.17
	40	1.40	0.03	1.10	5.97	0.27
	60	1.05	0.05	1.00	25.92	0.32
	80	1.00	0.01	1.00	0.42	0.42
	100	1.07	12.00	1.07	7.64	7.64
	Average		1.104	2.42	1.034	8.02
Test Set Average		1.058	22.081	1.037	25.528	8.707

Table 1 Berkey and Wang test set consolidated results

MARTELLO AND VIGO TEST SET(1998) (2BP R F)						
Lodi, Martello, and Vigo (1999)						
Class	<i>n</i>	TP TS LB	average time	ATS-BP LB	average time	time to match
VII	20	1.11	30.00	1.11	0.77	0.77
	40	1.08	48.06	1.07	99.28	36.93
	60	1.06	59.45	1.04	613.98	115.98
	80	1.10	60.12	1.08	259.78	54.06
	100	1.08	60.36	1.07	231.66	30.08
	Average		1.086	51.60	1.074	241.09
VIII	20	1.10	30.01	1.10	1.34	1.34
	40	1.10	54.22	1.08	77.91	0.28
	60	1.07	56.17	1.07	51.74	51.74
	80	1.08	60.11	1.07	529.40	10.69
	100	1.09	60.14	1.08	232.88	0.72
	Average		1.088	52.13	1.080	178.65
IX	20	1.00	0.06	1.00	0.26	0.26
	40	1.01	18.85	1.01	1.25	1.25
	60	1.01	18.03	1.01	2.54	2.54
	80	1.01	30.51	1.01	1.65	0.65
	100	1.01	36.86	1.01	52.69	1.39
	Average		1.008	20.86	1.008	11.68
X	20	1.12	6.01	1.12	0.36	0.36
	40	1.06	24.01	1.06	3.93	3.88
	60	1.06	30.44	1.06	235.17	235.17
	80	1.05	39.04	1.05	78.27	78.27
	100	1.05	43.38	1.04	269.68	12.25
	Average		1.068	28.58	1.066	117.48
Test Set Average		1.063	38.292	1.057	137.227	31.931
Grand Average		1.060	28.565	1.045	70.208	17.996

Table 2 Martello and Vigo test set consolidated results

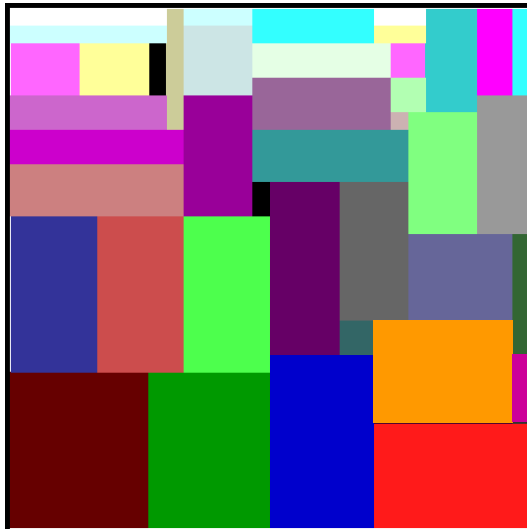


Figure 15 Berkey and Wang Class II Size 40 Instance 1. (99.44% fill)

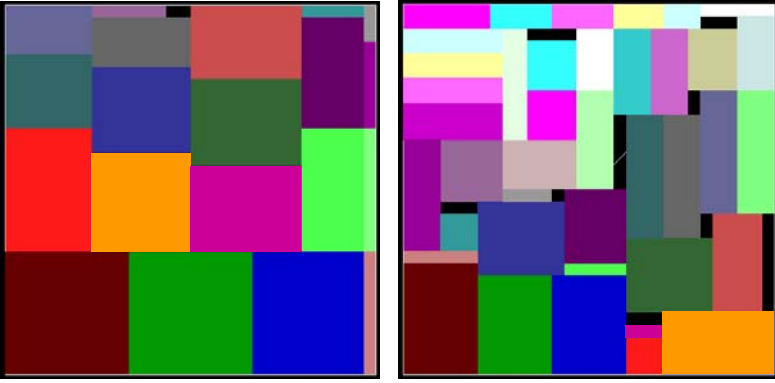


Figure 16 Berkey and Wang Class II Size 60 Instance 2. (99.77% and 96.44% fill)

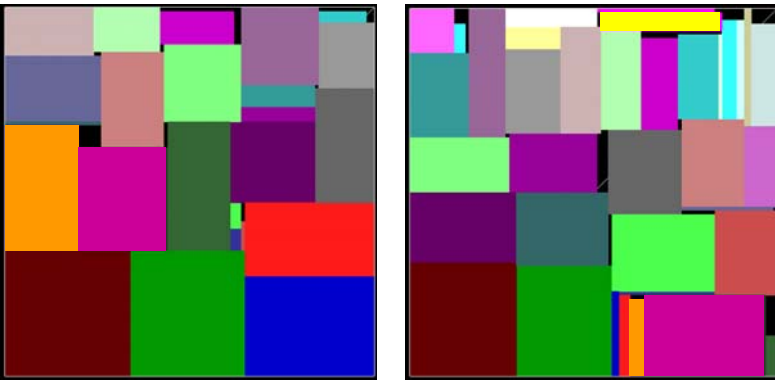


Figure 17 Berkey and Wang Class IV Size 60 Instance 4. (97.77 and 98.86% fill)