Zalando Logistics
Mathematical Optimization meets Machine Learning to optimize stock distribution

Amin Jorati, Applied Scientist
Francisco Madaleno, Junior Applied Scientist
Stephanie Ziegenhagen, Senior Applied Scientist

GOR/Real world optimization meeting
October 7, 2022

zalando

Agenda

Introduction

Demand Forecasting

Network Item Distribution

In-Warehouse Item Relocation

Large scale optimization problems need to be solved to deliver orders quickly and efficiently

-> need to predict what will be ordered where

Zalando at a glance

Agenda

Introduction

Demand Forecasting

Network Item Distribution

In-Warehouse Item Relocation

Demand Forecasting Problem

136, 175, 215, 180, 166, 235, 199, ?, ?, ?, …

Time Series Forecasting Problem

136, 175, 215, 180, 166, 235, 199, 195, ?, ?, …

Moving Average (window_size = 4)

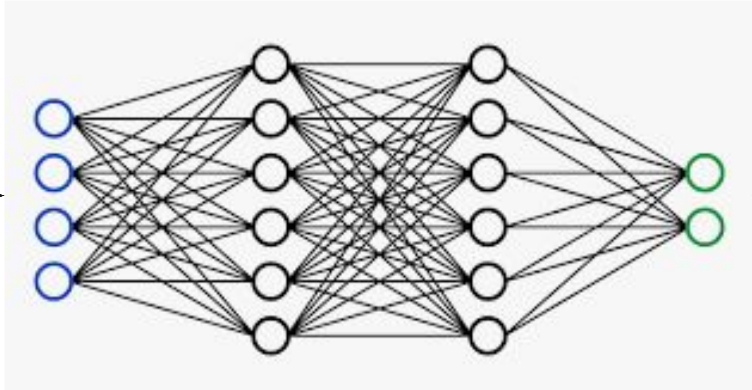
Probabilistic Demand Forecasting

136, 175, 215, 180, 166, 235, 199, X_t , X_{t+1} , X_{t+2} , …

- X_t **Random Variable**
- **Problems**
  - **Demand vs Sales**
  - **Article Data**

Deep Learning

- Sales
- Web Views
- App Views
- Season
- Brand
- Size
- …



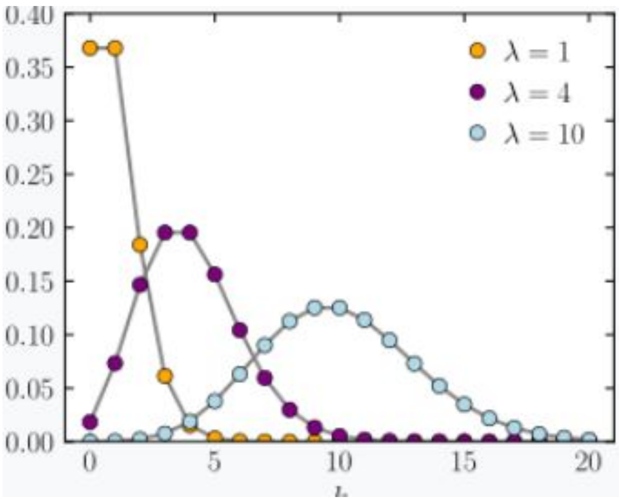
X_t

Deep Learning: Output

- Parameters of a distribution
  - Poisson distribution

$$f(k;\lambda) = \Pr(X{=}k) = \frac{\lambda^k e^{-\lambda}}{k!},$$

$$\lambda = \mathrm{E}(X) = \mathrm{Var}(X).$$



Deep Learning: how to train?

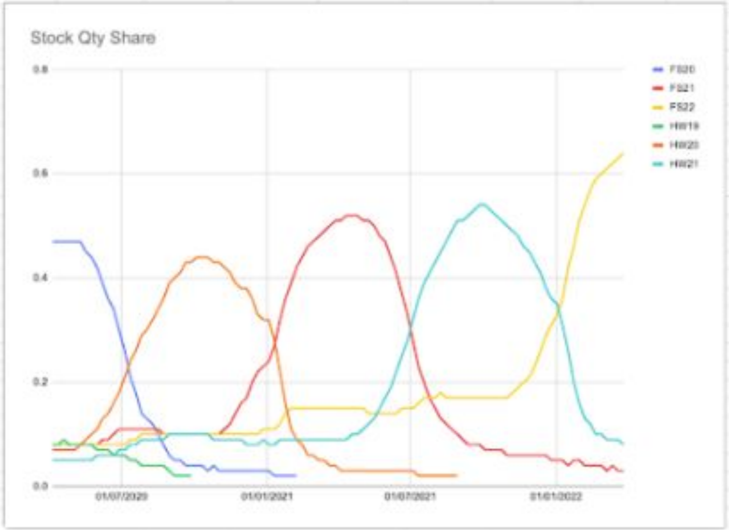
- Fit Parameters to observed data

$$
\begin{aligned}
NLL(\lambda, s) &= -\ln \mathbb{P}\left(D_\lambda = s\right) \\
&= -\ln\left(\frac{e^{-\lambda}\lambda^s}{s!}\right) \\
&= \lambda - s\ln(\lambda) + \ln(s!) \qquad \forall(\lambda, s) \in \mathbb{R}^{+*} \times \mathbb{N}
\end{aligned}
$$

- Negative-log likelihood (NLL)
  - Loss Function
- Architecture: Recurrent Neural Network

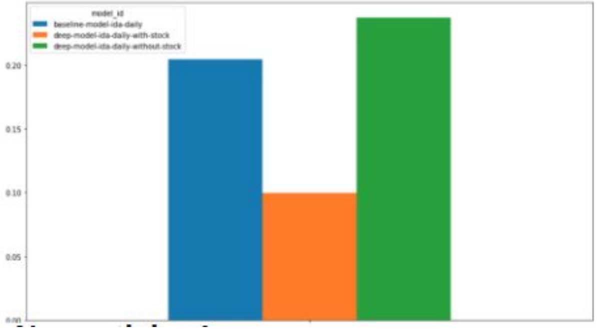
Forecasting: Challenges

- Seasonality of Articles
- Stock-outs
    - Sales = min (Demand, Stock)
- Quantifying uncertainty

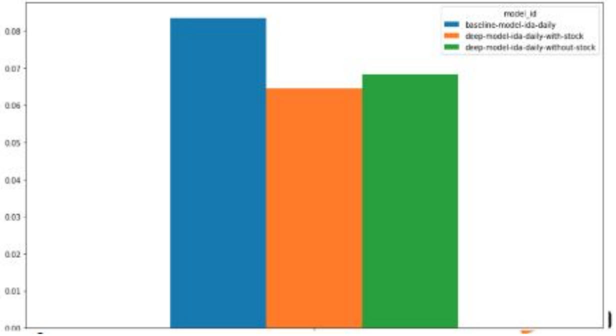


Forecasting: Challenges – stock-out

- Stock-outs
    - Mask data if not in stock



New Articles



Old Articles

- Seasonality
    - New Articles

Agenda

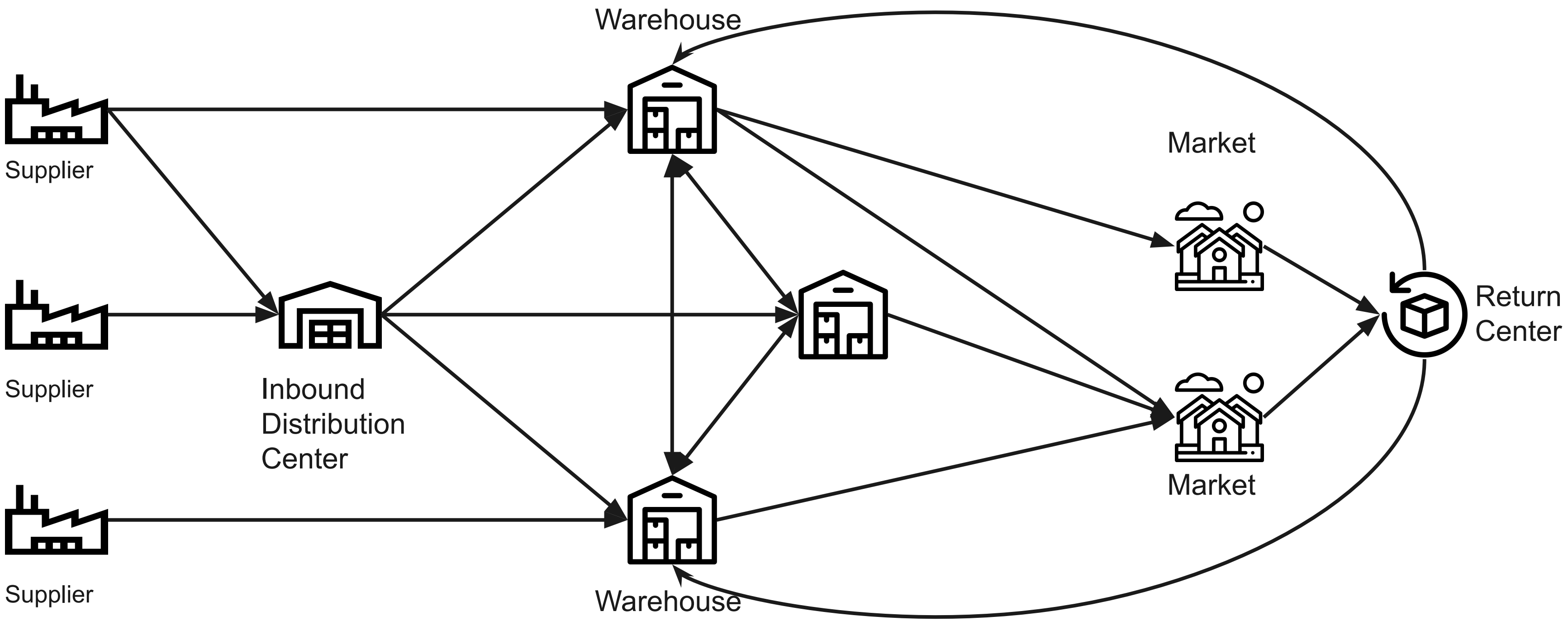
Introduction

Demand Forecasting

Network Item Distribution

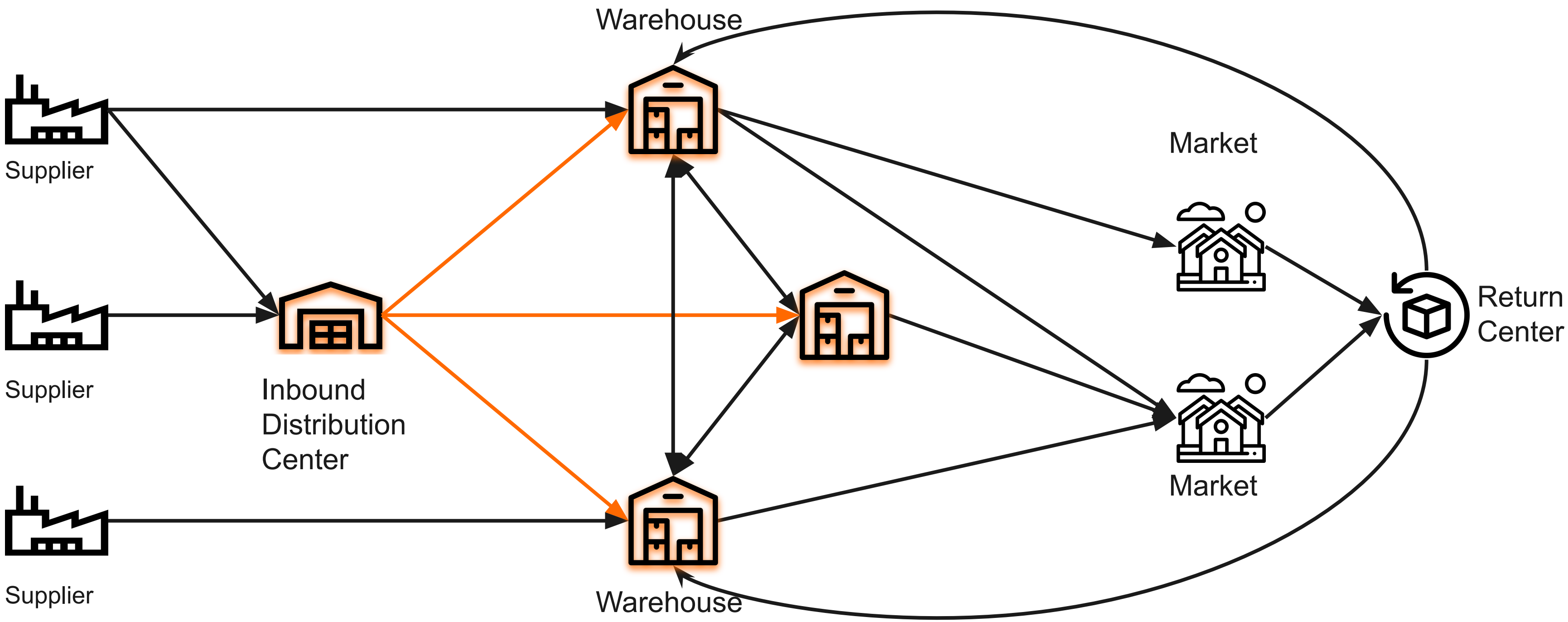
In-Warehouse Item Relocation

The Network Distribution Problem



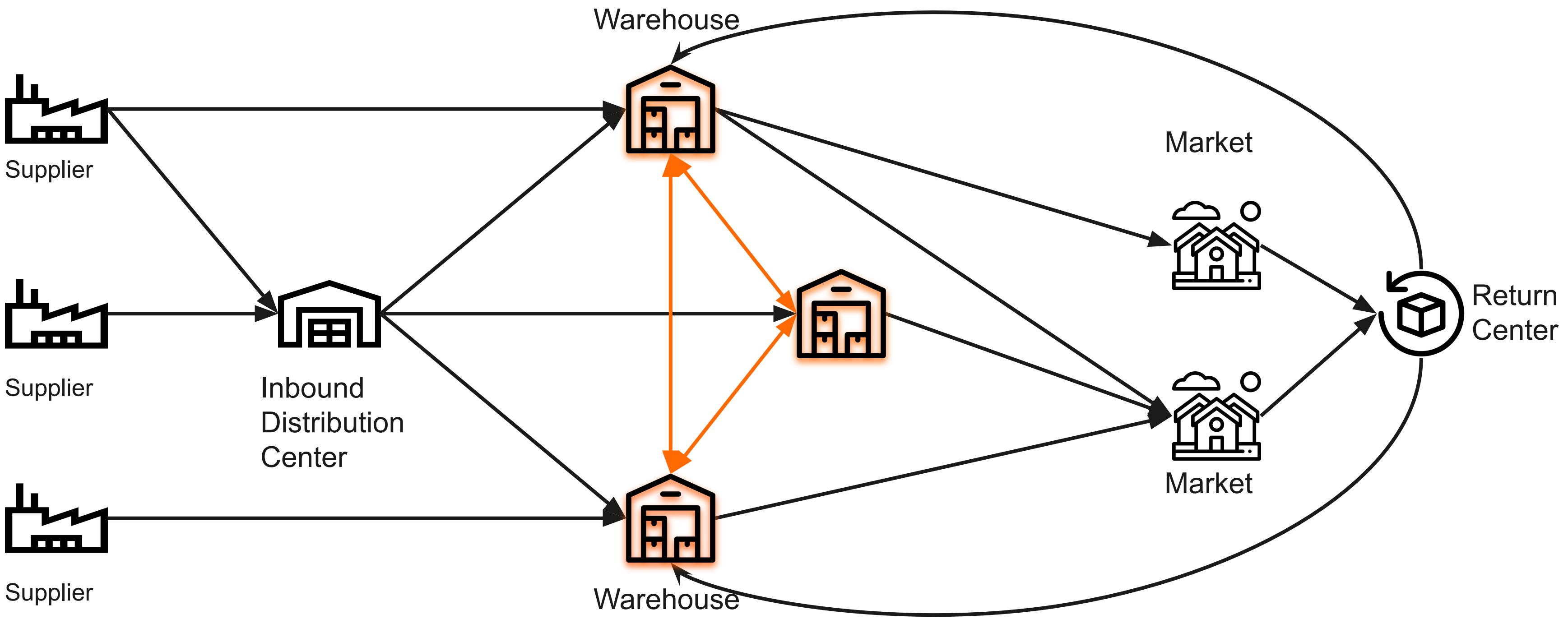
images: Flaticon.com

The Network Distribution Problem



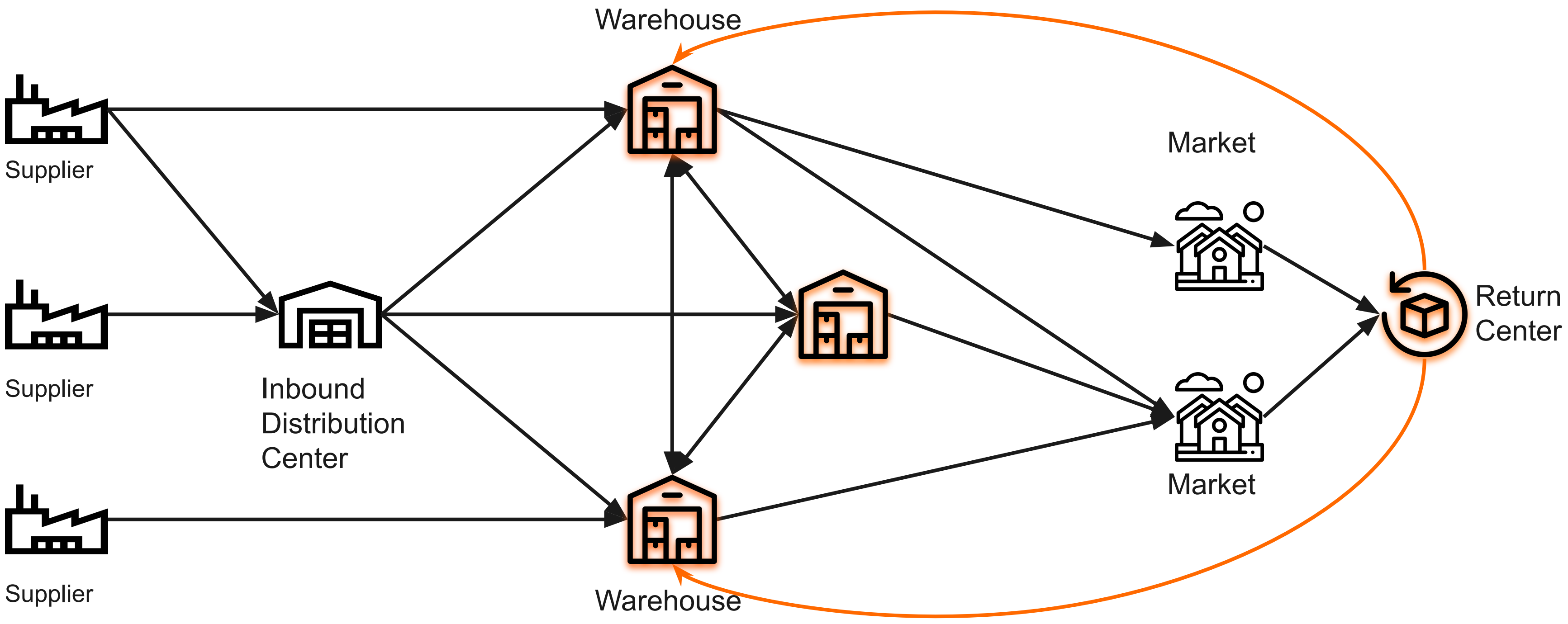
images: Flaticon.com

The Network Distribution Problem



images: Flaticon.com

The Network Distribution Problem

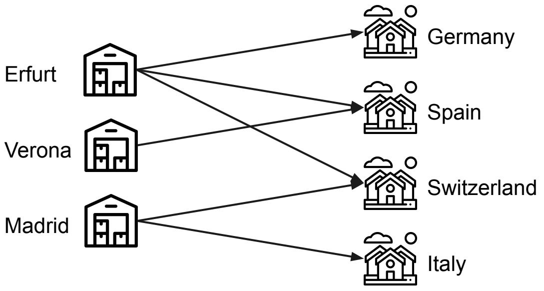


images: Flaticon.com

KPIs

Sales

Maximize the number of items sold.



Flawless Choice

Maximize the number of articles that are available in all countries where they can be sold.

Warehouse Movement Orders

Minimize the number of items that are moved between warehouses to fulfill an order.

images: Flaticon.com

Algorithm Outline

Stock State

The stock state represents the number of items of each article in each location (including transfers).

Evaluation of Stock States

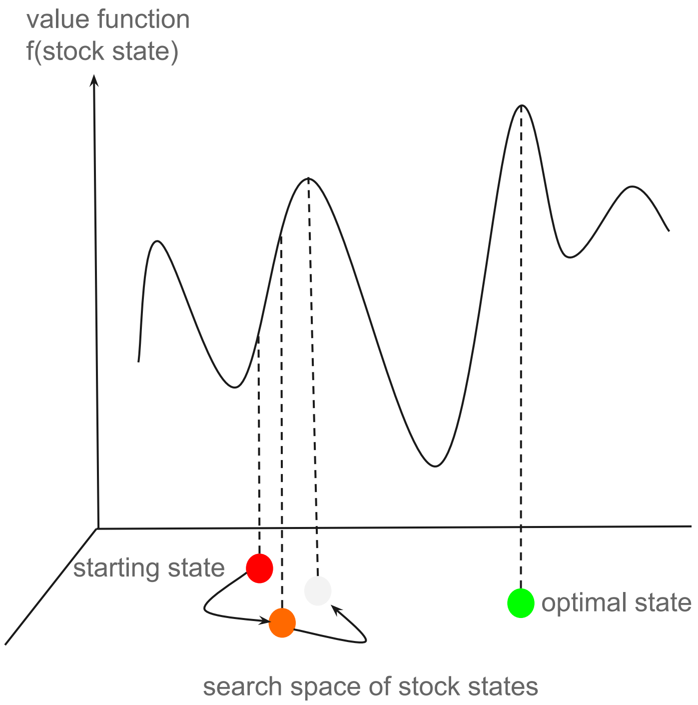
We assign a value to any stock state using some **value function**.

Item Distribution as an Optimization Problem

Given capacities for moving items between locations, find the stock state with the maximum value that can be reached from a given starting stock state through a series of movements using those capacities.

Local Search

A new stock state can be reached from a previous stock state by moving exactly one item from location A to location B.
We apply a simple hill climbing heuristic.



Evaluation of Stock States

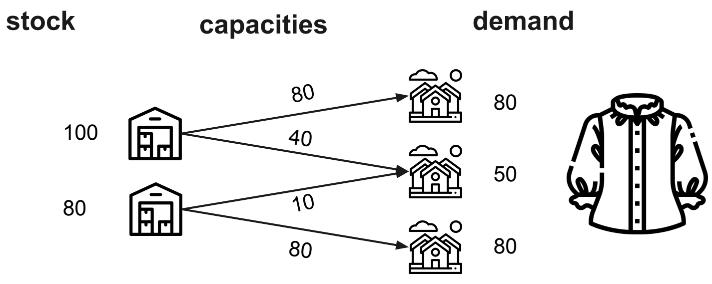
What is the value we gain from fulfilling customer orders from a given stock state?

Simulated Outbound Allocation

For a given article A, we use

- stock of A in each warehouse (stowed & transferred)
- expected demand of A for each market for each of the next 14 days
- linehaul capacities of the warehouse-market channels

Output: Expected #items sold via each warehouse-market channel.



Evaluation of Stock States

What is the value we gain from fulfilling customer orders from a given stock state?

Simulated Outbound Allocation

For a given article A, we use

- stock of A in each warehouse (stowed & transferred)
- expected demand of A for each market for each of the next 14 days
- linehaul capacities of the warehouse-market channels

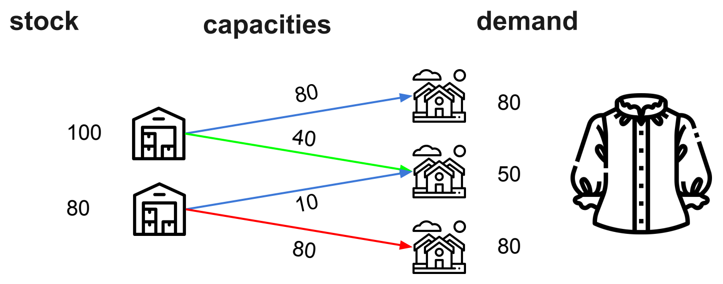
Output: Expected #items sold via each warehouse-market channel.

Evaluating expected sales

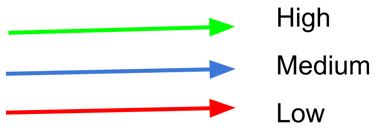
Assign to each stock state the value of its corresponding expected outbound allocation:

Value of allocations depends on

- fulfillment quality of the channels used
- a penalty for unbalanced usage of linehaul capacities



Fulfillment Quality Factors



AB-Testing Setup and Results

Algorithmic changes are evaluated via AB-testing:

- Split the universe of articles into different shards
- Split capacities according to shard size

Current baseline: Distribute stock so that the first time a customer wants to buy an item but can't is delayed as much as possible (+ guardrails)

Outlook

What’s next?

- Currently integrating handling returns
- Probabilistic handling of demand
- Using 8 weeks of demand forecast
- Be more optimal: Explore techniques besides simple hill climbing

Agenda

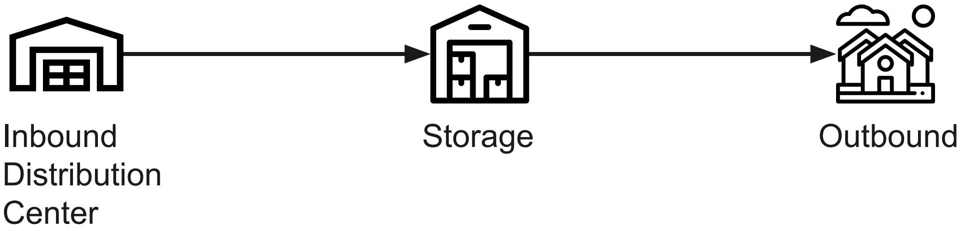
Introduction

Demand Forecasting

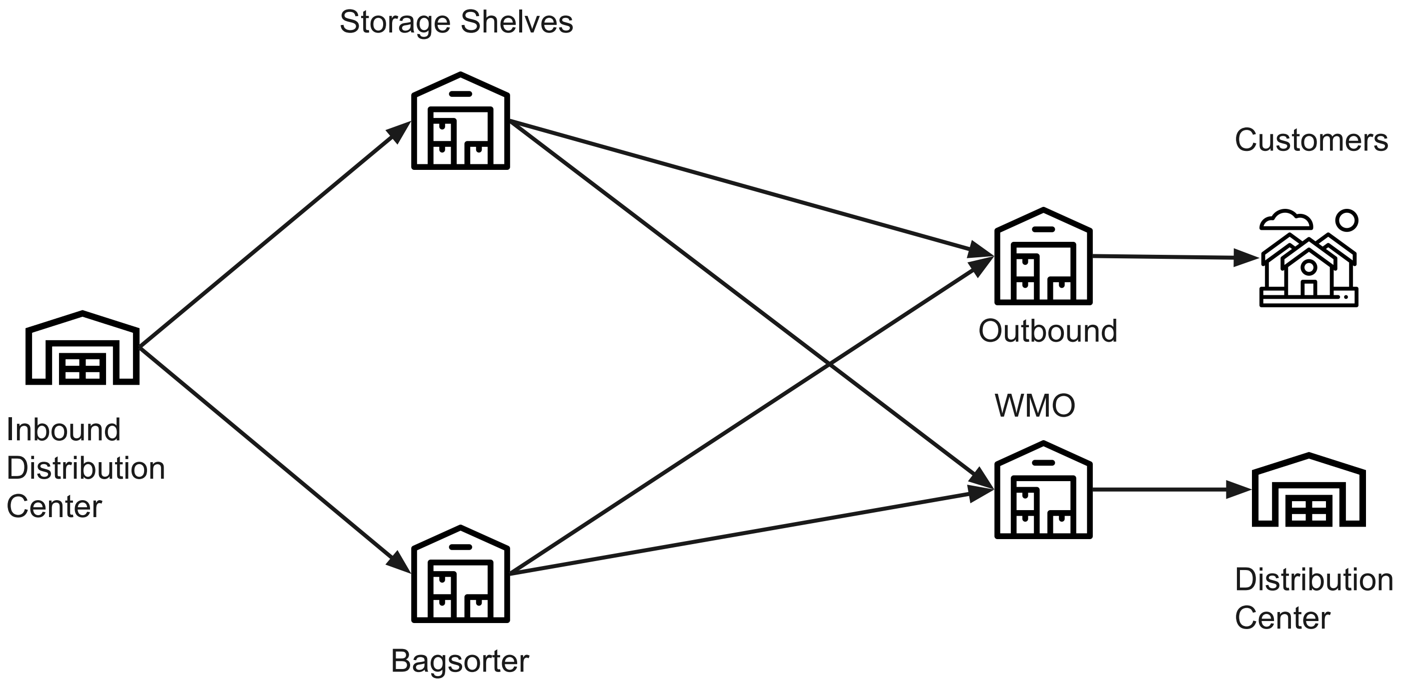
Network Item Distribution

In-Warehouse Item Relocation

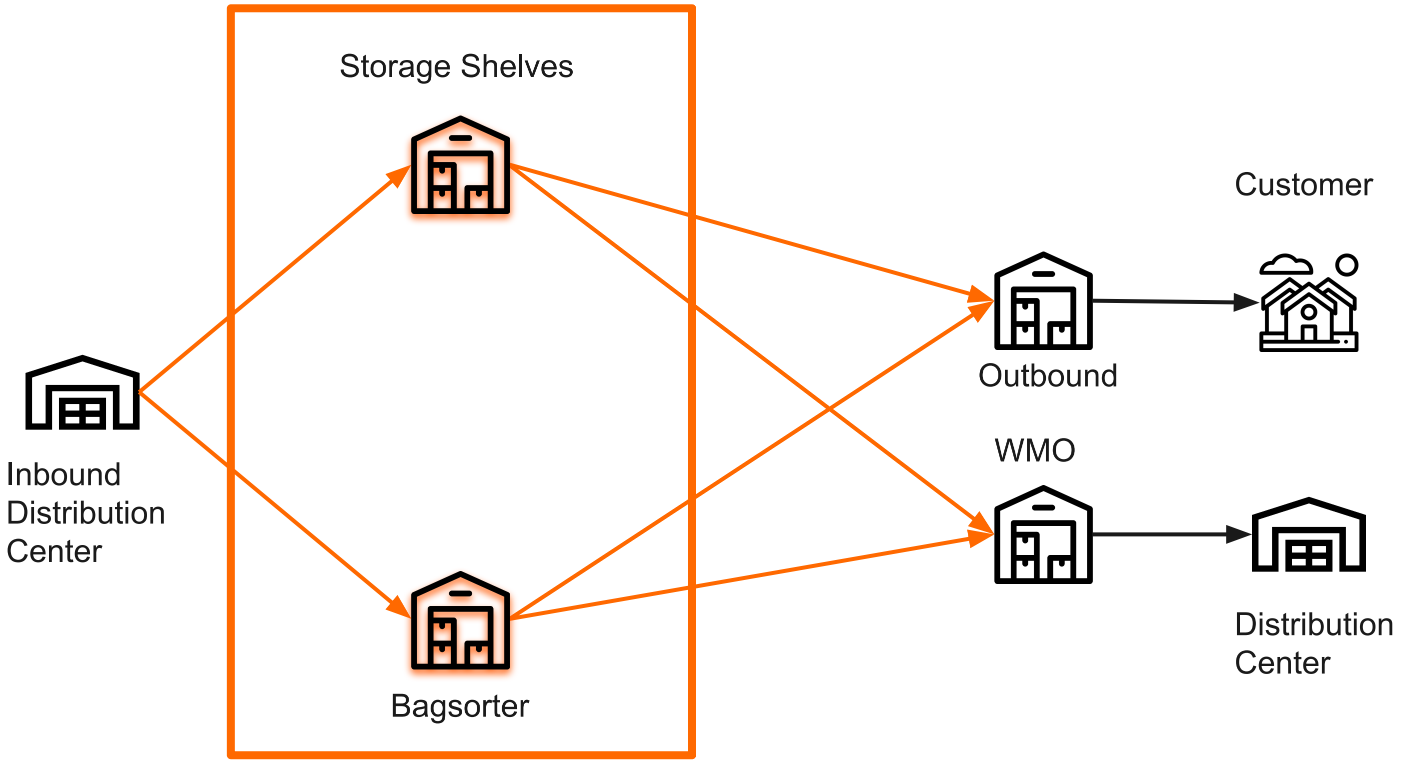
In-Warehouse Item Relocation



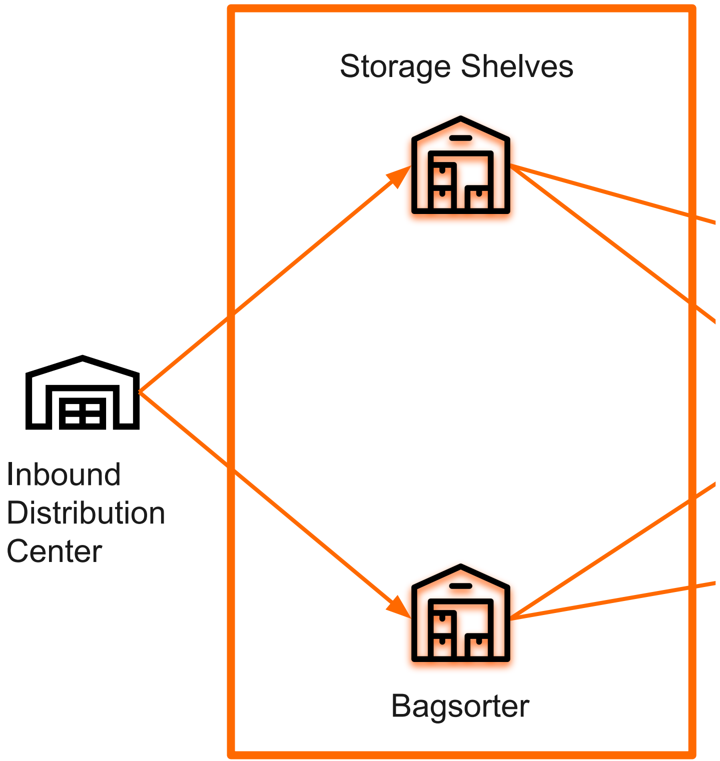
In-Warehouse Item Relocation



In-Warehouse Item Relocation

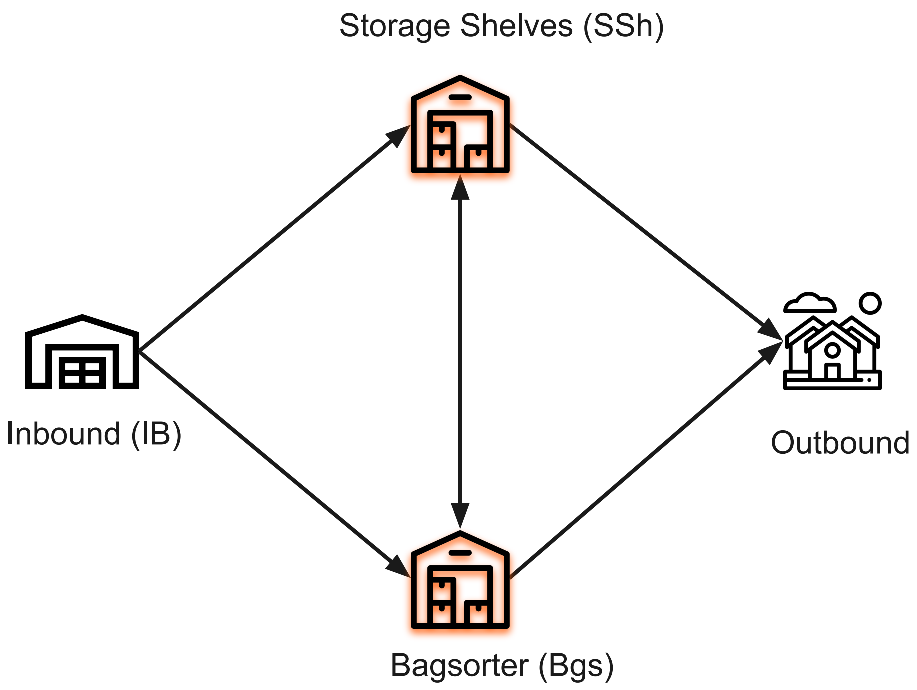


In-Warehouse Item Relocation



How to distribute load optimally?

Notation



Domain:

- Item
- Article (a set of similar items)
- Sales, demand, stock
- Time horizon

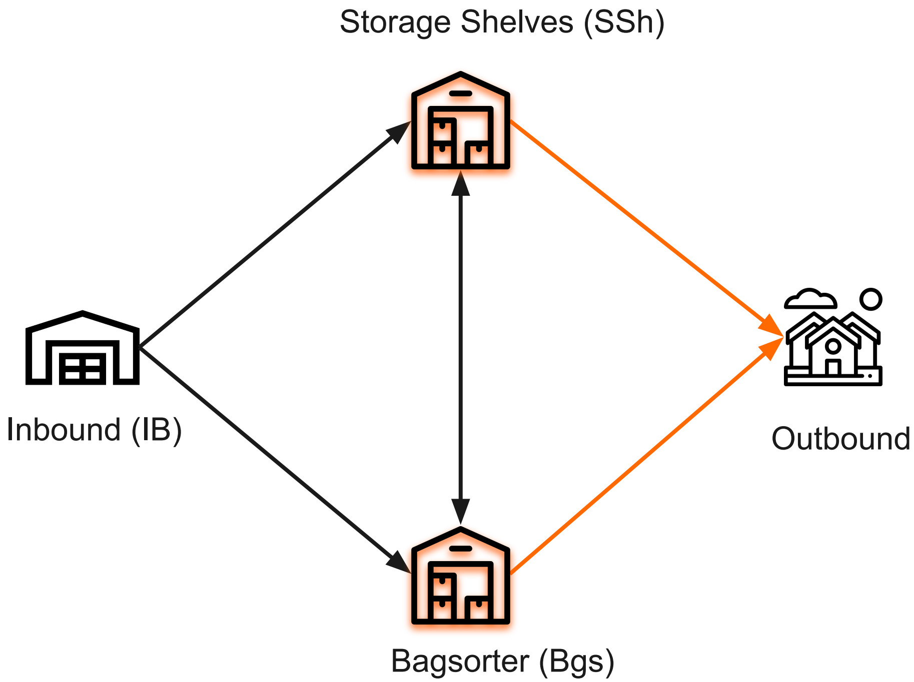
Input:

- Initial stock of items
- Storage capacity
- Demand per each Article
- Sales per Article per time
- Item to Article mapping
- Time horizon

Output:

- Set of movements per item per time

Notation

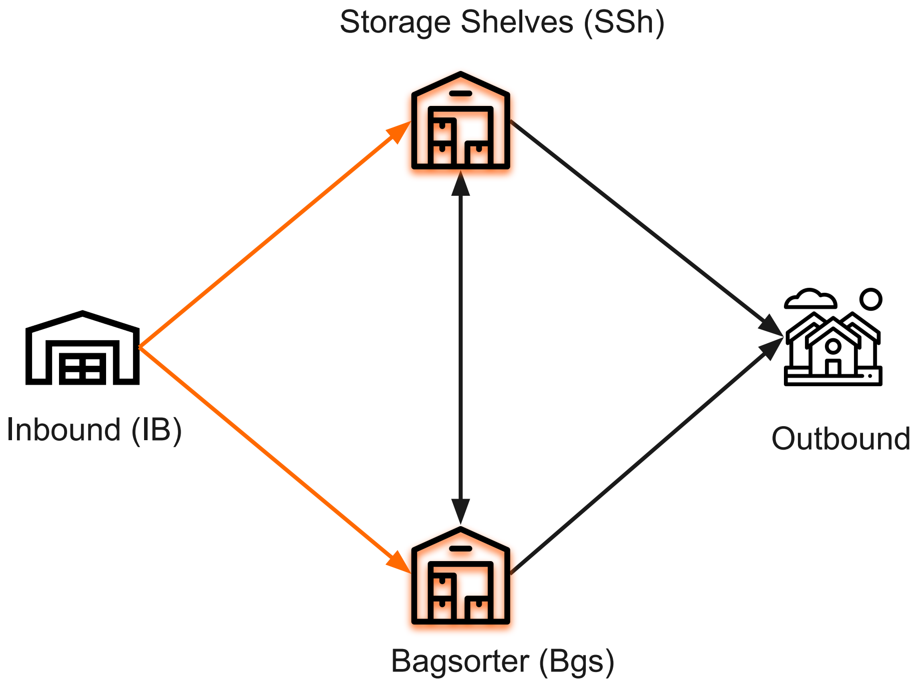


Domain:
- Item
- Article (a set of similar items)
- Sales, demand, stock
- Time horizon

Input:
- Initial stock of items
- Storage capacity
- Demand per each Article
- Sales per Article per time
- Item to Article mapping
- Time horizon

Output:
- Set of movements per item per time

Notation



Domain:

- Item
- Article (a set of similar items)
- Sales, demand, stock
- Time horizon

Input:

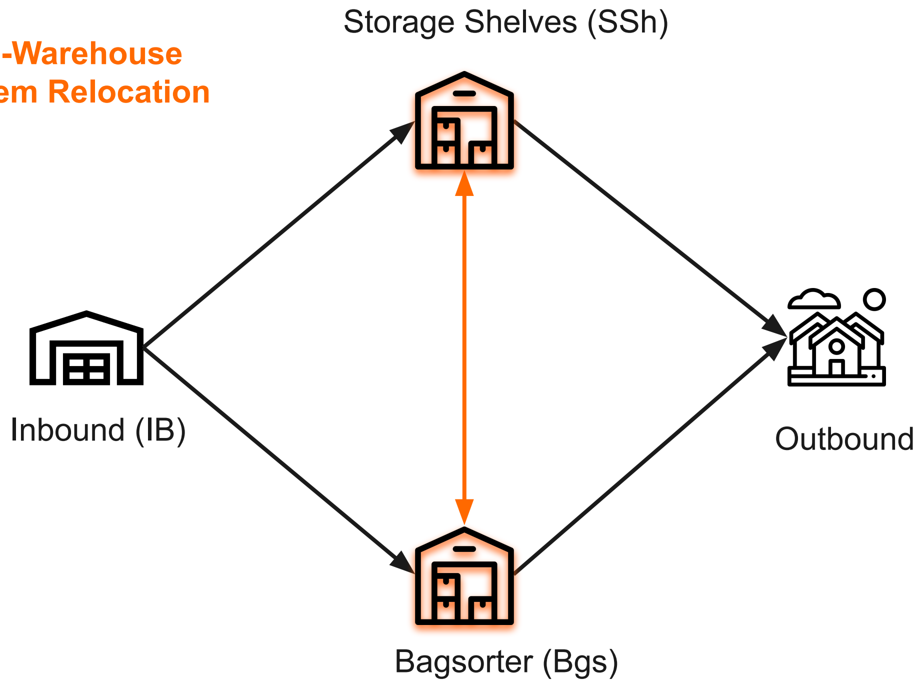
- Initial stock of items
- Storage capacity
- Demand per each Article
- Sales per Article per time
- Item to Article mapping
- Time horizon

Output:

- Set of movements per item per time

Notation

In-Warehouse Item Relocation



Domain:

- Item
- Article (a set of similar items)
- Sales, demand, stock
- Time horizon

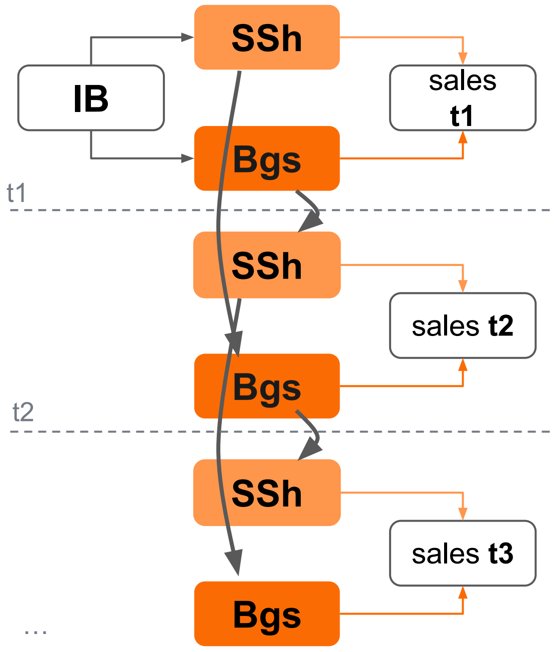
Input:

- Initial stock of items
- Storage capacity
- Demand per each Article
- Sales per Article per time
- Item to Article mapping
- Time horizon

Output:

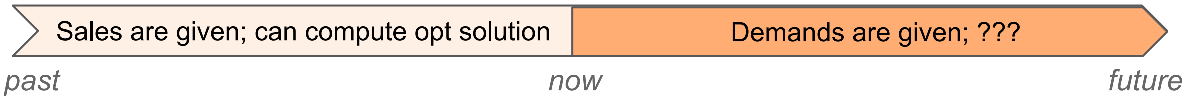
- Set of movements per item per time

Modelization



Input:

- initial stock in items
- capacity of each storage location
- demand per article per time *t (in the future)*
- sales per article per time *t (in the past)*
- **value of outbound channel:**
  SSh → Outbound = 1,
  Bgs → Outbound = 3
- item-to-article mapping ItA
- time horizon *T*



Algorithm evaluation:

Given sampled input,

1. define the optimal solution by solving the problem **using sales**
2. solutions generate movements by solving the problem **using demand**
3. estimate solutions: *obj(solution)/obj(opt)*

LP model: definition

Decision variables:

- movements of items:

$x_{i,bi,bf,t}$ item *i* moves from location *bi* to *bf* at time *t*

- stock (location) of items at time *t* :

$s_{i,b,t}$ item *i* located at *b* at time *t*

We sum movements in particular outbound channels

$$\max \sum_{i,bi,bf,t} \boxed{x_{i,bi,bf,t}} \cdot \boxed{v_{bi,bf}}$$

$$\text{s.t} \quad \sum_{bi,bf} x_{i,bi,bf,t} = 1 \qquad \forall i \in \mathbb{I}, t \in \mathbb{T} \qquad \text{(each item can only be moved once at a time)}$$

$$\sum_{i} x_{i,b_i,b_f,t} \leq C_{bi,bf,t} \qquad \forall b_i \in \mathbb{B}, b_f \in \mathbb{B}, t \in \mathbb{T} \qquad \text{(movements are under capacity)}$$

$$\sum_{bf} x_{i,bi,bf,t} \leq s_{i,bi,t} \qquad \forall i \in \mathbb{I}, bi \in \mathbb{B}, t \in \mathbb{T} \qquad \text{(movement in t is under stock in t)}$$

$$\sum_{bi} x_{i,bi,bf,t} = s_{i,bf,t+1} \qquad \forall i \in \mathbb{I}, bf \in \mathbb{B}, t \in \mathbb{T}/\{t_{last}\} \qquad \text{(stock in t+1 is fixed by movement in t)}$$

$$\sum_{i} x_{i,b,bf,t} \boxed{\cdot\, ISM_{i,j}} \leq n_{j,t} \qquad \forall j \in \mathbb{J}, b \in \mathbb{B}, bf \in \{\text{OUTBOUND}\}, t \in \mathbb{T} \qquad \text{(outflow is under demand)}$$

We match items to SKUs to satisfy demand/sales

LP model: definition

Decision variables:

- movements of items:

$x_{i,bi,bf,t}$ item *i* moves from location *bi* to *bf* at time *t*

- stock (location) of items at time *t* :

$s_{i,b,t}$ item *i* located at *b* at time *t*

We sum movements in particular outbound channels

$$\max \sum_{i,bi,bf,t} \boxed{x_{i,bi,bf,t}} \cdot \boxed{v_{bi,bf}}$$

$$\text{s.t} \sum_{bi,bf} x_{i,bi,bf,t} = 1 \qquad \forall i \in \mathbb{I}, t \in \mathbb{T} \qquad \text{(each item can only be moved once at a time)}$$

$$\sum_{i} x_{i,b_i,b_f,t} \le C_{bi,bf,t} \qquad \forall b_i \in \mathbb{B}, b_f \in \mathbb{B}, t \in \mathbb{T} \qquad \text{(movements are under capacity)}$$

$$\sum_{bf} x_{i,bi,bf,t} \le s_{i,bi,t} \qquad \forall i \in \mathbb{I}, bi \in \mathbb{B}, t \in \mathbb{T} \qquad \text{(movement in t is under stock in t)}$$

$$\sum_{bi} x_{i,bi,bf,t} = s_{i,bf,t+1} \qquad \text{(stock in t+1 is fixed by movement in t)}$$

$$\sum_{i} x_{i,b,bf,t} \cdot \boxed{ISM_{i,j}} \le \boxed{n_{j,t}} \qquad \text{(outflow is under demand)}$$

We match items to SKUs to satisfy demand/sales

Number of fulfilled items

Outlook

- In-Warehouse Item Relocation Problem
- Underutilization of Bgs → suboptimal distribution of item flow
- Model presented: LP
- What's next?
  - Create the software infrastructure to allow these movements
  - Make experiment with real data (instead of synthetic data like so far)
  - Productionize algorithm and evaluate if Bagsorter throughput improved
  - Fine tune the algorithm
  - Fine tune inbound algorithm and have a more holistic approach

Thank You