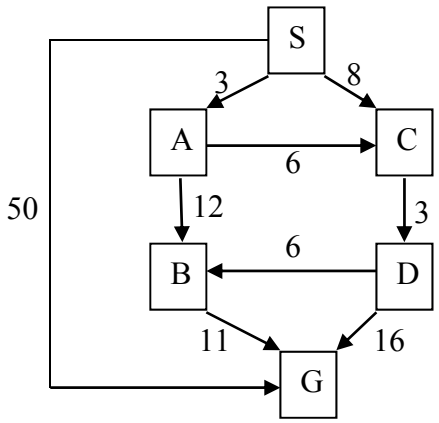
FACIT

Exam - Artificial Intelligence 1DL340 2011-10-21

You may use a dictionary. You can answer in either English or Swedish.

---

1) Search (5p)
Suppose we have a graph with nodes S, A, B, C, D and G, where S is the start and G the goal node. The distances between connected nodes are:



The heuristic estimates of the distance to G are:

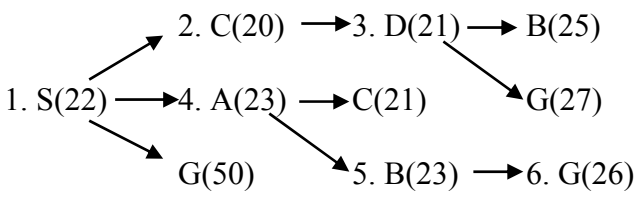
from:	S	A	B	C	D	G
distance:	22	20	8	12	10	0

a) Perform a search with algorithm A. [1p]

I write N(P,E) for node N with parent P and estimate E = f(N) = g(N) + h(N).

Open	**Closed**
S(-,22)	
C(S,20),A(S,23),G(S,50)	S(-,22)
D(C,21),A(S,23),G(S,50)	C(S,20),S(-,22)
A(S,23),B(D,25),G(D,27)	D(C,21),C(S,20),S(-,22)
B(A,23),G(D,27)	A(S,23),D(C,21),C(S,20),S(-,22)
	- C(A,21) is worse than C(S,20) and does not show
G(B,26)	B(A,23), A(S,23),D(C,21),C(S,20),S(-,22)
done.	G(B,26),B(A,23)A(S,23),D(C,21),C(S,20),S(-,22)

Another representation (the number before nodes gives in which order nodes are closed):



b) Is the heuristic optimistic? Explain your answer. [1p]

Yes.

to G from:	S	A	B	C	D	G
heuristic estimate:	22	20	8	12	10	0
actual distance:	26	23	11	19	16	0

The heuristic estimate is always smaller (≤) than the actual distance along the shortest path.

c) What does it mean *for the search process* if the heuristic function optimistic? [1p]

When the goal node is closed, the optimal path to the goal has been found.

d) Is the heuristic function monotonic? Explain your answer. [1p]

No. Between S and C, the distance is 8, but the estimate drops by 10.
A monotonic heuristic never overestimates the distance between nodes. As a consequence, the estimated value can never drop – in the search it drops from 22 to 20 (S to C).
(You can also note the same effect between A and C.)

e) What does it mean *for the search process* if the heuristic function is monotone? [1p]

For every node, when it is closed, the optimal path to the node has been found. This "accidentally" happens in the search above, but there would be no guarantee. For instance, if the distance from S to C were 10, we would first close C(S,22), and later find C(A,21).

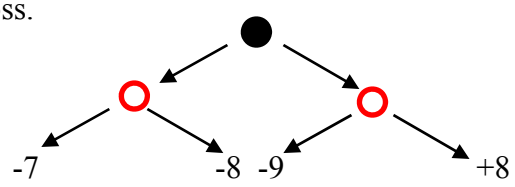
2) Minimax (5p)

a) The minimax algorithm cannot be applied to all games. What properties must a game have for minimax to be applicable? [2.5p]

two players
full information (The game state is completely visible to both players.)
zero-sum (If one loses, the other wins. No win-win situations)
deterministic (A move has a given outcome. No chance effects like dice, random traps)
turn-based = asynchronous (I move, then you move. No simultaneous moves.)
Note: minimax *can* be applied to a game like "go", with a broad search space and hard-to-define static evaluation; it just won't work very well.

b) You are writing a game-playing program that uses minimax to determine its moves. Give *two* reasons why you would program a deviation from minimax in some situations (*which* situations?). Comment: alpha-beta pruning is *not* a deviation from the minimax strategy, it's just an efficient implementation of that strategy. [2.5p]

1. Minimax assumes that your opponent plays optimally (according to your own evaluation). In a situation where all options lead to your loss under this assumption, you might want to choose an option that lets you win if your opponent makes a mistake – rather than minimizing your loss.



Minimax says that black should move left, but moving right may have a better chance of winning.

2. A pure minimax player is deterministic, and thus becomes very predictable for an opponent. In situations where the differences in the evaluations are small, you might randomly pick an acceptable move, rather than always picking the best one. Otherwise the opponent might learn a strategy to defeat you, and apply it all the time.
3. The purpose of the game playing program might not be to play as strongly as possible, but to play at a certain level that is fun for the opponent. (A quick implementation would be to use minimax with a limited search depth, but in a more intelligent implementation the program could with some probability "decide" to make mistakes.)
4. You could include a database for the start of the game (why search the same tree for every game? – just store the result) and/or for endgames (where minimax might not work very well – depends on the game).

The horizon effect is sometimes mentioned. This is indeed a weakness of minimax. But the solution is usually to continue with minimax, and locally deepen the search. So you're not really deviating from the minimax strategy.

3) Natural language (4p)

a) Give at least 3 examples of different kinds of ambiguity in natural language. [2p]

Grammatical: “John saw Bill with a telescope”. Who had the telescope?

Noun-noun: “A coffee cup contains coffee, and a steel cup contains steel”.
(this actually also has a *lexical* ambiguity, as “contains” in the first part means “encloses”, while in the second part it means “partly consists of”)

Referential: “See the red pyramid next to the green block? Put it on it.” Put what on what?

Pragmatics: is “yeah, right” a sign of approval, or sarcasm? In spoken text, the intonation might convey this information.

b) How can we deal with such ambiguities when creating a program that translates text? [2p]

1. Try to resolve the ambiguity.
    a. World knowledge: knowing properties of coffee, steel and cup. Knowing that you cannot put a block on a pyramid (the word order might help too).
    b. Contextual information. “Hi Bill, are you studying astronomy?”, John asked. (it requires world knowledge to link “astronomy” to “telescope”).

2. Statistical information. Know that “coffee cup” often translates to “CUP FOR COFFEE” and “steel cup” often translates to “CUP OF STEEL”. (Lacking another common language, I translate to “capitalistic” here.)

4) Bayes' Theorem (5p)

I don't have a car. I come to work either by bike or by bus. If I take the bus, there is a 10% chance that I'm late. If I take the bike, there is a 2% chance that I'm late. I take the bike 4 days out of 5. Today I was late.

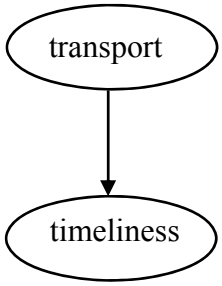
a) Write down and explain the formula used in Bayes' theorem for this problem. [2p]

$$P(\text{bus} \mid \text{late}) = \frac{P(\text{late AND bus})}{P(\text{late})} = \frac{P(\text{late} \mid \text{bus}) * P(\text{bus})}{P(\text{late} \mid \text{bus}) * P(\text{bus}) + P(\text{late} \mid \text{bike}) * P(\text{bike})}$$

b) Use Bayes' theorem to calculate the probability that I took the bus today. [1p]

$$P(\text{bus} \mid \text{late}) = \frac{0.1 * 0.2}{0.1*0.2 + 0.02*0.8} = \frac{2}{2+1.6} = 20/36 = 5/9.$$

c) Model the situation as a Bayesian network with 2 nodes, and give the conditional probability distribution for both nodes. [2p]



	bus	bike
	0.2	0.8

transport	late	on time
bus	0.1	0.9
bike	0.02	0.98

5) Decision trees (5p)

a) Explain briefly the concept *decision tree*. [2p]

A decision tree classifies "something" by asking questions. Each internal node contains a question, and has a child for each possible answer to that question. Starting from the root, answer each question and go to the child that corresponds to your answer. When you reach a leaf, no more questions are needed and the leaf tells you what your "something" is.

The ID3 algorithm "learns" a decision tree from examples. These examples would be consistent with several different decision trees. Describe

b) *which* decision tree ID3 learns [0.5p], and

The simplest one.

c) *why* this tree is preferable over others [1p], and

Occam's Razor: as a heuristic, simpler explanations are usually more correct than complex ones. So: the argument is about what is likely to be correct*, it's not about efficiency*.

d) outline *how* ID3 constructs this tree from the examples. [1.5p]

Recursively from the top.

The root question is the one that provides the highest *information gain*.

Each possible answer results in a smaller problem (fewer examples, one less possible question).

Base cases:
1. All examples belong to one category. This is the result.
2. Out of questions. The available data cannot distinguish the remaining examples.

Information gain = [information contained in all examples] – $\sum_i$ P(Answer = a_i)[further information contained in examples that match answer a_i].

6) AI Concepts (6p)

Describe the following concepts with a few sentences.

a) Iterative deepening [1p]

An iteration of bounded depth first searches, with an increasing depth. It combines the low memory use of depth-first search with the completeness (and guarantee to find the shortest solution) of breadth-first search.

b) Abductive reasoning (abduction) [1p]

From observation q, and rule p implies q, derive p as a plausible explanation of q. This can be done purely in logic, or probabilistically (the observation q increases the likelihood of p).

c) Inductive bias [1p]

What you learn is determined by what you can learn. The search space for learning is always huge, so some limitation is necessary. A limitation will make it impossible to learn some things, and make it easier to learn other things. You cannot learn what you cannot represent.

d) Reinforcement learning [1p]

is a learning situation that is based on rewards (positive or negative). In contrast to supervised learning, the reward is not linked to an immediate question and answer, but it can be given after a number of steps. For example, the reward can be given for winning a game; it's up to the program to figure out which moves contributed most. (Picture: "learning inspired by nature, slide 6)

e) Genetic programming [1p]

is a learning technique. There is a population of potential solutions (computer programs). There are 3 aspects: selection, combination and mutation. The best ("fittest") solutions are selected for reproduction, there is a combination operator that takes aspects from the selected solutions and combines them to (hopefully) even better solutions. In order to explore more of the search space, some random mutations may be added.

f) Turing test [1p]

The Turing test provides a possible definition of AI. In its simplest form, it proposes an experiment where in interviewer has to distinguish a person from a computer system only by written questions and answers. The system is intelligent if the interviewer cannot make the distinction. This is a "black box" definition: it does not discuss the methods used by the computer system, only its output.

Grades

*Important note: these boundaries do apply to **this** exam. Future exams may have different bounds or grading principles.*

up to 13:	U (fail)
13.5 - 17.5:	3
18 - 22:	4
22.5 - 30:	5