Safety First: Strategic Navigation in Adversarial Environments

(Extended Abstract)

Ofri Keidar, Noa Agmon
Bar Ilan University
Israel
ofri.keidar@biu.ac.il, agmon@cs.biu.ac.il

ABSTRACT

This work deals with the problem of navigation while avoiding detection by a mobile adversary, which is a novel variant of pursuit-evasion featuring adversarial modeling. In this problem, an evading agent is placed on a graph, where one or more nodes are defined as *safehouses*. The agent's goal is to find a path from its current location to a safehouse, while minimizing the probability of meeting a mobile adversarial agent at a node along its path (i.e., being captured). We examine several models of this problem, where each one has different assumptions on what the agents know about their opponent, all using a framework for computing node utility, introduced herein. We use several risk attitudes for computing the utility values, whose impact on the constructed strategies is analyzed both theoretically and empirically. Furthermore, we allow the agents to use information gained along their movement, in order to efficiently update their motion strategies on-the-fly. Analytic and empirical analysis show the importance of using this information and these on-the-fly strategy updates.

Keywords

Robot Navigation, Pursuit-Evasion, Adversarial Modeling

1. INTRODUCTION

The modern battle field has moved towards civilian population. Cities and villages are surrounded by hostiles, making the delivery of food, water and medical aid a life endangering mission. Thus, emerges the problem of path planning in an environment with a hostile entity, with the advantage of strategic behavior. In this paper we introduce a new variant of traditional path planning problem: **Strat**egic **N**avigation in **A**dversarial En**v**ironments (or StratNAV, in short). In this problem, we aim at planning a path for our agent (denoted as R), while avoiding being captured by a mobile adversarial agent (denoted as C). The agents travel about a graph, representing a map of the environment, where some nodes in this graph are defined as *safehouses*. The goal of R is to arrive at one of the safehouses without being intercepted by C on its way there (*capture* it).

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

The problem of traveling in an environment while avoiding threats has been studied from different perspectives [1,3,5–7,11]. In the problem of Pursuit-Evasion [2,4,9,10], two rival agents move around the environment (e.g., along the edges of a graph), until the pursuer moves to the evader's location. Most research in pursuit evasion focuses on aspects concerning topology of the graph, for example, on defining properties related to graph theory of the given graph, in order to characterize graphs where the pursuer is guaranteed to capture the evader or finding minimal number of pursuers required. In our problem, however, the evader R moves to a certain destination, and does not try only evade its pursuer indefinitely. Furthermore, our problem addresses strategic behavior and game theory concepts. R's path is planned based on strategies that take into account its risk attitude.

We formally define the StratNAV problem, and examine its different variants. A framework for computing utilities associated with each node in the graph is presented, used for finding solutions to the StratNAV problem in all examined variants. Theoretical guarantees are proven, e.g., expected utility maximization, or equilibrium. Assuming the agents may gain information about their opponent while they are on their way to their target—either a safehouse or interception—we use this information to adjust the strategy efficiently. Theoretical analysis and extensive experiments show that these updates significantly increase the chances of R to reach its destination safely.

2. PROBLEM DEFINITION

The StratNAV (**Strat**egic **N**avigation in **A**dversarial En**v**ironments) problem is formally defined as follows:
Given a graph $G = (V, E)$, representing a map of the environment (referred to as *map graph*), $V_G \subseteq V$ a set of goal nodes (*safehouses*) and two distinct initial positions of an agent R and an adversarial agent C, find a strategy that will maximize R's chances of reaching some node $v_g \in V_G$ without being captured by C. R is captured by C if both agents reside the same node. R wins if it reached a goal node $v_g \in V_G$ without being captured, while C wins if it captures R.

Note that the strategy may be deterministic or stochastic, and changes based on the knowledge the agents have on their opponent's strategy and location, and on the risk attitude adopted by the agents. We assume C is capable of performing the same computations as R.

3. ESTIMATING SAFETY OF MAP NODES

A *utility* value is defined for each map graph node $v \in V$. The utility of $v \in V$ expresses how *safe* it is for R if moves to v, i.e., how probable it is to evade capture and reach a goal node and win. This value is obtained by evaluating the game configurations (i.e., game states) where R resides at v.

Specifically, a game configuration holds current location of both agents R,C. The *configuration graph* $G_{conf} = (V_{conf}, E_{conf})$ is defined with a node for each configuration and an edge between any pair of consecutive game states. Configurations matching *win configuration* – game states where R wins – are given a utility of 1, while those matching game states where R loses (*lose configurations*) are given a utility of 0. For all other configurations, the utility value depends on that of its neighbors, and this value is propagated from the win and lose configurations (*terminal configurations*): Configurations are traversed along G_{conf} in ascending order of distance (in hops) from any terminal configuration. The utility of a configuration $\mathcal{V}$ is computed based only on its neighbors whose utility value had already been computed. Various risk attitudes can be used: risk averse (utility of $\mathcal{V}$ is minimal utility among visited neighbors) or risk neutral (average utility among neighbors).

Once all configurations have been given a utility value, the utility for a map graph node $v \in V$ is the average utility of all configurations $\mathcal{V} \in V_{conf}$, such that $\mathcal{V} = < v, u >$, $u \in V$. This manner relates to a risk neutral type of player. Other risk attitudes can be applied, e.g., risk averse (node v's utility is the minimal utility of a configuration $\mathcal{V} = < v, u >$).

C's objective is opposed to R's, i.e., a win configuration for R is a lose one for C and vice versa – as expected in a zero-sum game. Therefore, C plans its strategies based on utility values that are opposed to R's: a terminal configuration where R is captured is given a value of 1 and a terminal configuration where R resides at a goal node (and C at another node) is given a value 0, while for any other configuration $\mathcal{V} \in V_{conf}$, $U_{\mathrm{C}}(\mathcal{V}) = 1 - U_{\mathrm{R}}(\mathcal{V})$ (where $U_{\mathrm{R}}(\mathcal{V})$ is the utility value of $\mathcal{V}$ as computed for R). In that case, greater utility values relate to nodes where it is more probable for C to capture R. Unless stated otherwise, the term *utility values* refers to R's utilities.

4. OFFLINE STRATEGIES

In the *offline* model the agents are not visible during the game, unless they occupy the same map node. An exception is made for the initial location: four cases are addressed herein, differ in which agent knows where its opponent starts. In cases C does not know R's initial location, R follows a deterministic motion strategy, maximizing the sum of node utilities along R's path, proven[1] to maximize its probability to reach some goal node safely. C follows a stochastic strategy, proven[1] to maximize C's probability to capture R. We have observed[1] that in case only R knows its opponent's initial location, R benefits more from planning its path ignoring this information.

In case C knows R location, R moves stochastically (otherwise will be easily intercepted by C): R chooses stochastically a neighbor, biased towards neighbors with greater node utility values. C follows a similar stochastic strategy, based on its utility values (opposed to R's). We have proven[1] that if the players apply risk neutral as configurations and map nodes utility function when computing utility values, and follow their stochastic strategies, then at each time step R,C maximize their expected payoff.

[1]Full proofs and analysis are available in the full version of this paper, at: `https://sites.google.com/site/ofrikeidarhomepage/publications` Please remove space at newline break

5. STRATEGY UPDATES ON-THE-FLY

In the offline problem variant (Section 4), strategies aim to reduce the probability that C captures R, based on the map graph's topology. However, relying solely on graph topology, i.e., offline planning, means no reaction to new information gained while moving around the map graph. Now, some nodes can be observed by the other nodes, which can be considered as *viewpoints*. Such nodes provide information regarding an agent, e.g., tracks the agent had left behind or perhaps whether the agent currently resides at the node. When the agents follow the stochastic strategies computed as stated in Section 4, they can use these viewpoints in order to acquire information concerning their opponent's location or visited nodes, and update their strategies accordingly (each agent and its own objective). The updates are performed in each time step and are very efficient, i.e., linear in the number of neighbors. The contribution of these runtime updates had been proven[1] both theoretically and empirically. Moreover, we have proven[1] that the game converges into a a Markov perfect equilibrium [8].

6. EMPIRICAL EVALUATION

We have evaluated our navigation strategies, using different risk attitudes for computing the utility functions, and examined the effect of online strategies update. A collection of graphs with 30 to 100 (with jumps of 5) nodes was randomly generated (40 of each size), so were the viewpoints. For each number of nodes, 10% of the nodes where randomly set as goal nodes (i.e., safehouses). Utility functions were used in order to compute a utility value for a map graph node or a configuration node and also used to evaluate the information obtained at a node visited by an agent.

An experiment was executed for each combination of node and configuration utility functions for R,C. Each experiment was repeated 20 times for each graph, choosing randomly new starting locations for both agents (not among the safehouses). For each graph size, combination of node, configuration and information utilities, the average winning rate of R was calculated (*R's winning rate*).

We have examined R's winning rate when both agents perform on-the-fly updates (i.e., run online) and when only C does. In both scenarios, the agents were set with risk-averse utility for configurations and risk-neutral for map nodes. In order to specifically examine the influence of on-the-fly updates, when R ran online, executions where R did not observe C towards the last quarter of the game were discarded. Same for C when R ran offline. When running online, R's winning rate was 85% but only 76% without on-the-fly updates. T-Test with $\alpha = 0.05$ has confirmed that if C runs online, R's winning rate is significantly greater with on-the-fly updates.

Acknowledgments

We gratefully acknowledge support by ISF grant #1337/15.

REFERENCES

[1] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984.

[2] B. Alspach. Searching and sweeping graphs: a brief survey. *Le matematiche*, 59(1, 2):5–37, 2006.

[3] R. B. Borie, C. A. Tovey, and S. Koenig. Algorithms and complexity results for pursuit-evasion problems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 9, pages 59–66, 2009.

[4] T. H. Chung, G. A. Hollinger, and V. Isler. Search and pursuit-evasion in mobile robotics. *Autonomous Robots*, 31(4):299–316, 2011.

[5] F. V. Fomin and D. M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science*, 399(3):236–245, 2008.

[6] M. Marzouqi and R. A. Jarvis. Covert path planning for autonomous robot navigation in known environments. In *Proceedings of the Australasian Conference on Robotics and Automation, Brisbane*. Citeseer, 2003.

[7] M. Marzouqi and R. A. Jarvis. Covert robotics: Covert path planning in unknown environments. In *Proceedings of the Australasian Conference on Robotics and Automation, Brisbane, Australia*. Citeseer, 2003.

[8] E. Maskin and J. Tirole. Markov perfect equilibrium, i: Observable actions. *Journal of Economic Theory*, 100(2):191–219, 2001.

[9] R. Nowakowski and P. Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2):235–239, 1983.

[10] J. Sgall. Solution of david gale's lion and man problem. *Theoretical Computer Science*, 259(1-2):663–670, 2001.

[11] R. Yehoshua and N. Agmon. Adversarial modeling in the robotic coverage problem. In *Proceedings of International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015.