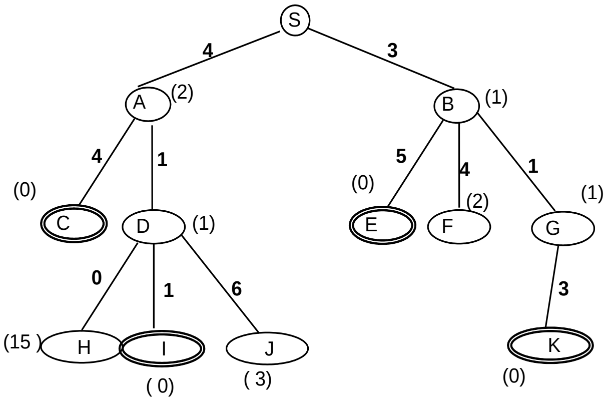
COMP3308/3608 Artificial Intelligence

Week 3 Tutorial exercises
A* algorithm. Heuristics. Local search algorithms.

<u>Exercise 1 (Homework).</u> **A* search**

Consider the tree below. Step costs are shown along the edges and heuristic values *h* are shown in brackets. The goal nodes are circled twice, i.e. they are C, I, E and K.



Run the A* search algorithm. Show the list of expanded nodes and the solution path found with its cost. If there are nodes with the same priority for expansion, expand the last added first.

Expanded nodes:
Path found:
Path cost:

<u>Exercise 2.</u> **Admissible heuristics – 4-queens puzzle**

You are given a 4x4 chess board and the goal is to place 4 queens on the board so that no queen can capture any other. That is, only one queen can be in any row, column or diagonal of the 4x4 array. Suppose we try to solve the puzzle using the following problem space: the start node is labeled by an empty 4x4 array; the successor function creates new 4x4 arrays containing one additional <u>legal</u> placement of a queen anywhere in the array; the goal test is satisfied if and only if there are 4 queens in the array that are legally positioned. Note that all goal nodes are precisely 4 steps from the start node as at each step we place 1 queen and the total number of queens is 4.

Explain whether each of the following is an admissible heuristic:
a) *h(n) = depth (n),* i.e. where *depth(n)* is the depth of the node *n* in the search tree
b) *h(n) = 4 -queens(n),* where *queens(n)* is the number of queens on the board in state *n* (i.e. the number of queens already placed)
c) *h(n) = 4 -queens(n) -available(n)/16*, where *available(n)* is the number of available squares in the array that can receive a new queen in state *n*
d) Can you solve the 4-queens puzzle?

<u>Exercise 3</u>**. Admissibility and consistency**

a) Does an admissible heuristic *h* lead to non-decreasing *f* values along any path? In other words, are all admissible heuristics consistent? Give an example to illustrate your answer.
b) Does a non-decreasing *f* value along any path imply admissibility of *h*?

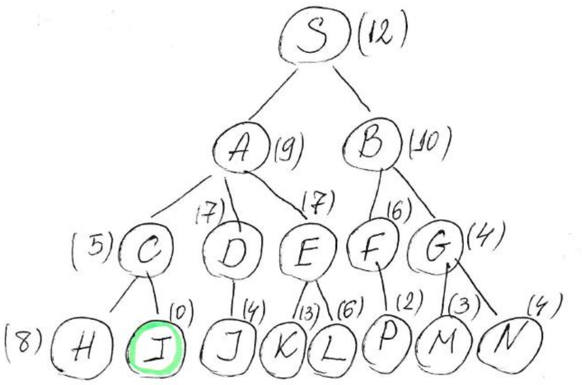
c) A* uses admissible heuristics. What happens if we use a non-admissible one? Is it still useful to use A* with a non-admissible heuristic?

Exercise 4*. A

A * does not terminate until a goal node is selected for expansion from the fringe. However, a path to a goal node might be reached long before the node is selected for expansion, i.e. there may be already a goal node in the fringe. Why not terminate as soon as a goal node has been added to the fringe? Illustrate your answer with an example.

***Exercise 5*. Hill-climbing and beam search**

Consider the tree below. The *v*-value for each node is shown in brackets (this is the value of the heuristic evaluation function). The goal node is I, shown in green.



a) Run the **hill-climbing** algorithm; the smaller the value, the better (i.e. hill-climbing descent).

Show the list of expanded nodes. Was the goal node found?

b) Now run the **beam search** algorithm with k=2. Again - the smaller the value, the better.

Show the list of expanded nodes. Was the goal node found?

***Exercise 6*. Genetic algorithms** (adapted from Dunham, Data Mining, 2003, Pearson Education)

Given is the following initial population: {101010, 001100, 010101, 000010}. Apply the following genetic algorithm to find a better population:

Fitness function: It is defined as the sum of the bit values of each individual.
Selection: At each iteration select for crossover: 1) the best individual and the second best individual and 2) the best individual and the third best individual. If there are ties, resolve them randomly.
Crossover: Always choose the same crossover point, between the 3rd and 4th bit.
Mutation: Mutation means negating a bit. Always mutate bit 2 (the bit numbering starts from 1, from the left).
Stopping condition: The average fitness for the entire population is greater than 4.

Show the new population. How many iterations were needed?

<u>Exercise 7 (Advanced students only)</u> **Gaschnig's heuristic**

At the lectures we defined a relaxation of the 8-puzzle problem in which a tile can move from square A to square B if B is blank. The exact solution of this problem defines the Gaschnig's heuristic. One way to compute this heuristic is the following:

Let B be the location of blank in the current state.
Compare the current state with the goal state:

> If B in the current state is occupied by a tile X in the goal state, move X to B in the current state (i.e. swap X and B);
> Otherwise, move any misplaced tile to B.

The value of the heuristic for a given state is the number of switches required to reach the goal state.

a) Apply the Gaschnig heuristic to solve the 8-puzzle from this initial state:

start	goal
2B6	123
134	456
758	78B

b) Explain why the Gaschnig's heuristic dominates the heuristic *h1* from the lectures (*h1* - the number of misplaced tiles).

c) Recall the heuristic *h2* from the lectures - the Manhattan distance. Show cases when the Gaschnig heuristic is more accurate than both *h1* and *h2,* i.e. returns a higher value.