The University of British Columbia
Department of Computer Science
Midterm Examination - Fall 2003

Computer Science 322
(Introduction to Artificial Intelligence)

Total marks: 48
Time: 60 minutes

Question 1 [7 marks]

Consider the following knowledge base, KB:

$a \leftarrow c \wedge b \wedge d.$	$f.$
$b \leftarrow c.$	$d \leftarrow e \wedge k.$
$c \leftarrow e.$	$h.$
$c \leftarrow f \wedge h.$	$j \leftarrow f.$

(a) [2 marks] Give 3 atoms that are logical consequences of KB.
(b) [2 marks] Give 3 atoms that are *not* logical consequences of KB.
(c) [3 marks] Give the final consequence set produced by a bottom-up proof procedure from KB.

Solution

(a) Any three from b, f, h, c, j.
(b) Any three from a, d, e, k.
(c) An example sequence:

$f.$
$h.$
$j.$
$c.$
$b.$

1 mark for the 2 facts within the knowledge base, 2 marks for the logical consequences that were the heads of rules.

Question 2 [12 marks]

Five variables, each with a set of possible values, and a set of constraints have been used to describe a particular creature. Draw a consistency graph to illustrate the given information and use the ARC-3 algorithm to determine which of the possible creatures listed has been described.

Possible creatures: elephant, dragonfly, rhinoceros, tiger, eagle.

Variables:

Horns or Tusks(H)	values: {0, 1, 2}
Eyes(E)	values: {0, 2, 4, 1000}
Wings(W)	values: {0, 2, 4}
Legs(L)	values: {0, 2, 4, 6}
Tails(T)	values: {0, 1}

Constraints:

$E > W$	$W > 2 \times T$	$T > H$
$H = 0$	$H \neq W$	$L < E$
$L > W$	$L > T$	

Solution

See Figure 1.

6 marks for a correct initial graph - 1/2 for each correct node, 1/2 for each correct arc.
1/2 mark for each correct domain reduction - max 4 marks as this can be done in approx. 8 steps
1 mark for naming the correct animal

Question 3 [6 marks]

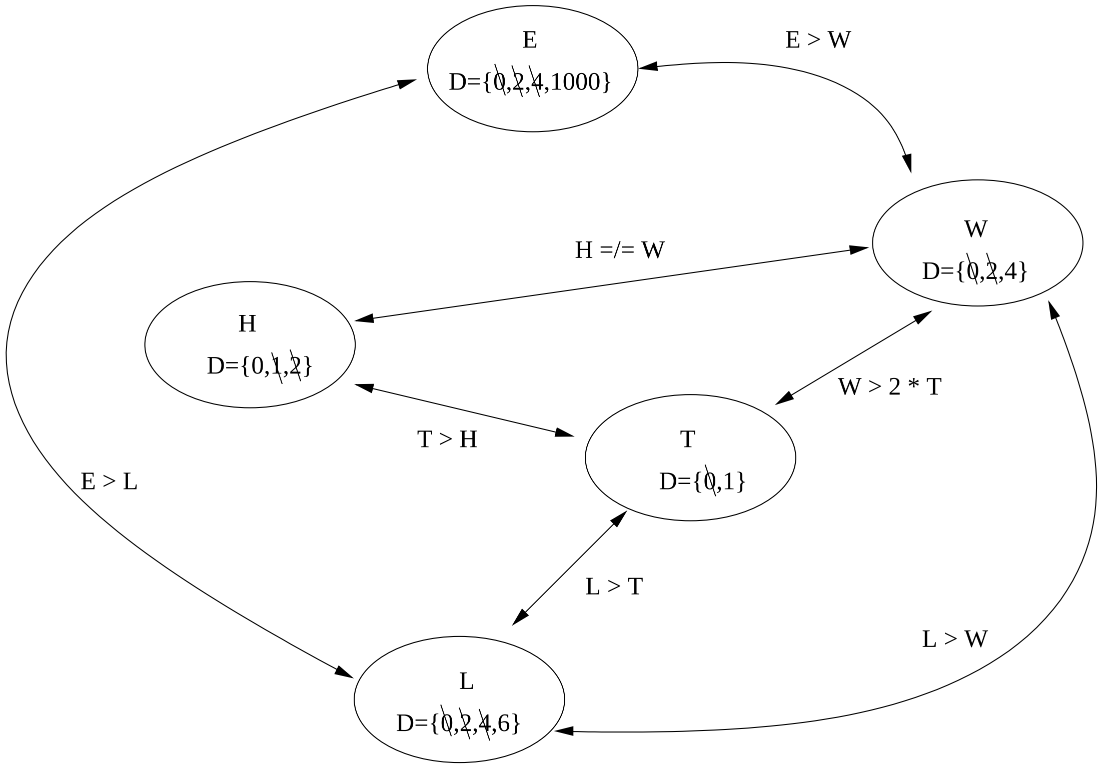
Consider the following knowledge base, KB:

1. $a(X,Y) \leftarrow b(X) \wedge c(Y)$.
2. $a(X,Y) \leftarrow c(Y) \wedge e(X)$.
3. $b(X) \leftarrow d(X)$.
4. $b(X) \leftarrow e(X)$.
5. $c(m)$.
6. $c(n)$.
7. $d(n)$.
8. $d(p)$.

Give a successful top-down derivation with substitutions for the query

$$?a(N,M)$$

State which clause is chosen from the knowledge base and the substitution set used at each step.



The animal is a DRAGONFLY.

Figure 1: Constraint graph for the animal domain

Solution

Here is one derivation, always selecting the leftmost atom in the body and the first appropriate clause in KB.

```
yes(N,M) <- a(N,M).
   choose clause 1, substitution: {X/N, Y/M}
yes(N,M) <- b(N) & c(M).
  choose clause 3, substitution: {X/N}
yes(N,M) <- d(N) & c(M).
  choose clause 7, substitution: {N/n}
yes(n,M) <- c(M).
  choose clause 5, substitution: {M/m}
yes(n,m) <-
```

The answer is *N* = *n*, *M* = *m*.
Other possible answers were:

- $N = n, M = n$,
- $N = p, M = m$,
- $N = p, M = n$

You got 1/2 a mark for each appropriate choice of clause, and each correct substitution set (a total of 4 marks). 1/2 a mark was given for the correct initial answer clause, 1/2 a mark was given for the correct formation of the 3 intermediate clauses in the derivation.

Question 4 [17 marks]

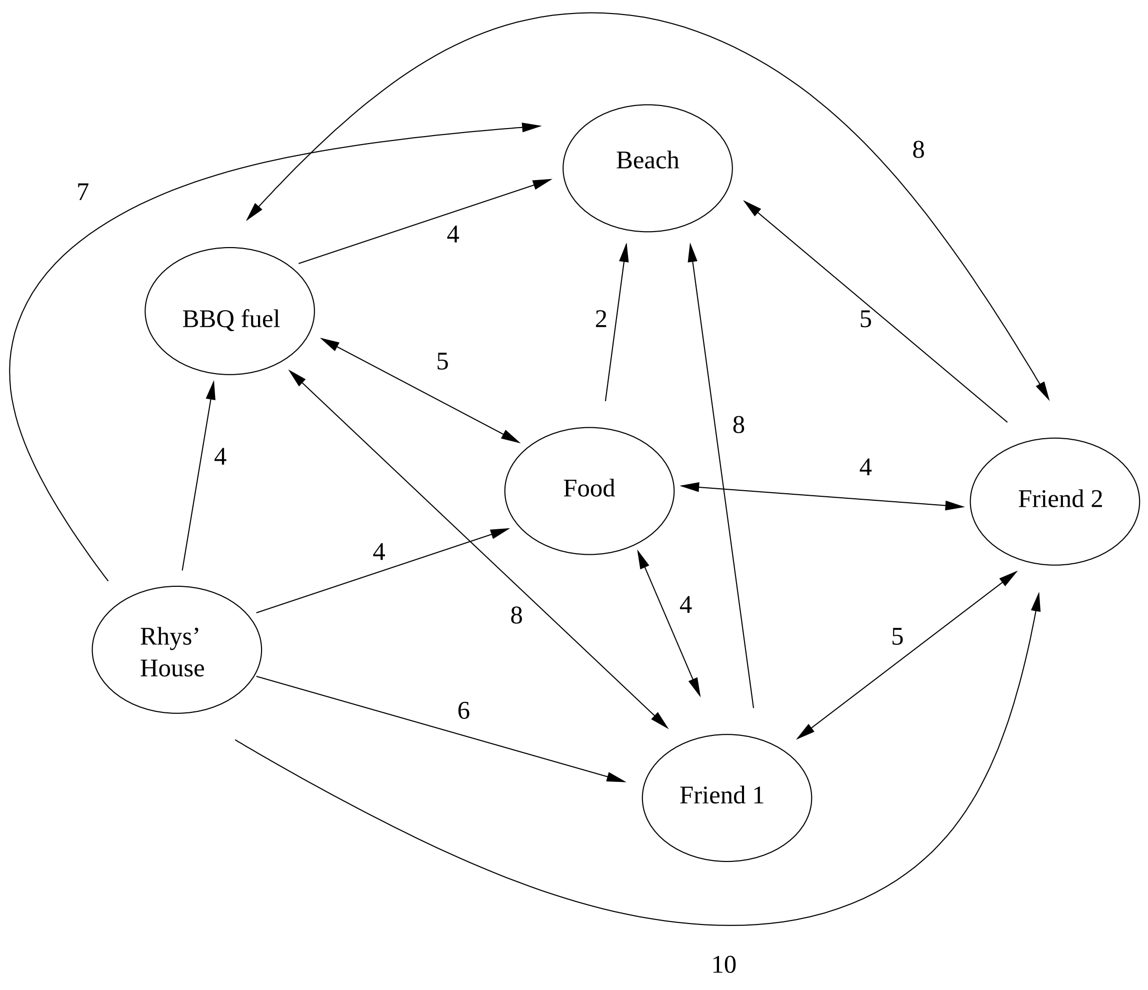


Figure 2: Travel Times between Locations

Rhys is attempting to get from his house to the beach for a barbeque in the shortest possible time. He also needs to collect a few items on the way: fuel for the barbeque, food to cook, and two friends that live in different locations. The graph in Figure 2 shows all of the locations that he must visit, with the travel times between pairs of locations listed in minutes on the arcs between the nodes.

For each of the following search algorithms answer these questions:

1. Is the algorithm guaranteed to halt and return an answer?
2. If it always halts, is the algorithm guaranteed to return the best answer? Why?
3. Give an example of the first 4 paths that will be evaluated.The order in which the nodes are chosen is up to you, there is no 'correct' order for this question.
4. What effect would multiple-path pruning have on the performance of the algorithm?

(a) Depth first
(b) Breadth first
(c) Lowest cost first
(d) A* - Assume that $h(n)$ is equal 2 times to the minimum number of additional arcs that must be travelled in order to arrive at the beach having visited all the nodes in the graph. e.g.

Nodes visited	Nodes to visit	h
Home, Friend1	Friend2, Food, BBQ, Beach	8
Home, Food, BBQ	Friend2, Friend1, Beach	6
Home, Friend1, Friend2, BBQ, Food	Beach	2

For parts 2 and 4, does $h(n)$ satisfy the monotone restriction?

Solution

Let us produce a key for the node names in the graph:

R = Rhys' house
Fd = Food
$F1$ = Friend 1
$F2$ = Friend 2
BBQ = BBQ Fuel
Bch = Beach

(a) Depth first

1. [1 mark] No, depth-first is very unlikely to halt at all.
2. n/a. 0 marks, but you lost 1/2 a mark if you put something in here even if you said no to the earlier question.
3. [1 mark] e.g. R, $R \rightarrow F1$, $R \rightarrow F1 \rightarrow F2$, $R \rightarrow F1 \rightarrow F2 \rightarrow F1$
4. [1 mark] It will enable it to halt because cycles will be removed.

(b) Breadth first

1. [1 mark] Yes, although it may take some time.
2. [1 mark] No. Breadth first does not take the cost of paths into account.
   1/2 mark for saying no, 1/2 mark for saying why.
3. [1 mark] e.g. R, $R \rightarrow F1$, $R \rightarrow F2$, $R \rightarrow Fd$.
   1/4 mark per correct path.
4. [1 mark] It may speed up the algorithm a little by reducing the size of the frontier, but will do little else.

(c) Lowest-cost first

1. [1 mark] Yes, although it may get stuck in a few small loops.

2. [1 mark] Yes, because any lower cost solution would already have been explored and returned. 1/2 mark for saying yes, 1/2 mark for saying why.

3. [1 mark] e.g. R, $R \to Fd$, $R \to BBQ$, $R \to F1$. This is the only answer, although the 2nd and 3rd paths may be explored in reverse order.
   1/4 mark per correct path.

4. [1 mark] It may speed up the algorithm by reducing the size of the frontier, but little else.

(d) A*

1. [1 mark] Yes.

2. [2 marks] Yes, because h is an underestimate. $h(n)$ also obeys the monotone restriction, but that just means that it will find the optimal path more quickly.
   1/2 mark for saying yes, 1 & 1/2 marks for saying why.

3. [1 mark] R, $R \to BBQ$, $R \to Fd$, then one of $R \to F1$, $R \to Fd \to F1$, or $R \to Fd \to F2$.

   1/4 mark for each correct path.

4. [2 marks] None because $h(n)$ obeys the monotone restriction.
   1 mark for saying nothing happens, 1 mark for saying why.

Question 5 [6 marks]

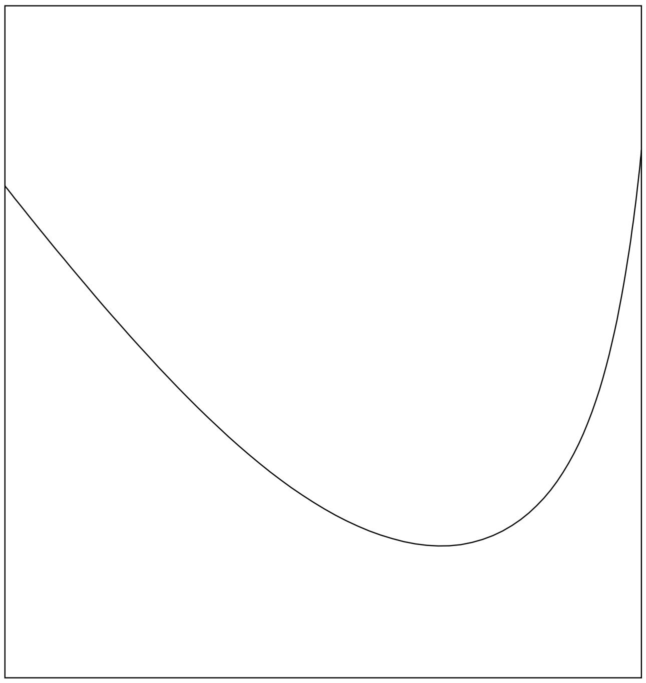
Two 2-dimensional heuristic landscapes are illustrated in Figure 3. For each landscape, say whether simple hill-climbing or stochastic beam search would reach an acceptable solution more quickly (or if they would give roughly the same performance) and why. Assume that the 2 highest peaks in each landscape have been classified as acceptable solutions.

Include the important details of the two methods in your answer. You may present your reasoning about both landscapes within a single answer if you wish.

Solution

Landscape (a) Simple Hill Climbing (SHC) will move to the neighbour of its current position that has the best increase in heuristic value. Once it has reached a local maxima then it will remain there. Since we don't know where the hill-climbing method will start it could go to either one side of the landscape or the other, however once it has started towards one of the two peaks then it will head directly towards it. Since both peaks are classed as acceptable then hill climbing will be guaranteed to find a solution very quickly.

Stochastic beam search (SBS) will start with a set of randomly picked points (let us say, for example, that $k = 5$). From the set of all neighbours, 5 of these neighbours will be picked as the new set of points. The probability that a neighbour is picked is based on its heuristic score. Since randomness and probability are included within the algorithm, we


Landscape a

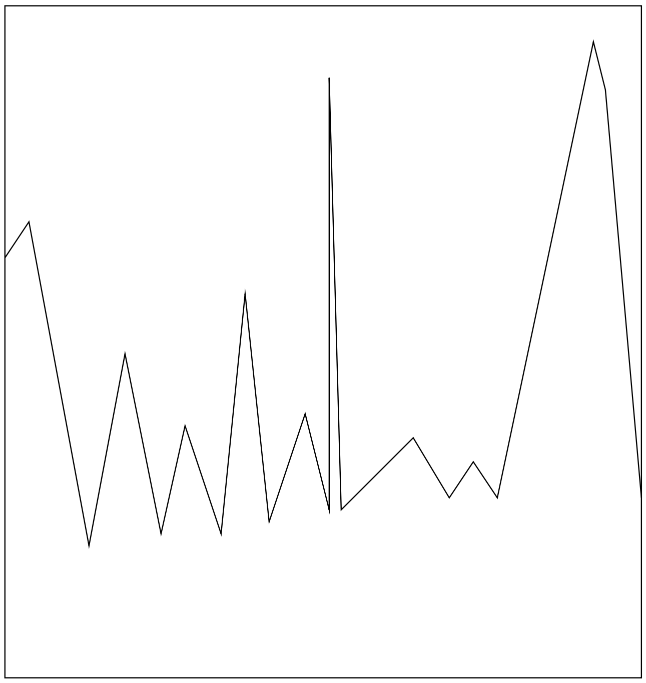

Landscape b

Figure 3: Two 1-dimensional landsapes of heuristic values

cannot say precisely what it will do. At its most efficient, SBS will perform as well as SHC, and may have a slight advantage due to multiple starting points. At its worst, SBS may wander about for quite a while before reaching a peak.

Acceptable choices were either that it was not clear cut as to which algorithm would reach an acceptable peak first on each occasion, or I would accept arguments for SHC based on efficiency.

Landscape (b) Simple hill climbing is very likely to get stuck on local maxima on this one (unless we are very lucky) because it will only climb from its initial starting point and then halt when it reaches a peak. Stochastic beam search will give a much better performance as it starts from multiple points, but more importantly it has the ability to rescue itself from local maxima. Since the choice of new points is probabilistic, it is possible for the algorithm to pick neighbours that are lower than the current points as well as ones that are higher. This will allow the algorithm to escape from difficult situations, including if all the points are located on the same local maxima.

The whole question had 6 marks to be given for a good argument that demonstrated knowledge of the two methods being compared. The marks were not divided into 3 per part as you may have demonstrated more knowledge in the first answer. Key points I was looking for:

2 marks A reasonably argued decision for each landscape, this could include deciding that neither is better.

1 mark What hill-climbing and stochastic beam search actually do, i.e. a brief overview of the algorithms.

1.5 marks State that hill climbing gets stuck on local maxima, and why.

1.5 marks State that Stochastic beam search does not get stuck on local maxima, and why.