Multiagent Search Algorithm

Project Report

Kirsten Hagenaars - s1020609
Thomas Kolb - s1027332
Andrea Minichová - s1021688

May 2021

1 Introduction

As metioned in [1], the multiagent search algorithms can be used within environments where a certain target needs to be found. A real life example of such situation might be a rescue mission for missing people inside a forest or in the sea after a shipwreck. In such scenarios, time is often critical, that is why research on how to improve these algorithms is important.

In this assignment, we have implemented three different versions of this algorithm. The first two versons implement a greedy approach, where the agents select their next step based on the knowledge of possible states only 1 step into the future. The first version works without communication. Having multiple agents increases the chance of finding the target faster, but with decentralized communication, the efficiency is much greater as the agents share their common belief and can therefore take actions of other agents into account. Such communication was implemented in the second version. The third version implements continuous agents, which more accurately simulates the real world agents, plus the search is no longer greedy since the multiagents compute their next step based on the knowledge about the future states N steps ahead. In the end, they each take the decision which is collectively the best decision.

2 Multiagent greedy search without communication

In the provided template code, the belief is flattened from a twodimensional matrix to a onedimensional array. We decided to remove the flattening of the matrix since we find it more intuitional and therefore easier to implement. We also decided to remove the id attribute from the Agent class, since it did not serve a purpose in our implementation.

Another thing we decided on was to plot the agents and their belief separately in its own figure, since plotting multiple beliefs in one figure would be difficult to read.

When running the code each agent takes more or less a similar trajectory, they all go through the middle first and then move to the two narrow mountains on the sides. It makes sense that they move this way since it is the most 'greedy' trajectory, and the agents don't influence each other so all end up doing the same thing, depending on their initial coordinates.

3 Multiagent greedy search with a common belief

For this part of the exercise we used the same Agent class as the previous part. To make sure that all of our agents were making decisions using a common belief rather than their own we introduced a new common belief variable in the Environment class. In every iteration of the algorithm we loop through the list of agents. In this loop we assign the common belief to an agent, make a step, update the belief (of that agent) and lastly assign the (updated) belief of the agent to the common belief variable in our Environment class. This way we update the common belief one step at the time for each agent.

Compared to the search without common belief, this algorithm is way more efficient since it allows the agents to communicate their findings, therefore avoiding that multiple agents will search in the same cell.

4 Multiagent search with N-step piece-wise continuous optimization

In each iteration of the algorithm, we find the optimal turn rates for each agent for all N steps ahead. We decided to then let each agent only take the first corresponding step, and not all N steps. This way, every agent sets a single step per iteration, but it does base the corresponding turn rate on N steps into the future. We chose this over taking all N steps because this way each step taken is based on N steps into the future, which wouldn't have been the case otherwise.

To find the optimal turn rates for N steps ahead for each agent, we need to use the optimizer. We can not pass the optimizer an N-by-M matrix (where M is the number of agents) as an initial value since the optimizer does not accept multidimensional input. Therefore we decided to flatten the N-by-M matrix with initial turnrate values before passing it to the optimizer. Since we do prefer working with a true matrix over a flattened version, we reshape the result of the optimizer into an N-by-M matrix for use in the algorithm. Compared to the greedy algorithm with common belief, this algorithm is even more efficient. That is because agents now not only communicate their findings, but they also collectively decide on who searches where based on N steps into the future.

At each iteration, the algorithm optimizes $N*M$ values, where M is the number of agents and N is the number of steps we look into the future. On top of that the optimization is calculated on the turn rate, which is a continuous value. This is way more computationally complex compared to finding the next best state for each agent as we do in the greedy implementations. Since in the greedy approach we are only considering one agents actions, and there are only 9 actions to choose from. We are also looking only 1 step into the future (rather than N).

5 Heterogeneous sensors

In case of heterogeneous sensors, the parameters of the sensor for each agent could be stored inside the agent class. While computing the utility, we could access the parameters applicable to the current agent and compute the probability of non-detecting the target using these parameters.

6 Contribution

Almost all of the work in the project was done with the whole group together. We communicated via Discord where one person would share their screen and we would make all the assignments together and discuss the problems we were facing. This way of working worked well in the last practical assignment, so we decided to do the same thing here.

7 Conclusion

This report discussed three different versions of the mutliagent search algorithm: greedy search without communication, greedy search with a common belief and a search with N-step piece-wise continuous optimization. We have explained the decisions made throughout the implementation and compared the complexities. We also collaboratively wrote the report.

References

[1] Pablo Lanillos, Seng Keat Ganc, Eva Besada-Portasa, Gonzalo Pajaresb, Salah Sukkarieh. Multi-uav target search using decentralized gradient-basednegotiation with expected observation. `https://www.sciencedirect.com/science/article/pii/S0020025514006173`, 2014. [Online; accessed 04-June-2021].