COMP417 Artificial Intelligence
Exercise Set 2

Pantourakis Michail AM 2015030185
School of Electrical and Computer Engineering
Technical University of Crete

4 April 2018

Exercise 1

Part A

Using the minimax algorithm, we assume that both players play optimally (namely the MAX player maximizes his utility value, while the MIN player minimizes the same value). Furthermore, the algorithm performs a complete depth-first exploration of the corresponding game tree (Figure 1), assigning the minimum or maximum utility value out of all its children nodes as a minimax value to every MIN or MAX node respectively.

In this case, MIN nodes B, C, D and E would be given the values -12, -7, -9, 2 respectively, while the starting MAX node will take the maximum of these four values, namely 2. Therefore, the resulting optimal move for the MAX player would lead to E, and the optimal response by the MIN player would then lead to the terminal node with utility value 2.

Part B

Figure 1 depicts the game tree.

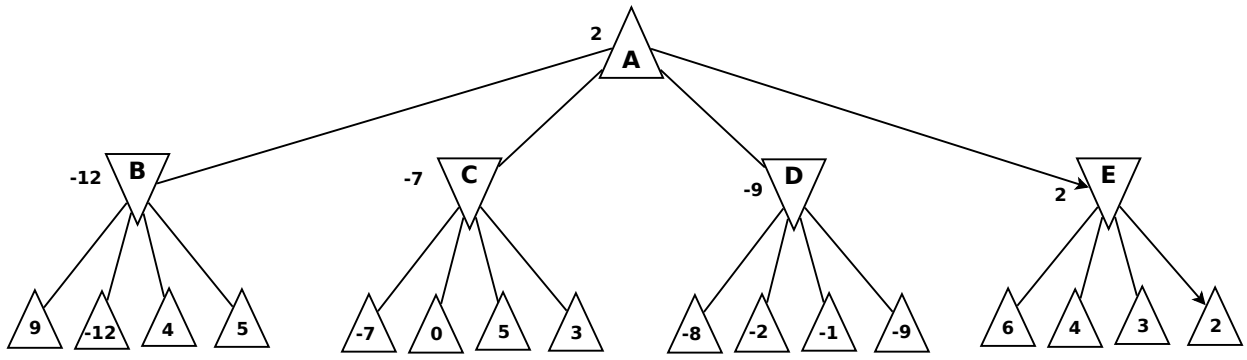

Figure 1: The complete game tree of exercise 1. The $\triangle$ nodes represent MAX ones, whereas the $\triangledown$ nodes represent MIN ones. The terminal nodes (leaves) show the utility values for MAX, whilst the other nodes are labeled with their minimax values.

Exercise 4

As explained in Exercise 1, the minimax algorithm guarantees the optimal solution for the MAX player given that both players make optimal decisions. Assuming that the MIN player chooses a suboptimal action instead, then the value of the respective MIN node (and thus the resulting minimax value of its parental MAX node, and inductively for the whole game tree) cannot decrease because the optimal action already led to the minimum value. The same argument can be expanded inductively to prove the assertion.

Giving an example, in Figure 1 let us assume that the MIN player always selects the second best available action. These suboptimal moves would then mean that the MAX player would receive 4, 0, -8 or 3 if he opted for B, C, D or E respectively. Since the optimal move found by minimax leads to E, the resulting value would increase from 2 to 3. However, moving to B would actually result in an even better value (4), thus illustrating that when the MAX player can predict a suboptimal play by MIN, there may be better strategies than following the minimax decision.

Exercise 2

Given a binary mask M and two parental solutions A and B, their offspring C produced by the genetic algorithm will inherit the value A_i for each $M_i = 0$ and B_j for each $M_j = 1$, where i, j are indices of A, B, M, C. Therefore, for $A = 00110101$ and $B = 11010100$ the offsprings would be:

a) $M = 00000111 \Rightarrow C = 00110100$
b) $M = 00111000 \Rightarrow C = 00010101$
c) $M = 00110110 \Rightarrow C = 00010101$

Exercise 3

Part A

First, we need to consider that Part B requires a binary constraint satisfaction problem (CSP) because the AC-3 algorithm can only be used with binary constraints. Let $f(T_i)$ be the number of offices of department T_i, then the following observations lead to such a binary CSP formulation:

1. We can see from the data that $\sum_{i=1}^{12} f(T_i) = 120$, in other words the company needs 120 offices in total. Given that each floor can provide at most 40 offices, each floor must have all its offices occupied (otherwise the sum of occupied offices will not be 120 and consequently not all departments will be housed).

2. Since each floor will have exactly 40 occupied offices, we can easily find the lower and upper bound for the number of departments per floor. By summing the offices of the three largest departments ($f(T_3) + f(T_4) + f(T_6) = 36$), we deduce that each floor will have $\geq$4 departments. Moreover, if a floor houses >4 departments, then at least one floor must have <4 departments, which is a contradiction. Therefore, each floor will house exactly 4 departments.

3. We know that some departments should be on the same floor ($f(T_1)$ with $f(T_3)$, $f(T_2)$ with $f(T_4)$, $f(T_8)$ with $f(T_9)$). Since no combination of these pairs can occupy exactly 40 offices ($f(T_1) + f(T_3) + f(T_2) + f(T_4) = 39$, $f(T_1) + f(T_3) + f(T_8) + f(T_9) = 38$ and $f(T_2) + f(T_4) + f(T_8) + f(T_9) = 37$), we conclude that each pair should be assigned to a different floor.

We are now able to define the binary CSP problem. Let $X_i \in \{T_5, T_6, T_7, T_{10}, T_{11}, T_{12}\}$, $i \in \{1, 2, 3, 4, 5, 6\}$ be the remaining departments to be allocated, then the binary constraints will be:

$$
\begin{aligned}
&X_i \neq X_j, \forall (i, j) : i \neq j \\
&f(X_1) + f(X_2) + f(T_1) + f(T_3) = 40 \Rightarrow f(X_1) + f(X_2) = 20 \\
&f(X_3) + f(X_4) + f(T_2) + f(T_4) = 40 \Rightarrow f(X_3) + f(X_4) = 21 \\
&f(X_5) + f(X_6) + f(T_8) + f(T_9) = 40 \Rightarrow f(X_3) + f(X_4) = 22
\end{aligned}
$$

Part B

The aforementioned constraint equations contain 30 arcs in total (since all nodes of its graph representation are connected).