Multi-Robot Coordination Induced in Hazardous Environments through an Adversarial Graph-Traversal Game

James Berneburg, Xuan Wang, Xuesu Xiao, and Daigo Shishika

***Abstract*— This paper presents a game theoretic formulation of a graph traversal problem, with applications to robots moving in hazardous environments, particularly in the presence of an adversary. The blue team of robots moves in an environment modeled by a time-varying graph, attempting to reach some goal with minimum cost, while the red team controls how the graph changes in time to maximize the cost. The problem is formulated as a stochastic game, so that Nash equilibrium strategies can be computed numerically. Bounds are provided for the game value, with a guarantee that it solves the original problem. Numerical simulations demonstrate the results and the effectiveness of this method, particularly showing the benefit of mixing actions for both players. Additionally, we observe beneficial coordinated behavior in cases with multiple blue agents, where they split up and/or synchronize to traverse risky edges, so that at least a subset of the team takes the cheaper path.**

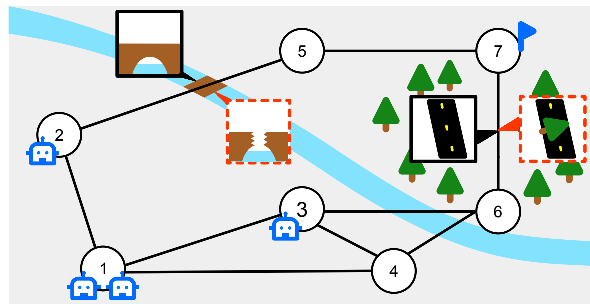


Fig. 1. An example scenario motivating our problem. The team of robots intend to reach their destination node 7 with minimal risk, but the terrain is uncertain and time-varying, with the potential for the bridge to be destroyed or trees to be downed on the road. Formulating this as a game can capture the worst case scenario.

I. Introduction

Consider a scenario where multiple robots must traverse difficult terrain to reach a goal, as shown in Figure 1. The robots wish to reach the goal with minimum risk, and so must plan a path through the environment. Such environments are modeled as graphs which indicate the different paths robots may take to reach the goal, and the difficulty of the terrain may be captured by the weights of the graph. For constant graphs, there exist algorithms for finding the shortest path, which corresponds to minimizing risk. However, in cases where the environment may change with time, we must consider finding shortest path through time-varying graphs. As this is a one-sided optimization, there exist many works which solve this problem. For example, [1] proposes an algorithm for finding the shortest time path through graphs with time-varying delays on edges. Other works, like [2] and [3], consider both time-varying costs and delays on edges, and minimize one under a constraint on the other. One work which specifically considers planning for a team of robots on a graph is [4], which considers that the edge costs are a function of the positions of other robots which can provide support, and formulates an optimization problem to minimize the cost to reach a goal.

However, such solutions may be unsuitable when the environment may change in highly unpredictable ways because they rely on knowledge of how the graph changes in time. One method to handle such uncertainty is to consider an adversarial scenario, which can model the worst-case scenario in an environment which changes due to natural causes, or the case where the environment changes due to an actual adversary. To analyze such scenarios, we turn to game theory for results, where are many works which analyze adversarial interactions on graphs.

A similar problem which uses graphs and frequently employs a game theoretic approach is attack graph problems [5], [6]. These are used to model cyber-security problems where the graph represents the possible avenues of attack, which the defenders have some control over. For example, [5] considers that the defender can add "honeypots" to the network, modifying the graph nodes to be better defended in a Stackelberg game formulation. Closer to our problem, [6] considers that both players incorporate feedback into heuristic strategies to approximate the Nash equilibria. However, this sort of formulation does not suit our problem, because rather than the attacker moving from one node to another in the graph, the attacker gains permanent control over the resources represented by nodes. Another work which also considers a network security problem over graphs is [7], which considers an intrusion detection problem where an attacker can compromise nodes and formulates it as a stochastic game [8].

One particular type of game that is more closely related to our problem is barrier coverage, which considers how a defender may place sensors to detect intruding robots, which move through the environment in an attempt to avoid detection [9], [10]. Here, rather than the cost for the moving robots indicating the hazard level of the environment, it indicates the probability of detection. While these two works analyze this problem from a game theoretic perspective and provide Nash equilibrium results, they are not directly applicable to our problem, because they cannot consider real-time feedback in the defender's sensor placement.

The authors are with the Department of Mechanical Engineering, George Mason University, Fairfax, VA 22030, USA, {jbernebu,xwang64,xiao,dshishik}@gmu.edu

Therefore, this paper's contributions are as follows. We formulate the adversarial graph traversal problem as a novel stochastic game, which allows for numerical computation of a mixed Nash equilibrium. We provide theoretical results by bounding the game value with security strategies for both players, and we guarantee that the blue player reaches the goal almost surely under its Nash equilibrium policy. Finally, we demonstrate the preceding theoretical results and show the advantage of mixed strategies and coordinated behavior for multiple robots, by numerically solving illustrative examples and through a statistical analysis of games on randomly generated graphs.

II. Problem Formulation

We consider a two-player game where the blue player moves its robots through a weighted digraph $G(t) \in \mathcal{G}$, where $\mathcal{G}$ is the set of known possible digraphs, while the red player controls the graph $G(t)$. We assume that $G(t) = (\mathcal{V}, \mathcal{E}, W(t))$, where $\mathcal{V} = \{1, 2, \dots N\}$ is the set of nodes for $N \in \mathbb{Z}_{>0}$, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of edges, and $W(t) \in \mathbb{R}^{N \times N}$ is the time-varying weighted adjacency matrix, where $W_{i,j}(t) \geq 0$ if $(i,j) \in \mathcal{E}$ and $W_{i,j}(t) = 0$ otherwise, for $i, j \in \mathcal{V}$. Specifically, this indicates that the set of nodes and the set of edges are constant with time, and only the weights of the edges are time-varying. We consider that there is a finite number $K \in \mathbb{Z}_{>0}$ of possible weighted adjacency matrices, so that $W(t) \in \mathcal{W}$, where $\mathcal{W} = \{W^1, W^2, \dots, W^K\}$. Then, the set of all possible digraphs is $\mathcal{G} = \{G^1, G^2, \dots, G^K\}$, where $G^k = (\mathcal{V}, \mathcal{E}, W^k)$ for $k = 1, 2, \dots, K$.

We assume that there are M blue robots, where $M \in \mathbb{Z}_{>0}$, and that each robot m at time t has position $p_m(t) \in \mathcal{V}$. We collect all robot positions together into the vector $p(t) = [p_1, p_2, \dots, p_M]^T \in \mathcal{V}^M$. Their movement is constrained by the graph, so that if $p_m(t) = i$, then $p_m(t+1) \in \overrightarrow{\mathcal{N}}_i$, where $\overrightarrow{\mathcal{N}}_i$ is the set of out neighbors of node i. Without loss of generality, node N is the destination node, and the game ends when $p(t) = \mathbf{1}_M N$, so that all blue robots have reached the destination node. To ensure this is possible to achieve for any initial conditions, we assume that there exists a path from each node in G to node N. Additionally, we assume that there exists a self-loop on node N with an edge weight of 0, so that any blue robots that reach the goal can remain there without incurring any additional costs.

The red player's action is to select the graph at the next time step, so that $a_{\text{red}}(t) = k \in \{1, 2, \dots, K\}$ and $G(t+1) = G^k$. Specifically, we consider that the red player's available actions at time t depend on the current graph $G(t)$, being determined by the red player's action graph, $\mathcal{G}^{\text{red}} = (\mathcal{V}^{\text{red}}, \mathcal{E}^{\text{red}})$, where $\mathcal{V}^{\text{red}} = \{1, 2, \dots, K\}$ are the nodes of this graph, which each correspond to a position graph G^k, and $\mathcal{E}^{\text{red}} \subset \mathcal{V}^{\text{red}} \times \mathcal{V}^{\text{red}}$ is the set of edges. We assume that the action graph is unweighted, directed, and weakly connected, and that every node has a self loop. The red player's action is then determined by the edges of $\mathcal{G}^{\text{red}}$, so that $a_{\text{red}}(t) \in \overrightarrow{\mathcal{N}}_k^{\text{red}}$, where $\overrightarrow{\mathcal{N}}_k^{\text{red}}$ is the set of out neighbors of k in $\mathcal{G}^{\text{red}}$, and $G^k = G(t)$ is the graph at time t. This allows us to model cases where the red player may require multiple time steps to change the environment and some changes may be irreversible. See Figure 2 for a representation of $\mathcal{G}^{\text{red}}$.

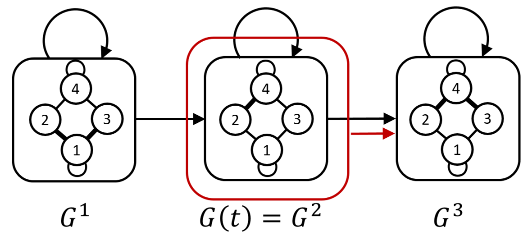


Fig. 2. This shows an example of red's action graph, where each square node corresponds to one of the position graphs for the blue player, shown inside. The outlined node indicates $G(t)$ and the arrow indicates the action of the red player to select $G(t+1) = G^3$.

The red player consumes some limited resource, which we refer to as *ammo*, by changing the position graph. More specifically, we consider that the red player has ammo $\alpha(t)$ which is initialized to $\overline{\alpha} \in \mathbb{Z}_{\geq 0}$, so that $\alpha(0) = \overline{\alpha}$. Then, if $\alpha(t) > 0$, $a_{\text{red}}(t) \in \overrightarrow{\mathcal{N}}_k^{\text{red}}$, and if $\alpha(t) = 0$, $a_{\text{red}}(t) = k$ where $G(t) = G^k$. Finally, this limited resource is consumed each time the graph is changed, so that

$$\alpha(t+1) = \begin{cases} \alpha(t) - 1, & \text{for } G(t+1) \neq G(t) \\ \alpha(t), & \text{for } G(t+1) = G(t) \end{cases}. \quad (1)$$

Now, we can write the entire game state, containing the positions of the blue robots, the current graph, and the remaining amount of red team's ammo, as $S(t) = (p(t), \mathcal{G}(t), \alpha(t)) \in \mathcal{S} \triangleq \mathcal{V}^M \times \mathcal{G} \times \{0, 1, \dots, \overline{\alpha}\}$.

At each time step, the blue player incurs a stage cost of

$$C\left(S, a_{\text{blue}}(t)\right) = \sum_{m=1}^{M} W_{p_m, p_m^+}(t), \quad (2)$$

where, with some abuse of notation, p_m is the position of robot m at time t and p_m^+ is its position at time $t+1$, for $m \in \{1, 2, \dots M\}$. Recall that the red player's action at the previous time, $a_{\text{red}}(t-1)$, indirectly affects this cost through the graph $G(t)$. The total cost incurred by the blue player is the sum of the weights for the edges traversed by all robots, and the goal of the blue player is to minimize the cost incurred to reach the goal. Due to the presence of the antagonistic red player in this zero-sum game, this cannot simply be solved as a shortest path problem.

The set of valid actions for the blue player is $A_{\text{blue}}(S) = \overrightarrow{\mathcal{N}}_{p_1}^{\text{out}} \times \overrightarrow{\mathcal{N}}_{p_2}^{\text{out}} \times \cdots \times \overrightarrow{\mathcal{N}}_{p_M}^{\text{out}}$ and the set of valid actions for the red player is

$$A_{\text{red}}\big((p, G^k, \alpha)\big) = \begin{cases} \overrightarrow{\mathcal{N}}_k^{\text{red}}, & \text{if } \alpha(t) > 0 \\ k, & \text{if } \alpha(t) = 0 \end{cases}. \quad (3)$$

The game dynamics, given the actions of the players, can be described as

$$\begin{aligned} p(t+1) &= a_{\text{blue}}(t) \in A_{\text{blue}}(S) \\ G(t+1) &= G^{a_{\text{red}}(t)}, a_{\text{red}}(t) \in A_{\text{red}}(S) \\ \alpha(t+1) &= \begin{cases} \alpha(t) - 1, & \text{for } G^{a_{\text{red}}(t)} \neq G(t) \\ \alpha(t), & \text{for } G^{a_{\text{red}}(t)} = G(t) \end{cases}. \end{aligned} \quad (4)$$

To facilitate describing mixed strategies, we formulate a stochastic game, where the state dynamics are a Markov chain and the transition probabilities are influenced by the actions of both players [8]. The Markov chain is $(\mathcal{S}, A_{\text{blue}}, A_{\text{red}}, P)$, where we consider deterministic dynamics, such that the transition probabilities $P : \mathcal{S} \times \mathcal{S} \times \mathcal{V}^M \times \{1, 2, \ldots, K\} \to \{0, 1\}$ are in accordance with (4).

We define mixed policies for the blue player and the red player as mappings $\pi_{\text{blue}} : S \to \Delta A_{\text{blue}}(s)$, where $\Delta A_{\text{blue}}(s) \subset [0, 1]^{|A_{\text{blue}}(s)|}$ is the simplex over $A_{\text{blue}}(s)$, and $\pi_{\text{red}} : S \to \Delta A_{\text{red}}(s)$, respectively, for $s \in \mathcal{S}$. Now, we can formally write the expected cost of the game as

$$J(S(0), \pi_{\text{blue}}, \pi_{\text{red}}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t C\left(S(t), a_{\text{blue}}(t)\right)\right], \quad (5)$$

where $\gamma \in (0, 1]$ is a discount factor. Note that, due to the presence of a zero cost self-loop on the goal node, no further costs will be accrued once all blue robots reach the goal. Recalling that we consider a zero-sum game, we seek a pair of strategies $\pi^*_{\text{blue}}(\cdot), \pi^*_{\text{red}}(\cdot)$ which corresponds to a Nash equilibrium, so that

$$\begin{aligned} J(s, \pi^*_{\text{blue}}, \pi^*_{\text{red}}) &\leq J(s, \pi_{\text{blue}}, \pi^*_{\text{red}}) \\ J(s, \pi^*_{\text{blue}}, \pi^*_{\text{red}}) &\geq J(s, \pi^*_{\text{blue}}, \pi_{\text{red}}), \end{aligned} \quad (6)$$

for all $s \in \mathcal{S}$ and all valid mixed strategies $\pi_{\text{blue}}, \pi_{\text{red}}$. Although, in our original problem, $\gamma = 1$, having $\gamma \in (0, 1)$ allows us to use the results in [8] to guarantee that such strategies exist and that the outcome associated with them is unique, for our zero-sum stochastic game. This allows us to define the value function for the equilibrium strategies as

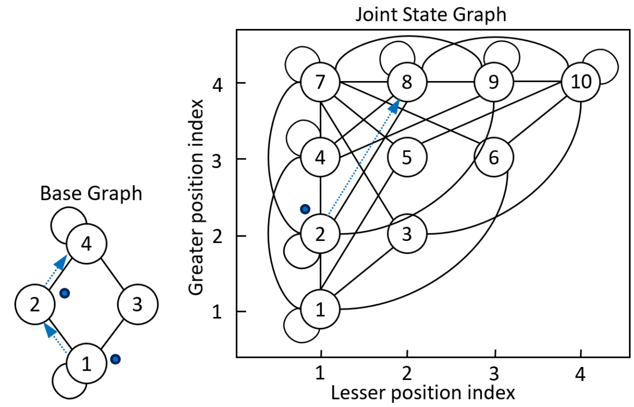
$$V(s) = J(s, \pi^*_{\text{blue}}, \pi^*_{\text{red}}), \quad (7)$$

for $s \in \mathcal{S}$, which simply gives the (discounted) value of the game outcome, provided both players use the equilibrium strategies. Note that γ is not a parameter of the original problem we want to solve, but rather it is chosen to get an approximate solution, because the presence of a discount factor allows us to make guarantees that Nash equilibria of the game exist and have the same value [11], [8].

III. Theoretical Analysis

A. Handling Multiple Blue Robots

Thus far, we have considered that there are $M \geq 1$ blue robots moving in graph $G(t)$. In this section, we show how we can consider only a single blue robot, without losing generality by constructing a joint state graph [12]. To handle this case, instead of considering multiple robots moving on the graph $G(t)$, we can equivalently consider a single robot moving on a modified joint graph $\mathcal{G}'(t) = (\mathcal{V}', \mathcal{E}', \mathcal{W}'(t))$, where each node in $\mathcal{G}'(t)$ corresponds to a position vector $p(t)$. Each weighted adjacency matrix $W(t) \in \mathcal{W}$ will have a corresponding joint weighted adjacency matrix $\mathcal{W}'(t) \in \overline{\mathcal{W}}'$, and so $|\overline{\mathcal{W}}'| = |\mathcal{W}|$. Similarly, we have a joint graph set $\overline{\mathcal{G}}' = \{\mathcal{G}'^1, \ldots, \mathcal{G}'^K\}$, where $\mathcal{G}'^k = (\mathcal{V}', \mathcal{E}', \mathcal{W}'^k)$, for $k = 1, \ldots, K$. Instead of needing a position vector $p(t) \in \mathcal{V}^M$



(a) (b)

Fig. 3. (a) shows a simple example graph, and (b) shows the structure of its joint state graph with two blue robots. The position of each node of the JSG indicates what nodes in the original graph it corresponds to. We have assumed that robots are indistinguishable. The markers show a position of two robots in the base graph and the corresponding position in the JSG, and the arrows show an equivalent action in both graphs.

which gives the node locations of the individual robots on the graph $G(t)$, we only need a scalar joint position $p'(t) \in \mathcal{V}'$.

To construct this, we assign each position vector $p(t) \in \mathcal{V}^M$ to a node ℓ in $\mathcal{G}'$, by defining a one-to-one mapping $\mathcal{T} : \mathcal{V}^M \to \mathcal{V}'$, where $\mathcal{V}' \triangleq \{1, 2, \ldots, N^M\}$ and $\mathcal{T}(\mathbf{1}_M N) = N^M$ which is the goal node in the new graph. Then, for each node $\ell \in \mathcal{V}'$, we determine its out-neighbors by finding the next position for each action of the blue player. More specifically, for $\ell \in \mathcal{V}'$, we have a corresponding position vector $p(t) = [i, \ldots, j]^T$ with $i, \ldots, j \in \mathcal{V}$, and the next position is $p(t+1) = [i^+, \ldots, j^+]^T$, where $i^+ \in \overrightarrow{\mathcal{N}}_i, \ldots, j^+ \in \overrightarrow{\mathcal{N}}_j$. For each $p^+ = p(t+1)$, we have an $\ell^+ \in \mathcal{V}'$, which means that (ℓ, ℓ^+) belongs to the set $\mathcal{E}'$. Next, for each $k = 1, 2, \ldots, K$ we determine the corresponding weight in $\mathcal{W}'^k$ for that edge. For each edge $(\ell, \ell^+) \in \mathcal{E}'$, we set

$$\mathcal{W}'^k_{\ell,\ell^+} = W^k_{i,i^+} + \cdots + W^k_{j,j^+}, \quad (8)$$

where $\ell = \mathcal{T}([i, \ldots, j]^T)$ and $\ell^+ = \mathcal{T}([i^+, \ldots, j^+]^T)$. Therefore, we are able to convert any game with $M > 1$ on graph $G(t)$ to a game with $M = 1$ on graph $\mathcal{G}'(t)$

Finally, because blue robots are identical, we can reduce the size of the joint state graph by considering that robots are indistinguishable, allowing for simpler games. For the case of two blue robots, for example, the number of nodes in $\mathcal{G}'$ is $\frac{1}{2}N(N+1) < N^2$, for $N > 1$. Figure 3 shows the joint state graph for two robots for a simple base graph.

B. Game Solution and Value

Here we present some theoretical results on the game solution and value, before moving on to numerical methods for more complete solutions. Firstly, we provide some preliminaries to aid in the analysis. Define the "highest cost" graph as $\overline{G} = (\mathcal{V}, \mathcal{E}, \overline{W})$, where

$$\overline{W}_{ij} \triangleq \max_k W^k_{ij}. \quad (9)$$

To simplify notation, let $d_G(i, j)$, for $i, j \in \mathcal{V}$ denote the distance from node i to node j on graph G, where the distance between two nodes is the length of the shortest path between them. Now, we provide a condition on the discount factor γ, which will be needed for certain results. The condition is

$$\gamma \geq 1 - \frac{C_{\text{min}}}{d_{\text{max}}}, \tag{10}$$

where

$$C_{\text{min}} \triangleq \min_{(i,j) \in \mathcal{E} \setminus (N,N), k \in \{1,\dots,K\}} W_{i,j}^k > 0 \tag{11}$$

$$d_{\text{max}} \triangleq \max_{p \in \mathcal{V}} d_{\overline{G}}(p, N). \tag{12}$$

C_{min} is the minimum possible stage cost that can be achieved without $p(t) = N$. Note that this is strictly positive by assumption on the graph weights. d_{max} is the maximum distance of any node from the goal on the highest cost graph $\overline{G}$. This condition is sufficient, for example, to guarantee that it is more costly for the blue player to never reach the goal. Finally, note that

$$\lim_{\gamma \to 1^-} \frac{C_{\text{min}}}{1-\gamma} = \infty. \tag{13}$$

Because d_{max} does not depend on the value of γ, there always exists $\gamma \in (0, 1)$ such that $\overline{J}\,(s) < \frac{1}{1-\gamma} C_{\text{min}}$. Since the discount factor γ is a parameter we can choose to help find a solution, rather than being defined by the problem, we can always chose it so that (10) is satisfied.

C. Bounds for the Outcome

Here we provide security strategies for both players which allow us to bound the expected game outcome $J(s, \pi_{\text{blue}}, \pi_{\text{red}})$, for $s \in \mathcal{S}$. The motivation here is that, although we may not know how to find the value or the Nash equilibrium policies theoretically, and approximating them numerically may be computationally intensive, these bounds will be relatively simple to compute and can be achieved by simple naive strategies.

1) Blue Player's Security Strategy: Consider a policy for the blue player $\hat{\pi}_{\text{blue}}$ where the action in state $s \in \mathcal{S}$ is

$$a_{\text{blue}} \in \arg \min_{p^+ \in A_{\text{blue}}(s)} W_{p(t)p^+}^k + d_{\overline{G}}(p^+, N), \tag{14}$$

and a corresponding upper bound

$$\overline{J}\,(s) = \min_{p^+ \in A_{\text{blue}}(s)} W_{p(t)p^+}^k + d_{\overline{G}}(p^+, N). \tag{15}$$

This means that the blue player optimizes over the edge costs of the current graph with the distance over the highest cost graph $\overline{G}$ as an estimate of the future value.

Theorem 1: For $M = 1$, if the blue player follows the policy $\hat{\pi}_{\text{blue}}$ defined by (14), then

$$\overline{J}\,(s) \geq J(s, \hat{\pi}_{\text{blue}}, \pi_{\text{red}}) \text{ and } \overline{J}\,(s) \geq V(s), \tag{16}$$

for all $s \in \mathcal{S}$ and π_{red}, with $\overline{J}\,(s)$ defined in (15).

Proof: For reasons of space, we provide only a sketch of the proof.

If the blue player follows the shortest path on $\overline{G}$, then $J(s, \dots) \leq d_{\overline{G}}(p, N)$, regardless of the red player's policy. Now, if the blue player follows policy $\hat{\pi}_{\text{blue}}$ at the current time step, but follows the shortest path on $\overline{G}$ for all future time, then we have $J(s, \dots) \leq \overline{J}\,(s)$. However, we need to show that this holds if the blue player follows policy $\hat{\pi}_{\text{blue}}$ for all future time. Because $W_{p(t)p^+}^k \leq \overline{W}_{p(t)p^+}$, we have $\overline{J}\,(s) \leq d_{\overline{G}}(p, N)$, trivially from (15). This means that $d_{\overline{G}}(p^+, N)$ can act as an upper bound for the future cost, if the blue player uses $\hat{\pi}_{\text{blue}}$ for all future time. This allows us to guarantee that the bound is satisfied if the blue player chooses its action according to (14). Finally, to bound the value, note that

$$\overline{J}\,(s) \geq J(s, \hat{\pi}_{\text{blue}}, \pi_{\text{red}}^*) \geq V(s). \tag{17}$$

■

Note that this may be a conservative bound, because it assumes that the red player will know all the blue player's actions, that the red player has complete control over the graph (not being limited by ammo for instance), and that the blue player will not use any information about $G(t)$ for all future times.

2) Red Player's Security Strategy: Now, consider that the red player selects its action according to

$$a_{\text{red}} \in \arg \max_{k^+ \in A_{\text{red}}(s)} \min_{p^+ \in A_{\text{blue}}(s)} W_{pp^+}^k + \gamma^{N-1} d_{G^{k^+}}(p^+, N), \tag{18}$$

with a corresponding lower bound

$$\underline{J}\,(s) \triangleq \max_{k^+ \in A_{\text{red}}(s)} \min_{p^+ \in A_{\text{blue}}(s)} W_{pp^+}^k + \gamma^{N-1} d_{G^{k^+}}(p^+, N). \tag{19}$$

Theorem 2: Under the condition (10), for $M = 1$, if the red player follows the policy $\hat{\pi}_{\text{red}}$ defined by (18), then

$$\underline{J}\,(s) \leq J(s, \pi_{\text{blue}}, \hat{\pi}_{\text{red}}) \text{ and } \underline{J}\,(s) \leq V(s), \tag{20}$$

for all $s \in \mathcal{S}$ and π_{blue}, with $\underline{J}\,(s)$ defined in (19).

Proof: Again, for reasons of space, we provide only a sketch of the proof. First, consider the case where the red player never changes the graph, letting π'_{red} be such that $a_{\text{red}}(t) = k$ for $G(t) = G^k$. Now, define

$$d_{G^k}^{\gamma}(p) \triangleq \min_{\pi_{\text{blue}}} J((p, G^k, \alpha), \pi_{\text{blue}}, \pi'_{\text{red}}), \tag{21}$$

which is the minimum discounted distance to the goal on this graph, and is the result of the blue player's best response to this policy π'_{red}. If condition (10) is satisfied, then it can be shown that shortest paths to the goal do not contain repeated nodes. Therefore, we can upper bound the number of edges traversed on such a path by $N - 1$. This allows us to claim

$$d_{G^k}^{\gamma}(p, N) \geq \gamma^{N-2} d_{G^k}(p, N), \tag{22}$$

because γ^{N-2} is the lowest discount which can be achieved on a shortest path. Now, consider that the red player selects its action according to (18) for the current time step and leaves the graph constant for future time steps, in which case the outcome is lower bounded by $\underline{J}\,(s)$ for all π_{blue}, by its definition. We must again show that this still holds if the red player follows $\hat{\pi}_{\text{red}}$ for all future time. This follows from the fact that

$$\underline{J}\left((p, G^k, \alpha)\right) \geq \gamma^{N-2} d_{G^k}(p, N). \tag{23}$$

Finally, to bound the value, note that

$$\underline{J}(s) \leq J(s, \pi^*_{\text{blue}}, \hat{\pi}_{\text{red}}) \leq V(s). \tag{24}$$

■

This bound may be conservative because it assumes that the red player can only change the graph at the current time step and doesn't account for future changes, and because it assumes that the blue player will choose its best response actions.

D. Game Solution

Here we guarantee that, if both players use Nash equilibrium strategies π^*_{blue} and π^*_{red}, then the blue player is guaranteed to reach the goal, under mild assumptions. Note that if $\gamma < 1$, the blue player can achieve a finite cost without reaching the goal.

Theorem 3: For $M = 1$, given an initial condition $S(0) \in \mathcal{S}$, under the Markov chain with state transitions given by (4) with actions chosen according to $\pi^*_{\text{blue}}, \pi^*_{\text{red}}$ satisfying (6), if γ satisfies (10), then $\lim_{t\to\infty} \Pr(p(t) = N) = 1$.

Proof: We present only a sketch of the proof for reasons of space. If the policies of both players are fixed to their Nash equilibrium values, the game reduces to a Markov chain. Therefore, we simply need to show that any state $s = (p, k, \alpha) \in \mathcal{S}$ where $p = N$ is an absorbing state and any other states are not absorbing for this Markov chain [13]. Then, if we can show that the Markov chain is absorbing, we have our desired result.

Because the goal node N has a self-loop with an edge cost of zero, we know that if the blue player reaches the goal, then it accrues no further costs under an equilibrium blue policy, so $J((N, k, \alpha), \pi^*_{\text{blue}}, \pi^*_{\text{red}}) = 0$ and so we must have $p(t) = N$ for all future time and these states are absorbing.

If, under some policy π_{blue}, the blue player never reaches the goal (the probability of reaching the goal from state $S(t) \in \mathcal{S}$ is zero), then the value can be lower bounded:

$$J(s, \pi_{\text{blue}}, \pi_{\text{red}}) \geq \sum_{t=1}^{\infty} \gamma^t C_{\min} = \frac{1}{1-\gamma} C_{\min}, \tag{25}$$

by assuming that it gets the minimum cost at every time and from evaluating the geometric series.

Because we have an upper bound for the value $\overline{V}(s)$, if we can guarantee that $\overline{V}(s) < \frac{1}{1-\gamma} C_{\min}$, then we know that there is a positive probability that, under Nash equilibrium policies, the blue robot reaches the goal at some future time step from state s. Noting that $\overline{V}(s) \leq d_{\max}$, from the condition (10), we know that $V(s) \leq \overline{V}(s) < \frac{1}{1-\gamma} C_{\min}$ for all $s \in \mathcal{S}$, so there must be a strictly positive probability that the blue robot reaches the goal from every state s. This also guarantees that no state with $p \neq N$ is absorbing.

Now, because we have shown that, from every state $S(t) \in \mathcal{S}$ there is a strictly positive probability that $p(t') = N$ for some $t' \geq t$, we can claim that the Markov chain induced by NE policies is absorbing. Therefore, we can guarantee that $\lim_{t\to\infty} \Pr(p(t) = N) = 1$. ■

IV. Numerical Methodology

Because we believe that solving for these equilibrium strategies $\pi^*_{\text{blue}}, \pi^*_{\text{red}}$ analytically is not feasible within this work, we turn to numerical methods. We use a value iteration method based on the work of Shapley [11], [8]. First, we define a matrix valued Q function $\mathcal{Q} \in \mathbb{R}_{\geq}^{|A_{\text{red}}(s)| \times |A_{\text{blue}}(s)|}$ as a function of the state $s \in S$, where each element is

$$\mathcal{Q}_{a_r, a_b}(s) = C(s, a_{\text{blue}}) + \gamma V(s^+), \tag{26}$$

where $a_r \in A_{\text{red}}(s)$ is an action of the red player, $a_b \in 1, 2, \ldots, |A_{\text{blue}}(s)|$ is an index corresponding to blue player's action a_{blue}, and s^+ is the state at the next time step from (4), given a_{blue} and a_{r}. This function gives the value of the game outcome if each player takes an arbitrary action at the current time step, but then follows the equilibrium policy (π^*_{blue} or π^*_{red}) for all future time steps. According to Shapley's Theorem, the equilibrium policies $\pi^*_{\text{blue}}(s)$, $\pi^*_{\text{red}}(s)$ and the game value can then be found by solving the matrix game corresponding to $\mathcal{Q}(s)$, for each state $s \in \mathcal{S}$, where here the red player is the row player and maximizing and the blue player is the column player and minimizing [8]. However, because we do not know the value function, we instead use numerical methods to calculate the Q function.

For each iteration τ, we compute the estimate of the policies π^τ_{blue}, π^τ_{red} for the given estimate of the Q function $\hat{Q}^\tau$, by solving the matrix game corresponding to $\hat{\mathcal{Q}}^\tau(s)$ for each state $s \in \mathcal{S}$. Next, we update the estimate of the Q function $\hat{Q}^{\tau+1}$ using those policies as follows:

$$\hat{\mathcal{Q}}^{\tau+1}_{a_b, a_r}(s) = C(s, a_{\text{blue}}) + \gamma {\pi^\tau_{\text{red}}}^T \hat{\mathcal{Q}}^\tau(s^+) \pi^\tau_{\text{blue}}, \tag{27}$$

for each $s \in \mathcal{S}$, $a_b \in 1, 2, \ldots, |A_{\text{blue}}(s)|$, and $a_{\text{r}} \in A_{\text{red}}(s)$, and where s^+ is found using (4) and a_{blue} is the action corresponding to a_{r}. When this converges, $\pi^\tau_{\text{blue}}, \pi^\tau_{\text{red}}$ can approximate the equilibrium strategies $\pi^*_{\text{blue}}, \pi^*_{\text{red}}$.

A. Simplifying Computation

Here we provide methods which will simplify the numerical computation of the value.

1) Dominated Blue Actions: Under certain circumstances, it is possible to determine that there are some edges that it is never beneficial for the blue robot to take as it traverses the graph. Specifically, we look for dominated actions. For a given game state $S(t) \in \mathcal{S}$, let $p(t) = p_0 \in \mathcal{V}$, and consider two actions of the blue player, $p_1, p_2 \in A_{\text{blue}}(S)$. A sufficient condition for action p_2 to dominate action p_1 is

$$\begin{aligned} &C(S, p_1) + \gamma \underline{J}((p_1, G^{a_{\text{r}}}, \alpha_{a_{\text{r}}})) \geq \\ &\quad C(S, p_2) + \gamma \overline{J}((p_2, G^{a_{\text{r}}}, \alpha_{a_{\text{r}}})), \end{aligned} \tag{28}$$

for all $a_{\text{r}} \in A_{\text{red}}(S)$, where $\alpha_{a_{\text{r}}}$ is the red player's new ammo count as determined by S and a_{r} according to (4).

Because the blue player never benefits from using a dominated strategy, we can use the condition (28) to remove edges from the graphs before solving for the value, without changing the results. Specifically, if condition (28) is satisfied for some p_0, k, p_1, p_2 and all α, then we can remove edge (p_0, p_1). Therefore, one could iterate over all pairs of actions for all nodes, removing the edges corresponding to dominated actions, and finally remove any nodes from which the goal node cannot be reached. This allows the game to be solved for fewer states and actions for the blue player, without changing the results.

2) Sub-Game Formulation: From equations (5) and (7), we can write the value, for a given game state $S(t) = s = (p, G, \alpha) \in \mathcal{S}$, as

$$V(s) = \sum_{s' \in \mathcal{S}} \Pr\Big\{s^+ = s'\Big\}\Big(C(s, p') + \gamma V(s')\Big),$$

where $s^+ = (p^+, G^+, \alpha^+) = S(t+1) \in \mathcal{S}$ and $s' = (p', G', \alpha') \in \mathcal{S}$, and $\Pr\{s(t+1) = s'\}$ is found according to (4) when the actions are chosen according to Nash equilibrium policies, π^*_{blue} and π^*_{red}. Now, the two possibilities are either the graph remains the same, in which case $s^+ = (p^+, G, \alpha)$, or it is changed by the red player, in which case $s^+ = (p^+, G^+, \alpha - 1)$. This is important, because it implies that the value of the game at some ammo state α depends only on the value at lower ammo states $\alpha' \leq \alpha$. We can take advantage of this to formulate sub-games which allow us to solve the original game.

Instead of considering the graph G and the ammo α to be part of the game state, we consider them to be parameters of a sub-game. The game state of the sub-game is simply blue's position p, and the sub-game ends when red takes action to change the graph. The actions of the players, the game dynamics, and the stage costs are the same as before, but now we consider a terminal cost when the red player ends the sub-game by changing the graph. The value for the sub-game defined by graph G and ammo α is $V_{G,\alpha}(p)$ for sub-game state $p \in \mathcal{V}$, and the terminal costs will be $V_{G(T),\alpha-1}(p_T)$, where p_T is blue's position when red takes action to change the graph to $G(T) \neq G$ at time $T-1$. We can write the outcome of this sub-game, for $p(0) = p$, as

$$V_{G,\alpha}(p) = E\Big[\sum_{t=0}^{T-1} \gamma^t C\left(S(t), p(t+1)\right) + \gamma^T V_{G',\alpha-1}(p(T))\Big]. \tag{29}$$

Note that $V_{G,0}(p) = d_G(p, N)$, allowing us to solve for the value of the full game by iteratively solving the sub-games, using the value of the sub-games with ammo α to solve for the value of the sub-games with ammo $\alpha + 1$.

Compared to the full game, if we have K different graphs with N nodes and a maximum ammo of $\overline{\alpha}$, the trade-off is considering $K\overline{\alpha}$ sub-games with N states each, instead of considering one game with $NK\overline{\alpha}$ states. Because the number of game states corresponds to the number of Q-matrices which must be updated and the number of auxiliary matrix games which must be solved at every iteration, solving the game as sub-games may result in less computation, as only the Q-matrices and value relevant to the currently considered sub-game must be updated at a time.

V. Numerical Results

Here we show the results of applying the numerical methods in Section IV. Unless otherwise specified, the discount factor was chosen to be $\gamma = 1 - 10^{-9} \approx 1$ and the red player's action graph $\mathcal{G}^{\text{red}}$ is complete. First, we show and discuss the results for some interesting cases, and then we perform a statistical analysis of randomly constructed graphs.

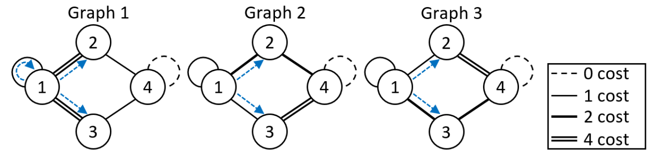


Fig. 4. This graph results in mixed strategies; node 4 is the goal node. The blue arrows indicate the possible actions of the blue player under the equilibrium strategy from node 1. On graph 1, the red player also mixes over its three options, but on graph 2 or 3, the red player only mixes between selecting graph 2 and graph 3.

A. Illustrative Examples

a) Benefit of Mixing: Figure 4 shows an example graph where each player mixes between three options from node 1, with the blue player mixing between remaining at that node and between taking either branch. With $s(0) = (1, G^1, 1)$, the blue player chooses its action according to $[\Pr(p^+ = 1), \Pr(p^+ = 2), \Pr(p^+ = 3)] = [0.5, 0.25, 0.25]$, while the red player chooses its action according to $\pi^*_{\text{red}}(s(0)) = [0.5, 0.25, 0.25]^T$. Intuitively, the blue player seems to mix between remaining at the current node and moving towards the goal when the current graph has a high immediate edge cost, while the red player is incentivized to change the graph because the current graph has lower costs closer to the goal. On the other hand, if $G(0) = G^2$ instead, the blue player chooses its action according to $[\Pr(p^+ = 1), \Pr(p^+ = 2), \Pr(p^+ = 3)] = [0, 0.5, 0.5]$, and the red player chooses its action according to $\pi^*_{\text{red}}(s(0)) = [0, 0.75, 0.25]^T$. In this case, the players both only mix between two different actions, because one path is cheaper on one graph, and the other path is cheaper on the other graph.

b) Benefit of Multi-Robot Coordination: The graph in Figure 5 shows a case where having more robots is beneficial for the blue player. The only differences between the two graphs are that the $(5, 7)$ edge weight is cheaper in Graph 1 and the $(6, 7)$ edge weight is cheaper in Graph 2. Due to the longer length of these branches (compared to the graphs in 4), the red player can simply choose $G(t+1) = G^1$ when $p(t) = 4$ or $G(t+1) = G^2$ when $p(t) = 3$, forcing the blue player to take the higher edge cost. If the blue player has multiple robots, however, then they can split up in a coordinated fashion, so that at least one of them takes the cheaper path. To formalize this benefit, we define the average value per robot as $\tilde{V}_M(s) \triangleq V(s)/M$. When $\alpha(0) \geq 1$, for a single robot $(M = 1)$, $\tilde{V}_1(S(0)) = 20$ with $p(0) = 1$, but for two robots, $\tilde{V}_2(S(0)) = 13$ with $p(0) = [1, 1]^T$, $G(t) = G^2$, and $\alpha(0) \geq 1$. This indicates that, depending on the graphs, the blue player can gain an advantage from having multiple robots which split up in a coordinated fashion.

Another type of coordinated behavior seen in Figure 5 is that the robot which could traverse fewer edges to reach the goal instead waits at node 5. If this leading robot on node 5 goes to the goal immediately, then the red player could

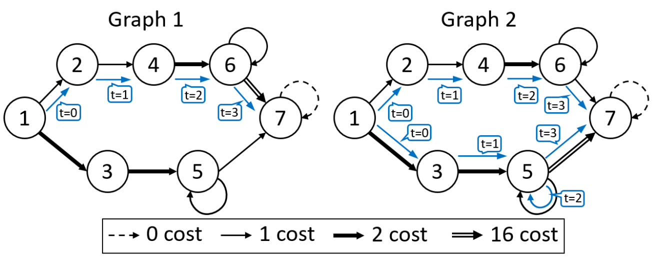


Fig. 5. Here, it is better for the blue player to have multiple robots which split up to reach the goal node 7. The blue arrows, shown on graph 1, show the policy of a single robot starting at node 1, with the speech bubbles indicating the time of each action. The blue arrows shown on graph 2 indicate the paths followed by two blue robots starting from node 1. For $M = 1$, and $\alpha(0) = 1$, the red player will always change the graph so that the blue player will get a cost of 16 for taking the $(6, 7)$ edge, but for two robots the red player's policy has no effect on the game outcome and exactly one robot will take the cheaper path to the goal.

switch the graph so that both robots must take the 16 cost to the goal. Instead, that robot waits, which is more costly for itself, but since both robots traverse the final risky edges synchronously, at least one of them achieves the cheaper cost.

c) A More Practical Example: Here, we discuss how these strategies we've discussed on smaller graphs fit into a larger graph. Figure 6 shows portions of an example trajectory under equilibrium policies for a larger graph of $N = 10$ nodes, with $M = 4$ blue robots, and a maximum ammo count for the red player of $\overline{\alpha} = 5$, while Table I contains more details. Both players employ mixed strategies, so this is only one possible outcome for these initial conditions. For example, at $t = 1$, the blue player's policy can be described by $[\Pr(p^+ = [1,1,5,6]^T), \Pr(p^+ = [1,1,6,6]^T), \Pr(p^+ = [1,2,6,6]^T)] = [0.4928, 0.3482, 0.1590]$ and the red player's is $\pi^*_{\text{red}}(s(1)) = [0.6536, 0.0818, 0.2646]^T$. Additionally, the blue player makes use of some self-loops to allow some of its robots to wait, and the blue robots also split up. The value per robot is $\tilde{V}_4(S(0)) = 14.4667$, while the value per robot for a single robot is higher at $\tilde{V}_1(S(0)) = 15.2474$, again suggesting that having more robots is helpful for the blue player. Finally, this value is between the lower and upper bounds for the value, $\underline{V}(S(0)) = 12$ and $\overline{V}(S(0) = 80$, as we expect.

t	$p(t)$	$G(t)$	$\alpha(t)$	$\sum_{t'=0}^{t-1} C(t')$
0	$[1,1,1,1]^T$	G^1	5	0
1	$[2,2,1,1]^T$	G^1	5	18
2	$[6,6,2,1]^T$	G^1	5	31
3	$[10,10,5,1]^T$	G^2	4	38
4	$[10,10,10,2]^T$	G^3	3	50
5	$[10,10,10,6]^T$	G^3	3	54
6	$[10,10,10,10]^T$	G^3	3	62

TABLE I
EXAMPLE NASH EQUILIBRIUM TRAJECTORY FOR THE GRAPHS SHOWN IN FIGURE 6

B. Statistical Results

To statistically demonstrate some properties of this game and its solution, we constructed graphs in a random fashion. More specifically, we used a modification of the canonical Erdős–Rényi model [14]. For a given set of nodes $\mathcal{V}$ of size $N_{\max}$, the probability that any given pair of nodes $(i, j) \in \mathcal{V} \times \mathcal{V}$, for $i \neq j$, is in the edge set of the graph is a constant $\varphi \in [0, 1]$. We also consider directed edges and possible self-loops, and we take $\varphi = 0.5$. To ensure the goal can be reached from all nodes, we find the longest directed distance between any two nodes, and assign one of those ending nodes to be the goal node and a corresponding starting node to be node 1. Then, we remove any nodes, after the initial generation, from which the goal node cannot be reached, which results in a graph with $N \leq N_{\max}$ nodes. Finally, we add a self-loop to the goal node N, if necessary. We set $K = 3$, and randomly assign edge weights for each graph G^k so that, for $(i, j) \in \mathcal{E}$ and $i \neq j$, $(W^1_{ij}, W^2_{ij}, W^3_{ij})$ is a permutation of the set $\{2, 4, 8\}$, and each permutation is chosen with equal probability. The self-loop weights are assigned according to $W^k_{ii} = 1$ for $(i, i) \in \mathcal{E}$ and $i \neq N$ and $W^k_{NN} = 0$. Using this method, 100 graphs were generated for each initial graph size $N_{\max}$ between 4 nodes and 8 nodes.

We consider a common initial condition, where all blue robots are on node 1 in graph 1, so that $p(0) = \mathbf{1}_M$ and $G(0) = G^1$, and the red player has ammo $\alpha(0) = 6$. In Figure 7, the values for this initial condition have been normalized so that the lower bound $\underline{J}(s(0))$ is zero and the upper bound $\overline{J}(s(0))$ is one in the plot. Some trivial graphs have been excluded from the results because $\underline{J}(s(0)) = \overline{J}(s(0))$ on them. The bounds are respected in all cases, and are sometimes tight. Additionally, the median and mean *normalized* value exhibits a downward trend as the number of nodes increases. This is likely because more nodes makes it more probable that the blue player will have more options for paths to take to the goal, allowing it to gain an advantage through mixed policies, for example.

a) Effect of the Number of Blue Robots: As discussed in the preceding sections, we expect that having more blue robots will decrease the cost per robot. Figure 7 also plots the values per robot $\tilde{V}_M(S(0))$ for the random graphs for different numbers of robots. For the sake of computation time, the game was solved for $M = 4$ only for graphs with $N \leq 7$. We observe that the mean value per robot does decrease as the blue player's number of robots increases for these graphs, for each graph size. This is evidence that having more robots is indeed an advantage for the blue player.

b) Effect of the Red Player's Ammo Count: Figure 8 shows the effect of the red player's ammo count on the normalized game value. Here, the same initial position for the blue player is considered, but the initial amount of ammo for the red player is varied. The average of the normalized values is taken over all the random graphs and plotted against the ammo count. As expected, the value is nondecreasing with respect to the ammo count, and the value plateaus around 3 or 4 ammo, below the upper bound for the value. This suggests that having more ammo stops benefiting the red player in terms of the value, once it is high enough.

VI. CONCLUSIONS

To handle uncertainty in a time-varying environment, we have formulated a novel zero-sum game where the blue

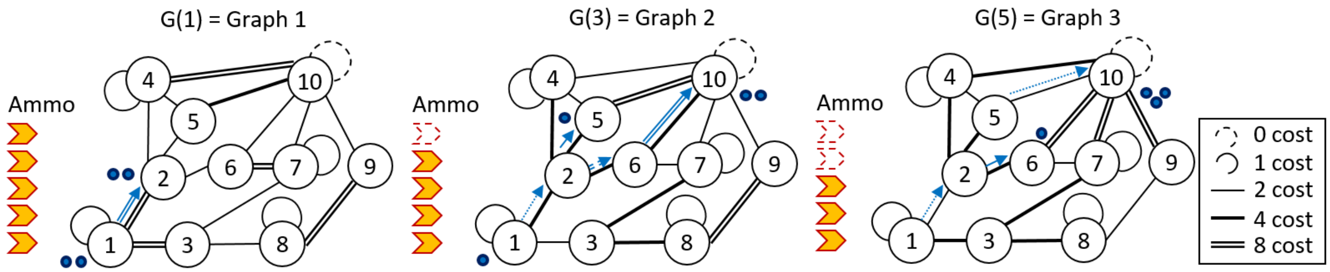


Fig. 6. This shows part of an example trajectory, under equilibrium strategies, for a game on a larger graph with the goal as node 10, with four blue robots. The game state is shown for $t \in \{2, 4, 6\}$. The circle markers indicate the positions of the blue robots, the filled chevrons indicate the remaining ammo, and the arrows indicate the blue robots' movement on previous time steps. Specifically, the double-lined arrows indicate two robots moving together, the solid lines indicate movement on the preceding time step, and the dotted lines indicate movement on the time step before that. See Table I for the full trajectory. We also observe that the blue robots wait at node 1 and split up, so that not all take the expensive first edges from node 1 on graph 1.

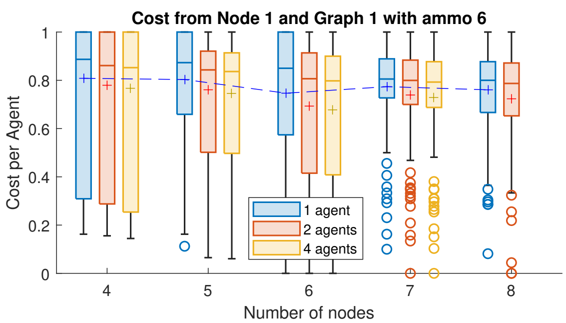


Fig. 7. This shows the value per robot $\tilde{V}_M(S(0))$, from a common initial condition, over about 500 random graphs, which has been normalized so that the lower bound is zero and the upper bound is 1. For each different number of blue robots, the value is plotted against the number of nodes in the random graphs. The plus sign indicates the mean, the center horizontal line indicates the median value, the middle 50% of the values are inside the box, the whiskers indicate the furthest non-outlier values, and the circles indicate the outliers. The values are always between the upper and lower bounds, there is a general downward trend in the median and mean value as the number of nodes increases, and the mean decreases with increasing numbers of robots.

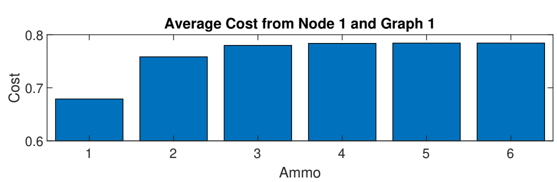


Fig. 8. This shows the average, over about 500 random graphs, cost from the starting node to the ending node, as a function of ammo. It's been normalized so that the lower bound is zero and the upper bound is 1, but the plot bounds are only between 0.6 and 0.8 to save space. The value appears to be a nondecreasing function of the ammo and it asymptotically approaches a value which is significantly lower than the upper bound.

player's robots move through a graph with time-varying costs to reach a goal, while the red player controls controls those costs. This can be appropriate for robots moving through hazardous environments, either to model the worst-case scenario or to handle an actual adversary. Due to the complexity of the game, we provided security strategies for both players, and demonstrated the game solution through numerical methods, with a theoretical guarantee that the blue robots reach the goal. We provided methods to simplify computation, and find that both players benefit from mixed strategies, while the blue player benefits from having multiple robots which use coordinated strategies. These include splitting up and waiting to traverse risky edges synchronously, ensuring at least some take the cheaper paths. In future work, we may analytically solve the game in some specific cases.

References

[1] B. Ding, J. X. Yu, and L. Qin, "Finding time-dependent shortest paths over large graphs," in *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, pp. 205–216, 2008.

[2] X. Cai, T. Kloks, and C.-K. Wong, "Time-varying shortest path problems with constraints," *Networks: An International Journal*, vol. 29, no. 3, pp. 141–150, 1997.

[3] Y. Yuan, X. Lian, G. Wang, Y. Ma, and Y. Wang, "Constrained shortest path query in a large time-dependent graph," *Proceedings of the VLDB Endowment*, vol. 12, no. 10, pp. 1058–1070, 2019.

[4] C. A. Dimmig, K. C. Wolfe, and J. Moore, "Multi-robot planning on dynamic topological graphs using mixed-integer programming," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5394–5401, IEEE, 2023.

[5] K. Durkota, V. Lisy, B. Bošansky, and C. Kiekintveld, "Optimal network security hardening using attack graph games," in *Proceedings of IJCAI*, pp. 7–14, 2015.

[6] T. H. Nguyen, M. Wright, M. P. Wellman, and S. Baveja, "Multi-stage attack graph security games: Heuristic strategies, with empirical game-theoretic analysis," in *Proceedings of the 2017 Workshop on Moving Target Defense*, pp. 87–97, 2017.

[7] K. C. Nguyen, T. Alpcan, and T. Basar, "Stochastic games for security in networks with interdependent nodes," in *2009 International Conference on Game Theory for Networks*, pp. 697–703, IEEE, 2009.

[8] J. Filar and K. Vrieze, *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[9] S. Kloder and S. Hutchinson, "Partial barrier coverage: Using game theory to optimize probability of undetected intrusion in polygonal environments," in *2008 IEEE International Conference on Robotics and Automation*, pp. 2671–2676, IEEE, 2008.

[10] D. Shishika, D. G. Macharet, B. M. Sadler, and V. Kumar, "Game theoretic formation design for probabilistic barrier coverage," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11703–11709, IEEE, 2020.

[11] L. S. Shapley, "Stochastic games," *Proceedings of the national academy of sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.

[12] M. Limbu, Z. Hu, S. Oughourli, X. Wang, X. Xiao, and D. Shishika, "Team coordination on graphs with state-dependent edge costs," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 679–684, IEEE, 2023.

[13] J. G. Kemeny and J. L. Snell, *Finite markov chains*, vol. 26. van Nostrand Princeton, NJ, 1969.

[14] P. Erdős, A. Rényi, *et al.*, "On the evolution of random graphs," *Publ. math. inst. hung. acad. sci*, vol. 5, no. 1, pp. 17–60, 1960.