T-93.4400 Principles of Artificial Intelligence
HUT, Laboratory of Software Technology

Exam
07.01.2011

Task 1.

Describe the behaviour of depth-first, breadth-first and uniform-cost search, and compare their relative advantages and disadvantages. (3 points)

How does "informed search" try to improve on "uninformed search"? (3 points)

Task 2.

(a) Describe the principles of the A* -search procedure.

(b) Apply the A* -search procedure to a problem, where the following state transitions are possible:

Transition	Its cost
S → A	2
S → B	1
S → C	2
A → D	2
B → E	1
C → F	1
D → H	2
E → H	8
E → I	7
F → I	2
H → G	1
I → G	2

The problem is to find a path from the state S to the state G and the values of the function h in the different states are the following $h(A) = 3, h(B) = 3, h(C) = 3, h(D) = 2, h(E) = 2, h(F) = 2, h(H) = 1$, and $h(I) = 1$. Which one of the two paths giving the minimun costs your algorithm will find and why?

(evaluation: point a) max. 3 points, point b) max. 3 points)

Task 3.

Let us consider the following statements:

- Jussi likes all foods.
- Apples are food.

- Frankfurters are food.
- Everything, which somebody eats and which is not lethal (i.e. poison), is food.
- Ville eats potatoes and is still alive.
- Sari eats everything, which Ville eats.

(a) Transform the statements into expressions in the predicate logic. (1 p) Pay special attention to the representational adequacy of your representation, i.e. that the inferences below are possible. Add, if needed, new statements to make the implicit knowledge needed explicit.

(b) Show that Jussi likes potatoes using backward chaining. (1 p)

(c) Transform the expressions created in step a in the clause form. (1 p)

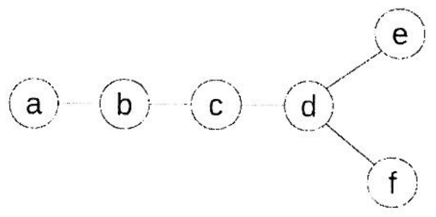
(d) Show that Jussi likes potatoes using resolution. (1 p)

(e) Find, using resolution, an answer to the question: "What (food) does Sari eat?" (1 p)

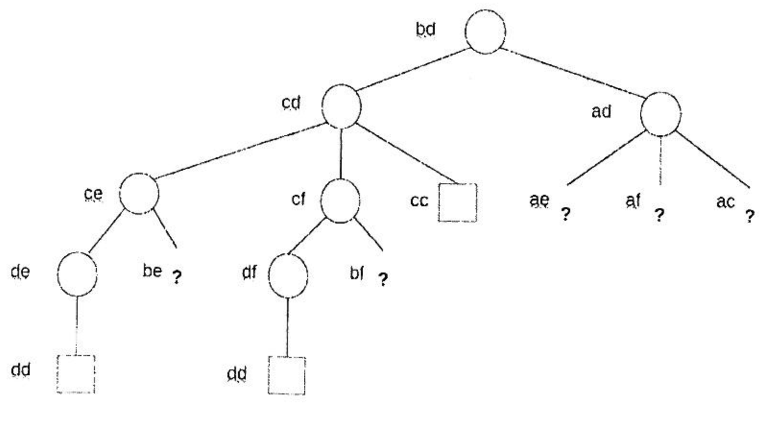
(f) + 1 p, if all the steps a - e are correct.

Task 4.

We have a two-player *pursuit-evasion* game, where two players P and E take turns moving. P tries to catch E and E tries to run away from P. The game ends only when the playes are on the same node. The payoff to P, the pursuer, is minus the total time taken. The evader is considered to win if never being caught. The map is depicted in the figure. Arcs represent access from node to node and the cost of moving for each arch is 1. Initially pursuer P is at node **b** and evader E at node **d**.



Partially constructed game tree for the game on the map is shown below. Each node is labelled with the P , E positions. P moves first. The structure of the tree at the nodes marked with question mark are currently unknown. You need to replicate the game tree in your answer and complete it with the requested information.



(a) Mark the values of the terminal nodes.

(b) Inside each internal node, write the strongest fact you can infer about its value (either a number, one or more inequalities such as ≥ 14, or a *?*).

(c) Can shortest-path lengths on the map be used to bound the values of the *?* leaves. If so, why and how? If not, why not?

(d) Mark inequalities on all the *?* leaves according to the method in (c). Remember the cost to get to each node as well as the cost to solve it.

(e) Now suppose the tree as given, with the leaf bounds from (d), was evaluated left-to-right. CIRCLE those nodes *?* nodes that would not need to be expanded further, given the bounds from part (d), and CROSS OUT those that need not be considered at all.

(f) Can you say anything precise about who wins the game on a map that is a tree?

Task 5.

Let us consider the following planning problem. In the start state Rocket1, Parcel1, and Parcel2 are on the Earth. In the goal start state Parcel1 and Parcel2 are on the Moon and Rocket1 is on the Earth. We have at our disposal three operators Load(parcel, rocket), Unload (parcel, rocket), and Fly(rocket, starting-point, destination), which may be divided into two operators (for flying loaded and flying unloaded), if needed. The capacity of Rocket1 is one parcel, i.e. only one parcel can be carried at a time - not two parcels. Define the operators as STRIPS- operator schemas, code the problem using these STRIPS-operators, and solve the problem in the POP-planner style. (You do not need to write down the POP (Partial-Order Planner) code, only to show, e.g. through pictures, how the solving of this problem goes.)