Artificial Intelligence: Project 2

Path Finding For Robots

Agathe Benichou, Wassim Gharbi, Greg Shindel, Nick Turney, Thanh Vu

Problem statement

A cutting edge ramen restaurant has opened up in Easton

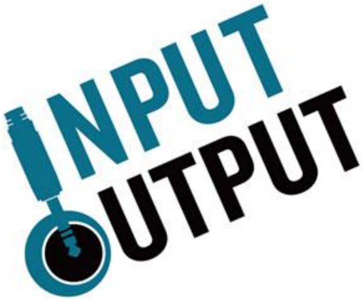
- Utilizing vending kiosk and robot servers to run the restaurant

The problem is a case study of an automated restaurant where

- Orders are placed on a connected electronic platform
- Customers are served by autonomous robots communicating with a centralized system

Parameter Highlights

- From the restaurant
  - **Map** of the restaurant
  - Customer **flow rate**
- From on-field robots
  - **Obstacle** detection
  - Current State
- From customers
  - Order
  - **Seat location**
- **Notification** when order is ready



- **Efficient, obstacle-free paths** from
  - Waiting station to order station
  - Order station to delivery table(s)
  - Delivery table(s) to order station or waiting station
- Procedures for handling **dynamic obstacles**

Important Assumptions

- Robots are able to **exchange** tasks
- Restaurant layout is **static** and provided during setup
- Real-time **Synchronization** between control system and robot
  - Control system knows what the robot knows at a time step
- Static and dynamic obstacles are at an order of magnitude of the size of a robot

Important Constraints

- **All** paths between tables are **one robot wide**, can only fit one robot
- **All** collisions must be avoided
- Each robot can carry **4 units** of a meal where **ramen** serving is **2 units**, each drink serving is **1 unit**, and each **dessert** serving is **1 unit**
- Orders must be delivered to a location **directly adjacent** to the specified seat

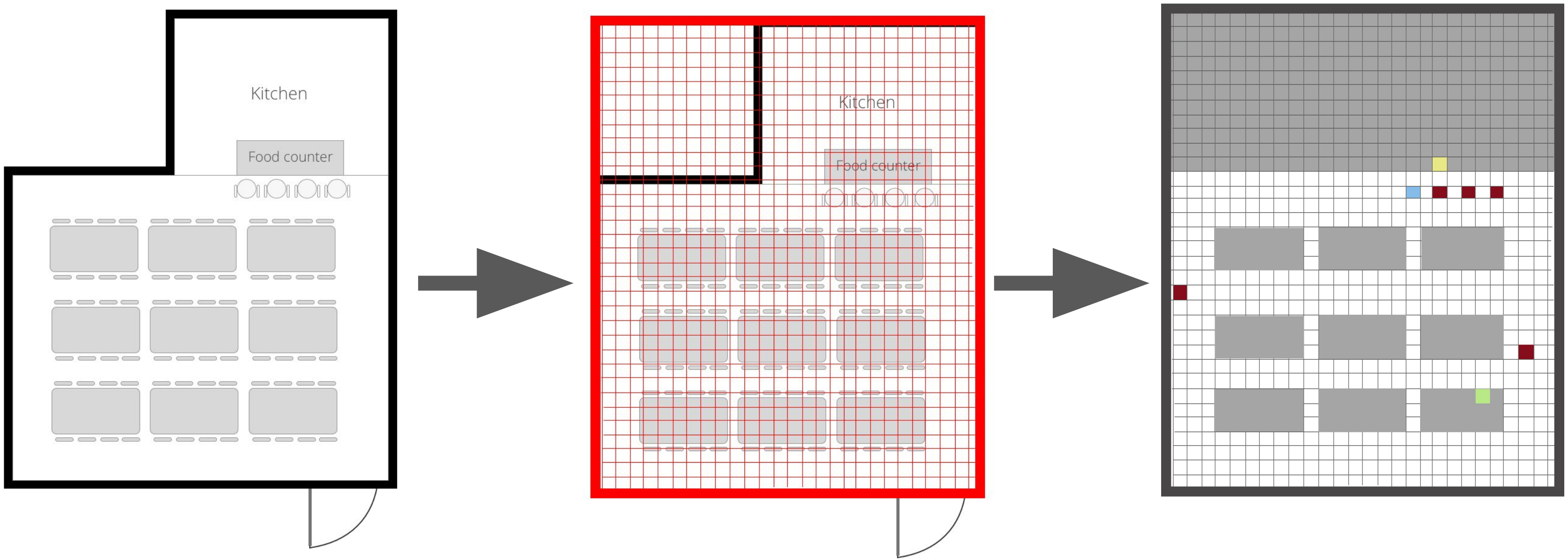
Evaluation criteria

- Are the solutions optimal? Suboptimal?
- How are collisions and deadlocks handled? Dynamic obstacles?
- Time
  - Runtime to generate solutions?
  - Time to execute solutions?
    - Makespan - length of longest delivery route
    - Flowtime - sum of all routes
- Optimal number of robots?
- Different use cases? Extremes ones?
- Strengths?
- Limitations?

Solution Overview

Solution: High Level

- Preprocessing steps
    - Divide restaurant into a grid of elementary, equally sized squares
        - Permanent Obstacle
        - Temporary Obstacle (clients and other robots)
        - Target (client’s position)
        - Source (food counter)
    - Obtain orders from customers



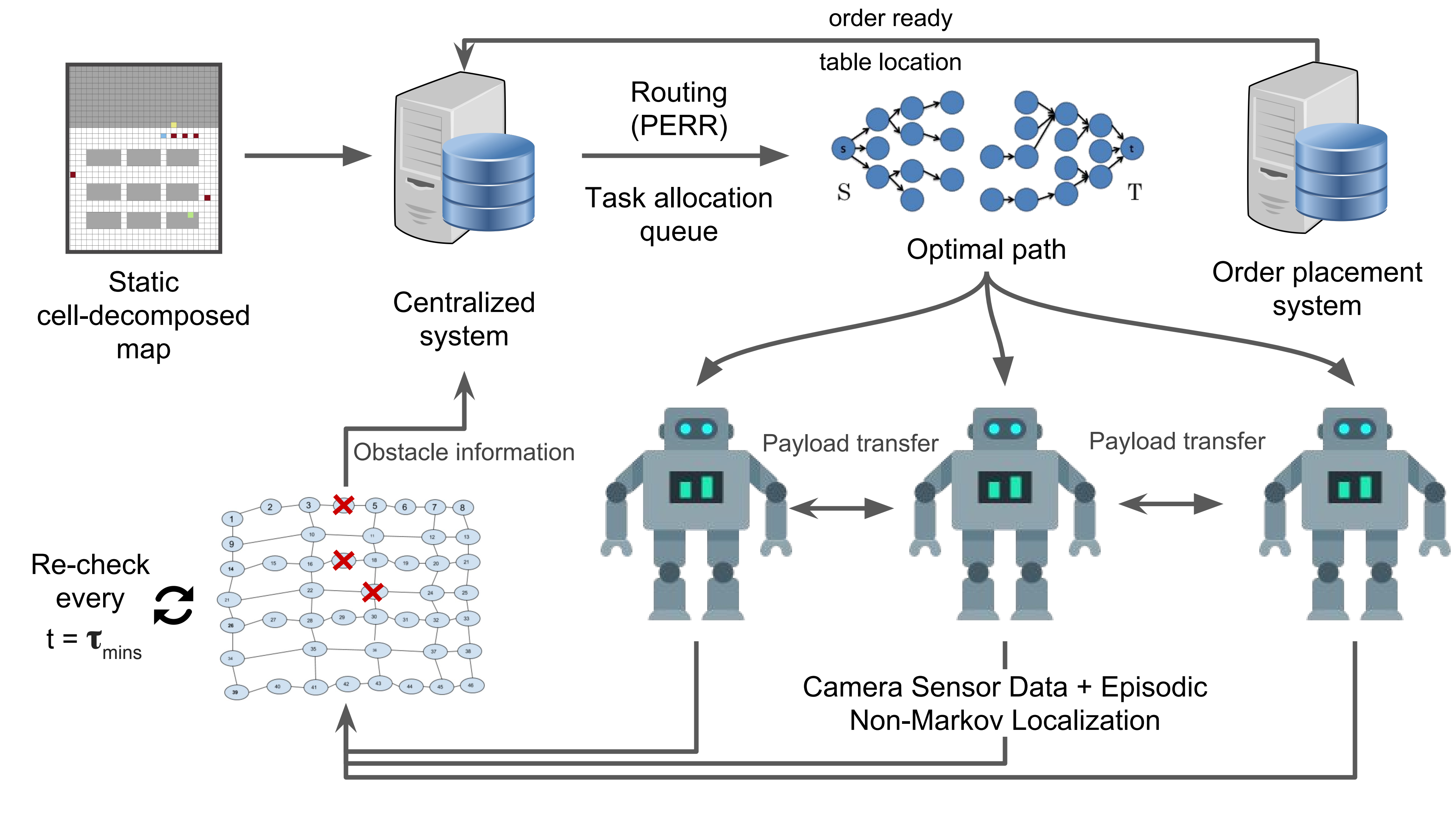
Blueprint

Divide into squares

Map stationary obstacles

Solution - high level

- Main algorithm
  - Graph search to find a connected component between start and end node
  - Flow-based ILP (Integer Linear Programming) to solve a PERR (Path-Exchange Robot Routing Problem)
  - Episodic Non-Markov Localization for obstacle detection at the individual robot level


order ready
table location
Routing
(PERR)
Task allocation
queue
S
T
Optimal path
Static
cell-decomposed
map
Centralized
system
Order placement
system
Payload transfer
Payload transfer
Obstacle information
Re-check
every
$t = \tau_{mins}$
Camera Sensor Data + Episodic
Non-Markov Localization

Preprocessing Steps Overview
Map decomposition and knowledge discovery

Step 1 - Cell Decomposition

- Decompose the blueprint of the restaurant into a data structure

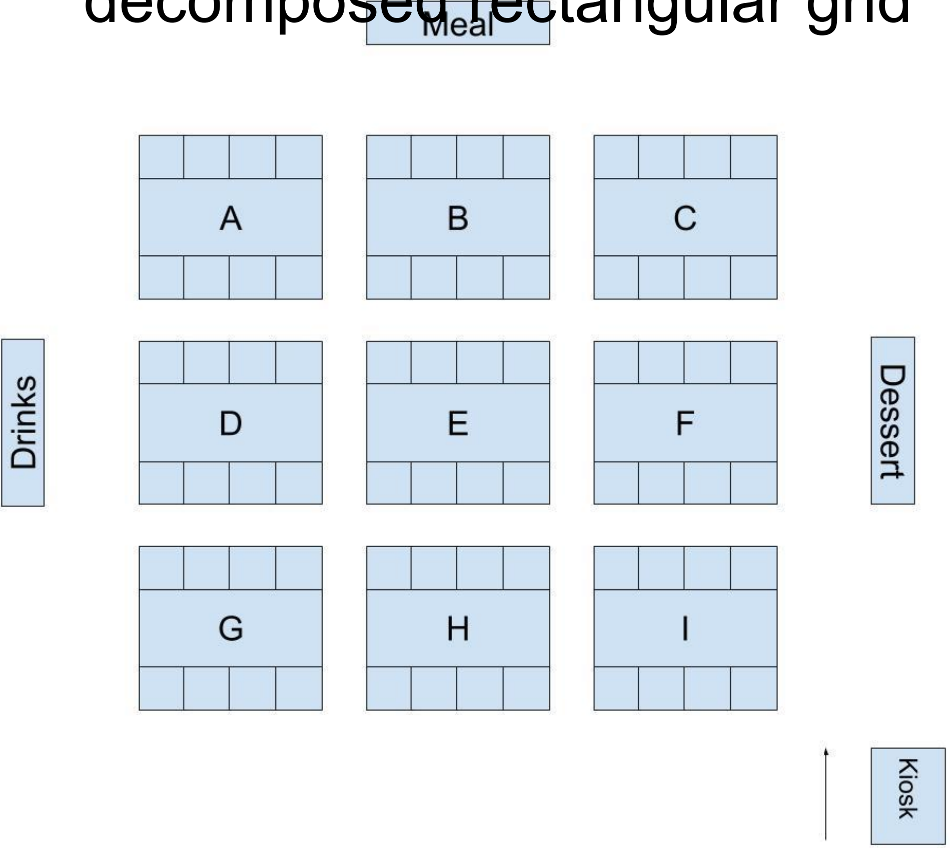
Step 2 - Survey and Simulation

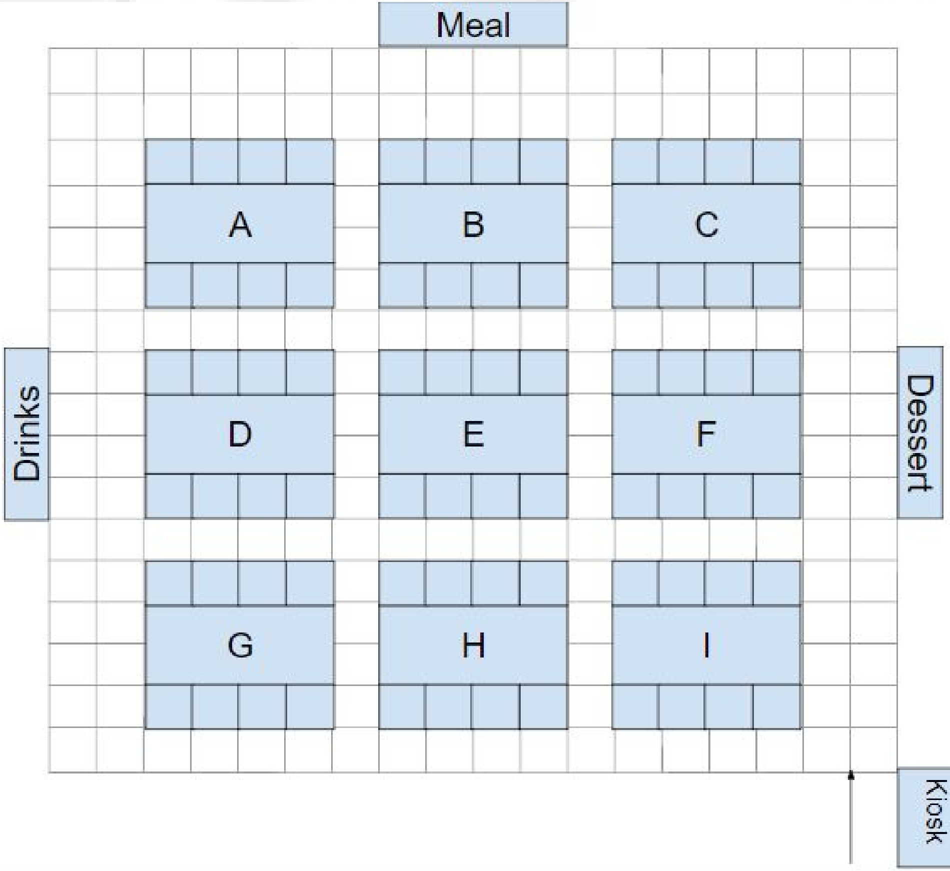
- Collect data from current Easton restaurants about customer flow rate
- Utilize information to create simulation to determine number of robots

Preprocessing Step 1
Approximate Cell Decomposition

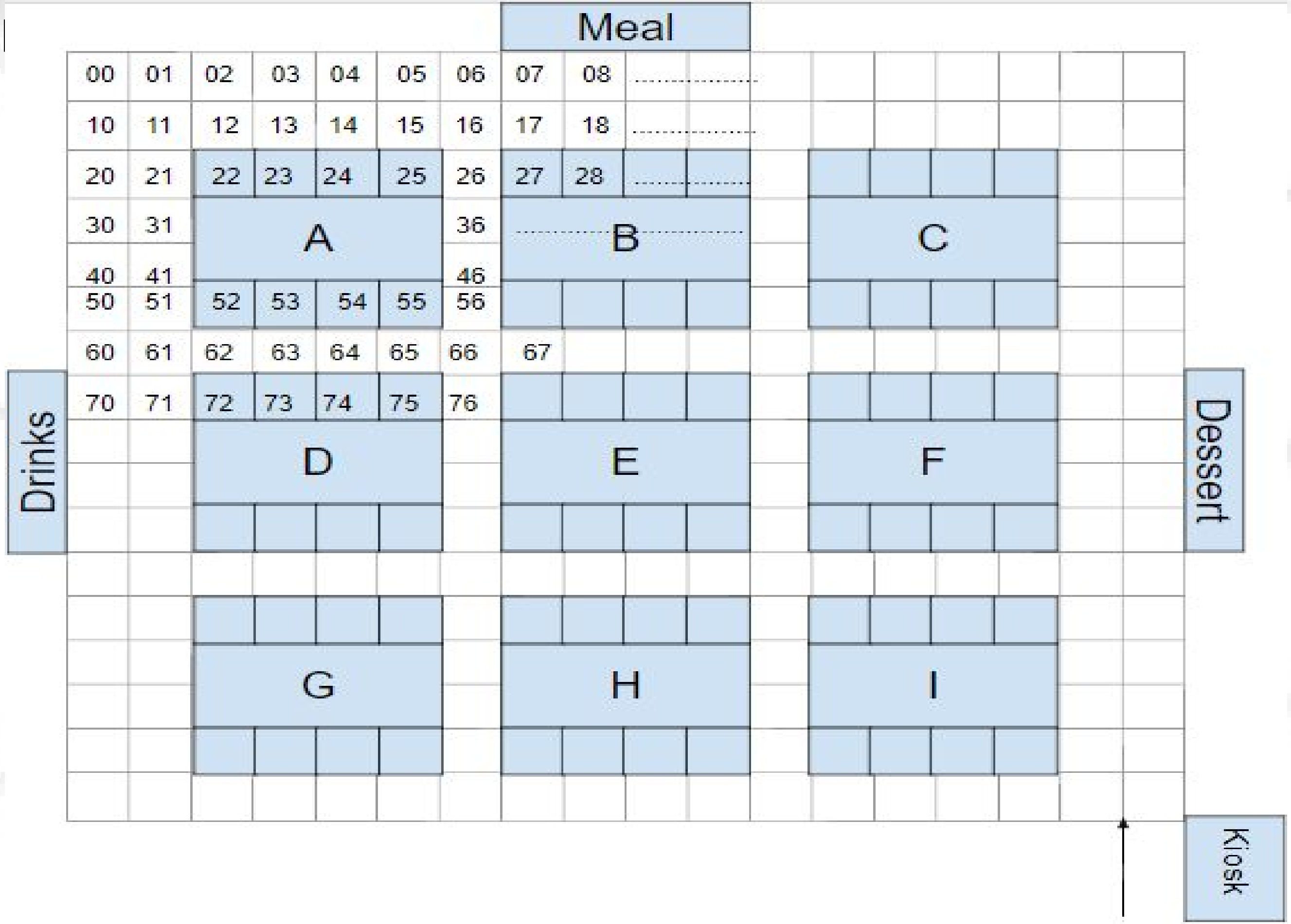
- Robots need an accurate model of their environment
- Use approximate cell decomposition to transform workspace representation
  - Configuration space -› grid with cells -› connectivity graph
- Approximate means that all cells share a predefined shape (square)
  - Any kind of obstacle will take up an entire cell

Preprocessing Step 1 continued.. Blue print of restaurant to decomposed rectangular grid

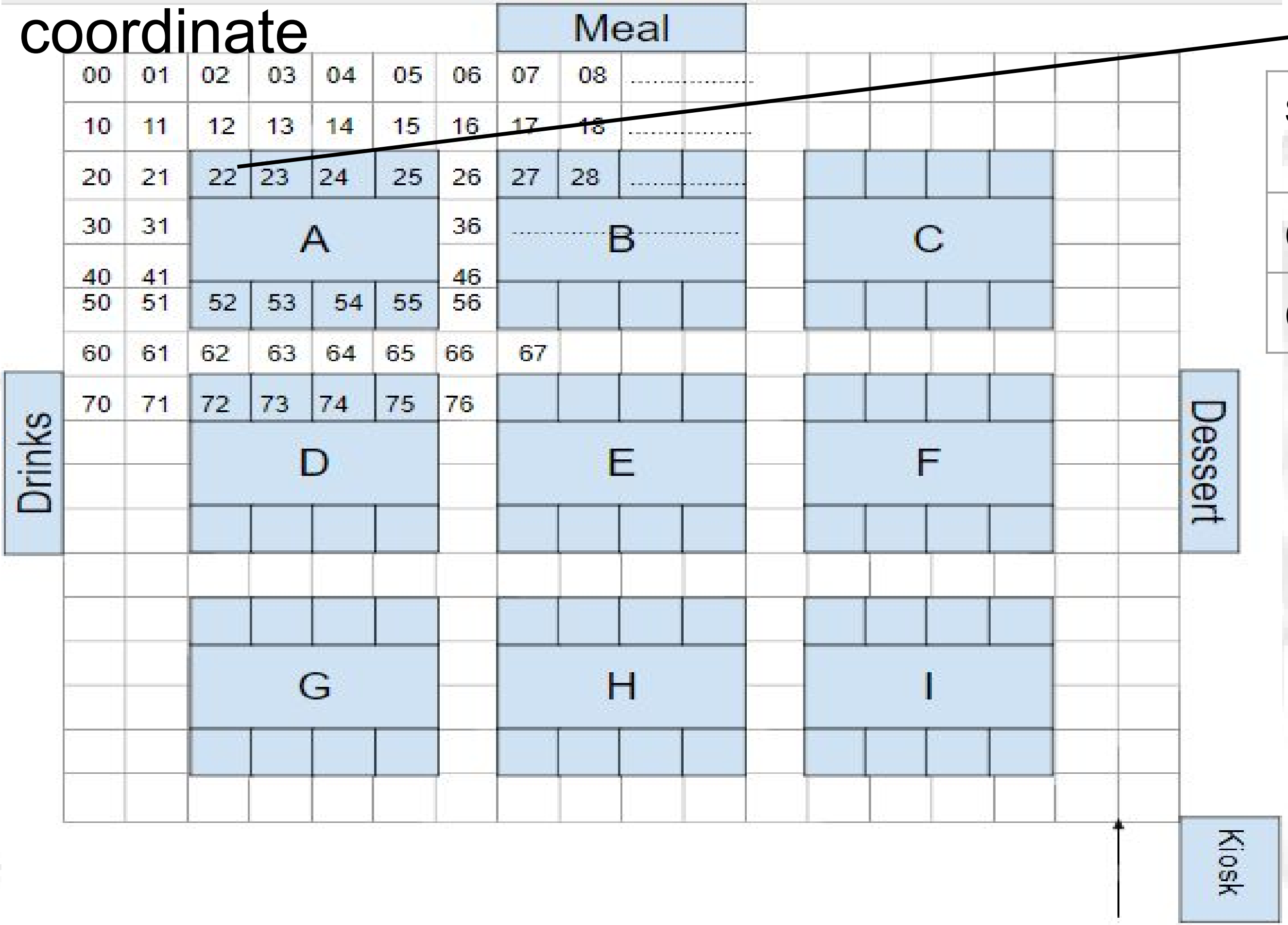




Preprocessing Step 1 continued...   Assign each cell a coordi

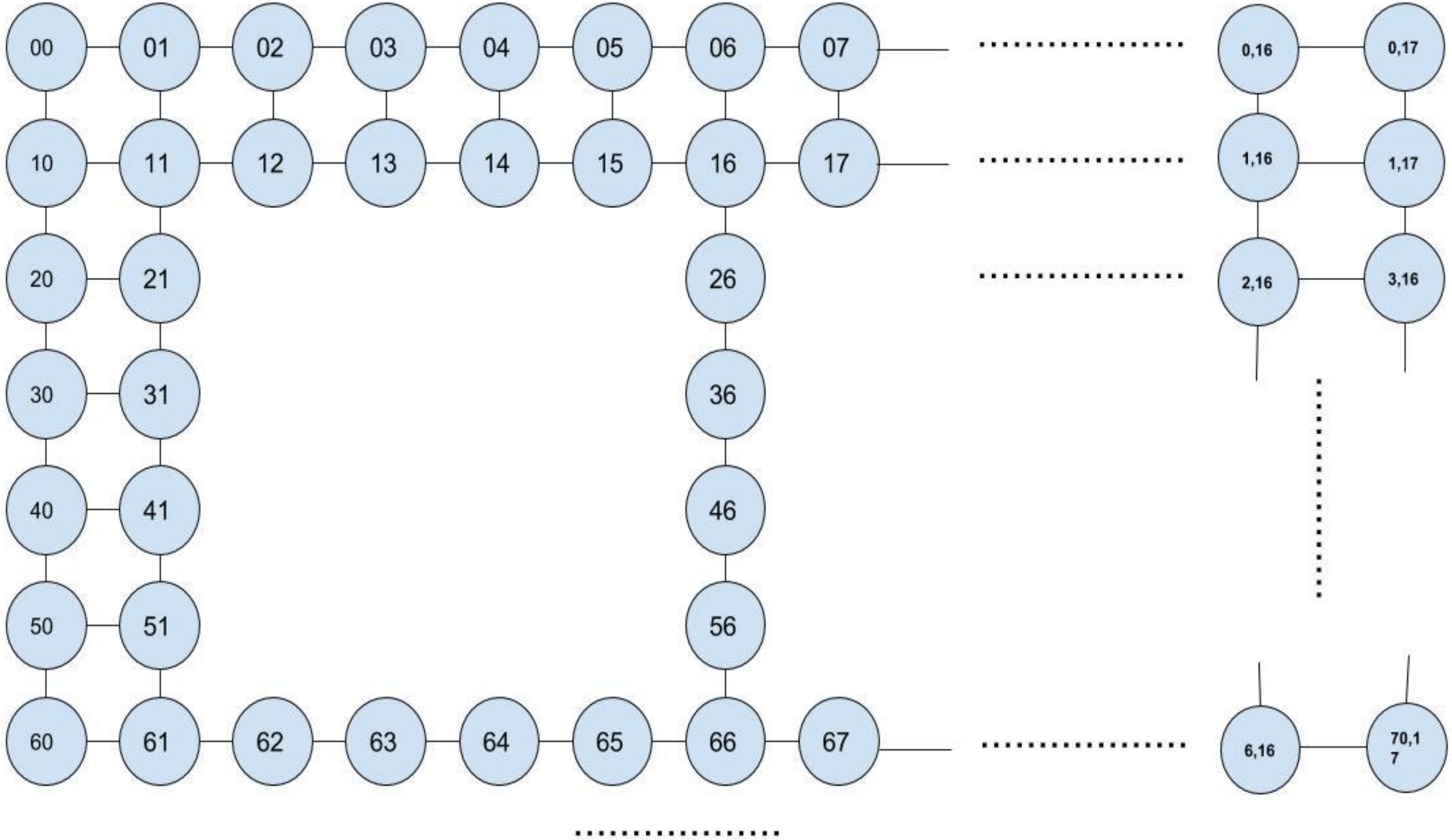


Preprocessing Step 1 continued... Assign each cell a coordinate



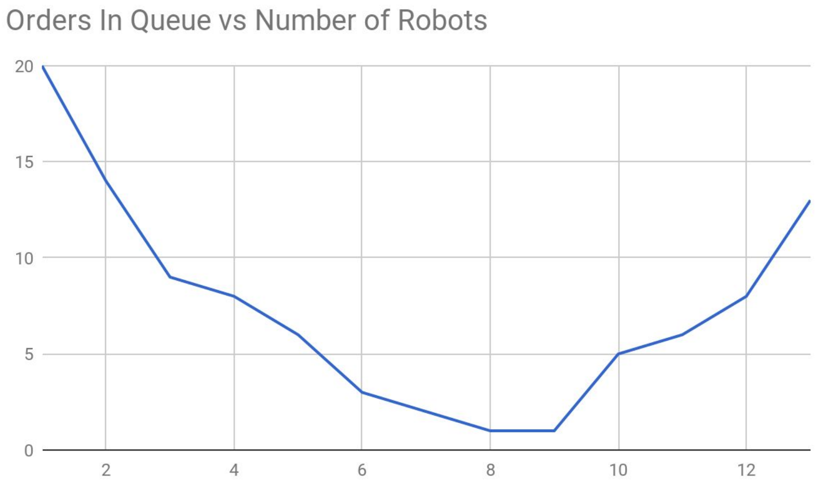
Seat	Dest Node
(2,2)	(1,2)
(2,3)	(1,3)

Preprocessing Step 1 continued... Translate the cells into a connectivity graph

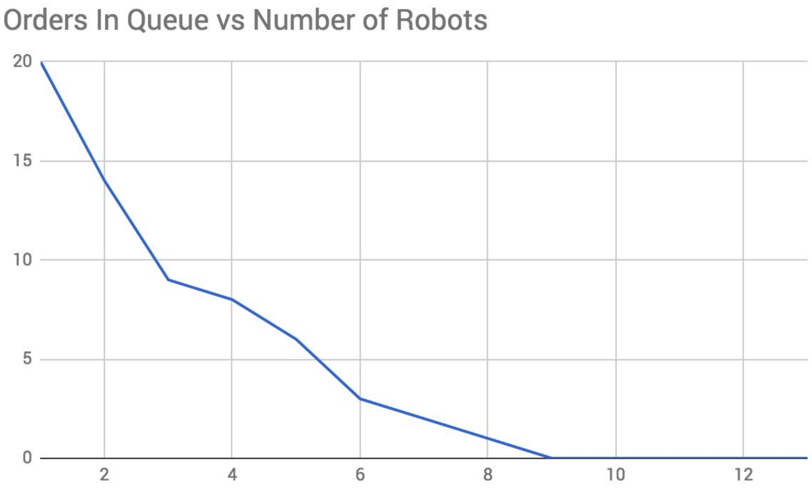


Preprocessing step 2
Determining number of robots needed

- Survey nearby restaurants for
    - Average customer flow
    - Average order preparation time
- Simulate the algorithm and plot robots used vs average ready orders in queue at each time interval



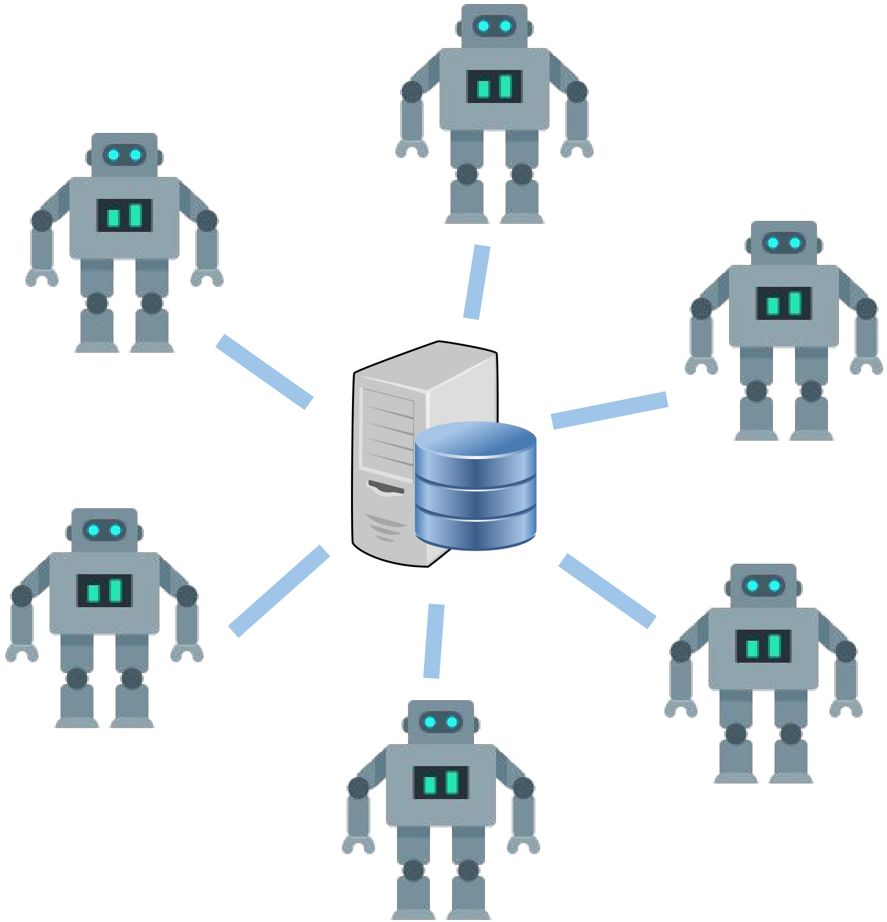
CBS For MAPF (Conflict-Based Search For Optimal Multi-Agent Path Finding)



Flow-based ILP (Integer Linear Programming) for PERR (Path-Exchange Robot Routing Problem)

General view
Main algorithms

- Centralized, decoupled system with discrete timesteps
    - Task Assignment
    - Pathfinding
- Each robot
    - Task execution
    - Obstacle detection

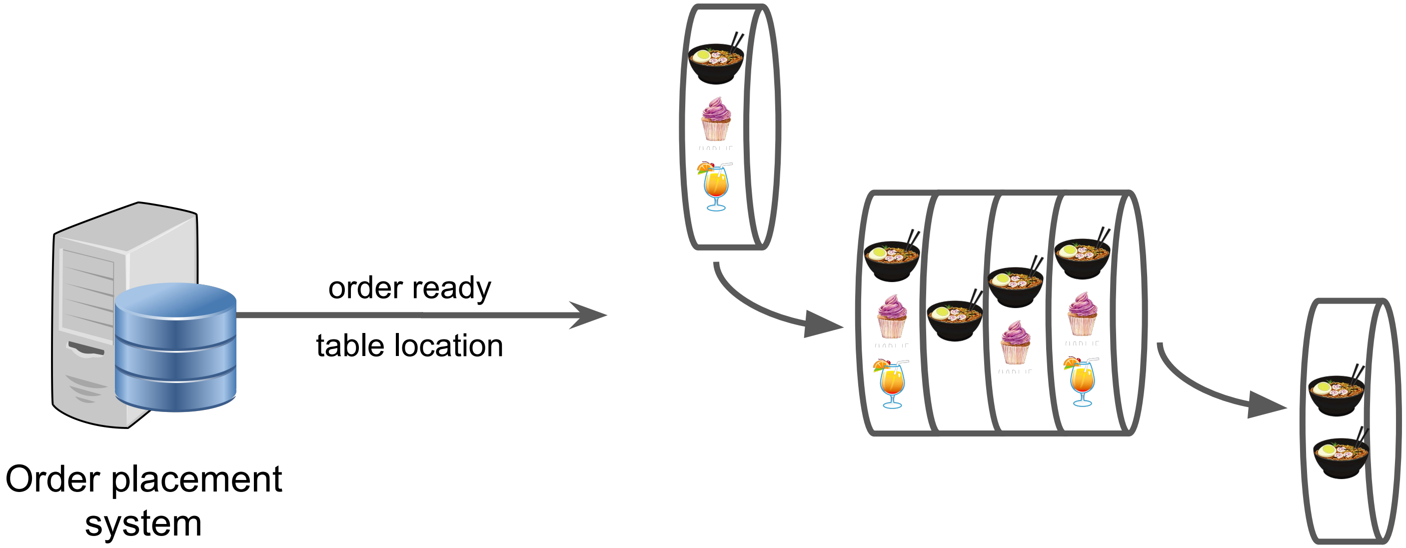


General view
Main algorithms

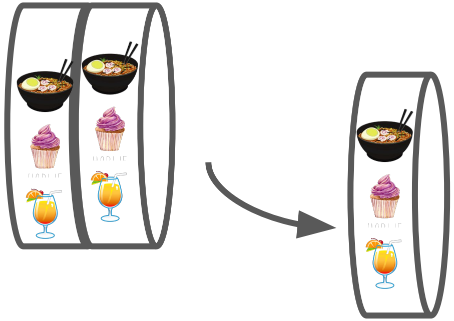
- Centralized, decoupled system with discrete timesteps
    - Task Assignment
    - Pathfinding
- Each robot
    - Task execution
    - Obstacle detection

Challenge	Solution
Collision Avoidance	- Package-exchange robot-routing problem - Optimal, reduction-based algorithm
Continuous Task Assignment	- Rerunning pathfinding algorithm upon task completion
Obstacle Handling	- Obstacle classification - System-wide updates

Main Algorithms: Subproblem 1
Assigning Tasks



Main Algorithms: Subproblem 1
Assigning Tasks

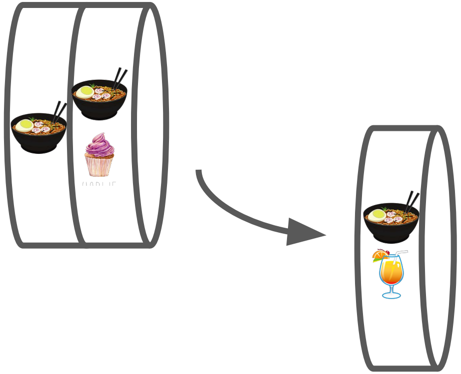


Complete orders
Use simple queue structure

Order placement system

order ready

table location

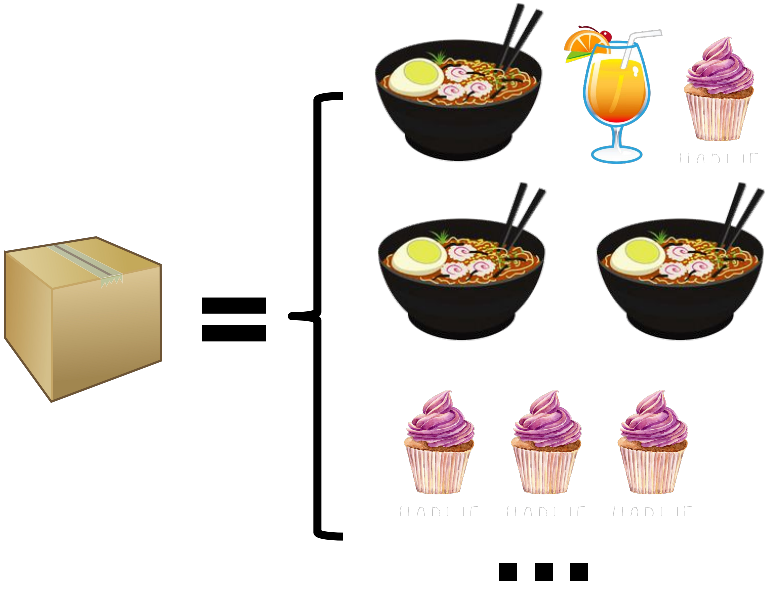


Partial orders
Group successive orders, first come first served based on robot capacity and meal size

Main Algorithms: Subproblem 2

Pathfinding: Formalization

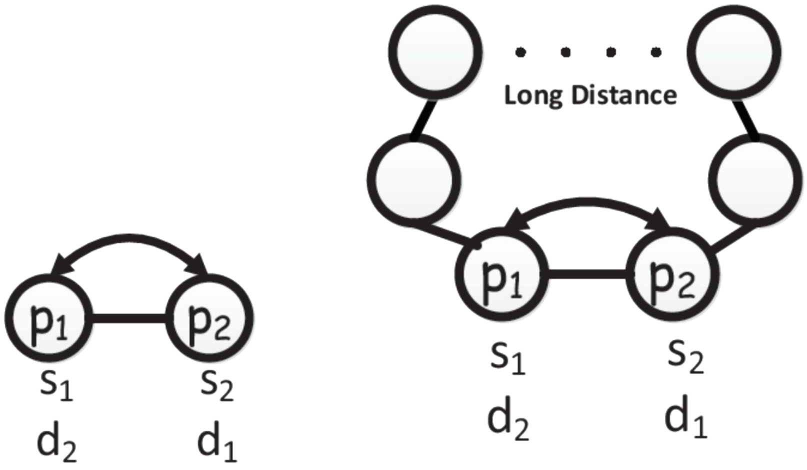
- Package-exchange robot-routing problem (PERR)
  - Agents deliver packages to destinations
  - Packages can be transferred between two agents in adjacent vertices



Main Algorithms: Subproblem 2

Pathfinding: Motivation

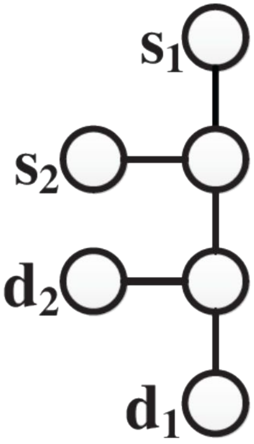
- Orders are not anonymous. But robots can be.
- All PERR instances are proven to be solvable

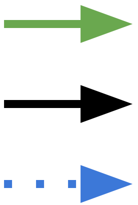


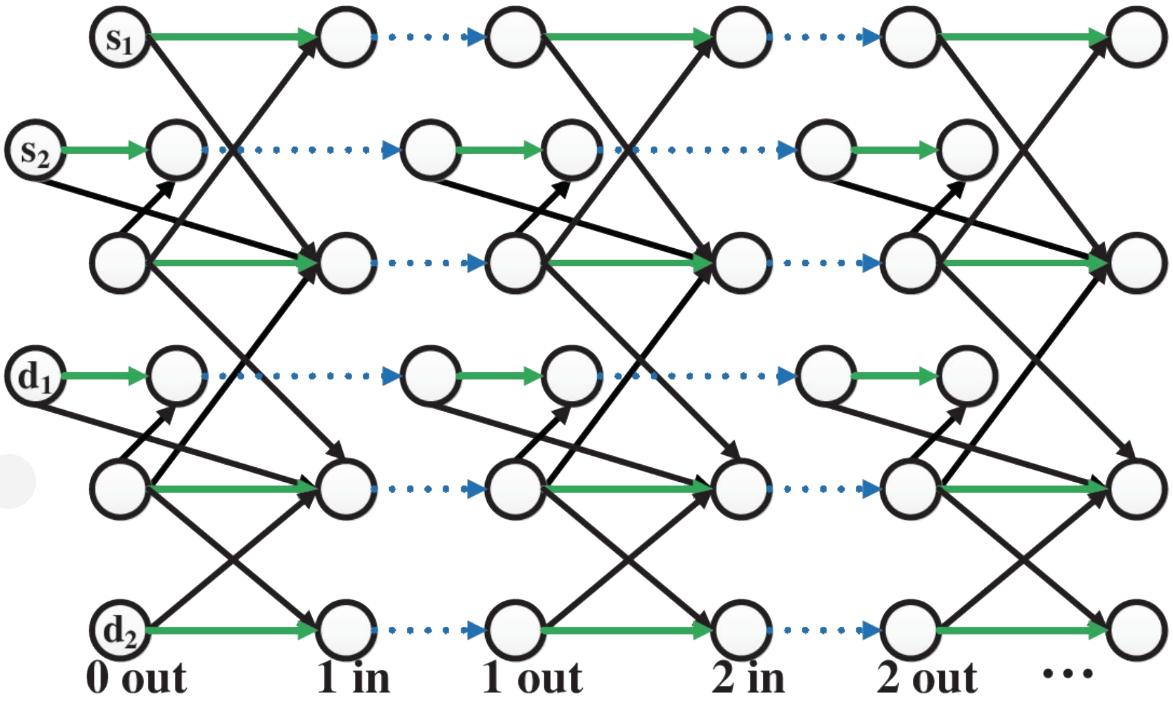
(a) (b)

Main Algorithms: Subproblem 2

Pathfinding: Network-flow Reduction







Main Algorithms: Subproblem 3

Obstacle Handling

Inspired by Episodic Non-Markov Localization (Biswas, Veloso 2014)

1) Dynamic Obstacle (e.g. a person, a pet)

- For each timestep over a **small** period of time (n), **re-check** the obstacle:
  - If the path is **clear**:
    - **Recalculate** optimal routes accounting for the time delay
- If after n number of timesteps the obstacle **remains**
  - Handle the obstacle as if it was **Static**

2) Static Obstacle (e.g. backpack, wheelchair)

- **Update** the shared map for other agents
  - Remove edges of decomposition which are blocked
- **Recalculate** optimal routes
- Maintain a **Timer** (m) mapped to the obstacle
  - On expiration, assume the obstacle has cleared and re-add edges

Future work

- Learning from real-life data and re-running the simulation to optimize for number of robots needed (either ask for more or less robots)
- Evaluate for other generic restaurant plans where new constraints might arise

Questions?

References

[1] H. Ma, C. Tovey, G. Sharon, T. K. S. Kumar, and S. Koenig. *Multi-agent path finding with payload transfers and the package-exchange robot-routing problem.* In AAAI Conference on Artificial Intelligence, pages 3166–3173, 2016.

[2] H. Ma, S. Koenig, N. Ayanian, L. Cohen, W. Hoenig, S. Kumar, T. Uras, H. Xu, C. Tovey and G. Sharon. *Overview: Generalizations of Multi-Agent Path Finding to Real-World Scenarios.* In Proceedings of IJCAI-16 Workshop on Multi-Agent Path Finding, 2016.