National University of Singapore
School of Computing
CS3243 Introduction to AI

Tutorial 3: Informed Search

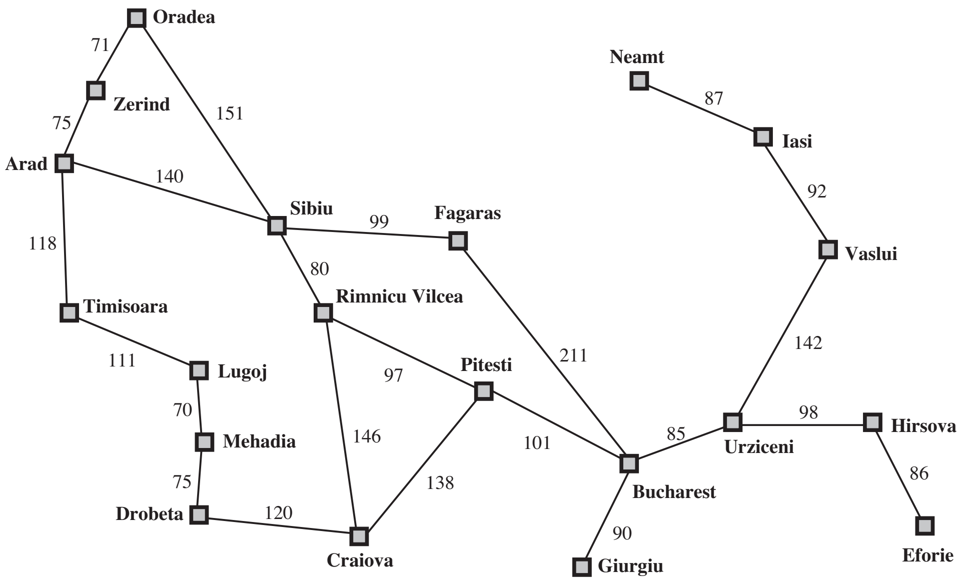
Issue: February 5, 2015 Due: February 13, 2015

Important Instructions:

- *Your solutions for this tutorial must be TYPE-WRITTEN.*
- *Make TWO copies of your solutions: one for you and one to be SUBMITTED TO THE TUTOR IN CLASS. Your submission in your respective tutorial class will be used to indicate your CLASS ATTENDANCE. Late submission will NOT be entertained.*
- *YOUR SOLUTION TO QUESTION* 5 *WILL BE GRADED for this tutorial.*
- *You may discuss the content of the questions with your classmates. But everyone should work out and write up ALL the solutions by yourself.*

1. Consider the 8-puzzle that we discussed in class. Suppose we define a new heuristic function h_3 which is the average of h_1 and h_2, and another heuristic function h_4 which is the sum of h_1 and h_2. That is,
$$h_3 = \frac{h_1 + h_2}{2}$$
$$h_4 = h_1 + h_2$$
where h_1 and h_2 are defined as "the number of misplaced tiles", and "the sum of the distances of the tiles from their goal positions", respectively. Are h_3 and h_4 admissible? If admissible, compare their dominance with respect to h_1 and h_2.

2. Refer to the Figure 1 below. Apply the best-first search algorithm to find a path from Fagaras to Craiova, using the following evaluation function $f(n)$:
$$f(n) = g(n) + h(n)$$
where $h(n) = |h_{SLD}(\text{Craiova}) - h_{SLD}(n)|$ and $h_{SLD}(n)$ is the straight-line distance from any city n to Bucharest given in Figure 3.22 of AIMA 3rd edition (reproduced in Fig. 1).
   (a) Trace the best-first search algorithm by showing the series of search trees as each node is expanded, based on the `TREE-SEARCH` algorithm below (Fig. 2).



Values of h_{SLD} - straight-line distances to Bucharest

Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Figure 1: Graph of Romania.

(b) Prove that $h(n)$ is an admissible heuristic (*Hint: Consider using the Triangle Inequality*).

3. (a) Given that a heuristic h is such that $h(G) = 0$, where G is any goal state, prove that if h is consistent, then it must be admissible.

   (b) Give an example of an admissible heuristic function that is not consistent.

   (c) Is it possible for a heuristic to be consistent and yet not admissible? If not, prove it. If it is, define one such heuristic.

```
function TREE-SEARCH(problem) returns a solution, or failure
  initialize the frontier using the initial state of problem
  loop do
     if the frontier is empty then return failure
     choose a leaf node and remove it from the frontier
     if the node contains a goal state then return the corresponding solution
     expand the chosen node, adding the resulting nodes to the frontier
```

Figure 2: Tree search algorithm.

4. Assume that we have the following initial state and goal state for the 8-puzzle game. We will use h_1 defined as "the number of misplaced tiles" to evaluate each state.

1	2	8
	4	3
7	6	5

initial state

1	2	3
8		4
7	6	5

goal state

   (a) Apply the hill-climbing search algorithm in Figure 4.2 of AIMA 3rd edition (reproduced below). Can the algorithm reach the goal state?

function HILL-CLIMBING(*problem*) **returns** a state that is a local maximum

  current ← MAKE-NODE(*problem*.INITIAL-STATE)
  loop do
    neighbor ← a highest-valued successor of *current*
    if neighbor.VALUE ≤ current.VALUE **then return** *current*.STATE
    current ← *neighbor*

Figure 4.2 The hill-climbing search algorithm, which is the most basic local search technique. At each step the current node is replaced by the best neighbor; in this version, that means the neighbor with the highest VALUE, but if a heuristic cost estimate h is used, we would find the neighbor with the lowest h.

   (b) Identify a sequence of actions leading from the initial state to the goal state. Is it possible for simulated annealing to find such a solution?

5. You have learned before that A* using graph search is optimal if $h(n)$ is consistent. Does this optimality still hold if $h(n)$ is admissible but inconsistent? Using the graph in Figure 3, let us now show that A* using graph search returns the non-optimal solution path (S, B, G) from start node S to goal node G with an admissible but inconsistent $h(n)$. We assume that $h(G) = 0$.

   Give nonnegative integer values for $h(A)$ and $h(B)$ such that A* using graph search returns the non-optimal solution path (S, B, G) from S to G with an admissible but inconsistent $h(n)$, and tie-breaking is not needed in A*.

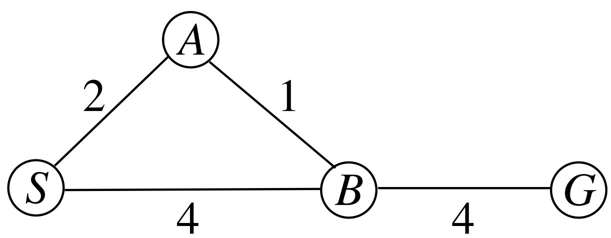


Figure 3: Graph.