Experiment

-Yixi rao u6826541

Exercise 10

1 Rule I

Rule 1 is if a package was at its destination, then it cannot leave. The motivation is that after reaching the goal location, moving the package is meaningless and it wastes lots of time to consider this situation. It can reduce the search space because it let some clauses that claim the arrived packages at another location to be false.

2 Rule II

Rule 2 is invariant control knowledge, which is a truck (or an airplane) appears at one and only one location (or airport) in a time step. The motivation is the physical property that for every object it only can appear at one location at the same time, which indicates that for every time step, each truck (or airplane) can only at one location, if it appears in others location then it will be false. This control knowledge can reduce tremendous amount of search space since it declares only one “at planeX locX” proposition to be true and all others “at planeX locX” propositions to be false.

3 Rule III

Rule 3 is do not unload a package from an airplane if the airplane is not in the package’s goal city, which means that an airplane must carry a package and the landing city is not the package’s goal city. I got the inspiration from the Y.-C. Huang, B. Selman, H. Kautz paper [1], it is a category I control rule (Control that involves only static information derivable from the initial state and goal state) and which can prune nodes in this planning graph [1]. The motivation is that if an airplane carries a packet then it cannot unload it until it reaches the package’s city, if it unloads somewhere not the goal’s city, it is inefficient to carry it again onto the airplane.

4 Experiment

Total time

Problem	***Basic Encoding (s)***	***With Control Knowledge (s)***
logistics03	*8.47028*	*1.68022*
logistics04	*14.1447*	*1.98169*
logistics07	*1.72532*	*1.61796*
logistics09	*35.9430*	*6.82274*

solution time

Problem	*Basic Encoding (s)*	*With Control Knowledge (s)*
logistics03	*7.36459*	*0.53857*
logistics04	*12.7515*	*0.63237*
logistics07	*0.74474*	*0.58726*
logistics09	*19.4403*	*1.23054*

Exercise 11

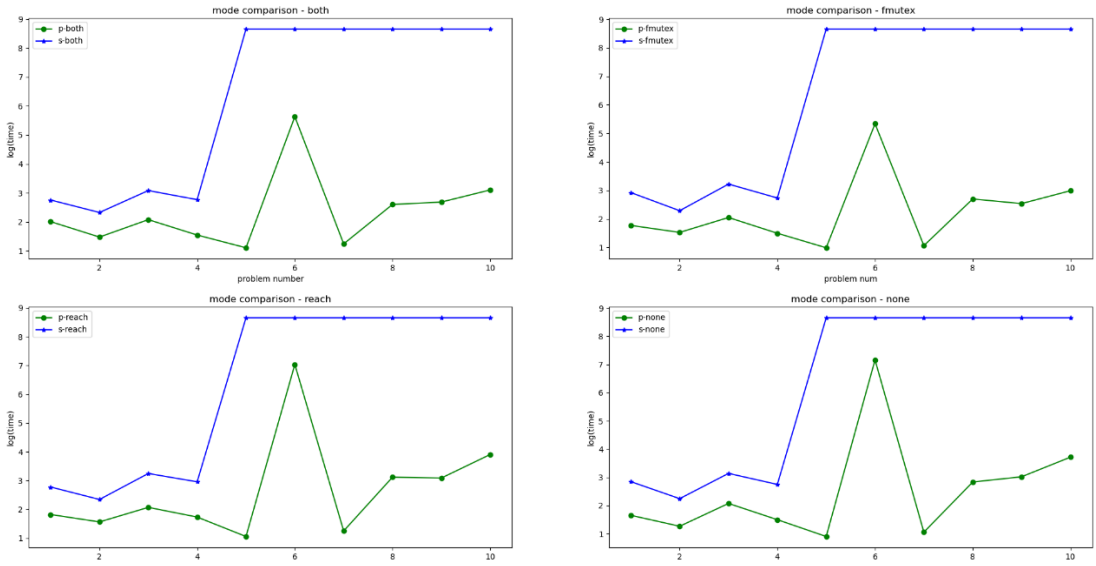
The three domains I select the rover's domain, blocks' domain, and the miconic's domain, and I decide to conduct two experiments which is the implication of different execution semantics comparison experiment and the implication of the different constraints experiment. The experiment criterion is the total planning time, if the planner cannot solve the problem within 100s, the total planning time will be 101s. To show the difference obviously, the logarithm is used to the total time because some problem can be solved within 0.1s while some big problem will use 60-90s, the huge difference makes it hard to do the visualization.

1 serial vs parallel

1.1 Rover's domain

The graph shows the result of the two execution semantics which is the serial mode and parallel mode. Theoretically, the serial mode will be slower than the parallel mode because serial mode only allows at most one action can be executed per time step, as a result, the serial will need more steps to solve the planning problem which means it needs more total time.

On the experiment, the result shows that the parallel mode planning spends less time than the serial planning because almost every green line (parallel) is below the blue line which means it uses less time. And another observation should be noticed is that the serial planning cannot find the plan within 100s after problem 5, but the parallel mode can solve the problem within 5s. for the problem 6 there is a outlier, I check the result script, and find some abnormal behaviors, which is the planner spends about 30s in step 18 to prove the problem is UNSAT, but when the horizon increases to 20, the planner can find the plan in 3s, this maybe is the problem-specific problem because when I test it in 16 horizons case, it spends approximately 3s.

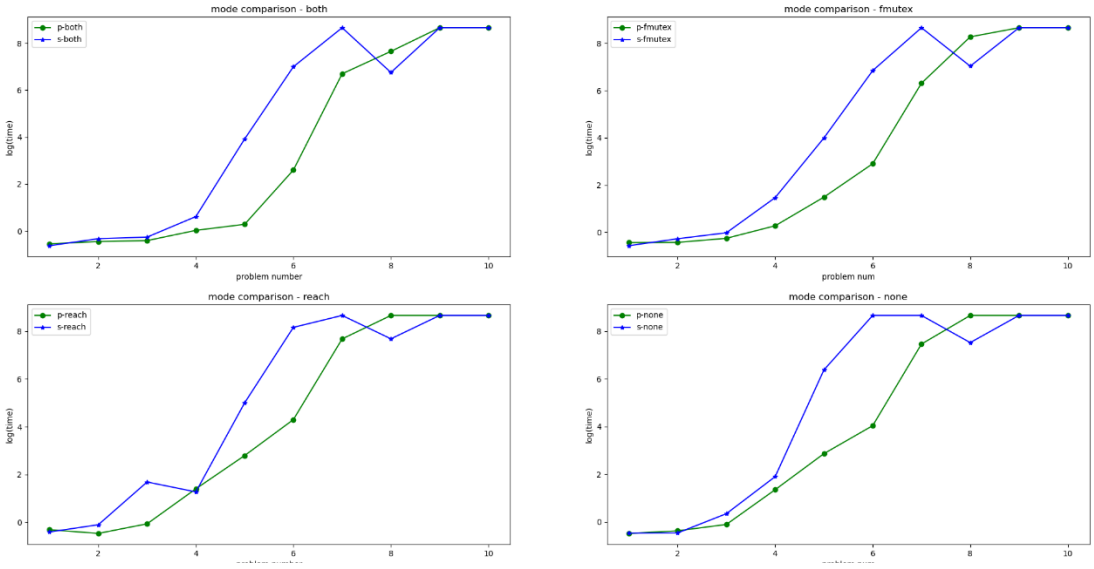


Rover domain

1.2 Miconic's domain

In this domain, the experiment result also corresponds to the theoretical result, and almost all the problem can be solved by using the serial semantic, I think this is because the miconic's domain is affected by the serial mode very little, as only the elevator can move.

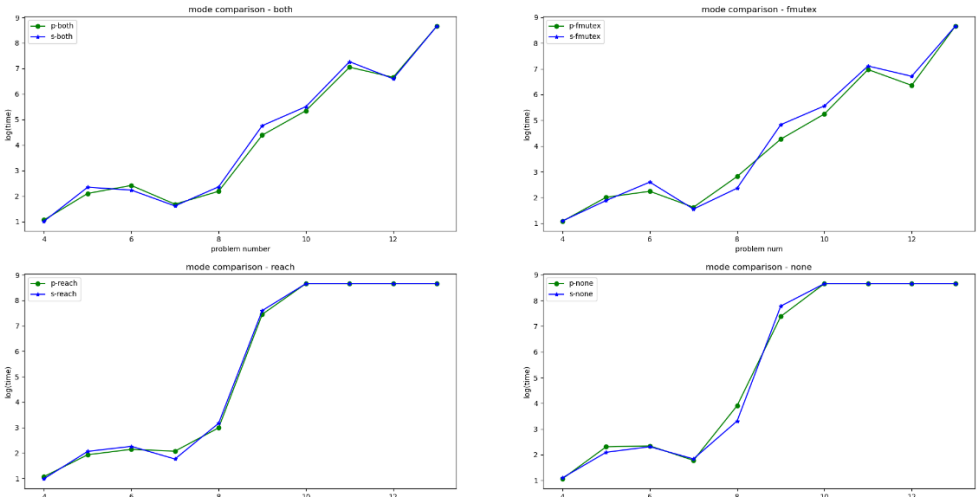
there is also a special abnormal value in problem 8, when I check the problem description, I find that almost all the passages are on different floor, so using the serial mode is more efficient.



1.3 Block's domain

In this domain, the experiment result also corresponds to the theoretical result but the

difference is not obvious.



2 constraints comparison

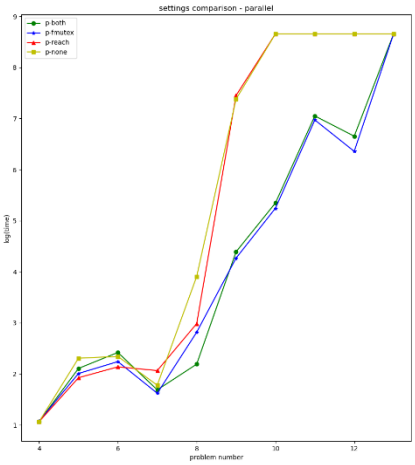
The graphs show the comparion Comparison between these four configurations (fmutex, reachable, both, -p false), with good plangraph constaints many problems can become easier to solve, the plangraph constaints is used to obtain plans of better quality by constraining the search space. The fluent mutex clauses can usually make planning faster by causing the SAT solver to backtrack earlier, and the reachable actions may improve the performance.
The green line represents the “both” configuration, the blue line represents the “fmutex” configurations and the red line represents the “reach” configuration, the yellow line represents “none” configuration. See the three graphs below

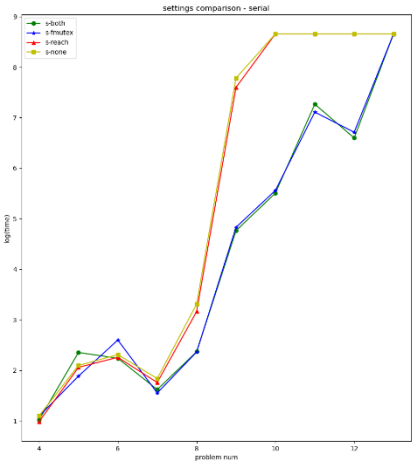
- In the three experiments, we can conclude that using the plangraph constraints can improve the performance, it can solve some big problem that the planner without this constraints cannot solve it.

- I also find that “both” and “fmutex” configurations are better than “none” and “reach” configurations, I think this is due to the strong fmutex constraint which adds fluents mutex clause to restrict the future actions.

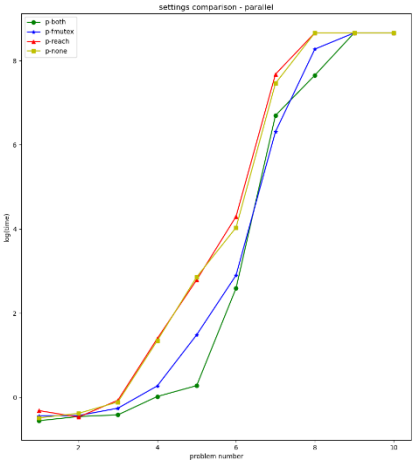
- Another finding is that “both” better than the “fmutex”, The reason is obvious, more constraints is not harmful. Also, “reach” better than “none”.

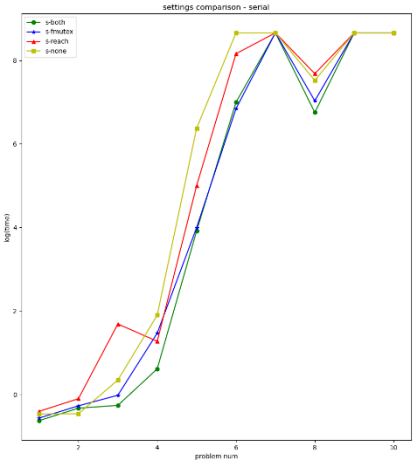
- The total time dramatically increases after the problem 5 and it is like an exponential increase



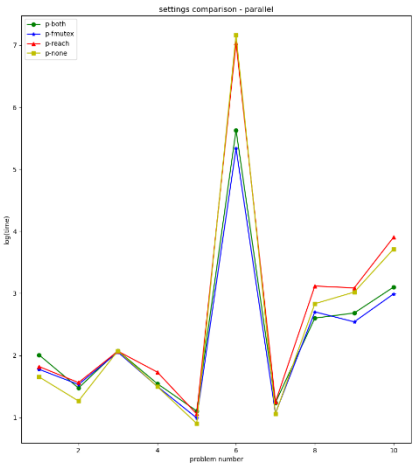


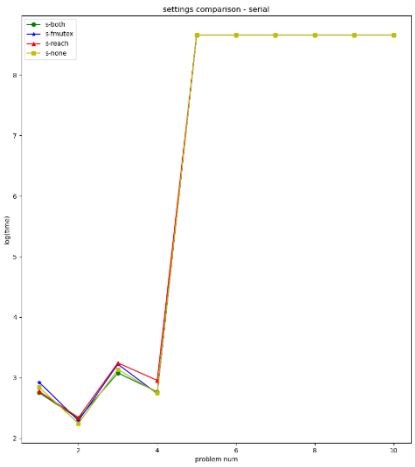
blocks' domain





Miconic's domain





Rovers' domain

3 query strategy

In order to make it easier/quicker to find plans, the query strategy should be considered, we hope the planner uses the 100s to find a plan rather than to prove the problem with a specific horizon is UNSAT. First, I try the serial query strategy ("1:30:1"), some small problems can be solved within 1s and use less than 10 steps. However, it spends a lot of time to prove the problem is UNSAT for some big problems, and it is very time-consuming. So, I do some experiments to find what is the minimal steps to solve the problem, and results show that I can divide it into three categories:

- Small problem: The first four or five questions (e.g. problem01.pddl to problem05.pddl). This problem can be solved within 1s and uses at most 10 steps, so I decide to use the "1:30:1" query strategy.
- Medium problem: problem six to eight. These problems probably require more than 15 or 20 steps to solve the problem, and if it still uses the "1:30:1" query strategy, it will waste time on proving 17,18,19 is UNSAT. So, I decide to use the "16:32:2" query strategy.
- Big problem: like the last two problems, it always takes more than 100 seconds to find the solution, and using 20 or more steps. So, I decide to use the "30:60:5" query strategy.

REFERENCE

[1] Y.-C. Huang, B. Selman, H. Kautz Control knowledge in planning: benefits and tradeoffs Proc. AAAI-99, Orlando, FL (1999), pp. 511-517