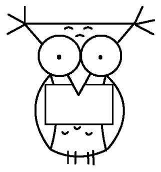
Exercise 1 :

The owl Steffi wants to deliver a letter containing the result of the student council elections from the CS student council to the AStA building. After delivering the results, she wants to go back to her nest in the student council. Steffi is currently located at the Mensa building. She can fly between any two locations.



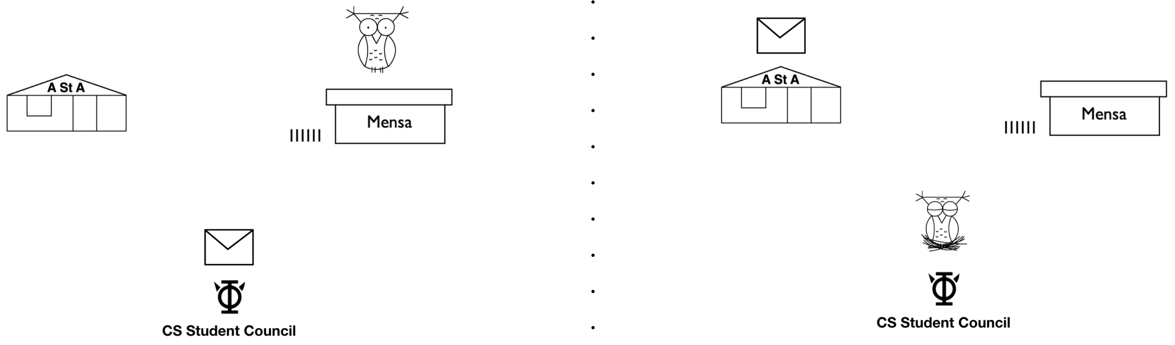


Figure 1: Initial state (left) and goal (right) of the planning task

(i) Complete the formalization of the planning task. “Empty” stands for Steffi not carrying anything. Fill in all gaps indicated by a question mark (?). You may introduce new predicates if necessary.

$$P = \{at(\mathit{Steffi}, x), at(\mathit{Letter}, x), \mathit{carrying}(\mathit{Letter}), \mathit{empty}\}$$
$$\text{for } x \in \{Mensa, AStA, Council\}$$
$$I = \{at(\mathit{Steffi}, Mensa), at(\mathit{Letter}, Council), \mathit{empty}\}$$
$$G = \{?\}$$
$$A = \{fly(x,y) | (x,y) \in \{?\}\}$$
$$\cup \{\mathit{pickup}(i,x) \mid x \in \{?\}, i \in \{?\}\}$$
$$\cup \{\mathit{drop}(i,x) \mid x \in \{?\}, i \in \{?\}\}$$

- $fly(x,y)$: pre: $\{?\}$, add: $\{?\}$, del: $\{?\}$
- $\mathit{pickup}(i,x)$: pre: $\{?\}$, add: $\{?\}$, del: $\{?\}$
- $\mathit{drop}(i,x)$: pre: $\{?\}$, add: $\{?\}$, del: $\{?\}$

(ii) Give an optimal plan for the task. What is the value of $h^*(I)$?

Solution:

(i) $I = \{at(Steffi, Mensa), at(Letter, Council), empty\}$

$G = \{at(Letter, AStA), at(Steffi, Council)\}$

$$\begin{aligned} A &= \{fly(x,y) \mid x, y \in \{Mensa, AStA, Council\}, x \neq y\} \\ &\cup \{\mathit{pickup}(i,x) \mid x \in \{Mensa, AStA, Council\}, i \in \{Letter\}\} \\ &\cup \{\mathit{drop}(i,x) \mid x \in \{Mensa, AStA, Council\}, i \in \{Letter\}\} \end{aligned}$$

- $fly(x, y)$:

$$\begin{aligned} pre &: \{at(Steffi, x)\} \\ add &: \{at(Steffi, y)\} \\ del &: \{at(Steffi, x)\} \end{aligned}$$

  for all $x, y \in \{Mensa, AStA, Council\}$, $x \neq y$

- $pickup(i, x)$:

$$\begin{aligned} pre &: \{at(i, x), at(Steffi, x), empty\} \\ add &: \{carrying(i)\} \\ del &: \{at(i, x), empty\} \end{aligned}$$

  for all $x \in \{Mensa, AStA, Council\}$, $i \in \{Letter\}$

- $drop(i, x)$:

$$\begin{aligned} pre &: \{carrying(i), at(Steffi, x)\} \\ add &: \{at(i, x), empty\} \\ del &: \{carrying(i)\} \end{aligned}$$

  for all $x \in \{Mensa, AStA, Council\}$, $i \in \{Letter\}$

(ii) $\pi = \langle fly(Mensa, Council), pickup(Letter, Council), fly(Council, AStA),$
$drop(Letter, AStA), fly(AStA, Council)\rangle$
$h^*(I) = 5$

Exercise 2 :

Consider the following planning task, in which we have only one letter, Steffi is already holding the letter and Steffi can only move between the AStA and the Student Council:

$P = \{atS(C), atS(A), atL(C), atL(A), atL(S)\}$
$I = \{atS(C), atL(S)\}$
$G = \{atS(C), atL(A)\}$
$A = \{fly(x, y), pickup(x), drop(x)\}$, where $x, y \in \{A, C\}$; and $x \neq y$ for $fly(x, y)$

$$\begin{array}{lcll} fly(x,y) & : & pre: & \{atS(x)\} \\ & & add: & \{atS(y)\} \\ & & del: & \{atS(x)\} \\ \\ pickup(x) & : & pre: & \{atL(x), atS(x)\} \\ & & add: & \{atL(S)\} \\ & & del: & \{atL(x)\} \\ \\ drop(x) & : & pre: & \{atL(S), atS(x)\} \\ & & add: & \{atL(x)\} \\ & & del: & \{atL(S)\} \end{array}$$

(i) How many states are there in this planning task?

(ii) Draw the reachable part of the state space. Denote every state by the facts that are currently true, i.e., the current positions of Steffi and the Letter (for example, the initial state can be denoted by CS).

(iii) How many reachable states are in this planning task?

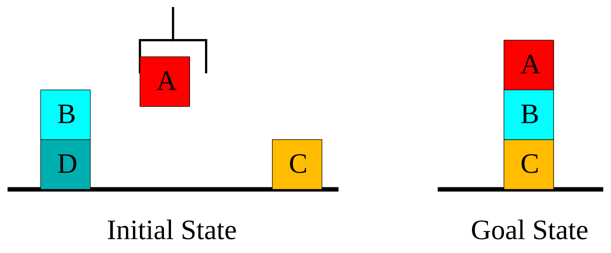
Solution:

There are 2^5 states, out of which 6 are reachable.

$$\begin{tikzcd} {} \arrow[r] & CS \arrow[r, bend right, "fly(C,A)"'] \arrow[d, bend right, "drop(C)"'] & AS \arrow[l, bend right, "fly(A,C)"'] \arrow[r, bend right, "drop(A)"'] & AA \arrow[l, bend right, "pickup(A)"'] \arrow[d, bend right, "fly(A,C)"'] \\ & CC \arrow[u, bend right, "pickup(C)"'] \arrow[r, bend right, "fly(C,A)"'] & AC \arrow[l, bend right, "fly(A,C)"'] & CA \arrow[u, bend right, "fly(C,A)"'] \end{tikzcd}$$

Exercise 3 :

Consider the following Blocksworld task:



The problem is formalized as the following STRIPS planning problem $\Pi = (P, A, I, G)$:

$$\begin{aligned} P =& \{stacked(x,y) |\ x,y \in \{A,B,C,D\}\} \\ & \cup \{ontable(x) |\ x \in \{A,B,C,D\}\} \\ & \cup \{clear(x) |\ x \in \{A,B,C,D\}\} \\ & \cup \{holding(x) |\ x \in \{A,B,C,D\}\} \\ & \cup \{empty\} \end{aligned}$$

$$\begin{aligned} A =& \{stack(x,y), unstack(x,y) \mid x,y \in \{A,B,C,D\}, x \neq y\} \\ & \cup \{putdown(x), pickup(x) \mid x \in \{A,B,C,D\}\} \end{aligned}$$

where

- $stack(x,y)$ for $x,y \in \{A,B,C,D\}$, $x \neq y$
  - $pre : \{holding(x), clear(y)\}$
  - $add : \{stacked(x,y), clear(x), empty\}$
  - $del : \{holding(x), clear(y)\}$
- $unstack(x,y)$ for $x,y \in \{A,B,C,D\}$, $x \neq y$
  - $pre : \{stacked(x,y), clear(x), empty\}$
  - $add : \{holding(x), clear(y)\}$
  - $del : \{stacked(x,y), clear(x), empty\}$
- $putdown(x)$ for $x \in \{A,B,C,D\}$
  - $pre : \{holding(x)\}$
  - $add : \{ontable(x), clear(x), empty\}$
  - $del : \{holding(x)\}$
- $pickup(x)$ for $x \in \{A,B,C,D\}$
  - $pre : \{ontable(x), clear(x), empty\}$
  - $add : \{holding(x)\}$
  - $del : \{ontable(x), clear(x), empty\}$

a) Provide I and G, according to the figure (we do not consider $clear(A)$ part of the goal).
b) Provide an optimal plan, what is the value of $h^*(I)$?
c) Provide an optimal delete-relaxed plan, what is the value of $h^+(I)$?

Solution:

a) $I = \{ontable(D), stacked(B, D), clear(B), ontable(C), clear(C), holding(A)\}$,
$G = \{stacked(A, B), stacked(B, C), ontable(C)\}$ ($ontable(C)$ would be optional)

b) Optimal plan: $\langle putdown(A)$, $unstack(B, D)$, $stack(B, C)$, $pickup(A)$, $stack(A, B)\rangle$. $h^*(I)$ = 5

c) Optimal relaxed plan: $\langle stack(A, B)$, $unstack(B, D)$, $stack(B, C)\rangle$. $h^+(I) = 3$

Exercise 4 :
For all of the following statements, decide if they are correct. Briefly justify your answer (1–3 sentences).

1. In the delete relaxation, an optimal plan (if it exists) has length at most $|G|$.
2. Our implementation of h^{FF} is an admissible heuristic.
3. Let $\mathcal{R}$ be a relaxation mapping that drops all preconditions in a given planning task. Then the heuristic defined as $h^{\mathcal{R}} := h^* \circ \mathcal{R}$ is an admissible heuristic.
4. For every state it holds that if it is unsolvable in the original problem, it is unsolvable in the delete relaxation.
5. For every state it holds that if it is solvable in the original problem, it is solvable in the delete relaxation.
6. Let $\Pi = (P, A, I, G)$ be a STRIPS planning task with empty delete lists, i.e. $\text{del}_a = \emptyset$ for all action $a \in A$. The task $\Pi' = (P, A, I, G \setminus I)$ in which all facts that are initially true have been removed from the set of goal facts is solvable if and only if Π is solvable.

Solution:

1. False. This was true for the only-adds relaxation, but only becuase we did not have any preconditions. Consider the following counterexample in which the optimal plan in the delete relaxation has length $2 > |G| = 1$.

$$\begin{aligned} P &= \{p, q\} \\ A &= \{a_1, a_2\} \\ I &= \{\} \\ G &= \{q\} \end{aligned}$$

$$\begin{aligned} \text{pre}_{a_1} &= \{\}, \quad \text{add}_{a_1} = \{p\}, \quad \text{del}_{a_1} = \{\} \\ \text{pre}_{a_2} &= \{p\}, \quad \text{add}_{a_2} = \{q\}, \quad \text{del}_{a_2} = \{\} \end{aligned}$$

2. False. Our implementation gives us **some** plan for the delete-relaxation, not the shortest. The claim would only hold if h^{FF} always computed the shortest plan in the delete-relaxation, since $h^+(s) \leq h^*(s)$.

3. True. Any plan in the original problem would still be a plan in this relaxation, as all action are applicable because of absent preconditions.

4. False. Consider this task in which our agent shall be at two locations (•) **at the same time**. He can move left or right if he stays inside the 3 squares.

$Agent$	•	•

Obviously this is impossible. However, in the delete relaxation the agent can move to the very right. After doing this, the agent is everywhere in the delete-relaxation, as our previous positions have never been erased. Therefore, $h^+(I) = 2$ for this example.

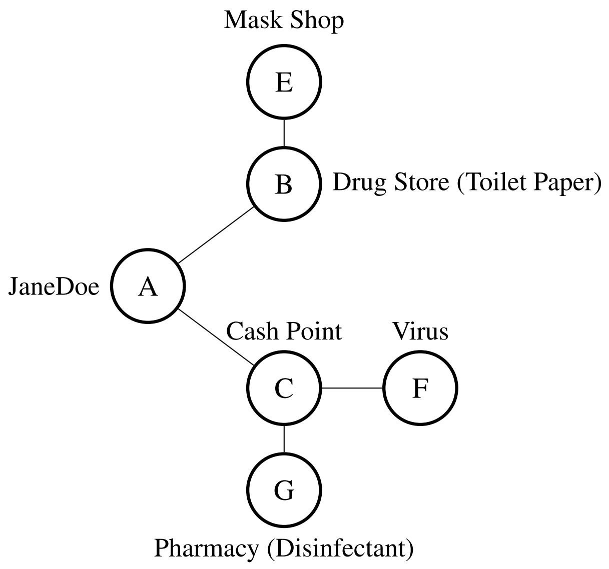
5. True. Any plan in the original problem is a plan in the delete relaxation.

6. True. Since we have no deletes, the goal facts that were initially true stay true forever. Therefore, any plan in this relaxation is also a plan in the original problem.

Exercise 5 :

Jane Doe's country was hit by a virus. She wants to fight the virus. In order to do so, she either needs disinfectant, or a mask and toilet paper. Currently, Jane Doe is at home at location A, ToiletPaper can be bought at the Drug Store at location B, a mask can be bought at location E, and disinfectant can be bought at the pharmacy at location G. Note that she first needs to withdraw money at the cashpoint at location C.

The problem was formalized as (P not given explicitly):



$$
\begin{aligned}
I &= \{at(A)\} \\
G &= \{VirusDead\} \\
A &= \{buyDisinfectant, buyMask, buyToiletPaper, withdrawMoney, \\
&\quad useToiletPaper, useDisinfectant\} \\
&\quad \cup \{walk(x,y) \mid x,y \in \{A,\ldots,G\} \text{ are connected on the map}\}
\end{aligned}
$$

$walk(x, y)$:
pre : $\{at(x)\}$
add : $\{at(y)\}$
del : $\{at(x)\}$

$buyDisinfectant$:
pre : $\{at(G), has2Money\}$
add : $\{hasDisinfectant\}$
del : $\{has2Money\}$

$buyMask2$:
pre : $\{at(E), has2Money\}$
add : $\{hasMask, has1Money\}$
del : $\{has2Money\}$

$buyMask1$:
pre : $\{at(E), has1Money\}$
add : $\{hasMask\}$
del : $\{has1Money\}$

$buyToiletPaper2$:
pre : $\{at(B), has2Money\}$
add : $\{hasToiletPaper, has1Money\}$
del : $\{has2Money\}$

$buyToiletPaper1$:
pre : $\{at(B), has1Money\}$
add : $\{hasToiletPaper\}$
del : $\{has1Money\}$

$withdrawMoney$:
pre : $\{at(C)\}$
add : $\{has2Money\}$
del : $\{has1Money\}$

$useDisinfectant$:
pre : $\{at(F), hasDisinfectant\}$
add : $\{VirusDead\}$
del : $\{hasDisinfectant\}$

$useToiletPaper$:
pre : $\{at(F), hasToiletPaper, hasMask\}$
add : $\{VirusDead\}$
del : $\{hasToiletPaper\}$

a) Give an optimal plan for Π. What is the value of $h^*(I)$?

b) Compute h^{FF} for the initial state. Write down the action and fact sets (A_i and F_i) for each iteration of the RPG algorithm. In the plan extraction, for each step t, write down the fact set G_t, which actions are selected for each fact in G_t, and the resulting fact set updates for each selected action. The order in which you select the actions that generate the facts is not relevant. **Try to avoid using the disinfectant during plan extraction.** What is the value of $h^{FF}(I)$? What is the value of $h^{max}(I)$?

c) Run A* search on the problem using h^+ as the heuristic. If several nodes have the same $g + h$ value, rely on the heuristic and **choose the one with the smaller h value**. If there are several nodes with same g and same h value, break ties by using the alphabetical order on location names. In each search node, mention the facts that are true, the g and h values as well as the expansion order. In the search graph, mark duplicate nodes (e.g. by crossing them out).

Note: Use the following abbreviations.
h2M := has2Money, hD := hasDisinfectant, VD := VirusDead

Solution:

a) i) $walk(A, C) \rightarrow withdrawMoney \rightarrow walk(C, G) \rightarrow buyDisinfectant \rightarrow walk(G, C) \rightarrow walk(C, F) \rightarrow useDisinfectant$

ii) $h^*(I) = 7$

b) Forward search to compute the RPG:

- $F_0 = I = \{at(A)\}$
- $A_0 = \{walk(A, B), walk(A, C)\}$
  $F_1 = F_0 \cup \{at(B), at(C)\}$
- $A_1 = A_0 \cup \{walk(B, E), walk(B, A), walk(C, A), walk(C, F), walk(C, G), withdrawMoney\}$
  $F_2 = F_1 \cup \{at(E), at(F), at(G), has2Money\}$
- $A_2 = A_1 \cup \{walk(E, B), walk(F, C), walk(G, C), buyToiletPaper2, buyMask2, buyDisinfectant\}$
  $F_3 = F_2 \cup \{hasToiletPaper, hasMask, hasDisinfectant, has1Money\}$
- $A_3 = A_2 \cup \{useToiletPaper, useDisinfectant, buyToiletPaper1, buyMask1\}$
  $F_4 = F_3 \cup \{VirusDead\} \subseteq G \rightarrow stop$

Backward search to extract a plan:

Initially: $G_0, G_1, G_2, G_3 = \emptyset,\ G_4 = \{VirusDead\}$

Layer	Selected fact	Inserted Action	Marked facts
4	$VirusDead$	$useToiletPaper$	$G_4 = \{VirusDead\}$
3	$hasToiletPaper$	$buyToiletPaper2$	$G_3 = \{hasToiletPaper, hasMask\}$
	$hasMask$	$buyMask2$	
2	$has2Money$	$withdrawMoney$	$G_2 = \{has2Money, at(E), at(F)\}$
	$at(E)$	$walk(B, E)$	
	$at(F)$	$walk(C, F)$	
1	$at(B)$	$walk(A, B)$	$G_1 = \{at(B), at(C)\}$
	$at(C)$	$walk(A, C)$	

$h^{FF}(I) = 8$ since eight actions were selected during plan extraction. $h^{max}(I) = 4$ since the goal was achieved in layer F_4.

c) The search graph is depicted in figure 2.

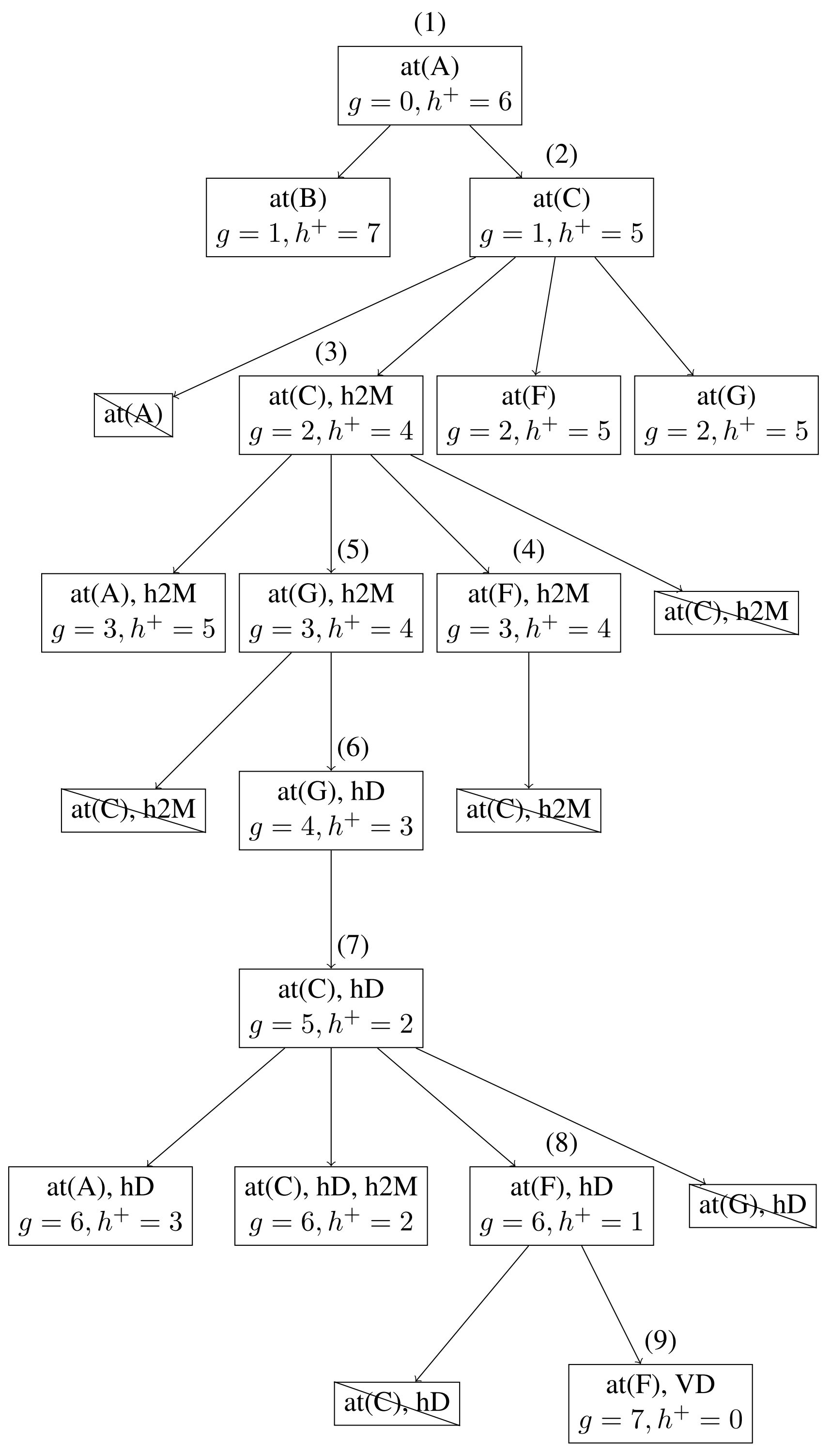


Figure 2: The search graph of part (b).

Exercise 6 (Optional):

Optional but recommended, e.g., as part of the extra exercise sessions

In this exercise, we consider the problem of collecting objects of different colors, while moving along a road map that takes the form of a directed tree of depth 3 (Figure 3). Each tree node might contain up to three objects having any of three different colors (red, green, blue); moving to that node immediately collects all the objects present.

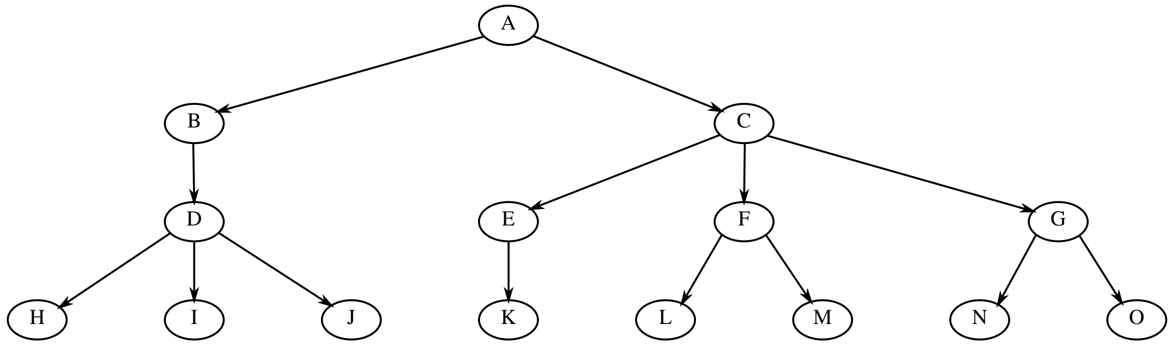


Figure 3: Empty road map

The corresponding STRIPS planning task $\Pi = (P, A, I, G)$ looks as follows:

$$\begin{aligned} P &= \{at(x) \mid x \in \{A, \ldots, O\}\} \cup \{got(c) \mid c \in \{red, green, blue\}\} \\ A &= \{move(x, y) \mid (x, y) \text{ is and edge in the tree}\} \\ I &= \{at(A)\} \\ G &= \{at(J), got(red), got(green), got(blue)\} \end{aligned}$$

The goal is to be at location J, while having the three colors.

Here, the action $move(x, y)$ is specified as follows:

- In case that the destination y does not contain any objects: $move(x, y) =$

  $$\begin{aligned} &pre : \{at(x)\} \\ &add : \{at(y)\} \\ &del : \{at(x)\} \end{aligned}$$

  for all x, y in the tree with an edge from x to y.

- In case that the destination y does contain objects (shown here for red and blue; other cases are defined accordingly): $move(x, y) =$

  $$\begin{aligned} &pre : \{at(x)\} \\ &add : \{at(y), got(red), got(blue)\} \\ &del : \{at(x)\} \end{aligned}$$

  for all x, y in the tree with an edge from x to y, where y contains both a red object and a blue object.

Note in the figure that the edges are directed, i.e., you can pass them only in one direction. Note further that any colors placed in A will not be taken (there is no move action going to A).

Your task is to give three different colorings of the road map. Such a coloring assigns each node of the road map up to three colors (it can have none, any of the three colors, any two of them, or all three). Then, you will use Greedy Best-First Search (GBFS) with the goal counting heuristic and lexicographic tie-breaking (so, if there are multiple nodes with the same h value, the one with a letter coming first in the alphabet will be selected for expansion). The goal is to specify a coloring so that GBFS returns a solution after expanding a specified number of states. For this note that if a goal state is to be expanded next, the search stops without generating any successors, so that we do not count this as an expansion.

For each of the parts (a), (b), and (c) do the following: (i) draw the road map with your coloring and (ii) draw the search space, showing all states generated during the search, including an edge from each expanded state to the child nodes generated when expanding it. Identify each state s by the name of the respective graph node (the node x for which "$at(x)$" is true in s), as well as the subset of colors already collected in the state (those colors x for which "$got(x)$" is true in s). Annotate each state with its heuristic value, as well as a number indicating the expansion order. (For generated states that are not expanded, use "–" as the expansion order number.)

The goal counting heuristic is defined as follows: For state s, $h^{gc}(s)$ is the number of goal facts $p \in G$ that are not true in s. (In other words: $h^{gc}(s) = |G \setminus s|$.)

(a) Color the graph and perform Greedy Best-First Search, so that it expands exactly three states.

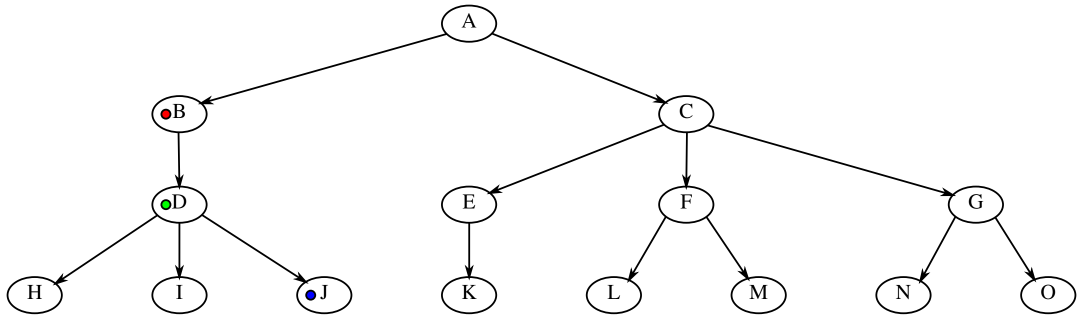
(b) Color the graph and perform Greedy Best-First Search, so that it expands at least six and at most nine states.

(c) Color the graph and perform Greedy Best-First Search, so that it expands exactly twelve states.

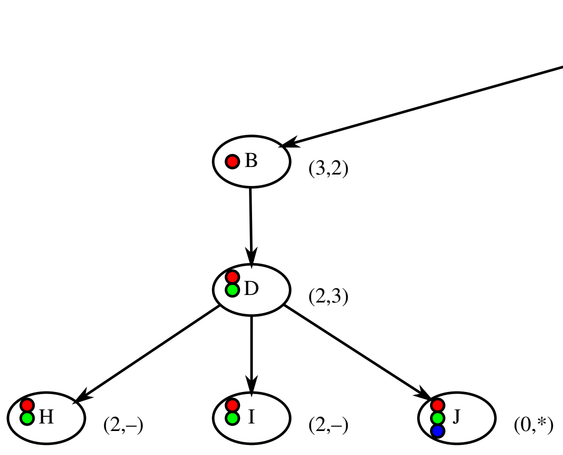
Solution:

All nodes in the following search space graphs are annotated with a pair of numbers (h, e), where h is the heuristic value and e the number of that node in the expansion order.

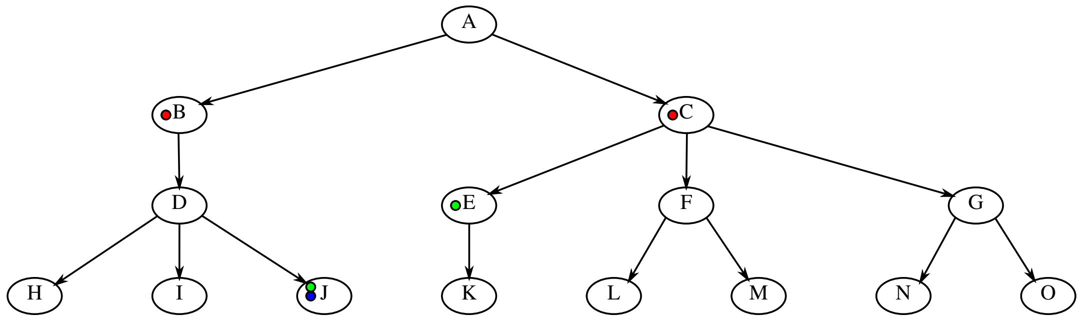
(a) Example coloring giving three expansions:



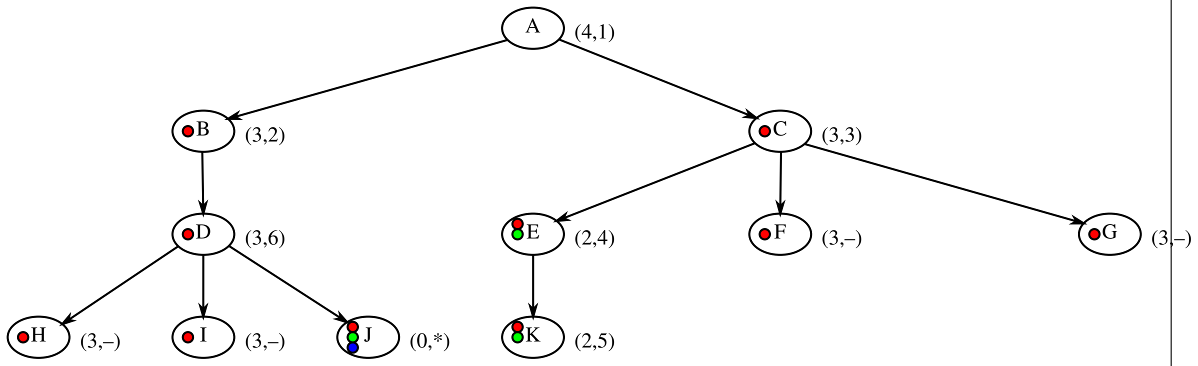
Corresponding search space of greedy best-first search using h^{gc}:



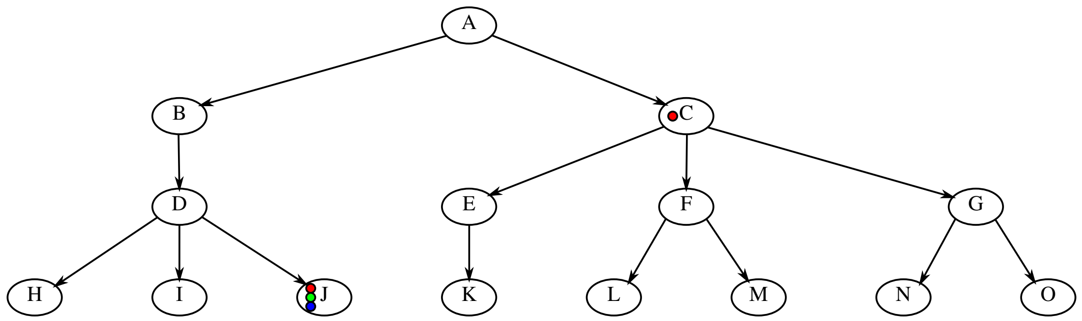
(b) Example coloring giving six expansions:



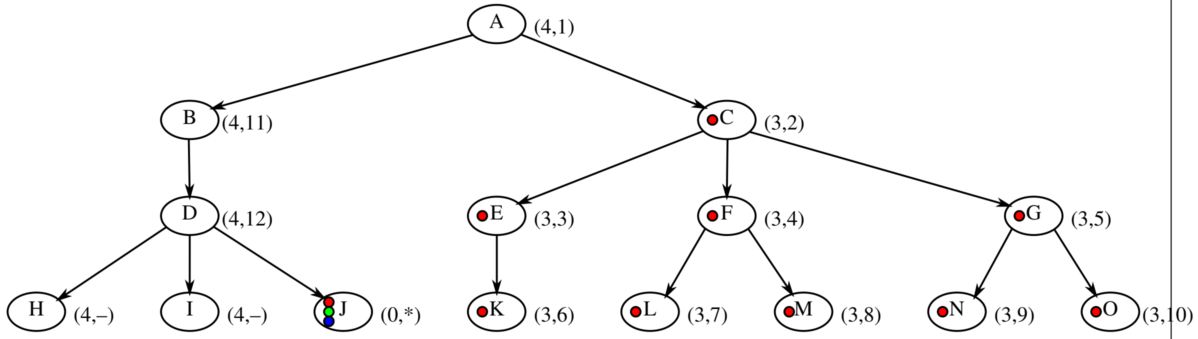
Corresponding search space of greedy best-first search using h^{gc}:



(c) Example coloring giving twelve expansions:



Corresponding search space of greedy best-first search using h^{gc}:


A (4,1)
B (4,11)
C (3,2)
D (4,12)
E (3,3)
F (3,4)
G (3,5)
H (4,–)
I (4,–)
J (0,*)
K (3,6)
L (3,7)
M (3,8)
N (3,9)
O (3,10)