A two-dimensional extension of Allen's temporal logic as an intelligent support for the solution of packing problems

A two-dimensional extension of Allen's temporal logic as an intelligent support for the solution of packing problems

Gerrit K. JANSSENS[1]

[1]Operations Management and Logistics
Hasselt University
Campus Diepenbeek - Agoralaan
B-3590 Diepenbeek, Belgium
e-mail: *gerrit.janssens@uhasselt.be*

Abstract

Inland vessels move goods along canals and rivers and they visit ports. Due to tides on the rivers, the vessels make use of locks to enter ports or canals. From a port management point of view, a fast access to and from the port and high utilisation of a lock are important objectives. The policy to access the lock works as follows. Vessels wait in front of the lock. A port administrator assigns a place in the lock based on the knowledge of the vessels' dimensions. As such, there is no FIFO-discipline, but a 'group-FIFO'-discipline, i.e. if *n* vessels are allowed to the lock, they are the first *n* vessels in the arrival queue. A heuristic algorithm is formulated for the placement of vessels in the lock. This algorithm supports the decision where to place the vessel in the lock, aiming to place as many as possible vessels from the arrival queue. The paper describes the implementation of the heuristic which is based on a two-dimensional extension of Allen's temporal logic.

Keywords: packing problem, heuristic, temporal logic.

1. Introduction

A port is a complex industrial system, which consists of different entities and services of which the components continuously are in change. The changes are caused by internal factors (e.g. harbour policy rules) and by external factors (e.g. evolution in types of trade), which influence the entities. Examples of such entities are: vessels, canals, space, quays, companies, locks, warehouses, and financial entities as costs and revenues. Each entity has its own characteristics, making the whole a complex world (Hassan 1993). From an organisational point of view many entities in the harbour are involved making the harbour activities that intensive that they are hard to manage. The demand for decision support is large. The complexity makes that the study of a harbour system is difficult without a computer simulation program (Bruzzone et al. 1999).

When river traffic is involved, also the use of locks comes in to the dynamics of shipping. The application under study also considers a river system and a river port. The port of Antwerp is one of the major ports in the world. It is an upstream urban port on the river Scheldt, nearly one hundred kilometers inland of Belgium. The access to the port goes through a number of locks.

A traffic simulation model for the river Scheldt has been developed and is described in Thiers and Janssens (1998). The simulation model is the object of a study, which has been ordered by the Belgian Waterways regarding the plans to build a second container quay on the river Scheldt just downstream (i.e. North of) the locks of Berendrecht-Zandvliet. For the planned quay, these manoeuvres have to take place on an area with very dense traffic. This would result in hindrance for the other traffic on the river. Opinions amongst officials were divided about this hindrance, resulting from future increasing traffic.

Inland shipping depends on the use of locks. They serve shipping operations by maintaining adequate depths on waterways. In some countries locks also serve water management purposes. Additionally, by maintaining water in canals at different desired levels, locks conserve large amounts of water that can be used for various purposes (Bolten 1980).

2. Lock operations in a port model

Vessels passing through a lock take more time than they would if the lock were not there. This delay consists both of the time waiting to enter the locks as well as the time spent within the lock during a cycle. Water management increases delays at locks. While vessel operators would like frequent lock cycles, water management requires the opposite.

The design of a lock requires many different aspects, including construction and cost aspects, but also security, efficiency and reliability as well as influence on environment and society. For our purposes, we are more interested in the efficiency aspect, which can be translated into a fluent, uninterrupted traffic flow. Traffic should be organised in such a way they do **not** hinder each other too much. (Mc Cartney 1998).

Lock operations are described shortly. Vessels approaching the lock moor while waiting to enter the lock. If the lock is open, they enter and tie up inside the lock until all waiting vessels are entered or the lock is full. Afterwards the doors are closed and the water level within the lock is raised or lowered up to the level with the waterway on the opposite side. When the levels are equal, the doors open again and the vessels leave the lock. The lock can be cycled back in the other direction.

A lock complex can consist of one or more locks arranged in parallel. A lock operator will direct the vessels to a specific lock. The choice depends on: vessel and lock dimensions, lock availability, weather conditions, safety considerations and queue lengths. Normally vessels enter locks in their order of arrival. Exceptions can exist in order to pack the lock for maximum capacity.

The port of Antwerp is an inland port and is vulnerable to tide differences of on average 4.35 metres. These differences hinder efficient goods handling. Lots of harbour activities take place at constant water level within a set of docks, within which the constant water level is secured by a complex of locks. In the port of Antwerp six locks offer access to the docks.

The locking operations are briefly explained using Figure 1. The text in the figure should be understood as: 'SCHELDE' = the name of the river Scheldt in Dutch; 'SAS' means lock; 'DOKKEN' means docks.

Assume a vessel arrives at low tide at the lock doors from the river Scheldt side (part A). For the vessel to enter the lock, the lock doors at the riverside should be opened. This can only be done if both water levels are equal. In this case openings are made so that the water in the lock chamber runs into the river. When levels are equal, the doors are opened and the vessel can enter the lock (part B). After the doors on the riverside are closed and the vessel is tightened to the lock walls, openings are made along the doors on the dock's side, allowing the water to flow from the docks into the lock (part C). When levels are equal the doors at the dock's side can be opened and the vessel can enter the port through the dock (part D) (Dehousse 1991).

In many practical cases it happens that more than one vessel is waiting to enter the port and that a lock can harbour more than one vessel. The location of more than one vessel during one set of lock operations is directly linked to efficiency. The more vessels are allowed to enter at once, the shorter the waiting times (Bolten 1981, pp.97-99).

Lock capacity is not an easy concept to define. Capacity could be thought of as the case when, in subsequent locking cycles, no waiting times occur and the lock is completely filled. But the number of vessels, which can take place in the lock, depends on the distribution of vessel types, which arrive and their arrival pattern. Both aspects are of random nature. The randomness can be thought of in two ways: either it relates to the number of vessels which can pass the lock in a fixed time period (e.g. a week), or it relates to the static chamber capacity, i.e. the maximum number of vessels in the chamber.

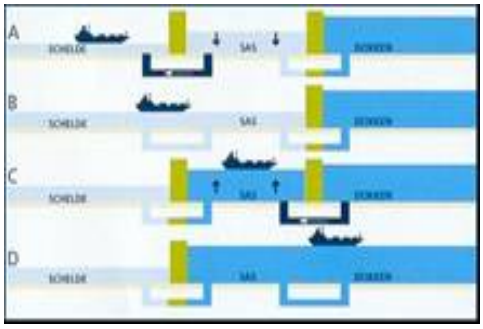


Figure 1: A sequence of locking operations

3. Placement of vessels in a lock

The placement of vessels in a lock resembles a classical OR problem, called the *packing* problem (Janssens 1998, 2002). It is an optimisation problem concerned with finding a best arrangement of multiple items in larger containing regions. The usual objective is to maximise material utilisation and hence to minimise the 'waste' area. In our case, the items are vessels. The containing region represents the lock. The waste area represents the surface of water not covered by vessels. Implicitly, we assume by minimising the waste area the same objective as putting a maximum number of vehicles in the lock. A review on packing problems can be found in Dowsland and Dowsland (1992).

The placement of vessels is a two-dimensional (2D)-rectangular packing problem. A set of vessels has to be placed in a rectangular object, the lock. The lock is modelled as a rectangle with length L and width W where $L > W$. A vessel is also modelled as a rectangle with length l_v and width w_v where $l_v > w_v$. The placement process ensures there is no overlap between the items. In Figure 2 the placement with knowledge of the real shape of the vessels would be acceptable. With the rectangular assumption this placement is not acceptable.

There are many variants of the 2D-rectangular packing problem. Dyckhoff (1990) has given a topology of packing problems in terms of: dimension of the problem, the type of assignment, the type of items, the region (single object) in which the objects have to be placed, the rotation of items and the guillotine feature.

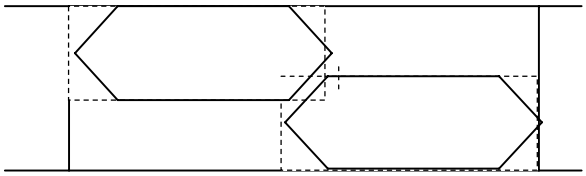


Figure 2: Placement of two vessels with perfect information on shapes and their rectangular approximations

The *dimensionality* equals the minimal number of dimensions to characterise the problem. As our case deals with vessels of specific length and width to be placed in a lock with a specified length and width, we deal with a two-dimensional problem. The *type of assignment* describes whether all objects have to be assigned to a container. In our case, all vessels waiting in front of the lock have to be assigned to one or the other locking operation. *The type of items* relates in our case to the vessels, which are represented as rectangular objects. The *region* is also a rectangle with finite length and width. The aspect of *rotation* relates to the possibility of rotating items by 90°. In the lock the vessels cannot be rotated and as the length is assumed to be larger than the width of the vessel, the larger part of the vessel is placed parallel to the length of the lock

Baker, Coffman and Rivest (1980) consider packings, which are orthogonal and oriented. An *orthogonal* packing is one in which every edge of every rectangle is parallel to either the bottom edge or the vertical edges of the region. An orthogonal packing is also *oriented* if the rectangles are regarded strictly as ordered pairs; i.e. a rectangle (x_i, y_i) must be packed in such a way that the edges of length x_i are parallel to the bottom edge of the region. Thus rotations to a degree of 90° are not allowed. This feature is certainly applicable to our case. Another type of constraint considered in packing is the guillotine packing. The *guillotine* feature relates to a restriction of the partitioning of the lock area into rectangles. Guillotine cuts are cuts from one edge of a previously cut rectangle to the opposite edge. Cuts are assumed to be orthogonal.

Dowsland (1992) makes a difference between the problems where the complete list of objects is known or not known at the moment of decision. If the complete list is known, the problem is called **an** *off-line problem*. If it is not the case, it is called an *on-line problem.* In our case we deal with an off-line problem.

Looking only at *two-dimensional rectangular orthogonal bin packing problems* (2BP), Lodi, Martello and Vigo (1999) characterise according to the orientation and guillotine patterns. Four classes are described:

- 2BP/O/G: the objects are oriented (O) and guillotine cuts (G) are required;
- 2BP/R/G: the objects can be rotated (R) and guillotine cuts (G) are required;
- 2BP/O/F: the objects are oriented (O) and the cutting patterns are free (F)
- 2BP/R/F: the objects can be rotated (R) and the patterns are free.

Our planning problem falls within the class: 2BP/O/F.

As expected, the two-dimensional rectangular bin packing problem is NP-complete, so we have to rely on heuristic methods to solve the problem. A classification of heuristic algorithms for this problem can be made as (Lodi et al. 2002):

- one-phase algorithms: they place the objects immediately in a finite set of containers;
- two-phase algorithms: they place the objects in a strip (i.e. a container with finite width but infinite length). In a second phase the strip solution is used to place the objects in to a set of finite-length containers.

Our interest goes to the class of one-phase algorithms, in which in some way the sequence of arriving vessels has to be respected. In a two-phase algorithm this can also happen, but we must take care that the last arrived vessel is not taken in the packing if another one is excluded. One-phase algorithms can be sub-divided into two classes: *level* algorithms or *non-level* algorithms. In level algorithms the objects are assigned within the container into different levels. For our application a non-level algorithm is most applicable.

Within the class of non-level algorithms we mention three heuristics, of which one will be implemented afterwards. Within this idea, a first heuristic has been formulated called the *bottom-up left-justified* (BL) algorithm (Baker et al. 1980, Liu and Teng 1999). Such an algorithm puts the vessels one at a time as they are drawn in sequence from the list of arriving vessels.

When a vessel is placed in to the lock it is first placed into the lowest possible location (along the lock's walls, closest to the lock doors at the port side), and then left-justified at this vertical location in the lock (along the lock doors, closest to the left wall). A second heuristic is called the *alternative directions* algorithm (Lodi et al. 1999, pp. 345-350). Based on the length and width information of the vessels, a lower bound on the required number of locks is computed. At the front side of the locks a number of vessels are placed using the Best Fit Decreasing Height-principle. A vessel is placed, most to the left, where it fits and where the unused length is minimal. If, after this left-to-right placement, the right wall is reached, the procedure is continued but now from right to left. A third heuristic is called the *less-flexibility-first heuristic*, which will be dealt with in detail.

4. The Less-Flexibility-First-Heuristic

The less-flexibility-first heuristic is chosen for implementation as it generates dense packings with short computation time. The name of the algorithm refers to the fact that priority is given to placements with low flexibility either of the lock or of the vessels to be placed (Wu et al., 2002).

The flexibility of the lock can be looked at in three ways: corners have the least flexibility, the sides (walls and doors) come next, and the open spaces have the most flexibility. The levels of flexibility can be intuitively interpreted as follows. A corner is defined by two sides and the object, which is placed in a corner, has only one place to be positioned, so there is no choice and by this no flexibility. An object, which is placed along one side, can move along that side provided it is not placed in a corner. The degree of freedom is of a one-dimensional type. If an object is placed in an open space, it can move in any direction as long as it is not placed along a wall side or in a corner. Its degree of freedom is of the two-dimensional type.

The flexibility of objects depends on the shape and the size. In our case only rectangles are used, so it can be stated that larger objects show less flexibility.

From figure 3, it can be seen that object A can be placed anywhere in the free area whereas object B fits only in the central part.

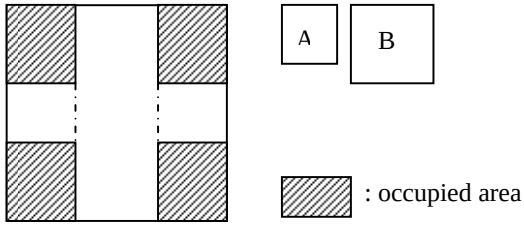


Figure 3: Example of flexibility of objects A and B

The heuristic makes use of a specific coordinate system (x_{b1}, y_{b1}, x_{b2}, y_{b2}) with (x_{b1}, y_{b1}), the coordinates of the lower-left corner of the lock, and (x_{b2}, y_{b2}), the coordinates of the right upper corner of the lock. The *m* vessels to be placed have width and length using the symbols (w_i, l_i).
The problem consist of finding a solution, which consists of *m* coordinate sets: (x_{11}, y_{11}, x_{12} y_{12}), …, (x_{m1}, y_{m1}, x_{m2}, y_{m2}), where $x_{i1} < x_{i2}$ and $y_{i1} < y_{i2}$ for all $1 \le i \le m$ and where ($x_{i1,}$ y_{i1}) represent the coordinates of the lower left corner of the vessel placement and (x_{i2}, y_{i2}) represent the coordinates of the right upper corner of the vessel placement in the lock. The *m* coordinate sets satisfy the following constraints:

1 $(x_{i2} - x_{i1}) = w_i$ and $(y_{i2} - y_{i1}) = l_i$,
2 for all $1 \le i, j \le m$, at least one of the constraints is satisfied:
$y_{j1} \ge y_{i2}$, $y_{i1} \ge y_{j2}$, $x_{j1} \ge x_{i2}$, $x_{i1} \ge x_{j2}$,
3 for all $1 \le i \le m$,
$x_{b1} \le x_{i1}$, $x_{i2} \le x_{b2}$ en $y_{b1} \le y_{i1}$, $y_{i2} \le y_{b2}$.

The first constraint takes care that the surface taken by the vessel in the lock corresponds to the size of the vessel. The second constraint avoids that vessels overlap and the third constraint takes care that the vessels are placed with the boundaries of the lock. The left lower corner of the lock is defined as the zero point of the coordinate system.

The algorithm works with corner-occupying decisions (COD). This means that a decision is taken to put the vessels in a free corner. Given the 4-tuple (w_i, l_i, x_{i1}, y_{i1}) where x_{i1} and y_{i1} represent the placement coordinates of the lower left corner of the vessel, the complete occupied area can be determined. At each iteration the algorithm generates a list of all candidate COD, given the current situation of definitively placed vessels. One object can lead to several COD.

Figure 4 shows that for one vessel several (in this case eight) COD might be generated.

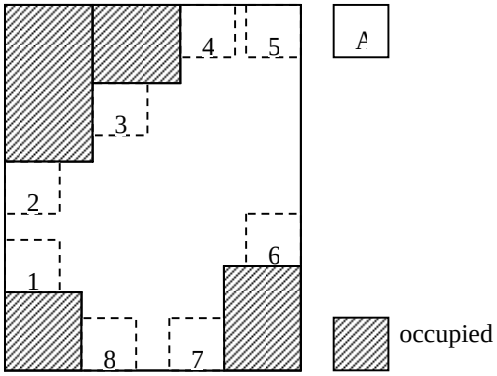


Figure 4: Candidate corner-occupying decisions for vessel A to be placed

The list of COD candidates is sorted on decreasing length, and in case of ties, on decreasing width. For each candidate COD a function value is calculated. This is done by allocating the candidate COD in the lock, followed by locating all other COD (if feasible) in order of the COD-list (the so-called greedy candidates). This step in the heuristic is called the pseudo-packing of the candidate CODs.

A pseudo-packing of an element of the list is defined by the packing of the element (the vessel in a specific place) and as many other vessels as possible, packed in a greedy way, i.e. by running down the list and provisionally packing as soon as it is feasible. For such a packing solution, a function value is calculated, defined by the sum of the surfaces occupied by the provisionally packed vessels. The candidate COD with highest function value is selected for definitive placement. Note that only the candidate COD is placed definitively and not all others, which were provisionally placed by the greedy procedure. Larger objects, with less flexibility, are prioritized by this procedure. The process is repeated until either all vessels are placed or no further vessels can be placed in the remaining space. It should be clear that, during the pseudo-packing procedure, objects cannot be placed in corners which are occupied by other objects during the same pseudo-packing run.

The procedure is illustrated by means of an example, inspired by Wu et al. (2002), but adapted to our special case, shown in Figure 5. Assume that the vessels A (width=2, length=5), B(width=4,length=4) and C (width=4,length=6) are to be placed in a lock of length 8 and width 8. In a previous step of the algorithm the vessel P (3,6) has been placed. The candidate COD list can have at maximum 3 * 5 = 15 elements, i.e. the number of remaining vessels times the number of corners. The number of elements in the list will mostly be lower due to infeasibilities in location, e.g. vessel A cannot be placed in corners 4 and 5., or due to identical placements, e.g. the placements of a vessel with length 2 and width 2 in corners 4 and 5 are identical.

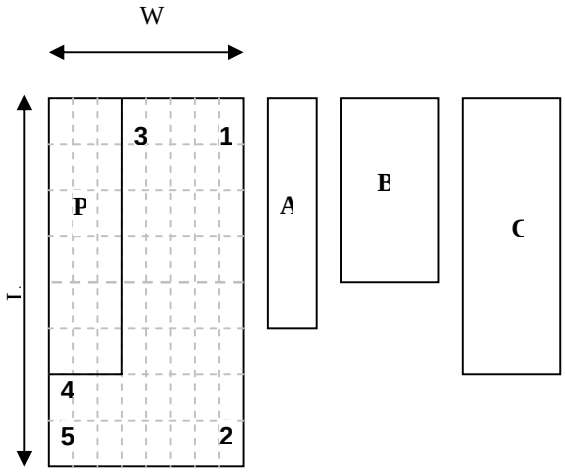


Figure 5 : Example of the calculation of a function value

Let us define the coordinate of the left-lower corner to be (0,0). Feasible placements, sorted on decreasing vessel length and, if required, on decreasing vessel width are shown in Table 1. For each candidate a function value is calculated. According to the earlier described procedure we work this out for the first COD candidate in the list with coordinate system (4,6,4,2). The pseudo-packing starts from a situation as shown in figure 5. From this situation it is checked whether remaining elements on the list can be place in a feasible way. By excluding the candidates, which have their placement in corner 1, four candidates remain, but none of them can be placed in a feasible way. The function value of this pseudo-packing equals 42, which is the sum of the surfaces of the vessels P and C. The procedure is repeated for the remaining eight elements on the COD list. The solution with the highest function value is chosen. This example is simple as only one object can be placed. The fitness function values obtained are shown in the third column of table 1.

Vessel C is chosen to be placed in corner 1. In a next iteration it has to be decided which element from the new COD list of vessels A and B has the higher function value. In this example, it can be seen that neither A nor B can be placed and by this, the algorithm stops. A description of our heuristic, adapted from Wu et al. (2002), is listed below:

Action	*Coor-dinate system*	*Fitness Function Value*
Vessel C placed in corner 1	(4,6,4,2)	42
Vessel C placed in corner 2	(4,6,4,0)	42
Vessel C placed in corner 3	(4,6,3,2)	42
Vessel A placed in corner 1	(2,5,6,3)	28
Vessel A placed in corner 2	(2,5,6,0)	28
Vessel A placed in corner 3	(2,5,3,3)	28
Vessel B placed in corner 1	(4,4,4,4)	34
Vessel B placed in corner 2	(4,4,4,0)	34
Vessel B placed in corner 3	(4,4,3,4)	34

Table 1: Function values for the example of the heuristic

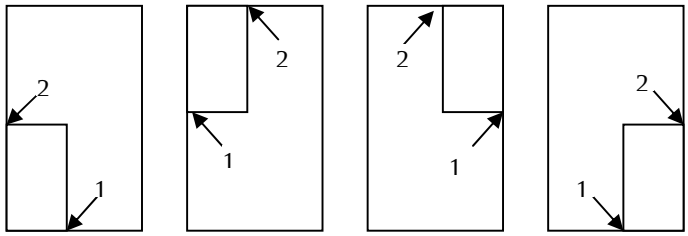
Start with an empty workspace.

1. Based on the current placing information, find all possible CODs for each vessel remaining to be placed. Represent each COD by a quadruple <length, width, x_1, y_1>.
2. Sort all these quadruples in lexicographical order, i.e. in decreasing length, and if required by ties, in decreasing width.
3. For each of these COD, do 3.1 till 3.3 to calculate its fitness function value (FFV).
   3.1. Pseudo-place this COD.
   3.2. Pseudo-place all the remaining vessels based on the current COD list and with a greedy approach, until no more COD can be packed.
   3.3. Calculate FFV of this candidate COD as the occupied area.
   Note: before the pseudo-packing for the next candidate COD is tried, one needs to remove the previous pseudo-packed CODs.
4. Pick the COD candidate with the highest COD and really place the corresponding vessel according to the COD.
5. Mark the vessel as 'placed'.

Return to Step 1 until no more placements can be made.

5. Implementation Issues

While the idea of the heuristic is straightforward and easy to understand, the major difficulty is found in the generation of free corners. The list of free corners needs to be updated after each definitive placement. To illustrate this, assume that in Figure 4, candidate COD number 6 is chosen to be the definitive one. The free corner in which the vessel is placed disappears from the list but two new free corners (which we call base corners) appear on the list. The new corners can be identified when the type of placement is known. A vessel can be placed right-above (RA), right-under (RU), left-under (LU) or left-above (LA) in the free corner (see figure 5). In the examples in figure 6 two new base corners are generated, but if the width of the vessel would be equal to the width of the free area, only one new corner appears.



in corner RA | in corner RU | in corner LU | in corner LA

Figure 6: Generation of new base corners after a placement in a free corner

As an illustration we also show some other side effects, which can occur by placement of a vessel, especially when length, width or a corner of the placed vessel touches other vessels, which were placed before. Such an example is shown in Figure 7. After a first placement of vessel 4 in corner 1 with coordinates (0,0), two new base corners are generated: (4,0) and (0,6). Assume that in a second iteration vessel 1 is placed in corner 3. This placement generates new basic corners (8,5) and (4,10). But additionally two other corners should be generated: (4,5) and (4,6). Many of those special cases may occur.

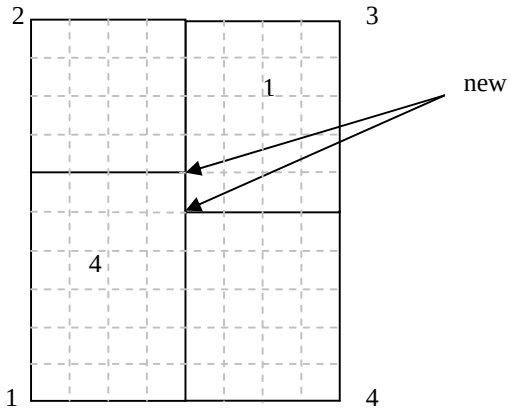


Figure 7: Generation of extra corners after a placement

Therefore a formal framework needs to be developed to enumerate all special cases and their consequences in terms of corners to be generated or deleted.

Such a logic can be based on a formal framework using relations between time intervals (Allen 1983). As, after placement, the length and width side of a vessel can be modelled as finite intervals, also this type of relations are exhaustive and can be enumerated, be it in two space-dimensions and not in one time dimension. If two intervals X and Y in one dimension are considered, Allen (1983) defines seven types of relations from X to Y. Six more relations can be defined from Y to X (one less because of the relation 'equals' – see list). For the illustration, we focus on the length dimension, but the reader should be aware that all thirteen relations also exist on the width dimension. They are:

- X before Y: X is placed before Y, i.e. some place is left between both vessels,
- X meets Y: the end of X coincides with the beginning of Y,
- X equals Y: both the beginnings en ends of X and Y coincide,
- X overlaps Y: the beginning of X lies before the beginning of Y, but the end of X is not before the beginning of Y,
- X during Y: the beginning of X lies after the beginning of Y and the end of X lies before the end of Y,
- X starts Y: the beginnings of X and Y are the same but the end of X lies before the end of Y,
- X finishes Y: the ends of X and Y are the same but the beginning of X lies before the beginning of Y.

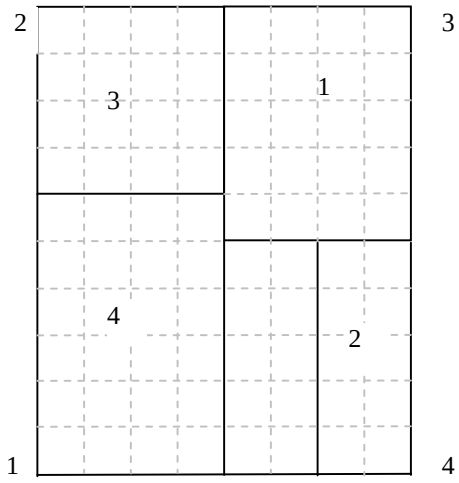


Figure 8: Illustration of the test on existing and new corners

This logic is illustrated by means of the case in Figure 8.

Assume vessels 1,2 and 4 have already been placed, and in a last step vessel 3 is placed in corner 2. The program logic, implementing Allen's ideas in two dimensions, generates the following output, which can easily be checked on Figure 8.

```
Potential basic corners
-----------------------
(  0,  6)
(  4, 10)

Check whether existing corners are
made unavailable by placement
-----------------------------------
---------------------------

available corner (0,6) occupied by
vessel 3
available corner (4,6) occupied by
vessel 3
available corner (4,10) occupied by
vessel 3

Test of potential basic corners and
of newly made corners
-----------------------------------
--------------------
4 meets    3 in length at left side
3 equal    4 in width at upper side
4 meets    3 in length at right side
3 finishes 1 in length at left side
new corner = (4,6)
newcorner (4,6) is occupied by
vessel 4
3 meets    1 in width at upper side
basic corner (0,6) is occupied by
vessel 4
basic corner (4,10) is occupied by
vessel 1
```

6. Conclusion

The heuristic implements some human ideas in problem solving, which may ease the acceptance by the operators in the port. Furthermore the heuristic needs only very limited computer time, so it can advise the lock responsible even in very busy working environments. The heuristic shows its limitations in terms of shape of the vessel, using only rectangular shapes. It needs further integration in a higher level management system, helping the responsible to decide whether to wait for an extra vessel to have a better occupation of the lock or to go for less average waiting times for the vessels already moored in the lock.

7. References

Allen J.F. 1983. Maintaining knowledge about temporal intervals, *Communications of the ACM*, vol. 26 no. 11, pp. 832-843.

Baker B.S., Coffman E.G. and Rivest R.L. 1980. Orthogonal packing in two dimensions, *SIAM Journal on Computing*, vol. 9 no. 4, pp. 846-855.

Bolten J.G. 1980. *Inland shipping and water management decisions in the Netherlands.* Ph.D. Thesis, The Rand Graduate Institute, Santa Monica, California.

Bolten J.G. 1981. *Policy Analysis of Water Management for the Netherlands – vol. IX: Assessment of the Impacts on Shipping and Lock Operations.* Report RAND/N-1500/9, The Rand Graduate Institute, Santa Monica, CA.

Bruzzone A.G., Giribone P. and Revetria R. 1999. Operative requirements and advances for the new generation simulators in multi-modal container terminals. In: *Proc. of the 1999 Winter Simulation Conference* (Phoenix, Arizona), pp. 1243-1252.

Dehousse N.M. 1991. *Exploitation and Management of the Inland Navigation – Locks*, Institut de Formation Internationale aux Transports, Université de Liège, Liège, Belgium.

Dowsland K.A. and Dowsland W.B. 1992. Packing problems, *European Journal of Operational Research*, vol. 56, pp. 2-14.

Dyckhoff H. 1990. A typology of cutting and packing problems, *European Journal of Operational Research*, vol. 44, pp. 145-159.

Hassan S.A. 1993. Port activity simulation: an overview, *Simulation Digest*, vol. 23 no. 2, pp. 17-36.

Holguin-Veras J. 1998. Towards new paradigms of modelling container terminals, *Infrastructure*, vol. 3 no. 2, pp. 3-9.

Janssens G.K. and Tielemans W. 1998. Embedding a lock filling optimisation model into a river port simulation model. In Y. Merkuryev, A. Bruzzone and L. Novitsky (eds.), *Proc. of the International Workshop on Modelling and Simulation within a Maritime Environment*, September 1998, Riga, Latvia, pp. 43-48.

Janssens G.K. 2002. Increasing lock throughput in an inland port through simulation-optimisation, *JORBEL (Journal of the Belgian Operations Research Society)*, vol. 40 no.3-4, pp. 235-247.

Liu D. and Teng H.F. 1999, An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles, *European Journal of Operational Research*, vol. 112, pp. 413-420.

Lodi A.S., Martello S. and Vigo D. 1999. Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems, *INFORMS Journal of Computing*, vol. 11 no. 4, pp. 345-357.

Lodi A.S., Martello S. and Vigo D. 2002. Recent advances on two-dimensional problems, *Discrete Applied Mathematics*, vol. 123 no. 1-3, pp. 379-396.

McCartney B.L., George J., Lee B.K., Lindgren M. and Neilson F. 1998. *Inland Navigation: locks, dams, and channels*, American Society of Civil Engineers (ASCE), Manuals and Reports on Engineering Practice no. 94.

Thiers G.F. and Janssens G.K. 1998. A port simulation model as a permanent decision instrument, *Simulation*, vol. 71 no. 2, pp. 117-125.

Wu Y.-L., Huang S.-C., Wong C.K. and Young G.H. 2002. An effective quasi-human based heuristic for solving the rectangle packing problem, *European Journal of Operational Research*, vol. 141, pp. 341-358.