Planning and Optimization

M. Helmert, G. Röger
C. Büchner, R. Christen, S. Dold

University of Basel
Fall Semester 2022

Exercise Sheet 2

Due: October 09, 2023

Important: For submission, consult the rules at the end of the exercise. Non-adherence to these rules might lead to a penalty in the form of a deduction of marks or, in the worst case, that your submission will not be corrected at all.

Exercise 2.1 (1+0.5+0.5+1 marks)

Recall the running example from lectures B2 and B3, a planning task $\Pi = \langle V, I, O, \gamma \rangle$ with

- $V = \{i, w, t_1, t_2\}$;
- $I = \{i \mapsto \mathbf{F}, w \mapsto \mathbf{T}, t_1 \mapsto \mathbf{F}, t_2 \mapsto \mathbf{F}\}$;
- $O = \{m_1, m_2, l_1, l_2, u\}$ where
  - $m_1 = \langle \top, (t_1 \rhd \neg t_1) \wedge (\neg t_1 \rhd t_1), 5 \rangle$,
  - $m_2 = \langle \top, (t_2 \rhd \neg t_2) \wedge (\neg t_2 \rhd t_2), 5 \rangle$,
  - $l_1 = \langle \neg i \wedge (w \leftrightarrow t_1), i \wedge w, 1 \rangle$,
  - $l_2 = \langle \neg i \wedge (w \leftrightarrow t_2), i \wedge \neg w, 1 \rangle$, and
  - $u = \langle i, \neg i \wedge (w \rhd ((t_1 \rhd w) \wedge (\neg t_1 \rhd \neg w))) \wedge (\neg w \rhd ((t_2 \rhd w) \wedge (\neg t_2 \rhd \neg w))), 1 \rangle$; and
- $\gamma = \neg i \wedge \neg w$.

(a) Compute *effcond*$(w, \mathit{eff}(u))$ and simplify the resulting propositional formula.

(b) Describe in 1–2 sentences what *effcond*$(w, \mathit{eff}(u))$ expresses in terms of the interpretation of Π as a logistics problem with t_1 and t_2 denoting the locations of two trucks (L if true, R if false) and i and w denoting the location of a package.

Example: pre$(l_1) = \neg i \wedge (w \leftrightarrow t_1)$ denotes that loading the package into truck t_1 is only possible if the package is not inside a truck already and it is at the same location as t_1.

(c) Consider the state $s = \{i \mapsto \mathbf{T}, w \mapsto \mathbf{F}, t_1 \mapsto \mathbf{T}, t_2 \mapsto \mathbf{F}\}$. Is u applicable in s? If yes, provide the successor state $s[\![u]\!]$.

(d) Define an operator $\hat{u}$ that is flat and equivalent to u.

Exercise 2.2 (2+2 marks)
Consider a simplification of Π from the exercise above: Let $\Pi' = \langle V', I', O', \gamma' \rangle$ be a planning task with

- $V' = \{i, p, t\}$;
- $I' = \{i \mapsto \mathbf{F}, p \mapsto \mathbf{T}, t \mapsto \mathbf{F}\}$;
- $O' = \{m', l', u'\}$ where
  - $m' = \langle \top, (t \rhd \neg t) \wedge (\neg t \rhd t), 5 \rangle$,
  - $l' = \langle \neg i \wedge (p \leftrightarrow t), i, 1 \rangle$, and
  - $u' = \langle i, \neg i \wedge (t \rhd p) \wedge (\neg t \rhd \neg p), 1 \rangle$; and
- $\gamma' = \neg i \wedge \neg p$.

Note: The variable p denotes the position of the package when it is not inside the truck. Otherwise, it denotes the location where it was loaded, which is irrelevant in this problem.

(a) Provide a graphical representation of $\mathcal{T}(\Pi')$. Label states to clearly express the value of all variables in each state, make sure to indicate initial and goal states, and label the transitions.

(b) Transform Π' into an equivalent planning task Π'' in positive normal form.

Exercise 2.3 (2+1 marks)

(a) Model a binary counter with four bits as a propositional planning task $\Pi_1 = \langle V_1, I_1, O_1, \gamma_1 \rangle$. The initial value of the counter is 0 and the goal is to obtain a value of 15. The only allowed operation is to increment the current value by 1, so your model should have a single operator.

(b) Replace O_1 in your previously defined model with a set of operators O_2 without conditional effects. Your model should still resemble a binary counter.

Hint: You need more than one operator to implement the single logical operation of incrementing the counter by 1.

Submission rules:

- Exercise sheets must be submitted in groups of 2–3 students. Create a team on ADAM including all members of your group and submit a single copy of the exercises per group.
- Create a single PDF file (ending .pdf) for all non-programming exercises. Use a file name that does not contain any spaces or special characters other than the underscore "_". If you want to submit handwritten solutions, include their scans in the single PDF. Make sure it is in a reasonable resolution so that it is readable, but ensure at the same time that the PDF size is not astronomically large. Put the names of all group members on top of the first page. Either use page numbers on all pages or put your names on each page. Make sure your PDF has size A4 (fits the page size if printed on A4).
- For programming exercises, only create those code text file(s) required by the exercise. Put your names in a comment on top of each file. Make sure your code compiles and test it. Code that does not compile or which we cannot successfully execute will not be graded.
- For the submission: if the exercise sheet does not include programming exercises, simply upload the single PDF. If the exercise sheet includes programming exercises, upload a ZIP file (ending .zip, .tar.gz or .tgz; *not* .rar or anything else) containing the single PDF and the code text file(s) and nothing else. Do not use directories within the ZIP, i.e., zip the files directly. After creating your zip file and before submitting it, open the file and verify that it complies with these requirements.
- Do not upload several versions to ADAM, i.e., if you need to resubmit, use the same file name again so that the previous submission is overwritten.