HANOI UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY
Spring 2009, Artificial Intelligence

TUTORIAL 4: INFORMED SEARCH ALGORITHMS

Tutor: Tran Quang Anh

In this tutorial, we will work on the Greedy Best First Search and A* search algorithms.

1. General information

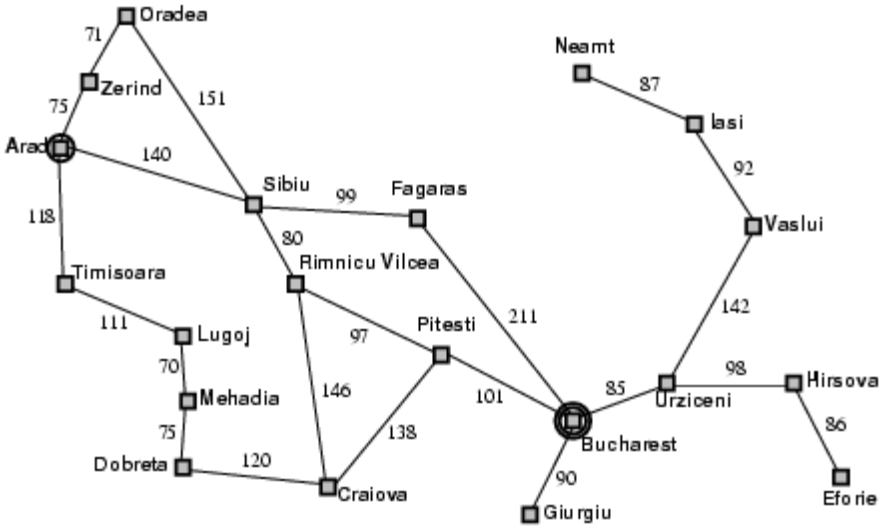
- Server: 192.168.6.251
- Username: YOUR NAME, for example: haintv_1c05
- Password: YOUR NAME, for example: haintv
- aima code (read only): /usr/local/acl81_express/aima_code/
- Common Lisp: /usr/local/acl81_express/allegro-express

2. Run Common Lisp and load the code:

- Run CommonLisp:
/usr/local/acl81_express/allegro-express
- Load AIMA code:
(load "/home/ait/aima_code/aima.lisp")
- Load aima's search part
(aima-load 'search)
- Compile (you need to do this one time and when the code is changed):
(aima-compile)

3. Test the problem-solving agent:

3.1 The Romanian map:



Note: The agent knows about the whole map at the beginning: The map is stored in the file: search/domains/route-finding.lisp.

3.2 Problem formulation

(setq p1 (make-romanian-problem :initial-state 'Arad :goal 'Bucharest))

Note: A problem is defined by four items: initial state, actions, goal test and path cost. The actions and path cost are defined as the features of the Romanian map. Therefore, in the problem formulation, we need to specify only the initial-state and the goal.

3.3 Search for a solution using Greedy best first search algorithm

- Breadth-first search algorithm:
(setq r1 (greedy-search p1))

Note: Greedy best first search - "Best-first search using H (heuristic distance to goal). [p 93]"

- Display the solution:
(solution-actions r1)

Note: Solution is a sequence of actions that lead to the goal states.

3.4 Search for a solution using A*-search algorithm

- A*- search algorithm:
(setq r1 (tree-a*-search p1))

Note: A-search – "Best-first search using estimated total cost, or (F = G + H). [p 97]"*

4. The missionaries and cannibals problem

4.1 The problem

In the missionaries and cannibals problem, two missionaries and two cannibals must cross a river using a boat which can carry at most two people, under the constraint that, for both banks, if there are missionaries present on the bank, they cannot be outnumbered by cannibals (if they were, the cannibals would eat the missionaries.) The boat cannot cross the river by itself with no people on board.

4.2 Can you solve the problem by pen and paper? What is the result?

4.3 Using AI to solve the problem

4.3.1 Problem formulation

"The problem is to move M missionaries and C cannibals from one side of a river to another, using B boats that holds at most two people each, in such a way that the cannibals never outnumber the missionaries in any one place."

(setq p1 (make-cannibal-problem :initial-state (make-cannibal-state :m1 2 :c1 2)))

4.3.2 Search for a solution using A*-search algorithm

- A*- search algorithm:
(setq r1 (tree-a*-search p1))

Note: A-search – "Best-first search using estimated total cost, or (F = G + H)."*

"The state says how many missionaries, cannibals, and boats on each side. The components m1,c1,b1 stand for the number of missionaries, cannibals and boats, respectively, on the first side of the river. The components m2,c2,b2 are for the other side of the river."

We need to represent both sides (rather than just one as on); because we have generalized from 2+2 people to M+C. Incidently, we also generalized from 1 boat to B boats:

(m1 2) (c1 2) (b1 1) (m2 0) (c2 0) (b2 0)

4.3.3 Display the solution:

(solution-actions r1)

Note: Solution is a sequence of actions that lead to the goal states.

An action is a triple of the form (delta-m delta-c delta-b), where a positive delta means to move from side 1 to side 2; negative is the opposite. For example, the action (1 0 1) means move one missionary and 1 boat from side 1 to side 2.

4.3.4 Run the agent

(run-environment (problem->environment p1))

Is this solution as the same as your solution above?

4.4 What is the heuristic function in the missionaries and cannibals problem?