Planning and Optimization

M. Helmert, G. Röger
R. Christen, P. Ferber, T. Keller

University of Basel
Fall Semester 2022

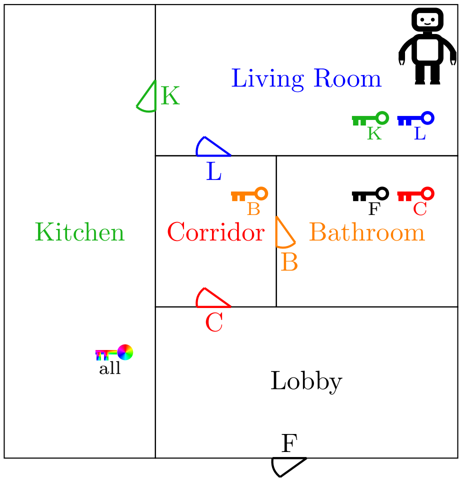
Exercise Sheet 7

Due: November 14, 2022

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 7.1 (1.5+1.5 marks)

It has happened again! Despite the bug fix, your household robot's has gone crazy once more. It locked all doors inside and to your appartment and dropped all keys in arbitrary positions. You are locked out of your appartment, and only the robot can let you back in by opening the front door. The current situation in your home is exactly the same as last time:



(a) As the provided bugfix didn't work for long you decide to look into the reason for the chaotic behaviour yourself. You realize that the robot's naive planning mechanism creates the entire transition system with a procedure that starts by generating all states (no matter if the state is reachable or not), which requires more memory than available. You will have to wait for another bugfix that resolves this issues. In the meantime, you decide to upload an equivalent planning task in *finite-domain representation* that makes your robot open the door.

Provide the FDR planning task. You may use parts of the model solution to exercise 2.1., but your model must use at least one (reasonable!) variable with a domain that is larger than 2. You may use the following sets in your solution:

- the set of doors: $\mathit{Doors} = \{B, C, F, K, L\}$,
- the set of keys: $\mathit{Keys} = \{K_{all}, K_B, K_C, K_F, K_K, K_L\}$,
- the set of rooms: $\mathit{Rooms} = \{R_B, R_C, R_{Li}, R_{Lo}, R_K\}$,
- the set of connections: $\mathit{Cons} = \{(R_{Li}, K, R_K), (R_{Li}, L, R_C), (R_C, B, R_B), (R_C, C, R_L)\}$
- the initial locations of the keys:
  $\mathit{Locs} = \{(K_{all}, R_K), (K_B, R_C), (K_C, R_B), (K_F, R_B), (K_K, R_{Li}), (K_L, R_{Li})\}$.

(b) Compare your model to the model of the task that is given in the model solution to exercise 2.1. Compare the state spaces of the two formulations. How many states do they have? How many of those states are reachable?

Exercise 7.2 (0.5+1+0.5 marks)

Consider propositional planning tasks Π_1, Π_2 and Π_3 with state variables $\{a, b, c, d, e, f, g\}$.

(a) Let $\{a, b, c\}$ and $\{d, e\}$ be mutex groups in Π_1. Provide an invariant that covers exactly this information.

(b) Let v_1 and v_2 be mutex in Π_2 for all

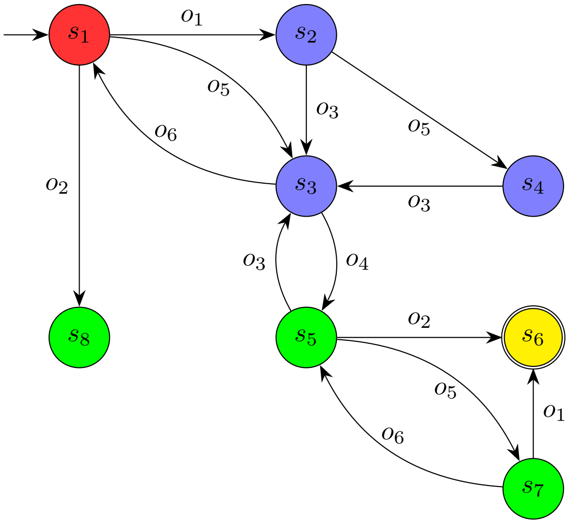
$$\{v_1, v_2\} \in \{\{a, d\}, \{a, f\}, \{a, g\}, \{b, c\}, \{b, d\}, \{b, e\}, \{c, d\}, \{c, e\}, \{c, g\}, \{d, e\}, \{d, f\}\}.$$

Provide all mutex groups G for which there is no mutex group G' such that $G \subset G'$.

(c) Let $\{\{a, b\}, \{a, c, g\}, \{a, d, e\}, \{b, c, e, f\}\}$ be a set of mutex groups for Π_3. Provide two different mutex covers M_1 and M_2 for Π_3 with $|M_i| \leq 3$ for $i = 1, 2$.

Exercise 7.3 (0.5+0.5+0.5+1+1.5+1 marks)

Consider the transition system $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ with S, L, T, s_0 and $S_\star$ as depicted below and with $c(o_i) = i$ for all $1 \leq i \leq 6$ (colors only matter in parts (c) and (d)).



(a) Graphically provide a transition system $\mathcal{T}_1$ such that $\mathcal{T}_1 \sim \mathcal{T}$ and $\mathcal{T}_1 \neq \mathcal{T}$ and provide the functions φ and λ that are used in the definition of isomorphic transition systems.

(b) Graphically provide a transition system $\mathcal{T}_2$ such that $\mathcal{T}_2 \overset{\text{G}}{\sim} \mathcal{T}$ and $\mathcal{T}_2 \not\sim \mathcal{T}$. Provide the function φ that is used in the definition of graph-equivalent transition systems and argue why $\mathcal{T}_2 \not\sim \mathcal{T}$.

(c) Consider the abstraction α that maps all states depicted in the same color to the same abstract state, i.e., $\alpha(s_1) = s_r$, $\alpha(s_2) = \alpha(s_3) = \alpha(s_4) = s_b$, $\alpha(s_5) = \alpha(s_7) = \alpha(s_8) = s_g$ and $\alpha(s_6) = s_y$. Graphically provide $\mathcal{T}^\alpha$ and give h^α.

(d) Assume you may change the abstraction α from part (c) by mapping one concrete state to another (already existing) abstract state. If you care about having some positive effect on the heuristic quality, which change do you make? Justify your answer. (There are multiple reasonable options.)

(e) Provide an abstraction β of $\mathcal{T}$ such that $|S^\beta| = 4$ and such that there is no abstraction $\beta' \neq \beta$ with $|S^{\beta'}| = 4$ and $h^{\beta'}(s_1) > h^\beta(s_1)$. Graphically provide the transition system $\mathcal{T}^\beta$.

(f) Consider the abstraction α from part (c). Provide an abstraction α' of $\mathcal{T}$ and a function α'' such that

- $\alpha' \neq \alpha$,
- α' is a coarsening of α, and
- $\alpha' = \alpha'' \circ \alpha$ with $h^{\alpha'}(s_1) = h^{\alpha}(s_1)$.

Submission rules:

- Exercise sheets must be submitted in groups of two or three students. Please submit a single copy of the exercises per group (only one member of the group does the submission).
- 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 textfiles 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 textfile(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.