The Potted Plant Packing Problem

Towards a practical solution

René Schumann and Jan Behrens

OFFIS, Escherweg 2 26121 Oldenburg Germany
rene.schumann|jan.behrens@offis.de

Abstract. The potted plant packing problem was presented in [1] as a practical planning problem. The plants are packed on trolley for transportation and the transportation costs directly depends on the number of trolleys. As a result the effective packing of trolleys is an important practical problem. Effective packing of trolleys becomes even more import as this sector evolves to a consumer market.
In this paper we present our work towards solving this problem. We designed a framework algorithm and developed a prototype capable of solving a simplified version of the problem. Here we focus on the algorithms and the prototype that were developed so far.

1 Introduction

The transportation of plants is comparably expensive as they require careful treatment due to there sensitivity. For standardized transport, potted plants are loaded on transport trollies (shown in figure 1). The cost of transportation depends on the number of these trollies. In order to minimize transportation costs, effective packing of trollies is necessary. The need for optimize resources increase as this sector evolves to a consumer market. For that reason ongoing research evaluates strategies for this sector [2].
In this paper we discuss the potted plant packing problem which is a special 3D bin packing problem. It is a practical planning problem one of our costumers asked us to solve. The problem was first introduced in [1]. So far we worked on a solution that can be integrated into or supplement the software of our costumer. The scientific challenge of this packing problem is not just solving this problem, it is although already difficult to formalize the problem description in an accepted representation. So first we give a short summery of the problem statement. After that we present our steps towards a formalization of the problem. Thereafter in section 3 we present the methods developed so far. On these methods the prototype, presented in section 4, base on. Finally we discuss further extensions needed for practical use.

2 The packing problem

2.1 Problem statement

The task is to compute a packing instruction for a given transportation order. This packing instruction should assigns every plant an exact placement on a layer and for each layer an exact placement (mounting height) into a trolley. To clarify the problem a trolley is shown in figure 1. It is allowed to stack plants

Fig. 1. trollies for plants, picture taken from [3]

on a layer. The placement of layers into trolleys has to respect the stability of trollies and the effective usage of the space offered by the truck in which the trollies are transported.

2.2 Towards a formal representation of the packing problem

A first step towards a formal description of the potted plant packing problem was proposed in [1]. The problem was presented as a 6-tuple according to the representation of scheduling problems presented in [4]. The 6-tuple consists of

- resources (trollies),
- objects (plants),
- order (defining the quantity of plants to pack),
- hard constraints (e.g. stability),
- soft constraints (contiguously placement of plants of an order item) and
- objective function (minimizing the needed number of trollies).

Another common description for packing problems was presented by Dyckhoff [5]. Dyckhoff pointed out a number of typical characteristics of cutting and packing problems. He advocated to use some of these characteristics to identify similar groups of cutting and packing problems. These characteristics are:

- dimensionality (1, 2, 3, n)
- kind of assignment (B, V)
- assortment of large objects (O, I, D)

– assort of small objects (F, M, R, C)

For a detailed discussion of this notation see [5] or [6]. Thus a problem description of a cutting or packing problem is a 4-tuple describing the problem in respect of the mentioned characteristics.
The potted plant packing problem belongs to the class $3/V/D/R$. This notation encodes that,

– the problem has three relevant dimensions (length, width, height)
– all small objects (plants) has to be placed within the large objects (trolley)
– that the large objects (trollies) can have different figures, (trolley with or without add-on modules)
– there are many small objects of relatively few different figures.

This typology is widely used to describe the main characteristics of cutting and packing problems. Actually an improved typology is discussed by [6]. According to [6] the here discussed problem can be seen as a special type of the *Three Dimensional Multiple Bin Size Bin Packing Problem*. An actual survey about existing literature of packing and cutting problems can be found in [6], too.
The disadvantage of those notations is, that the problem can not be described entirely. Additional constrains has to be added to describe the actual problem. Nevertheless these notations are widely in use because they allow a classification of such problems.

3 Towards a practical solution for the packing problem

As already mentioned it is quit challenging to describe such a practical planning problem in terms of a formal model. Even more challenging is solving such a problem.
To handle the complexity the original problem can, as already discussed in [1], be decomposed into the subproblems

– distribute the plants on layers
– distribute the layers on trolleys

As a consequence of this decomposition a computed solution may be not optimal, but it can be computed faster, due to the reduction of complexity. Both subproblems will be discussed in more detail in the following subsections. But first some techniques to scale down the search space are presented.

3.1 Scaling down the search space

As already stated in the formal representation of this problem in the notation proposed by Dyckhoff there exists a large number of different small objects. Actually we have to deal with at least 1600 different articles. To simplify the problem and make it tractable we build categories of plants with similar dimensions. A category can be seen as a box which can contain different plants with

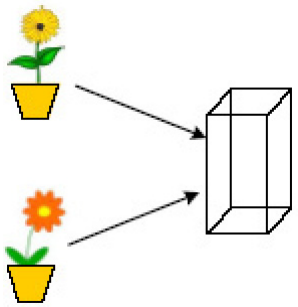


Fig. 2. concept of categories

similar dimensions. This is sketched in figure 2. Building categories necessitate a design decision concerning the number of different categories. Reducing the number of categories implies a decreasing precision but reduce the heterogeneity of figures to pack. One can think of two extrem alternatives.

1. there exists only one category, which implies the maximal wasted space
2. for each group of plants with differing dimensions there exist a category, which implies no wasted space.

To find an appropriate number of categories we advocate to compute the wasted space for a given number of categories and analyzing the gradient of the resulting curve. The graph of such an analysis is shown in figure 3.

To reduce the needed time for computing an packing instruction for a given

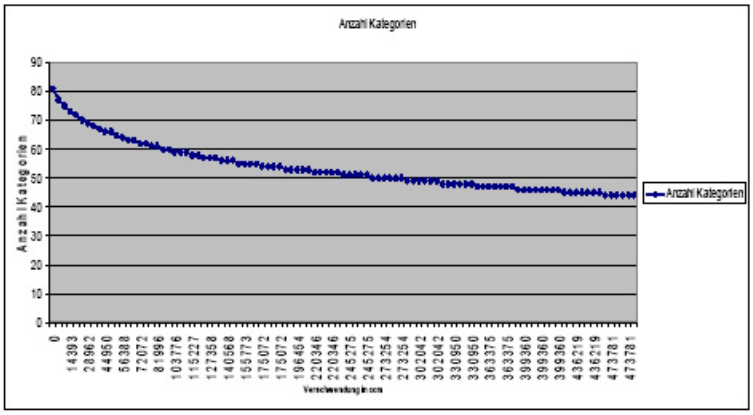


Fig. 3. x-axis: wasted space; y-axis: number of categories

order, we decide to use offline computed packing patterns stored in a database.

A packing pattern represents an entire packed layer. No additional plant of any category already placed on the layer can be added. But it has to be stated that this offline computation needs an enormous amount of time. This is because large number of categories. In our experiments starting with approx 350 different plants we use 59 categories. To compute all possible packing pattern with say 3 different categories would need several years to evaluate nearly 20 million layers. As a consequence the complete computation of all possible packing patterns seems to be impractical. As a result we only computed offline the pattern containing only one category. All other layers, which are computed online are stored into a database. There they can be retrieved for planning future packing orders.

3.2 The framework of an packing algorithm

We describe here the main planning process which is an extended version, of the algorithm presented in [1]. First all plants of an order are stored into queues, a

1. $Ts = \emptyset$ // trolley set
2. $Ws = \emptyset$ // working set of packing units
3. $Ls = \emptyset$ // layer set
4. $Ap = \emptyset$ // applicable packing pattern
5. FOR EACH $o \in O$ // where O is an order and o a order item
   - $P =$ packing-unit(o)
   - $q =$ queueForCategory(category($p \in P$)) //a Queue for each category
   - enqueue(q,p)
6. WHILE NOT($\exists q \in Q; |q| > 0$ OR $|Ap| > 0$)
   - IF ($|Ap| > 0$)
     - p = findBestPattern(Ap)
     - $Ls = Ls \cup$ newEbene(p)
     - $Ws = Ws -$ elementsOf(p)
   - ELSE
     - $Ws = Ws \cup$ deque($q; q \in Q, \forall q' \in Q; |q'| > 0, |q| \geq 0,$ h(q) $\geq$ h(q'))
   - $Ap =$ findapplicable(Ws)
7. IF $|Ws| > 0$
   - $Ls = Ls \cup$ onlineLayerComputations(Ws)
8. $Tr =$ distributeLayers(Ls)

Algorithm 1: packing frame algorithm

queue for each height. The plants that can be respected for packing are stored in a working set. Plants are added to the working set if there exists a non-empty queue and no stored packing pattern is applicable. If a packing pattern can be applied a corresponding layer is introduced and the elements placed on the layer are removed from the working set. This is repeated until all queues are empty and no packing pattern can be applied. If the working set then contains further elements

additional layers have to be computed by the online layer packing algorithm. After that all plants are placed on layers. Then these layers are distributed to trolleys. These packed trollies form the computed packing instruction.

3.3 Packing of layers

For our first prototype we simplified this problem, concerning only round flowerpots. So the packing of layers corresponds to a relativ common packing problem. It can be seen as the problem of packing circles into a rectangular. In the notation of Dyckhoff the problem can formalized as $2/V/I/R$. By this notation it is encoded that

- the problem has two relevant dimensions (length, width)
- all small objects (plants) has to be placed within the large objects(layers)
- that there are identical large objects
- there are many small objects of relatively few different figures.

The height of the plants can be ignored considering the placement of circles into a rectangular. But the height of the plants define the height of a layer. The height of a layer ($h(L)$) is indicated by the height of the tallest plant ($h(p)$) on the layer. This can be stated as

$$h(l) = p_i; 1 \leq j \leq |elements(l)|, h(p_i) \geq h(p_j).$$

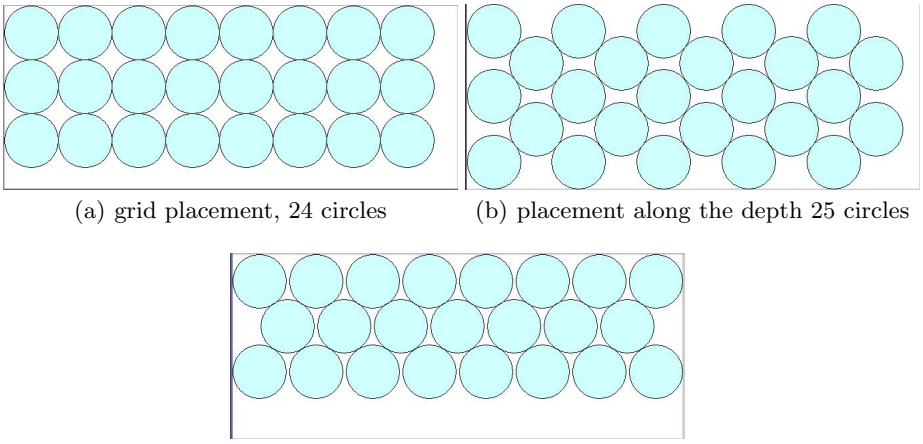
The height of the layers determines the solution quality of the second planning step. As a consequence it is desirable that the summed height of all layers is minimal.

$$h(\mathcal{E}) = \sum_{1 \leq i \leq n} h(E_i)$$

To minimize $h(\mathcal{E})$ the plants are sorted according to their height. The sequence in which the plants are added to the working set depends on this sorting. This effects that layers contains in general plants of similar height. So the summed height of all layers can be minimized.
The problem of packing circles into rectangulars falls into two cases, namely packing circles of equal and unequal size into a rectangular. These cases differ in their complexity and are discussed in two separate sections.

Packing of equal circles into a rectangular Finding an optimal solution for the problem of packing equal circles into a rectangular is a NP-hard task, and in fact optimal solutions are only known for up to 20 circles (see [7] for instance). But there can be designed fast heuristics, which compute solutions with sufficient quality very fast. These heuristics base on a regular placement of circles. We implemented three regular placement strategies, namely

- grid placement
- placement along the depth



(a) grid placement, 24 circles (b) placement along the depth 25 circles

(c) placement along the width, 23 circles

Fig. 4. regular circle placements

– placement along the width

As one can see in figure 4 the number of placements can differ depending on the placement strategy. So far we can not predict which placement strategy will offer the best performance for a given circle and rectangular size. But since these heuristics are very fast we compute always all three alternatives and then choose the best one.

Packing of unequal circles into a rectangular The placement of circles of different size into a rectangular is more difficult. In fact there exists only a few articles dealing with this problem, for example [8], [9], [10] and [11].
We implemented the solutions proposed by [9] and [11]. In our experiments the solution quality using the simulated annealing approach by [9] was not acceptable. We were not able to compute solutions of a quality as presented by the authors. So we have to assume that our implementation is not correct. As a consequence we implemented the maximum hole degree algorithm (B1.0) presented in [11]. The main idea of this algorithm is two place an circle into a corner. A corner can be defined by two sides of the rectangular, a rectangular side and a circle or two circles. The first two circles are placed by a simple placement strategy. Then for each circle not placed already within the rectangular all possible corner placements are computed. The circle which belongs to the corner placement with the minimal distance to another circle or side is chosen. As a consequent a number of existing corner placements become invalid and a number of new corner placements can be defined. A detailed description of the algorithm as well as a complexity analysis can be found in [11]. We are confident with the performances of this algorithm, expect for a large number of circles (about 50)

then the number of possible corner placements grew very fast and the computation time rise. An example of a layer consisting two different circles is shown in figure 5.

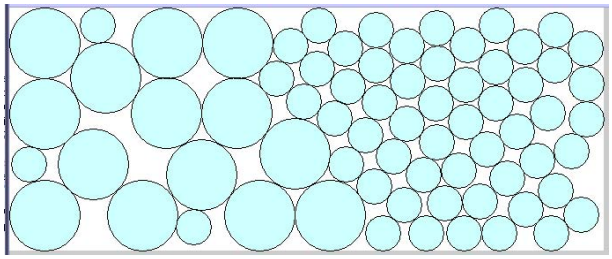


Fig. 5. placement computed by the maximum hole degree algorithm

3.4 Trolley packing

The trolley packing has to answers two questions.

1. Which layers are mounted into which trolley?
2. Where is each layer placed within the trolley?

The first question concerning the distribution of layers to trolleys, which is a classical bin packing problem. This problem can be solved with standard algorithms like next fit and best fit.
The second task is to compute the correct mounting point for each layer. This has to be done in respect of the height and weight of each layer and the layout of an trolley. The layout of the trolley is sketched in figure6. Mounting points are each 5 cm from a base of 20 cm up to a height from 190 cm, without the usage of add-on modules. The strategy for placing the layers within the a trolley is as follows.
The tallest layer is placed on top using the highest possible mounting point ensuring that the total height of the packed CC is just below the maximum allowed hight a truck can store. All other layers are sorted by their weight and sorted in ascending order from top to bottom into the trolley. This strategy aims at to goals. It tries to

- use the available space the truck offers and
- place the center of mass as low as possible for stability reasons.

Due to the use of add-on modules the layout of a trolley changes, but the method for computing the placement of a layer does not need any changes.

4 The implemented prototype

The developed prototype has some limitations, namely these are

- only plants potted in round pots are considered (which is the majority of packed plants)
- no stacking of plants is allowed
- the stored packing pattern contain only pattern of equal circles

The prototype is a eclipse rich client application. A screenshot is shown in figure 6. After an order is specified the packing instruction can be computed and

Fig. 6. Screenshot showing a packed trolley

displayed. **TODO, bla bla**

5 Future work

The next step to improve the computed solution and to generate competitive packing instructions is to allow stacking of plants. To ensure that plants are not demaged by stacking, for every plant a non-stackable area in the center of the plant have to be defined. This is sketched in figure: 7. The existing algorithms have to be extended. Packing algorithms addressing the stacking of circles respecting non-stackable areas are, to the best knowledge of the authors, not addressed in the literature so far.
When stacking is available for our packing algorithms, we are looking forward that it is possible to compute layers with acceptable quality from practical perspective. We then plan to evaluate our solution. Because of the complexity of the problem the evaluation here can only be done by comparing the results of

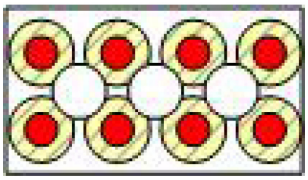


Fig. 7. stacked plants, in red the non-stackable areas are shown

computed packing instructions with real packing data as a benchmark.
As already mentioned we use offline computed packing pattern to improve the computation of packing instructions. We assume that there are going to be a lot of different packing pattern of different quality and different relevance for practical purposes in terms of frequency of usage. To ensure that the time needed for pattern retrieval stays in acceptable boundaries we assume that it is necessary to cache frequent used patterns. Therefore bookkeeping information about the usage and quality of packing pattern is necessary.
The aim of our project is a software potentially integrated into or supplement the software solution of our customer. To offer him more flexibility it is desirable that he can modify stored packing patterns manually.
Most of the plants are potted in round pots. But there exists trays, which have a rectangular shape and offer space for a more then a single plant. As a consequence the packing algorithm has to handle boxes and circles. This results in a further extension of existing algorithms as well as the development of new algorithms. In actual literature the packing of rectangular or circles into a rectangular is focussed (see [12] for packing rectangular and e.g. [11] for packing circles). To the best knowledge of the authors there exists no paper that deals with the problem of packing both circles and rectangulars into a rectangular.
So far we assume that the dimensions of the packed plants are static and inflexible. This is in fact not true for real plants. There exists research for packing flexible circles into rectangular by [13]. To use more realistic models it would be useful to investigate if this extension of our model would improve the quality of packing instructions.

References

1. Schumann, R., Behrens, J., Siemer, J.: The potted plant packing problem. In Sauer, J., ed.: 19. Workshop Plan, Scheduling und Konfigurieren / Entwerfen (PUK). Fachberichte Informatik, Koblenz, Universität Koblenz-Landau Fachbereich Informatik (2005) 1, 2, 3, 5
2. OFFIS: AmmLog, http://www.ammlog.de. (2006) accessable on 21.04.06. 1
3. Foko Lübsen und Sohn Internationale Spedition: Hompage Focko Lüpsen & Sohn GmbH, http://www.luepsen.de/seite01e.htm. (2006) accessable on 21.04.06. 2
4. Sauer, J.: Wissensbasiertes Lösen von Ablaufplanungsproblemen durch explizite Heuristiken. DISKI 37. Infix Verlag (1993) 2

5. Dyckhoff, H.: A typology of cutting and packing problems. European Journal of Operational Research (EJOR) **44** (1990) 145 – 159 2, 3
6. Wäscher, G., Haußner, H., Schumann, H.: An improved typology of cutting and packing problems. Working paper no. 24, (Otto von Guericke Universität Magdeburg) Revision 2006-01-06. 3
7. Erich Friedman: Erich's Place, http://www.stetson.edu/ efriedma/cirinsqu/. (2006) accessable on 19.04.06. 6
8. George, J.A., George, J.M., Lamar, B.W.: Packing different sized circles into a rectangular container. European Journal of Operational Research (EJOR) **84** (1995) 693 – 712 7
9. Correia, M.H., Oliveira, J.F., Ferreira, J.S.: Cylinder packing by simulated annealing. Pesquisa Operacional **20** (2000) 269–286 7
10. Schöning, U., Toran, J., Thierauf, T., Messner, J., Blubeck, U.: Three algorithms for packing variable size bobbins. Report, Abt. Theoretische Informatik Universität Ulm (2002) 7
11. Huang, W.Q., Li, Y., Akeb, H., Li, C.M.: Greedy algorithms for packing unequal circles into a rectangular container. European Journal of Operational Research (EJOR) **56** (2005) 539 – 548 7, 10
12. Scheithauer, G., Terno, J.: The G4-Heuristic for the Pallet Loading Problem. Journal of the Operational Research Society (JORS) **47** (1996) 511 – 522 10
13. Albrecht, A., Cheung, S.K., Hui, K.C., Leung, K.S., Wong, C.K.: Optimal Placements of Flexible Objects Part II: A Simulated Annealing Approach for the Bounded Case. IEEE Transaction on Computers **46** (1997) 905 – 929 10