Optimized Packaging – INFORM

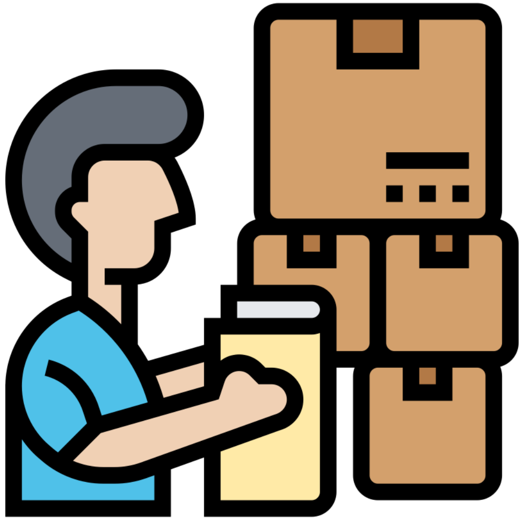
Final Presentation | 16th of July 2020

Team

Clinton Charles	M.Sc. DDS
Moritz Dederichs	M.Sc. Computer Science
Akshay Ganesh	M.Sc. DDS
Kalyan Keesara	M.Sc. DDS
Luca Koczula	M.Sc. Industrial Engineering
Yordan Manolov	M.Sc. Computer Science
Janos Piddubnij	M.Sc. Data Science
Tobias Wagner	M.Sc. Industrial Engineering

Agenda

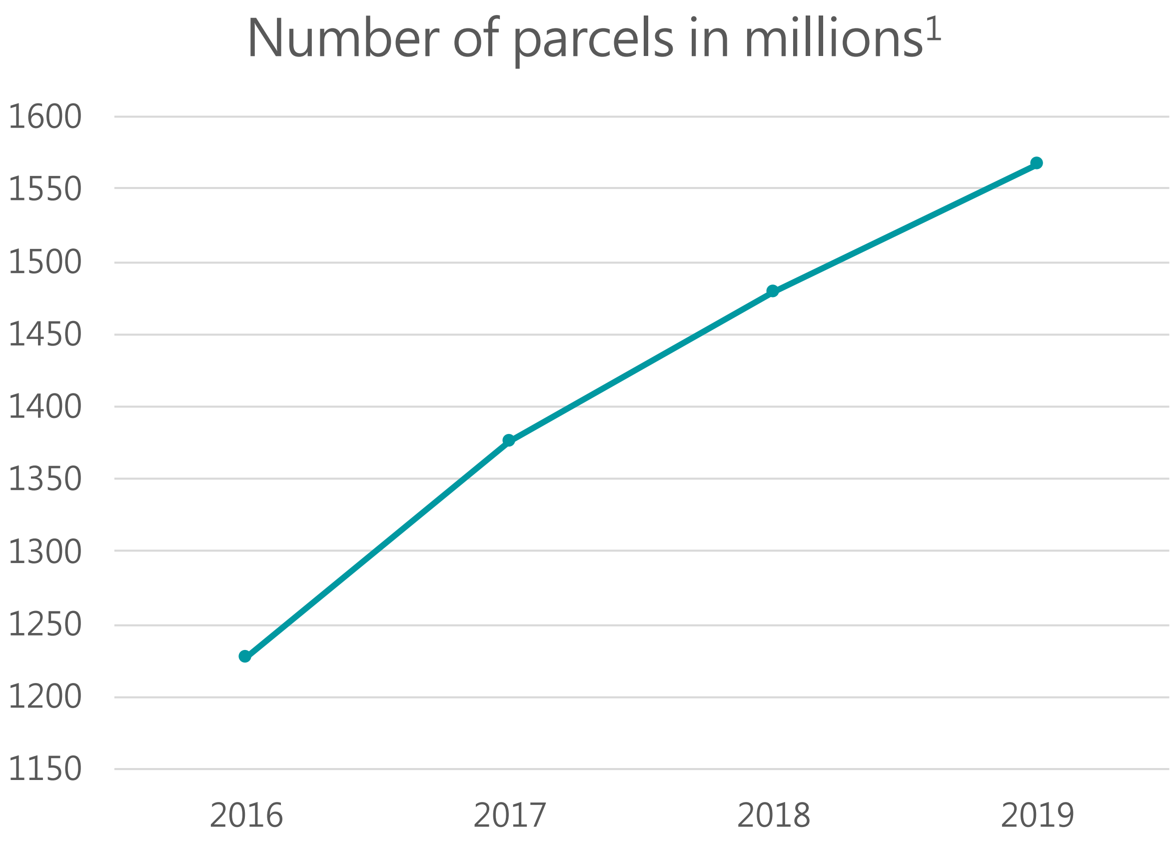
1. Introducing the problem
2. Tackling the problem: Assign Items to Boxes
   1. Exact Approach: Split Model
   2. Heuristic Approach: Genetic Algorithm & Best Match Heuristic
   3. Heuristic Approach: Extreme Points
3. Tackling the problem: Assign Boxes to Pallets
   1. Heuristic Approach: Peak Filling Slice Push
4. Comparing the Results
5. Working with our Python Package
6. Live Demo



https://www.flaticon.com/de/autoren/eucalyp

RWTH AACHEN UNIVERSITY INFORM

Introducing the Problem: Motivation



- Number of transported parcels by Deutsche Post has risen steadily over the last years
- The global delivery market had an estimated value of 430 bn USD in 2019[2]
- The global pallets market size was estimated to be 59,91 bn USD in 2018 with wood being the most used material[3]
- Improved Efficieny = Better Margins + Reduced Ecologocical Footprint

[1]Deutsche Post. (2020). Anzahl der beförderten Pakete durch die Deutsche Post in Deutschland von 2016 bis 2019 (in Millionen Stück). Statista. Statista GmbH. Zugriff: 13. Juli 2020. https://de.statista.com/statistik/daten/studie/476935/umfrage/anzahl-der-befoerderten-pakete-durch-die-deutsche-post/ ,
[2]https://apex-insight.com/product/global-parcel-delivery-market/ ,
[3]https://www.fortunebusinessinsights.com/industry-reports/pallets-market-100674

Introducing the problem: Problem statement

Formulate a model that optimizes 3D packaging, addressing the following challenges	
Packaging efficiency	Complexity of execution
Computational effort	Customization to the end user

Introducing the problem: Industry benchmarks

Highlights on Amazon's solution to this problem:

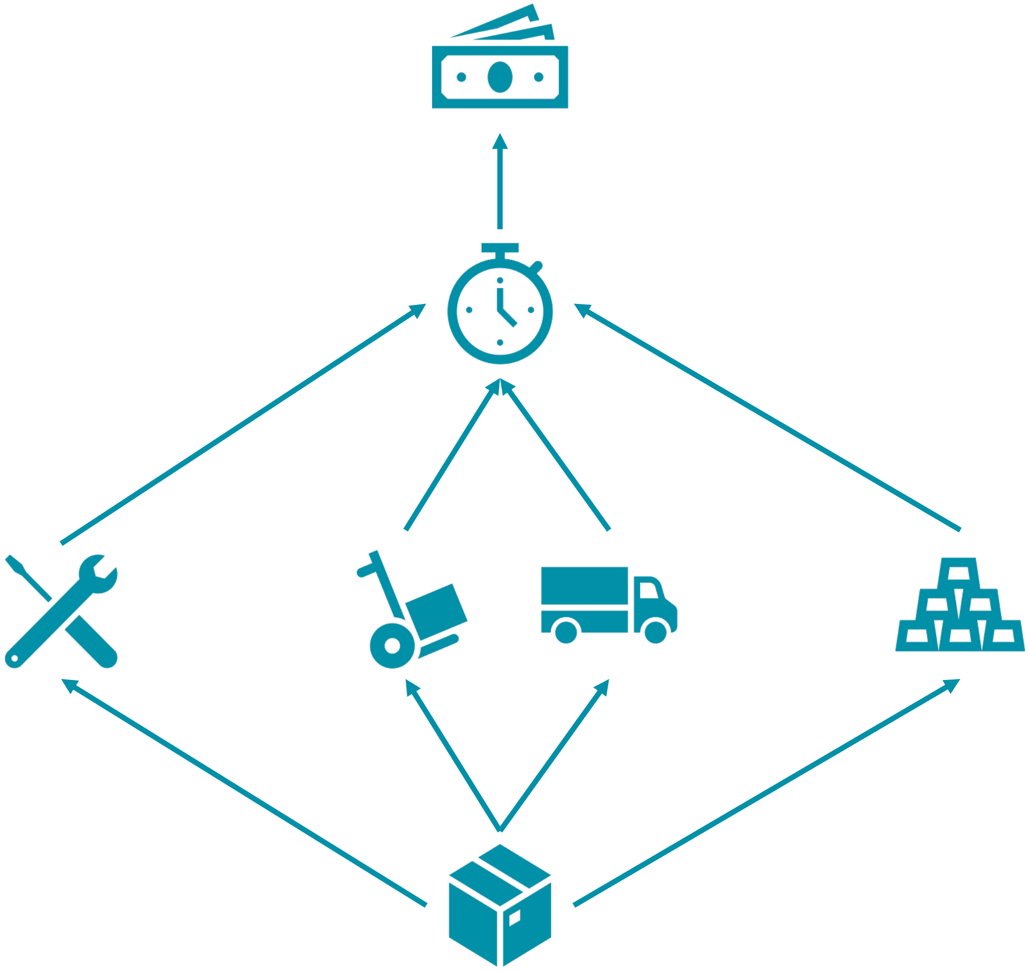
- Software system displays the suitable box sizes to pick from
- Launches a pilot project on testing robots that create custom sized carton wrapping[4]

- Limitations:
  - Practical feasibility
  - Not possible to fully replace human workers in the near future

- Currently, only a handful of small players are offering solutions in this area

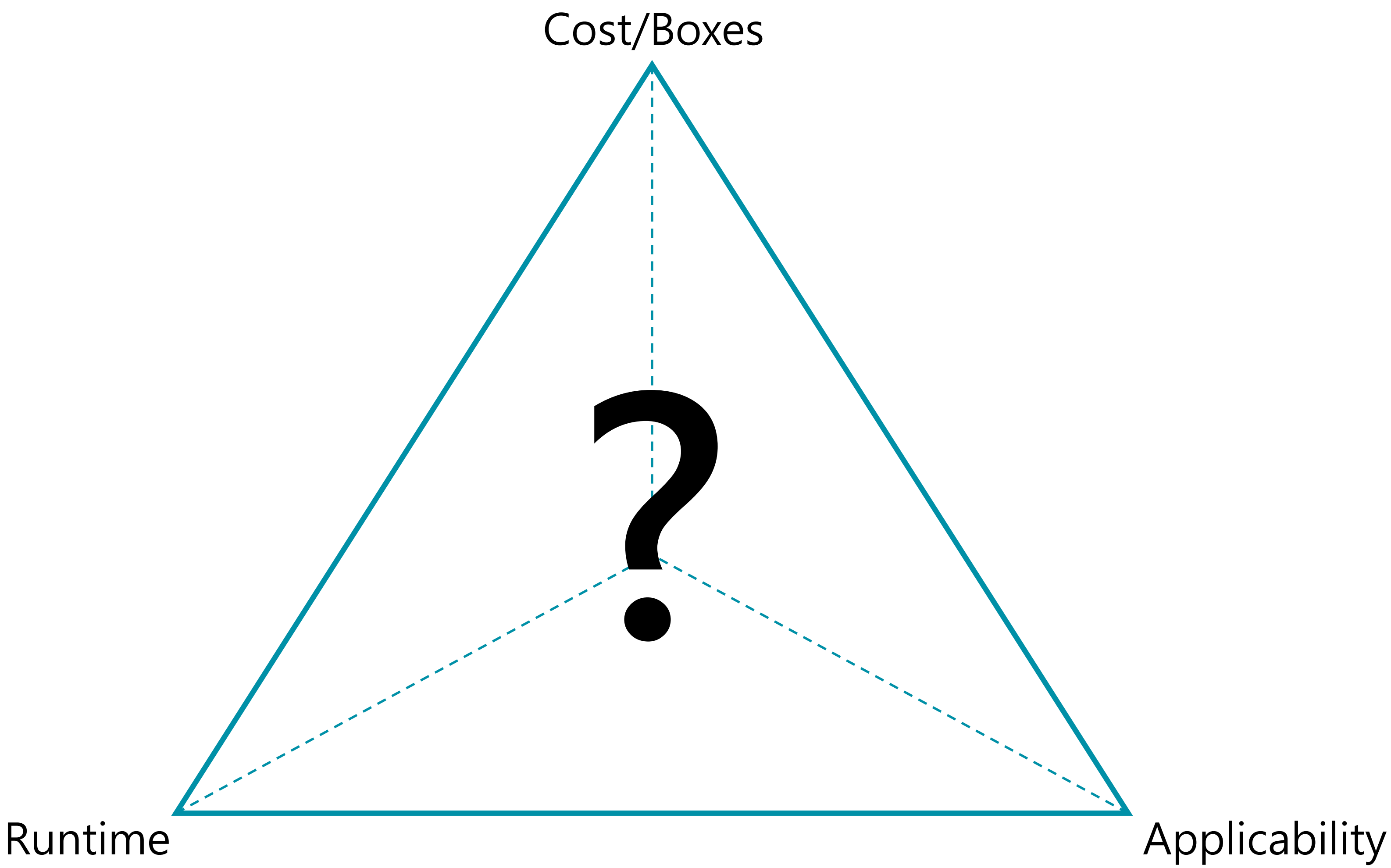
[4] https://www.reuters.com/article/us-amazon-com-automation-exclusive/exclusive-amazon-rolls-out-machines-that-pack-orders-and-replace-jobs-idUSKCN1SJ0X1

Introducing the problem: Business Use Case @INFORM



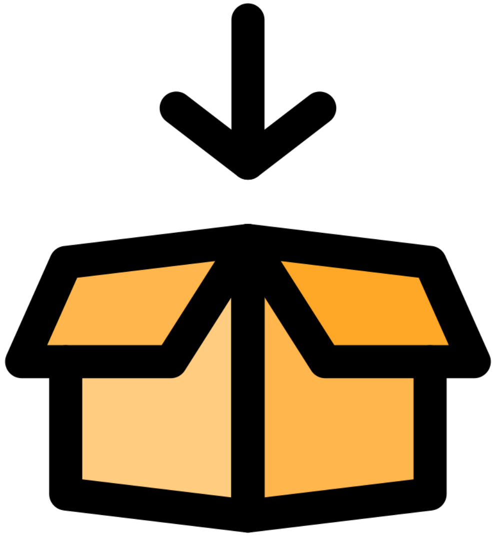
- Great complement to SyncroTess
- First mover advantage as a big player
- Reap the benefits of trickle-up
- In-line with INFORM's characteristic on being environment-friendly

Introducing the problem: Conflict of objectives



Agenda

1. Introducing the problem
2. Tackling the problem: Assign Items to Boxes
   i. Exact Approach: Split Model
   ii. Heuristic Approach: Genetic Algorithm & Best Match Heuristic
   iii. Heuristic Approach: Extreme Points
3. Tackling the problem: Assign Boxes to Pallets
   i. Heuristic Approach: Peak Filling Slice Push
4. Comparing the Results
5. Working with our Python Package
6. Live Demo



https://www.flaticon.com/authors/pixel-perfect

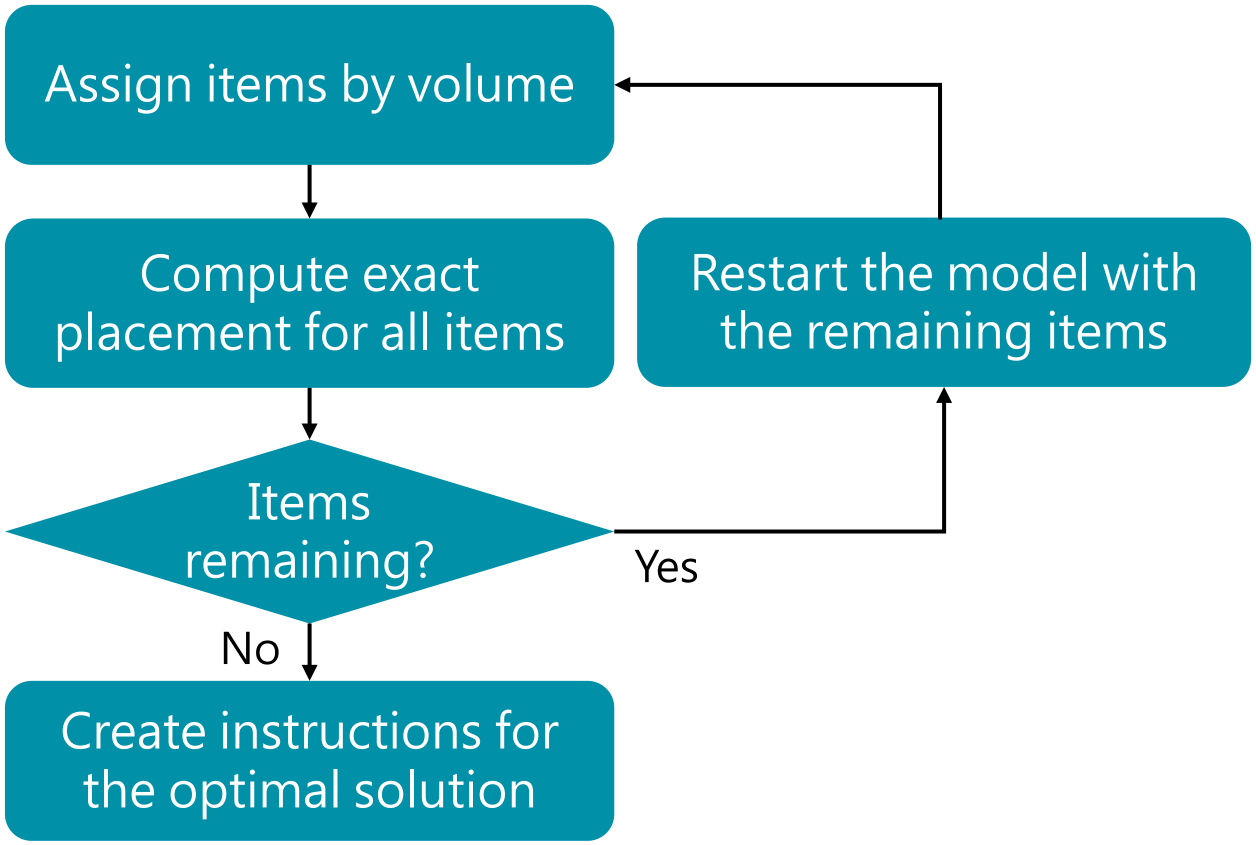
Split Model: Why?

- 3D bin packing is a well-investigated problem in operations research
- The optimization problem often can't be solved in reasonable time
- Improve run time by splitting the problem into two sub-problems

Split Model: How it works

- Objective: Minimize the number of boxes needed to fit all items in an order

1. Assign items to a minimal number of boxes based on volume
2. Compute the exact placement for all items inside a box
3. If items could not be placed restart model for those items



Split Model: Improvements

- Problem: There exists a vast number of ways to fit the items into a box
  - Terminate optimization program once a feasible packing is found

- Allow processing of multiple orders simultaneously

Split Model: Highlights

- ✓ **Objective:** Minimize number of boxes

- ✓ Implemented in Python 3
- ✓ Gurobi backend
- ✓ Multiprocessing
- ✓ Support for rotation of items
- ✓ Support for pallet packing with PFSP

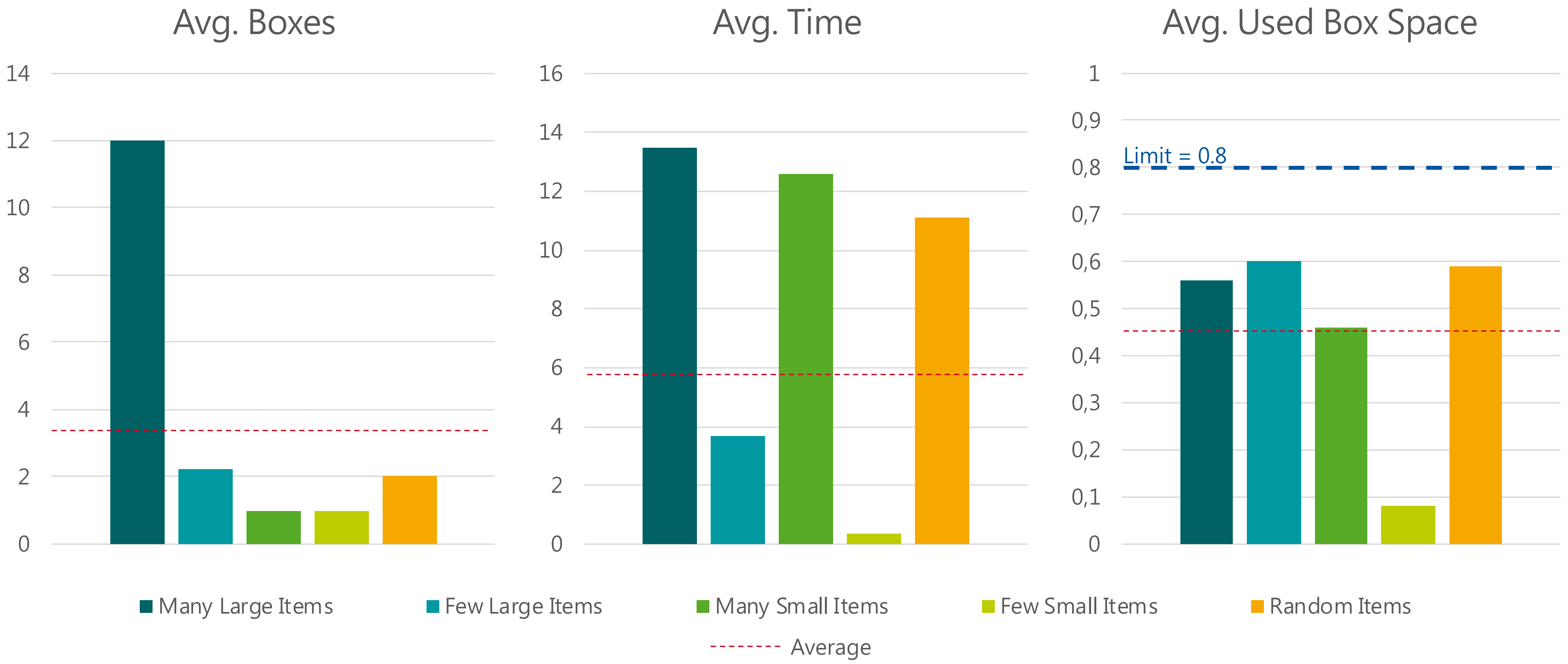
Benchmarking: Datasets and Metrics

Set 1	Set 2	Set 3	Set 4	Set 5
Many Large Items	Few Large Items	Many Small Items	Few Small Items	Random Items

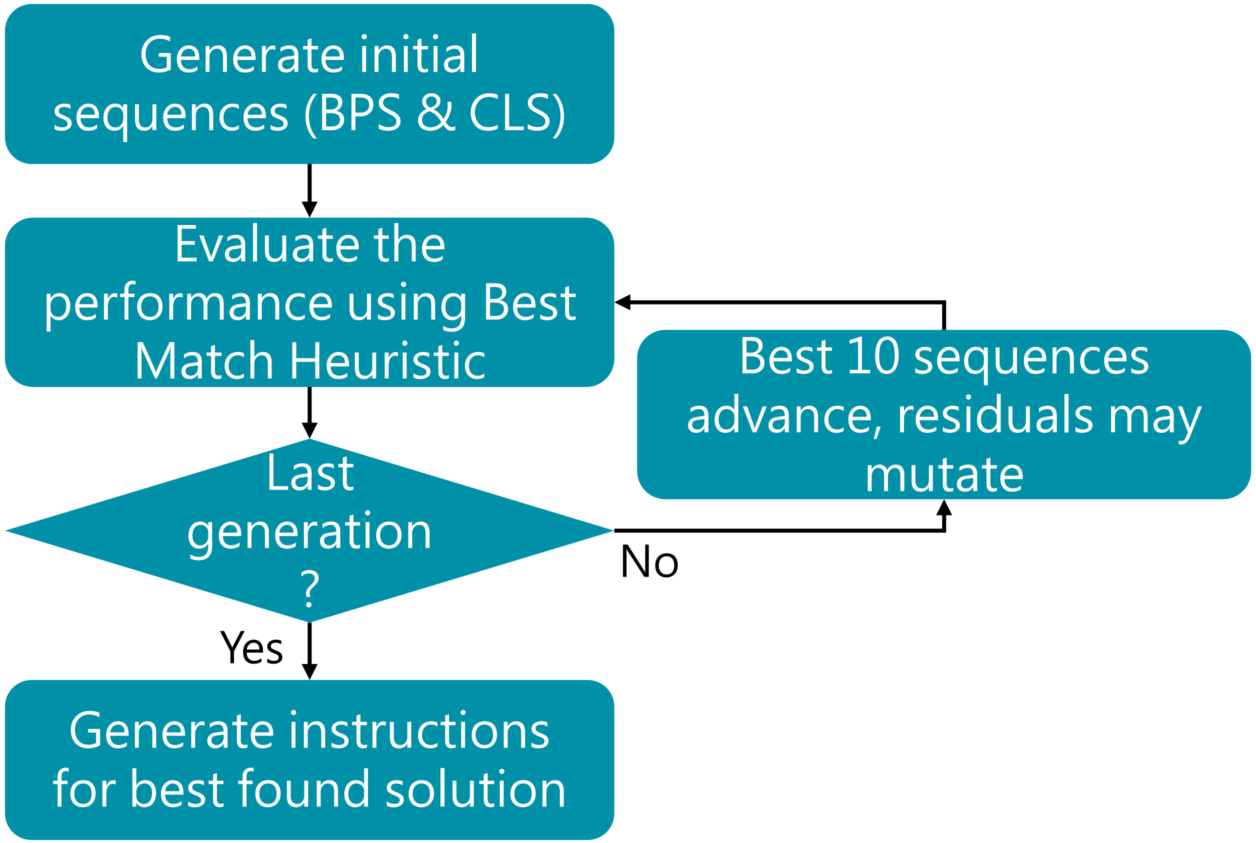
Avg. Boxes | Avg. Time | Avg. Used Box Space

https://www.flaticon.com/de/autoren/pixel-perfect

Split Model: Benchmark Results



Genetic Algorithm: How it works

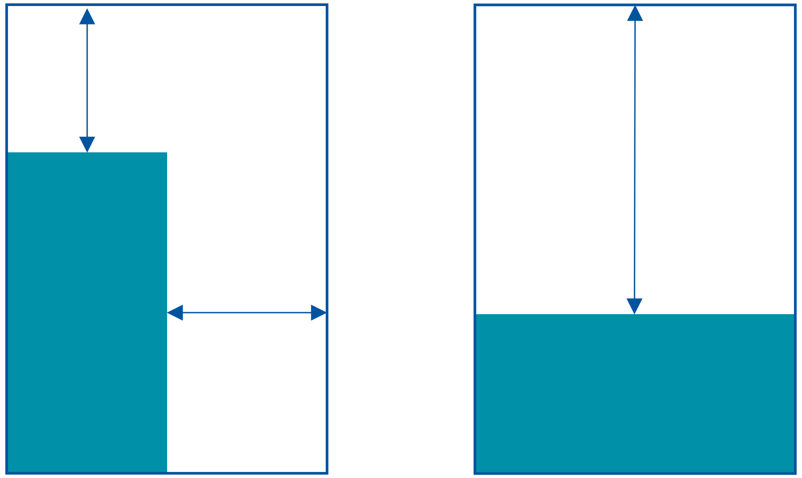


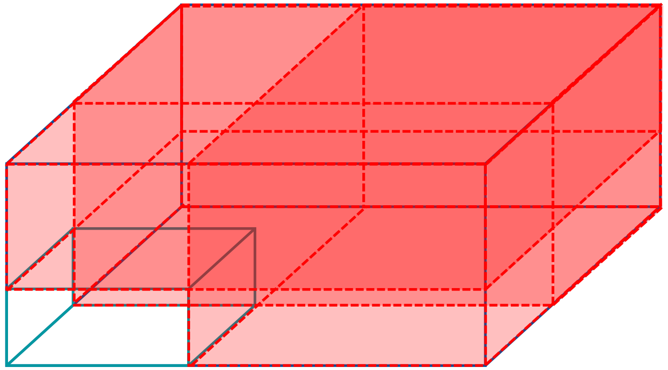
- Evolutionary approach:
  Generations and chromosomes
- Based on multiple fixed sequences of items and boxes
- Initialization with ordered and random sequences
- Evaluation of sequences via 'fitness' calculated by a heuristic
- Chromosomes might advance directly or mutate in each generation
- Based on a publication by Li et al. (2014)[5]

[5] „A genetic algorithm for the three-dimensional bin packing problem with heterogeneous bins“. Li et al. 2014. Proceedings of the 2014 Industrial and Systems Engineering Research Conference.

Best Match Heuristic: How it works

1. Open a box of the CLS and initialize its Empty Maximal Space (EMS) as the size of the selected box
2. Select an item from BPS while maximizing the used space of the considered EMS and item
3. Determine the item orientation by the minimal margin method
4. Place the item into the box
5. Update Empty Maximal Spaces (EMSs)
6. If the next item fits into an EMS, go to Step 2
7. Else open the next box, go to Step 1
8. Repeat until all items are packed





Images inspired by: „A genetic algorithm for the three-dimensional bin packing problem with heterogeneous bins ". Li et al. 2014. Proceedings of the 2014 Industrial and Systems Engineering Research Conference.

Genetic Approach: Objectives

Costs

Business pricing models
- only available on request
- highly individual

Environmental Impact

- Difficult to measure/quantify

Used Boxes

- Used space can get very low
- Already considered by the split model

Used Box Space

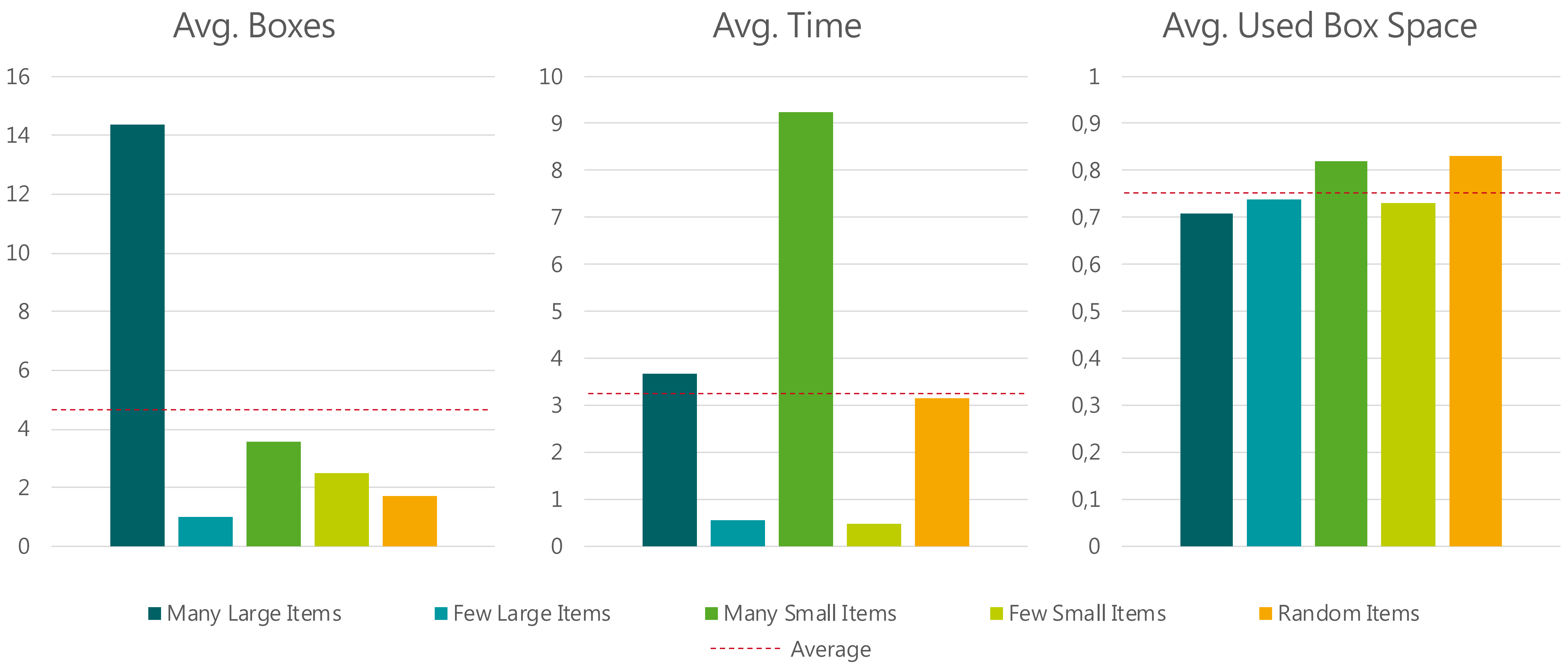
1. Less wasted space
2. Less cushioning and packing material
3. Comparably low cost
4. Higher vehicle capacity utilization

Genetic Approach: Highlights

✓ **Objective:** Maximize Used Box Space

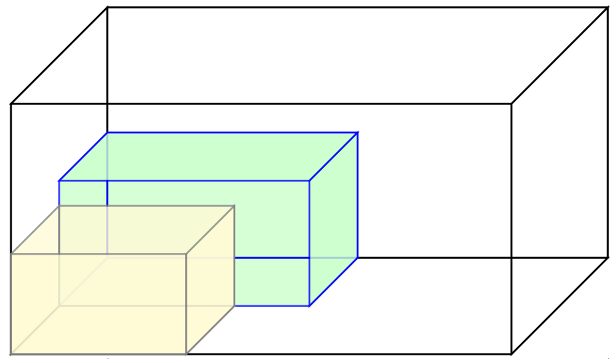
✓ Implemented in Python 3
✓ Standard (Open Source) Libraries
✓ Highly customizable by the end user without touching a line of code
✓ No expensive solver license needed (like Gurobi)
✓ Calculation of total and individual weight of boxes
✓ Support for rotation of items
✓ Support for pallet packing by using PFSP heuristic

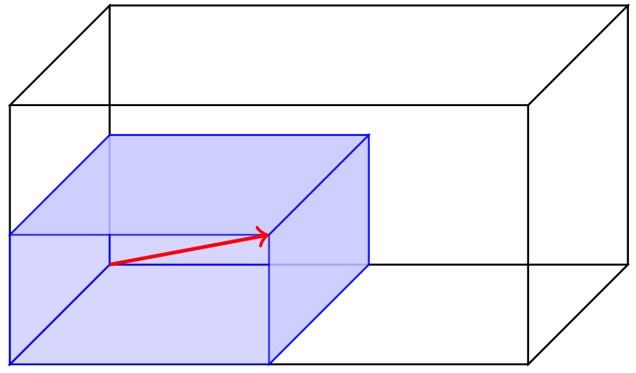
Genetic Approach: Benchmark Results



Extreme Points: How it works

- Based on a publication by Crainic et al. (2008) [6]
- Whenever a new item is to be placed into a box,
  - Put the item next to the 'walls' of the current EPs
  - If any fits, choose the smallest increase
  - New EPs are the maximum corner points of all items

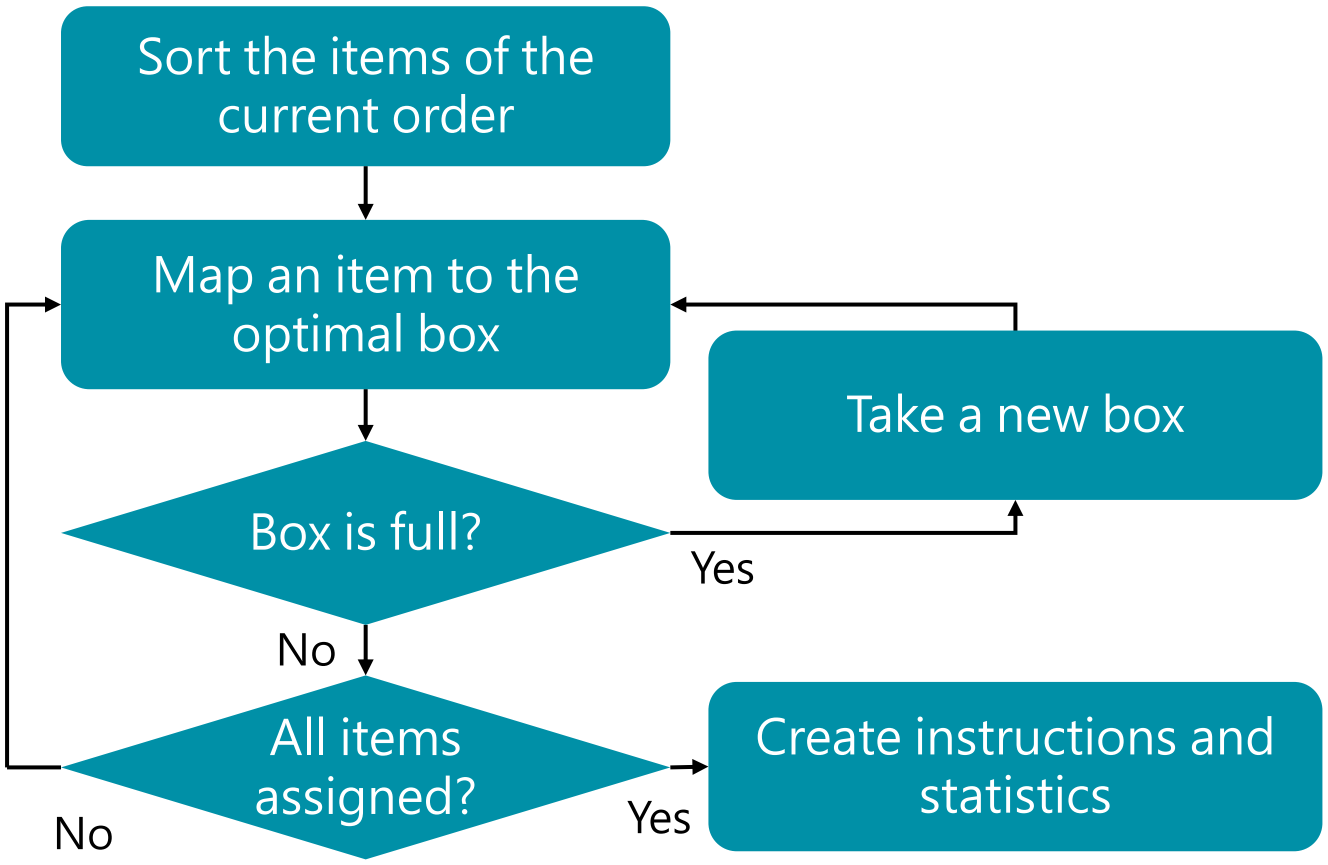




[6] „Extreme Point-Based Heuristics for Three-Dimensional Bin Packing". Carinic et al. 2008. INFORMS Journal on Computing Vol. 20 p. 368-384

Extreme Points: How it works

- 'Optimal' depends on the options specified to the heuristic



Extreme Points: How it works

- Input: order containing **N** items
- Output: list of **M** boxes packed with the input items

- All items are sorted in decreasing order
- Restrictions/Properties
  - Successor items are of sizes less or equal to predecessor ones
  - All items keep their orientation after insertion into a box
- Criteria for sorting items
  - Area
  - Height
  - First height, then volume
  - First volume, then height
  - First area, then height
  - First height, then area

Extreme Points: Variants

EP-FFD: first fit

- Put an item into a box
- If the current box is too full for the item, pick a new box
- Time complexity:
  $\Omega(3 \cdot M \cdot N) \subseteq O(N^2)$

EP-BFD: best fit

Before inserting the current item:

- If no previously packed box has enough room, place the item in a new box
- Otherwise pick one which maximizes f_m
- Time complexity: depends on f_m

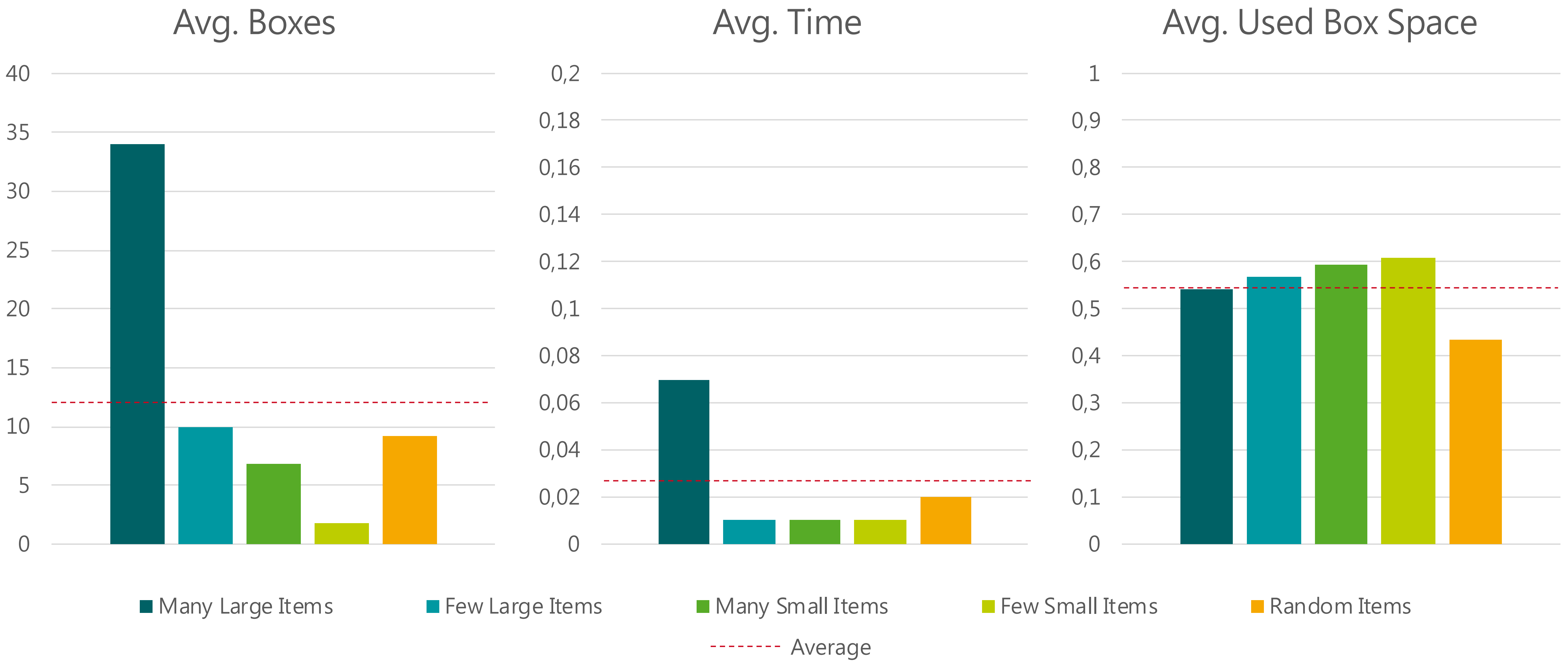
The merit function f_m is one of: maximize free volume after item insertion, minimize packing size, minimize packing size leveled, maximize residual spaces

Extreme Points: Highlights

- ✓ **Objective:** Minimize Processing Time

- ✓ Implemented in Python 3
- ✓ Standard Libraries for (algebraic) computation, data structures, sorting
- ✓ Very fast
- ✓ 100 % FOSS:
  - ✓ No 3rd party vendor lock-in
  - ✓ Easy set-up
  - ✓ Highly customizable (transpiling)

Extreme Points: Benchmark Results



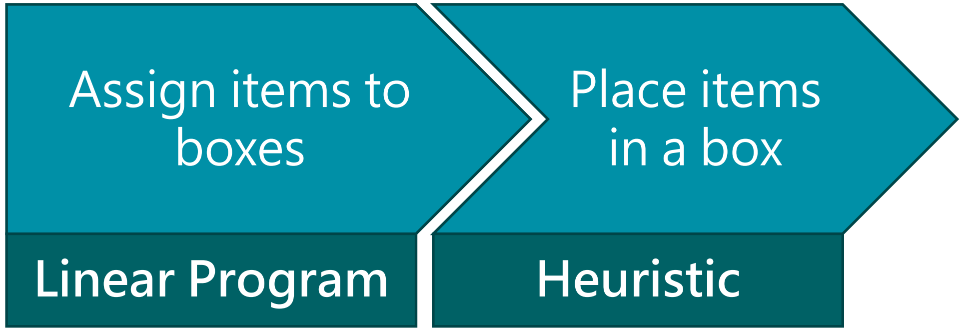
Combine Split Model and Heuristics: Why?

- Combine strengths of both model approaches
  - Minimal number of boxes by mathematical formulation
  - Speed from a heuristic

- Common approach in practice

- Easy to implement: both models are already implemented
  - Only thing to do: connect both approaches

Combine Split Model and Heuristics: How it works



- Objective: Minimize the number of boxes needed to fit all items of an order
  1. Use the linear program to assign items to boxes
  2. Use a heuristic to place items into box
- Try to achieve a considerable run time improvement while maintaining a comparably high solution quality

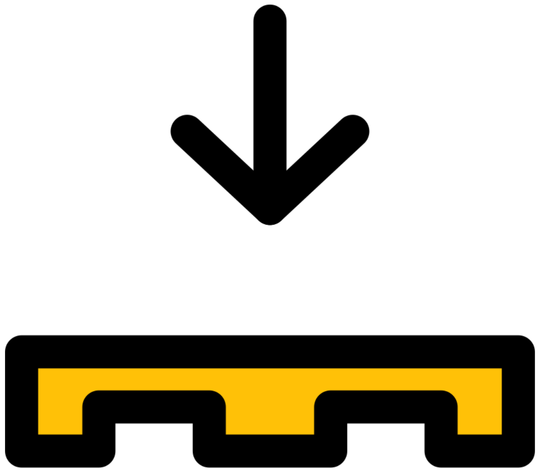
Combine Split Model and Heuristics: Results

- Problem: Only combined the worst of both approaches
- Combination is only slightly faster than the linear program
- Needs many more boxes than the linear program
  - Often even more boxes than the heuristic on its own

→ **Therefore, not further pursued**

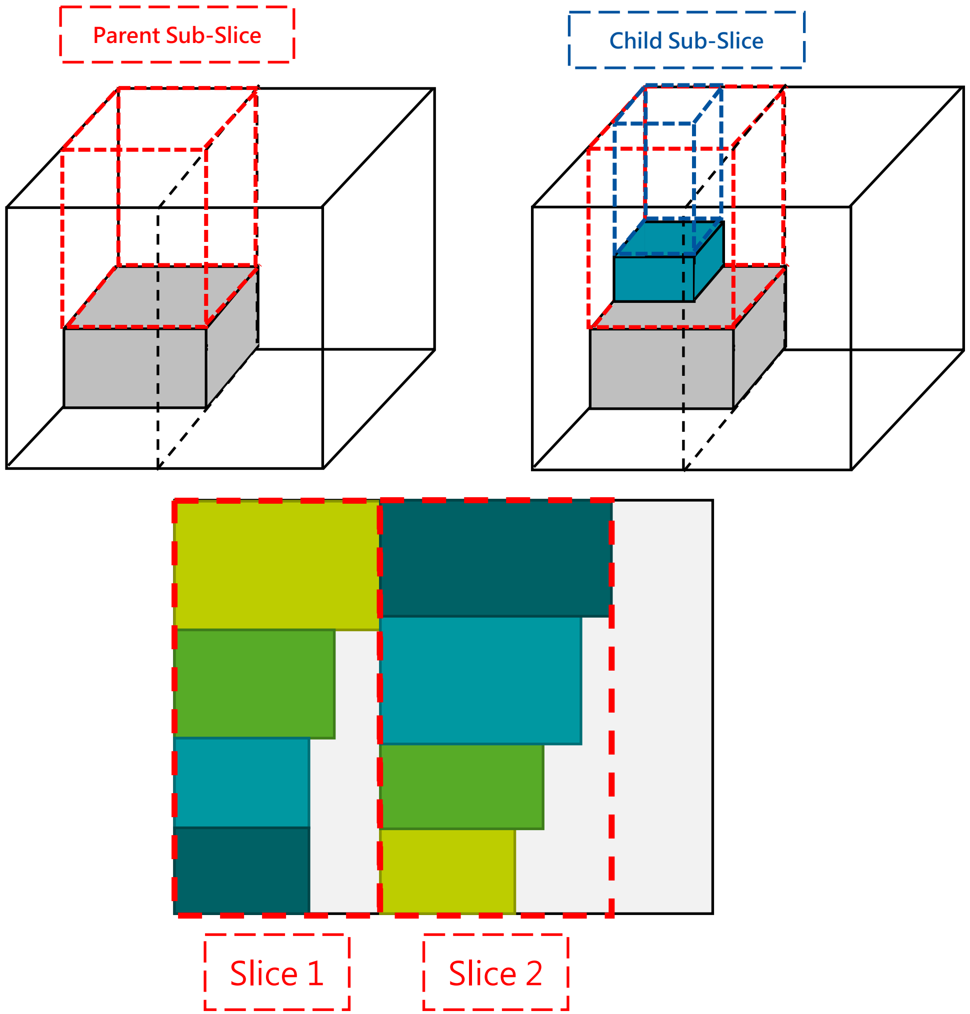
Agenda

1. Introducing the problem
2. Tackling the problem: Assign Items to Boxes
   i. Exact Approach: Split Model
   ii. Heuristic Approach: Genetic Algorithm & Best Match Heuristic
   iii. Heuristic Approach: Extreme Points
3. Tackling the problem: Assign Boxes to Pallets
   i. Heuristic Approach: Peak Filling Slice Push
4. Comparing the Results
5. Working with our Python Package
6. Live Demo



https://www.flaticon.com/authors/pixel-perfect

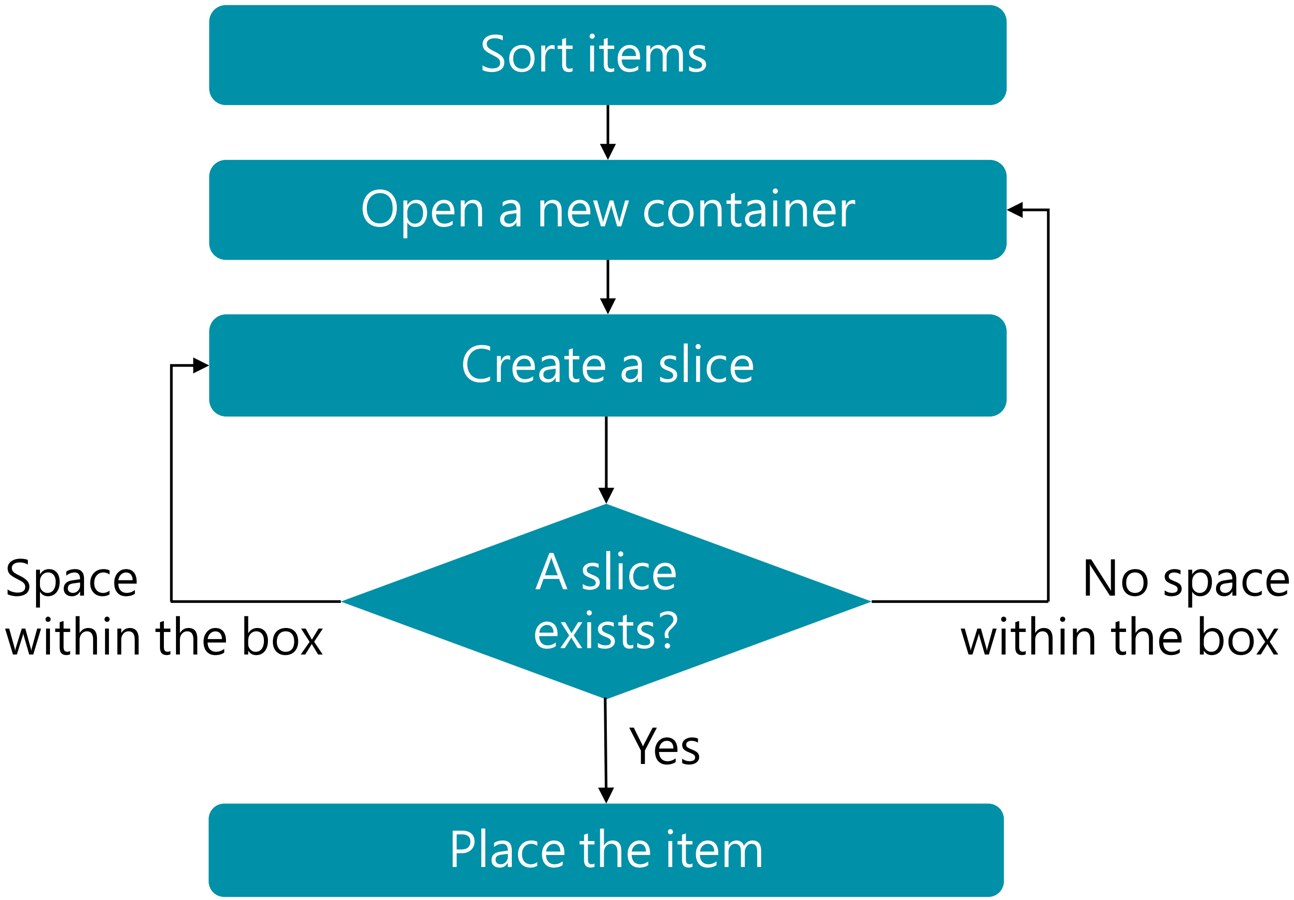
Peak Filling Slice Push: Why?



- A recursive divide and conquer algorithm
- A combination of first fit and sweep heuristic
- The box is divided into slices and sub-slices based on the dimension of items
- The largest box always goes to the bottom
- Forms a pyramid arrangement, making it easier to pack
- Based on a publication by Maarouf et al. (2008)[7]

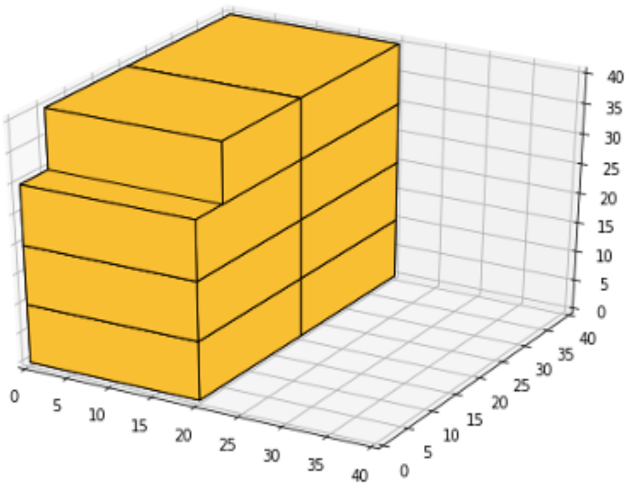
[7] and Images inspired by: "A New Heuristic Algorithm for the 3D Bin Packing Problem". Maarouf et al. 2008. Elleithy K. (eds). Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering. Springer, Dordrecht

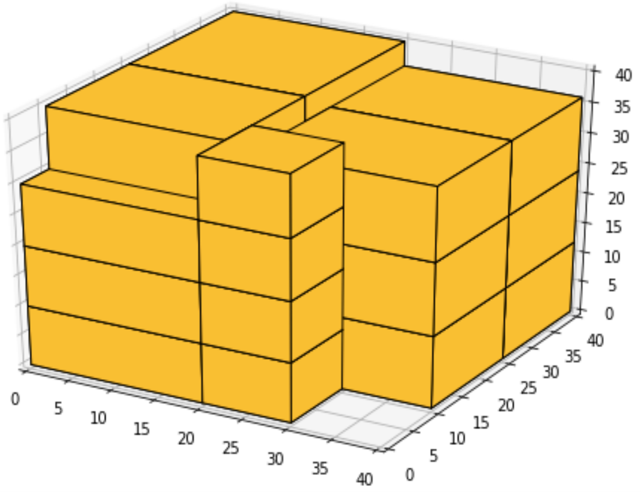
Peak Filling Slice Push: How it works

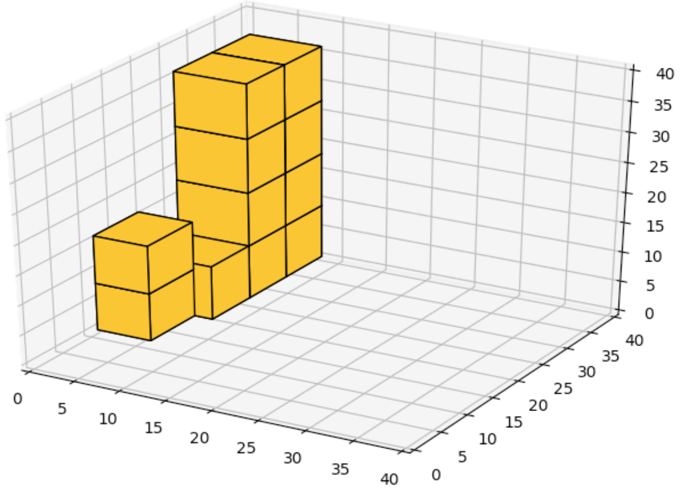


- Items are sorted in decreasing order of dimensions
- Peak filling packs the items one on top of another until the top of the container is reached
- Proceeds to the next slice when there is no more room for any of the items within the current slice
- Proceeds to the next box when there is no room for any more slices

Peak Filling Slice Push: Graphical representation







Slice 1 → Slice 2 → Next Pallet

Peak Filling Slice Push for Pallet Packing: Why?

Bin packing

- ✓ Split Model
- ✓ Genetic Approach
- ✓ Extreme Points Heuristic
- ✓ Combination of approaches

Pallet packing

- ✓ Peak Filling Slice Push Heuristic

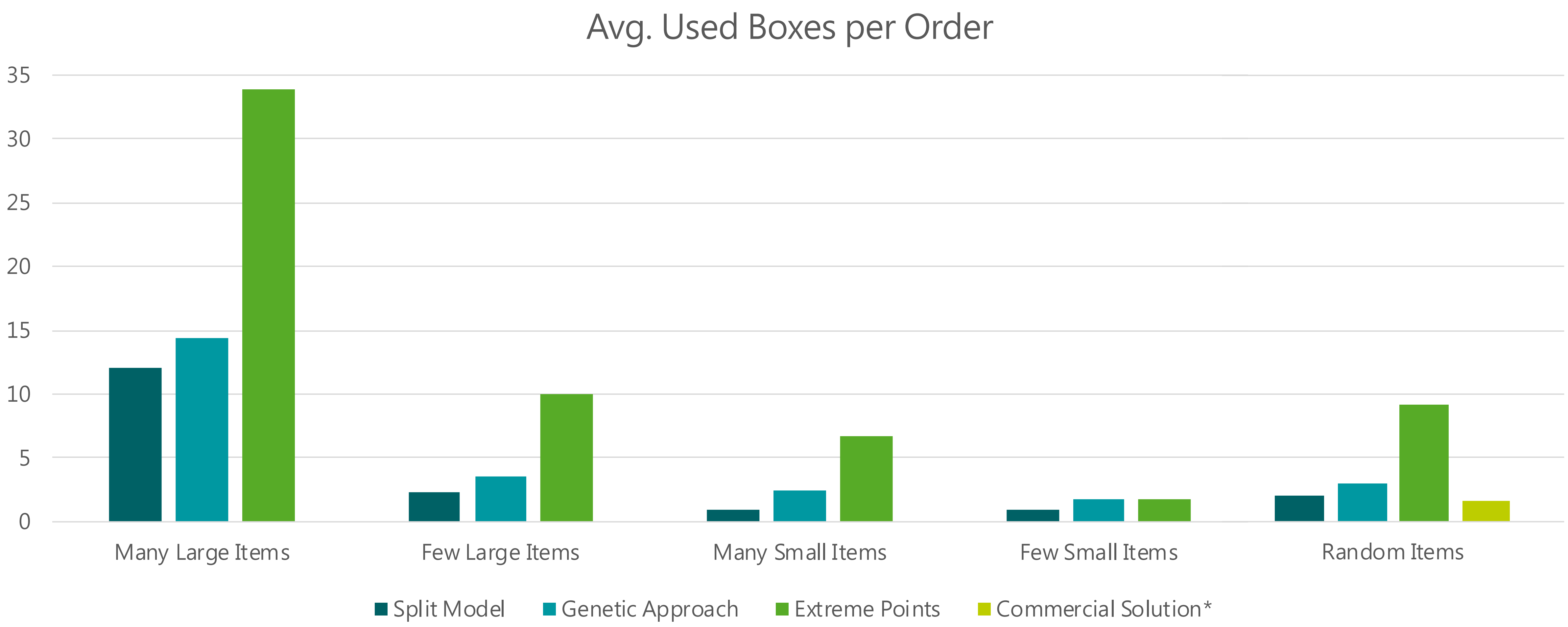
- PFSP considers only one type of container which is generally the case with pallet packing
- PFSP inherently builds a pyramid arrangement of items, thereby facilitating easy loading and unloading for human packers
- PFSP can easily be combined with other approaches and is also very fast

Agenda

1. Introducing to the problem
2. Tackling the problem: Assign Items to Boxes
   i. Exact Approach: Split Model
   ii. Heuristic Approach: Genetic Algorithm & Best Match Heuristic
   iii. Heuristic Approach: Extreme Points
3. Tackling the problem: Assign Boxes to Pallets
   i. Heuristic Approach: Peak Filling Slice Push
4. Comparing the Results
5. Working with our Python Package
6. Live Demo

https://www.flaticon.com/de/autoren/eucalyp

Comparing the Results: Used boxes

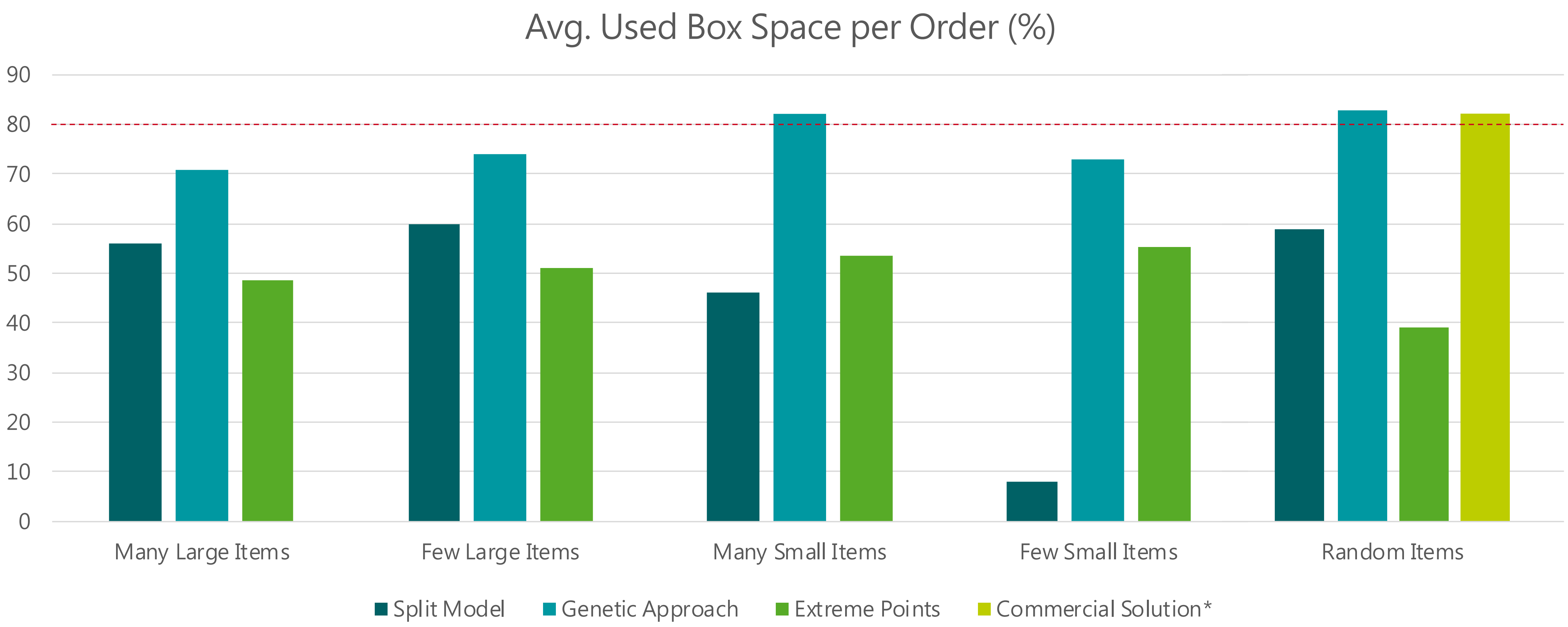


*: 3Dbinpacking (https://www.3dbinpacking.com/en/). All benchmarks for this solution were generated using the demo tool on the website.

Comparing the Results: Spent time

*: 3Dbinpacking (https://www.3dbinpacking.com/en/). All benchmarks for this solution were generated using the demo tool on the website.

Comparing the Results: Used box space



*: 3Dbinpacking (https://www.3dbinpacking.com/en/). All benchmarks for this solution were generated using the demo tool on the website.

Comparing the Results: Conflict of objectives

SM

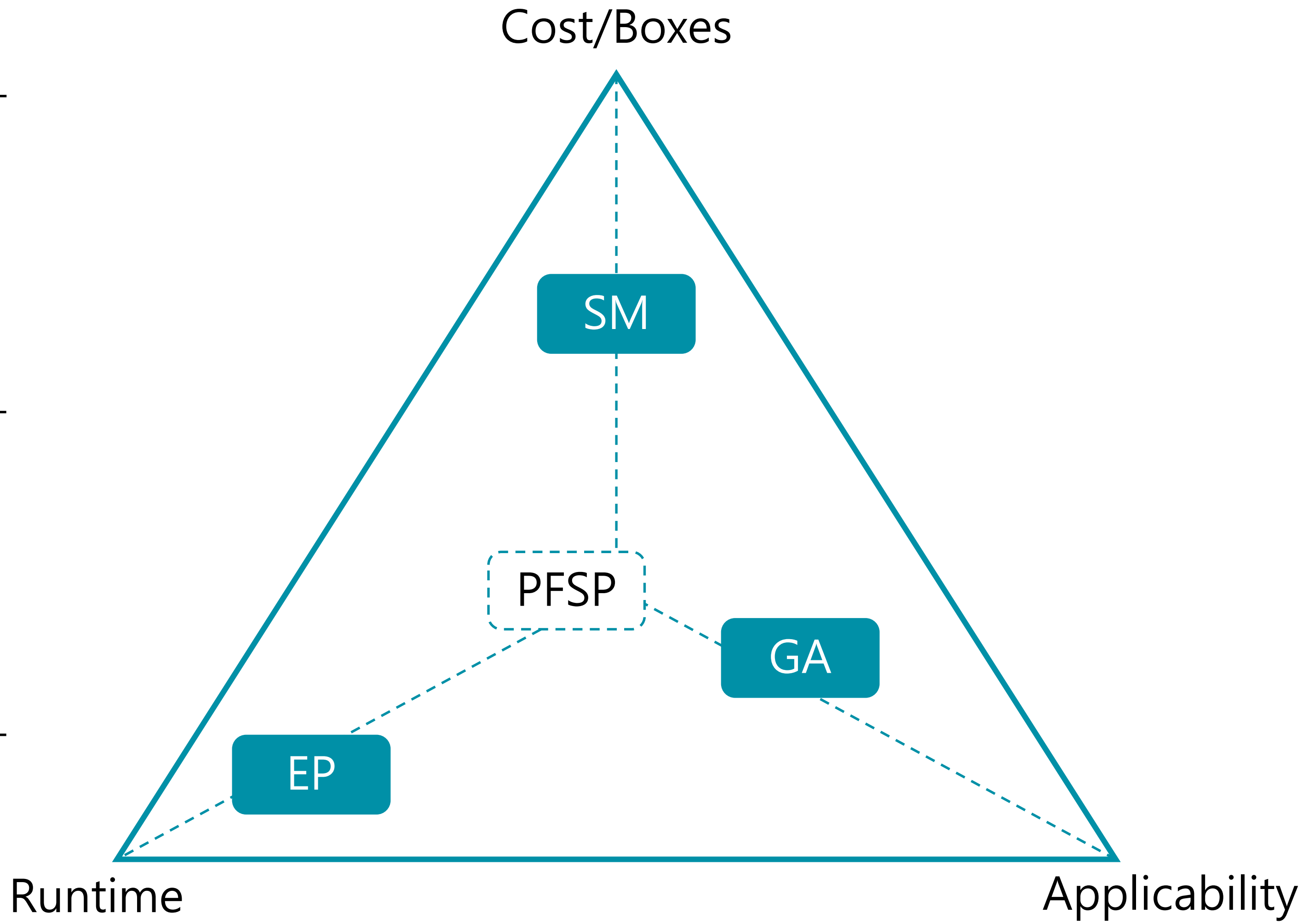
+	−
▪ Exact mathematical formulation ▪ Multiprocessing	▪ Slow compared to heuristics ▪ Relies on third-party software (Gurobi)

GA

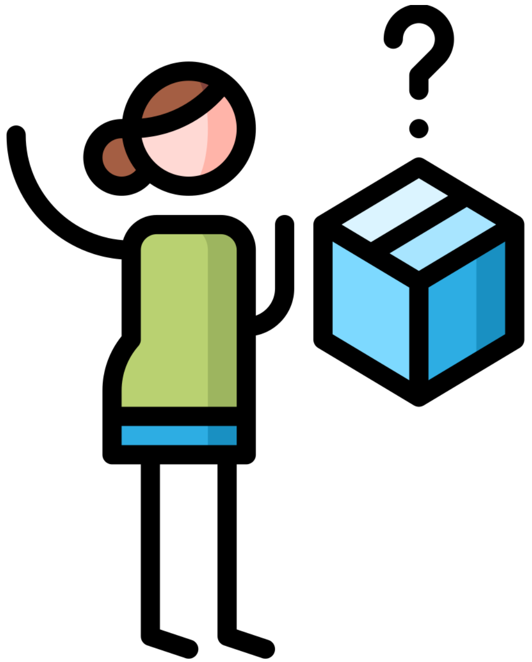
+	−
▪ Relatively fast ▪ Minimal waste of space ▪ Customizable	▪ Solution might be more expensive

EP

+	−
▪ Very fast ▪ Customizable	▪ Wasteful for significantly different items



Comparing the Results: Rough guideline

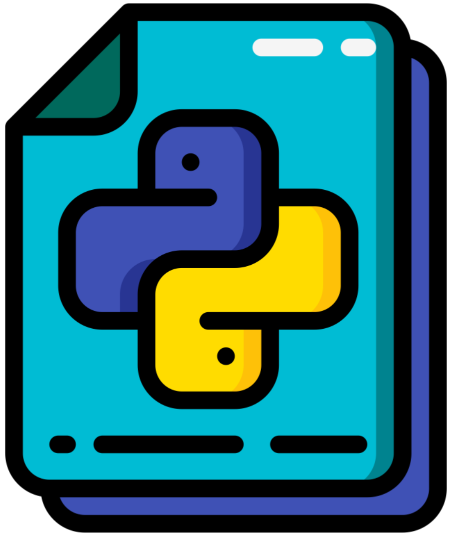


- Time Critical → Extreme Points Heuristic
- Cost Critical → Split Model
- Balanced → Genetic Algorithm & Best Match Heuristic

Pallet Packing → Peak Filling Slice Push Heuristic

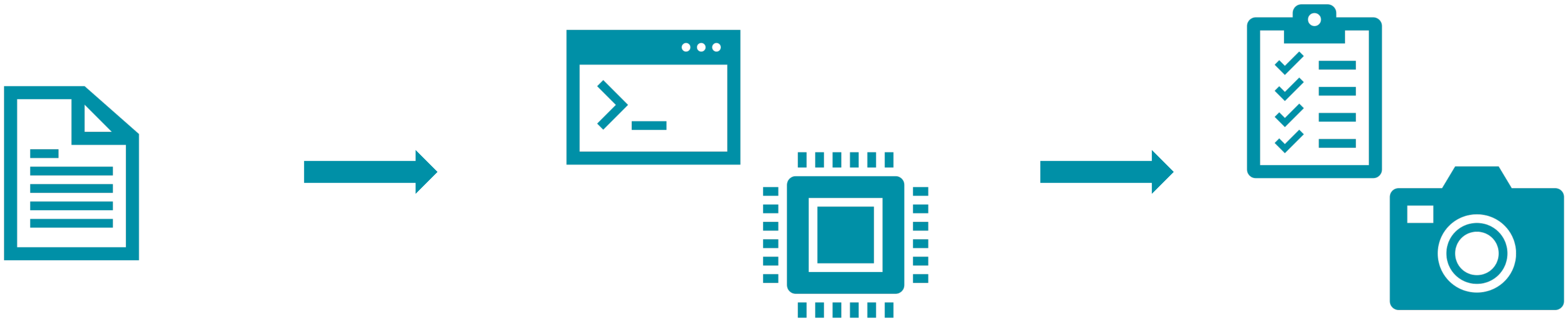
https://www.flaticon.com/de/autoren/pixel-perfect, https://www.flaticon.com/de/autoren/freepik

Agenda



https://www.flaticon.com/authors/smashicons

Working with our Python Package: Workflow



Input data in .csv format

Select orders & parameters and run the calculations

Obtain packing instructions and visualizations

Working with our Python Package: Quickstart guide

- Install our Python package using wheel
- All presented modules can be used via command line
- Start the desired solving approach by executing its dedicated run script
- Detailed documentation of customization options available via -h/--help
- What's needed as input:
  1. .csv file containing the orders
  2. .csv file containing the box dimensions

```
usage: run_split_model.py [-h] [-f FIRST] [-l LAST] [--version]
                          input boxes output

positional arguments:
  input                 path to the input file to be read
  boxes                 path to the csv file containing the dimensions of all
                        boxes
  output                path to the folder to store all output files. Expected
                        to end on '\' or '/' depending on the operating system

optional arguments:
  -h, --help            show this help message and exit
  -f FIRST, --first FIRST
                        order ID of the first order in the input file to be
                        packed. If none is given the packing will start with
                        the first order in the input file
  -l LAST, --last LAST  order ID of the last order in the input file to be
                        packed. If none is given the packing will end with the
                        last order in the input file
  --version             show program's version number and exit
```

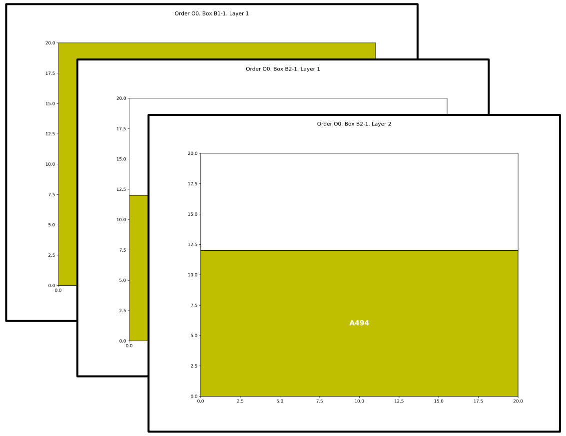
Working with our Python Package: Output & visualization

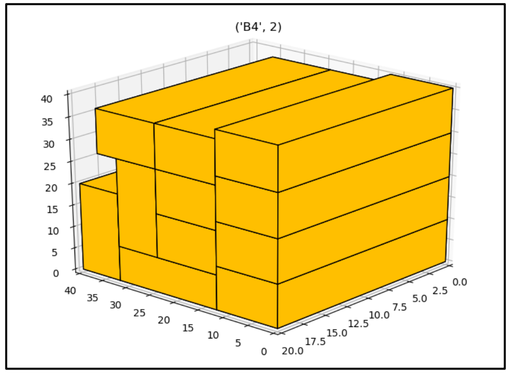
```
Packing instructions for order O0
Place each item at the specified location on top of the previously packed items
-------------------------------------------------------
Prepare the following items:
10 x A494
3 x A96

Use a box of type B7
Rotate the box such that the length side faces you
-------------------------------------------------------
Place item of type A494 at position x: 0.0, y: 0.0
Align the height of the item along the length of the box
Align the width of the item along the width of the box

Place item of type A96 at position x: 10.0, y: 0.0
Align the height of the item along the length of the box
Align the length of the item along the width of the box

Place item of type A494 at position x: 20.0, y: 0.0
Align the height of the item along the length of the box
```





1. Textual instructions
2. Top-down 2D view
3. 360° 3D animation

- Directory structure:
  - Automatic creation of a directory for each processed order to enable easy access to the generated instructions

Summary

Conclusions:

- The choice of the model depends on your needs
- Each model has advantages and disadvantages
- Visualization is a powerful tool that simplifies the packing process

Possible improvements:

- Further simplify the packing instructions
- Provide more sophisticated visualization tools
- Experiment with other heuristics
- Explore additional objective functions based on the business customer pricing models of different dispatchers

https://www.flaticon.com/authors/surang

Agenda

1. Introducing to the problem

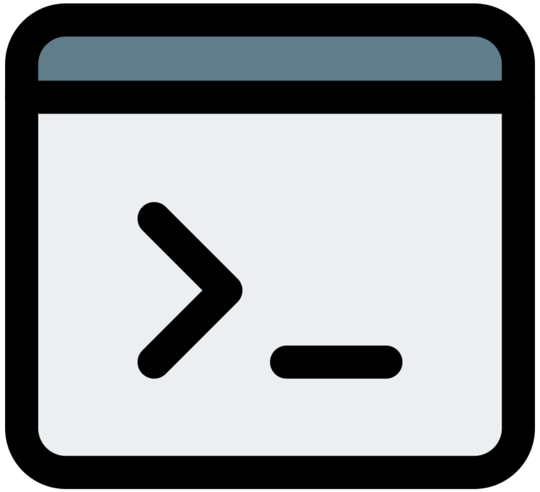
2. Tackling the problem: Assign Items to Boxes
   i. Exact Approach: Split Model
   ii. Heuristic Approach: Genetic Algorithm & Best Match Heuristic
   iii. Heuristic Approach: Extreme Points

3. Tackling the problem: Assign Boxes to Pallets
   i. Heuristic Approach: Peak Filling Slice Push

4. Comparing the Results

5. Working with our Python Package

6. Live Demo



https://www.flaticon.com/authors/pixel-perfect

Live Demo

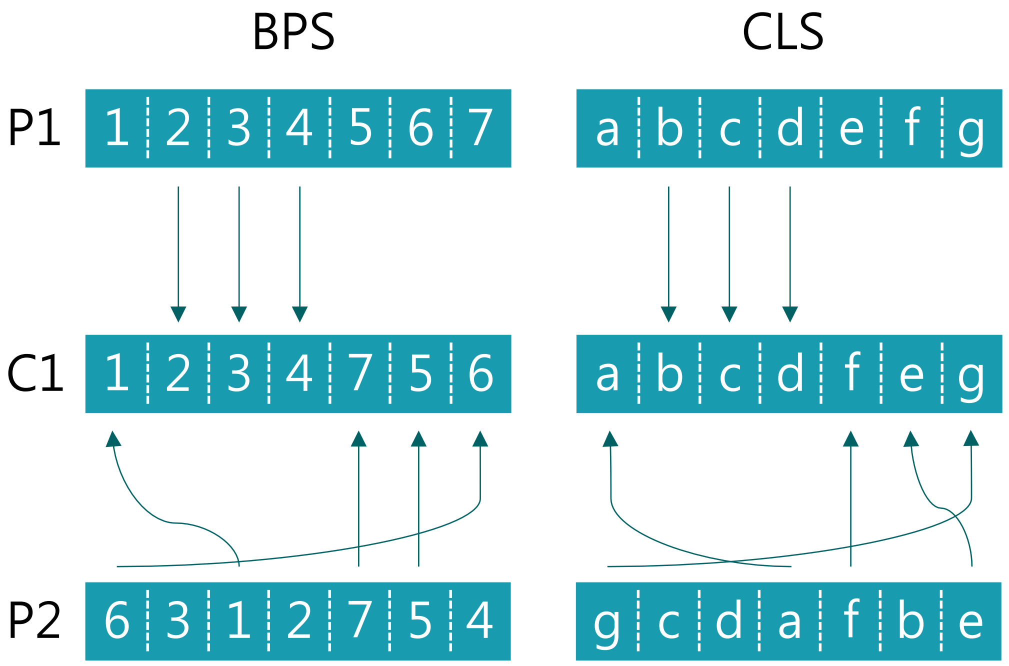
Thank you for your attention!

Any questions? Please feel free to ask!

Backup

Genetic Approach: Crossover/Mutation

- two parents generate two offsprings/children
- two cutting points are selected, named i,j with i < j
- Generating C1
  - Copy elements between i and j from P1
  - Missing elements by sweeping P2 circularly from j + 1
- Generating C2
  - Analogous to C1 with exchange of P1 and P2



Extreme Points: Merit Functions

Convention: the parts after *max* are introduced by the complexity of each merit function

1. Maximize free volume: pick the box which would be left with the most free volume after accommodating the item. Time complexity: $O(N^2 + \mathrm{N} * \max\{1{,}1\}) = O(N^2)$

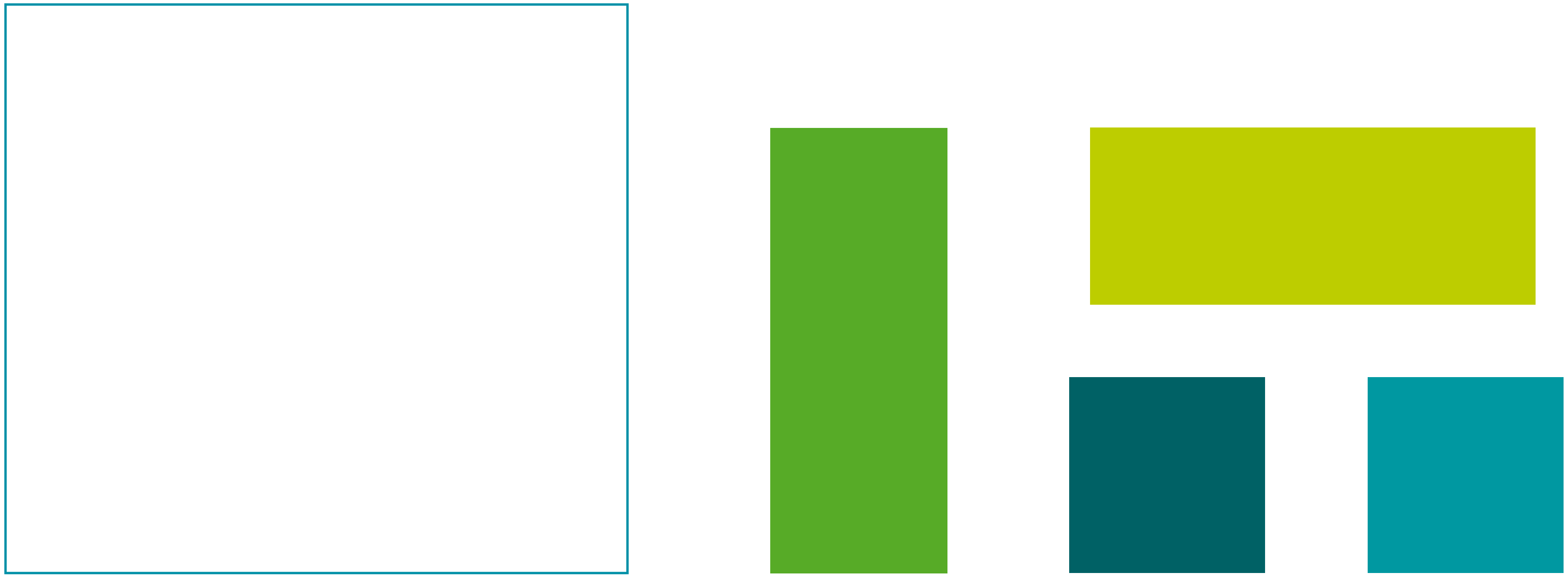
2. Minimize the maximum packing size: choose the box where either the item is placed on top or, if not possible, the box with the most free surface. Time complexity: $O(N^2 + N * \max\{1, N\}) = O(N^2)$

3. Level the EPs: choose the box whose EPs will have the least increase in height. Time complexity: as above. Time complexity: $O(N^2 + N * \max\{1, N\}) = O(N^2)$

4. Maximize the utilization of the Residual Space (RS). RS is roughly the same concept as EMSs in the Genetic Algorithm, namely the cubes defined by projecting the EPs to the walls of a box. Pick the box with the smallest RS still fitting the item.
   Time complexity: $O(N^2 + N * \max\{1, N\}) = O(N^2)$

Extreme Points: Why sort?

Example assuming sorting by area



Extreme Points: why sort?

With sorting:

Without sorting: