Anytime Algorithms for Multi-agent Visibility-based Pursuit-evasion Games

(Extended Abstract)

Viliam Lisý, Branislav Bošanský, Michal Pěchouček
Agent Technology Center, Dept. of Computer Science and Engineering, FEE, Czech Technical University
Technická 2, 16627 Prague 6, Czech Republic
{lisy, bosansky, pechoucek}@agents.felk.cvut.cz

ABSTRACT

We investigate algorithms for playing multi-agent visibility-based pursuit-evasion games. A team of pursuers attempts to maintain visibility contact with an evader who actively avoids tracking. We aim for applicability of the algorithms in real-world scenarios; hence, we impose hard constraints on the run-time of the algorithms and we evaluate them in a simulation model based on a real-world urban area. We compare Monte-Carlo tree search (MCTS) and iterative deepening minimax algorithms running on the information-set tree of the imperfect-information game. The experimental results demonstrate that both methods create comparable good strategies for the pursuer, while the later performs better in creating the evader's strategy.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms

Algorithms, Experimentation

Keywords

Pursuit-evasion game, Monte-Carlo tree search, Information-set search, Anytime algorithm

1. PROBLEM DEFINITION

The problem of visibility tracking is of particular interest for defense or security domains in which the target actively avoids being seen by the tracking agents. Game theory provides theoretic and algorithmic foundations for such situations and a game modeling these scenarios is defined as a *visibility-based pursuit-evasion game* with simultaneous moves — a two-player zero-sum extensive-form game between the *pursuer* (that controls multiple pursuing agents) and the *evader*. We focus on variants of these games played in a Euclidean environment discretized as a graph. We assume that both players have a full knowledge about the topology of the environment, but do not know the position of the opponent's agents unless one of their agents can see them.

We adopt the definition of the visibility-based pursuit-evasion game from [3], and we assume a single evading agents and multiple centrally-controlled pursuing agents. The main objective of the pursuers is to minimize the *mean size* of the set of possible positions of the evader based on the shared information of the pursuer's agents (we denote this measure *MS*). The objective of the evader is exactly the opposite; however, the evader needs to approximate this value. The exact value depends on trajectories of the pursuers which may be unknown to the evader. Besides the mean size objective we evaluate two other performance measures. The first is the number of times the evader has been spotted by a pursuing agent (denoted *NS*). The second is the size of the set of possible positions of the evader at the end of the game (denoted *ES*), which is the objective used in [3].

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

2. ANYTIME ALGORITHMS

Both evaluated algorithms search in the same search space. It is the information set tree[3], where plies of agent's decisions are interleaved with plies of possible observations.

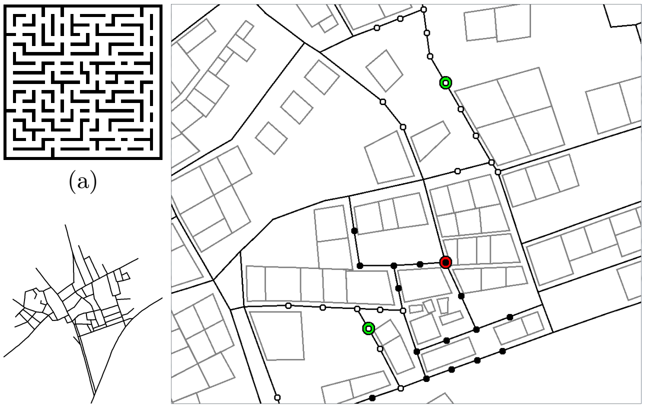
Iterative deepening minimax.

The first algorithm (denoted MM) we use is based on the state of the art technique presented in [3]. It is a depth-limited minimax search with a heuristic evaluation function and the paranoid opponent model. The empirical distribution of computation times of this method with fixed look-ahead depth has a very long tail. In order to meet the anytime requirement, we use iterative deepening and alpha-beta pruning.

Monte-Carlo tree search.

The second algorithm (denoted MC) is MCTS with UCT [2] selection on the same information set tree as in the MM case. The performance of the algorithms was not significantly influenced by the choice of the UCT parameter hence we set it to two in the experiments. We run expansion in each iteration of the algorithm and we select the first child generated for simulation without preference ordering.

We have evaluated several simulation strategies with varying amounts of domain-specific knowledge and cut-off depths. However, consistently with [1], we found that shorter simulations perform better. We achieved the best results when using evaluation functions instead of simulation and back-propagating the returned value in the MCTS tree.



(a) (b) (c)

Figure 1: The environment maps used for experimental evaluation. (a) full maze map used in [3]; (b) a detail of the road-network map with agents visualized as the larger circles, the current set of possible positions of the evader as the black circles, and the positions visible to the pursuer as the white circles; (c) the complete road-network map.

Evaluation functions.

The experimental evaluation in [3] identifies the relaxed lookahead heuristic (RLA_p) as the most successful for the pursuer. RLA_p computes the mean number of positions where the evader can be present after d steps of the game ($d = 10$ in our experiments) and cannot be spotted under any movement of the pursuers. The authors, however, do not define any heuristics for the evader. They assume the worst case behavior of the evader that knows the position of the pursuers all the time in their experiments (E. Raboin 2011, pers. comm. 2 February). In this paper, we aim to achieve realistic behavior of the evader as well. Hence we define RLA_e as the same heuristic computed from the perspective of the evader, i.e., with certain evader's position and uncertain pursuer's positions. We also use a modified version of the evaluation function computed as a sum of the objective value MS and RLA. For the case of evader, MS is the mean of sum of possible positions set sizes of the evaders.

If the set of possible positions of a pursuer is too large (e.g., all the currently unseen positions), it renders all the strategies of the evader almost equally bad. The (paranoid) evader always expects the pursuer to appear just in front of it. Therefore in our implementations, the evader ignores actions of any pursuer that can possibly be at more than a certain number of positions (250 in our scenarios).

3. EXPERIMENTAL EVALUATION

In the experiments, two agents of the pursuer are tracking one evader. The implementation of each player uses only one thread and its computation time is limited to one second on Intel(R) i7 CPU @ 2.80GHz. Each scenario runs for 100 time steps and the results are mean of 100 runs. For initial positions of the game, we follow [3]. We use randomized settings with the evader visible to at least one of the pursuing agents, but far enough form the pursuers to make the tracking difficult.

We use two maps in the experiments. The first is the map from [3] for a fair comparison with the state-of-the-art

pursuers $\rightarrow$	MM(MS+RLA)			MC(RLA)		
evader $\downarrow$	NS$\uparrow$	MS$\downarrow$	ES$\downarrow$	NS$\uparrow$	MS$\downarrow$	ES$\downarrow$
Route-network Map						
MM(MS+RLA)	56.5	89.1	146.3	58.1	88.2	132.8
MC(MS)	63.0	70.6	107.6	71.5	39.6	52.0
Maze Map						
MM(MS+RLA)	58.5	60.3	111.0	56.3	67.0	120.0
MC(MS)	80.4	11.0	17.9	80.7	11.3	17.5

Figure 2: The best Monte-Carlo tree search and iterative deepening minimax approaches. The pursuer maximizes and the evader minimizes the measures marked by $\uparrow$.

algorithm. The topology of the map in form of 50x49 pixels bitmap is presented in Figure 1a. White pixels represent possible position of the agents, black pixels are obstacles and agent can move to the up to four adjacent pixels in one time step. Line-of-sight visibility with Euclidean distance limitation of 10 pixels is assumed.

The second map is based on the topology of a small real-world urban area. Figure 1b presents the overview of the complete road network and Figure 1c is a detail from the center of the map. The road network was discretized as a graph with a node placed every 25 meters, creating 465 nodes. We assume symmetric visibility and the agents can see each other if they are not further than 200 meters form each other and there is no building in their line of sight. An anytime solution is clearly needed with this map. The information set search with fixed lookahead of 8 finishes in less than one second in more than 50% of positions from our experiments, but still takes more than 10 seconds in approximately 3% of cases.

The results in Figure 2 demonstrate that both iterative deepening minimax and MCTS can be used to create good anytime algorithms for the pursuer. Each of them slightly outperforms the other on one of the domains. This is not true for the evader. The minimax-based player is much stronger on the evader's side in both domains. The main difference between the two players in the game is in the amount of uncertainty about the world state and in the branching factor. The decision nodes of the evader represent moves of one player and the decision nodes of the evader represent joint moves of two agents. Furthermore, the number of new nodes that can be observed after a move is also larger for the pursuer. This indicates that, as in perfect information games, minimax-based approaches perform better on games with smaller branching factors and MCTS on games with larger branching factors.

4. ACKNOWLEDGEMENTS

The work presented is supported by the U.S. Army grant no. W15P7T-05-R-P209 (contract BAA 8020902.A), the Grant Agency of the Czech Technical University in Prague, grant No. SGS10/188/OHK3/2T/13 and the Czech Science Foundation, grant No. P202/12/2054.

5. REFERENCES

[1] P. Ciancarini and G. P. Favini. Monte Carlo tree search in Kriegspiel. *Artif. Intell.*, 174:670–684, July 2010.

[2] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *ECML*, pages 282–293, 2006.

[3] E. Raboin, D. Nau, U. Kuter, and S. Gupta. Strategy generation in multi-agent imperfect-information pursuit games. In *AAMAS*, pages 947–954, 2010.