The Potted Plant Packing Problem

René Schumann, Jan Behrens and Johannes Siemer
OFFIS Escherweg 2, 26121 Oldenburg
rene.schumann—jan.behrens—johannes.siemer@offis.de

Abstract

In this article we describe a practical planning problem, the potted plant packing problem, and our first approaches to solve this problem. The transportation of plants is comparably expensive as they require careful treatment due to there sensitivity. For standardised transport, potted plants are loaded on transport trollies. The cost of transportation depends on the number of these trollies. In order to minimise transportation costs, effective packing of trollies is necessary. This packing problem is herein presented and a first proposal for a solution, based on problem decomposition and multi-dimension bin packing, is given.

1 Introduction

The packing problem presented in this article is a problem a customer of our research group asked us to solve. The transportation of potted plants is comparably expensive, because plants are sensitive products and their size to value ratio can be exceedingly low. The standardised mode of transport for potted plants is on transport trollies. Such a trolley, a so called 'CC-trolley', is shown in figure 1.

As one can see in the picture the plants were placed on layer. These layers can be mounted into a trolley. According to the height of the plants on each layer a trolley can carry a varying number of layers.

The cost of transportation is mainly a function of the number of trollies used for the transport of one particular order. Effective trolley loading therefore is a key requirement for cost reduction. The resulting planning problem is a variation of a 3-D bin packing problem, with a number of additional constraints. We will present this problem in the following section. Thereafter we show a possible solution through problem decomposition. We conclude with a short summary and an outlook on further work in the last section.

Figure 1: A 'CC-trolley' for plants
Picture taken from [1].

2 The packing problem

2.1 Detailed description of the packing problem

This section offers a detailed description of the potted plant packing problem. At first we describe the business process that leads to the transportation of an order:
Initially an order is given. Such an order consists of a number of order items. Each order item relates to a specific article and contains the numbers ordered. Each article, e.g. a potted plant, has various attributes like height, width, depth, weight and type of its packing-unit (e.g. a flowerpot). There are of course further attributes, price for example, they are however neglect here as they are not problem related. The computation of these attributes, especially the dimensions, is not trivial, as the dimensions of a plant are subject to change in time. The available article data is based on the estimated size at a point in time some weeks into the plant season. The actual size at any given time varies from this and has to be estimated with the help of further parameters like seasonal information.
Each plant is packed into a packing-unit, a flowerpot or a tray for example. Consequently a packing-unit can hold one or more plants of one article. The form of a packing-unit can differ from rectangular to circular. Apart from its form, a packing unit also has further attributes relevant to the packing problem, e.g. height, width, depth and weight. In theory, packing-units can differ for one article, depending on the plant supplier for subcontracted part orders. We assume that all plants of an order item were delivered by one supplier and thus have the same type of packing-unit. Such a simplification can be made without restricting the general approach as we could introduce a middle tier that would map packing articles - with only one packing unit associated - to actual articles later.
In practice sometimes subcontracted part orders are supplied by external nurseries. These might arrive pre-packed on cc-trollies at the central packing facility. Such trollies are typically left as

is, because repackaging them would usually lead to a substantial additional effort. Except that, if they have additional space for entire layers available, this might be used for other order items of the same order.

After some pre-processing computation the relevant information for packaging of potted plants can be extracted from an order item. Recapitulating this information comprises of:

- An **order** consisting of a number of order items
- Each **order item** consists of a number of identical packing-units
- **Packing-units**: Each order item consists of a number of packing-units. Depending on the plant within these units. For each packing-unit, information about its dimension, weight and form are given
- **Pre-packed trollies** may combine some order items, on cc-trollies

Our main objective is the computation of a valid packing plan, based on the order data, as well as providing detailed packing-instructions for the staff. Any packing-instruction, and therefore also every packing-plan, consists of a number of trolley-packing-instructions, one for each cc-trolley in the order. Such a trolley-packing-instruction itself consists of a number of layer-packing-instruction. They define the exact position of the layer (mounting height) along with detailed packing instructions for all packing-units on the layer. Each single layer consists of a set of packing-units along with the exact position of every packing-unit on the layer. The position includes information about each packing-units height, because certain articles allow for stacking of further packing-units. This is shown in figure 2 where a second layer of flowerpots is stacked upon a first layer.

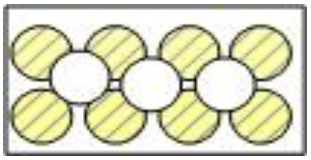

Figure 2: A simple example of stacked flowerpots

The generation of a valid, stable and realisable packing-instruction requires the observance of a number of constraints. Usually a differentiation is made between soft constraints (e.g. keep order items together) and hard constraints like the fixed size of layers or the fact that an item can not span over more than one layer.
When plants are stacked, stability is an important aspect. Another facet of the stack packaging is that it has to be ensured that those items at the bottom are not damaged by the stacked items. This constraint can be formulated as an additional artificial attribute of an article. Each article has an inner region which is of bounds for stacking. If this region has the same size as the packing-unit itself, the packing-unit is called not stackable. This concept is displayed in figure 3. The overall height of a loaded trolley is of course limited too, because the trollies are transported on truck.

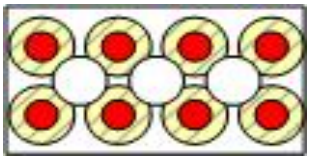

Figure 3: stacked flowerpots with non stackable regions (displayed in dark color)

Thus the available height varies from approx. 240 to 260 cm. For stability reasons the position of a layers in a cc-trolley must correspond to the weight of that layer, e.g. a heavy layer should be placed lower on the trolley. Furthermore each layer and cc-trolley has a maximum payload. The definition of rules leading to legal stacking strategies and so forth and so ensuring the adherence of all constraints, is not a trivial task.

As a further, soft, constraint it is desirable to keep order items together, therefore one order item should be loaded onto one layer - if fit - or at least contiguous onto subsequent layers and trollies. This is due to the fact that usually all packing-units of an order item will be provided together at the loading zone and the number of cc-trollies concurrently in loading is limited.

The overall goal is to provide said packing-instruction for any given order, with the minimisation of needed cc-trollies - and therefore costs of transport - as the objective function.

2.2 The formalised potted-plant packing problem

One approach to describe planning problems is to use a tuple of components, which are capable describing the entire problem. Scheduling problems for example are described in such a way by Keng et al [2] or Sauer [3]. The potted-plant packing problem can formalised in this way, too. This tuple contains the following components.

- **resources (trollies)** It may be doubtable why resources should be modelled explicit. Because this problem is a complex bin-packing problem the task is to place the entire order on trollies, the trollies are virtual unlimited. But there exist two reasons for introduce the resources in the problem description.
  At first it is possible to use an add-on module with the trolley to increase the height of an trolley. So far it is not known if the number if of add-on modules is limited. Second one has to respect the existence of pre-packed cc-trollies. These trollies may not loaded entirely. Thus it is possible that the planning start with a number of some pre-packed trollies with limited capacity.

- **objects (plants)** This are the objects which should be packed by the algorithm. In concrete we do not focus on plants itself. Instead we use an artificial entity called packing-unit. We use this simplification because a typical instance of the potted-plant packing problem has about 50.000 different articles. Due to the fact, that we use the abstraction of packing-units the number of different entities the problems has to respect can be reduced.

- **order (order items)** The following parameters are encoded in an order. At first the order specify is a set of order items each declaring a number of specific packing-units which have to be packed on trollies. Additionally the order has to be packed at a certain time. This is a parameter important to compute the height of the different plants, since plants grow over time. And an order has to specify the maximal allowed height for cc-trollies, too.
- **hard-constraints** This list is imperfect so far.
  - stability of packed plants.
  - safety-aspects for stacking plants, ensuring no plant is damaged by stacking.
  - all plant have to be placed entirely on a layer
  - layers have to be correct mounted in the trolley
  - the overall trolley high is below the maximal allowed level
- **soft constraints**: The plants of an order item are placed contiguous on layers.
- **objective function** The objective function is to minimising the number of needed trollies for packing all potted plants of an order on trollies.

3 Proposed solution

In this section we present our concept of an algorithm capable of solving the described potted plant packing problem. As this paper addresses work in progress, this algorithm has not been implemented so far and can therefore not be presented with actual test results.
The original potted-plant packing problem, which is a 3-D bin packing problem is decomposed in a 2-D bin packing and a 1-D bin packing problem. We are able to do so, because on can first compute the layers, which is a 2-D bin packing problem with a modified objective function. After that the layers have to distribute among the trollies, this corresponds to a 1-D bin packing problem. In the following section the algorithm is described in more detail.

Since there is a huge number of different plant species, which differ only in aspects irrelevant for packing, the first step of our solution is the forming of plant groups. All plants classified in one group will have similar packing parameters. There could be a number of articles only differing in colour for instance, as colour is not relevant to packing it can be neglected and all articles only differing in colour can be classified into the same packing group. Classification will go beyond the mere exclusion of irrelevant attributes however, it will also be necessary to look for similarities between different plants. Thus our classification will be looking at these parameters: width, height, depth, weight, type of packaging-unit, flexibility of article (e.g. can it be bend) and stack ability. Furthermore it must be possible to classify one article as belonging to different groups, with the assignment to a group being limited to a certain time frame (e.g. June to September). For each order item the packing groups have to be computed from the existing data base, which consist of the

- length,
- depth,
- hight and
- weight

of the packing-units. The before mentioned further attributes will be filled in at request as the data does not allow for an automatic computation of these.

Because of the huge number of possible solutions and to integrate already existing knowledge on good solutions, the use of packing patterns seems profiting. A packing pattern encodes a valid packing of packing-units of certain categories on a single layer. A packing pattern can be declared for one or more categories of packing-units. Figure 2 can be seen as an example of an packing-pattern packing only one kind of category on a layer. In figure 4 another example with different kinds of categories is shown. A preference coefficient can be assigned to any particular

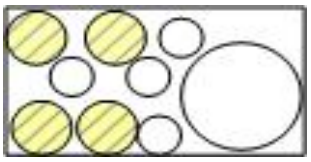

Figure 4: packing-pattern with different categories

packing pattern to distinguish favourable and less favourable patterns from one another. Such a preference coefficient can e.g. respect if categories with similar hight are used or how densely the objects are packed. A variety of packing-patterns is required for the proposed system to work. They can either be derived from expert knowledge or be computed offline, thus reducing online computing time. All packing patterns will be stored in a knowledge base. To compute such packing-patterns, a 2-D packing problem has to be solved. Therefore a heuristic like the G4 heuristic [4] can be used.
Alternate to the G4 heuristic in a fist approach a simpler strategy can be used, which is already used today by the human packers. The main idea of this strategy is to use the knowledge of how much plants of one category fit on a layer. If for instance 21 plants fit on one layer then one plant has the size of $\frac{1}{21}$ layer. So compute the load of one layer is simple fractional arithmetic. This heuristic has the advantage that it is fast and in fact can used for online-computation of packing instructions. But in fact it is not as optimal as other techniques since it results in less optimal plans when different categories are packed on one layer. For instance, one can think of a packing-pattern of three large plants, as shown in figure 5. Obvious there can only placed three plants of that size on one layer. Thus a plant would have the size of $\frac{1}{3}$ layer. But obvious one can pack smaller plants in between, like it is shown in figure 6.

The first step of the algorithm at runtime is to compute which order items are delivered in pre-packed trollies and thus how much trollies with possible free capacity exists. The pre-packed order items can be disregarded for the rest of the planing process.

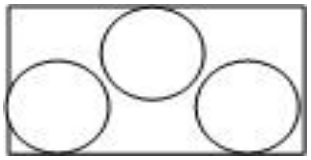

Figure 5: packing-pattern with three large plants)

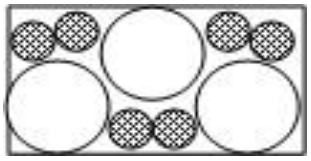

Figure 6: packing-pattern with three large and six small plants

The second step of the algorithm is to group all packing-units as per their classification. For each packing-group a queue is used, in which all corresponding packing-units are then stored. The order of packing-units in the queue is analogue to the order of items in the initial order.

To compute an initial packing plan a simple heuristic is used. The queues are sorted by their group height in descending order. Thus the algorithm starts with the queue containing the tallest plants. The algorithm then accesses the knowledge base and tries to find packing-patterns for the packing-units of the group stored in the active queue. If no matching packing-pattern can be found, or if there are not enough packing-units to fill an entire layer the packing-units of the next queue are added to the pool and the procedure is repeated. If there exists more then one packing-pattern, which can be applied, then the one with the pattern with the higher preference coefficient is chosen. The algorithm will try to assign all items to complete packing-patterns without a rest of unassigned items. It is however doubtful, that this will be possible, we anticipate to usually have a remainder of items that is unassigned after the pattern filling procedure. Said remainder will then be packed on layers using the same algorithm used for the offline computation of the packing-patterns or an appropriate heuristic. While this second step will ask for much higher computation times, we still expect to see a low overall runtime as we anticipate a pattern filling rate of 80% at the least.

The packing pattern knowledge-base basically improves the performance of the planning tool through the reuse of sub-solutions. Future steps will see the adoption of self-learning strategies such as the extraction of patterns from commonly used online computed layers. Such patterns would then be added to the knowledge base. Furthermore the preference coefficient could be automatically adjusted by the system for often used patterns, or those not used at all. The knowledge-base will start with patterns based on expert knowledge already existing and a set of computed patterns to broaden the base. It is expected to grow trough the aforementioned steps and the possible addition of further manually compiled patterns. The knowledge-base enables the planning system to use advantages learning systems offer, to improve computed packing instructions.

The algorithm will result in an initial plan, this initial plan than serves as the starting point for further improvement strategies. Modifications can be made for example by moving a group of packing-units on the layers. Another modification may the exchange of used packing-patterns. This may improve the density of packed layers and thus after some modification steps results in reducing the number of needed layers. The objective function for the improvement strategy is not the minimsation of the layers, like it would be in classic 2-D bin packing problems. Since a minimal number of layers does not guarantee a minimal number of cc-trollies. Assume the plants are placed on n layers. A solution leading to a minimal number of cc-trollies then can be computed using the following objective function for the improvement strategy $h(\mathcal{E})$:

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

$$h(E_i) = max_{1 \leq j \leq max\_count\_plantson\_layer} height(p_j)$$

This objective function leads to a minimal overall height of an order. The overall height of an order $h(\mathcal{E})$ influences the number of needed cc-trollies, since every trolley can be loaded only up to an upper bound.

The final step of the packing algorithm is the optimised distribution of the layers onto the trollies. This problem is equivalent to a 1-D bin packing problem. The objective is once again the minimisation of the number of cc-trollies needed to transport all layers. A point that differs from classic bin packing is the fact that there may exist different resources. There may be some trollies with a certain limited capacity due to the fact that they carry pre-packed layers. Although there may a number of trollies with extra capacity using add-on modules. And finally there is a virtual unlimited number of normal trollies. Other aspects like the maximal payload are constant for all trollies. The layers each have a known height, and so correspond to the weights that have to be distributed to the bins. We are convinced that an existing algorithm for the traditional bin packing problem can be adopted to solve this modified problem. One heuristic one can think of is the strategy to build groups of layers, with their common height is close to the maximal allowed height of an trolley.

After the layers are distributed onto the cc-trollies, they have to be sorted by their weight to compute all information necessary for a packing instruction.

4 Conclusion and Further work

In this article we present a real-world planning problem, the potted plant packing problem. This problem is a special instance of a 3-D bin packing problem. We described the problem and focus on some important aspects like stacking plants. After this we present a first formalisation of the problem. According to this formalisation a packing problem can be described by a 6-tuple consisting of resources, objects, order items, hard constraints, soft-constraints and an objective function

Finally we present our concepts solving the potted-plant packing problem. One of our main idea is the usage of packing-patterns and a knowledge base storing them. This allows the reuse of sub-solutions and can reduce online computation time, which is in fact a critical factor solving

a real-world problem. Because our costumer wants to integrate the packing planning tool in his software-architecture, if the quality of computed plans are satisfying.
Furthermore we think that using the knowledge base offers potentials developing the system in the direction of a self-learning system. Such self-adapting functionality would reduce the administration effort for maintaining the knowledge base, e.g. create new packing patterns if the typical category mix within an order change over time or so.
Due to the algorithm design we decompose the 3-D bin packing problem in a 2-D bin packing problem, packing the layers, and a 1-D bin packing problem, distribute the layers among the trollies. The so far challenging point we expect is realising the layer packing. Actually we discuss the promising data representation and solving strategies appropriate for the here described problem.

So far our project is still in a conceptual phase. The presented solution has to be implemented and has to be proven with real world data and in day to day business. We hope to have a first set of test results by August, which we then could present at the PUK-Workshop and in a final version of this paper.

References

[1] Foko Lübsen und Sohn Internationale Spedition: Hompage Focko Lüpsen & Sohn GmbH, http://www.luepsen.de/seite01e.htm. (2005) accessable on 02.05.05.

[2] Keng, N.P., Yun, D., Rossi, M.: Interaction sensitive planning system for jop-shop scheduling. In Oliff, M.D., ed.: Expert Systems and Intelligent Manufacturing. Elsevier (1988)

[3] Sauer, J.: Knowledge-based systems in scheduling. In Leondes, T.L., ed.: Knowledge-Based Systems Techniques and Applications. Academic Press, San Diego (2000) 1293–1325

[4] Scheithauer, G., Terno, J.: The G4-Heuristic for the Pallet Loading Problem. Journal of the Operational Reseach Society **47** (1997) 511–522