Foundations of Artificial Intelligence

M. Helmert, T. Keller
S. Eriksson
Spring Term 2020

University of Basel
Computer Science

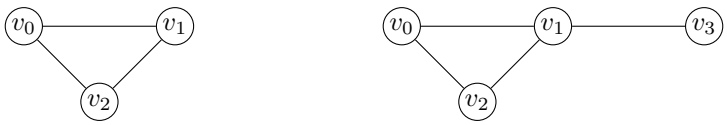
Exercise Sheet 10

Due: May 6, 2020

Important: For submission, consult the rules at the end of the exercise. Non-adherence to the rules will lead to your submission not being corrected.

Exercise 10.1 (1+1.5+1.5+2 marks)

Consider the following variant of the *Seven bridges of Königsberg* problem: for a given set of bridges that cross a river, is there a tour that starts and ends at the same location and crosses each bridge exactly once? Formally, the problem can be defined as follows: given a graph $G = (V, E)$ with a set of vertices V and a set of edges $E \subseteq V \times V$ and an initial vertex $v_0 \in V$, is there a sequence of vertices from V such that i) all pairs of subsequent vertices are connected by an edge from E, ii) each edge in E occurs exactly once in the sequence, and iii) the first and last vertex of the sequence is v_0? For an illustration of the problem, consider the following examples:



The initial vertex is v_0 in both cases. For the graph on the left side, there is such a tour, e.g., $\langle v_0, v_1, v_2, v_0 \rangle$, while there is no such tour for the graph on the right side.

(a) You can find a PDDL description of the (original instance of the) *Seven bridges of Königsberg* problem on the website of the course. The domain description (variables and actions) is given in the file `bridges-once-domain.pddl`, and the problem description (objects, initial state and goal description) is given in the file `koenigsberg-problem.pddl`. Provide a graphical representation of the problem in the same way as the example above. Please do not forget to mark which location is the initial location.

(b) Obtain the domain-independent planning system *Fast Downward* from

`http://www.fast-downward.org/Releases/19.12`.

Use *Fast Downward* with a configuration that performs greedy best-first search with the delete relaxation heuristic FF to solve the Seven bridges of Königsberg problem from the website. To do so, invoke the planner with

```
./fast-downward.py bridges-once-domain.pddl koenigsberg-problem.pddl
--search "eager_greedy([ff()])"
```

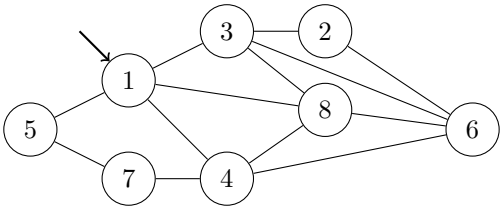
Provide the runtime and the number of expanded states. Discuss whether the solution is optimal. Is the problem solvable? If it is, provide the plan that was found.

(c) Modify the domain description (`bridges-once-domain.pddl`) such that it is possible to use the same bridge more than once. Solve the resulting problem with the same Fast Downward configuration that was used in part (b) of this exercise.

Provide the runtime and the number of expanded states. Is the problem solvable? If it is, provide the plan that was found.

(d) Formalize the following instance of the bridges domain in PDDL and solve it with the same Fast Downward configuration that was used in part (b) of this exercise. Use the original domain `bridges-once-domain.pddl`, where each bridge may be traversed only once.

Provide the runtime and the number of expanded states. Discuss whether the solution is optimal. Is the problem solvable? If it is, provide the plan that was found.



Exercise 10.2 (2 marks)

Prove that h^+ is admissible. For this, prove that if Π is a STRIPS planning task and π is a plan for Π, then π^+ is a plan for the relaxed planning task Π^+. Why does it follow from this that h^+ is admissible?

Exercise 10.3 (1+1+1+1 marks)

Consider the STRIPS planning task $\Pi = \langle V, I, G, A \rangle$ with $V = \{a, b, c, d, e, f\}$, $I = \{a\}$, $G = \{d, f\}$, and $A = \{a_1, a_2, a_3, a_4, a_5\}$ with $cost = \{a_1 \mapsto 2, a_2 \mapsto 4, a_3 \mapsto 1, a_4 \mapsto 1, a_5 \mapsto 1\}$ and

$$
\begin{array}{lll}
pre(a_1) = \{a\} & add(a_1) = \{b, c\} & del(a_1) = \{\} \\
pre(a_2) = \{a\} & add(a_2) = \{c, d\} & del(a_2) = \{\} \\
pre(a_3) = \{b, c\} & add(a_3) = \{d\} & del(a_3) = \{c\} \\
pre(a_4) = \{c\} & add(a_4) = \{e\} & del(a_4) = \{c\} \\
pre(a_5) = \{e\} & add(a_5) = \{f\} & del(a_5) = \{a, b\}.
\end{array}
$$

(a) Provide the relaxed planning graph for Π up to depth 4 (i.e., the resulting graph should have five variable layers, four action layers and a goal vertex).

(b) Compute $h^{\max}(I)$. Provide the values for all nodes in the RPG. Why does the heuristic underestimate $h^+(I)$?

(c) Compute $h^{\text{add}}(I)$. Provide the values for all nodes in the RPG. Why does the heuristic overestimate $h^+(I)$?

(d) Compute $h^{\text{FF}}(I)$. Provide the marked RPG. Why does the heuristic overestimate $h^+(I)$?

Submission rules:

- Create a single PDF file (ending .pdf) for everything except exercise 10.1 (c)/(d). If you want to submit handwritten parts, include their scans in the single PDF. Put the names of all group members on top of the first page. Use page numbers or put your names on each page. Make sure your PDF has size A4 (fits the page size if printed on A4).
- Upload the single PDF or prepare a ZIP file (ending .zip, .tar.gz or .tgz; not .rar or anything else) containing the single PDF and the .pddl files for exercise 10.1 (c)/(d) and nothing else. Please do not use subdirectories in the ZIP.
- Only upload one submission per group. Do not upload several versions, i.e., if you need to resubmit, use the same file name again so that the previous submission is overwritten.