Storage Assignment and Scheduling Optimization

Kunlei Lian

September 12, 2023

1 Solution Approaches

There are some potential solutions to solve the storage assignment and scheduling optimization problem, which are detailed in the following sections.

1.1 Dynamic Assignment and Scheduling

In the ideal case, storage assignment or reassignment could take place at the same time with order fulfillment. Starting with an initial, potentially randomized, storage plan, order lines could be continuously fulfilled by moving the requested frames to the picking stations. In the meantime, an optimization engine could look into the incoming order lines and recognize those orders with high volumes and re-assign those corresponding frames into storage units closer to the picking stations. There are a couple of challenges with this approach:

- The optimizer must be executed in an online fashion such that it can continusouly monitor incoming order lines and make recommendations to re-assign frames.
- The optimizer itself must be efficient enough in order not to block the online order fulfillment process.
- The re-assignment of frames can not interfere with order line fulfillment, which involves more sophisticated frame movement scheduling and coordination.

To reach a trully dynamic storage assignment and scheduling state would require lots of development and testing efforts, which is way beyond a few days' timeline.

1.2 Periodic Assignment and Scheduling

An intermediate, or baseline, solution could be dividing the order lines into batches and separate storage assignment from scheduling into discrete and alternating time periods. Specifically, the workflow of this approach could be:

1. Setup initial storage plan

2. Divide order lines into batches
3. Repeat until all batches are fulfilled
   (a) Retrieve a order batch
   (b) Optimize storage plans based on the order lines in the batch
   (c) Schedule the order lines to be fulfilled at the picking stations

This periodic assignment and scheduling approach utilizes a rolling horizon-based optimization framework. It decomposes the original problem into smaller problems and alternatively solves two easier subproblems, namely, order batching and storage re-assignment.

1.2.1 Order batching

The order batches could be of the same size or variable sizes. In addition, the generated order batches cannot be too small or too large. If the batches are too small, there would require frequent storage unit re-assignment; otherwise, the total travel distances of the frames might be too much if the order batches are too big.

1.2.2 Storage re-assignment

The inputs to this problem are:

- The existing storage configuration, denoted by e_{fij}. Note that $e_{fij} = 1$ if the frame f is assigned to storage unit at (i, j) and 0 otherwise.
- The distance between every storage unit (i, j) to the picking stations p, denoted by d_{ijp}. It is assumed that the distance could be approximated using the Manhattan distance.
- The next batch of order lines to be picked, denoted by $\mathcal{B}$
  - All the frames required to fulfill the given order lines are denoted by $\mathcal{F}_B \in \mathcal{F}$ where $\mathcal{F}$ is the set of all frames.
  - The number of orders required for frame f by picking station p is indicated by n_{fp}

We define the following decision variables:

- x_{fij}: a binary variable that equals 1 if the frame f is assigned to the storage unit (i, j); 0 otherwise

- y_{fij}: a binary variable that equals 1 if the frame f is assigned to a different storage unit; 0 otherwise

The storage re-assignment problem could be modeled as:

$$\text{min.} \quad \sum_{p\in\mathcal{P}} \sum_{(i,j)\in\mathcal{S}} \sum_{f\in\mathcal{F}_B} x_{fij} \times d_{ijp} \times o_{fp} + \alpha \sum_{(i,j)\in\mathcal{S}} \sum_{f\in\mathcal{F}} y_{fij} \tag{1}$$

$$\text{s.t.} \quad \sum_{(i,j)\in\mathcal{S}} x_{fij} = 1, \ \forall f \in \mathcal{F} \tag{2}$$

$$\sum_{f\in\mathcal{F}} x_{fij} \leq 1, \ \forall (i,j) \in \mathcal{S} \tag{3}$$

$$y_{fij} \geq x_{fij} - e_{fij}, \ \forall f \in \mathcal{F}, (i,j) \in \mathcal{S} \tag{4}$$

$$y_{fij} \geq e_{fij} - x_{fij}, \ \forall f \in \mathcal{F}, (i,j) \in \mathcal{S} \tag{5}$$

$$x_{fij}, y_{fij} \in \{0,1\}, \ \forall f \in \mathcal{F}, (i,j) \in \mathcal{S} \tag{6}$$

In this formulation, the objective function (1) tries to minimize two components: 1) the total traveling distance of all the frames that are required to fufilled the next batch of order lines. 2) the number of re-assignments, where α is the weight given to the second objective. A bigger α value will discourage moving too many frames at a time. Constraints (2) make sure that a frame must be assigned to one and only one storage unit. Constraints (3) ensure that each storage unit can hold at most one frame. Constraints (6) define the variable types.

Note on runtime: the problem considered here is NP-hard and there generally does not exist solution algorithms that run in linear time. The solving time increases exponentially with problem size.