Warehouse Layout Optimization Using PRM and the Capacity Scaling Algorithm

Final Project

Luis Brena

October 2, 2024

Network Models

Summary

Problem description

Optimize aisles position and orientation in a cross-docking facility

Solutions and proposed algorithm

Use of RPM, capacity scaling algorithm and stochastic sensitivity analysis

Conclusions and recommendations

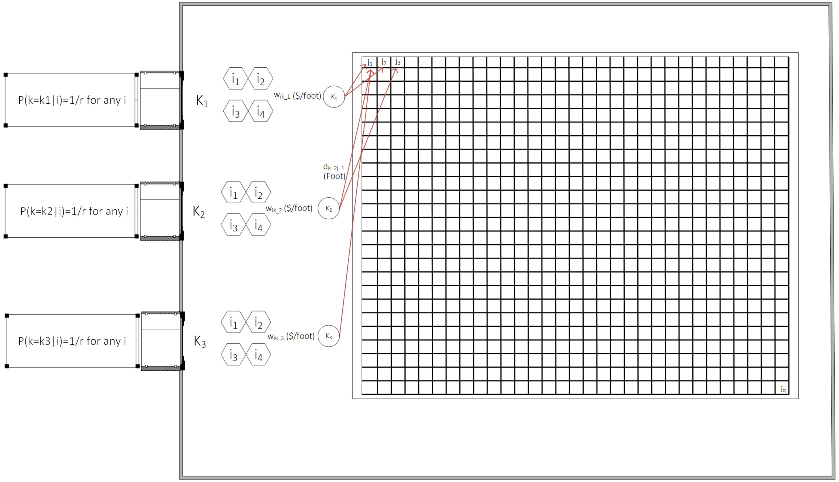
Alternative algorithms and results

Problem description

Motivation

- Lack of a simple solution for temporal or crossdocking warehouses.
- Finding an simple good sub-optimal solution to the warehouse layout design problem.
- Designing an algorithm that considers the most important cost variables in layout design.

Visual description



Linear program

Minimize

$$\sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}x_{ij}$$

Subject to

$$\textstyle\sum_{j=1}^{q} x_{ij} = x_{si} \quad \forall \ \ i = 1, \ldots, p$$

$$\sum_{i=1}^{p} x_{ij} = x_{jr} \quad \forall j = 1, \ldots, q$$

$$x_{si} \geq F_i$$

$$x_{jr} \geq 0, x_{jr} \leq 1 \ \ \forall \ \ j \leq q$$

$$\sum_{i}^{p} F_i \leq x_{rt}$$

$$x_{ij} \geq 0 \quad \forall \ \ i \leq p \text{ and } j \leq q$$

(1) (2) Arc balancing constraints (7) Demand constraint

Cost estimation and shortest path

The cost of every arc (i,j) is calculated as follows:

$$c_{ij} = w_{ik} d_{kj}$$

Where w_{ik} is the cost $(\frac{\$}{\text{foot}})$ of moving any item i from dock k. We consider items departing from different docks as different items. We want to find d_{kj}.

Minimize

$$\sum_{(u,v) \in A} x_{u,v} d_{u,v}$$

Subject to

$$\sum_{v:(u,v)\in A}^{n} x_{uv} - \sum_{v:(v,u)\in A} x_{vu} = \begin{cases} 1, v = k \\ -1, u = j \\ 0, \quad otherwise \end{cases}$$

Solutions and proposed algorithm

Literature Review

Zouein, P. P., & Tommelein, I. D. (December, 1999). Dynamic Layout Planning Using A Hybrid Incremental Solution Method. (JOURNAL OF CONSTRUCTION ENGINEERING AND MANAGEMENT). ASCE.

Rakesh, V., & Adil, G. K. (2015). Layout Optimization of a Three Dimensional Order Picking Warehouse. Shailesh J Mehta School of Management, Indian Institute of Technology Bombay, Mumbai, India.

Proposed solution

1. Creating a new layout adjusting parameters
2. Build a network using PRM
3. Find the shortest path using euclidean distance and Dijkstra’s algorithm
4. Use capacity scaling algorithm to find the minimum cost, optimally placing the items.
5. Repeat with a different configuration.

Creating a new layout

The proposed algorithm starts with the creation of a new layout. To do this, we defined a set of parameters.

Space size defined by boundaries and the origin: $B = (x_b, y_b)$

Number of aisles: $I = 3$, set to 3 for the example

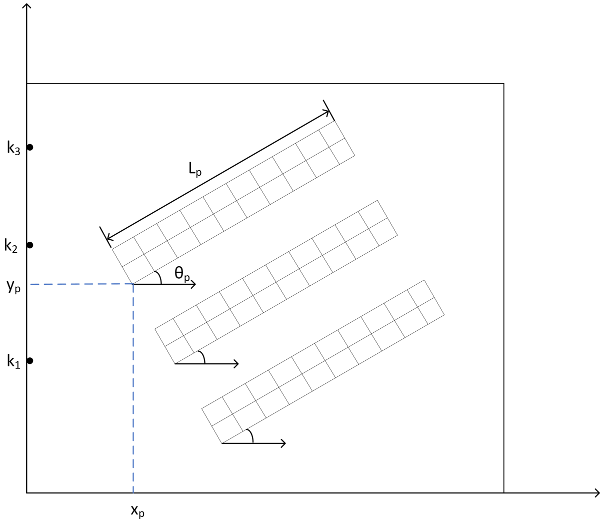
Most left extreme corner position: $pos_i = (x_p, y_p)$

Aisle longitude: L

Aisle orientation: θ

The docks position in the axis y: y_k

Creating a new layout



Build a connected undirected graph

1. Randomly sample nodes in the 2D space
2. Check for collisions and connect nodes using a circle of maximum distance
3. Add the terminal nodes j and source nodes k to the graph
4. Find the shortest path from k to j for all $k, j \in N'$
5. Save the shortest paths d_{kj}

Algorithm description

Generate sample points.

$$X = a_x + (b_x - a_x) \cdot rand[0, 1)$$

$$Y = a_y + (b_y - a_y) \cdot rand[0, 1)$$

Check for collisions:

$$x_i \geq x_0 + L \times tan(\theta) \ \ \forall \ \ 0 \leq L \leq L_1$$

$$y_i \geq x_0 - W \times tan(\pi - \theta) \ \ \forall \ \ 0 \leq W \leq 4 = \text{aisle width}$$

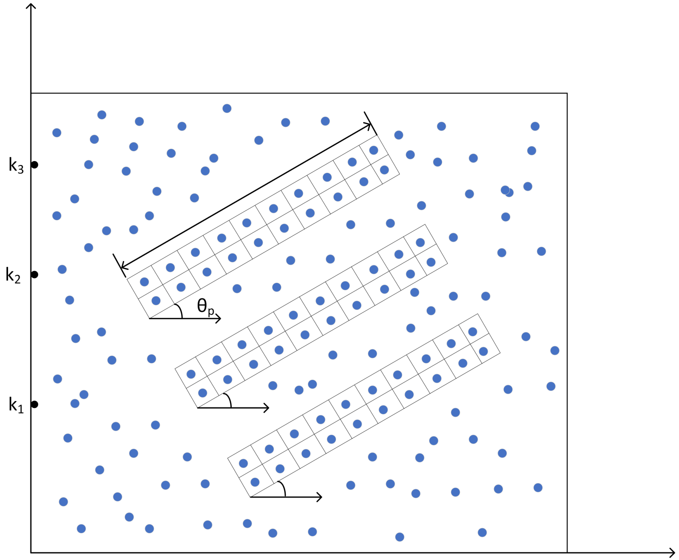
Add the dock and aisle space j

$$x^{[w]} = x_0 + l \times cos(\theta) - w \times sin(\theta) \ \ \forall \ \ l = \frac{L_1}{q/6}, w = 1, 3$$

$$y^{[w]} = y_0 + l \times sin(\theta) + w \times cos(\theta) \ \ \forall \ \ l = \frac{L_1}{q/6}, w = 1, 3$$

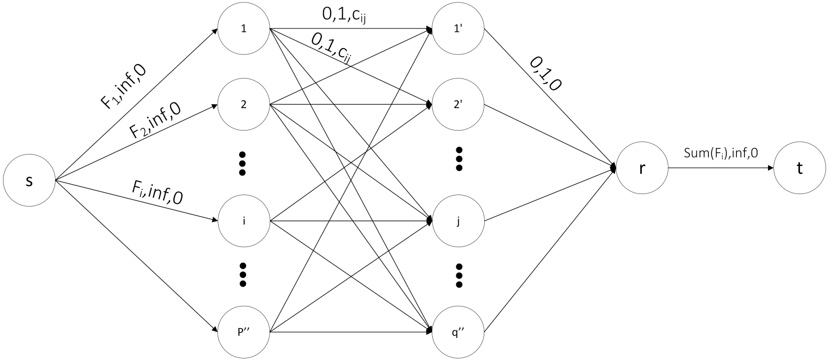
Find the shortest path from k to j using Dijkstra's algorithm.

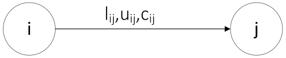
PRM



Find the optimal location for the items

This is a transportation problem but we will make it mode general





Capacity Scaling Algorithm to solve the Minimum-cost problem

$G(x, \Delta)$ Subgraph of $G(x)$ consisting of arcs whose residual capacity is at least Δ.

$S(\Delta) = \{i : e(i) \geq \Delta\}$ Nodes that act as a Source

$T(\Delta) = \{i : e(i) \leq \Delta\}$ Nodes that act as a Sink

$$e(i) = b(i) + \sum_{j:(j,i)\in A} x_{ji} - \sum_{j:(i,j)\in A} x_{ij} \quad \forall\, i \in N$$

$O(m \log U)(m + n \log n)$ Runs in polynomial-time

Capacity Scaling Algorithm to solve the Minimum-cost problem

```
1: $x \leftarrow 0$, $\pi \leftarrow 0$ and $\Delta \leftarrow 2^{logU}$
2: while $\Delta \geq 1$ do
3:     for every arc $(i, j)$ in the residual network $G(x)$ do
4:         if $r_(i, j) \geq \Delta$ and $c^{\pi}_{ij} < 0$ then
5:             send $r_{ij}$ units of flow along arc $(i, j)$,
6:             update $x$ and the imbalances $e(\cdot)$, $S(\Delta)$ and $T(\Delta)$
7:             while $S(\Delta) \neq \emptyset$ and $T(\Delta) \neq \emptyset$ do
8:                 select a node $k \in S(\Delta)$ and a node $l \in T(\Delta)$
9:                 determine shortest path distances $d(\cdot)$ from node $k$
10:                $\Delta$-residual network $G(x, \Delta)$ with respect to the reduced costs $c_{ij}$
11:                update $\pi \leftarrow \pi - d$
12:                augment $\Delta$ units of flow along the path (the shortest path $k - l$)
13:                update $x$, $S(\Delta)$, $T(\Delta)$, and $G(x, \Delta)$
14:            $\Delta \leftarrow \Delta/2$
```

Optimality conditions

Each iteration solves a shortest path problem with nonnegative arc lengths and strictly decreases the excess of some node. The iterations continue as long as a set of nodes has nonzero imbalance.

A feasible solution x^* is an optimal solution of the minumum flow problem iff:

$$c_{ij}^{\pi} \geq 0 \text{ for every arc } (i, j) \text{ in } G(x^*)$$

We can show this by using the Negative Cycle Optimality Conditions: The residual netowork $G(x^*)$ contains no negative cost directed cycles, i.e. there is no shortest path in a graph with a negative cycle; if there is, this does not hold $d(j) \leq d(i) + c_{ij}$.

$\sum_{(i,j)\in W} c_{ij}^{\pi} \geq 0$ for every directed cycle W in $G(x^*)$.

$\sum_{(i,j)\in W} c_{ij}^{\pi} = \sum_{(i,j)\in W} c_{ij} \geq 0$ so $G(x^*)$ does not contain a negative cycle.

Results and conclusions

Results

```
w = [0.5]*48
d = [3.062, 3.208, 3.373, 3.427, 3.481, 3.54,  3.965, 3.981,
     4.018, 4.268, 4.397, 4.497, 4.845, 5.039, 5.215, 5.84,
     5.898, 6.31,  6.584, 6.735, 6.901, 6.955, 7.75,  7.818]*2
F = [1,2,2,4,5,1]
Solution = {'x_3_p7': 1.0, 'x_4_p5': 1.0, 'x_4_p6': 1.0,
'x_4_p7': 1.0, 'x_4_p8': 1.0, 'x_5_p2': 1.0, 'x_5_p3': 1.0,
'x_5_p4': 1.0, 'x_5_p7': 1.0, 'x_5_p8': 1.0, 'x_6_p7': 1.0}
Minimum cost = 37.801
[Repeat the complete algorithm using a different configuration]
```

The problem could be closer to reality but may become more complex to solve
The simplex algorithm is commonly used by researchers because it is a one time procedure (in most cases)
Stochastic simulation can be another effective approach
New algorithms like Li Chen et al can run in almost linear time
Testing different configurations is necessary to identify the most efficient layout design.

Questions?