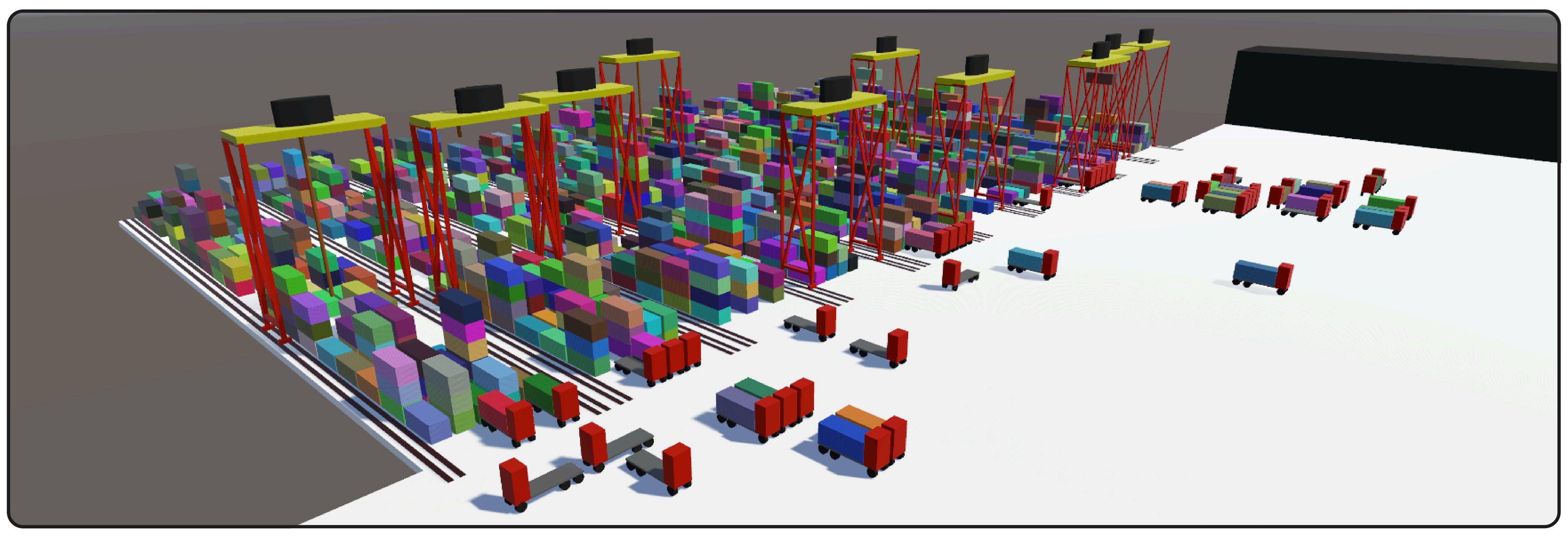
Planning for container warehouses

Author: Peter Kochelka Supervisor: Prof. RNDr. Roman Barták, Ph.D. 2025

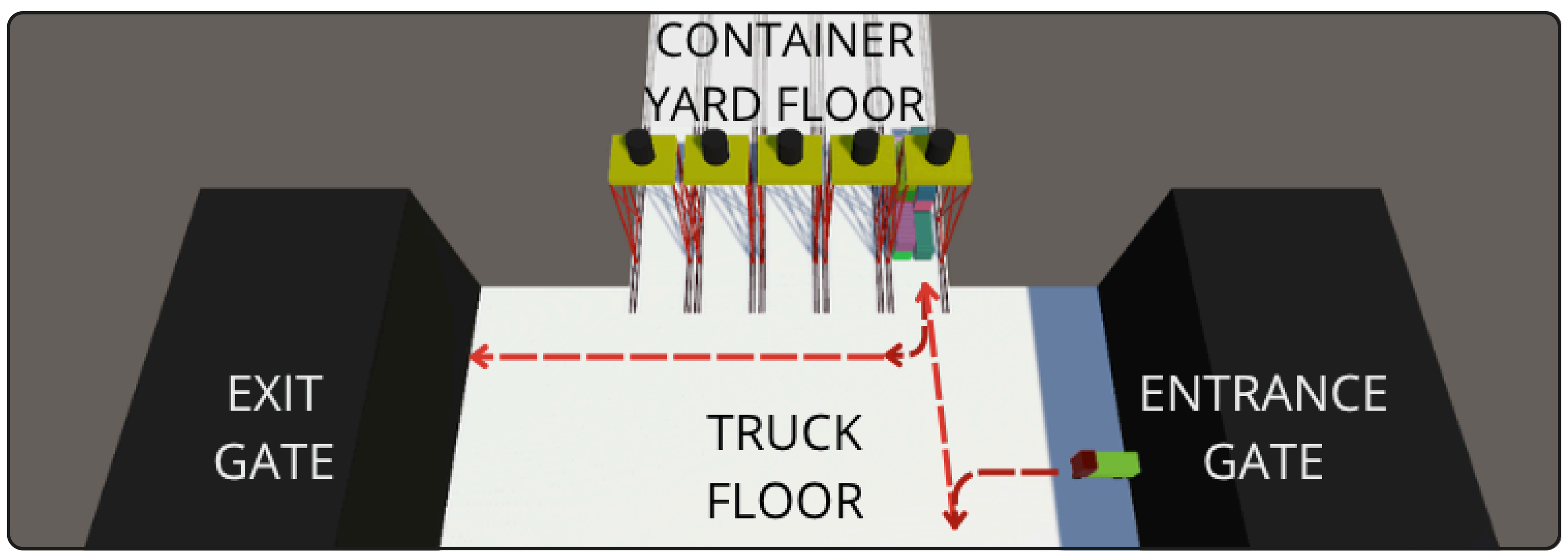
1 Introduction

- The surge in global trade demands better planning in container warehouses to **reduce time spent handling the containers**.
- We propose an **online truck path finding** approach with **decentralized collision and deadlock avoidance**. Although it does not guarantee optimality, its **flexibility and robustness to truck delays** make it practical for real-world use.
- It then serves as a basis for **experimental comparison of multiple stacking policies** in a custom-built simulation environment.



2 Problem definition

- A warehouse model in which **gantry cranes** and **trucks** cooperate to **import, store, and export** the containers.
- Cranes move along fixed tracks in the container yard to stack or retrieve containers. Trucks transport **one container per trip**, reversing into crane loading areas for loading or unloading.
- A **problem instance** is specified by **warehouse dimensions** (number of cranes, rows, stacks, trucks, paths, and stack height) and **predefined container/truck sequences**.
- A **valid solution** completes all imports and exports following the predefined sequences and without collisions. Better solutions **minimize time and in-warehouse reshuffles.**



3 Solving approach

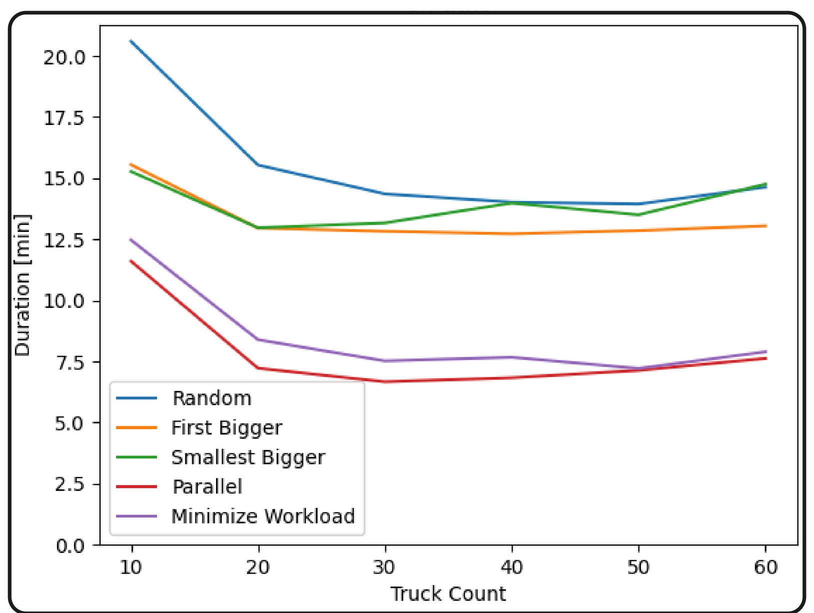
- Cranes operate on their own tracks and **cannot collide**. They load or unload trucks and prioritize exporting containers with higher priority.
- We propose six **stacking policies** for the online assignment of containers:
  - **First Free** – sequentially to first non-full stack.
  - **Random** – to a random non-full stack.
  - **First Bigger** – first non-full stack from among those without higher priority containers.
  - **Smallest Bigger** – non-full stack with the highest priority container from among stacks without higher priority containers.
  - **Parallel** – assigns cyclically to cranes, then it behaves as the Smallest Bigger policy.
  - **Minimize Workload** – assigns to crane for which the workload would increase the least (estimated by a heuristic), then it behaves as the Smallest Bigger policy.
- Entering trucks are directed to the row containing the stack assigned to their container (for import or export) and follow a **predefined path**.
- Trucks coordinate their movement in short time frames. In a **heuristical order based on proximity to exit** they **reserve movement areas** and then move in parallel.
- For sufficiently short time frames, this approach **provably prevents any collisions or deadlocks** from occuring and thus **produces a valid solution.**

4 Selected results

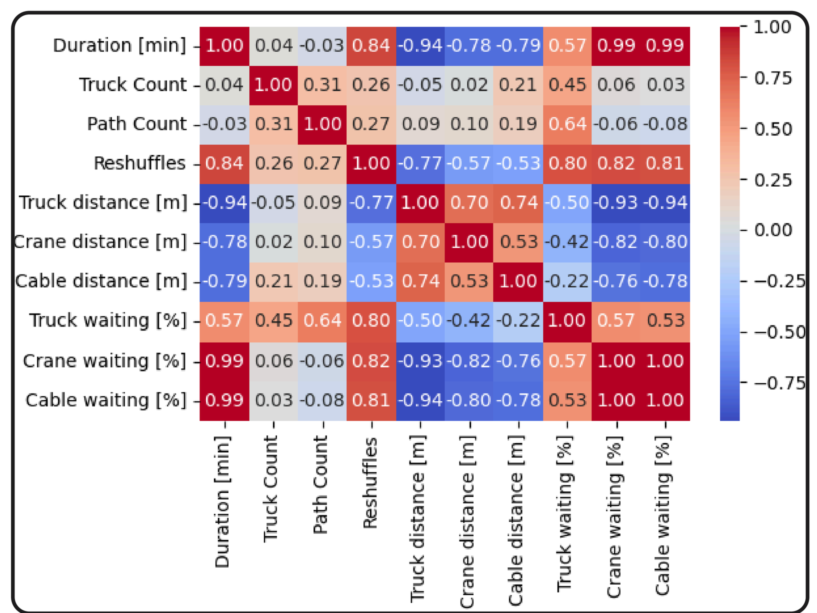
- We have **evaluated, compared, and improved the proposed stacking policies** by measuring how they affect the runtime of various simulated warehouses with 10 cranes, up to 60 trucks, and varying number of paths.

4.1 Truck count and performance

- At first, more trucks improve container throughput
- However, beyond around 30 trucks, the performance slowly worsens due to increased waiting and reshuffling.
- It is therefore important to find the optimal truck count for each layout.



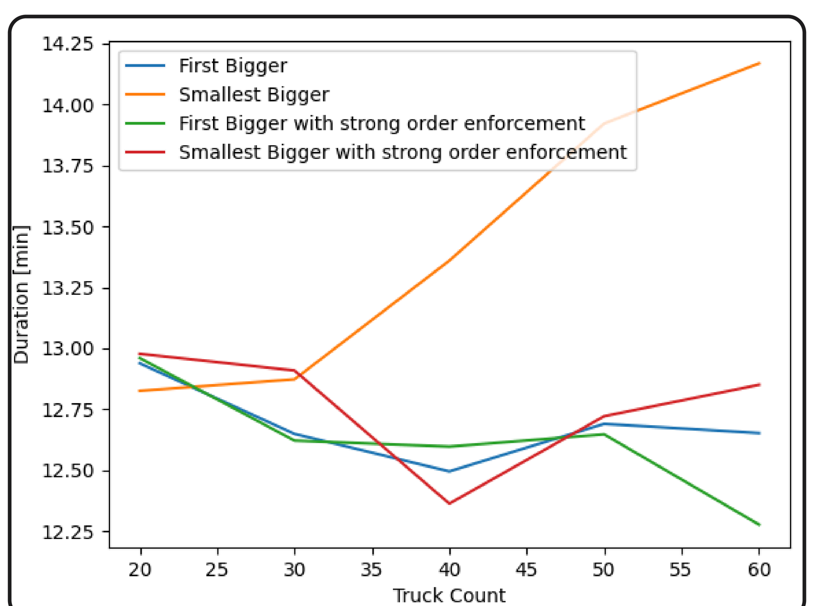
Simulation durations on 6 paths



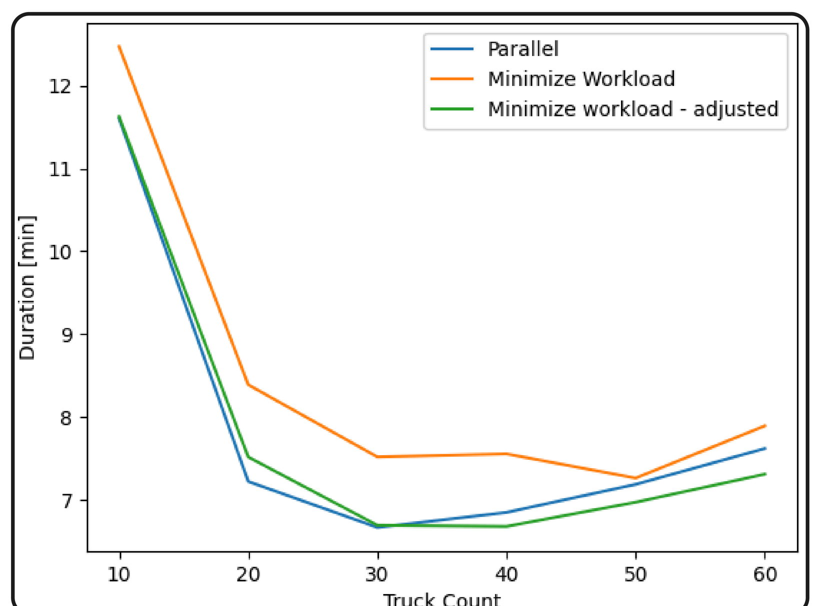
Statistics correlations for over 30 trucks

4.2 Policy comparison and improvement

- The **First Free** policy is half as fast as the **Random** policy and was therefore excluded from the graphs.
- The **First Bigger** and **Smallest Bigger** policies have similar runtimes for small truck counts.
- However, since both see **a tenfold increase in reshuffles** between 10 and 60 trucks, we **introduce stronger enforcement of truck order** to mitigate this, resulting in a **significant improvement** for both policies.
- The **Parallel** and **Minimize Workload** policies aim to utilize the available cranes and have them operate concurrently.
  - However, the straightforward cyclic Parallel policy outperforms the Minimize Workload policy, which **hints at imprecision** in how the workload heuristic is weighted.
  - We have therefore made an **adjustment in the weights** of the second policy heuristic, which resulted in it **successfully outperforming** the first one for higher truck counts.



Simulation durations on 10 paths



Simulation durations averaged

5 Conclusion

- Developed an approach for collision and deadlock avoidance of container trucks – more complex than traditional multi-agent pathfinding agents (points or symmetrical shapes).
- Designed and analyzed stacking policies, with **significant performance gains** based on measured statistics.
- Built an **experimentation visualization environment** in Unity 3D.

I would also like to thank my supervisor prof. Barták for his valuable guidance and support.

Contact:
kochelka.peter@gmail.com

Simulation environment source code:
gitlab.mff.cuni.cz/kochelkp/container-stacking