Multiagent Dynamic Optimization: Spiders and Flies

November 10, 2023

1 Information

- Due Date: 20 November 2023
- Upload scan of handwritten solution in Moodle
- Resources: Lecture-21

2 The Problem, [10 Marks]

This example is representative of a broad range of practical problems such as multirobot service systems involving delivery, maintenance and repair, search and rescue, firefighting, etc. Here there are m spiders and several flies moving on a 2-dimensional grid; cf. Fig. 1. **The objective is for the spiders to catch all the flies as fast as possible.**

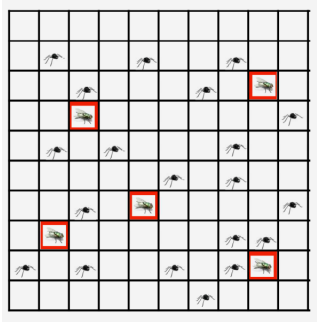

Figure 1: Illustration of a 2-dimensional spiders-and-fly problem with 20 spiders and 5 flies (cf. Example 1.6.4 in book). The flies moves randomly, regardless of the position of the spiders. During a stage, each spider moves to a neighboring location or stays where it is, so there are 5 moves per spider (except for spiders at the edges of the grid). The total number of possible joint spiders moves is a little less than 5^{20}.

During a stage, each fly moves to a some other position according to a given state-dependent probability distribution. Each spider learns the current state (the vector of spiders and fly locations) at the beginning of each stage, and either moves to a neighboring location or stays where it is. Thus each spider has as many as 5 choices at each stage: Left, Right, Up, Down, Stay. The control is $u = (u_1, \ldots, u_m)$, where u^ℓ is the choice of the $\ell-$th spider, so there are about 5^m possible values of u. For the example above, $m = 5$.

To apply multiagent rollout, we need a base policy (Recall travelling salesman problem). A simple possibility is to use the policy that directs each spider to move on the path of minimum distance to the closest fly position. According to the multiagent rollout formalism, the spiders choose their moves one-at-time in the order from 1 to m, taking

into account the current positions of the flies and the earlier moves of other spiders, and assuming that future moves will be chosen according to the base policy, which is a tractable computation.

In particular, at the beginning at the typical stage, spider 1 selects its best move (out of the no more than 5 possible moves), assuming the other spiders $2, \ldots, m$ will move towards their closest surviving fly during the current stage, and all spiders will move towards their closest surviving fly during the following stages, up to the time where no surviving flies remain. Spider 1 then broadcasts its selected move to all other spiders. Then spider 2 selects its move taking into account the move already chosen by spider 1, and assuming that spiders $3, \ldots, m$ will move towards their closest surviving fly during the current stage, and all spiders will move towards their closest surviving fly during the following stages, up to the time where no surviving flies remain. Spider 2 then broadcasts its choice to all other spiders. This process of one-spider-at-a-time move selection is repeated for the remaining spiders $3, \ldots, m$, marking the end of the stage.

Note that while standard rollout computes and compares $5m$ Q-factors (actually a little less to take into account edge effects), multiagent rollout computes and compares ≤ 5 moves per spider, for a total of less than $5m$. Despite this tremendous computational economy, experiments with this type of spiders and flies problems have shown that multiagent rollout achieves a comparable performance to the one of standard rollout.

1. Consider the spiders and flies problem above with two differences: the five flies stay still (rather than moving randomly), and there are only two spiders, both of which start at the fourth square from the right at the top row of the grid of Fig. 1. The base policy is to move each spider one square towards its nearest fly, with distance measured by the Manhattan metric (1-norm, that is only one move out of of left, right, up, down, stay moves allowed), and with preference given to a horizontal direction over a vertical direction in case of a tie. Apply the multiagent rollout algorithm (sequential minimization) mentioned in Slides: lecture-21, and compare its performance with the one of the ordinary rollout algorithm (simultaneous minimization, agent-by-agent), and with the one of the base policy. Identify two more real-life problems where this model can be used.

2. (*Optional) Write a python code to automatically solve this. The code takes number of spiders, number of flies, and grid size as parameters. The code implements both base policy, standard rollout, and multiagent roll out (that is agent-by-agent roll out). Additionally, show the animation and cost for multiple roll-outs.