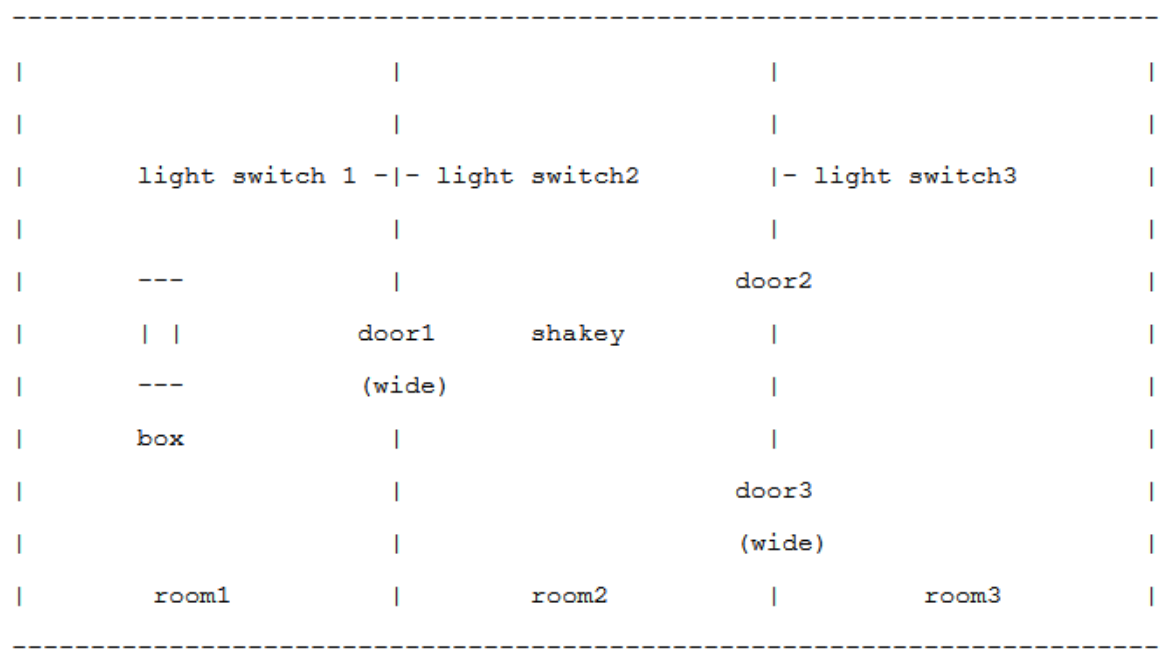
Mikaël Salasca (miksa599)
Jonas Michelberger (jonmi043)

Report TDDC17-Lab4

Task 1:

For task 1 we chose alternative 2: Shakey's World. Below you find the graphic illustration of the problem domain. Shakey has to turn on the lights, while the box hast to be in the same room. To push a box from one room to another the door hast to be wide. The goal of our domain is to turn on all the lights in every room. The world is completely dark at the beginning, no light is turned on. The lights are modelled as predicates of the object room. Further explanations regarding the concrete design of our problem domain implementation can be found in the commented .pdll files.



Task 2:

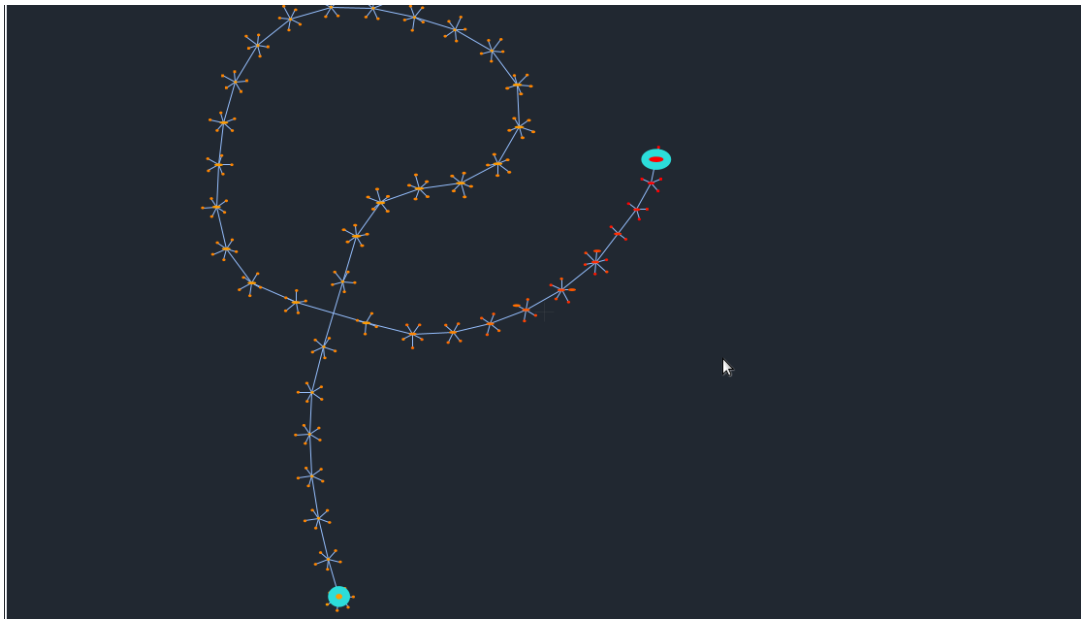
1. **Comparing heuristics, problem 03:**

    2. **Step both visualisations to time step 40 manually or through auto-stepping. Don't use the "go to" functionality (entering a step number) because you will need to see nodes as they appear!**

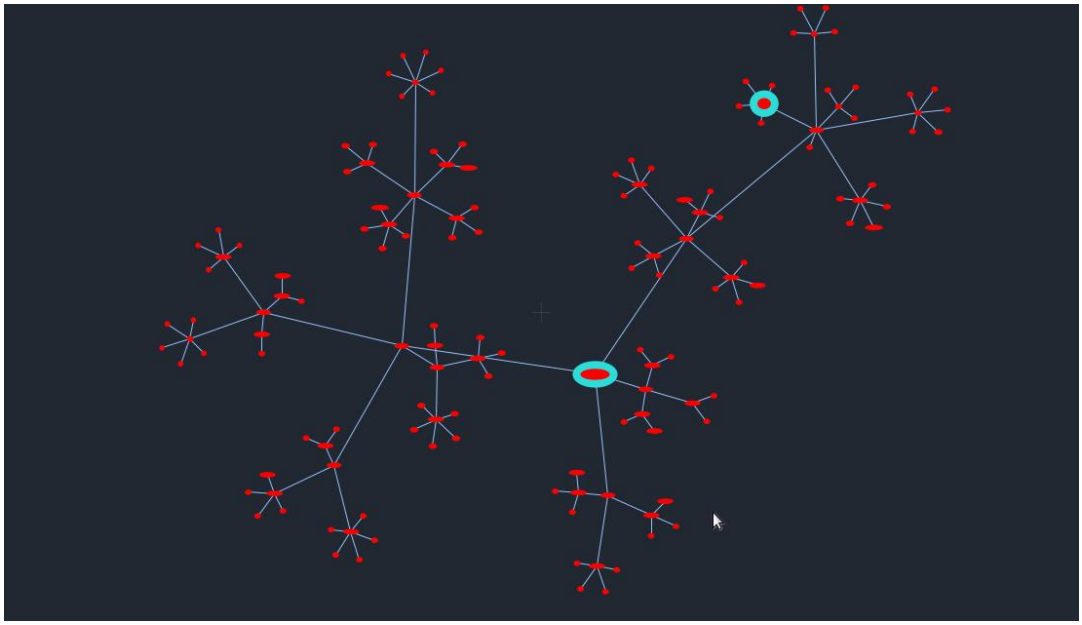
       How do the graphs differ, given the same problem and search method but different heuristics? Include screenshots in your report and discuss what you see!

       The FF heuristic is going down one path, it chooses the expanded-but-not-visited child node with the lowest heuristic value (see colouring by heuristic value).

       By using eager search, it will compute the heuristic value of the state when a new state is created.



The goal count heuristic is picking the expanded-but-not-visited nodes with the minimum number of facts left to achieve and therefore is going more in breadth than depth at the beginning.



3. **Do the different configurations use different actions? Zoom in on the edges to see which actions are used and discuss the differences.**

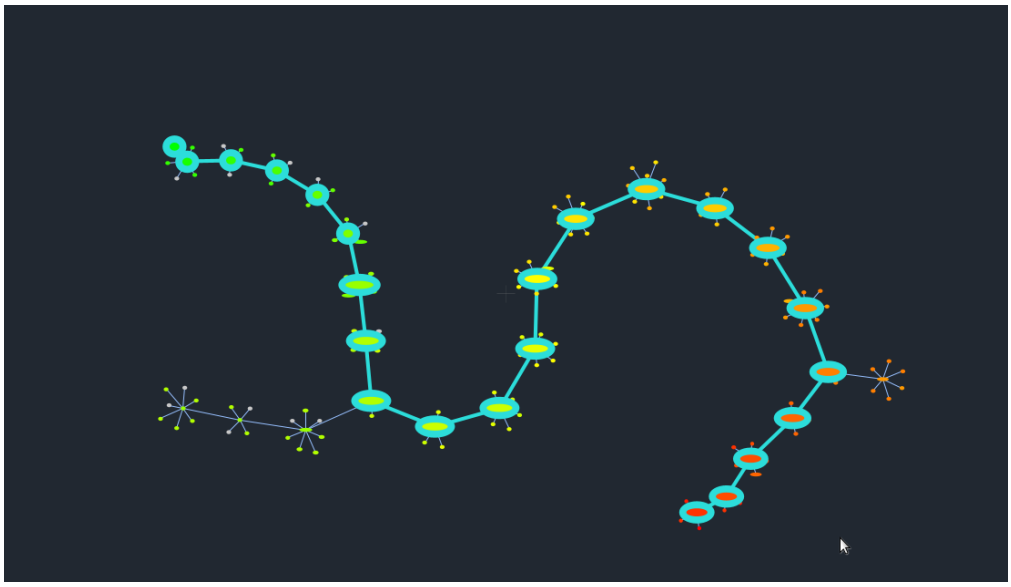
   For the first 40 steps: FF uses drive, load and unload. Goal count uses only drive and load. This is due to the preference of goal count to go in breadth first (in this problem), therefore it does not get the chance to unload within the first 40 steps. For termination: Both use the same actions: drive, load, unload.

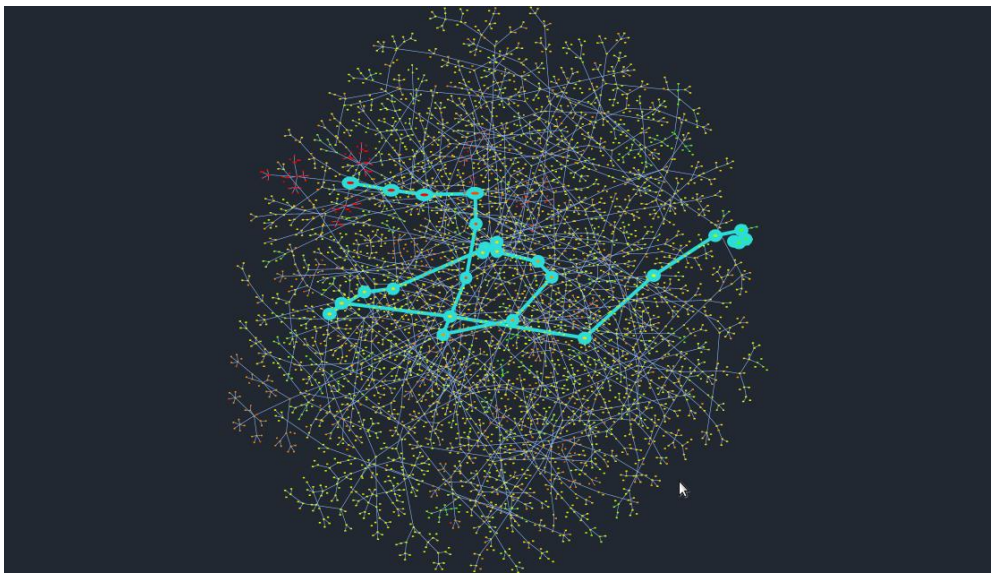
4. **Step the FF visualisation to the end through auto-stepping (hint: set the Auto-steps per second to a value greater than 1 but small enough so that you can follow the search process). Don't use the go to functionality.**

At the beginning, the FF visualisation expanded nodes along a long path before it started expanding at other places. Are the nodes in this path used in the final plan? Hint: the nodes are numbered incrementally as they are expanded so this can be used to find the answer.

The nodes that were expanded in the beginning (long path) are used up to a certain point in the final plan.

2. **Comparing heuristics, problem 02:**
    2. **Start two instances of the Fast Downward visualization, using the log files generated when solving problem 02 with (a) the configuration that uses FF-heuristic and (b) the configuration using GC heuristics with preferred operators from FF.**





    3. **When the planner adds a new action to a branch in the search tree, this doesn't necessarily cause the heuristic function to decrease. In which time step *does* the planner first find a new lower value for the main heuristic function in each example (FF and GC, respectively)?**

For FF this happens with the first step, when the first parent node is expanded. For GC this happens in Step 19 (Goal count =6):

4. **Step both visualisations to time step 27 (which is the last time step for the configuration using FF as heuristic, but not for the goal count configuration).**

   How many goal facts are left to achieve according to the goal count heuristic (see the description of *Eager greedy search, Goal count heuristic*) in the state that has progressed the longest?

   In time step 27 6 goal facts are left to achieve according to the goal count heuristic.

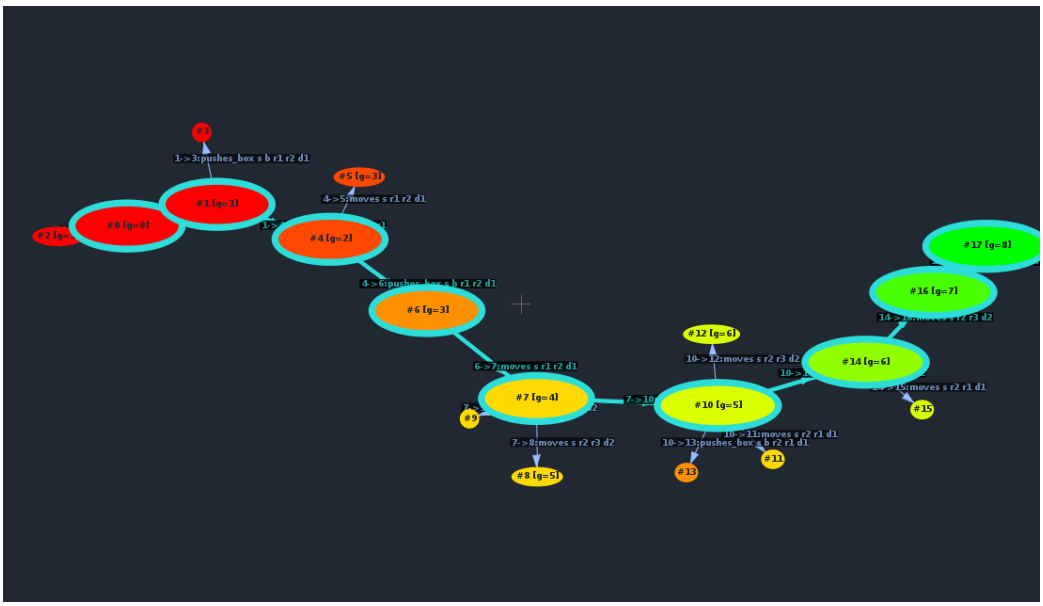
5. **Auto-step the visualisation for the goal count heuristic to the end.**

   Follow the marked path from the start node (the initial state) to the last node (the first found goal state) in the highlighted path in the graph (the plan). Does the solution ever increase the value of the goal count heuristic between one state and the next?

   Yes, the value of the goal count heuristic increases at some steps from 2 to 4 and from 3 to 4 between one state and the next.

3. **Run one of the configurations on your own domain and problem. Is the graph similar to any of those that you have examined above? Can you explain why or why not?**

We applied FF on our domain and problem. Basically, it looks structurally very similar. The graph is simpler and has fewer nodes which is due to our less complex domain / problem. The planner is starting at a parent node and from there expanding the child nodes with the goal to reduce the heuristic function by going one specific path which brings it closer to the goal.



The statistics from using the different configurations to solve the given problems. Was any configuration better than the other? Was it better on everything or just on some problems? Are your findings applicable to all the infinite many possible domains and problems?

Which search works best depends on the specific problem. We observed that for problem 03 GC outperformed FF, while for problem 02 it was the other way round. One cannot derive a general conclusion because it is highly sensitive to the setup and problem.