Reinforcement Learning for Zone Based Multiagent Pathfinding Under Uncertainty

Jiajing Ling, Tarun Gupta, Akshat Kumar
{jjling.2018，akshatkumar}@smu.edu.sg, tarun.gupta@cs.ox.ac.uk
SMU Singapore Management University, School of Information Systems

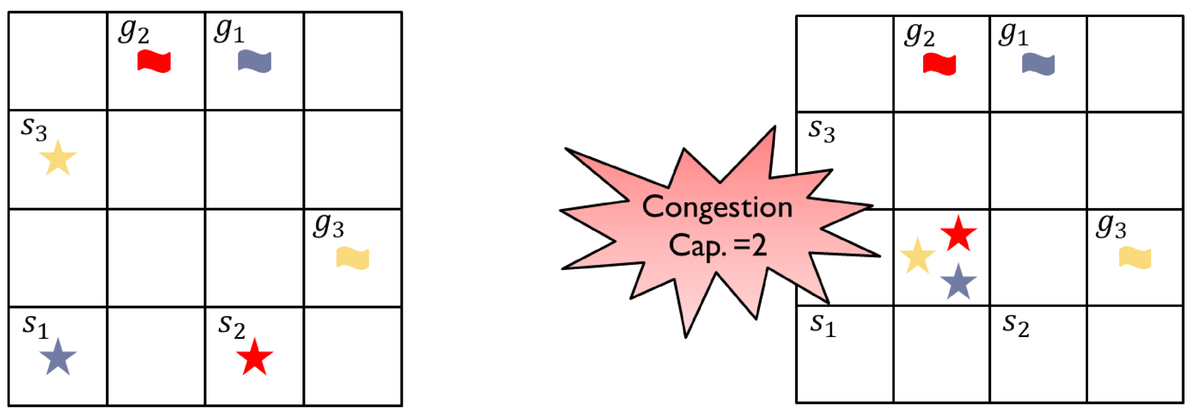
Motivation

How to navigate autonomous vehicles in a partially observable environment with uncertainty?

Our Contributions

- We formulated a zone based path finding problem (ZBPF)
  - Under uncertainty
  - Partial Observability
- We presented a novel formulation of policy optimization
  - Based on difference-of-convex functions (DC) programming
- We developed a simulator for ZBPF using Unity3D game engine

Problem Formulation

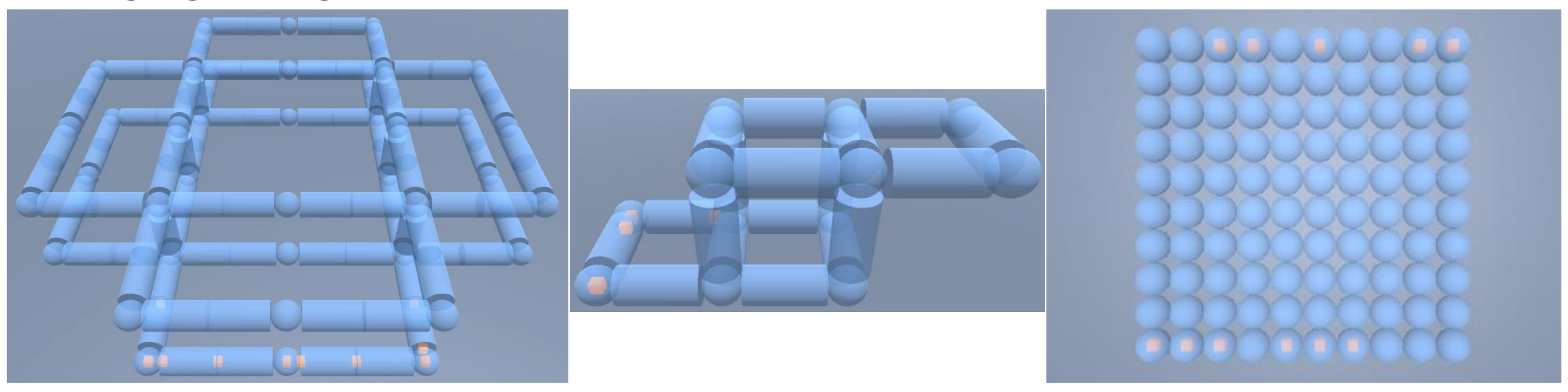


- A graph $G = (V, E)$
- Each zone has a capacity
- A set of agents with sources and destinations
- Crossing two zones requires minimum and maximum time

Objective: Minimize travel time and Congestion

Our Simulator

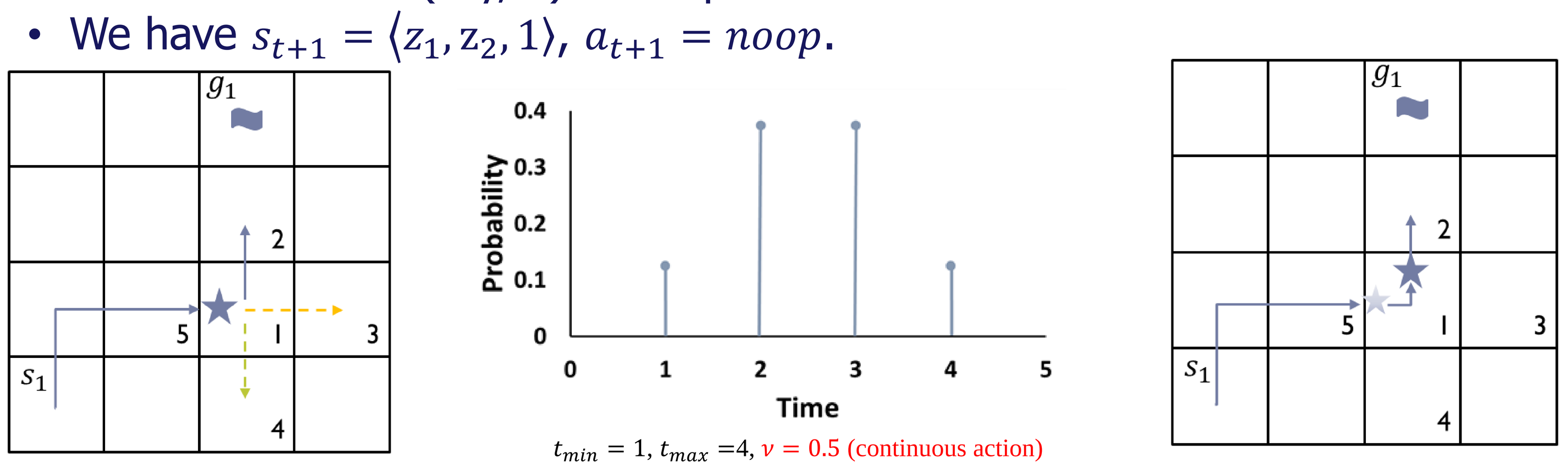
- Simulator: Unity 3D game engine
- The spheres are the zones, and cubes are the agents.
- Highlighted zone: there is congestion
- Highlighted agent: reached destination



TwoFloor　　SmallCube　　10x10 Grid Map

Agent's Decision Model

- **State**
  - State is a tuple of three components: $\langle current\ zone, next\ zone, remaining\ time\rangle$
- **Hybrid Action (discrete and continuous)**
  - Discrete action: which zone to go
  - Continuous action: with what speed
- **An example**
  - An agent is newly arrived in zone 1 at time t.
  - We have $s_t = \langle z_1, \Phi, \Phi\rangle$, $a_t = \langle z_2, 0.5\rangle$.
  - The realized time (say, 2) is sampled from a binomial distribution.
  - We have $s_{t+1} = \langle z_1, z_2, 1\rangle$, $a_{t+1} = noop$.



Newly Arrived　　Travel Time Sampling　　In-transit

- **Transition Function (exponential family)**
  - $p(s'^i | s^i, a^i, v^i) = f(s^i, a^i, s'^i)\exp\{v^i\phi(s^i, s'^i) - \mathcal{A}(v^i)\}$
  - Uncertainty movement can be modeled
  - Applicable in several applications
- **Partial Observation**
  - Agents can only observe local neighboring zones
- **Reward Function**
  - A positive reward for reaching the goal
  - A negative reward for time step or congestion

Policy Optimization in DC Form

- Original Objective Function

$$J(\boldsymbol{\pi}, \boldsymbol{\mu}) = \sum_{\varsigma} p(\varsigma) G(\varsigma)$$

- Why DC programming?
  - The objective is non-linear and nonconvex. Direct optimization is difficult.
  - Nonlinear solvers cannot scale to large number of agents.

DC Programming

❑$\min\{u(x) - v(x) : x \in \Omega\}$
❑$u(x)$ and $v(x)$ are convex functions
❑Concave-Convex Procedure (CCP) can solve it iteratively.

$$x_{k+1} = \operatorname{argimin}\{u(x) - x^T \nabla v(x_k) : x \in \Omega\}$$

- Objective Function in CCP

$$\max_{\pi,\mu} \mathbb{E}\Big[\underbrace{Q^k(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{v})}_{\text{fixed}} \underbrace{\Big(\sum_{i=1}^{N} \log \pi^i(a^i | s^i, y^i)\Big)}_{\text{concave}}\Big] + \gamma \mathbb{E}\Big[\underbrace{Q^k(\boldsymbol{s}', \boldsymbol{a}', \boldsymbol{v}')}_{\text{fixed}} \Big(\underbrace{\Big(\sum_{i=1}^{N} \phi(s', s'^i)\mu^i(a^i, s^i, y^i)\Big)}_{\text{linear}} - \underbrace{\Big(\sum_{i=1}^{N} A\big(\mu^i(a^i, s^i, y^i)\big)\Big)}_{\text{concave}}\Big)\Big]$$

Planning

❑With known model
❑Get a better policy π_{k+1} and μ_{k+1} iteratively
❑General nonlinear solver can be used to optimize it

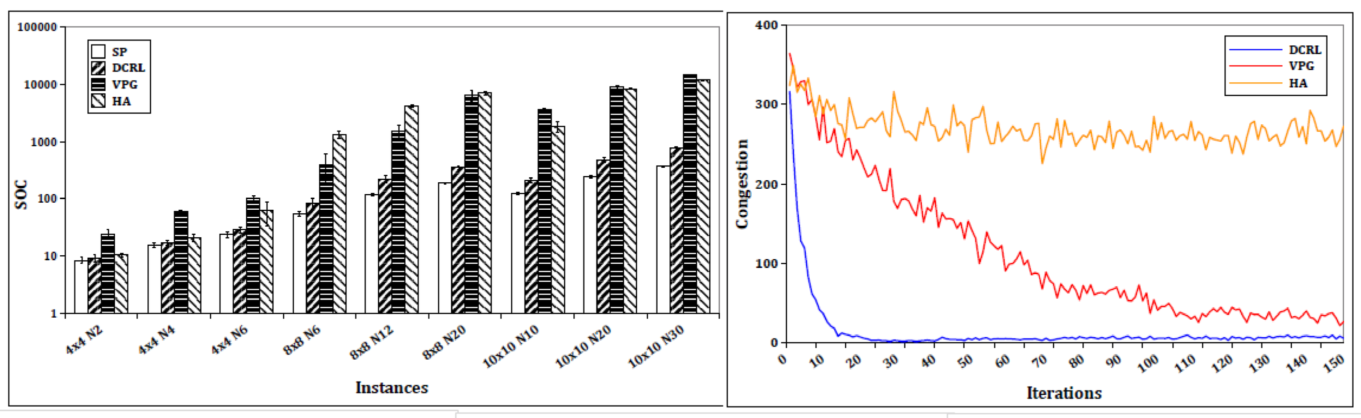
Learning

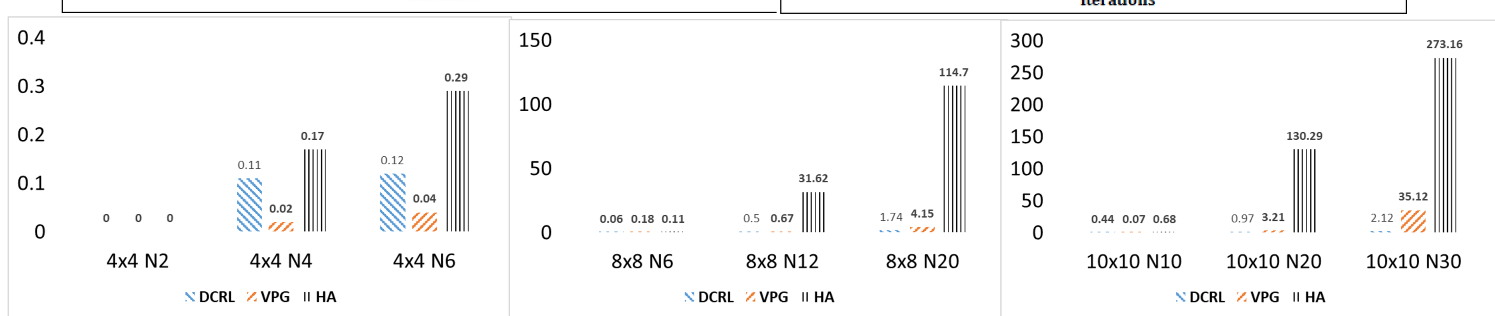
❑Model free setting
❑Assume parameterized policy π_θ and μ_θ
❑Multiagent credit assignment (Low variance gradient estimates)

Experimental Results

2D Open Grids

- **Settings**
  - 4x4 grid, 2 agents to 10x10 grid, 10 agents
  - Starting and goal locations were the top and bottom rows.
  - The capacity of each zone was sampled uniformly from a range e.g., [1,4]
  - t_{min}=1, t_{max}=5 (binomial distribution as travel time dist.)
- **Comparison against**
  - DCRL (our approach); VPG (vanilla Policy Gradient);
  - SP (each agent follows shortest path)
  - HA (multiagent Q-learning based for hybrid action space)
- **Results:**
  - Our approach DCRL provides much better SOC quality can minimize congestion
  - VPG suffers due to lack of effective credit assignment
  - HA isn't able to handle a large number of agents

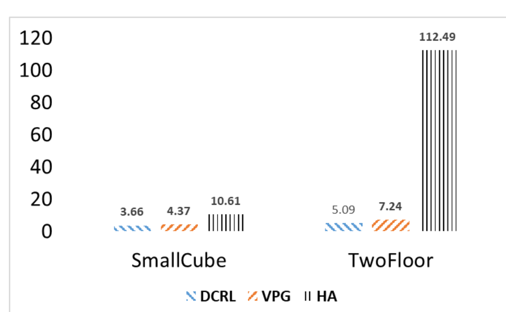




Congestion　　Congestion　　Congestion

3D Maps ("SmallCube" and "TwoFloor")

- **Settings**
  - 10 and 20 agents for SmallCube and TwoFloor
  - Capacity of was uniformly sampled from [1,3] and [1,4].



Congestion

Acknowledgements

This research is supported by the Ministry of Education, Singapore under its Academic Research Fund Tier 2 (MOE2018-T2-1-179).