Australian Government
Department of Defence
Defence Science and
Technology Organisation

Efficient Solutions in Load Planning

*Jane Sexton, Jamie Watson and Sharon Boswell**

Maritime Operations Division
Systems Sciences Laboratory

DSTO-TN-0587

ABSTRACT

The successful execution of an amphibious operation is a function of the efficient discharge of the ORBAT to the relevant points of entry in a given timeframe. Here, we consider a sea point of entry only where the planning phase considers the loading of the ORBAT onto watercraft.
The current load planning capability is achieved through a conference and the use of a trim and stability tool. In order to make this process more efficient – both in time and accuracy – a new method was required by the ADF. As a result, DSTO developed a command decision aid, the Littoral Battle Tool Set, LBaTS.
This report describes the loading algorithm as required by the landing phase and then extends the scope to the termination phase where the ORBAT is returned to the amphibious ship. Each method will be demonstrated for a fictitious component of an ORBAT.
Finally, upon introduction of a command decision aid into the distributed architecture environment, we recommend incorporating the methods described here to enhance the planning capability of the ADF.

RELEASE LIMITATION

Approved for public release

Published by

DSTO Systems Sciences Laboratory
PO Box 1500
Edinburgh South Australia 5111   Australia

Telephone: (08) 8259 5555
Fax: (08) 8259 6567

© Commonwealth of Australia 2004

AR-013-207
April 2004

APPROVED FOR PUBLIC RELEASE

Efficient Solutions in Load Planning

Executive Summary

An amphibious operation is highly dependent on the efficient discharge of the ORBAT to its point of entry. When considering a sea point of entry, the efficiency is a function of loading the watercraft. This efficiency may make the difference in meeting the planning objective and has the potential to reduce the number of watercraft to be used to carry out the operation.

Currently, load plans for amphibious operations require a planning conference which may take place over a considerable number of days. Up until 2001, load plans were constructed by scaled drawings with cardboard shapes representing cargo items which were then arranged on the outline of a ship or watercraft (depending on application) to see if the load would fit. More recently, the load plan has been integrated with Mariner to ascertain if the load is feasible in terms of trim and stability. Alternatively, the equivalent is done where cargo units are "dragged and dropped" into Mariner and load plans devised in this way. It must be noted that Mariner is a trim and stability package rather than a load-planning tool and does not have any load optimisation capability.

Due to this labour intensive task, the ADF required a method to aid the load planning to reduce the overall time taken as well as developing near optimal solutions. The implementation is required in real-time, with a feasible solution found in the order of seconds, rather than hours. As a result, MOD, DSTO developed a command decision aid, LBaTS, with the loading routine being a component.

This report describes the method to solve the loading problem and extends the scope of the initial problem to that of a logistical load where the priority can be relaxed. The implementation of the former has been incorporated in LBaTS and the latter is for internal usage only. In addition, we propose how the two problems can be solved by one method.

The benefit to the ADF will be through its enhanced planning capability gained through the growing maturity of LBaTS and subsequent knowledge for the acquisition of a command decision aid for the distributed architecture environment such as JCSS. Whilst the requirement arose through a requirement in amphibious operations, the methods described here have equal applicability to any two-dimensional loading problem. Three-dimensional problems can be solved through further extensions to this algorithm.

Authors

Jane Sexton
Maritime Operations Division

Jane Sexton joined MOD at DSTO Pyrmont as a Research Scientist in 2001 after spending three years in postdoctoral positions at the University of Sydney and University College, ADFA. Her background is in the mathematical modelling of combustion processes, especially concentrating on spontaneous ignition. Jane works in the Amphibious and Mine Warfare Operations Group and works primarily on the development of tools to support acquisition projects and operational aids to the RAN.

Jamie Watson
Maritime Operations Division

Jamie Watson graduated from ADFA and served eleven years with the RAN, most recently as Communications Officer and ASW Officer on HMAS Canberra. In 1998, Jamie joined DSTO's Amphibious and Mine Warfare Operations Group at MOD (Sydney). Since then, he developed the prototype Amphibious Warfare Command Support System (AWOCSS). This tool has been utilised primarily as an operational aid to the RAN, as well as in the support of acquisition projects in the ADF. In addition to software prototyping, Jamie has been heavily involved in the development of algorithms to support the understanding of amphibious operations and aiding the RAN in sourcing appropriate software to aid this development. In his spare time, Jamie is pursuing more traditional technologies as a crew member of the James Craig, a Sydney based square rigged sailing ship.

Sharon Boswell
Land Operations Division

Sharon Boswell joined DSTO in 1998 as a Research Scientist after 10 years as an academic mathematician at the University of Newcastle and Curtin University of Technology, where her research interests were in the fields of discrete mathematics and operations research, particularly combinatorial optimisation. In Land Operations Division she works primarily on modelling and decision theory in support of major and minor capability development projects. Sharon was promoted to Senior Research Scientist in 2001 and has continued as part of the Land Operations Division group co-located with major sponsors in Canberra.

Contents

1. Introduction

The six distinct phases of a complete amphibious operation consist of planning, embarkation, rehearsal, movement, assault and termination (PERMAT) where the planning phase is conducted in three stages; (a) issue of the directives, (b) amphibious planning process and (c) mounting, embarkation and landing planning**.** It is with stage (c) that this paper is concerned. "Plans for mounting, assembly of assault shipping and movement of troops to embarkation points, are prepared by the planning staff of CATF[1] and CLF[2] as separate documents in the form of a movement order and embarkation and loading plans. The plan for landing of assigned forces is conducted by CLF staff."[3]

However, in the non-ideal world, planning may not appear in the first phase and may instead occur enroute (EMPRAT)[4]. One of the 'great lessons of the Falklands' was the ability to reconfigure a load whilst at sea. Due to the speed at which the forces were assembled, the prerequisite planning, logistics and training efforts were not achieved and these shortcomings had to be remedied enroute.[5] The ability to conduct EMPRAT offers an increased flexibility to any amphibious capability.

As the Army concept of manoeuvre operations in the littoral environment (MOLE) concept relies on rapid and simultaneous actions to create operational shock and surprise, the ability to rapidly deliver combat power ashore through the efficient discharge of cargo and personnel to the point of entry (POE) is imperative. The entry from air and sea (EAS) phase may insert at a beach or at an inland objective via helicopter. Here, we consider the sea POE with the method of insertion by watercraft. . The discharge rate, measured in tonnage/hour, can be increased for a fixed watercraft if the cargo units are packed effectively from a given ORBAT[6].

In this report, we will consider not only the transfer of cargo units to and from an amphibious lift ship to the watercraft, but also the initial loading of an amphibious lift ship. We refer to these two problems as tactical and logistical loading requirements, and the aim of this report is to describe the approaches in finding efficient loading solutions for the ADF in amphibious operations.

This application is somewhat different to that typically encountered in other transport modes (such as air). In amphibious operations, there is a wide distribution of shape and size of the cargo units, with rectangular/square pallets and containers more commonly transported by air or road. The shape of the pallets and containers inherently makes loading easier, and in addition, these units can also be stacked. These attributes generally

[1] Commander Amphibious Task Force
[2] Commander Landing Force
[3] ADFP 12, Chapter 5, paragraph 5.3(c).
[4] ADFP 12, Chapter 5, paragraph 5.2.
[5] p48 Speller and Tuck (2001).
[6] Order of Battle - list of units to be deployed and their vehicles and equipment

do not apply to cargo units in amphibious operations, such as wheeled vehicles. As we discuss in section 4.1.9, the proposed approach is useful not only when items vary in size and shape, but also in content, e.g. hazardous goods.

1.1 Current practice

Currently, load plans for amphibious operations require a planning conference and take place over a considerable number of days. The load plans are constructed by scaled drawings with cardboard shapes, or at best, a computer representation of cargo items which are then arranged on the outline of a ship or watercraft (depending on application) to see if the load can fit. Finally, the load plan is integrated with the trim and stability tool Mariner[7] to ascertain if the load is feasible. Alternatively, the equivalent is done where cargo units are "dragged and dropped" into Mariner and load plans devised in this way. It must be noted that Mariner is a trim and stability package rather than a load-planning tool and does not have any load optimisation capability. That is, Mariner will determine the stability of a given load but does not have any inbuilt optimisation features to suggest a best or even a good load plan.

This current method is very labour intensive and inflexible, as late changes to the ORBAT require a complete re-working of the plan. With this difficulty came the requirement for DSTO to assist in the evaluation of contending landing craft in JP2048 Phase 1A – LPA Watercraft. As a result, a simulation program was developed by Lewis (1997) which included a loading algorithm. This initial program, AMPHIB, has subsequently been built upon by MOD Pyrmont to produce the Littoral Battlespace Toolset (LBaTS) [8]. The loading algorithm now reflects the tactical loading algorithm described in this report.

At this stage, LBaTS considers the loading of the ORBAT on watercraft only as it is assumed that the ORBAT can be loaded on the amphibious ship(s). With this algorithm we can then determine the minimum number of watercraft required to unload the ship, subject to constraints imposed by tidal windows, running costs, personnel etc. A typical ORBAT could include 4-wheel drive vehicles, armoured vehicles (eg ASLAV and M133), trucks, associated trailers, static objects and personnel. Therefore the weights, lengths and widths of the items on the load plan can be widely distributed. Constraints to be imposed include no rotation or stacking for most items.

Worldwide there are many academic and industrial operations research professionals studying similar problems and it seems apposite to apply relevant techniques to the ADF problem. The techniques(s) developed will first be applied within LBaTS and could be utilised in a further upgrade of Mariner or similar tool. To provide the necessary flexibility the technique must be implemented in real-time, with a feasible solution found in the order of seconds, rather than hours. This has implications with regard to the final choice of algorithm or heuristic.

[7] Developed by Baron & Dunworth Pty Limited for the RAN.

[8] LBaTS is the follow-on of the prototype AWOCSS.

The final computed solution is to be accepted as a recommendation only as the decision rests with the experienced load planner[9] and commanders, who could modify the recommendation to arrive at the final load plan. The use of an automated tool could aid the planners in this by quickly generating further recommended plans subject to additional constraints such as fixing the location of some important items and so on.

1.2 Outline

The outline for this report begins with a review of possible packing strategies before the tactical loading algorithm is described and demonstrated. Techniques for the logistical loading requirement (which does not exist in LBaTS currently) are then explored before concluding with a discussion on the relevance and importance of this solution to the ADF. The intention is to describe the choice of the tactical loading algorithm in LBaTS and how it works and to make recommendations regarding future versions of LBaTS and the possible inclusion of a logistical loading algorithm. The defence outcome will be an enhanced planning capability through the development and acquisition of command decision aids for amphibious operations.

Under the Defence Capability Plan, information capability has been identified as a priority. The major issues in this capability goal are intelligence, surveillance, communications and command, logistics and business systems. It is with the latter issue that this work can lend itself.

> Investment in systems to improve the efficiency and effectiveness of command and management functions in the ADF is a high priority. Improved command arrangements and systems are essential to our ability to deploy and operate effectively in complex environments at short notice.[10]

It is with this view that we see the potential of work such as this enhancing both command and control of amphibious operations as well as the associated logistics issues of that operation.

2. Packing Strategies

Several techniques have been utilised to solve packing problems in the operations research area, particularly in the combinatorial optimisation arena. Depending on the problem requirement, the techniques attempt to find optimal solutions (either maximisation of profit, minimisation of wasted space, minimisation of total containers etc) where the priority listing of the items is generally not a constraint. For instance, the classic knapsack problem is to maximise the profit by packing relevant items from a given list without exceeding the weight limit; see Chvatel (1980). In that case, it is not necessary that all items

[9] Such as the personnel in the Ship's Army Department whose role is to load/unload the cargo.

[10] paragraph 8.87 Defence White Paper 2000

be packed, unlike the requirement of the amphibious loading problem where all units in the ORBAT must be dispatched.

The watercraft loading problem has greater similarity with the cutting-stock problem however, where a given order must be 'cut' from a piece of material, steel or leather for example. In the watercraft loading problem the deck space is the material and the space to be occupied by each vehicle or item corresponds to the shapes that must be cut from the material. As an example, Cochard and Yost (1985) used a modified cutting stock heuristic to generate feasible cargo loads to improve aircraft utilisation and responsiveness in airlift operations for the US Air Force. Their Deployable Mobility Execution System (DMES) was used in three tests during military exercises with the result being that the load-planning man-hours were reduced by 90% and the airlift utilization was increased by 10%. They reduced a three-dimensional packing problem to a one-dimensional cutting stock problem which they solved with a simple heuristic. Once the desired load was determined, they could be resequenced in order to generate a feasible centre of balance. This is how Mariner would be used currently in determining whether the load has correct trim and stability conditions. The items were assumed to be rectangular, no rotation was allowed and the list was pre-sorted from longest to shortest. More recently, Amaral and Wright (2001) developed a new algorithm for the two-dimensional cutting stock problem using a very efficient branching strategy and geometric heuristics.

Algorithms which can be solved in polynomial time (P-time) are considered efficient. For a given algorithm which takes input data of length n, if we can find integers A and k such that the algorithm is always completed in at most An^k elementary operations, then the algorithm is said to run in polynomial time[11]. NP problems (Non-deterministic Polynomial) are ones for which there are no known deterministic P-time algorithm. The 'hardest' problems in NP are the NP-complete set of problems and it is strongly believed that no deterministic P-time algorithm exists for this class[12]. Problems which are not known to belong to NP (and are not known to belong to P) belong to NP-hard rather than NP-complete; see Figure 1. All problems belonging to NP-complete problems are equivalent, therefore, if a NP-complete problem is shown to belong to P, then all NP-complete problems will then belong to P; see Russell and Norvig (1995). NP-hard problems are at least as hard as NP-complete problems; see Seberry and Peiprzyk (1989). For this loading problem, which belongs to the NP-hard set of problems, we choose to implement a bin-packing algorithm. The next section gives a description of a selection of the heuristic approaches used with this algorithm.

An algorithm is defined as

> "a procedure for solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that frequently involves repetition of an

[11] Maths Thesaurus. http://thesaurus.maths.org/index.html

[12] The major unsolved question of theoretical computer science is to prove that no fast algorithm exists for an NP-complete problem, Tovey (2002).

> operation; *broadly* **:** a step-by-step procedure for solving a problem or accomplishing some end especially by a computer"[13]

whereas, a heuristic is described as

> "involving or serving as an aid to learning, discovery, or problem-solving by experimental and especially trial-and-error methods <*heuristic* techniques> <a *heuristic* assumption>; *also* **:** of or relating to exploratory problem-solving techniques that utilize self-educating techniques (as the evaluation of feedback) to improve performance"[14].

An algorithm then provides a definite procedure for carrying out a task, whereas a heuristic is a rule for how to solve a certain type of problem.[15] In relation to optimisation problems, an algorithm delivers an optimal solution and a heuristic instead delivers a quality solution in the absence of any knowledge of its quality. In determining the algorithm or heuristic to be used, the developer must understand the consequences of the chosen method. A heuristic might be computationally faster, but come at the expense of optimisation. The use of a heuristic technique that can be carried out in a few minutes but provides a load plan a few percentage points less than the theoretical optimum delivers a far superior combat effect than spending days planning "by hand". It must also be noted that the planning conference approach is unlikely to deliver the theoretical optimum load all the time.

[13] Merriam-Webster Online Dictionary: http://www.m-w.com/cgi-bin/dictionary

[14] ibid

[15] Maths Thesaurus. http://thesaurus.maths.org/index.html

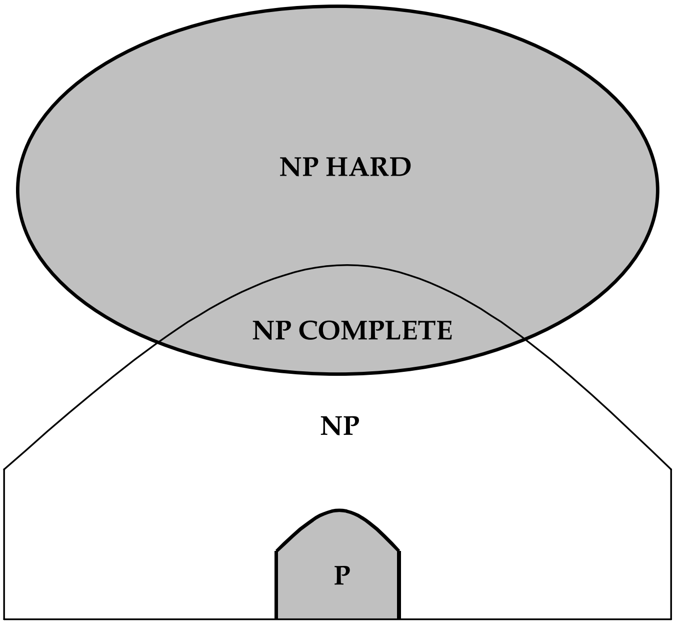


Figure 1: P is the set of easy problems. The NP-hard problems include the NP-complete problems and many hard problems that are not in NP. Almost all real problems are either easy or NP-hard. (Reproduced from Tovey 2002).

2.1 Bin-packing algorithms

It is not intended to document every algorithm applied in the bin-packing area, but to highlight a few that are relevant to the amphibious loading requirement. Further examples can be found in Sweeney and Paternoster (1992) who have compiled a bibliography of cutting and packing problems. The heuristics employed within the bin-packing algorithms have been devised with a certain application in mind, and therefore, it is unrealistic to compare the algorithms. However, it is hoped that certain elements of the heuristics will shed light on the amphibious loading problem.

The usual description of a bin-packing algorithm consists of allocating a given set of items to a minimum number of bins (containers). This term 'bin' is used irrespective of the application: items may be placed in bins, or required items cut from a given sheet of material. (In this application, a bin would correspond to the sheet of material (glass for example) and it would be required to cut as many items as possible out of the given sheet of material (windows and mirrors each of different dimensions).) According to the specific application, the items may either have a fixed orientation or they can be rotated. In industrial applications where items must be cut from material, guillotine-cutting constraints may be imposed. The majority of bin-packing algorithms described in the literature typically involve the minimisation of internal space wastage. Applications include the cutting of steel, material (which also fall into the cutting stock problem area) as well as cargo packing. (Chvatel (1980) differentiates between cutting-stock and bin-packing by the spectra of item widths required; narrow and wide respectively.) In ordering the items, either no priority is given, or the list is pre-ordered according to a specified requirement. Therefore, either items can be chosen from the list in a manner necessary to achieve an optimal solution, or the algorithm is restricted to cut/pack each item in turn from the pre-ordered list.

Bin-packing problems are members of the NP-hard class of problems, (Georgis Petrou and Kittler 2000). The number of possible combinations is too large to be explored exhaustively. If there are n items, then there are

$$(n-2)\ [\ n!\ (n-1)\ ] + 2n!$$

(note this is greater than $n!$) possible combinations in which the items can be arranged. Amongst these combinations, only a percentage will be legal solutions as some combinations will either place objects out of the packing area or overlap the boundary of the packing area. If instead there is a predefined region, the items are identical and one solution is known, then there are $n!$ ways in which to rearrange that particular solution. Therefore, for identical items with m packing strategies (where m is unknown), then there are $m\ n!$ possible rearrangements. Due to the large number of possible combinations, stochastic techniques such as simulated annealing and genetic algorithms which have methods to search the solution space, can be used to achieve the minimisation of internal wastage. These techniques will find a local solution, and are suitable to the logistical loading problem. Since an optimal solution is unlikely to be found in a practical time interval, a heuristic approach must be taken, i.e. an approximation algorithm.

One appropriate bin-packing algorithm is described in Lodi, Martello and Vigo (1999). They consider four problems consisting of combinations of orientation and guillotine cutting. The Alternate Directions (AD_{OF}) algorithm shows the closest similarity to the amphibious loading problem. Here, the items are oriented (that is, no rotation is allowed) and the notion of free cutting is utilised (that is, no guillotine cuts, as in some cutting-stock problems). Once the items have been ordered in decreasing height, the algorithm proceeds as follows; place an item in the bottom left corner of the bin and subsequent items are placed adjacent to existing items if possible. If it does not fit, it is placed at the bottom left of the next bin. Once no more items can be placed at the bottom of the bins, the packing strategy is changed to right to left and the items are placed above the existing items. Here, each bin represents a watercraft.

Since the algorithm can place items in the next bin before the first bin is full indicates a shortcoming for the amphibious loading requirement. In the algorithm, a bin represents a single watercraft as illustrated in Figure 2. However, the AD_{OF} algorithm can be adjusted to account for this problem. That is, constrain the algorithm so that all remaining items have failed to be packed into the first bin before another bin is opened. In addition, the ordering by decreasing height can be relaxed to account for the priority listing as required.

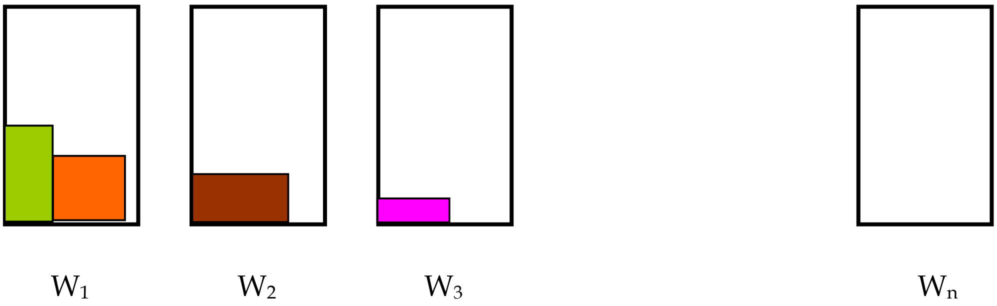


Figure 2: Using the AD$_{OF}$ algorithm, each "bin" is a separate watercraft.

Stripping their algorithm of the assumptions which are not relevant to the amphibious loading problem gives the amended AD$_{OF}$ algorithm AAD$_{OF}$:

```
Algorithm AADOF:

(i: current watercraft, j: current item)

  i = 0; j = 1;
  repeat
       i = i + 1, right_to_left = true, n_fail = 0;
       pack j, left justified, on the bottom of bin i
       repeat
          let j be the first unpacked item which can be packed
          in bin i according to the current value of
          right_to_left, if any;
             if j = nil then
                n_fail = n_fail + 1,
                right_to_left = not right_to_left
             else
                pack j into bin i according to right_to_left,
                n_fail = 0;
       until n_fail = 2
  until all items are packed
end
```

Therefore, the items shown in watercraft W2 and W3 in Figure 2 would be packed into watercraft W1 before a new watercraft "bin" is opened. Lewis (1997) was essentially outlining a bin-packing algorithm in the description of the cargo-loading algorithm for AMPHIB. The method was described as a "pigeon hole" approach where the cargo area is divided into a number of variable width and length slices. That is, each watercraft was separated into a number of bins. The assumptions are that the items are rectangular, could not be rotated, and space has to be left around each item (20 cm). Each item is slotted into one of these slices until no more items can be placed. An item was placed in a first come basis, which was therefore determined by the order of definition of the cargo items. If an item cannot be packed, it was skipped. Elements of the Alternate Directions (AD_{OF}) algorithm described by Lodi, Martello and Vigo (1999) are implemented within Lewis' (1987) packing algorithm where the items are oriented and free cutting is allowed. However, these algorithms offer no optimisation and are intended for *ordered* loading.

Heidelberg, Parnell and Ames (1988) attempt to overcome some of the requirements of standard bin-packing algorithms where items must be placed adjacent to a wall or another item and where (straight line) levels are constructed dividing the bin into sections. They dispense with the concepts of static levels or shelves and instead create dynamic barriers, which can 'bend' at 90^0 and at up to four places thereby defining up to a five-segment barrier. A possible drawback of this algorithm is that (at most) three bins are constructed. Due to the reasons outlined above, this algorithm is the one that we recommend for usage in LBaTS for the watercraft loading problem, and not for total ship loading.

2.1.1 Two-dimensional vs three-dimensional packing

Typically, cargo-loading applications in the transportation industries call for three-dimensional packing algorithms. In the context of amphibious operations, we will be restricted to two-dimensional loading, as the most typical items to be packed, (wheeled vehicles), obviously cannot be stacked. For watercraft such as commercial ferries and other multi-tiered platforms, such as HMAS JERVIS BAY (AHR45), a two-dimensional algorithm will still suffice. In the future however, three-dimensional loading algorithms may be required for sea transport and logistics problems.

The problem of packing a set of boxes into a container has prompted research efforts that have produced various solution techniques. Ivancic, Mathur and Mohanty (1989) present a heuristic algorithm based on an integer programming formulation (rather than a bin-packing algorithm) to a three-dimensional packing problem where the total cost of the containers must be minimised. The bibliography of cutting and packing problems compiled by Sweeney and Paternoster (1992) indicates an exponential increase in the amount of work devoted to three-dimensional problems over the two-dimensional problems.

Several commercial cargo-loading software products are available which target the shipping, aircraft and truck transportation sector. CargoSmith's software package, CubeIQ, for example, can load rectangular, cylindrical or L-shaped packages into containers and calculate the weight distribution and notify if the solution load is unbalanced. Cargo shapes and sizes can be pre-defined and the containers can have cut off corners, (particularly useful in aircraft loading). In addition to simply packing the listed items, an optimisation module is available to find the best solution which minimises the internal space wastage. Simulated annealing and genetic algorithm techniques are utilised for this requirement.

3. Tactical loading algorithm

The algorithm to be used for the cargo-loading module in LBaTS, called Boundary, will be based on the work of Heidelberg, Parnell and Ames (1988). The reason for this choice is the capability afforded by the use of barriers in dividing the bin into relevant sections. More flexibility and less rigidity is offered by the dynamic barriers in defining new regions than that of the AD$_{\mathrm{OF}}$ algorithm, for example. In this section, we describe the algorithm as applied to a tactical loading requirement so that elements of the ordering are maintained, and hence the minimisation of wasted space cannot be guaranteed. We defer the description of the logistical loading requirement to Section 4.

3.1 Components and terminology

In this section, a more detailed description of the algorithm of Heidelberg, Parnell and Ames (1988) is presented. To begin, the coordinate system is defined in the (x,y) plane and the packing direction is from bottom to top. The barrier defines the region between packed items and the available packing area. Each barrier is given a type (defined by up to three letters: **C**entre, **R**ight and **L**eft), which instructs the algorithm the order in which the items are placed. For example, CRL indicates that the algorithm will firstly attempt to place an item in the centre, then to the right and lastly to the left. The lowest level determines the type of barrier and Figure 3 shows the nine possible barrier types.

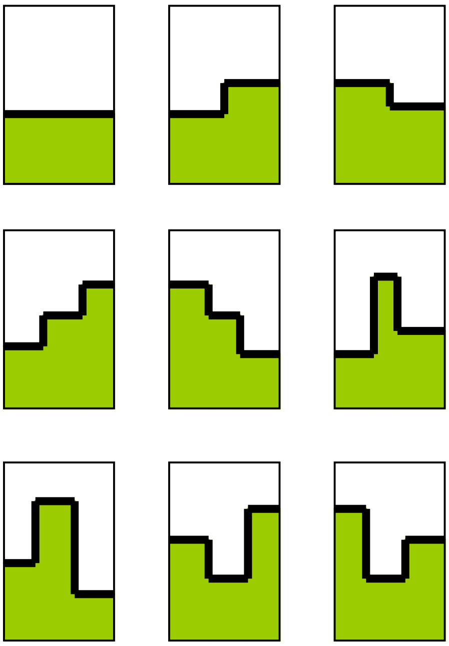


Figure 3: Barrier Types in the Heidelberg, Parnel and Ames heuristic

The section that is to be used to try to pack the next 'best' item is called the *candidate bin section.* Depending on the nature of the problem, the definition of 'best' will vary. For a tactical load plan, where the off-load has been determined with associated priorities, the "best" item will be chosen from a restricted search set. This search set will include the current item up to a further *m* in the list. For a logistical load plan the algorithm needs to minimise space wastage and therefore the "best" item will probably not be the current item in the list. An algorithm which employs strict ordering will choose the next item in the list, whereas a relaxed ordering algorithm will choose the best item up to the m^{th} item in the list.

It is likely that the three-levelled barriers cannot accommodate large items and therefore one- and two-levelled barriers (C, LR and RL) are more capable of holding these items. Generally however, the "best" item is packed on top of the lowest level and adjacent to the tallest barrier edge, as illustrated in Figure 4. A tolerance *t* may be added to the height of the tallest barrier edge if required to place the item as shown. Depending on the height of the new object relative to the already placed object, the tolerance will be defined as low or high. The value of *t* will depend on the application. In implementing this feature, we incorporate a tolerance in both directions i.e. beyond and below the level, as shown in Figure 5.

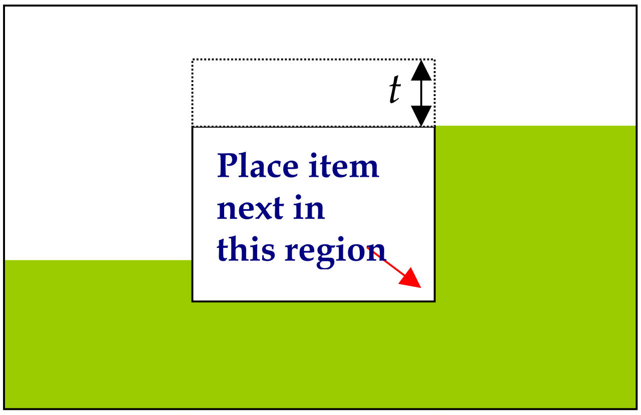


Figure 4: Region for placement of new item.

The next step of the algorithm is to adjust the barrier in a manner to minimise the wastage of space. The barrier adjustment must then result in one of the nine possible barrier types shown in Figure 3. Therefore, the barrier adjustment is made either using the width or height of the new item, with each possibility shown in Figure 6 and Figure 7 respectively. Of these two possibilities, the one which gives the least wasted space should be chosen.

In the event that an item cannot be placed in the candidate bin section then the barrier should be adjusted and the packing reattempted. If the failure occurs when the barrier type is C, then no more items can be packed and the algorithm terminates. For other barrier types, the lowest section is increased to the least height of the adjacent sections and the barrier level is reduced by one. Figure 8 shows a case where an item cannot be packed into a three-levelled barrier (CLR) and the barrier adjustment results in a two-levelled barrier type (CR).

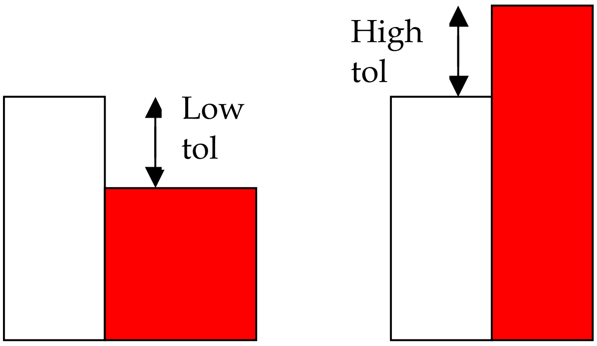


Figure 5: A low or high tolerance is calculated depending on the height of the new object (in red) relative to the already placed object (in white)

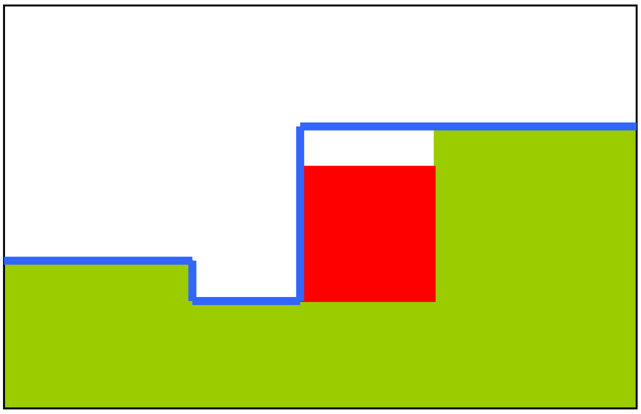


Figure 6: Adjustment of barrier using width of new object

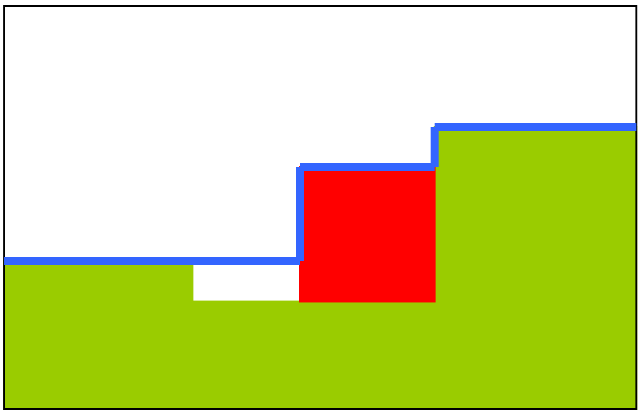


Figure 7: Adjustment of barrier using height of new object

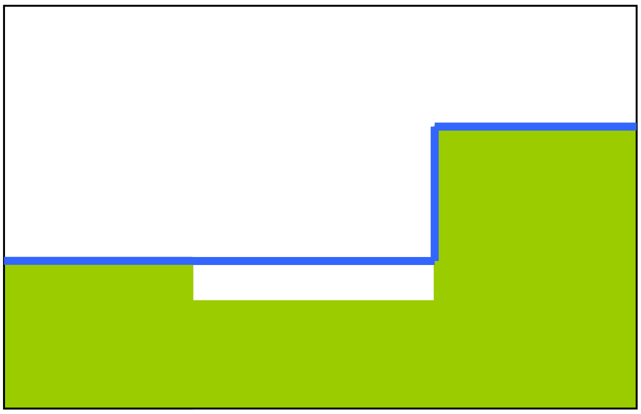


Figure 8: Adjustment of barrier when objects fail to be added (to C)

The adjusted barrier types are shown in Table 1, assuming that the tolerance *t* is set to zero. Note that a barrier type is reduced in level either when an item is an exact fit or the algorithm fails to place the item.

Table 1. Barrier adjustments

Original barrier	Item is added (*t*=0)	All items fail (*t*=0)
C	LR or RL	STOP
LR	LR	C
RL	RL	C
LCR	LCR or LR	LR
RCL	RCL or RL	RL
LRC	LRC or RLC	RL
RLC	RLC or LRC	RL
CLR	CLR or LCR or LR	LR
CRL	CRL or RCL or RL	RL

3.2 Structures

The previous section has described the algorithmic components for the tactical packing routine whilst this section describes the algorithmic structures. The two algorithms are based on strict and relaxed ordering of items to be packed with each approach offering advantages and disadvantages. Strict ordering whilst simpler to implement, will result in more wasted space. Relaxed ordering will pack more efficiently since the algorithm will search through items in the list to find one that fits the current barrier, but will come at the cost of increased computational time. In a tactical situation where the ORBAT has to be discharged in a particular way, strict ordering may present the better option.

The outline of each algorithm is as follows, where X and Y refer to the coordinate system in the usual way:

3.2.1 **Algorithm Load 1** (strict ordering of items)

```
Set barrier type to C and set X and Y values accordingly

1. Select first item not yet packed
2. Attempt to pack item
3. If packing of item is successful then
         Alter barrier type in two ways, using height or width
   of new object
         Test for amount of wasted space
         Choose new barrier for least wasted space
         Adjust X and Y values that describe barrier
   Else (i.e. failed to load any item)
         Adjust barrier to next lowest level
         Adjust X and Y values that describe barrier
4. Repeat from step 1
```

3.2.2 **Algorithm Load 2** (relaxed ordering of items)

```
Set barrier type to C and set X and Y values accordingly

1. Select first item not yet packed
2. Attempt to pack item
3. If packing of item successful then
         Alter barrier type in two ways, using height or width
   of new object
         Test for amount of wasted space
         Choose new barrier for least wasted space
         Adjust X and Y values that describe barrier
4. Repeat with next item, until all items packed or last item
fails
```

```
          (note, may stop last item at this priority level and then
          do next priority level when all or almost all at this
          level have been packed)

    5. Else (i.e. failed to load any item, or any item of correct
    priority)
            Adjust barrier to next lowest level
            Adjust X and Y values that describe barrier
    6. Repeat from step 1
```

3.3 Implementation

A necessary requirement for any software application is its suitability for the end users of the product. Therefore, a package is required which is straightforward to learn, easy to use and produces output that can be trusted. In this section, the capability of the packing algorithm is demonstrated on an example from an amphibious operation, illustrating the ease of use and the readily understandable output.

In a tactical load plan where the unloading order of items has been determined via a priority system, the algorithm is restricted in its look-ahead ability. To pack the next 'best' available item into the candidate bin section, the search set includes the current item *i* up to item *i + 5*. (It would be possible to give all items equal priority and look-ahead to the end of the list, but this would degrade the efficiency of the algorithm, especially when the number of items is large.) It is possible to incorporate other heuristic methods as the application dictates.

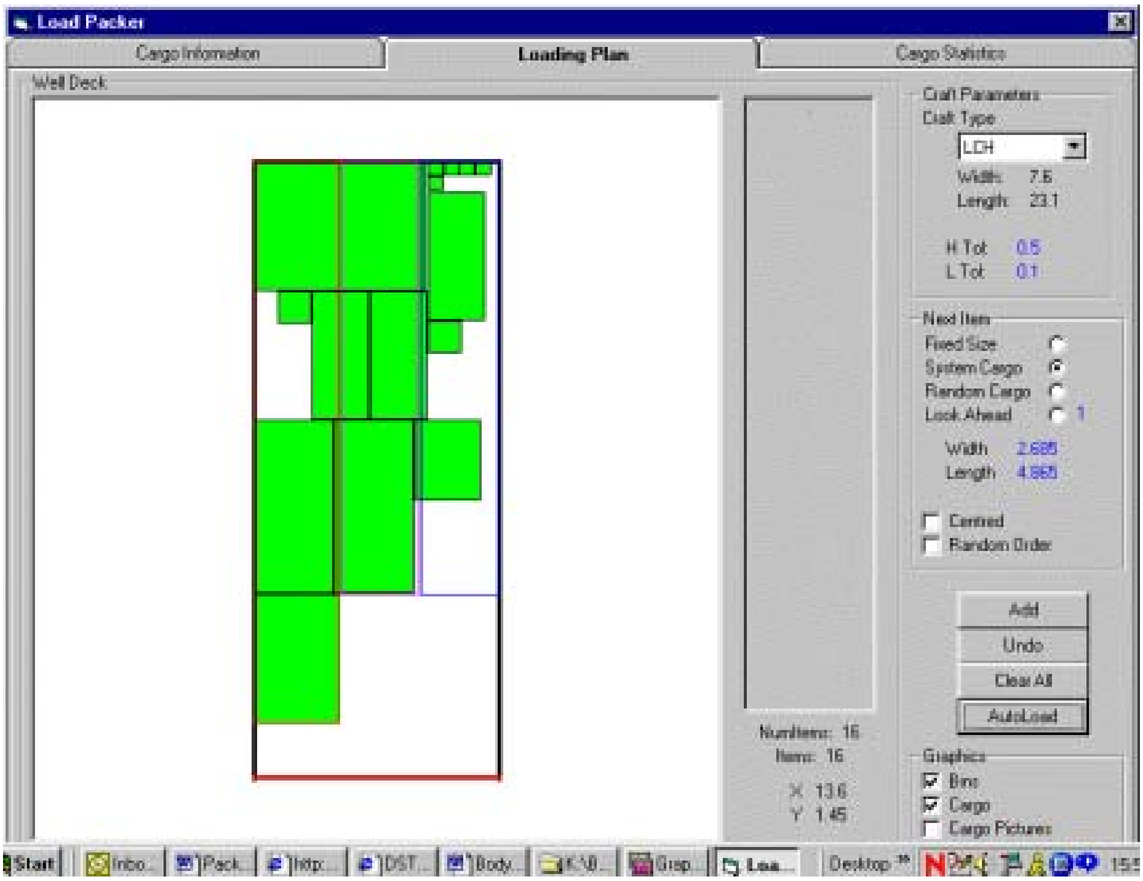
Techniques used in the logistical loading problem, where the ordering of the items is not required, will be addressed in the next section.

To demonstrate the packing algorithm, a test sample of items is chosen to illustrate a typical load plan for an amphibious operation. Table 2 gives a description of each item to be loaded, the number of each item and its dimensions.

Table 2. A sample load plan for an amphibious operation. The number of each item, its type and dimensions are shown. (As this problem is two-dimensional, the height is not required.)

n	**Item**	**Width (m)**	**Length (m)**
1, 2	APC	2.52	4.85
3, 4, 5	Land Rover 4x4	2.15	4.74
6, 7, 8, 9, 10	LOTS Equipment	0.5	0.5
11, 12	LOTS Equipment	1.2	1.2
13, 14	Unimog	2.84	6.54
15	Trailer	2.4	3
16	APC Command Post Carrier	2.685	4.865

The watercraft chosen for this example is a LCH having length 23.1 m and width 7.6 m. Using a look ahead of *i* +5, the algorithm packs all the items into one load. Without using a look ahead, the algorithm packs the items into two loads and using random packing it takes on average two loads. This shows that for small loads, the look ahead function enables more efficient tactical loading where offload must be conducted within the constraints of a predetermined order. Figure 9 shows the graphical interface of the Boundary module with the loading plan solution found with the look ahead function bin-packing algorithm.



Craft Parameters
Craft Type: LCH
Width: 7.6
Length: 23.1
H Tol: 0.5
L Tol: 0.1

Next Item
Fixed Size ○
System Cargo ●
Random Cargo ○
Look Ahead ○ 1
Width: 2.685
Length: 4.865
□ Centred
□ Random Order

NumItems: 16
Items: 16
X: 13.6
Y: 1.45

Graphics
☑ Bins
☑ Cargo
□ Cargo Pictures

Figure 9: Solution from cargo-packing algorithm for the data shown in Table 2. The watercraft chosen is an LCH with width 7.6m and length 23.1m.

The implementation carried out here suffered from rounding errors leading to an increasing number of smaller slices of large packing problems. Disregarding this, we found this bin-packing algorithm to be an efficient method in solving the tactical loading problem.

4. Logistical loading algorithm

For a logistical load plan where the position of items is not constrained, it would be desirable to find an optimal solution for the load plan. Due to the complexity of this problem however, (NP-hard, as described in Section 2), the identity of the global optimal solution is unknown. Modern meta-heuristic techniques such as simulated annealing (Aarts and Korst (1989)) and genetic algorithms (Goldberg (1989)) are capable of locating local optima. These methods can be re-implemented generating more and more local optimal solutions and the 'best' can be chosen from this set.

Hopper and Turton (1999) use a genetic algorithm to find a 'good' configuration to a two-dimensional industrial packing problem. They allow 90^0 rotation of items and implement two hybrid genetic algorithms where different heuristic routines have been embedded in the algorithm. The use of genetic algorithms in search and optimisation procedures has increased recently with applications in pattern recognition and telecommunication networks; see Goldberg (1989). The genetic algorithm meta-heuristic is based on the evolutionary processes observed in nature, where the reproduction of the next population (or solution) is based on the 'survival of the fittest' principle. Therefore, mechanisms such as mating and crossover are incorporated to achieve this.

Georgis, Petrou and Kittler (2000) use a simulated annealing algorithm as part of their constrained rectangle packing problem with good results. Why has the process of annealing been utilised for this optimisation problem? Annealing is the physical process of heating up a solid until it melts, followed by cooling it until it crystallizes into a state with a perfect lattice. During this process, the free energy of the solid is minimised. Practice shows that the cooling must be done carefully in order not to get trapped in locally optimal lattice structures with crystal imperfections. To apply this process to this optimisation problem, a correspondence between the objective function and the free energy as well as solutions and physical states, is established. In addition, the feature of slow cooling is introduced through a slowing changing control parameter.

To return to the description by Georgis, Petrou and Kittler (2000), the algorithm begins with a random valid configuration and then randomly applies various operators to generate a new valid configuration. The new configuration is accepted if the uncovered area is reduced, but it still may be accepted if the uncovered area is increased with a probability of $exp\ (-\Delta A/T)$, where ΔA is the increase in uncovered area and T is the control parameter.

4.1 Simulated annealing

The heuristic for the logistical loading solution incorporating the simulated annealing process is as follows (Georgis, Petrou and Kittler, 2000):

```
SimAnn
    1. Generate a valid initial configuration
    2. Randomly apply movement operators to this configuration
    3. Calculate the value of the objective function
    4. Accept new configuration if value is reduced, or accept an
       increase with a given probability: $exp\ (-\Delta E/T)$ where $\Delta E$ is
       the change in the objective function and $T$ is the control
       parameter
    5. Iterate towards convergence (when the difference in
       successive objective function values is within some
       tolerance), or stop after a given number of iterations
```

In order to define the objective function some definitions must first be made, as shown in Table 3.

Table 3. Relevant variables required in formulation of loading problem

Variable	Description	Units
m_i	mass of cargo item *i*	tonnes
P_i	priority of cargo item i	integer; highest = 1
w_i	width of cargo item *i*	metres
l_i	length of cargo item *i*	metres
W	width of watercraft	metres
L	length of watercraft	metres
n	total number of items	integer; $i = 1,\ldots,n$
p	total number of watercraft	integer; $j = 1,\ldots,p$
d_i	distance from centre of item *i* to door	metres
R	minimum enclosing rectangle	defined by the coordinates of the vertices of rectangle
A	area of minimal enclosing rectangle	metres^2
x_i	describes if item *i* is contained in *R*	$x_i = 1$ if $i \varepsilon R$, 0 otherwise

At each iteration in the algorithm, a movement operator is applied to the current configuration. These operators are Move, Exchange, Rotate and Compaction. As described by Georgis, Petrou and Kittler (2000), a string *S* is devised which represents the relative position of item *j* to item *k*. Each element of the string belongs to {*L*,*A*,*B*,*R*} (**L**eft, **A**bove, **B**elow or **R**ight). This string is not unique, since item *j* can be both below and to the left of item *k*. Their algorithm works on the string *S* using the operators Move, Exchange and Rotate, and then constructs the configuration with these relative positions and checks its validity. As the number of items increases, the percentage of invalid configurations generated increases. The string *S* is not altered with the compaction operator.

The objective function *E* in the simulated annealing process is to minimise the wastage of space, or equivalently, to find the minimum enclosing rectangle. How is this quantity determined? Consider the case shown in Figure 10. Here, two configurations of four items are shown with each revealing a different amount of wasted space. Figure 10 illustrates the idea of a 'minimum-enclosing rectangle' which describes a rectangle which must enclose all items. A constraint of the algorithm however, is that the structure of the string S must be maintained. That is, the relative placement of the items must remain in determining the minimum enclosing rectangle. Figure 10 shows how the area of the minimum enclosing rectangle has reduced from rectangle (a) to (b). In addition, the string S has not been altered.

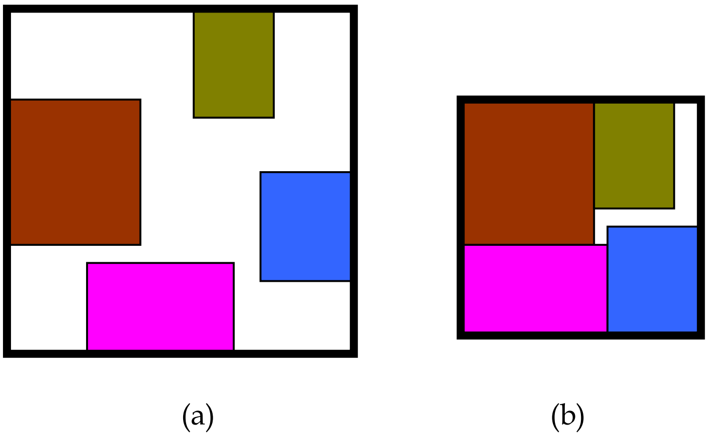


Figure 10: Illustration of a minimum enclosing rectangle R. Area of rectangle (b) is less than the area of rectangle (a) and the white space has been reduced from (a) to (b). In addition, the relative placement of objectives has remained intact.

As the algorithm proceeds, it may be that not all items are included in R. To accommodate this, the variable x_i is set to one when item $i \varepsilon R$, and zero otherwise. Therefore, the objective function for the simulated annealing procedure is defined as follows:

$$E = \text{minimise} \left(A - \Sigma_i w_i l_i x_i \right),$$

where the area of the minimum enclosing rectangle R is A.

4.1.1 Move

- Choose two items at random
- Randomly move item i to one of the four sides of item j
- Update S and item j inherits the attributes of item i

4.1.2 Exchange

- Choose two items at random
- Swap their relative positions and update S

4.1.3 Rotate

For the purposes of amphibious loading, the use of the Rotate operator would either be eliminated or only applied to particular items; static objects and trailers for example.

- Select an item at random
- Reverse its width and height attributes

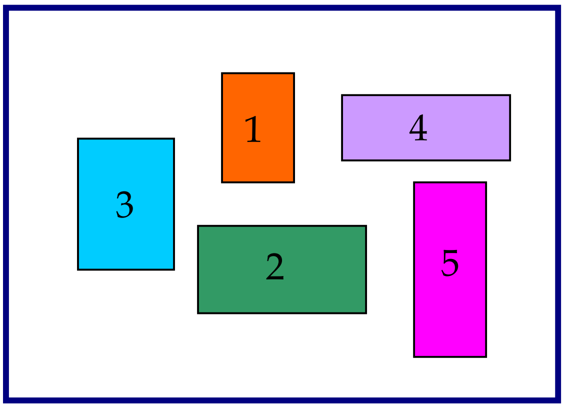
4.1.4 Compaction

Given a string *S* representing the relative position of items, this operator places the items to give absolute placement, once one of the above operators has been applied.

- Every item is checked to determine whether or not it has a neighbouring rectangle to the left. If not, it is then classified as a leftmost object and pushed onto the stack
- The *x* and *y* coordinates of all objects are then calculated by taking into account the relative left-to-right, and above-to-below placement respectively
- If some objects are fixed, their *x* and *y* coordinates must not change
- Calculates the minimum enclosing rectangle

Once Move, Exchange or Rotate is applied, the compaction operator is used to place the items and construct the minimum enclosing rectangle, and then the validity of the proposed change must be checked.

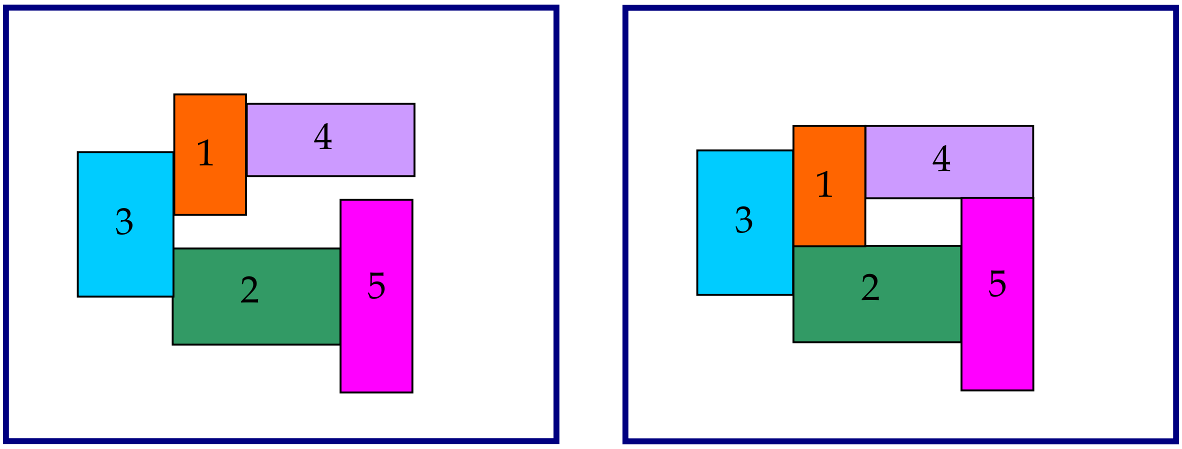
It is instructive to work an example of the compaction operator and the construction of the string *S*. The length of the string is $l = n\ (n\text{-}1)/2$, where *n* is the number of items, with the string represented as $S = s_1\ s_2\ ...\ s_l$. Each element of the string, s_i, describes the relative position of item *j* to item *k*, i.e. s_i is constructed from (*j*,*k*). Figure 11 illustrates a case with five items, therefore $l = 10$. Here, s_1 = (1,2), s_2 = (1,3), s_3 = (1,4), s_4 = (1,5), s_5 = (2,3), s_6 = (2,4), s_7 = (2,5), s_8 = (3,4), s_9 = (3,5) and s_{10} = (4,5).



S = {ARLLRBLLLA}

Figure 11: String S defining the relative position of item j to item k for n = 5

Figure 12 shows how the horizontal and vertical compaction operators determine the minimum-enclosing rectangle for the group of objects shown in Figure 11. The reader is left to verify that the string defined in Figure 11 remains unchanged once the two compaction operators have been applied.



Horizontal compaction *Vertical compaction*

Figure 12: Illustration of the horizontal compaction followed by the vertical compaction on the items shown in Figure 11.

4.1.5 Validity

- Ensure that items do not overlap
- Ensure items remain within the watercraft, or overall allowable area (if considering multiple watercraft or a ship)
- Ensure items are adjacent if required.

4.1.6 Cooling Schedule

As in the physical analogue of the simulated annealing process, a carefully controlled cooling process is required in the algorithm. Here we use the schedule defined in Georgis, Petrou and Kittler (2000) where the cooling coefficient T is updated via $T^* = 0.9\ T$. This choice is verified via experiment in Aarts and Korst (1989).

4.1.7 Initial conditions

To begin the solution procedure, an initial configuration is required. It may be random or generated from the bin-packing algorithm described in section 3.

4.1.8 Determining number of watercraft

To calculate the total number of watercraft needed to discharge the cargo, an overall solution representing the lower bound can be constructed. A simple lower bound for the number of watercraft can be constructed by dividing the total area of all items to be packed by the area available in each watercraft (see Lodi, Martello and Vigo (1999)), this gives the minimum number of watercraft to be $p = \Sigma_i w_i l_i / WL$. That is, the minimum size of the overall solution is (*width, length*) = (W, L*p). In the SimAnn algorithm, it would be possible to determine the number of watercraft by placing a barrier at intervals of length L and not allow items to be placed across the barrier. (There will be p barriers.) Once all the items have been packed, the number of watercraft n is then the length of the minimal enclosing rectangle A_L divided by the length of the watercraft L (and rounded up to the nearest integer). The number of watercraft ($n = A_l/L$) is greater or equal to the lower bound p .

4.1.9 Advantages

Once the final solution has been found using the simulated annealing algorithm described above, there is scope to fill gaps with additional items. The algorithm has found a solution with the given cargo list, but it may be possible to add further items, such as pallets for example, if space permits.

This algorithm also allows for additional rules to be incorporated. Such rules are relevant when loading dangerous items as we may:

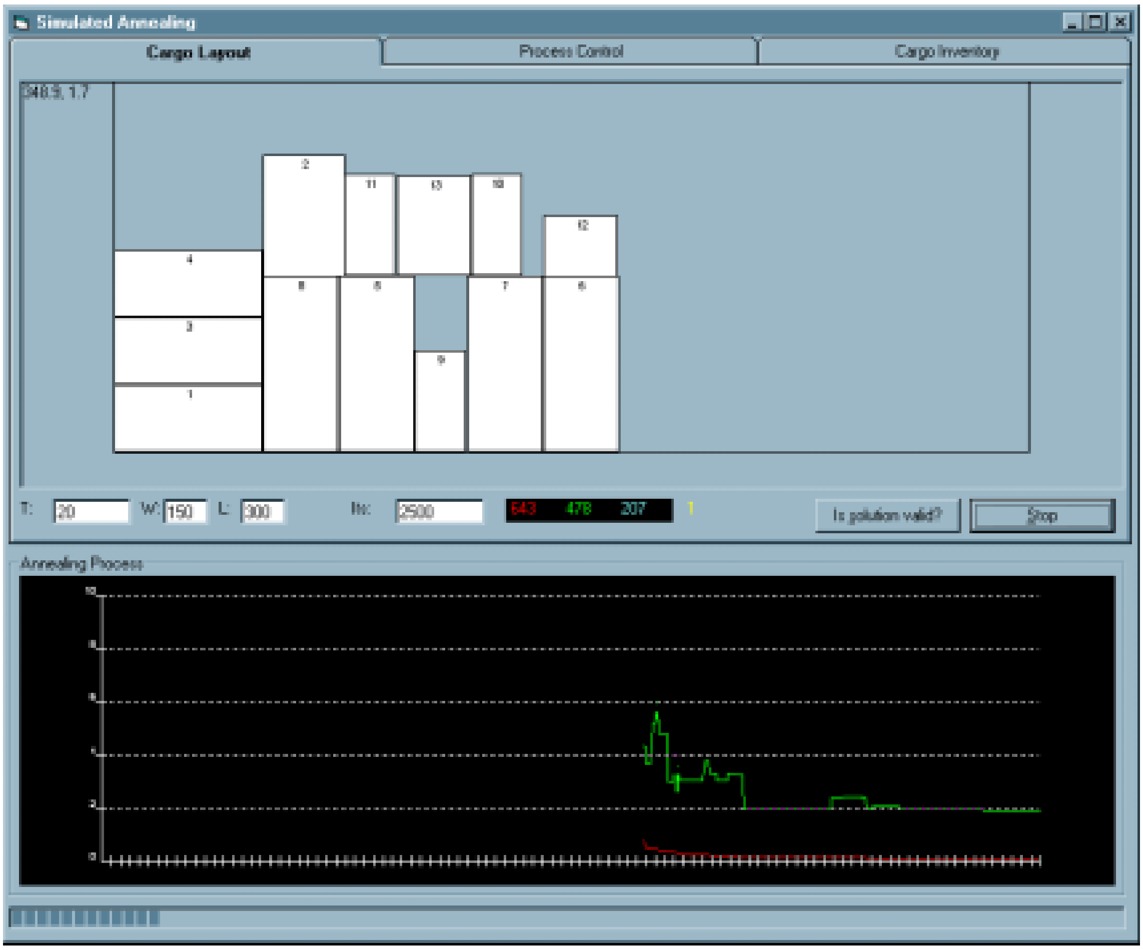
- disallow certain items to be loaded adjacent to each other or within a fixed distance (eg ammunition and fuel), or
- allow fixed items to be accounted for (eg fire extinguisher), or
- disallow loading in certain zones (eg fire and safety zones).

As demonstrated by Georgis, Petrou and Kittler (2000) with an loading example with a fixed item in place, these types of rules can be easily incorporated.

The items highlighted above provide the greatest strength of this algorithm to the amphibious loading problem and we would recommend that the logistical loading algorithm which is to be incorporated in LBaTS contain such features.

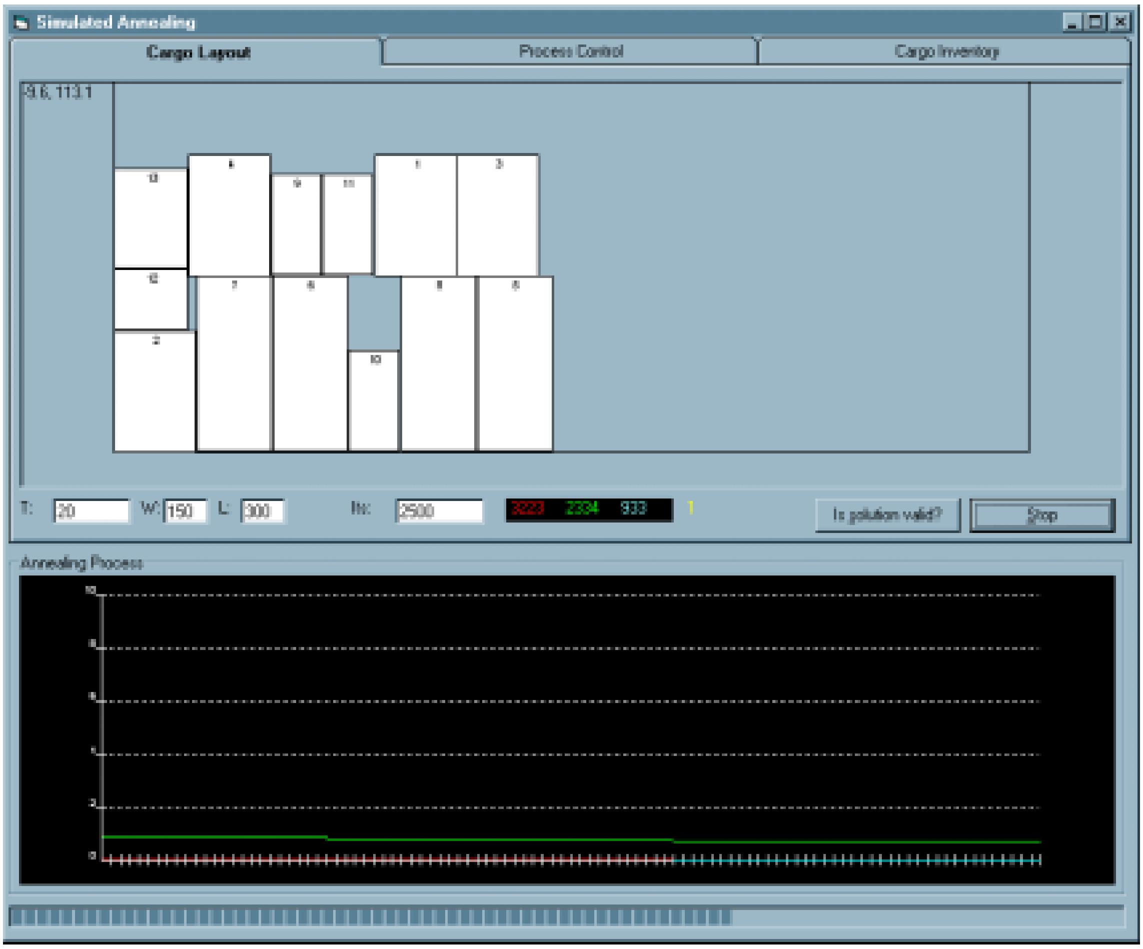
4.1.10 Example

To demonstrate the simulated annealing algorithm, 13 objects were chosen with random widths and lengths, and all operators were allowed (Move, Exchange and Rotate). The objective function to be minimised is the area of the enclosing rectangle. Figure 13-Figure 14 illustrate the convergence in the objective function (green) and also shows the cooling schedule (red). In this case, convergence was made by a human-in-the-loop decision where it was judged that value of the objective function was reasonable and did not change for an acceptable number of iterations. Convergence criteria such as a given number of iterations when the difference between successive solutions did not change, or when a particular value in the objective function is found, can also be used.



T: 20 W: 150 L: 300 Its: 2500

Figure 13: Iteration 233: value of the objective function (green) and cooling schedule (red).



T: 20 W: 150 L: 300 Itn: 2500

3223 2334 933 1

Figure 14: Iteration 933: value of the objective function (green) and cooling schedule (red).

4.1.11 Summary

Each item in the cargo list has associated dimensions and position of bottom-left corner and has the capability to use the Move and Exchange operator, with certain items allowed to use the Rotate operator. Once the items have been adjusted by one of these operators, a new configuration must be calculated using the compaction operator. That is, the compaction operator calculates the minimum-enclosing rectangle. Next, the validity of the new configuration must be determined. Finally, the objective function is calculated with the new configuration and it is either accepted or rejected. This process is re-iterated until either convergence is attained, or the maximum number of iterations is achieved.

5. Simultaneous solution

Here, we wish to use the simulated annealing process to find a packing solution to the tactical and logistical loading requirements simultaneously. To combine the two objective functions we assign a weighting for each requirement. The requirement in the tactical loading problem is to pack the items according to the priority listing of the load plan. With the logistical loading problem, the requirement is to minimise the wastage of space. Both problems require the minimisation of total watercraft needed to either discharge the items to the beach (tactical) or return all items to the main amphibious ship (logistical).

Additional constraints to consider include incorporating a disallowed area for certain items of cargo, for example ammunition may not be allowed to be packed adjacent to fuel trucks, or trucks may not be allowed to be packed so tightly that they don't allow access to fire hydrants.

To formulate the objective function, we consider Figure 15 where the black lines show the distance from the centre of an object to the centre of the watercraft door. The distance from the item to the door increases as an item is placed towards the rear of the watercraft. Ideally, we require items with highest priority (low P) be loaded closest to the door (low d_i), and the lowest priority items (high P) be loaded furthest from the door (high d_i).

To ensure the highest priority items are placed closest the door, we require a scaling factor for d_i which weights high distances with lower priority items (high P). Therefore, it is necessary to choose a continuously increasing nonlinear function. Here we choose $f(P_i) = (1 + tanh(P_i/n))/2$ so as not to dominate the objective function, see below. Therefore, the quantity $\Sigma_i f(P_i)\, d_i\, x_i$ will be maximised as low priority items are loaded towards the rear of R. Recall that maximising $\Sigma_i\ f(P_i)\ d_i\ x_i$ is equivalent to minimising $-\Sigma_i f(P_i)\, d_i\, x_i$.

Using the notation described in Table 3, the objective function becomes:

$$E = \text{minimise}\ [\ -z_1\, \Sigma_i f(P_i)\, d_i\, x_i + z_2\ (\ A - \Sigma_i\, w_i\, l_i\, x_i\ )\ ],$$

where z_1 and z_2 represent the weights associated with priority listing and minimisation of wasted space respectively, and where $z_1 + z_2 = 1$. For example, to find a solution for a tactical load only and we are not concerned with the minimisation of wasted space, then $z_1 = 1$ and $z_2 = 0$. Likewise, to solve the logistical loading problem separately, $z_1 = 0$.

Note, $L_1 = -z_1\, \Sigma_i f(P_i)\, d_i\, x_i$ decreases as items are placed in R, and as described previously, $L_2 = A - \Sigma_i\, w_i\, l_i\, x_i$ also decreases, with $L_2 \rightarrow 0$ as R fills. With $f(P_i)$ varying between 0 and 1 and $\Sigma_i\, d_i$ only exceeding A when R is full (noting that $0 < f(P_i) < 1$ will scale it), L_1 will not dominate the objective function as it will be of the same order as L_1. This has been confirmed through numerical experiment, and noting that $f(P_i)$ approaches A as n increases and R fills. It is not the intent to demonstrate this solution technique here, but to describe the concept of simultaneous solution technique, hence we do not explore how z_1 and z_2 should be chosen.

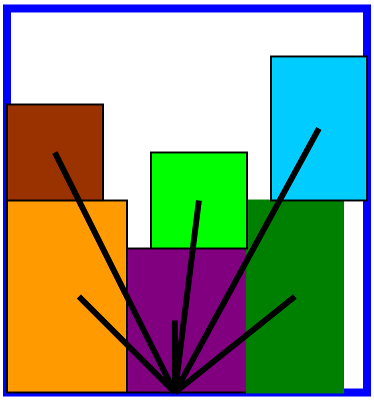


Figure 15: Illustration of objective function for the simultaneous solution. Items with highest priority are required to have smaller distances than those with lower priority. Therefore, the lowest priority items are forced towards the rear of the vessel.

To generate an initial configuration in the tactical loading problem, the items can either be placed with strict priority order, or the existing bin-packing algorithm could be implemented. In this manner, since z_1 will be greater than z_2, it is possible that the number of iterations required to find the final solution will be reduced with this initial configuration.

5.1 Weight considerations

To ensure the draft of the watercraft does not exceed its designated level, the weight distribution must be determined. Presently, the Mariner software package determines if the load is feasible in terms of trim and stability. It would be possible to use elements of the method described by Georgis, Petrou and Kittler (2000) for this problem as well. As illustrated in their paper, certain items may be fixed before implementing the algorithm. Therefore, the heaviest items in the cargo list could be placed along the centre line of the watercraft. Or, the objective function could be adapted to include not just the minimisation of uncovered area but also a constraint on the weight distribution (or centre of balance). We do not include these considerations here.

6. Software description

As a test of the algorithm a simple experimental implementation was coded in Microsoft Visual Basic. The graphical user interface (GUI) shown in Figure 9 illustrates the ease at which the module can be utilised. A selection from the list of available cargo is made to form a cargo load from Form 1 (Cargo Information), and then the algorithm is invoked in Form 2 (Loading Plan) by choosing the watercraft type (LCH, LCM8) and choosing AutoLoad. Cargo information and parameters are stored in an Access Database. This will enable rapid integration of the system with any new cargo packing technique. The cargo packing algorithms work by manipulating the Access DB. The database can be altered as required for the desired application, and is not restricted to military applications.

The tactical loading module of LBaTS, Boundary, as described in section 3 has been developed as a prototype and may be utilised in its current form. The simulated annealing component however, has been developed for internal testing only at this stage.

6.1 Overlapping

Once the compaction operator has been applied, the validity of the resulting configuration must be determined. That is, do any of the objects overlap, or have then been placed outside of the box? Figure 16 illustrates the eight distinct variations on the spatial relations between two 2-dimensional objects in a 2-dimensional space, where the blue object is the new object to be placed. (Reproduced from Mark (1999).) The concepts of stacking and adjacency rules can be invoked with the application of some of the overlapping rules shown in Figure 16. For example, in order to allow pallets to be stacked on the tray of a truck, the overlapping rule shown in Figure 16(e) or (g) could be utilised. If instead two items were required to be packed adjacently, then rule Figure 16(c) can be applied.

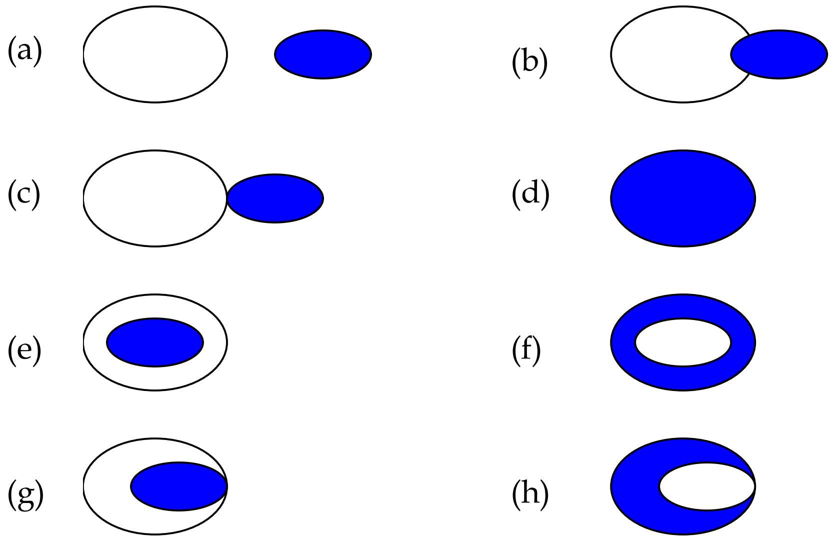


Figure 16: The eight distinct spatial relations between two 2-dimensional objects in a 2-dimensional space. (Reproduced from Mark (1999).)

6.2 Simulated annealing GUI requirements

The concept of modelling and waiting for an optimised solution is not one that is generally acceptable in the field. Operators expect real time solutions, therefore a graphic is necessary which is updated after each iteration of the simulated annealing process indicating:

- Iteration number
- Total number of iterations allowed

- Graph representing the convergent behaviour of the algorithm
    - Logistical: graph indicates percentage of space used
    - Tactical: graph shows the component from objective function
- Stop or pause capability
- Override capability for emergency situations (for example, certain operators or rules eliminated)

The process is computationally intensive, and the user needs to know that the program is still executing.

7. Discussion

The utility of the bin-packing algorithm Boundary has been demonstrated on a recent exercise (Tandem Thrust 01) with the RAN. With a given load plan, the number of watercraft necessary to discharge the load to the beach was calculated. This information was then incorporated into the overall simulation using the current sea-state and the tidal windows to estimate the operational time, ie. time to load items from amphibious ship, transit to beach, offload items, return to amphibious ship, and continue until the entire load was discharged. The output from the simulation (AWOCSS[16]) indicated that the original plan was not achievable with the given constraints. That is, the given ORBAT could not be deployed to a given location in a suitable time frame with the given number of watercraft. This information was then forwarded to the CATF[17] who made a decision to proceed at a more conducive time. This result demonstrated the usefulness of this tool in the decision-making process and confirmed that the tool could be used in a command and control environment.

The bin-packing algorithm of Heidelberg, Parnell and Ames (1998) could be extended to use more than three bins in dividing the watercraft area. However, given the typical dimensions of items required in an amphibious operation, it is unlikely that more bins are appropriate. In the evaluation of future watercraft, this restriction may have implications in determining the effectiveness of those craft.

The simulated annealing method has been used in a vast array of combinatorial optimisation problems, from the traditional travelling salesman problem, through to circuit design and image reconstruction. In the defence arena, this process may well suit a variety of problems, and not simply restricted to the amphibious problem described here as both the land and air domains have their own loading requirements. Other problems could include scheduling, resource allocation and location, optimal routing, detection avoidance and applications in mine warfare.

[16] AWOCSS was the prototype to LBaTS.

[17] Commander Amphibious Task Force.

There is scope to develop a 'smart' system where a catalogue of 'good' loads is maintained. When a new load plan is required for the logistical loading problem, a solution can be taken from the catalogue (from an item list with similar characteristics) and used as an initial condition in the simulated annealing process in the calculation of the new solution. (There is no guarantee of faster convergence with a so-called 'better' initial solution.) Further extensions to this algorithm include the incorporation of overlapping rules where the notion of stacking can be made.

Perhaps more importantly, the strength of the method is that constraints can be dynamically added, reflecting the ever-changing environment of military operations. As an example, recent operations in East Timor demonstrated the need for operationally flexible tools such as these. The ADF leased the fast catamaran HMAS JERVIS BAY which presented a challenge to make effective use of the platform as it consisted of various tiers with different weight restrictions.

8. Conclusions

We recommend that a logistical loading module to be incorporated in future versions of LBaTS containing the features of the combinatorial optimisation method highlighted in this report. The strengths include the ability to dynamically change the loading rules on various cargo items, such as where in the space can the item be loaded and with respect to which other items, and not be restricted to simple loading spaces.

In future, incorporating the simultaneous solution approach to the tactical and logistical loading problems would be advantageous. By having one method, the user requirements are greatly simplified.

The implementation of either of algorithm does not have to be restricted to amphibious operations. A trivial change to the database changes the watercraft dimensions as well as the inventory list, therefore, any kind of transportation could be modelled considering it in a two-dimensional plane.

9. References

Aarts, E. and J. Korst (1989) Simulated annealing and Boltzmann machines. John Wiley & Sons, Chichester, p59.

Amaral, A.R.S. and M. Wright (2001) Efficient algorithm for the constrained two-dimensional cutting stock problem. *International Transactions in Operational Research,* **8:** 3-13.

Australian Defence Force Publication ADFP 12 Amphibious Operations.

Australian Defence Force Publication ADFP 22 Sea Transport.

CargoSmith *http://www.cargosmith.com/*

Chvatal, V. (1980) Linear programming. W.H. Freeman and Company, New York.

Cochard, D.D. and K.A. Yost (1985) Improving utilization of Air Force cargo aircraft. *Interfaces,* **15:** 53-68.

Defence White Paper; Defence 2000: Our Future Defence Force. Australian Government Printing Office, Canberra, 2000. (Unclassified)

El-Bouri, A., N. Popplewel, S. Balakrishnan and A.S. Alfa (1994) A search-based heuristic for the two-dimensional bin-packing problem. *Infor,* **32:** 265-74.

Georgis, N., M. Petrou and J. Kittler (2000) On the constrained rectangle packing problem. *International Journal of Modelling & Simulation,* **20:** 293-9.

Goldberg, D.E. (1989) Genetic algorithms in search, optimisation and machine learning. Addison-Wesley Publishing company, Reading, Mass.

Heidelberg, K.R., G.S. Parnell and J.F. Ames IV. (1988) Automated air load planning. *Naval Research Logistics*, **45**: 751-68.

Hopper, E., and B. Turton (1999) A genetic algorithm for a 2D industrial packing problem. *Computers & Industrial Engineering,* **37:** 375-8.

Ivancic, N., K. Mathur and B.B. Mohanty (1989) An integer programming based heuristic approach to the three-dimensional packing problem. *Journal of Manufacturing & Operations Management*, **2:** 268-98.

Lewis, P.R. (1997) A program for the modelling of amphibious deployment and sustainment processes. *DSTO Technical Report DSTO-TR-0486.*

Lodi, A., S. Martello and D. Vigo (1999) Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *Informs Journal on Computing,* **11:** 345-57.

Mark, D.M. (1999) Spatial representation: a cognitive view. Geographical information systems, Volume 1, Principles and Technical Issues, Second Edition. Editors P.A. Longley, M.F. Goodchild, D.J. Maguire and D.W. Rhind. John Wiley & Sons, Inc. New York.

Martello, S., D. Pisinger and D. Vigo (2000) The three-dimensional bin packing problem. *Operations Research*, **48:** pp.256-67.

Russell, S. J. and P. Norvig (1995) Artificial Intelligence A Modern Approach. Prentice Hall Series in Artificial Intelligence, Prentice-Hall, Inc. New Jersey.

Seberry, J. and J. Pieprzyk (1989) Cryptography: an introduction to computer security. Advances in computer science series. Prentice Hall, New York. p29-34.

Speller, I. and C. Tuck (2001) Strategy and Tactics Amphibious Warfare, Spellmount Ltd, Staplehurst.

Sweeney, P.E. and E.R. Paternoster (1992) Cutting and packing problems: a categorized, application-orientated research bibliography. *Journal of the Operational Research Society*, **43:** 691-706.

Tovey, C.A. (2002) Tutorial on Computational Complexity. *Interfaces*, **32** (3), 30-61.

Vasko, F.J., J.A. McNamara, D.D. Newhart and F.E. Wolf (1994) A practical solution to a cargo loading problem at Bethlehem Steel. *Journal of Operational Research Society*, **45:** 1285-1292.

DISTRIBUTION LIST

Efficient Solutions in Load Planning

Jane Sexton , Jamie Watson and Sharon Boswell

AUSTRALIA

DEFENCE ORGANISATION

Task Sponsor	
CANAASG	**1**
S&T Program	
Chief Defence Scientist	shared copy
FAS Science Policy	
AS Science Corporate Management	
Director General Science Policy Development	
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	1
Scientific Adviser - Army	Doc Data Sht & Dist List
Air Force Scientific Adviser	Doc Data Sht & Dist List
Scientific Adviser to the DMO M&A	Doc Data Sht & Dist List
Scientific Adviser to the DMO ELL	Doc Data Sht & Dist List
Systems Sciences Laboratory	
Chief of Maritime Operations Division	Doc Data Sht & Dist List
Chief of Land Operations Division	Doc Data Sht & Dist List
Research Leader Littoral Warfare	1
Head Maritime Operations and Tactics Group	1
Jane Sexton	1
Jamie Watson	1
Sharon Boswell	1
DSTO Library and Archives	
Library Edinburgh	1
Defence Archives	1
Library, Sydney	Doc Data Sheet
Library, Stirling	Doc Data Sheet
Capability Development Group	
Director General Maritime Development	Doc Data Sheet
Director General Land Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Director Trials	Doc Data Sheet

Chief Information Officer Group

Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
Director General Information Services	Doc Data Sheet

Strategy Group

Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
Assistant Secretary Strategic Policy	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet

Navy

SO (SCIENCE), COMAUSNAVSURFGRP, NSW	Doc Data Sht & Dist List
Maritime Operational Analysis Centre, Building 89/90 Garden Island Sydney NSW	Doc Data Sht & Dist List
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet

Air Force

SO (Science) - Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data Sht & Exec Summ

Army

ABCA National Standardisation Officer, Land Warfare Development Sector, Puckapunyal	e-mailed Doc Data Sheet
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data & Exec Summ
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	Doc Data Sheet
SO (Science) – 3 Brigade, Lavarack Barracks, QLD	Doc Data Sheet

Intelligence and Security Group

DGSTA Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1 (PDF)
Assistant Secretary Capability Provisioning	Doc Data Sheet
Assistant Secretary Capability and Systems	Doc Data Sheet
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	Doc Data Sheet

Defence Materiel Organisation

Deputy CEO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet

Head Materiel Finance	Doc Data Sheet

Defence Libraries

Library Manager, DLS-Canberra	Doc Data Sheet
Library Manager, DLS - Sydney West	Doc Data Sheet

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy	
Library	1
Head of Aerospace and Mechanical Engineering	1
Serials Section (M list), Deakin University Library, Geelong, VIC	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	2
UK Dstl Knowledge Services	2
Canada Defence Research Directorate R&D Knowledge & Information Management (DRDKIM)	1
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5

Total number of copies: Printed 33 + PDF 1 = 34

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION **DOCUMENT CONTROL DATA**	 1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE Efficient Solutions in Load Planning	3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)
4. AUTHOR(S) Jane Sexton, Jamie Watson and Sharon Boswell*	5. CORPORATE AUTHOR Systems Sciences Laboratory PO Box 1500 Edinburgh South Australia 5111 Australia

6a. DSTO NUMBER DSTO-TN-0587	6b. AR NUMBER AR-013-207	6c. TYPE OF REPORT Technical Note	7. DOCUMENT DATE April 2004

8. FILE NUMBER 490/6/131	9. TASK NUMBER NAV 01/027	10. TASK SPONSOR COMANAASG	11. NO. OF PAGES 31	12. NO. OF REFERENCES 24

13. URL on the World Wide Web http://www.dsto.defence.gov.au/corporate/reports/DSTO-TN-0587.pdf	14. RELEASE AUTHORITY Chief, Maritime Operations Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Amphibious operations, Load tests, Amphibious ships, Littoral Zone, ORBAT

19. ABSTRACT

The successful execution of an amphibious operation is a function of the efficient discharge of the ORBAT to the relevant points of entry in a given timeframe. Here, we consider a sea point of entry only where the planning phase considers the loading of the ORBAT onto watercraft.

The current load planning capability is achieved through a conference and the use of a trim and stability tool. In order to make this process more efficient – both in time and accuracy – a new method was required by the ADF. As a result, DSTO developed a command decision aid, the Littoral Battle Tool Set, LBaTS.

This report describes the loading algorithm as required by the landing phase and then extends the scope to the termination phase where the ORBAT is returned to the amphibious ship. Each method will be demonstrated for a fictitious component of an ORBAT.

Finally, upon introduction of a command decision aid into the distributed architecture environment, we recommend incorporating the methods described here to enhance the planning capability of the ADF.