Tomi Peltola*, Tommi Pesu, Janne Lehtinen, and Kalle Raita

Data-driven pallet packing made easy

Abstract: Optimizing pallet packing decreases the resource use and cost of transporting goods. A classical optimization approach is bin packing. It assumes knowledge of item dimensions, is computationally demanding, and has many practical complications. We introduce an alternative approach that relies only on historical data of packed pallets and item identities. For a new order to be packed, the method first produces pallet templates from the data. Given the templates, a discrete optimization problem is solved to find the optimal item-to-pallet assignments. In addition, we designed a web-based user interface for logistics experts to examine and fine-tune the result. The user interaction is captured in the background to enrich the pallet template data, teaching the method better solutions. The application is deployed at a furniture maker, integrated into their enterprise resource planning software. Its maintenance is easy.

Keywords: logistics, pallet packing, bin packing

***Corresponding Author: Tomi Peltola:** Digisalix, E-mail: tomi.peltola@digisalix.fi
Tommi Pesu: Digisalix, E-mail: tommi.pesu@digisalix.fi
Janne Lehtinen: Digisalix, E-mail: janne.lehtinen@digisalix.fi
Kalle Raita: Digisalix, E-mail: kalle.raita@digisalix.fi

1 Background

Transportation of large shipments of goods is resource-intensive and costly. Optimizing packing of the items can help reduce the impact on the environment and the cost for the customer. Our motivating case is packing large orders of furniture on pallets at the factory for shipment to new or newly-furnished buildings. Manually planning the assignment of ordered items onto pallets requires slow-to-learn expertise and is burdensome. In addition to guiding packing, the automation also helps customer service to estimate shipment costs faster.

A classical mathematical optimization approach is *bin packing*. It assumes knowledge of the item packing dimensions and is computationally demanding. Further complications arise, for example, from irregularly shaped items, item stacking restrictions, or items requiring scaffolding for safe transportation.

We introduce a data-driven approach that only needs historical data on packed pallets and item identities (e.g., product codes) and does not rely on item dimensions. Such data is often readily available from enterprise resource planning software or similar source and is automatically kept up-to-date as part of packing new shipments. The advantages of the approach are:

- Interpretable and intuitive for logistics experts: The source data is in the form of historical packed pallets, familiar to the expert. The optimization result can be easily traced back to the source data.
- Instantly adaptable: The expert can modify the dataset, making the method produce better solutions. The improvement workflow can be automated in the background of natural user operation.
- Easy deployment and maintenance: Simple, readily-available source dataset. No costly and brittle retraining like in most machine learning methods[1]. The dataset improves naturally as part of operating the software and with more shipped deliveries.
- Validity of solutions: the solutions are always packable as long as the pallets in the dataset are.
- No need for special hardware such as graphical processing units.

2 Data-driven pallet packing

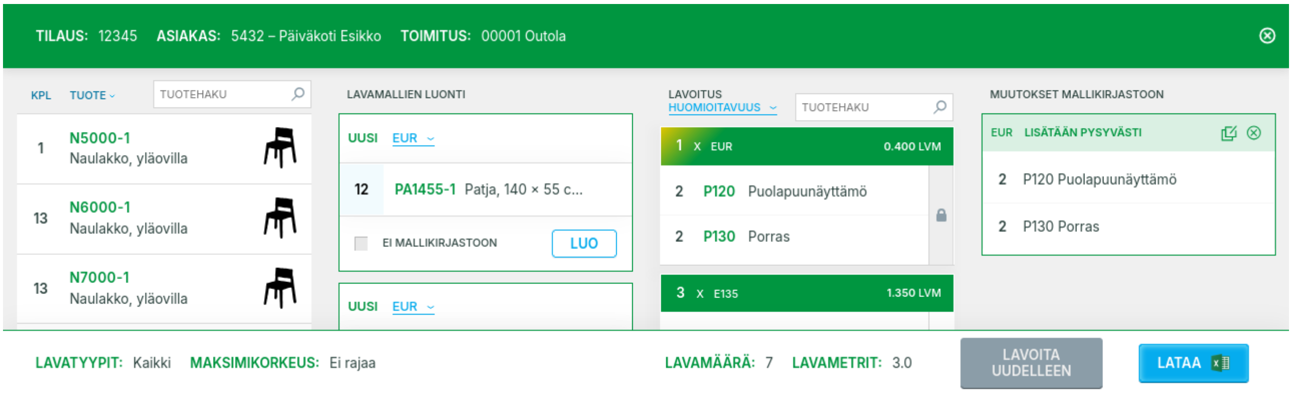
Our aim is to minimize the resource use or cost of assigning a given list of items (*current order*) to pallets (or other packages; for concreteness, we speak of pallets). The method is based on historical data of packed pallets: for each packed pallet in the data, we only require the identities and counts of the items.

The method then works in two steps:

1. Extract *maximal pallet templates* from the data containing items in the current order.
2. Using the templates, formulate and solve an *integer-programming problem* to minimize the cost of assigning the items to pallets.

In step 1, a *pallet template* is a set of items and their counts that can be packed to a single pallet. A *maximal pallet template* is a template that is a strict superset of any other pallet in the data with respect to the items in the order. As an example, consider items

[1] The approach can be seen as a non-parametric machine learning method in the vein of k-nearest neighbor classifier.



TILAUS: 12345 ASIAKAS: 5432 – Päiväkoti Esikko TOIMITUS: 00001 Outola

KPL TUOTE TUOTEHAKU

1 N5000-1 Naulakko, yläovilla

13 N6000-1 Naulakko, yläovilla

13 N7000-1 Naulakko, yläovilla

LAVAMALLIEN LUONTI

UUSI EUR

12 PA1455-1 Patja, 140 × 55 c...

□ EI MALLIKIRJASTOON LUO

UUSI EUR

LAVOITUS HUOMIOITAVUUS TUOTEHAKU

1 X EUR 0.400 LVM

2 P120 Puolapuunäyttämö

2 P130 Porras

3 X E135 1.350 LVM

MUUTOKSET MALLIKIRJASTOON

EUR LISÄTÄÄN PYSYVÄSTI

2 P120 Puolapuunäyttämö

2 P130 Porras

LAVATYYPIT: Kaikki MAKSIMIKORKEUS: Ei rajaa

LAVAMÄÄRÄ: 7 LAVAMETRIT: 3.0

Fig. 1. Screenshot of the web-based user interface (texts in Finnish). The header shows order meta data. The footer shows pallet constraints and output information. The main window has four columns: (1) listing of ordered items, (2) pallet template editor allowing creating new pallet suggestions, (3) current pallet packing solution, and (4) changes to the pallet templates.

A, B, and C, A and B of which are in the current order. Then, the template $(A, 2), (B, 3)$ is better than $(A, 1), (B, 3), (C, 4)$ since it can pack two items A while the latter only one (C is irrelevant because it's not in the order).

In step 2, an *integer-programming problem* is formulated to assign all of the order items to pallets. The use counts of the extracted set of templates form the space of candidate solutions, with constraints requiring all items to be assigned to a pallet. In the simplest case, the cost of a solution is the number of used pallets. Off-the-shelf discrete optimization software packages can be used to solve the problem[2].

The method can accommodate multiple extensions:

- Different types of pallets: If the history data includes packed pallets with different sizes, the optimization in step 2 can minimize the total transport size (loading meters). Restrictions for pallet types can be enforced by filtering the data in step 1. Similar filtering can be used for shipment height if the data contains heights of packed pallets.
- Item categories: Product codes might identify items too strictly. For example, products differing only in color can be packed exchangeably. Product categories can be then used as item identities.

A strong advantage of the method is that the source data, the packed pallets, is very intuitive for a logistics expert. As such, if the method suggests an overfilled pallet, the source template can be identified and removed from the data. If the expert recognizes more efficient packing solutions, new templates can be added to the dataset, making the algorithm learn better solutions. The method adapts instantly without re-training.

A disadvantage is that the method doesn't extrapolate outside of the template data. This makes it conservative in the sense that the solutions are valid (can be packed as long as all pallets in the data are valid). This, however, can give suboptimal solutions when not-yet-seen combinations of items could be used to pack more efficiently. Moreover, new products cannot be efficiently packed (unless they can be categorized together with an existing product). Machine learning based prediction could be used to alleviate the issues, for example, by predicting new templates or item size correspondences.

The optimization algorithm also doesn't inherently distinguish between solutions with the same cost. Logistics experts, however, might prefer certain solutions to others, for example, because of ease of packing or unpacking. Such preferences don't often explicitly exist in the recorded data, but might be reflected in the distribution of packed pallets. If available, the preferences could be directly used in the optimization cost. Otherwise, statistics or machine learning could estimate them from data and, for example, re-rank equivalent solutions. A good user interface can also alleviate the problem.

3 User inferface

Figure 1 shows the web-based user interface. The changes of the user to the optimization result are recorded in the back-end to improve the template data. The application is deployed at a furniture maker, integrated into their enterprise resource planning software.

[2] CP-SAT solver in Google OR-Tools, Python package version 9.7.2996, https://developers.google.com/optimization.