Foundations of Artificial Intelligence

M. Helmert
S. Eriksson
Spring Term 2022

University of Basel
Computer Science

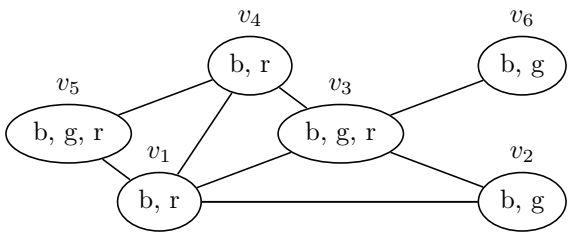
Exercise Sheet 7

Due: April 17, 2022

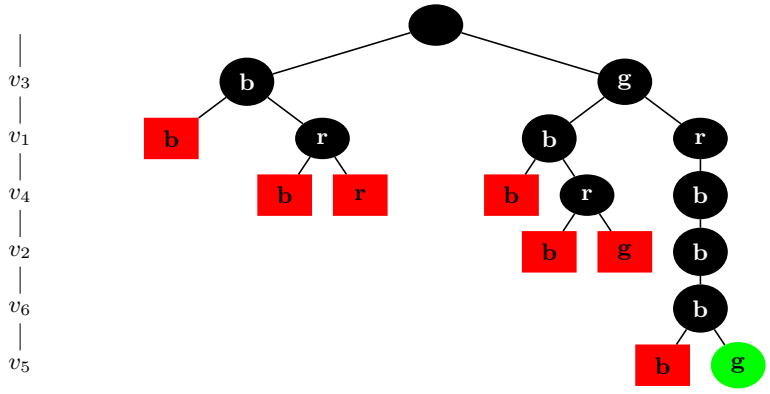
Important: for submission, consult the rules at the end of the exercise. Non-adherence to the rules will lead to your submission not being corrected.

Exercise 7.1 (1.5+0.5 marks)

Consider the following constraint network for a graph coloring problem:



Using naive backtracking with the static variable ordering $\langle v_3, v_1, v_4, v_2, v_6, v_5 \rangle$ and alphabetical value ordering yields the following search tree:



(a) Provide the search tree that is created by applying naive backtracking, using the following *static* variable and value orderings. Depict your search tree in a similar style.

- Variable ordering:
  - select a variable according to the *minimum remaining values* variable ordering criterion;
  - if there is more than one such variable, break ties according to the *most constraining variable* variable ordering criterion;
  - if the choice is still not unique, break ties by selecting the variable with the smallest index.
- Value ordering: alphabetical

Note: We use static orderings here, meaning they are precomputed before and are not adjusted during the search.

(b) Compare the size of your search tree to the one above. Which variable ordering was better?

Exercise 7.2 (1+2 marks)
Consider the 6 queens problem with the partial assignment $\alpha = \{v_1 \mapsto 2, v_2 \mapsto 4\}$:

	v_1	v_2	v_3	v_4	v_5	v_6
1						
2	q					
3						
4		q				
5						
6						

In the following, you may assume that the positions of the two queens that are already on the board are fixed, i.e., that the domain of the corresponding variables contains only the single entry that encodes the depicted position. The domain of the remaining variables contains all 6 possible values, though, which leads to the following domains for all variables:

$$\begin{aligned} \mathrm{dom}(v_1) &= \{2\} & \mathrm{dom}(v_4) &= \{1,2,3,4,5,6\} \\ \mathrm{dom}(v_2) &= \{4\} & \mathrm{dom}(v_5) &= \{1,2,3,4,5,6\} \\ \mathrm{dom}(v_3) &= \{1,2,3,4,5,6\} & \mathrm{dom}(v_6) &= \{1,2,3,4,5,6\} \end{aligned}$$

(a) Determine the domains of all variables after applying forward checking in α.

(b) Apply the AC-3 algorithm that has been presented on slide 21 of chapter 25 in the print version of the lecture slides on the constraint network $\mathcal{C}$ with the domains that are the result of (a) until arc consistency is enforced. Select the variables u and v in each iteration of the while loop such that the domain of u changes in the call to `revise`$(\mathcal{C}, u, v)$. Provide u, v, and $\mathrm{dom}(u)$ in each iteration. Note that you do *not* have to provide the elements that are inserted into the queue, and you may stop the algorithm as soon as there are no variables u and v such that $\mathrm{dom}(u)$ changes.

Submission rules:

- Exercise sheets must be submitted in groups of two 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.

- 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.