AUTOMATIC PLANNING - REPORT - LAB 3

Manu S. Joseph(mansj125)

Lab 3.1. Using Sequential Planners for Multiple Agents

Two problem files created to test with sequential satisfying planners,

- A small problem with 4 UAV's, 2 Carriers, 4 locations, 4 people, 4 crates and 4 goals
- A larger problem with 4 UAV's 2 Carriers, 8 locations, 10 people, 10 crates and 10 goals

These problem were run using the sequential planners.

For the small problem,

- *Arvand Planner* - The finial plan that was generated used all the 4 UAV's to deliver 4 individual crates. Some of the actions that was grouped together can could be thought of as parallel actions this was not constantly observed even for the small problem.
- *Cerberus, Jasper and lama Planner* - Generated plans that would only use 2 of the four Uav's and most parts of the plan generated cannot be executed in parallel.
- *Madagascar Planner* - Planner generated plans that was best using all 4 UAV's and performed action that be done in parallel together.

For the larger problem,

- *Arvand Planner* - The finial plan generated by Arvand used all the UAV's but failed to fully parallelise all the action possible there could be small batches of parallel tasks that can be seen grouped together.
- *Cerberus, Jasper and lama Planner* - Generated plans that would not use all four UAV's. There was only very few parallel task that could be seen grouped together.
- *Madagascar Planner* - Generated planner generated plans using all 4 UAV's. The plan generated used 3 UAV's in parallel but did necessary actions with the 4th one. The plan was much better for parallel execution when compared to the other planners.

The planners behaviours seem almost similar when we compare with both of the problems. I think the planners would have performed better if it knew of the potential for parallel execution. The planners used multiple UAV's but just did not group together the action that can be done together this might be because of the fact that the planner created a sequence and was not aware of the potential for parallel execution.

The problem files and the output's from running the problems on the planners are present in the folder *ouput_lab3.1*

Lab 3.2. Theory: Temporality and Concurrency

I assume that a locations is large enough for multiple UAV's to pick up crates from the same locations and that it is possible to deliver multiple crates to the same person at the same time. This can be thought to be true because it is have be assumed that a location is large enough for multiple UAV's to work together and delivering a crate is to a similar to delivering to a location. It has also been assumed that multiple UAV's will not be able to work with a carrier at the same time since it would not have the space to do so.

Based on the assumption the following actions cannot be done in parallel,

- Multiple UAV's cannot be pick the same crate at the same time.
- Multiple UAV's cannot deliver the same crate at the same time.
- Multiple UAV's cannot load crate on the same carrier at the same time (will not have the space to do it).
- Multiple UAV's cannot take crate from the same carrier at the same time (will not have the space to do so).
- Multiple UAV's cannot fly the same carrier at the same time
- A single UAV cannot do multiple actions at the same time.

Lab 3.3. Generating Plans for the Rover Domain

I ensured that the rovers does not send multiple data's at the same time by introducing a new predicate called *rover_free.* In the send action *rover_free* was made false at the start of the action and made true at the end of the action. The preconditions of the send action made sure that the action can be done only if the predicate *rover_free* is true.

The outputs from running the planner with the problem provided and the modified rover domain is given in the folder *output_lab3.3*

Lab 3.4. Concurrency in Emergency Service Logistics

The UAV domain was modified to handle concurrency. The *increase* effects were removed and *durations* were added. A new predicate called *heli_not_busy* was added to check if the helicopter is free and is currently not doing any task. This was required so as to restrict an UAV from doing multiple tasks at the same time. The *fly-cost* that was introduced from the last lab was used for handling the duration of the *fly_to* and *fly_carrier* action, other actions were given constant durations.

Actions that were not possible to be executed in parallel were handled using appropriate precondition and effects. Comments have been added to each effect and precondition of the actions, for more information check the *domain.pddl* file.

The domain was tested with the temporal planners and the following problems were solved by the planners in 1 minute.

ITSAT - 4 UAV's, 2 Carrier, 8 locations, 10 crates, 10 people and 10 goals.
Fast Downward - 4 UAV's, 2 Carrier, 8 locations, 8 crates, 8 people and 8 goals.
YAHSP3 - 25 UAV's, 5 Carrier, 20 locations, 50 crates, 30 people and 50 goals.

Even through YAHSP3 planner was able to solve very large problem under 1 minute the initial plans generated were pretty bad with very little concurrency but better plans were generated later on. ITSAT seemed to produced the best initial plans. Fast downward planner generated plans that were almost as good as ITSAT but it was slower when compared to ITSAT.

The planners were also tested with varying age different parameters,

- Increasing number of UAV's

    - *ITSAT* - Increase in 2 UAV's increased the time in finding solution by almost 10 seconds.
    - *Fast Downward* - Increase in 2 UAV's increased the time in finding solution by almost 40 seconds.
    - *YAHSP3* - Increase in 5 UAV's increased the time in finding solution by almost 13 seconds.
- Increasing number of People
    - *ITSAT* - Increase in 2 People increased the time in finding solution by almost 280 seconds.
    - *Fast Downward* - Did not increase the time to find solution
    - *YAHSP3* - Did not increase the time to find solution
- Increasing number of Locations
    - *ITSAT* - Increase in 2 locations added increased the time in finding solution by almost 110 seconds.
    - *Fast Downward* - Increase in 2 locations increased the time in finding solution by almost 50 seconds.
    - YAHSP3 - Increased the time to find solution by a very little amount. (Increase in 5 UAV's increased the time in finding solution by almost 4 seconds).
- Increasing number of Crates
    - *ITSAT* - Increase in 2 crates increased the time in finding solution by almost 190 seconds.
    - *Fast Downward* - Did not increase the time to find solution
    - *YAHSP3* - Increase in 5 crates increased the time in finding solution by almost 32 seconds.

The ITSAT planner was the one that was mostly effected drastically by any change in parameters. It was most affected by the increase in number of people. The Fast downward planner was most affected by the change in number of locations. The YAHSP3 planner was most affected by the change in number of crates.

The problems file that were generated for testing the planners are provided in *output_lab3.4/Problem_files* folder. The problems that were finished by the planners in 1 minute are denoted by the files starting with names - *fdUav_problem*… , *itsatUav_problem*…. and *yahspUav_problem*… the corresponding increased problems are also present in that folder. The outputs from running those planners are present in the *output_lab3.4* folder.