Decomposition and Learning Congestion for Multi-Agent Path Finding

Jake Gonzales, Joey Sullivan, Samuel A. Burden, Lillian J. Ratliff, Daniel Calderone
Department of Electrical and Computer Engineering, University of Washington

Motivation

- **Problem:** Multi-agent path planning for large-scale autonomous mobility where hundreds to thousands of robots are simultaneously completing tasks.

- **Challenges:**
- Problem scales exponentially in the number of agents and MAPF is NP-Hard.
- inherent sources of uncertainty such as item arrival estimations and kinodynamics modeling for robots.

- **Application:** motivated by modeling interactions of large amounts of robots planning paths in warehouses settings such as sorting centers at Amazon.

Sub-Region Decomposition

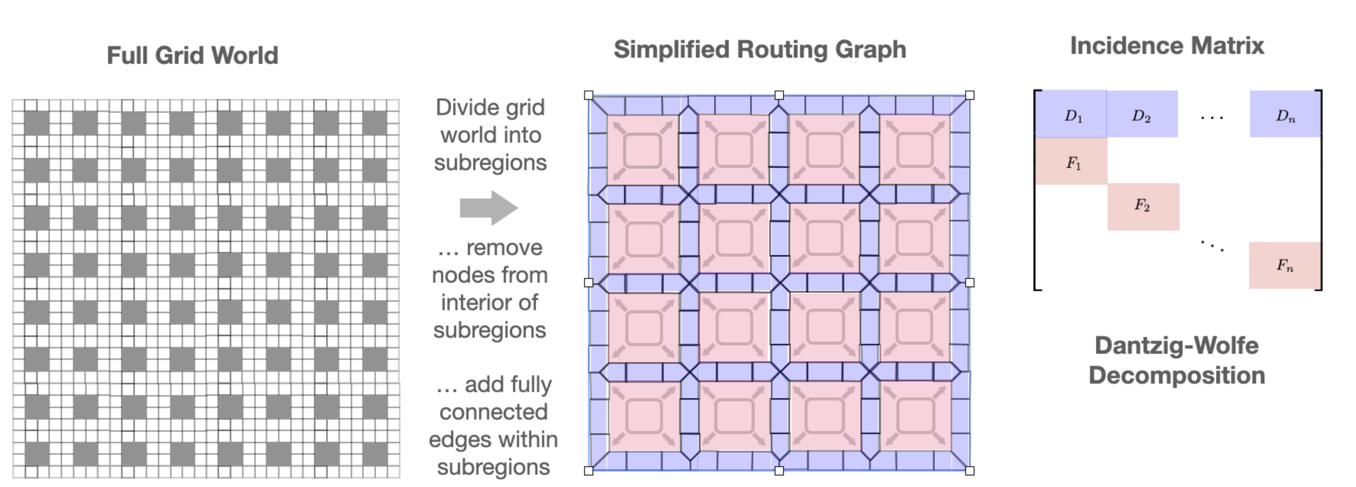
- The grid world is divided into spatial subregions by performing a Dantzig-Wolfe decomposition on the incidence matrix graph of the whole grid world.

We define a graph $G(\mathcal{V},\mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E}$. Each edge $e \in \mathcal{E}$ is associated with a flow variable $x_e \in \mathbb{R}_+$ that denotes how much population mass is on that edge and a latency function $\bar{\ell}_e(\bar{x}_e)$ that gives the travel time for taking a particular edge.

$$\left[E_{\mathrm{o}}\right]_{je} = \begin{cases} 1 & \text{if edge } e \text{ starts at node } j \\ 0 & \text{otherwise} \end{cases}$$

$$\left[E_{\mathrm{i}}\right]_{je} = \begin{cases} 1 & \text{if edge } e \text{ ends at state } j \\ 0 & \text{otherwise} \end{cases}$$

$$E = \begin{bmatrix} D_1 & D_2 & \dots & D_{|\mathcal{K}|} \\ F_1 & 0 & \dots & 0 \\ 0 & F_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & F_{|\mathcal{K}|} \end{bmatrix}$$



Computing Trajectory Rollouts using Subregion CBS

- The current state-of-the-art for multi-agent path finding (MAPF) algorithms is called conflict-based search (CBS) which is guaranteed to find an optimal solution when one exists [2].

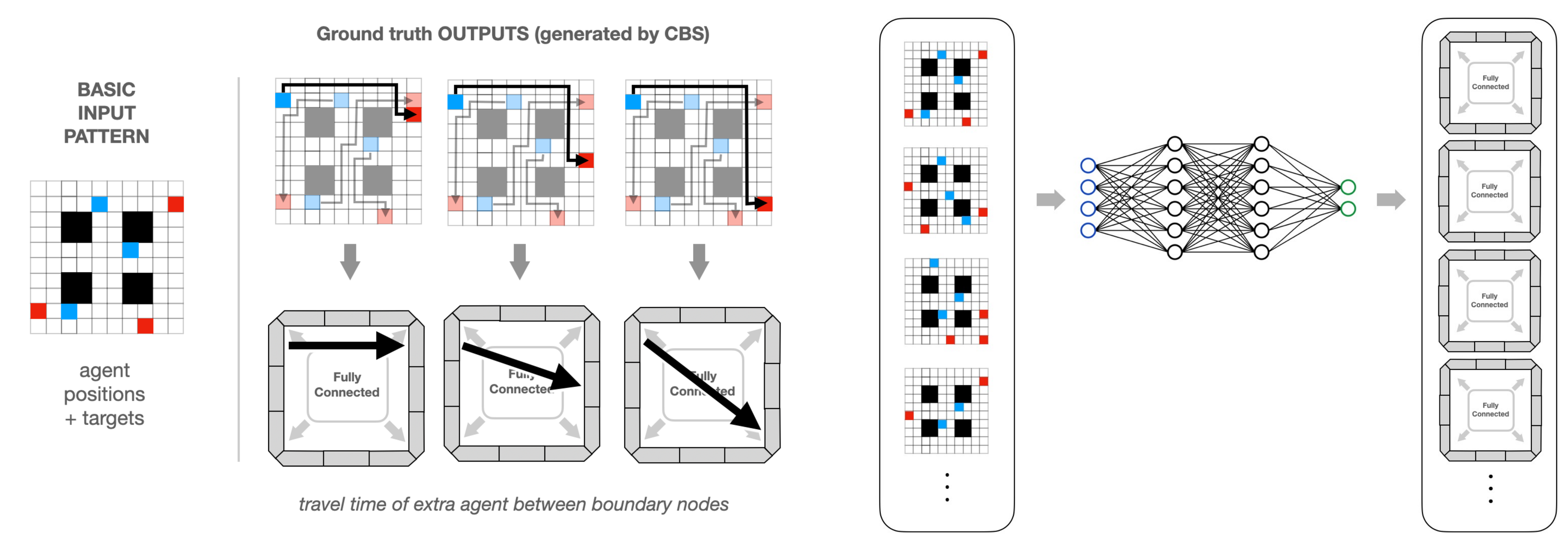
- For agents that each pass through a given sequence of subregions, we develop an algorithm to solve CBS within each subregion as agents pass in and out.

- This method turns rough trajectory estimates into viable, realistic paths that are locally optimal in space and time.

Learning Congestion

- We use a deep learning approach to predict congestion present in agent interactions from the CBS path planning in each sub-region to predict travel-times on edges in the graph.

- We develop a Graph Convolutional Network (GCN) for extracting spatial features on the graph to learn travel-times on each edge experienced by agents in the CBS trajectories.



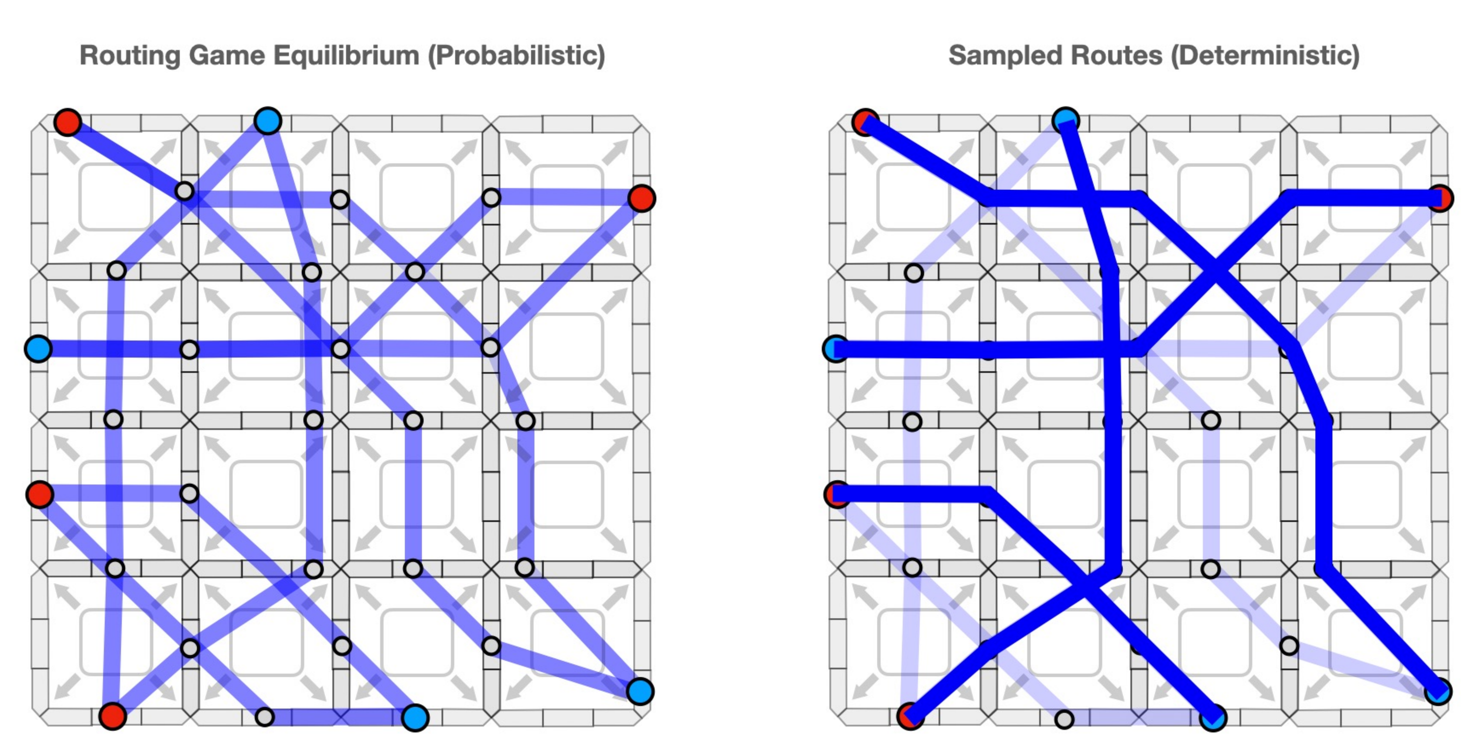
Routing Game Formulation

- The routing game is formulated as a convex optimization problem where we assume the edge latency functions are increasing. In the presence of congestion, we formulate the problem by introducing a routing game potential function

$$\bar{F}(\bar{x}) = \sum_{e\in\bar{\mathcal{E}}} \int_0^{\sum \bar{x}_e} \bar{\ell}_e(s)\, ds = \sum_{e\in\bar{\mathcal{E}}} \int_0^{\sum_{(o,d)} \bar{x}_{ode}} \bar{\ell}_e(s)\, ds$$

Giving the optimization formulation

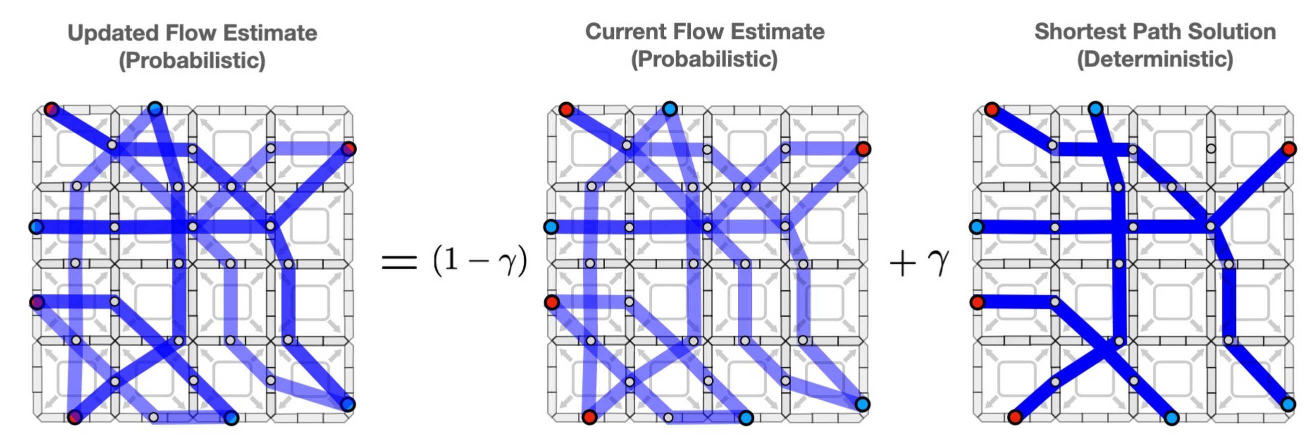
$$\begin{aligned} \min_{x,x_{od}} \quad & \bar{F}(\bar{x}) \\ \text{s.t.} \quad & \bar{E}\bar{x}_{od} = \bar{S}_{od},\ \bar{x}_{od} \geq 0 \ \ \forall o,d \\ & \bar{x} = \sum_{od} \bar{x}_{od} \end{aligned}$$



Our Approach

Initialization:
1. Represent the grid world abstraction as a graph.
2. Spatially decompose the graph into sub-regions using a Dantzig-Wolfe decomposition.
3. Train a GCN based on data from CBS rollouts using certain agent configurations.

Iteration:
1. Sample paths for agents from the current equilibrium estimate.
2. Rollout paths using CBS
3. Estimate edge latencies using the pre-trained GCN from the CBS rollouts
4. Compute the new shortest paths given current edge latencies
5. Update the equilibrium estimate using Franke-Wolfe style update.
6. Repeat steps 1-5.



We compute an approximate latency function from the graph convolutional network as $\bar{L} : x \in \mathbb{R}^{|\mathcal{E}|} \mapsto \mathbb{R}^{|\bar{\mathcal{E}}|}$ and x defined by the rollouts from the robot trajectories we implement the FW style update as

$$\begin{aligned} \min_{\xi,\xi_{od}} \quad & \bar{L}\left(x^{(k)}\right)^{\top} \xi^{(k)} \\ \text{s.t.} \quad & E_{od}\xi^{(k)}_{od} = S_{od},\ \xi^{(k)}_{od} \geq 0 \ \ \forall o,d \\ & \xi^{(k)} = \sum_{od} \xi^{(k)}_{od} \end{aligned}$$

Algorithm 1 Franke-Wolfe with Shortest Paths

1: **Input** $x^{(0)} \in \mathcal{X}$
2: **Output** $x^{(k)} \in \mathcal{X}$
3: **Given** Edge travel-times $\bar{\ell}_e(\bar{x}_e)$
4: **for** $k = 1, \cdots, T$ **do**
5: Compute shortest paths $P_{o,d}$ and the associated costs using Dijkstra's algorithm
6: Update $x^{(k+1)}_{od} \leftarrow (1-\gamma)x^{(k)}_{od} + \gamma\xi^{(k)}_{od}$ for $\gamma = \frac{2}{k+2}$
7: **end for**

Discussion and Future Work

- **Discussion:** Our approach combines theoretical techniques from algebraic graph theory and convex optimization formulations of routing games with popular multi-agent path finding (MAPF) algorithms for large-scale planning problems.

- In future work we plan to combine our path planning approach with linear task assignment algorithms such as the Hungarian (Kuhn-Munkres) algorithm [1].

[1] Daniel Calderone, Kelly Ho, Lillian Ratliff, Bipartite Matching and Routing with Congestion Costs: A convex approach to robot task assignment and the multi-agent pathfinding problem. LCSS/CDC, 2024, submitted.
[2] Sharon, et al, Conflict-based Search for optimal multi-agent pathfinding, Artificial Intelligence, vol. 219, 2015.

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY of WASHINGTON

SPONSOR: UW + Amazon Science Hub