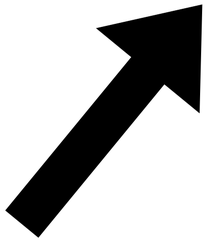
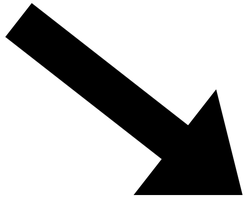
Locally Interdependent Multi-Agent MDP

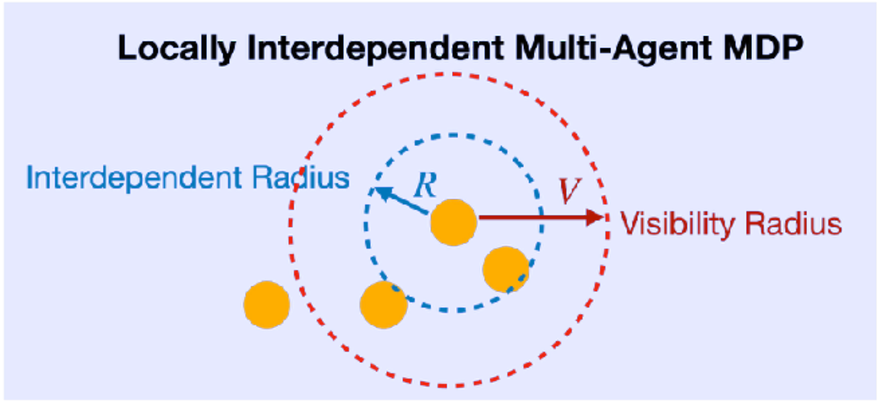
Guannan Qu, Associate Professor, CMU

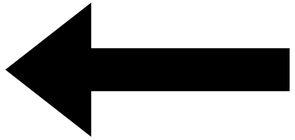
SIGMETRICS, June 8, 2026

Joint work with
Alex DeWeese, PhD student, CMU





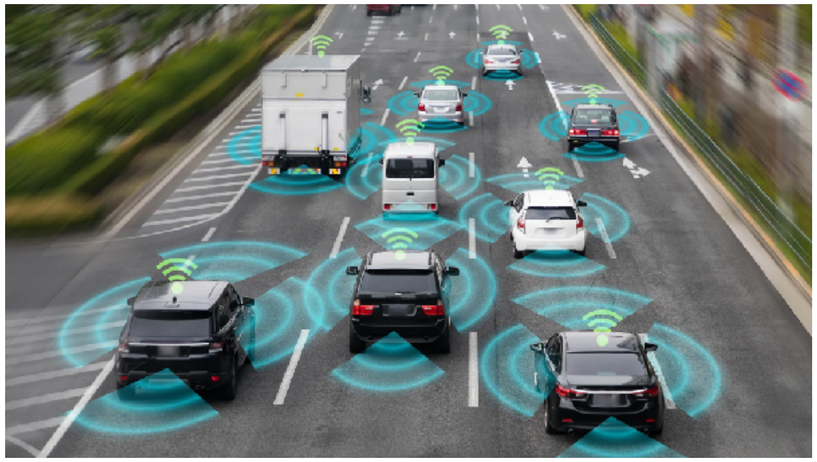




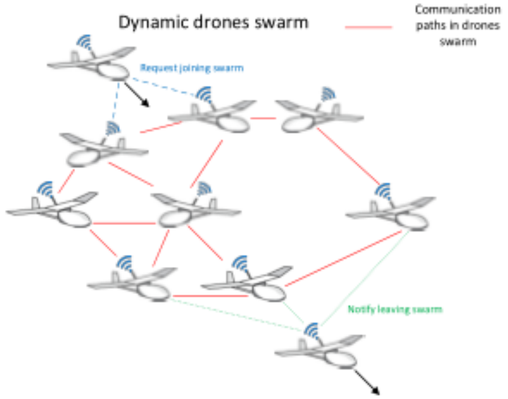
[DeWeese and Qu 2024]
Policy decompositions that are scalable and near optimal

Motivation

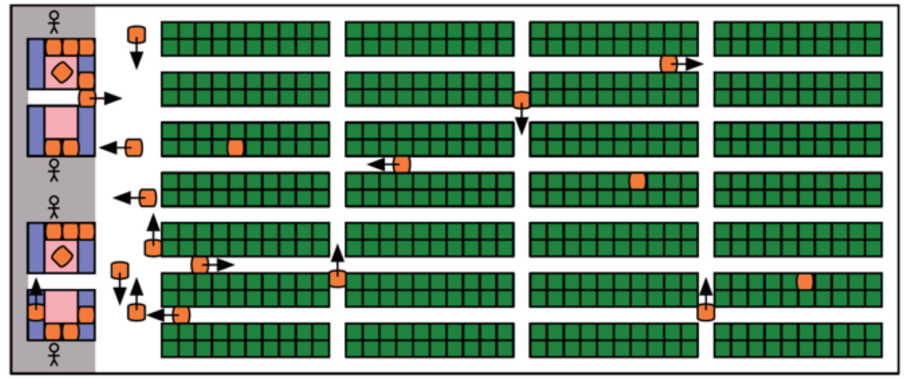
Many practical multi-agent systems are **decentralized** and have **time-varying dependencies**.



Connected Vehicles



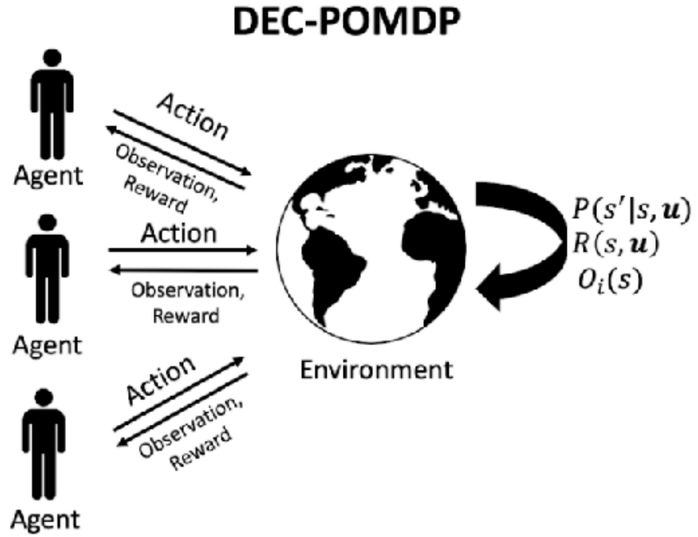
UAV swarms



Warehouse robot fleet

https://www.topgear.com/car%20news/what-are-sae-levels-autonomous-driving-uk
https://arxiv.org/abs/1708.05732
Jiaoyang Li et al, Lifelong multi-agent path finding in large-scale warehouses, 2019

Literature: Dec-POMDP

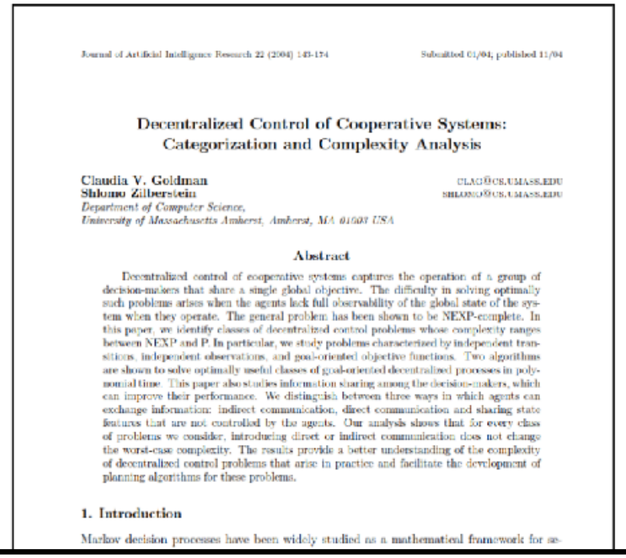


General Dec-POMDP is NEXP-Complete!

[Oliehoek et al. 2012], [Oliehoek et al. 2016]

Picture: https://www.kasimte.com/2019/12/12/what-is-qtran-learning-to-factorize-with-transformation-for-cooperative-marl.ht

Literature: Dec-POMDP



Journal of Artificial Intelligence Research 22 (2004) 143-174 Submitted 01/04; published 11/04

Decentralized Control of Cooperative Systems: Categorization and Complexity Analysis

Claudia V. Goldman CLAG@CS.UMASS.EDU
Shlomo Zilberstein SHLOMO@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts Amherst, Amherst, MA 01003 USA

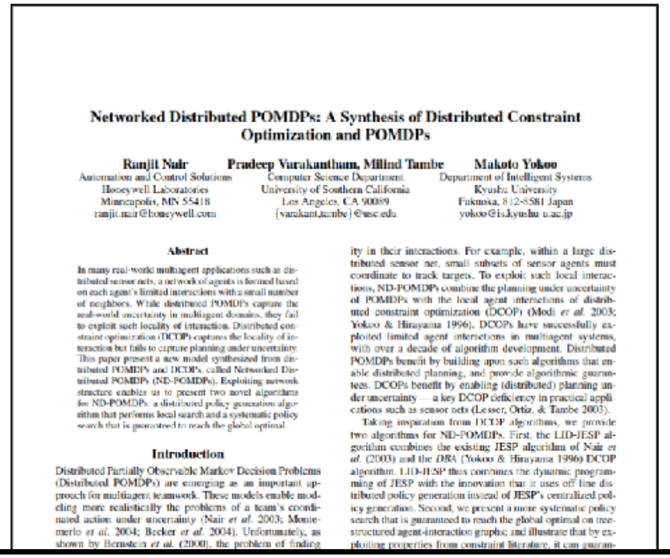
Abstract

Decentralized control of cooperative systems captures the operation of a group of decision-makers that share a single global objective. The difficulty in solving optimally such problems arises when the agents lack full observability of the global state of the system when they operate. The general problem has been shown to be NEXP-complete. In this paper, we identify classes of decentralized control problems whose complexity ranges between NEXP and P. In particular, we study problems characterized by independent transitions, independent observations, and goal-oriented objective functions. Two algorithms are shown to solve optimally useful classes of goal-oriented decentralized processes in polynomial time. This paper also studies information sharing among the decision-makers, which can improve their performance. We distinguish between three ways in which agents can exchange information: indirect communication, direct communication and sharing state features that are not controlled by the agents. Our analysis shows that for every class of problems we consider, introducing direct or indirect communication does not change the worst-case complexity. The results provide a better understanding of the complexity of decentralized control problems that arise in practice and facilitate the development of planning algorithms for these problems.

1. Introduction

Markov decision processes have been widely studied as a mathematical framework for se-

• • •



Networked Distributed POMDPs: A Synthesis of Distributed Constraint Optimization and POMDPs

Ranjit Nair, Automation and Control Solutions, Honeywell Laboratories, Minneapolis, MN 55418, ranjit.nair@honeywell.com
Pradeep Varakantham, Milind Tambe, Computer Science Department, University of Southern California, Los Angeles, CA 90089, {varakant,tambe}@usc.edu
Makoto Yokoo, Department of Intelligent Systems, Kyushu University, Fukuoka, 812-8581 Japan, yokoo@is.kyushu-u.ac.jp

Abstract

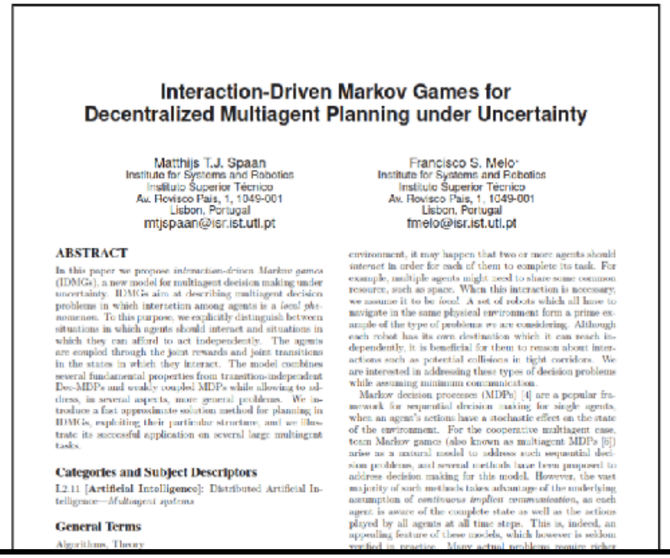
In many real-world multiagent applications such as distributed sensor nets, a network of agents is formed based on each agent's limited interactions with a small number of neighbors. While distributed POMDPs capture the real-world uncertainty in multiagent domains, they fail to exploit such locality of interaction. Distributed constraint optimization (DCOP) captures the locality of interaction but fails to capture planning under uncertainty. This paper presents a new model synthesized from distributed POMDPs and DCOPs, called Networked Distributed POMDPs (ND-POMDPs). Exploiting network structure enables us to present two novel algorithms for ND-POMDPs: a distributed policy generation algorithm that performs local search and a systematic policy search that is guaranteed to reach the global optimal.

Introduction

Distributed Partially Observable Markov Decision Problems (Distributed POMDPs) are emerging as an important approach for multiagent teamwork. These models enable modeling more realistically the problems of a team's coordinated action under uncertainty (Nair *et al.* 2003; Montemerlo *et al.* 2004; Becker *et al.* 2004). Unfortunately, as shown by Bernstein *et al.* (2000), the problem of finding

ity in their interactions. For example, within a large distributed sensor net, small subsets of sensor agents must coordinate to track targets. To exploit such local interactions, ND-POMDPs combine the planning under uncertainty of POMDPs with the local agent interactions of distributed constraint optimization (DCOP) (Modi *et al.* 2003; Yokoo & Hirayama 1996). DCOPs have successfully exploited limited agent interactions in multiagent systems, with over a decade of algorithm development. Distributed POMDPs benefit by building upon such algorithms that enable distributed planning, and provide algorithmic guarantees. DCOPs benefit by enabling (distributed) planning under uncertainty—a key DCOP deficiency in practical applications such as sensor nets (Lesser, Ortiz, & Tambe 2003).

Taking inspiration from DCOP algorithms, we provide two algorithms for ND-POMDPs. First, the LID-JESP algorithm combines the existing JESP algorithm of Nair *et al.* (2003) and the *DBA* (Yokoo & Hirayama 1996) DCOP algorithm. LID-JESP thus combines the dynamic programming of JESP with the innovation that it uses off line distributed policy generation instead of JESP's centralized policy generation. Second, we present a more systematic policy search that is guaranteed to reach the global optimal on tree-structured agent-interaction graphs; and illustrate that by exploiting properties from constraint literature, it can guaran-



Interaction-Driven Markov Games for Decentralized Multiagent Planning under Uncertainty

Matthijs T.J. Spaan, Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal, mtjspaan@isr.ist.utl.pt
Francisco S. Melo, Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal, fmelo@isr.ist.utl.pt

ABSTRACT

In this paper we propose *interaction-driven Markov games* (IDMGs), a new model for multiagent decision making under uncertainty. IDMGs aim at describing multiagent decision problems in which interaction among agents is a *local* phenomenon. To this purpose, we explicitly distinguish between situations in which agents should interact and situations in which they can afford to act independently. The agents are coupled through the joint rewards and joint transitions in the states in which they interact. The model combines several fundamental properties from transition-independent Dec-MDPs and weakly coupled MDPs while allowing to address, in several aspects, more general problems. We introduce a fast approximate solution method for planning in IDMGs, exploiting their particular structure, and we illustrate its successful application on several large multiagent tasks.

environment, it may happen that two or more agents should interact in order for each of them to complete its task. For example, multiple agents might need to share some common resource, such as space. When this interaction is necessary, we assume it to be *local*. A set of robots which all have to navigate in the same physical environment form a prime example of the type of problems we are considering. Although each robot has its own destination which it can reach independently, it is beneficial for them to reason about interactions such as potential collisions in tight corridors. We are interested in addressing these types of decision problems while assuming minimum communication.

Markov decision processes (MDPs) [4] are a popular framework for sequential decision making, for single agents, when an agent's actions have a stochastic effect on the state of the environment. For the cooperative multiagent case, team Markov games (also known as multiagent MDPs [5]) arise as a natural model to address such sequential decision problems, and several methods have been proposed to address decision making for this model. However, the vast majority of such methods takes advantage of the underlying assumption of *continuous implicit communication*, as each agent is aware of the complete state as well as the actions played by all agents at all time steps. This is, indeed, an appealing feature of these models, which however is seldom verified in practice. Many actual problems require richer

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Theory

Many variants have proposed, but most of these results are negative (NP-hardness)!

- Interaction Driven Markov Game (IDMG) [Spaan & Melo 2008], [Melo Veloso 2009]
- Networked Distributed POMDP [Nair et al. 2005] [Zhang & Lesser, 2011]
- [Goldman & Zilberstein 2004], [Berstein et al. 2002]; [Allen Zilberstein, 2009]

Picture: https://www.kasimte.com/2019/12/12/what-is-qtran-learning-to-factorize-with-transformation-for-cooperative-marl.ht

Our Goal

Much of the theoretical literature on Dec-POMDP have hardness results due to:

- Partially Observability
- Scalability (Curse of Dimensionality), i.e. factored state/action space

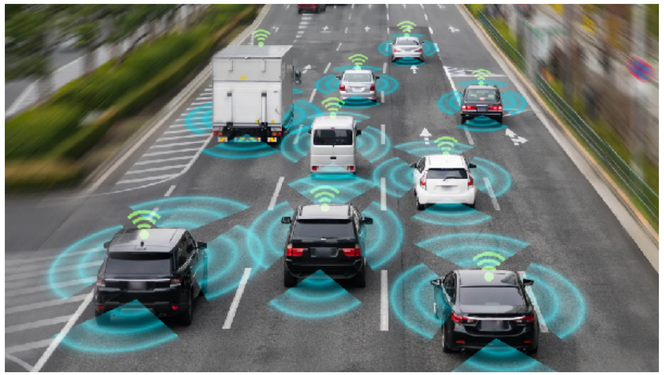
… but Dec-POMDP setting is overly broad and do not consider real-world problem structures

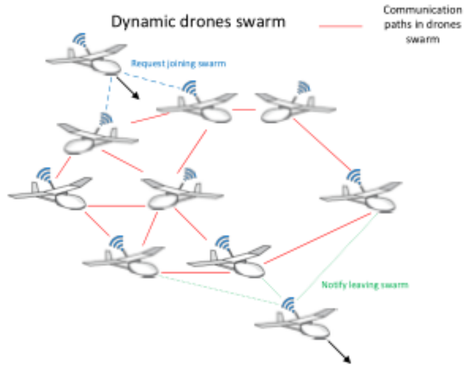
Our Goal

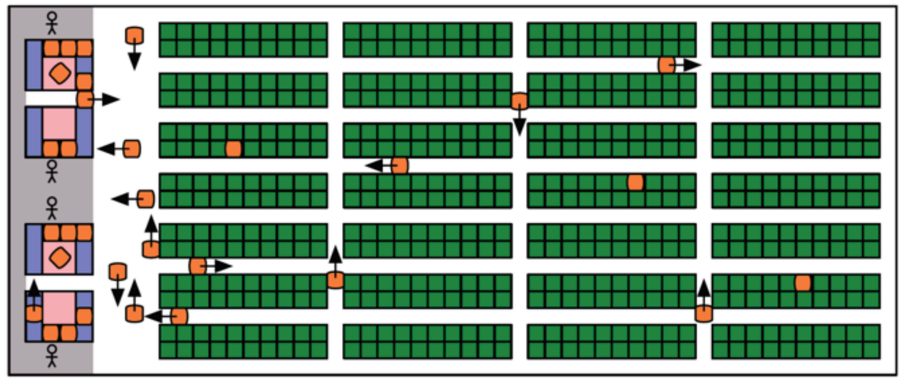
… but Dec-POMDP setting is overly broad and do not consider real-world problem structures

- In connected vehicles, each car interact with nearby cars
- Each drone in a swarm have aerodynamic interactions with nearby drones
- Warehouse robot fleets need to navigate while avoiding collision/congestion

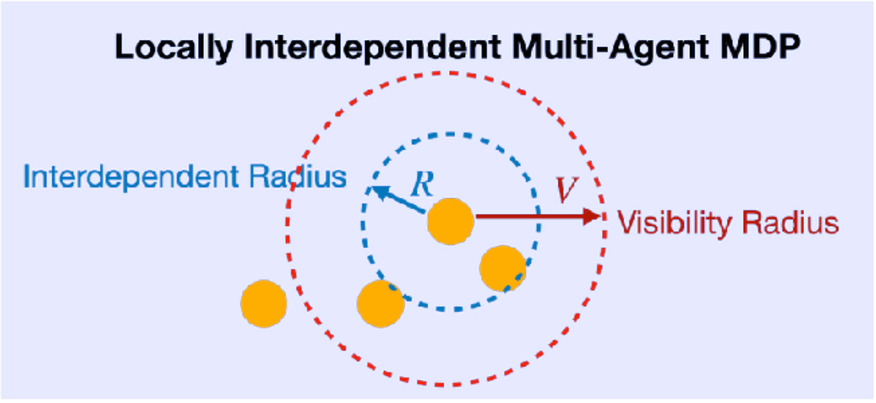
In all examples, there is sense of local interdependence (“nearby”), but it is dynamic as agents can move



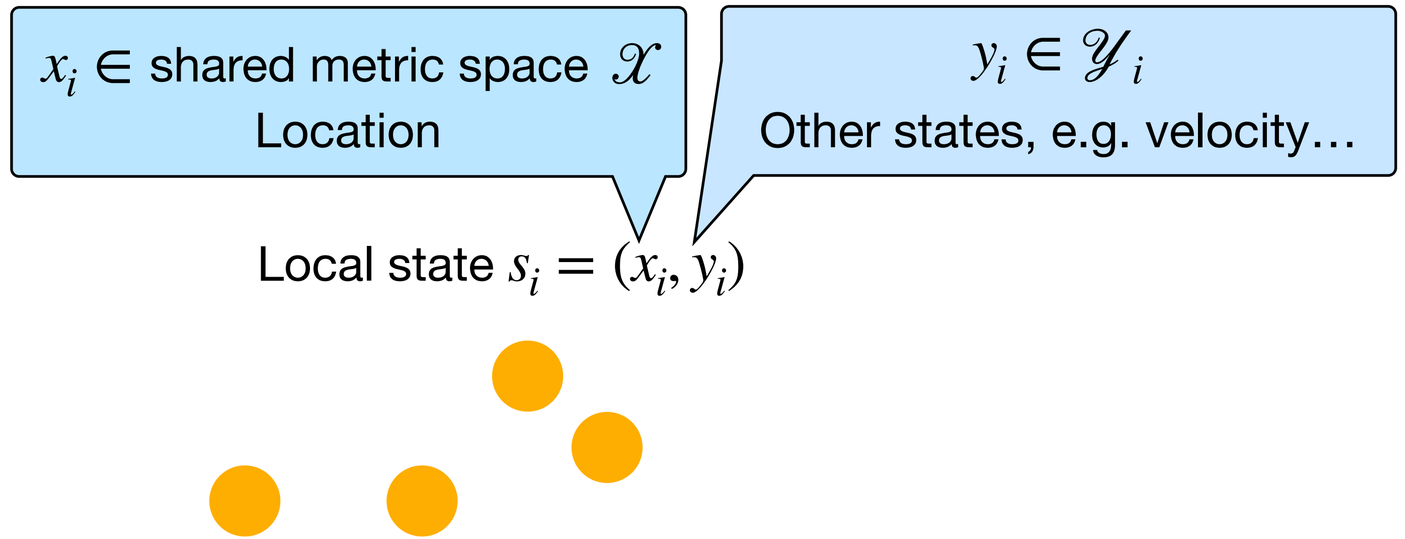




Up next!


Locally Interdependent Multi-Agent MDP
Interdependent Radius
R
V
Visibility Radius

Locally Interdependent Multi-Agent MDP



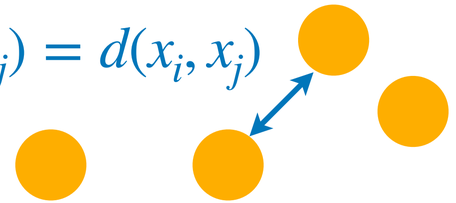
Locally Interdependent Multi-Agent MDP

Global State Space $\mathcal{S} := \mathcal{S}_1 \times \ldots \times \mathcal{S}_n$

Local State Space $\mathcal{S}_i = \mathcal{X} \times \mathcal{Y}_i$

Local state $s_i = (x_i, y_i)$

distance $d(s_i, s_j) = d(x_i, x_j)$



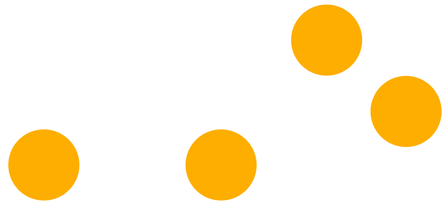
$s_j = (x_j, y_j)$

Locally Interdependent Multi-Agent MDP

Global Action Space $\mathscr{A} := \mathscr{A}_1 \times \ldots \times \mathscr{A}_n$

Local Action Space $\mathscr{A}_i$

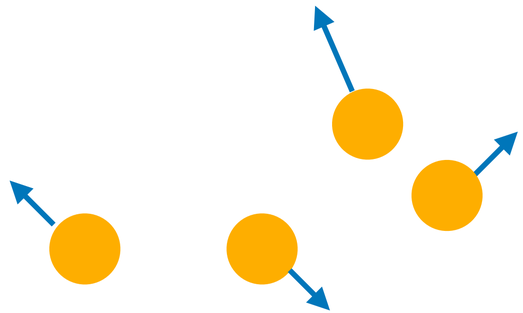
Local action a_i



Locally Interdependent Multi-Agent MDP

Each agent's state transits independently: $P(s' \mid s, a) = \prod_{k \in \mathcal{N}} P_k(s'_k \mid s_k, a_k)$

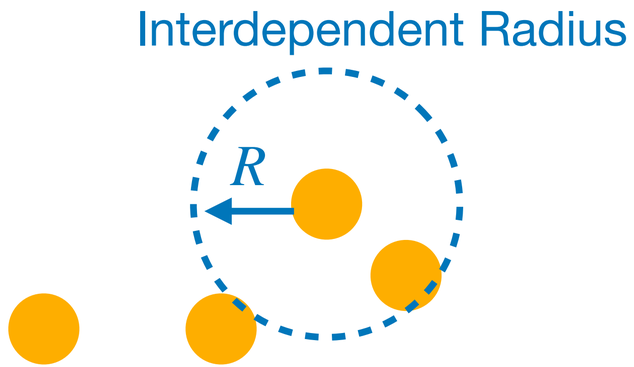
… but their distance can only **move up to distance 1**: $P_k(s'_k \mid s_k, a_k) = 0$ if $d(s_k, s'_k) > 1$



Locally Interdependent Multi-Agent MDP

$$r(s,a) = \sum_i r_i(s_i, a_i) + \sum_{j,k:d(s_j,s_k)\leq R, j\neq k} \bar{r}_{j,k}(s_j, a_j, s_k, a_k)$$

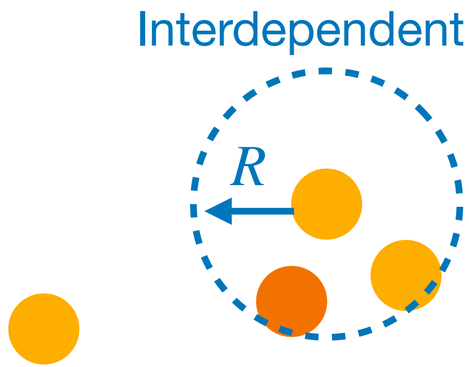
Every pair of agents within radius R will have interdepenent reward



Locally Interdependent Multi-Agent MDP

$$r(s,a) = \sum_i r_i(s_i, a_i) + \sum_{j,k:d(s_j,s_k)\leq R, j\neq k} \bar{r}_{j,k}(s_j, a_j, s_k, a_k)$$

Every pair of agents within radius R will have interdepenent reward



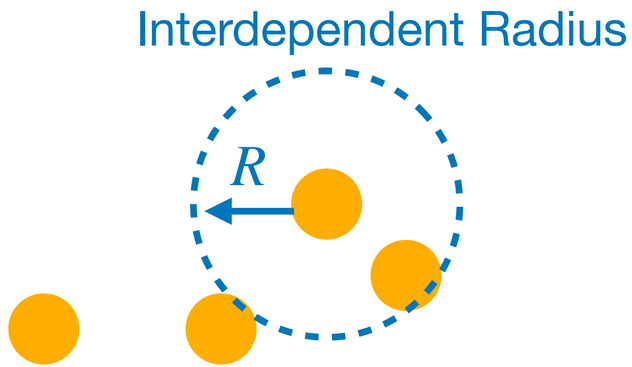
As agents can move, the interdependence is dynamic

Agents can join or leave one's interdependent radius

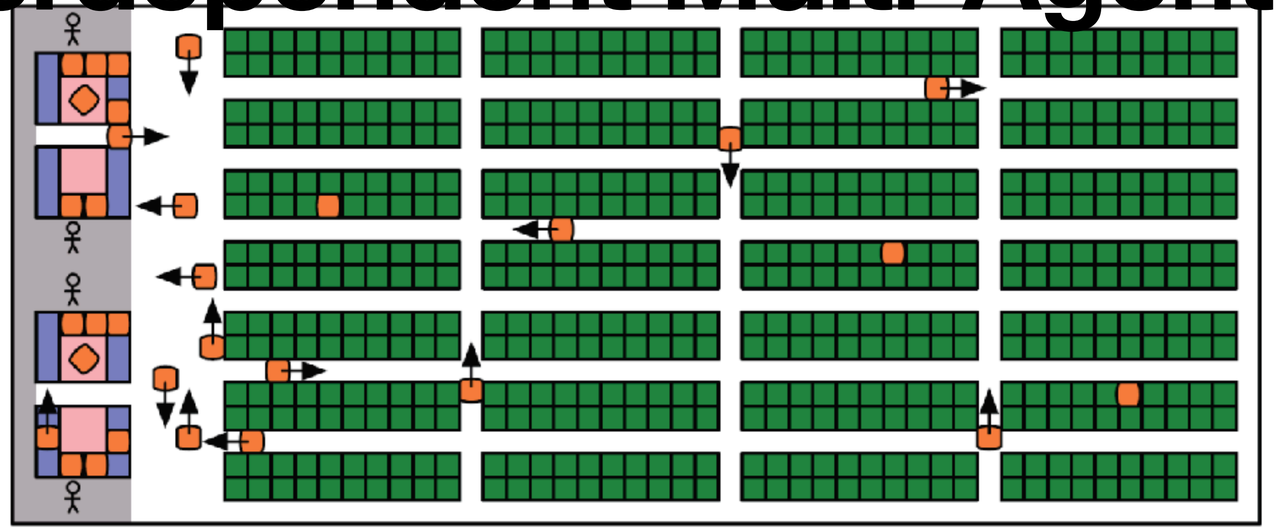
Locally Interdependent Multi-Agent MDP

Applications

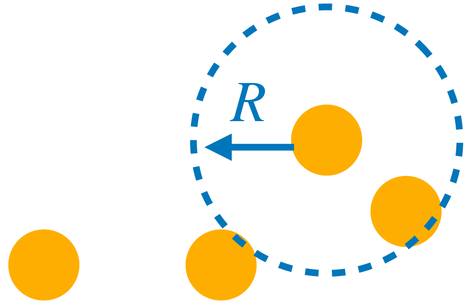
- Cooperative Navigation: each agent wants to reach goal **while avoiding collision within R**
- Formation Control: each agent maintains relative distance/angle **to the agents within R**



Locally Interdependent Multi-Agent MDP

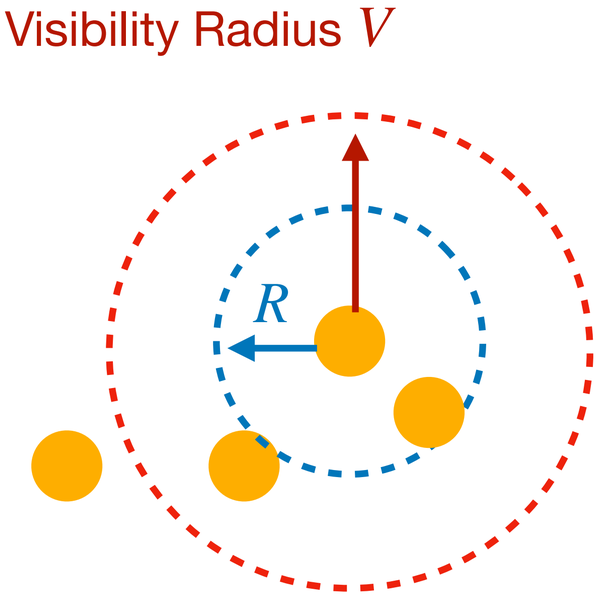


Interdependent Radius

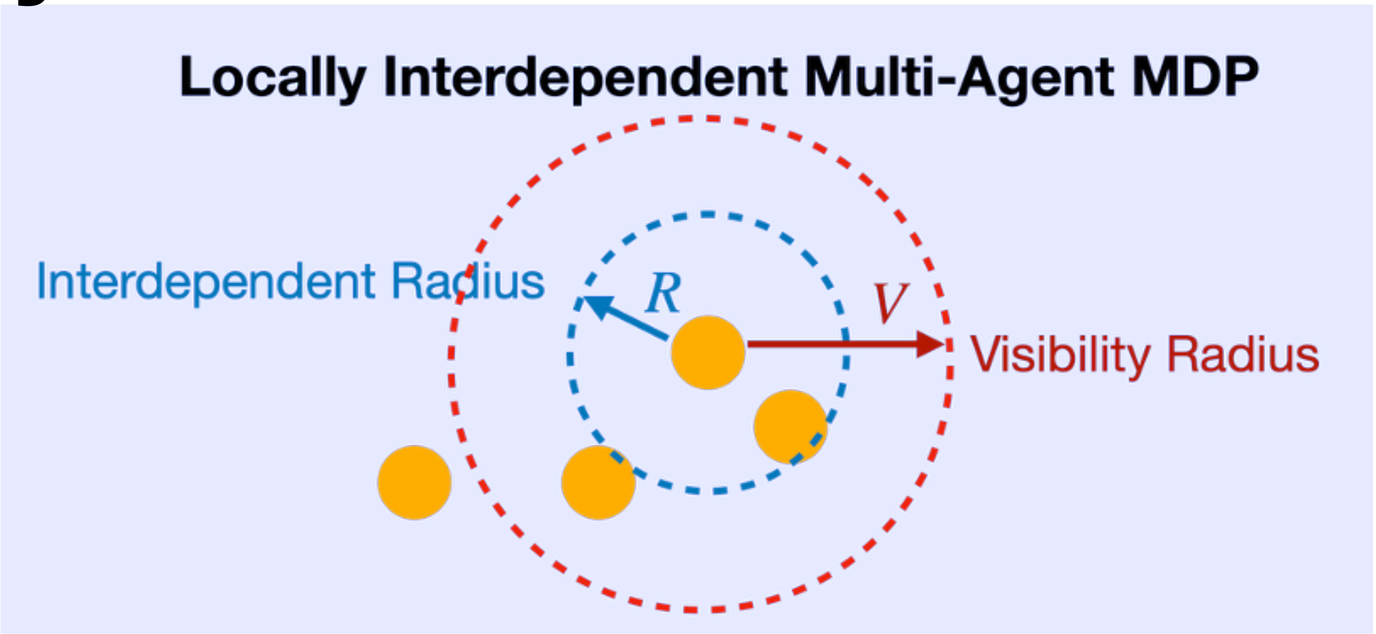


Locally Interdependent Multi-Agent MDP

Each agent can see and communicate with agents within radius V



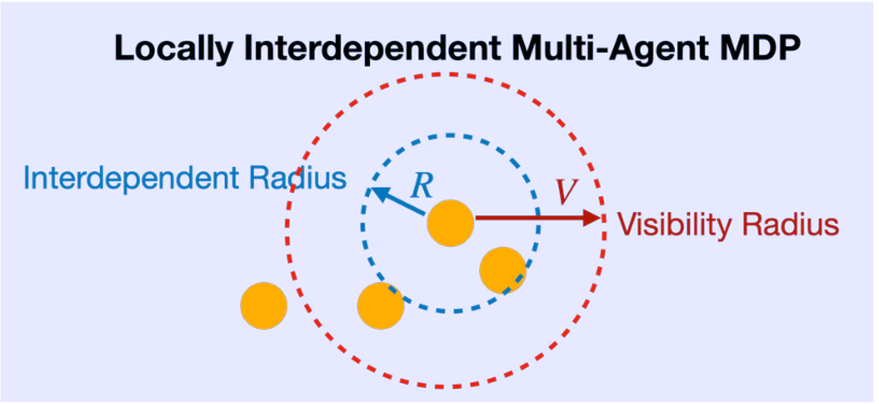
Summary of LIMDP



- Agents in metric space
- Independent transition with speed limit
- Reward dependence within radius R
- Visibility radius V

Up Next

[DeWeese and Qu 2024]
Policy decompositions that are scalable and near optimal



Does Locally Interdependent Multi-Agent MDP admits **scalable** and **near optimal** solution?

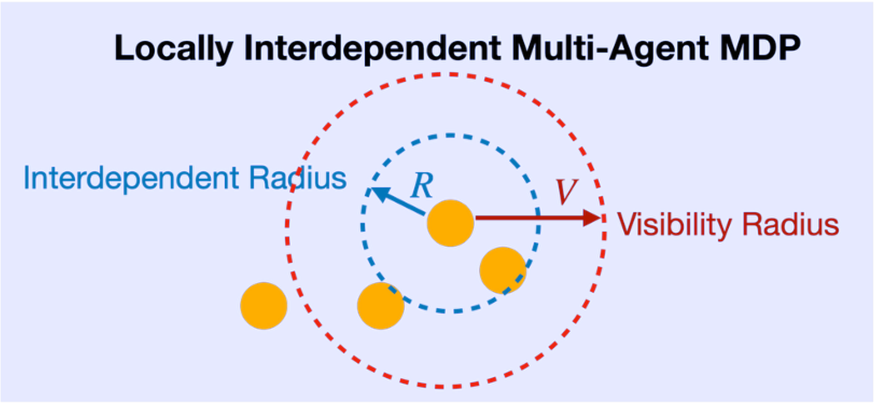
Key Idea: Group Decentralized Policy Class

Contain **near optimal** policies with closed form construction

Scalable to compute

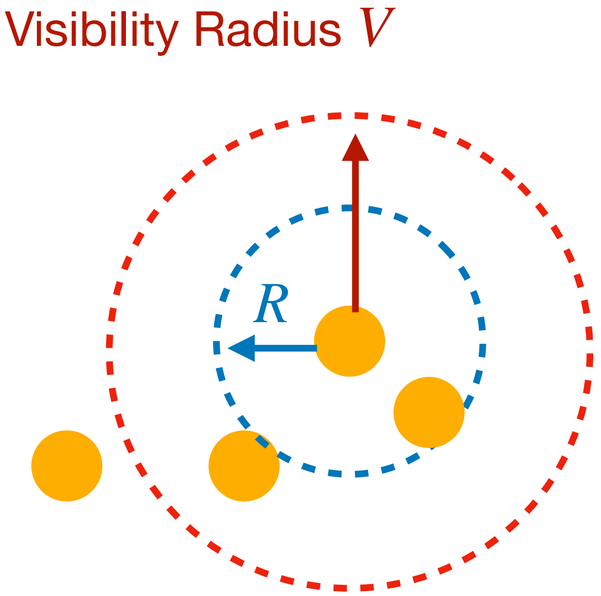
[DeWeese and Qu 2024]
Policy decompositions that are scalable and near optimal

Up Next



Recall: Visibility Radius

Each agent can see and communicate with agents within radius V

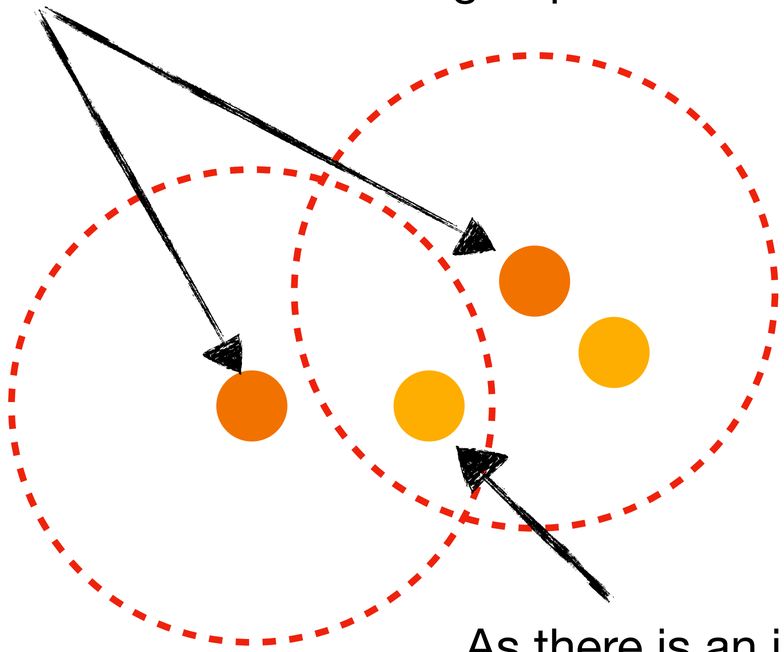


Group Decentralized Policies

Visibility Partition $Z(s)$ splits agents into visibility groups

Two agents are in the same vibility group if there are a chain of agents within visibility of each other.

These two agents are in the same group

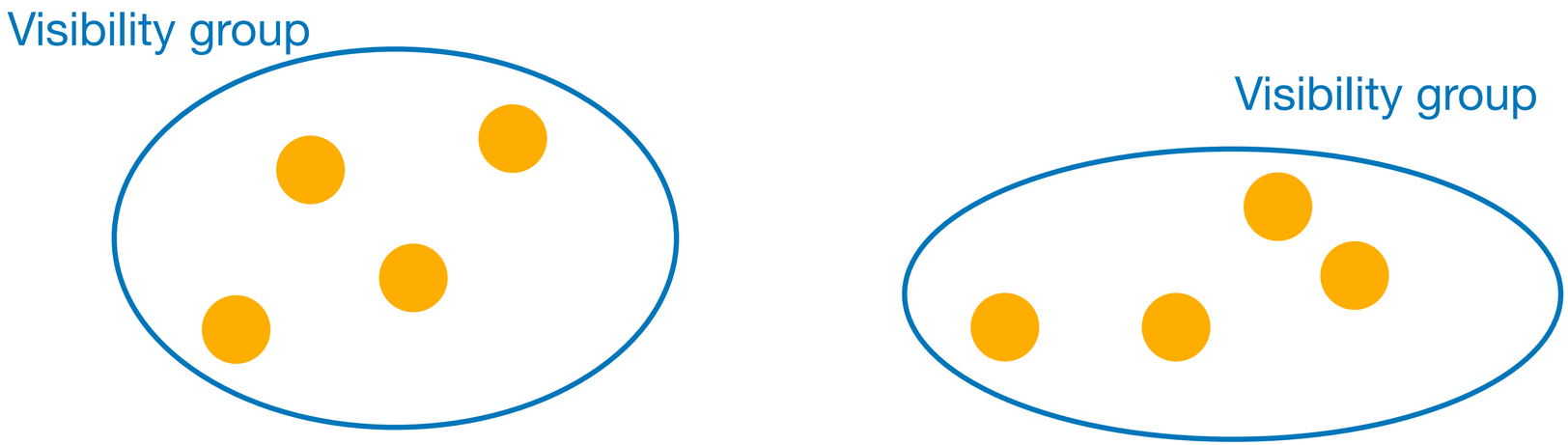


As there is an intermediate agent visible to both

Group Decentralized Policies

Visibility Partition $Z(s)$ splits agents into visibility groups

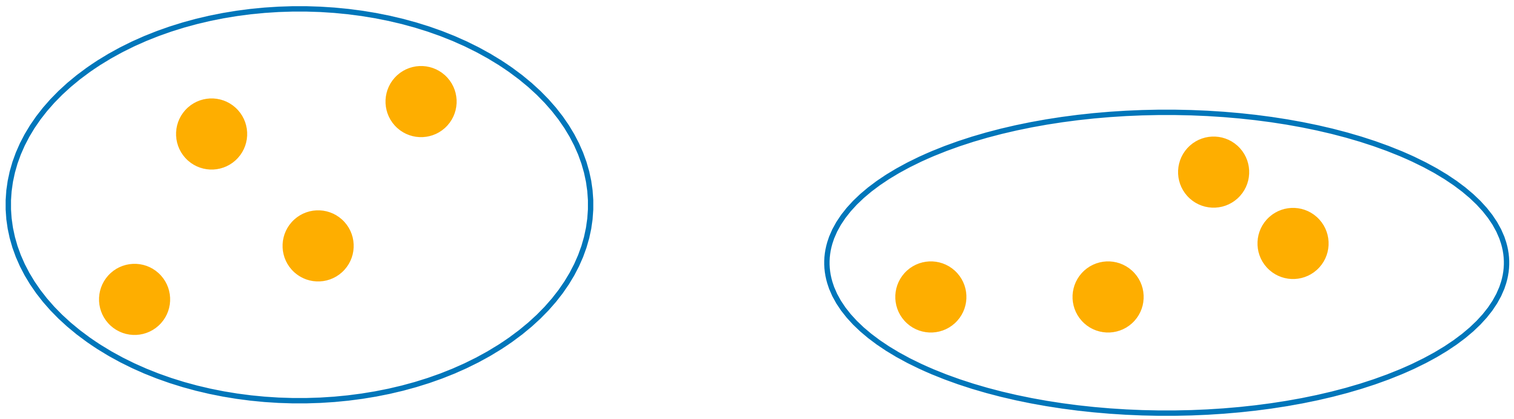
Two agents are in the same vibility group if there are a chain of agents within visibility of each other.



Group Decentralized Policies

Note: the groups can be time varing!

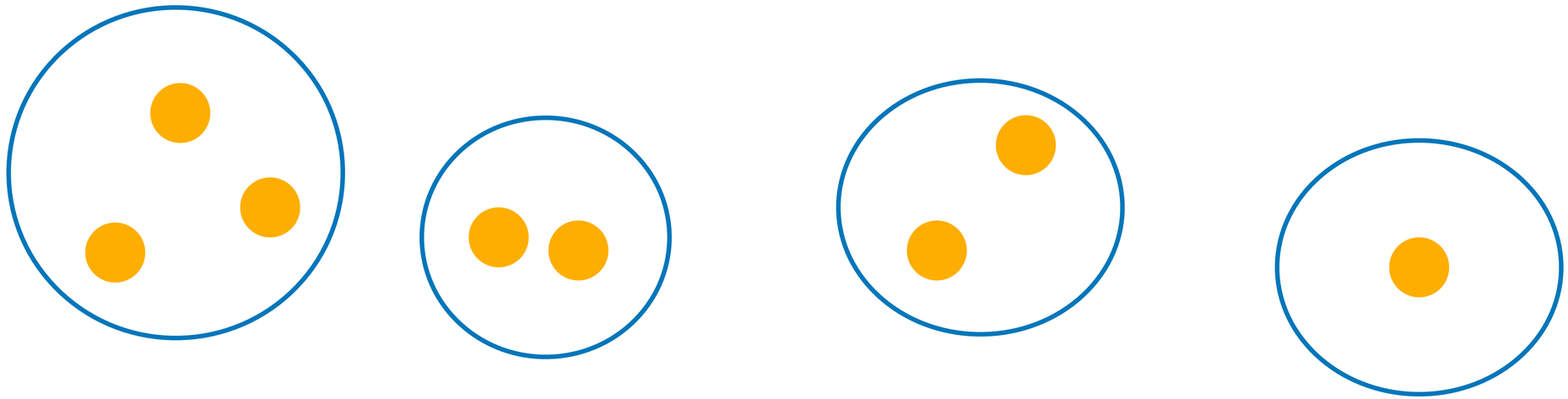
Groups can break up and new groups can form as agents move



Group Decentralized Policies

Note: the groups can be time varing!

Groups can break up and new groups can form as agents move

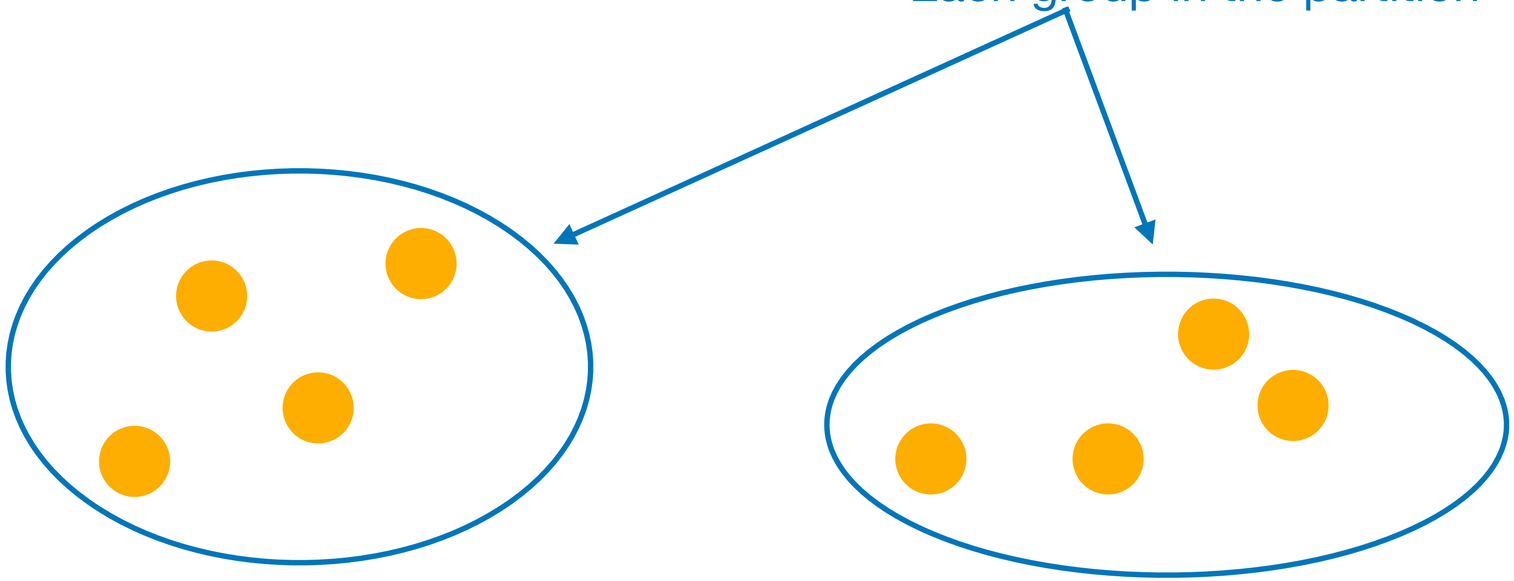


Group Decentralized Policies

Group Decentralized Policy

$$\pi(s) = (\pi_z(s_z) : z \in Z(s))$$

Each group in the partition

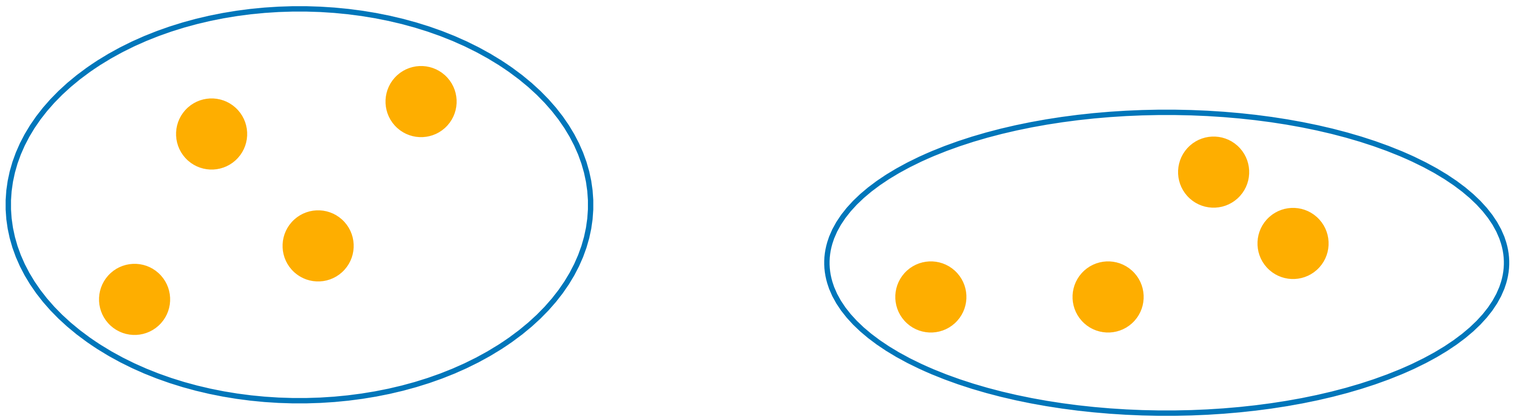


Group Decentralized Policies

Group Decentralized Policy

$$\pi(s) = (\pi_z(s_z) : z \in Z(s))$$

Each group acts independently from other groups!

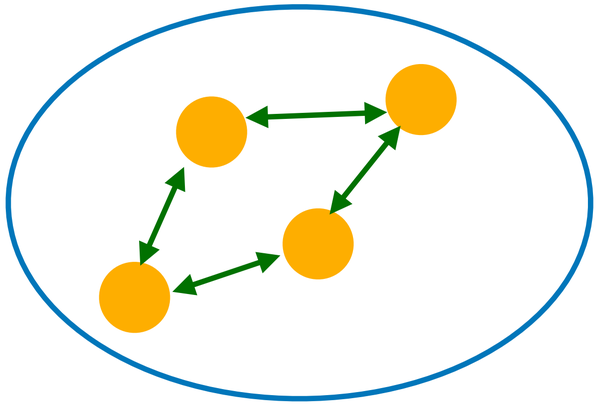


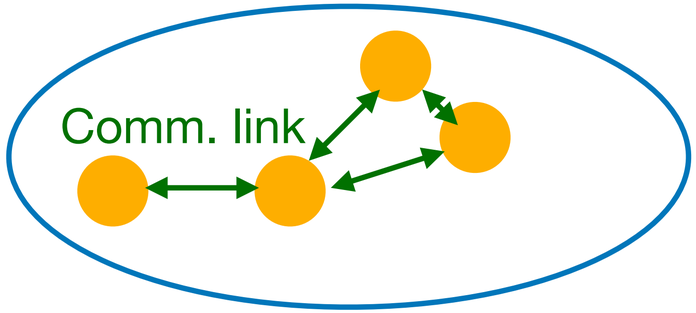
Group Decentralized Policies

Group decentralized policies are practical to implement

Assume two agents visible to each other can communicate

Agents in visibility group form a connected communication graph





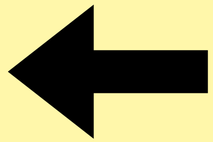
Does Locally Interdependent Multi-Agent MDP admits **scalable** and **near optimal** solution?

Key Idea: Group Decentralized Policy Class

Contain **near optimal** policies with closed form construction

Scalable to compute

Up next



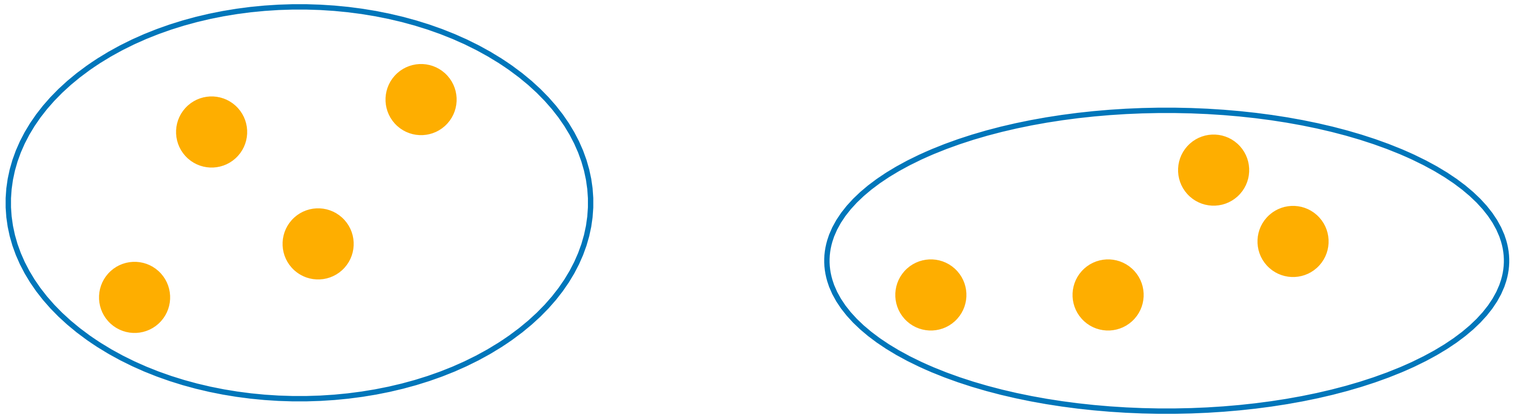
Contain near optimal policies with closed form construction

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Each group take optimal policy

pretending others don't exist!



Contain near optimal policies with closed form construction

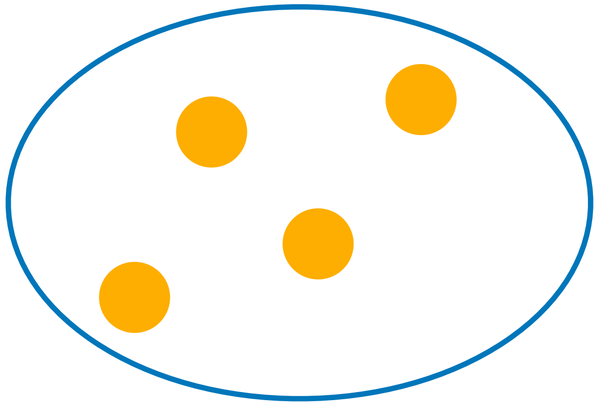
Amalgam Policy

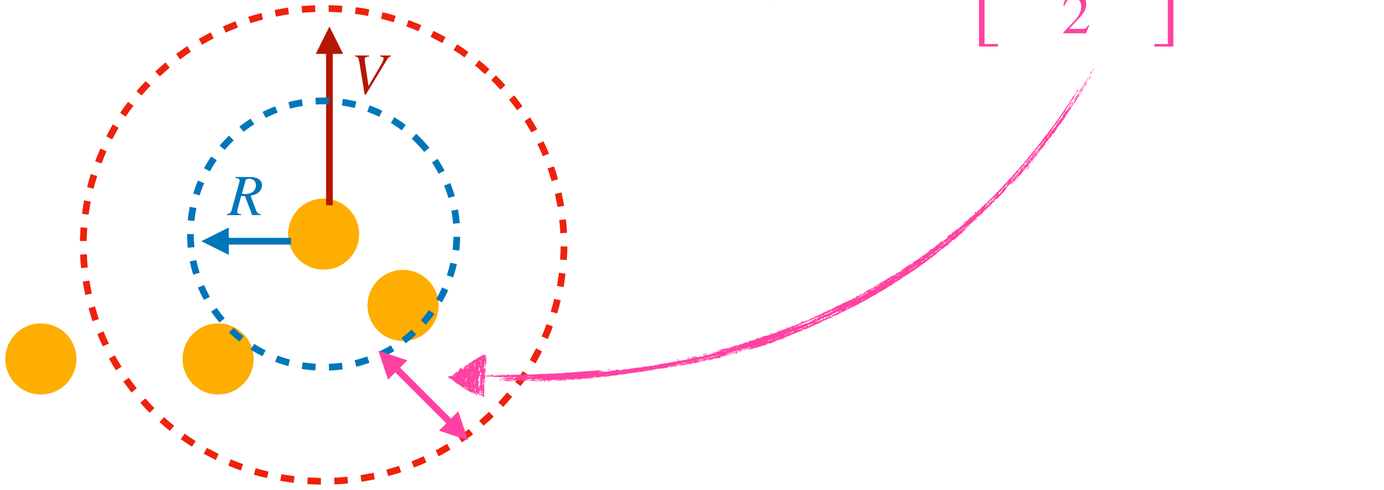
$$\lambda(s) = (\pi^*_z(s_z) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\lambda}(s) \leq \frac{2}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

Exponentially decaying in $c := \left\lfloor \frac{V-R}{2} \right\rfloor = \Theta(V)$





Contain near optimal policies with closed form construction

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\lambda}(s) \leq \frac{2}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

Exponentially decaying in $c := \left\lfloor \frac{V-R}{2} \right\rfloor = \Theta(V)$

This is **near opitmal** for reasonable large V (as it decays exponentailly in V)

Two other closed-form policies

Cutoff Policy

$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\chi}(s) \leq \frac{2-\gamma}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

This is **near opitmal** for reasonable large V (as it decays exponentailly in V)

Two other closed-form policies

First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Guarantee

$$V^*(s) - V^{\phi}(s) \leq \frac{2}{1-\gamma}\gamma^{c+1} r_{\max}$$

$\{\pi^{*,0}, \pi^{*,1}, \ldots, \pi^{*,c}\}$- are the set of discounted finite horizon optimal policies with horizon $c = \lfloor \frac{\mathscr{V} - \mathscr{R}}{2} \rfloor$

This is **near opitmal** for reasonable large V (as it decays exponentailly in V)

Comparison

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\lambda}(s) \leq \frac{2}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

Cutoff Policy

$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\chi}(s) \leq \frac{2-\gamma}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

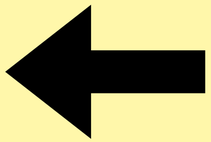
Guarantee

$$V^*(s) - V^{\phi}(s) \leq \frac{2}{1-\gamma}\gamma^{c+1} r_{\max}$$

Does Locally Interdependent Multi-Agent MDP admits **scalable** and **near optimal** solution?

Key Idea: Group Decentralized Policy Class

Contain **near optimal** policies with closed form construction

Scalable to compute  **Up next**

Scalable to compute

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Optimal policies for each group are exponetially* more scalable to store and compute

than optimal centralized policy

*Caveat: only holds when groups are small

may become harder and harder when groups become larger.

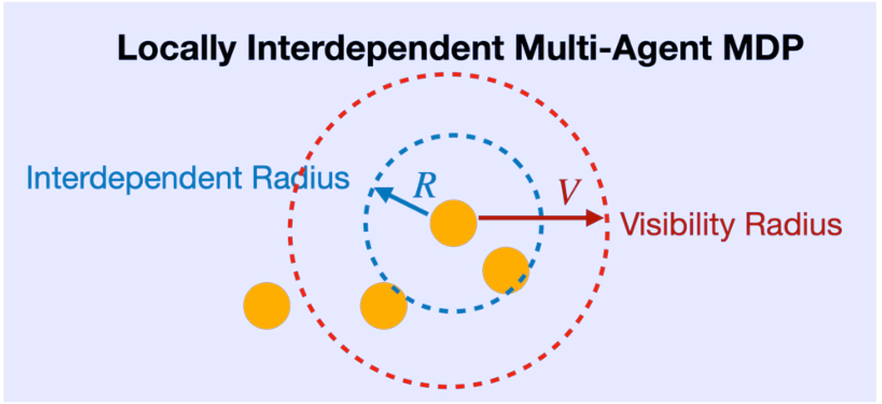
Does Locally Interdependent Multi-Agent MDP admits **scalable** and **near optimal** solution?

Key Idea: Group Decentralized Policy Class

Contain **near optimal** policies with closed form construction

Scalable to compute

[DeWeese and Qu 2024]
Policy decompositions that are scalable and near optimal



Proof Idea

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\lambda}(s) \leq \frac{2}{(1-\gamma)^2}\gamma^{c+1}r_{\max}$$

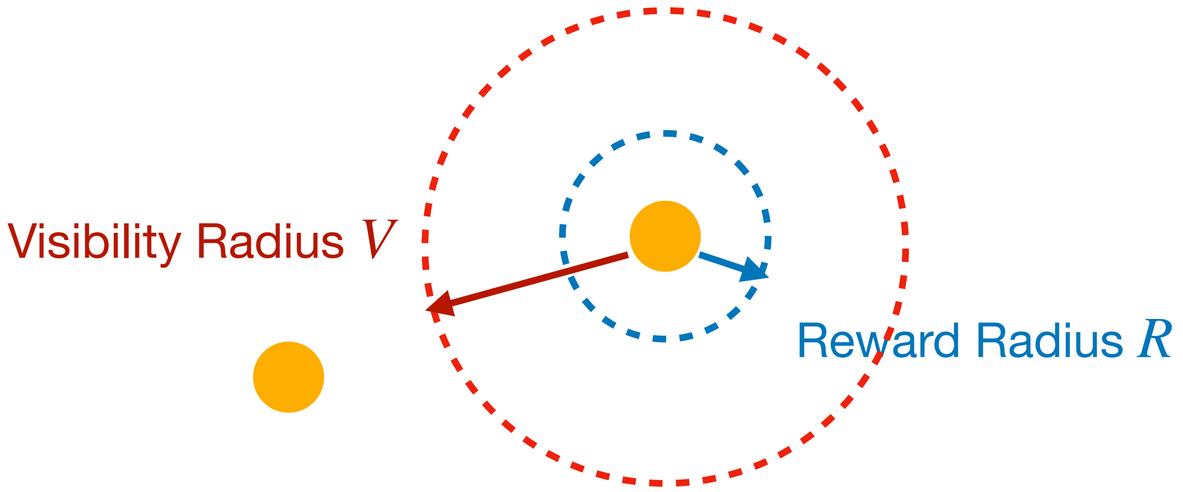
Exponentially decaying in $c := \left\lfloor \frac{V-R}{2} \right\rfloor = \Theta(V)$

How to prove this?

Proof Idea: Dependence Time Lemma

Suppose two agents are initially out of view

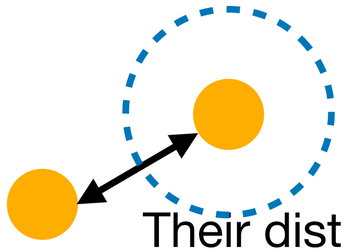
If they try to get close as much as possible in $c = \left\lfloor \frac{V - R}{2} \right\rfloor$ steps



Proof Idea: Dependence Time Lemma

Suppose two agents are initially out of view

If they try to get close as much as possible in $c = \left\lfloor \frac{V-R}{2} \right\rfloor$ steps



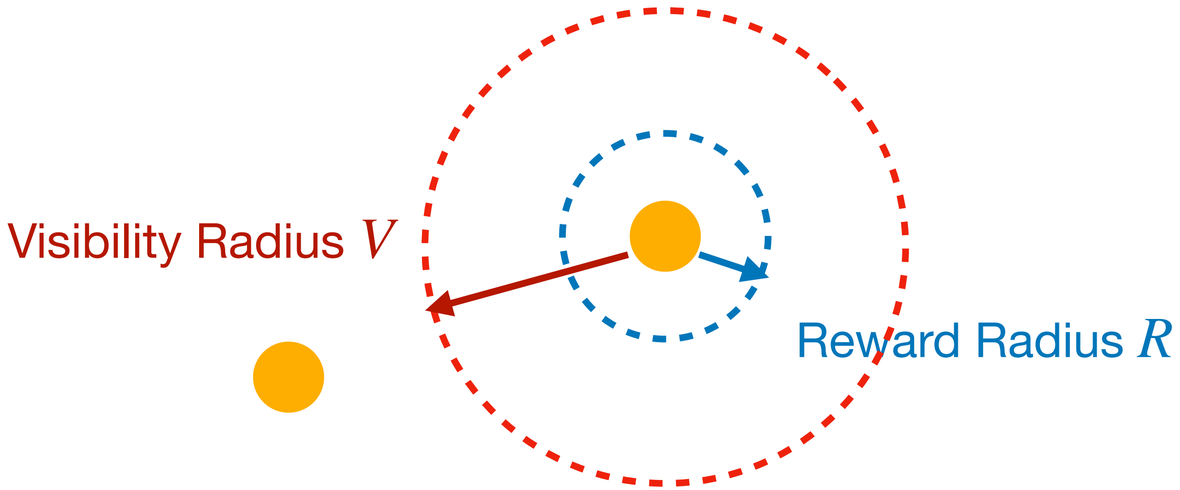
Their dist $> V - 2c = V - 2\lfloor \frac{V-R}{2} \rfloor > R$

Proof Idea: Dependence Time Lemma

Key Observation

Suppose two agents are initially out of view

Within $c = \left\lfloor \frac{V - R}{2} \right\rfloor$ steps, the two agents cannot have interdependent rewards!



Near Optimal and Scalable Policies

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

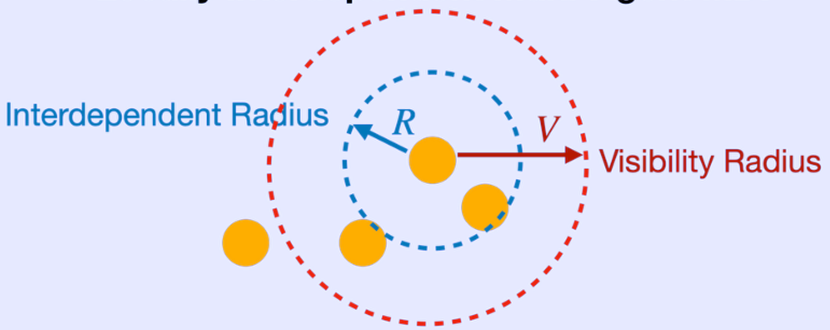
Cutoff Policy

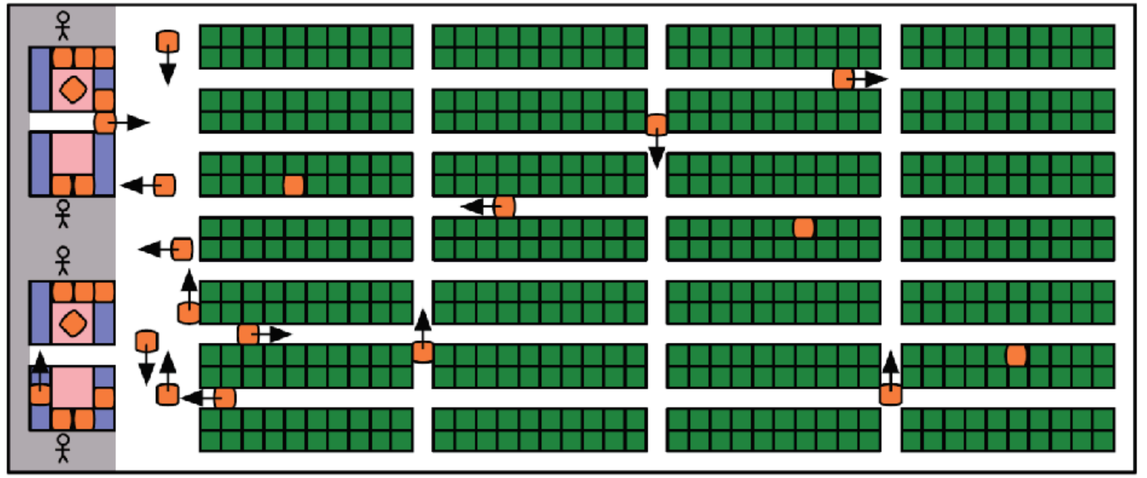
$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Locally Interdependent Multi-Agent MDP





Near Optimal and Scalable Policies

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

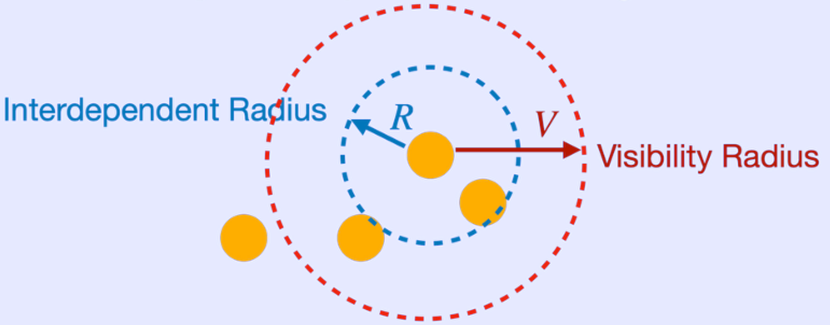
Cutoff Policy

$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

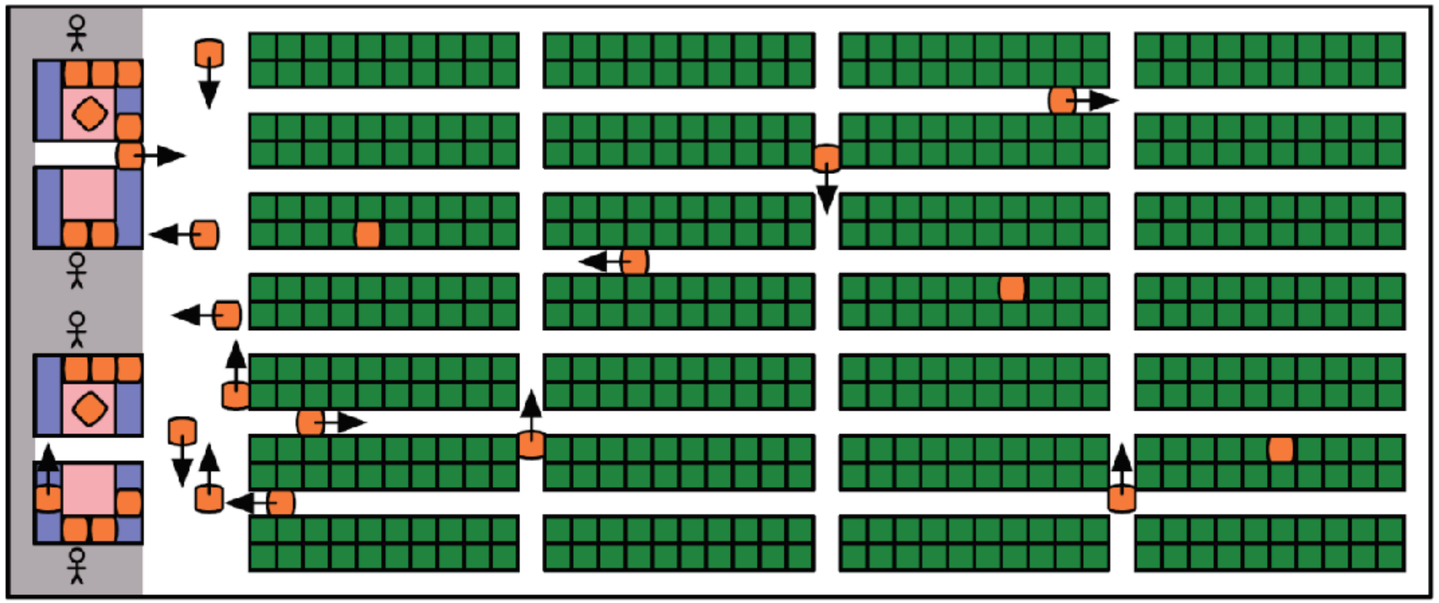
First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Locally Interdependent Multi-Agent MDP



Applications: Multi-Agent Path Finding



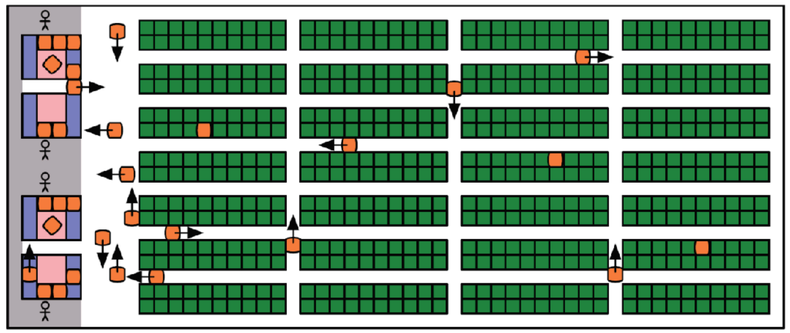
Each robot reaches target locations while avoiding collision

Applications: Multi-Agent Path Finding

State-of-the-art

Lifelong Multi-Agent Path Finding in Large-Scale Warehouses*

Jiaoyang Li,[1] Andrew Tinka,[2] Scott Kiesel,[2]
Joseph W. Durham,[2] T. K. Satish Kumar,[1] Sven Koenig[1]
[1] University of Southern California
[2] Amazon Robotics



Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

Cutoff Policy

$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

First Step Finite Horizon Optimal Policy

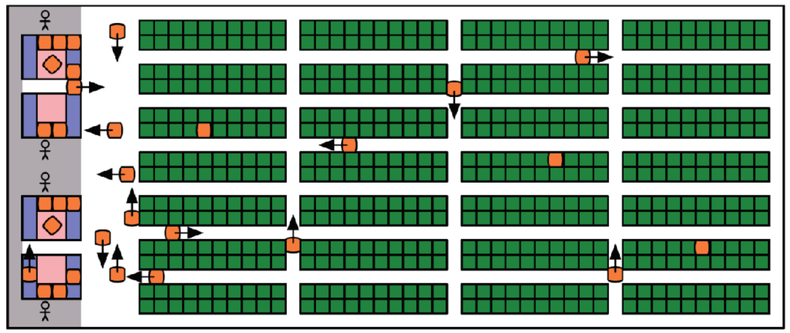
$$\phi(s) = \pi^{*,0}(s)$$

Applications: Multi-Agent Path Finding

State-of-the-art

Lifelong Multi-Agent Path Finding in Large-Scale Warehouses*

Jiaoyang Li,[1] Andrew Tinka,[2] Scott Kiesel,[2]
Joseph W. Durham,[2] T. K. Satish Kumar,[1] Sven Koenig[1]
[1] University of Southern California
[2] Amazon Robotics



Equivalent

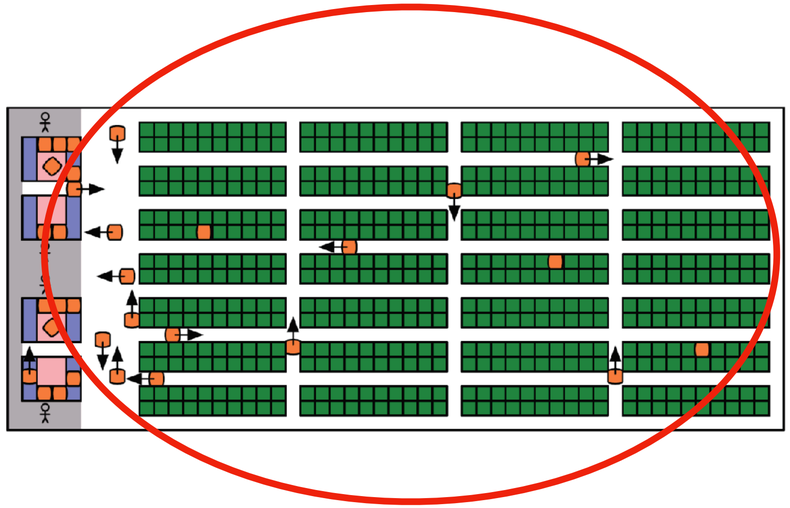
Both solve a finite-horizon planning problem and take the first action

First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Applications: Multi-Agent Path Finding

All agents form a giant group



State-of-the-art

Lifelong Multi-Agent Path Finding in Large-Scale Warehouses*

Jiaoyang Li,[1] Andrew Tinka,[2] Scott Kiesel,[2]
Joseph W. Durham,[2] T. K. Satish Kumar,[1] Sven Koenig[1]
[1] University of Southern California
[2] Amazon Robotics

. but they are using $V = \infty$

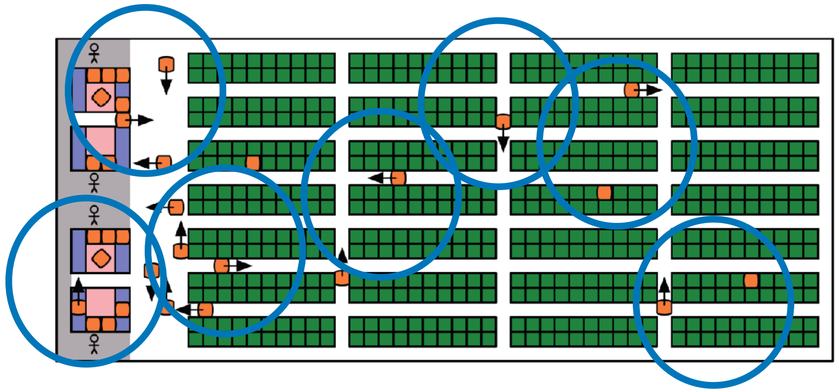
First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Applications: Multi-Agent Path Finding

All agents form small groups

Much imporved scalability



Our results natually suggest we can use **a small** V

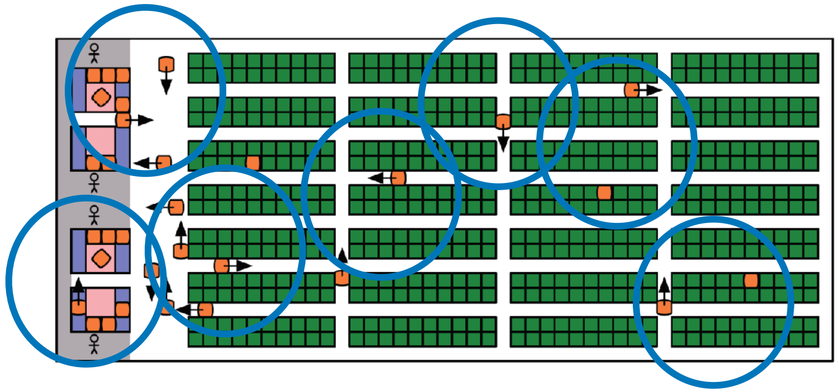
First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Applications: Multi-Agent Path Finding

All agents form small groups

Much imporved scalability



Our results natually suggest we can use **a small** V

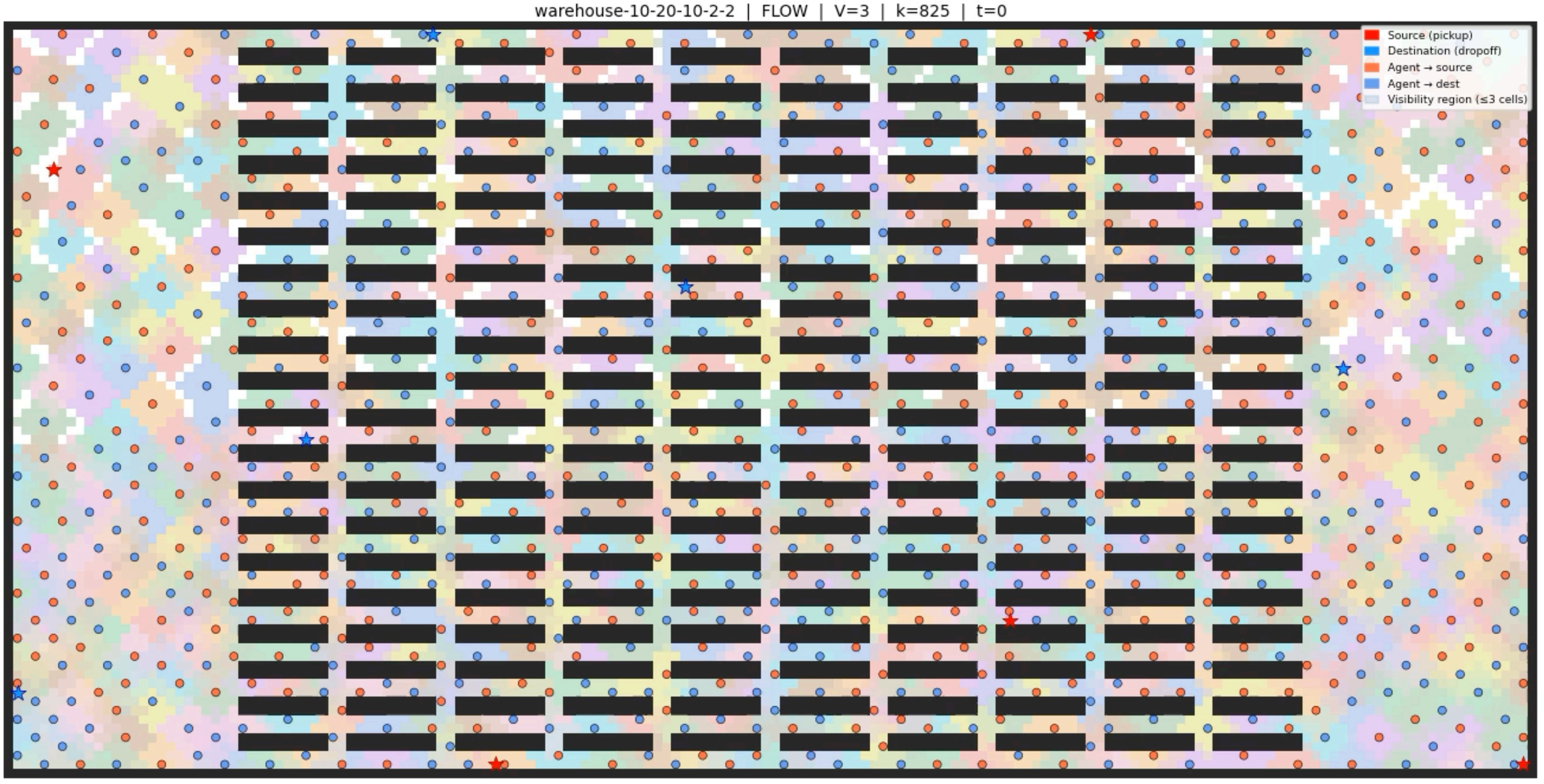
Only suffer a small loss of optimality

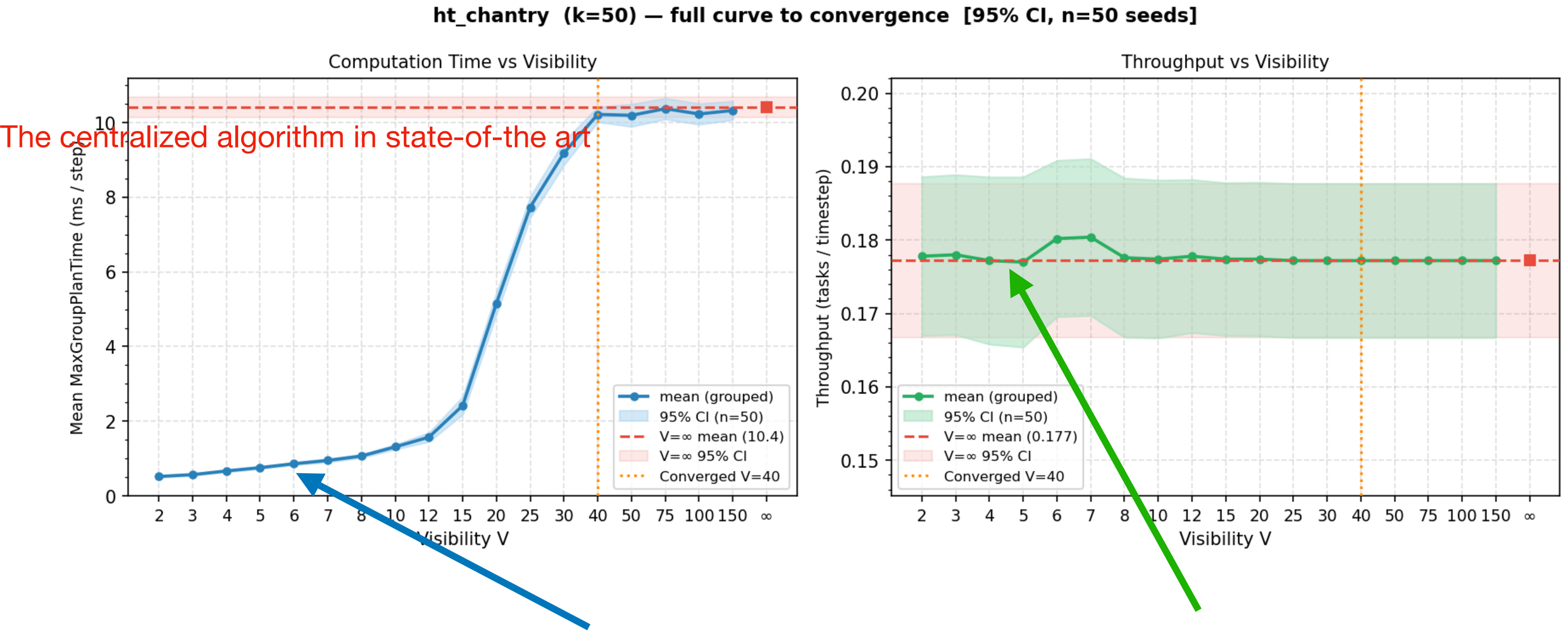
First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Guarantee

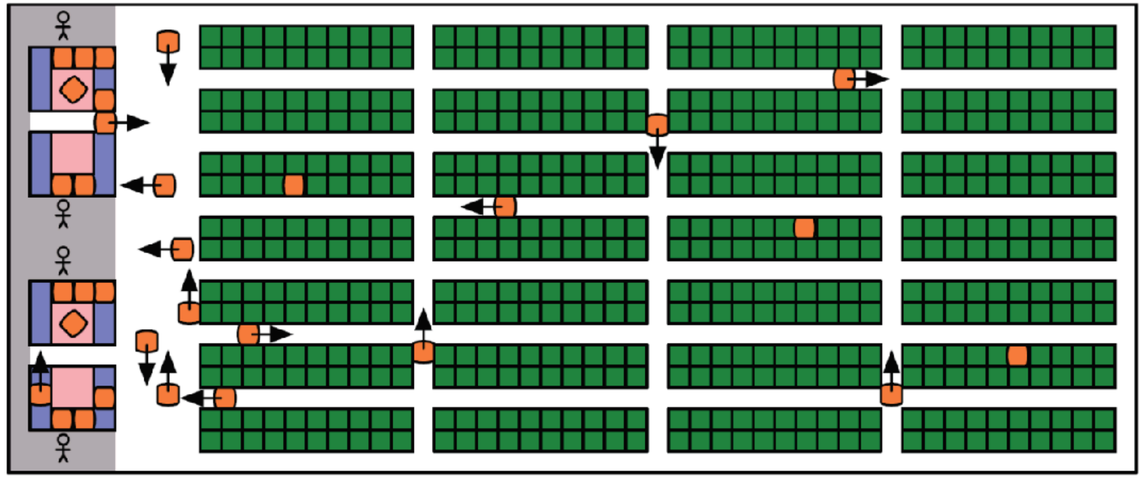
$$V^*(s) - V^{\phi}(s) \leq \frac{2}{1-\gamma}\gamma^{\Theta(V)} r_{\max}$$


warehouse-10-20-10-2-2 | FLOW | V=3 | k=825 | t=0
Source (pickup)
Destination (dropoff)
Agent → source
Agent → dest
Visibility region (≤3 cells)



The centralized algorithm in state-of-the art

A small visibility **reduces computation time by 10x**, suffers almost **no loss in throughput**



Near Optimal and Scalable Policies

Amalgam Policy

$$\lambda(s) = (\pi_z^*(s_z) : \forall z \in Z(s))$$

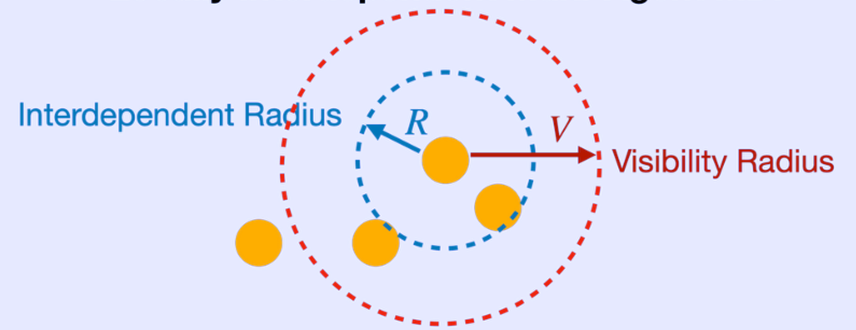
Cutoff Policy

$$\chi(s) = (\pi_z^{\mathscr{C}}((s_z, \{z\})) : \forall z \in Z(s))$$

First Step Finite Horizon Optimal Policy

$$\phi(s) = \pi^{*,0}(s)$$

Locally Interdependent Multi-Agent MDP



The policies are near optimal only when visibility is large

Amalgam Policy

$$\lambda(s) = (\pi^*_z(s_z) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\lambda}(s) \leq \frac{2}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

Cutoff Policy

$$\chi(s) = (\pi^{\mathscr{C}}_z((s_z, \{z\})) : \forall z \in Z(s))$$

Guarantee

$$V^*(s) - V^{\chi}(s) \leq \frac{2-\gamma}{(1-\gamma)^2}\gamma^{c+1} r_{\max}$$

First Step Finite Horizon Optimal Policy

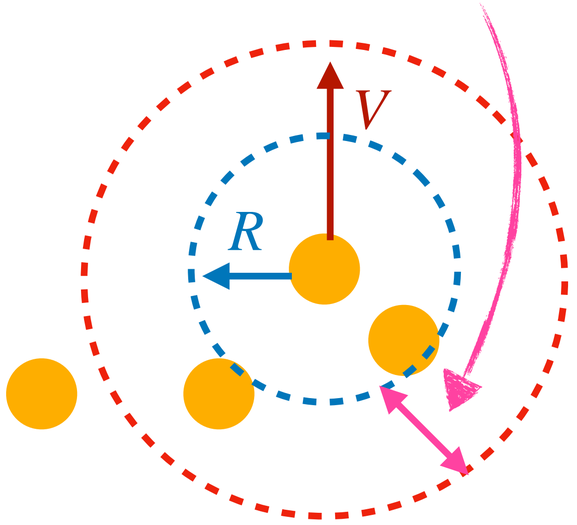
$$\phi(s) = \pi^{*,0}(s)$$

Guarantee

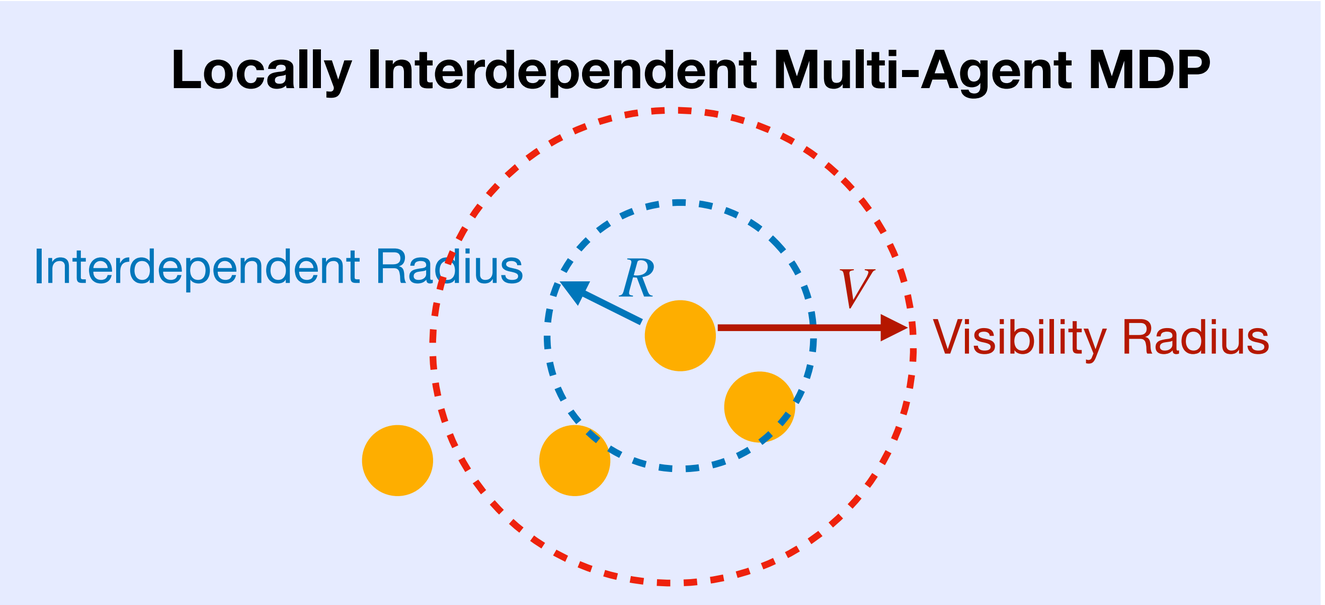
$$V^*(s) - V^{\phi}(s) \leq \frac{2}{1-\gamma}\gamma^{c+1} r_{\max}$$

Exponentially decaying in

$$c := \left\lfloor \frac{V-R}{2} \right\rfloor = \Theta(V)$$



Summary So Far



Models practial applications of decentralized agents with dynamic local dependencies!

Admits a scalable and near optimal solution!

scalable when groups are small

near optimal when the visibilities large

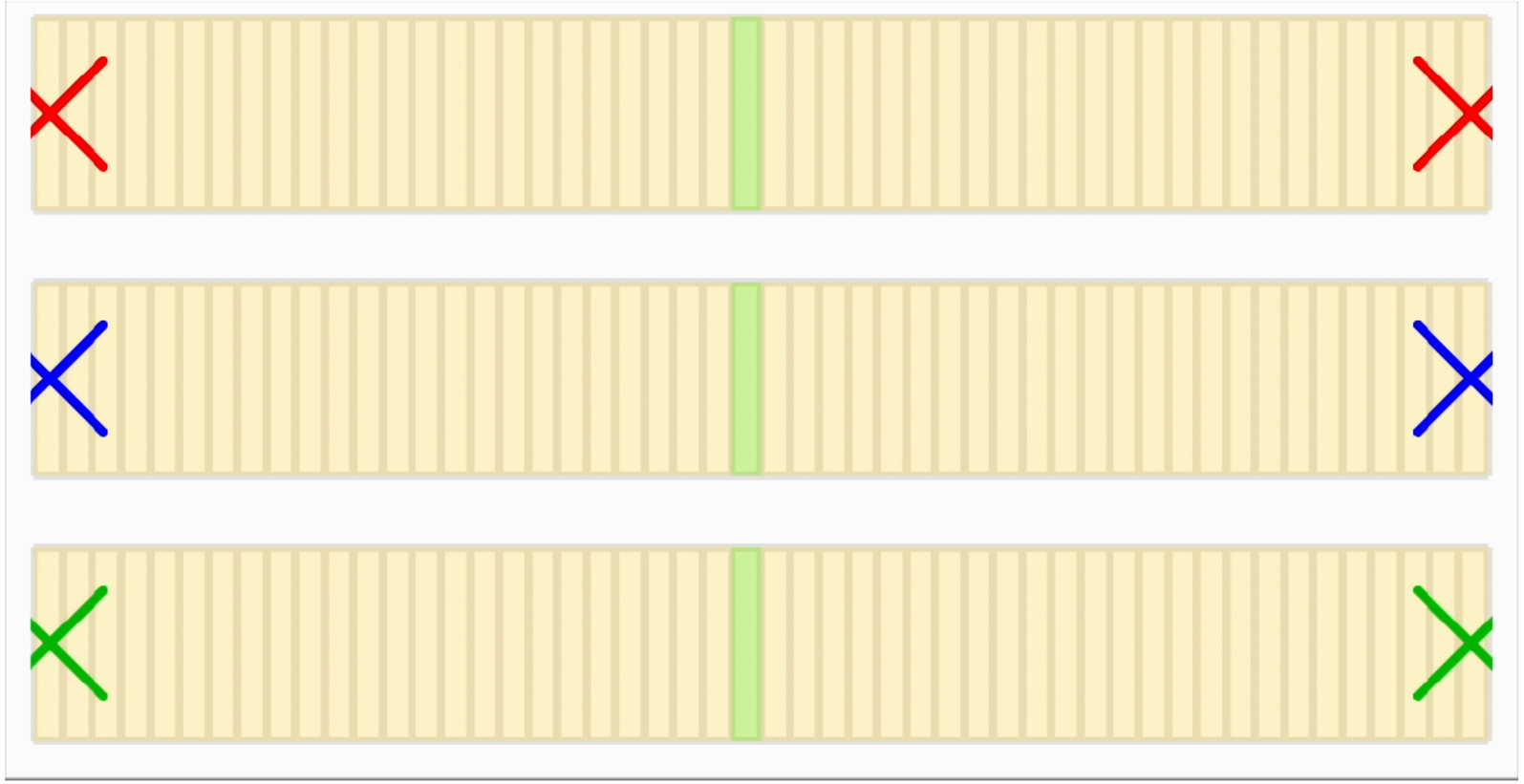
What if the visibilities are small?

When the visibility is small…

Optimal centralized policy

Amalgam with visibility 20.5

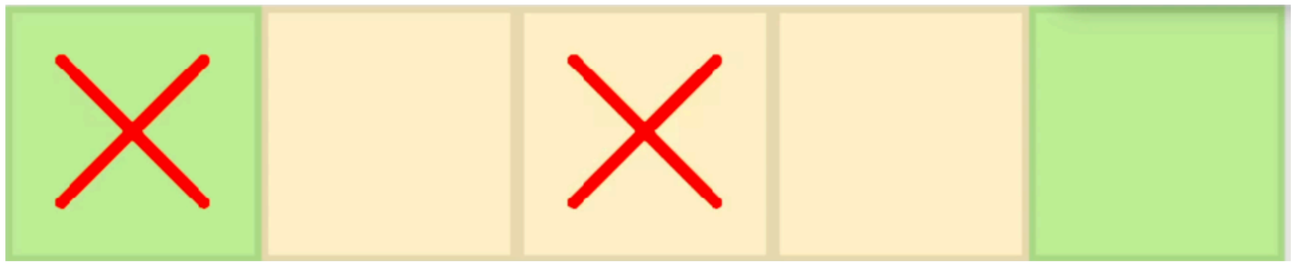
Cutoff with visibility 20.5



Note: the interdependent radius is $R = 20$

Penalty Jittering

Optimal
centralized policy

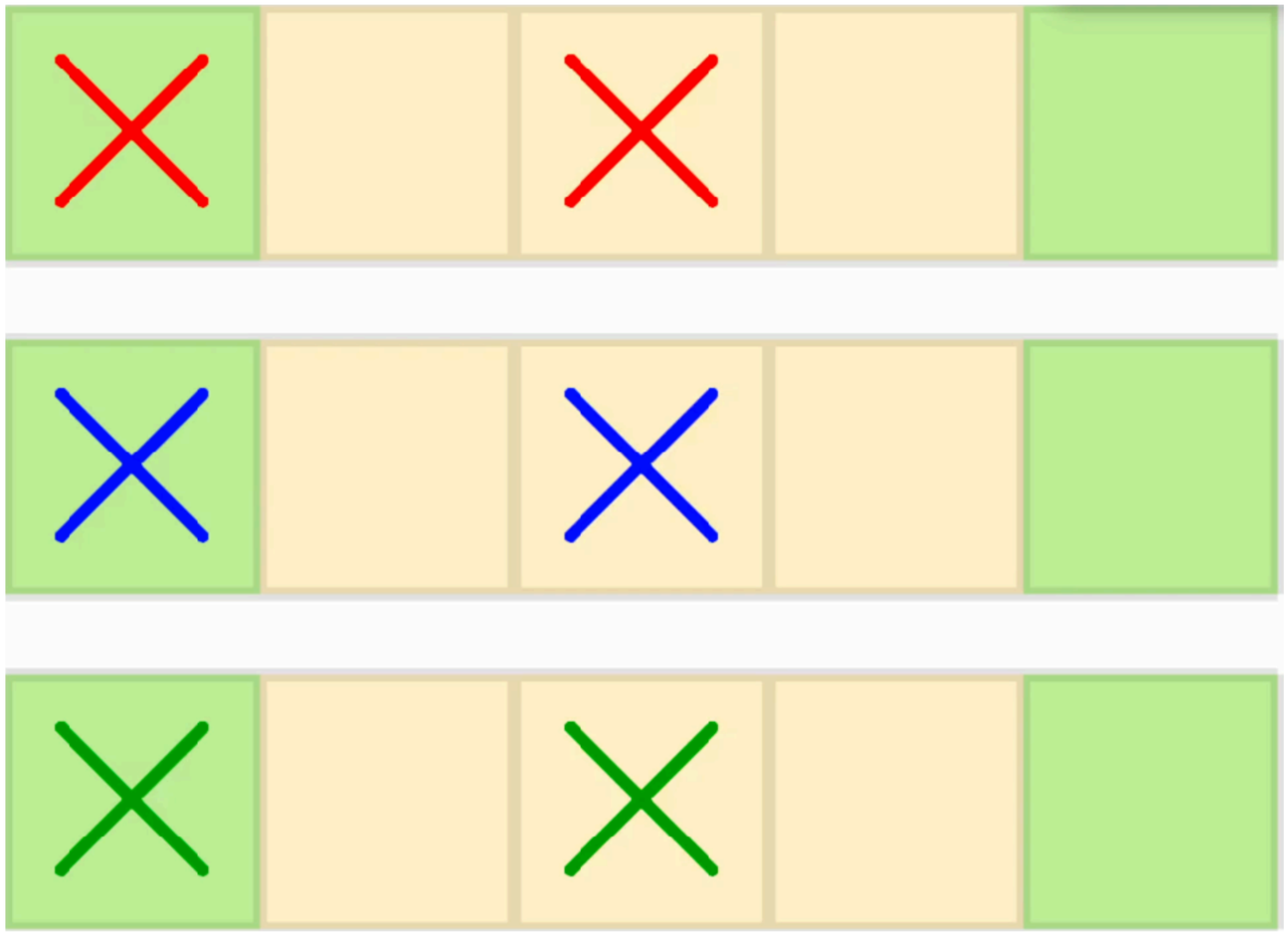


Penalty Jittering

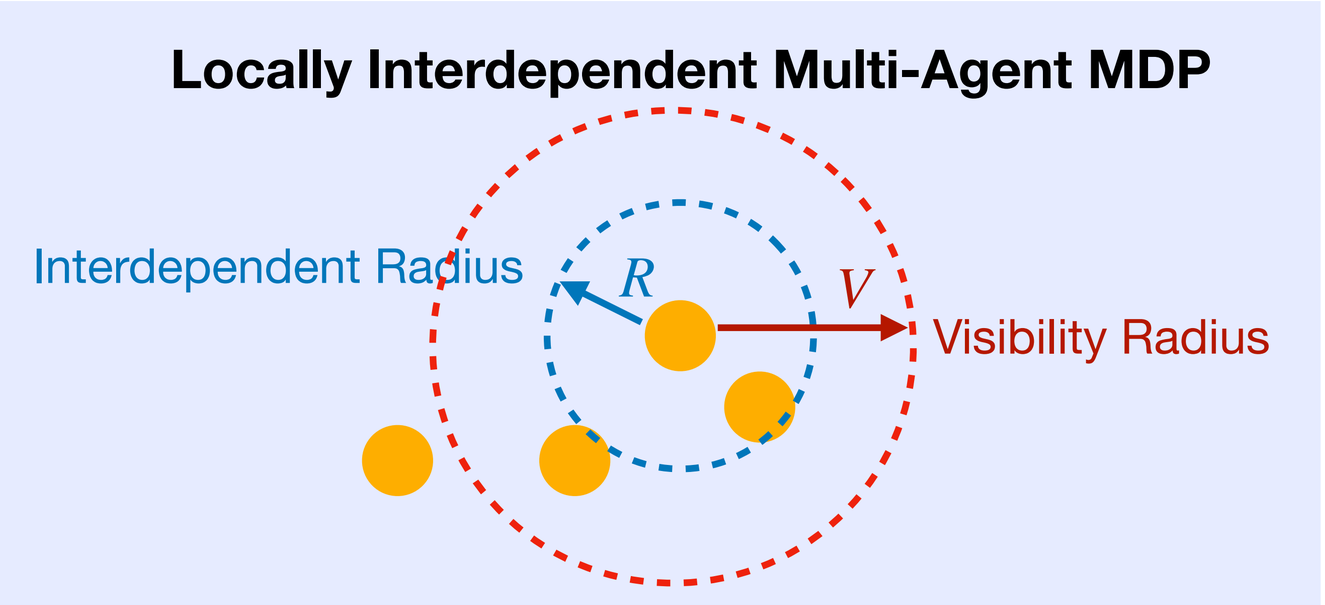
Optimal centralized policy

Amalgam with visibility 1

Cutoff with visibility 1



Summary So Far



Models practial applications of decentralized agents with dynamic local dependencies!

Admits a scalable and near optimal solution!

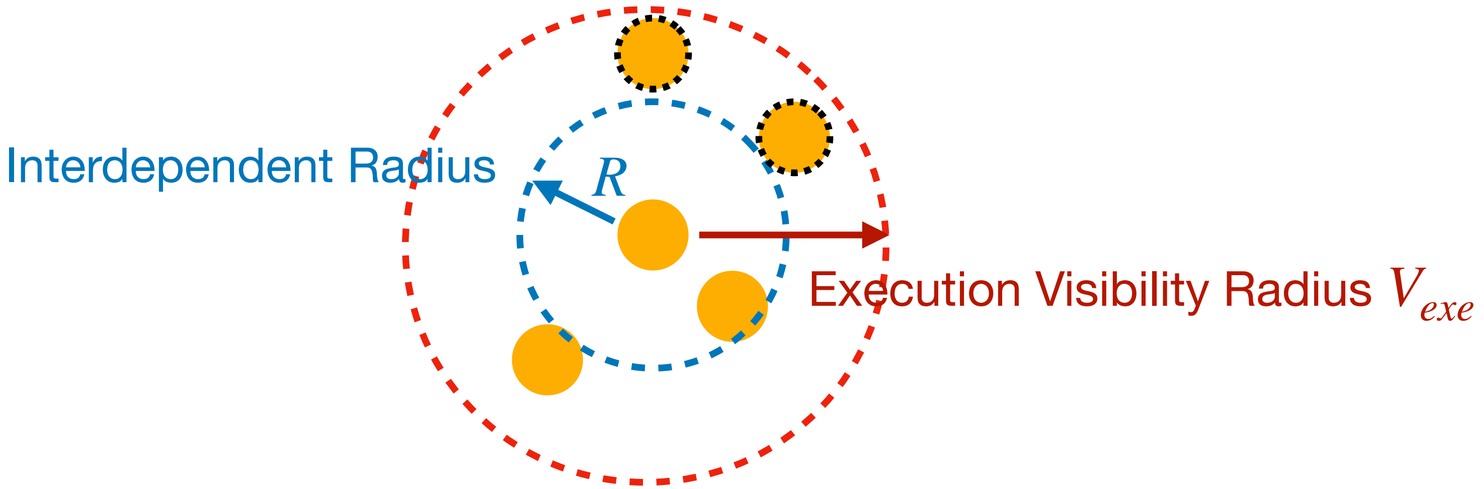
scalable when groups are small

near optimal when the visibilities large

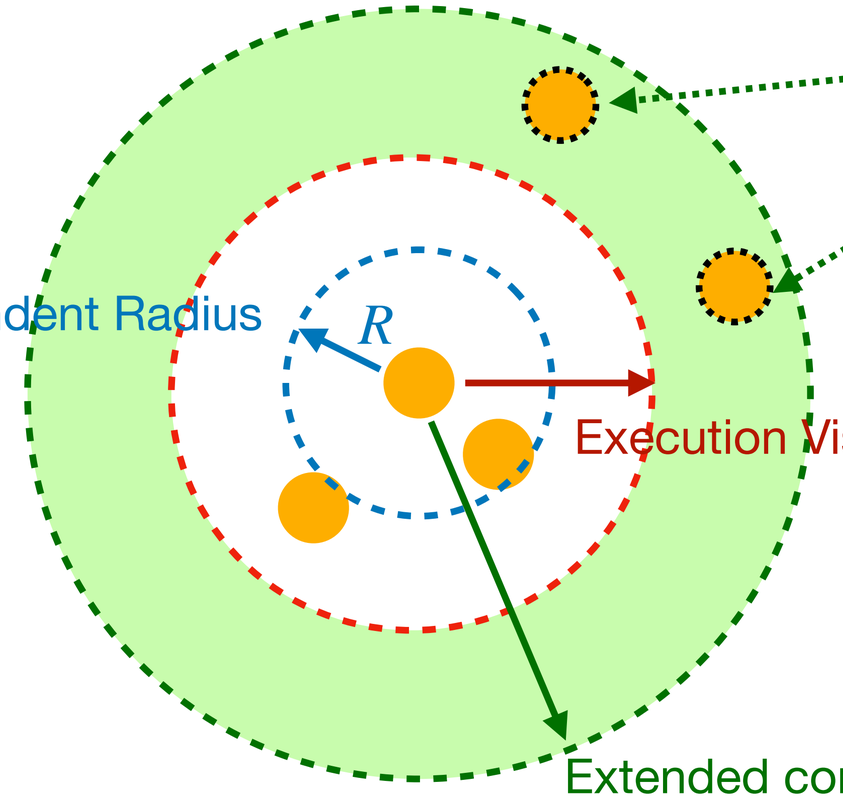
Up next

Handling small visibility

Idea: Think Beyond Visibility



Idea: Think Beyond Visibility



Extended cutoff policy

Obtain (probabilistic) estimates for out-of-view agents in the extended radius V_{comp}

Plug into the cutoff policy with extended radius V_{comp}

Think Beyond Visibility

Theorem [DeWeese and Qu 2025] Regardless of the estimation error for out-of-view agents,

$$V^*(s) - V^{\text{ext-cutoff}}(s) \leq \beta \frac{\gamma^{c-1}}{1-\gamma}$$

Exponentially decaying in $c := \left\lfloor \frac{V_{exe} - R}{2} \right\rfloor$

We maintain the same guarantee as before

even if the estimation of out-of-view agents are arbitrarily bad

Think Beyond Visibility

If the out-of-view estimates are good, we can get much better bounds

Theorem [DeWeese and Qu 2025]. Suppose:

- The transition is deterministic;
- Given an initial state s where all agents are within view.

When using a *Simple Memory Based Out-of-view Estimator*

$$V^{\text{ext-cutoff}}(s) \to V^{*}(s) \text{ as } V_{comp} \to \infty$$

Simple Memory Based Out-of-view Estimator works by

- For agents that go out of view, track where it was last seen
- Simulate its trajectory from last known position

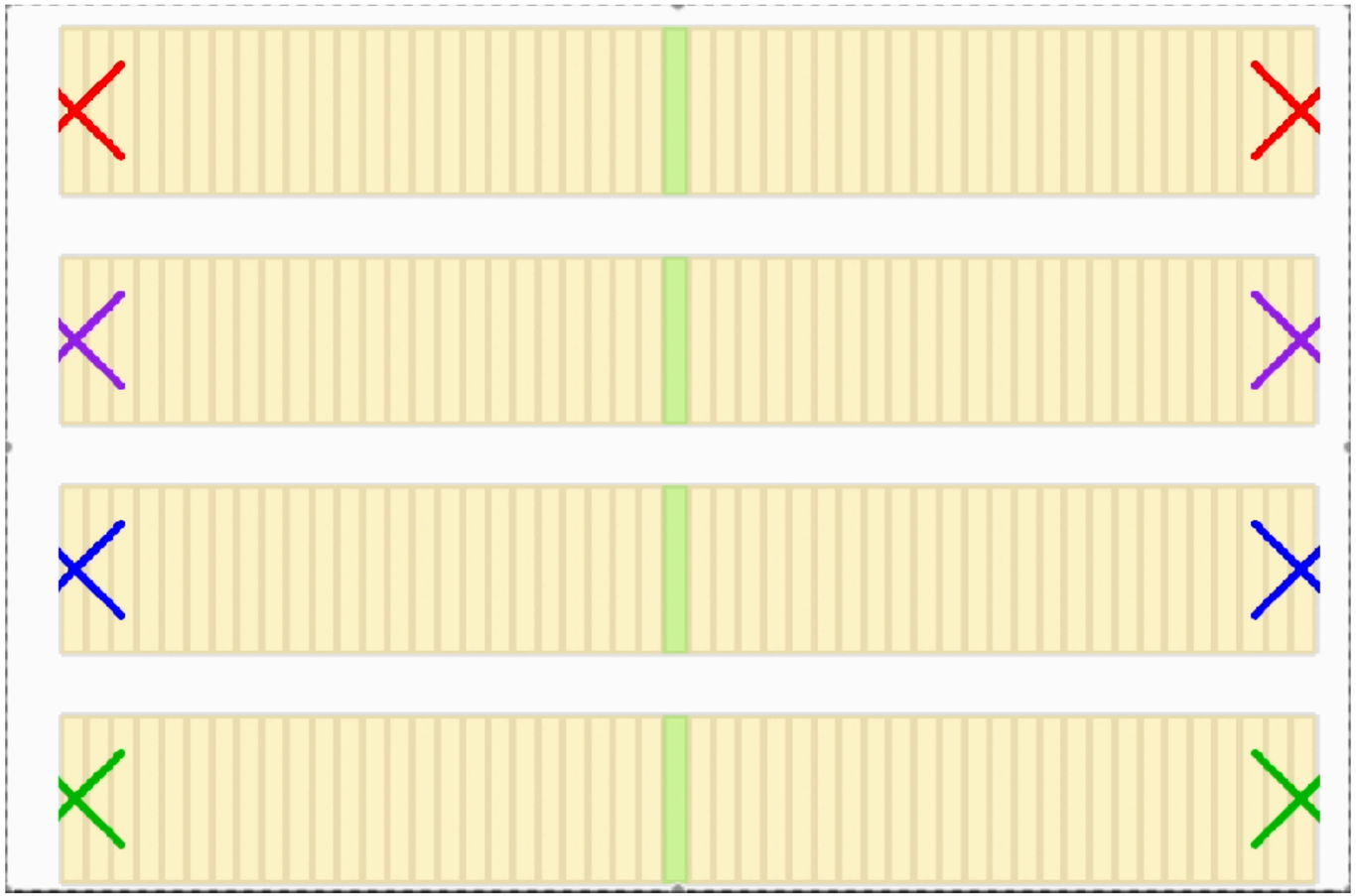
Penalty Jittering Revisited

Optimal centralized

Extended cutoff with execution visibility 20.5, computational visibility 30

Amalgam with visibility 20.5

Cutoff with visibility 20.5

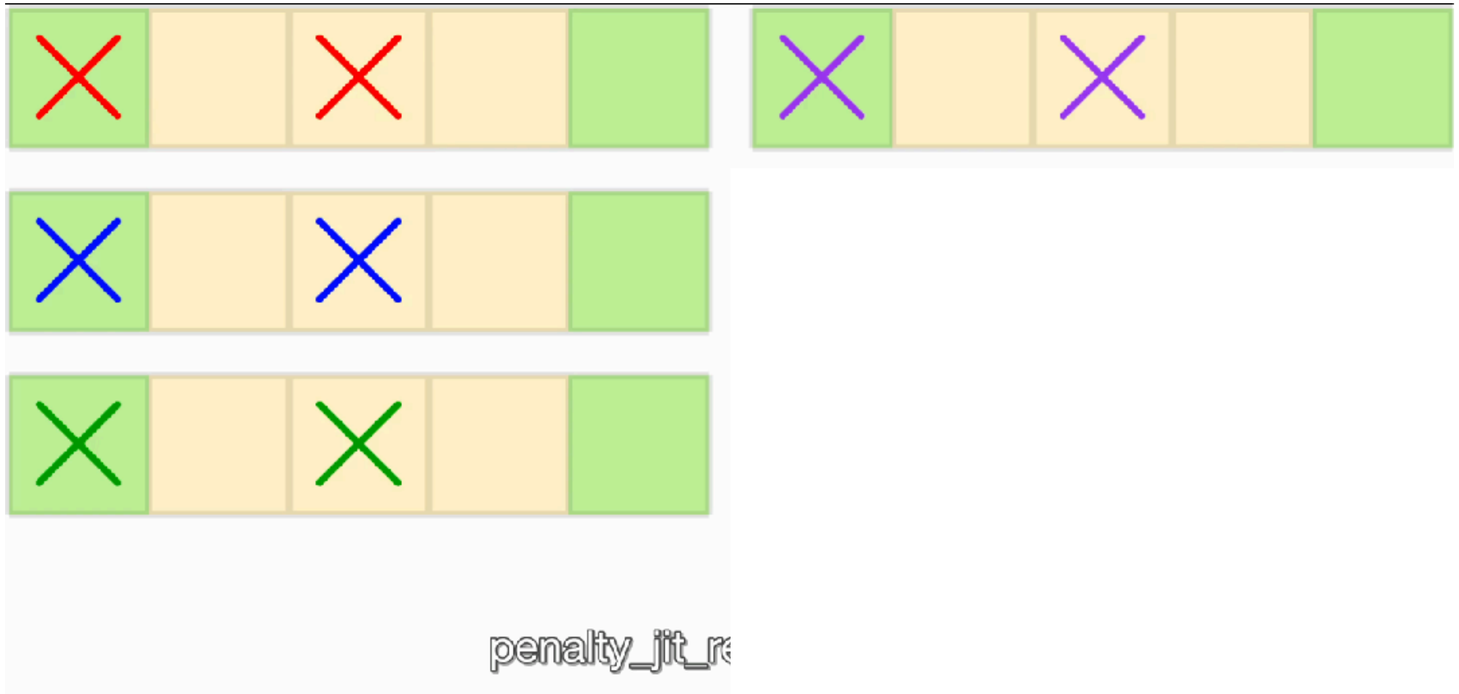


Penalty Jittering Revisited

Optimal centralized

Amalgam with visibility 1

Cutoff with visibility 1



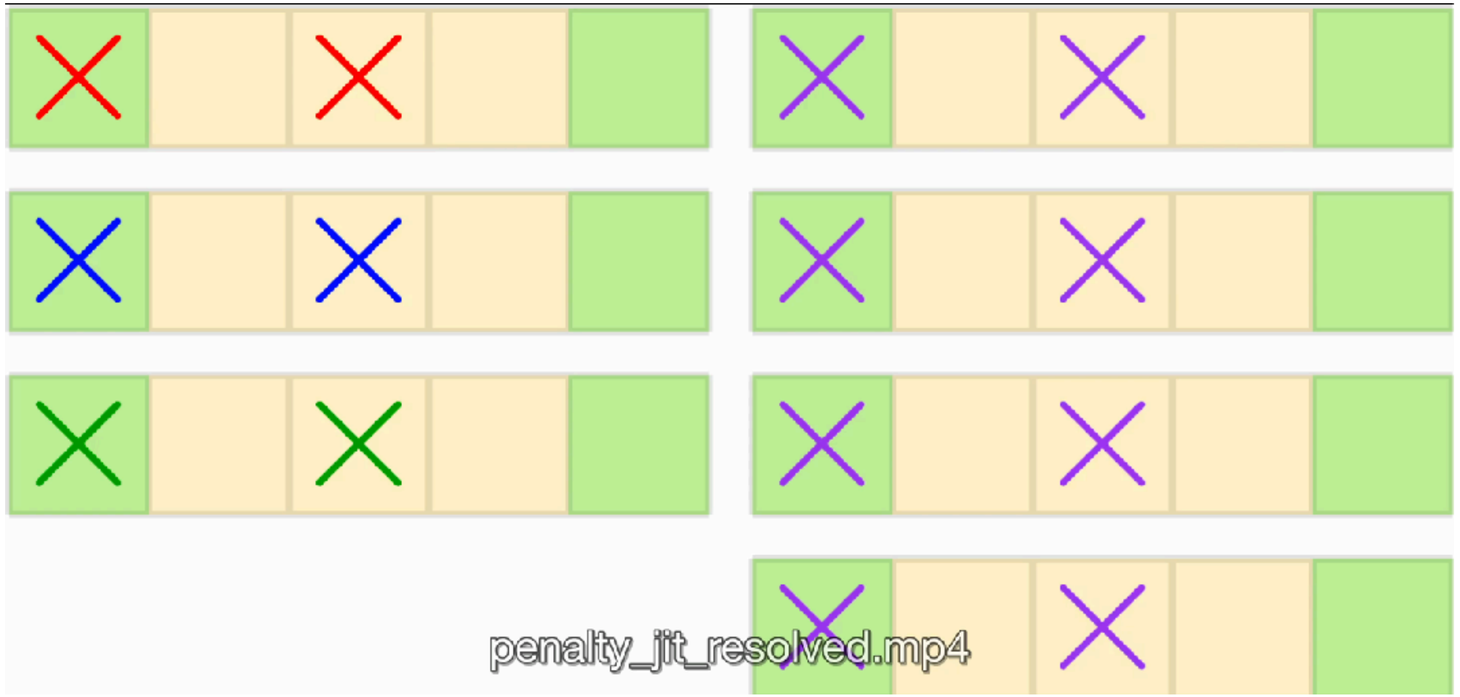
Extended cutoff with execution visibility 1, computational visibility 1

Penalty Jittering Revisited

Optimal centralized

Amalgam with visibility 1

Cutoff with visibility 1



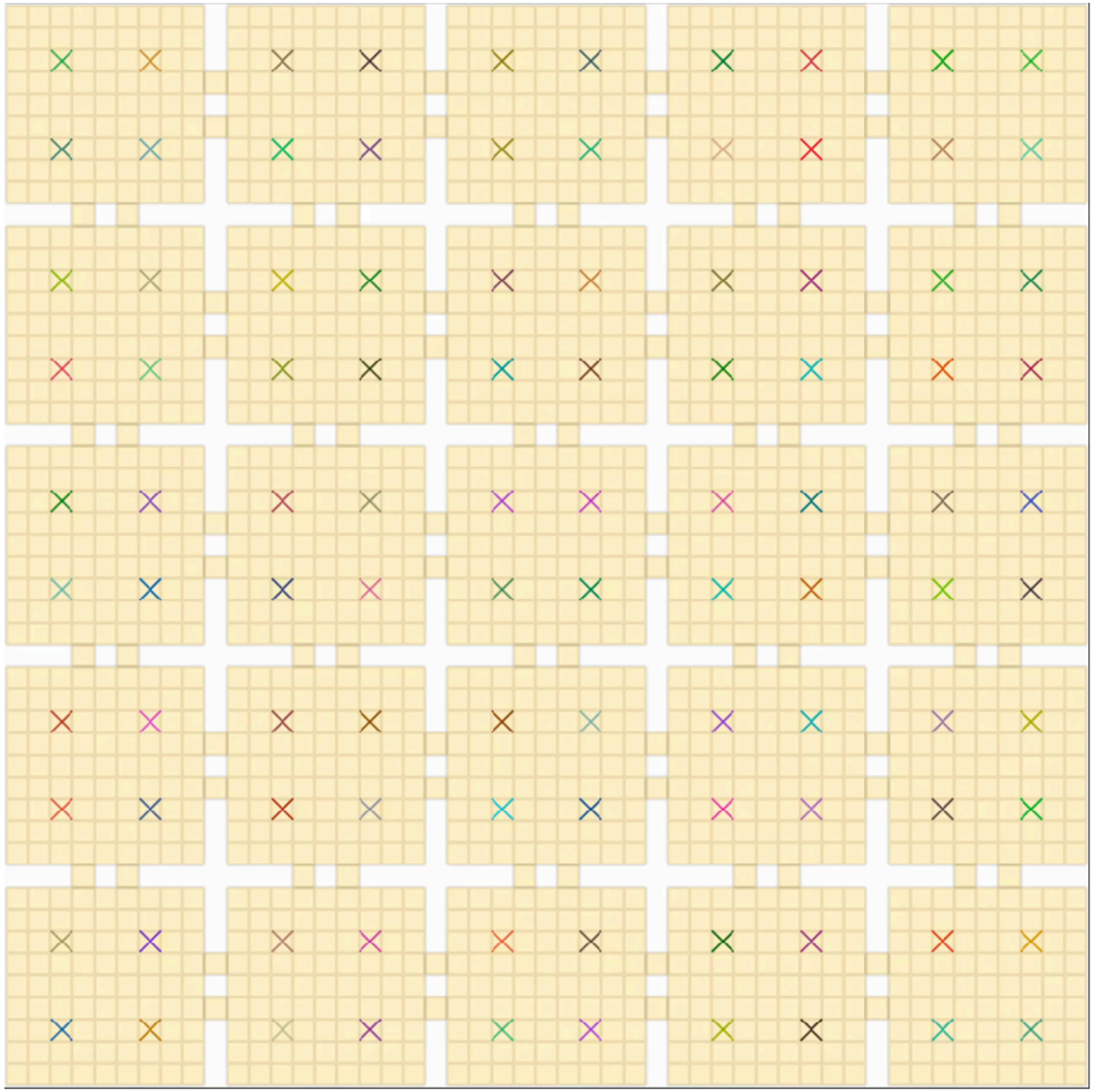
Extended cutoff with execution visibility 1, computational visibility 1

computational visibility 2

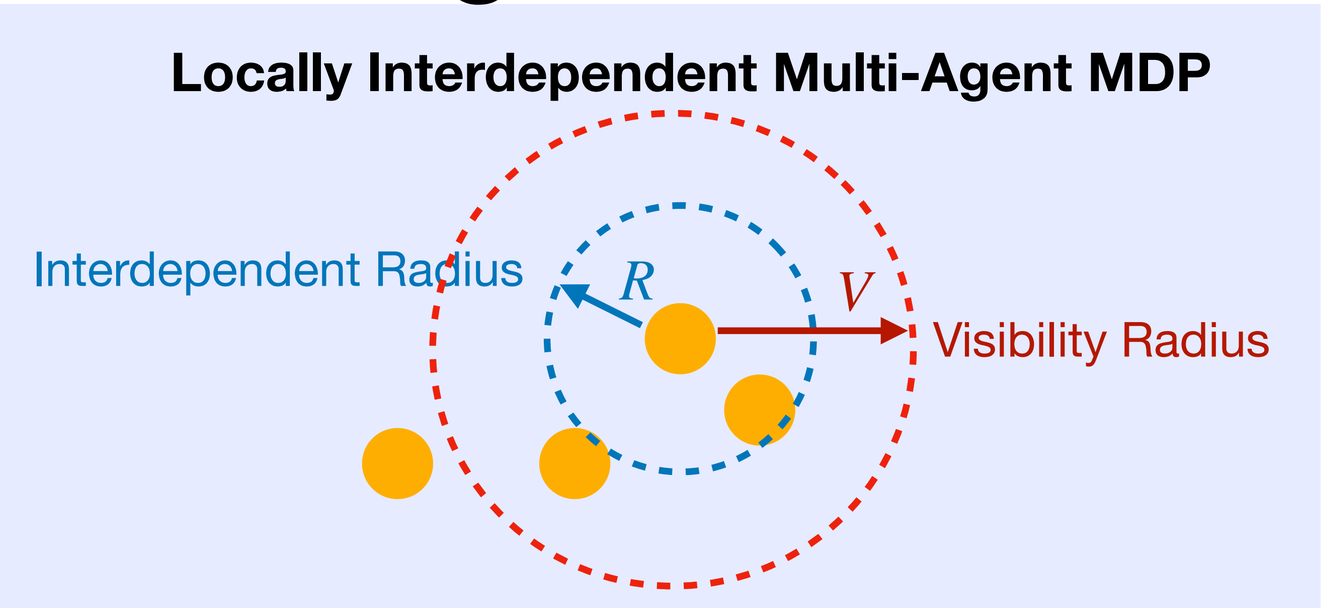
computational visibility 3

computational visibility 4

Large Scale Simulation



Take home message



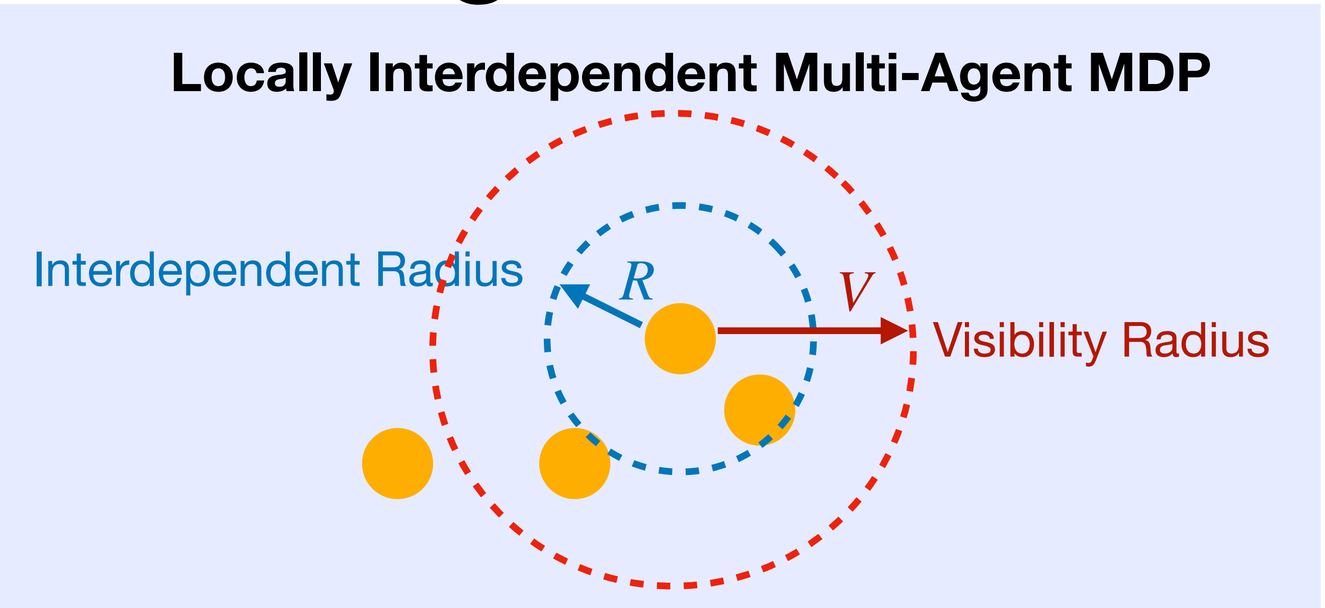
Models practial applications of decentralized agents with dynamic local dependencies!

Admits a scalable and near optimal solution?

scalable when groups are small. Larger groups handled by heuristics.

near optimal when the visibilities large. Small visibility? Extended cutoff

Take home message



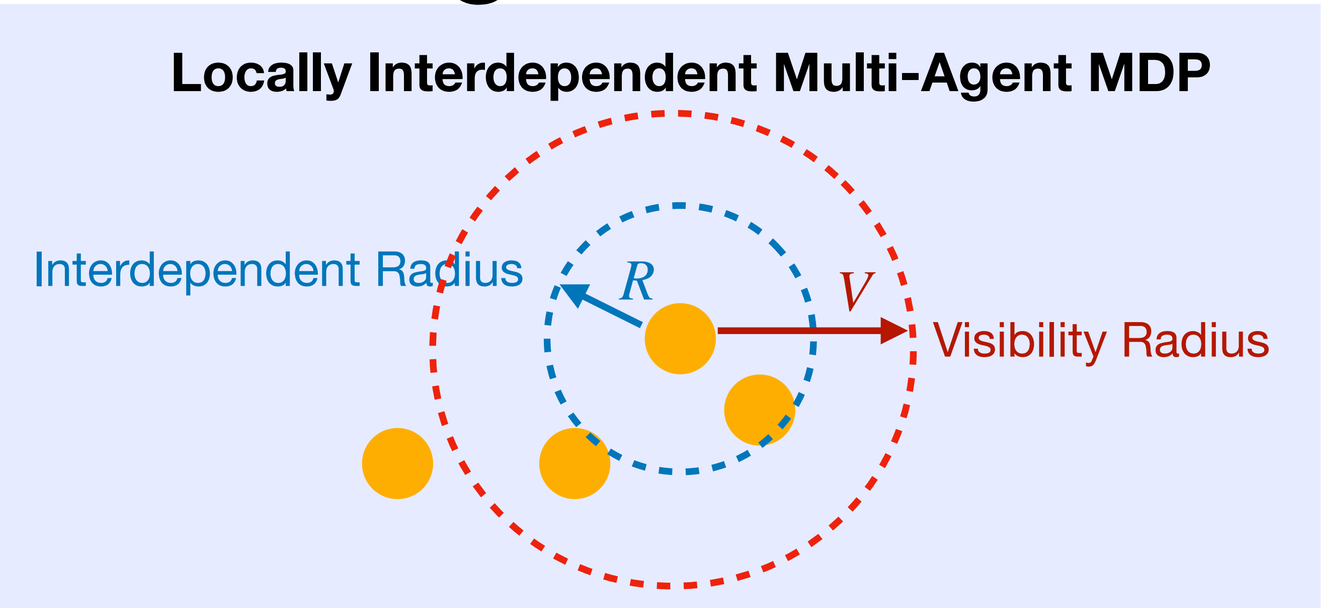
Future directions

Better ways to handle large groups

More systematic studies on out-of-view estimation

Develop RL algorithms based on LIMDP

Take home message



Alex DeWeese, Guannan Qu, Locally Interdependent Multi-Agent MDP: Theoretical Framework for Decentralized Agents with Dynamic Dependencies, ICML 2024

Alex DeWeese, Guannan Qu, Thinking Beyond Visibility: A Near-Optimal Policy Framework for Locally Interdependent Multi-Agent MDPs, arXiv preprint arXiv:2506.04215, 2025.