Foundations of Artificial Intelligence

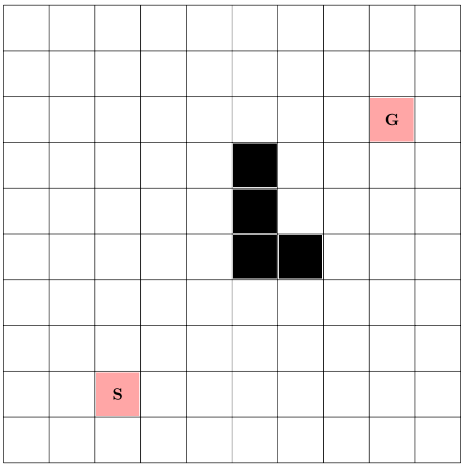
Prof. Dr. J. Boedecker, Prof. Dr. W. Burgard, Prof. Dr. F. Hutter, Prof. Dr. B. Nebel,
Dr. rer. nat. M. Tangermann
T. Schulte, M. Krawez, R. Rajan, S. Adriaensen, K. Sirohi
Summer Term 2020

University of Freiburg
Department of Computer Science

Exercise Sheet 3 — Solutions

Exercise 3.1 (Local search)

A robot tries to move from a start location (S) to a goal location (G). The robot can move between horizontally or vertically connected grid cells, one cell in each step. Black cells represent walls through which the robot cannot move. Each move incurs a uniform cost of 1. The following figure depicts an example.



(a) Explain how hill-climbing would work as a method of reaching a given goal location. Provide a suitable quality measure (or heuristic) for the states.

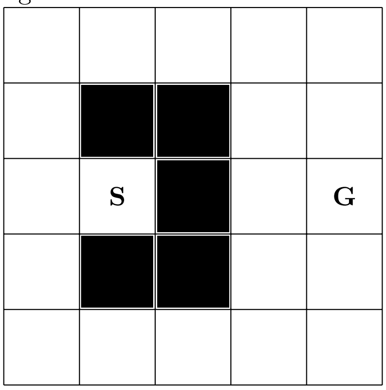
(b) Show how certain combinations of walls can result in a local maximum for the hill-climber. Use an example.

(c) Would simulated annealing always escape local maxima on this family of problems? Explain!

Solution:

(a) Hill-climbing is surprisingly effective at finding reasonable if not optimal paths for very little computational cost, and seldom fails in two dimensions. Let $d(x, y)$ be the Manhattan distance between cells x and y, and let the value of a cell p be $-d(p, goal)$. An agent can move to any adjacent cell (that is not a wall), and when hill-climbing it will choose the accessible cell closest to the goal.

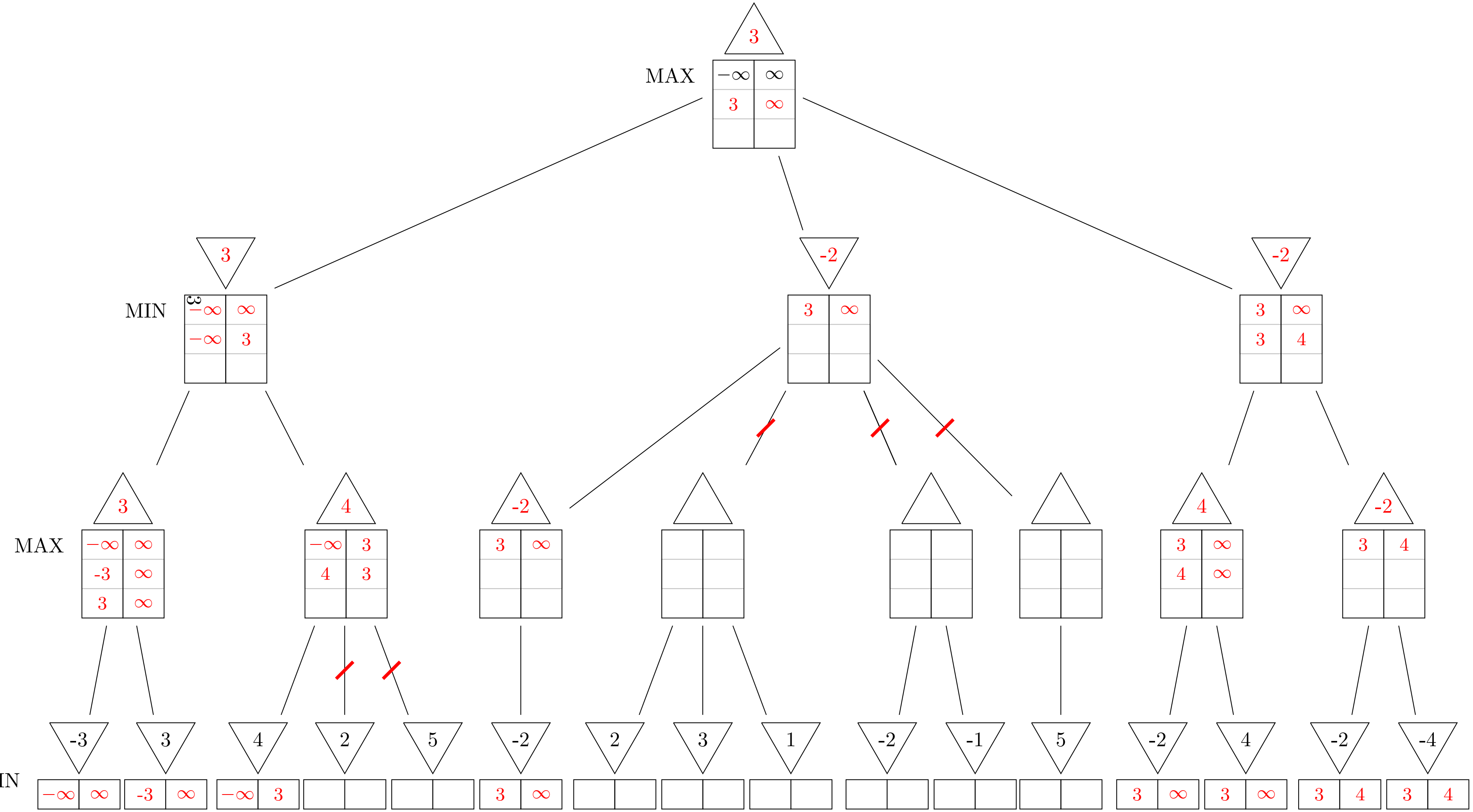
(b) This can occur when an agent is at a non-convex obstacle as shown in the figure below.



(c) If a solution exists (the agent can't be entirely walled-in) and one chooses an appropriate annealing schedule, simulated annealing can escape local maxima for this family of problems.

Exercise 3.2 (Board Games)

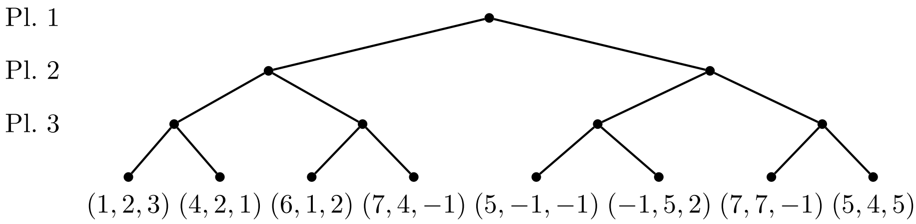
(a) Consider the game tree for the two-person board game depicted below. Simulate the behavior of the Minimax algorithm with α-β pruning (always expand children from left to right). Enter the computed node values into the triangles and all intermediate α-β values into the appropriate tables.



3
MAX
−∞ ∞
3 ∞
3
MIN
3
−∞ ∞
−∞ 3
-2
3 ∞
-2
3 ∞
3 4
MAX
3
−∞ ∞
-3 ∞
3 ∞
4
−∞ 3
4 3
-2
3 ∞
4
3 ∞
4 ∞
-2
3 4
-3
3
4
2
5
-2
2
3
1
-2
-1
5
-2
4
-2
-4
IN
−∞ ∞
-3 ∞
−∞ 3
3 ∞
3 ∞
3 ∞
3 4
3 4

(b) Consider the problem of search in a three-player game (you may assume that no alliances are allowed) without the zero-sum condition. The players are called 1, 2, and 3. Unlike in the case of two-player zero-sum games, the evaluation function now returns a triple (x_1, x_2, x_3) such that x_i is the value the node has for player i.

Complete the game tree given below by annotating all interior nodes and the root node with the backed-up value triples.



Solution:

- Level Pl. 3: $(1,2,3)$, $(6,1,2)$, $(-1,5,2)$, $(5,4,5)$
- Level Pl. 2: $(1,2,3)$, $(-1,5,2)$
- Level Pl. 1: $(1,2,3)$

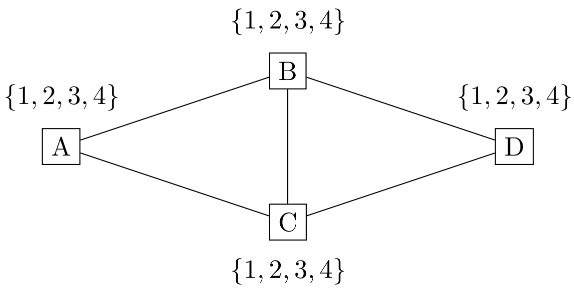
(c) Assume that the value triple $(5,4,5)$ at the rightmost leaf node is replaced by $(5,4,-1)$. Which problem arises now when you try to back up value triples? Suggest how to modify the back-up procedure to obtain a "robust" result at the root node.

Solution:

Problem: Player 3 is indifferent between multiple actions. His choice is not unique anymore. The back-up procedure could either break ties according to some strategy (e.g., in favor of actions that yield a low utility for the other players) or propagate sets of value tuples instead of a single value tuple.

Exercise 3.3 (Constraint Satisfaction Problems)

(a) Consider the constraint satisfaction problem given by the constraint graph below. To each of the four nodes $\{A, B, C, D\}$ a value of the respective domain (see node labels) should be assigned, such that all constraints given below are satisfied. Solve the CSP by backtracking search. Use the variable ordering A, B, C, D and assign values in increasing order: first 1, then 2, then 3, then 4. After each variable assignment apply *forward checking* to update the domains of unassigned variables. Complete the table below!



Constraints:

- $A \neq C$
- $B \neq D$
- $|A - B| > 1$
- $|B - C| > 1$
- $|C - D| > 1$

i	$Assign$	A	B	C	D
0	-	$1, 2, 3, 4$	$1, 2, 3, 4$	$1, 2, 3, 4$	$1, 2, 3, 4$
1	$A = 1$	1	$3, 4$	$2, 3, 4$	$1, 2, 3, 4$
2	$B = 3$	1	3	-	1,2,4
3	$B = 4$	1	4	2	1,2,3
4	$C = 2$	1	4	2	-
5	$A = 2$	2	4	1,3,4	1,2,3,4
6	$B = 4$	2	4	1	1,2,3
7	$C = 1$	2	4	1	3