Lab 4 TDDC17

Oskar Håstad, Martin Hinnerson

Version 0.1

2018-10-29

Status

Reviewed		
Approved		

Project identity

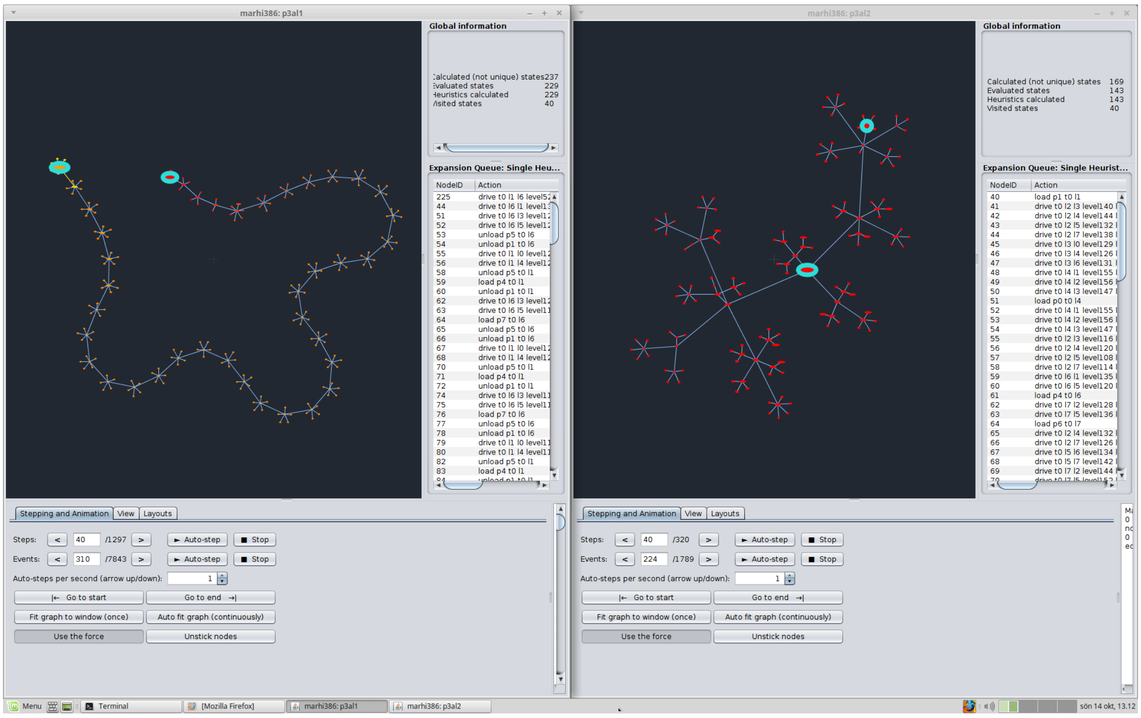
2018/Autumn term
Tekniska högskolan vid Linköpings universitet, IDA

Name	Telephone	Email
Oskar Håstad		oskha905@student.liu.se
Martin Hinnerson	072 210 43 72	marhi386@student.liu.se

Course responsible: Patrick Doherty, patric.doherty@liu.se
Lab Assistant: Axel Strandberg, axest863@student.liu.se

1 Part 1: Comparing heuristics, problem 03

Here we run the Fast Downward search visualizer twice, using either an FF-heuristic or a Goal Count heuristic. After stepping both visualizations 40 steps we see a big difference in choice of expanded nodes. In the left of figure 1 we clearly see that the fast forward heuristic prioritize further expansion. Since we are using a Fast Downward search we can look at the left search tree in fig 1 as following the slope down in one direction until we hit a local (or global) minima. The right figure is when using the Goal count heuristic. This heuristic makes the search more evenly spread out which gives it a greater chance to find a better solution but will in most cases need more expansions to find one.



Global information

```
:alculated (not unique) states237
:valuated states               229
Heuristics calculated          229
/isited states                 40
```

Expansion Queue: Single Heu...

NodeID	Action
225	drive t0 l1 l6 level52
44	drive t0 l6 l1 level13
51	drive t0 l6 l3 level12
52	drive t0 l6 l5 level12
53	unload p5 t0 l6
54	unload p1 t0 l6
55	drive t0 l1 l0 level12
56	drive t0 l1 l4 level12
58	unload p5 t0 l1
59	load p4 t0 l1
60	unload p1 t0 l1
62	drive t0 l6 l3 level12
63	drive t0 l6 l5 level11
64	load p7 t0 l6
65	unload p5 t0 l6
66	unload p1 t0 l6
67	drive t0 l1 l0 level12
68	drive t0 l1 l4 level12
70	unload p5 t0 l1
71	load p4 t0 l1
72	unload p1 t0 l1
74	drive t0 l6 l3 level11
75	drive t0 l6 l5 level11
76	load p7 t0 l6
77	unload p5 t0 l6
78	unload p1 t0 l6
79	drive t0 l1 l0 level11
80	drive t0 l1 l4 level11
82	unload p5 t0 l1
83	load p4 t0 l1
84	unload p1 t0 l1

Steps: 40 /1297
Events: 310 /7843
Auto-steps per second (arrow up/down): 1

Global information

```
Calculated (not unique) states   169
Evaluated states                 143
Heuristics calculated            143
Visited states                   40
```

Expansion Queue: Single Heurist...

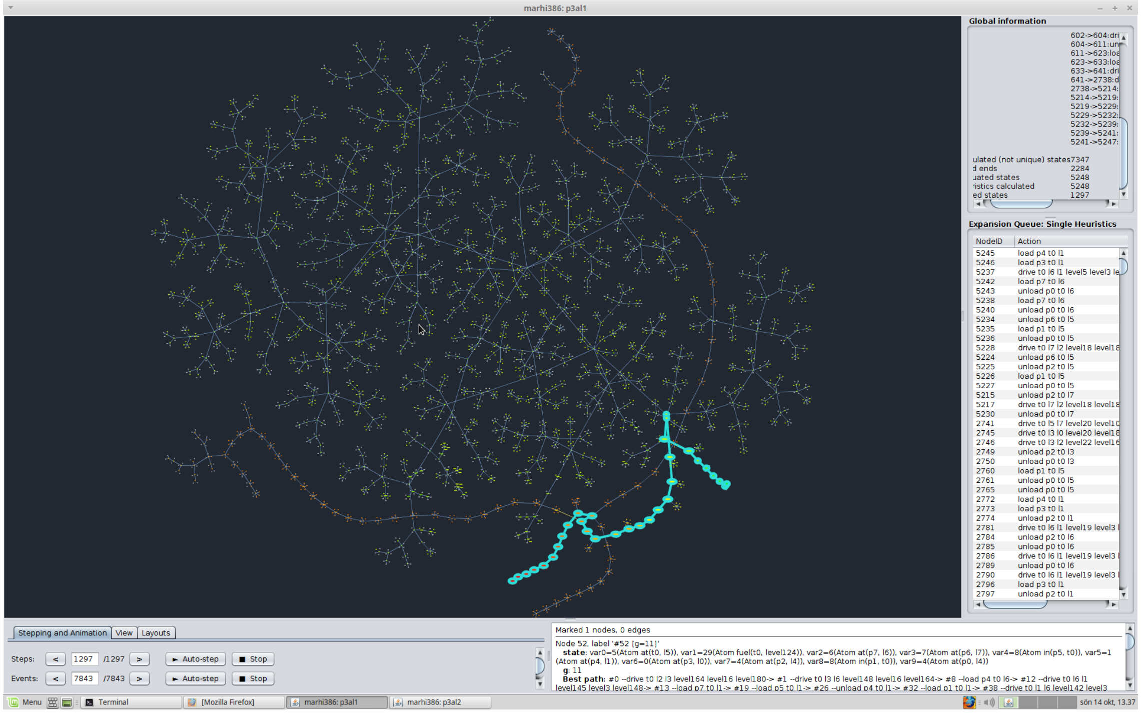
NodeID	Action
40	load p1 t0 l1
41	drive t0 l2 l3 level140 l
42	drive t0 l2 l4 level144 l
43	drive t0 l2 l5 level132 l
44	drive t0 l2 l7 level138 l
45	drive t0 l3 l0 level129 l
46	drive t0 l3 l4 level126 l
47	drive t0 l3 l6 level131 l
48	drive t0 l4 l1 level155 l
49	drive t0 l4 l2 level156 l
50	drive t0 l4 l3 level147 l
51	load p0 t0 l4
52	drive t0 l4 l1 level155 l
53	drive t0 l4 l2 level156 l
54	drive t0 l4 l3 level147 l
55	drive t0 l2 l3 level116 l
56	drive t0 l2 l4 level120 l
57	drive t0 l2 l5 level108 l
58	drive t0 l2 l7 level114 l
59	drive t0 l6 l1 level135 l
60	drive t0 l6 l5 level120 l
61	load p4 t0 l6
62	drive t0 l7 l2 level128 l
63	drive t0 l7 l5 level136 l
64	load p6 t0 l7
65	drive t0 l2 l4 level132 l
66	drive t0 l2 l7 level126 l
67	drive t0 l5 l6 level134 l
68	drive t0 l5 l7 level142 l
69	drive t0 l7 l2 level144 l
70	drive t0 l7 l5 level152 l

Steps: 40 /320
Events: 224 /1789
Auto-steps per second (arrow up/down): 1

Figure 1: Problem 03 after 40 steps

We can see that the configuration use different actions. The GC-heuristic configuration tends to for example try out all `drive` actions and then do a `load` while the FF-heuristic has a more random pattern.

In figure 2 the FF visualization has been stepped to the end using auto-stepping. In our case the visualization uses 11 steps of the initial path in the final plan.



Global information

```
602->604:dri
604->611:un
611->623:loa
623->633:loa
633->641:dri
641->2738:d
2738->5214:
5214->5219:
5219->5229:
5229->5232:
5232->5239:
5239->5241:
5241->5247:

ulated (not unique) states7347
d ends                    2284
uated states              5248
ristics calculated        5248
ed states                 1297
```

Expansion Queue: Single Heuristics

NodeID	Action
5245	load p4 t0 l1
5246	load p3 t0 l1
5237	drive t0 l6 l1 level5 level3 le
5242	load p7 t0 l6
5243	unload p0 t0 l6
5238	load p7 t0 l6
5240	unload p0 t0 l6
5234	unload p6 t0 l5
5235	load p1 t0 l5
5236	unload p0 t0 l5
5228	drive t0 l7 l2 level18 level18
5224	unload p6 t0 l5
5225	unload p2 t0 l5
5226	load p1 t0 l5
5227	unload p0 t0 l5
5215	unload p2 t0 l7
5217	drive t0 l7 l2 level18 level18
5230	unload p0 t0 l7
2741	drive t0 l5 l7 level20 level10
2745	drive t0 l3 l0 level20 level18
2746	drive t0 l3 l2 level22 level16
2749	unload p2 t0 l3
2750	unload p0 t0 l3
2760	load p1 t0 l5
2761	unload p0 t0 l5
2765	unload p0 t0 l5
2772	load p4 t0 l1
2773	load p3 t0 l1
2774	unload p2 t0 l1
2781	drive t0 l6 l1 level19 level3 l
2784	unload p2 t0 l6
2785	unload p0 t0 l6
2786	drive t0 l6 l1 level19 level3 l
2789	unload p0 t0 l6
2790	drive t0 l6 l1 level19 level3 l
2796	load p3 t0 l1
2797	unload p2 t0 l1

```
Marked 1 nodes, 0 edges
Node 52, label '#52 [g=11]'
  state: var0=5(Atom at(t0, l5)), var1=29(Atom fuel(t0, level124)), var2=6(Atom at(p7, l6)), var3=7(Atom at(p6, l7)), var4=8(Atom in(p5, t0)), var5=1
(Atom at(p4, l1)), var6=0(Atom at(p3, l0)), var7=4(Atom at(p2, l4)), var8=8(Atom in(p1, t0)), var9=4(Atom at(p0, l4))
  g: 11
  Best path: #0 --drive t0 l2 l3 level164 level16 level180-> #1 --drive t0 l3 l6 level148 level16 level164-> #8 --load p4 t0 l6-> #12 --drive t0 l6 l1
level145 level3 level148-> #13 --load p7 t0 l1-> #19 --load p5 t0 l1-> #26 --unload p4 t0 l1-> #32 --load p1 t0 l1-> #38 --drive t0 l1 l6 level142 level3
```

Figure 2: Problem 3 with FF when finished

2 Part 2: Comparing heuristics, problem 02

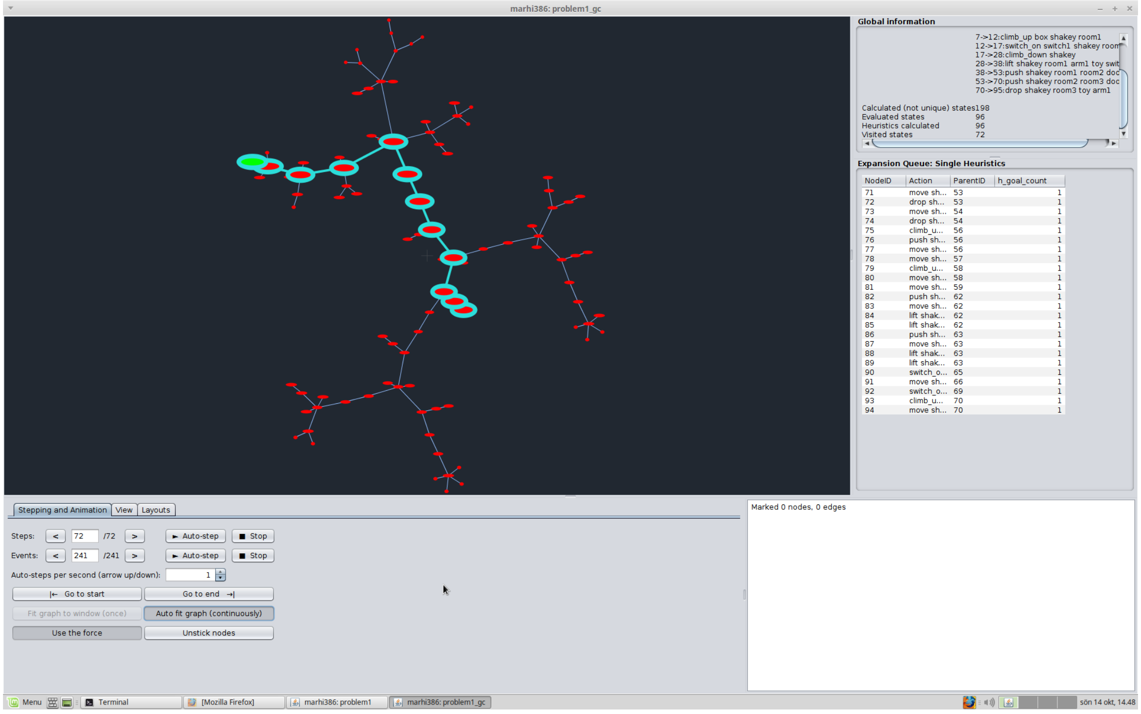
We now run problem 02 in the Fast Downward visualization. The configurations used now is FF-heuristic and GC-heuristics with preferred operators from FF.

When running the FF the planner first finds a new lower value for the heuristic function in time step 3. Here the value of the heuristic decrease from 17 to 16. When running GC with preffered operators the planner first finds a new lower value at time step 19. Her the value of the heuristic decrease from 7 to 6. We found that the easiest way to see this was by looking in the expansion queue on the right side of the simulator.

After stepping both visualizations 27 steps the FF configuration is finished while the GC configuration still have 6 goal facts to achieve.

When the GC configured simulation has finished we can see that the value of the goal count heuristic never increase between one state and the next.

3 Visualization of own domain



```
7->12:climb_up box shakey room1
12->17:switch_on switch1 shakey room
17->28:climb_down shakey
28->38:lift shakey room1 arm1 toy swit
38->53:push shakey room1 room2 doo
53->70:push shakey room2 room3 doo
70->95:drop shakey room3 toy arm1

Calculated (not unique) states198
Evaluated states              96
Heuristics calculated         96
Visited states                72
```

NodeID	Action	ParentID	h_goal_count
71	move sh...	53	1
72	drop sh...	53	1
73	move sh...	54	1
74	drop sh...	54	1
75	climb_u...	56	1
76	push sh...	56	1
77	move sh...	56	1
78	move sh...	57	1
79	climb_u...	58	1
80	move sh...	58	1
81	move sh...	59	1
82	push sh...	62	1
83	move sh...	62	1
84	lift shak...	62	1
85	lift shak...	62	1
86	push sh...	63	1
87	move sh...	63	1
88	lift shak...	63	1
89	lift shak...	63	1
90	switch_o...	65	1
91	move sh...	66	1
92	switch_o...	69	1
93	climb_u...	70	1
94	move sh...	70	1

Steps: < 72 /72 > ► Auto-step ■ Stop
Events: < 241 /241 > ► Auto-step ■ Stop
Auto-steps per second (arrow up/down): 1
|← Go to start | Go to end →|
Fit graph to window (once) | Auto fit graph (continuously)
Use the force | Unstick nodes

Figure 3: Visualization of Problem 1 in the shakey domain with GC heuristics

Figure 3 shows the visualization of Problem 1 on in the Shakey Domain. We can see similarities to the graph when running problem 03 with GC-heurisitcs. One big difference is that we have more branches where there are only one new possible action after every step and most of them end up in a dead end. One reason for this is that our domain is much smaller that the one in problem 3.

4 Conclusions

In table 1 we can see the cost of the solution for the different algorithms. In the case of problem 2 the goal count heuristic creates a slightly cheaper plan. This goal is also found faster than when using FF.

With problem 2, GC again creates a cheaper plan than FF. However it takes GC 1575 steps to find a plan while FF only needs 27. With problems like this it might be worth using the faster planner even though it creates a worse plan. This is called Satisficing Planning.

Problem	Algorithm	Cost	Iterations
3	FF	31	1297
3	GC	30	320
3	GC with FF help	30	249
2	FF	22	27
2	GC	20	12222
2	GC with FF help	20	1575

Table 1: Cost of problems using the different algorithms