Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2017-08-21
Sal (1)	TER3(15)
Tid	8-12
Kurskod	TDDC17
Provkod	TEN1
Kursnamn/benämning **Provnamn/benämning**	Artificiell intelligens En skriftlig tentamen
Institution	IDA
Antal uppgifter som ingår i tentamen	8
Jour/Kursansvarig Ange vem som besöker salen	Olov Andersson
Telefon under skrivtiden	070 574 33 43
Besöker salen ca klockan	ca kl. 10
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Anna Grabska Eklund, anna.grabska.eklund@liu.se, ankn. 2362
Tillåtna hjälpmedel	Miniräknare/Hand calculators
Övrigt	
Antal exemplar i påsen	

Linköpings Universitet
Institutionen för Datavetenskap
Patrick Doherty

Tentamen
TDDC17 Artificial Intelligence
21 August 2017 kl. 08-12

Points:

The exam consists of exercises worth 36 points.
To pass the exam you need 19 points.

Auxiliary help items:

Hand calculators.

Directions:

You can answer the questions in English or Swedish.
Use notations and methods that have been discussed in the course.
In particular, use the definitions, notations and methods in appendices 1-2.
Make reasonable assumptions when an exercise has been under-specified.
State these assumptions explicitly in your answer.
Begin each exercise on a new page.
Write only on one side of the paper.
Write clearly and concisely.

Jourhavande: Olov Andersson, 070 5473343. Olov will arrive for questions around 10.00.

1. The following questions pertain to the course article by Newell and Simon entitled *Computer Science as an Empirical Enquiry: Symbols and Search.*
   (a) What is a *physical symbol system* (PSS) and what are its structural and conceptual components? **[2p]**
   (b) What is the Physical Symbol System Hypothesis? **[1p]**
   (c) Do you think the Physical Symbol System Hypothesis provides an adequate description of the structural and conceptual components required for a system exhibiting intelligence? Provide reasonable justifications for your opinion. **[1p]**
   (d) What is the heuristic search hypothesis? **[1p]**

2. The following questions pertain to machine learning. Give detailed answers.
   (a) Explain the significance of the *backpropagation* algorithm for training neural networks. **[1p]**
   (b) Give some intuition of why a deeply layered neural network can be more effective than a shallow one. **[1p]**
   (c) Consider a Q-learning agent with the update equation shown below. Assume it has converged to an optimal policy π^* for its environment.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R(s_t) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \quad (1)$$

      i. Explain how the function $Q(s_t, a_t)$ relates to $R(s_t)$.**[1p]**
      ii. Explain *the curse of dimensionality* in the context of Q-learning. **[1p]**

3. Consider the following logical theory about registered voters (where hilary, bernie and shawn are constants) and we view grounded atomic formulas as propositional atoms. (In this case unification of two grounded atomic formulas is successful when they are identical.):

$$\mathit{Democrat}(\mathsf{hilary}) \quad (2)$$

$$\mathit{Likes}(\mathsf{hilary}, \mathsf{bernie}) \quad (3)$$

$$\mathit{Likes}(\mathsf{bernie}, \mathsf{shawn}) \quad (4)$$

$$(\mathit{Likes}(\mathsf{hilary}, \mathsf{bernie}) \wedge \mathit{Likes}(\mathsf{bernie}, \mathsf{shawn})) \rightarrow \mathit{Likes}(\mathsf{hilary}, \mathsf{shawn}) \quad (5)$$

$$\neg(\neg \mathit{Republican}(\mathsf{shawn}) \wedge \neg \mathit{Democrat}(\mathsf{shawn})) \quad (6)$$

$$\neg \mathit{Democrat}(\mathsf{shawn}) \quad (7)$$

   We would like to show using resolution that a registered Democrat likes a registered Republican. To do this, answer the following questions:
   (a) Convert formulas (1) - (6) into conjunctive normal form (CNF) with the help of appendix 1. **[1p]**
   (b) Prove that $(\mathit{Democrat}(\mathsf{hilary}) \wedge \mathit{Republican}(\mathsf{shawn}) \wedge \mathit{Likes}(\mathsf{hilary}, \mathsf{shawn}))$ is a logical consequence of (1) - (6) using the resolution proof procedure. **[3p]**
      - Your answer should be structured using one or more resolution refutation trees (as used in the book or course slides).

4. Constraint satisfaction problems consist of a set of variables, a value domain for each variable and a set of constraints. A solution to a CS problem is a consistent set of bindings to the variables that satisfy the contraints. A standard backtracking search algorithm can be used to find solutions to CS problems. In the simplest case, the algorithm would choose variables to bind and values in the variable's domain to be bound to a variable in an arbitrary manner as the search tree is generated. This is inefficient and there are a number of strategies which can improve the search. Describe the following three strategies:

   (a) Minimum remaining value heuristic (MRV). **[1p]**

   (b) Degree heuristic. **[1p]**

   (c) Least constraining value heuristic. **[1p]**

   Constraint propagation is the general term for propagating constraints on one variable onto other variables. Describe or provide the following:

   (d) What is the Forward Checking technique? **[1p]**

   (e) What is arc consistency? **[1p]**

   (f) Provide a constraint graph that is arc consistent but globally inconsistent. **[1p]**

5. A* search is the most widely-known form of best-first search. The following questions pertain to A* search:

   (a) Explain what an *admissible* heuristic function is using the notation and descriptions in (c). **[1p]**

   (b) Suppose a robot is searching for a path from one location to another in a rectangular grid of locations in which there are arcs between adjacent pairs of locations and the arcs only go in north-south (south-north) and east-west (west-east) directions. Furthermore, assume that the robot can only travel on these arcs and that some of these arcs have obstructions which prevent passage across such arcs.

   Provide an admissible heuristic for this problem. Explain why it is an admissible heuristic and justify your answer explicitly. **[2p]**

   (c) Let $h(n)$ be the estimated cost of the cheapest path from a node n to the goal. Let $g(n)$ be the path cost from the start node n_0 to n. Let $f(n) = g(n) + h(n)$ be the estimated cost of the cheapest solution through n.

   Provide a sufficiently rigid proof that A* is optimal if $h(n)$ is admissible. You need only provide a proof for either tree-search (seminar slides) or graph-search (in course book). If possible, use a diagram to structure the contents of the proof to make it more readable. **[2p]**

6. The following questions pertain to Nonmonotonic Reasoning:

   (a) Let $\models$ denote the classical entailment relation of First-Order Logic. Let $|\!\approx$ denote the entailment relation of a nonmonotonic logic. Using $\models$ and $|\!\approx$ as a basis, explain one of the major distinguishing characteristics between classical and nonmonotonic logic. **[1p]**

   (b) Reasoning about action and change is one of the major topics of Knowledge Representation. Describe two of the following three problems, with specific examples represented using either Propositional or First-Order Logic: **[2p]**

      i. The Frame Problem;

      ii. The Ramification Problem;

      iii. The Qualification Problem.

7. Use the Bayesian network in Figure 1 together with the conditional probability tables below to answer the following questions. Appendix 2 may be helpful to use. If you do not have a hand-held calculator with you, make sure you set up the solution to the problems appropriately for partial credit.

   (a) Write the formula for the full joint probability distribution $P(A, B, C, D, E)$ in terms of (conditional) probabilities derived from the bayesian network below. **[1p]**

   (b) What is $P(a, \neg b, c, d, \neg e)$? **[1p]**

   (c) What is $P(b \mid a, d)$? **[2p]**

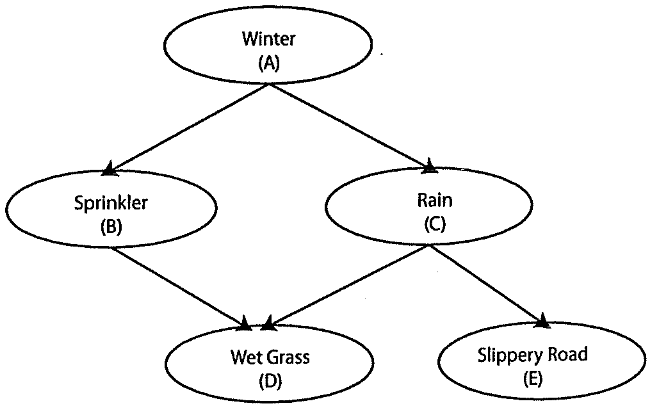

Figure 1: Bayesian Network Example

A	P(A)
T	.6
F	.4

A	B	$P(B \mid A)$
T	T	.2
T	F	.8
F	T	.75
F	F	.25

A	C	$P(C \mid A)$
T	T	.8
T	F	.2
F	T	.1
F	F	.9

B	C	D	$P(D \mid B, C)$
T	T	T	.95
T	T	F	.05
T	F	T	.9
T	F	F	.1
F	T	T	.8
F	T	F	.2
F	F	T	0
F	F	F	1

C	E	$P(E \mid C)$
T	T	.7
T	F	.3
F	T	0
F	F	1

8. These questions pertain to automated planning:

   (a) *Relaxation* is an important method for finding admissible heuristic functions for use in automated planning. How can this be achieved? Specifically:

      - Assume we are interested in solving some planning problem instance P, and that we already have access to a relaxed version P' of this problem instance.
        (Note that we are not interested in knowing *how* we can find such a P'. The question assumes that we already have both P, which could be a problem instance in the blocks world, the logistics domain or any other classical planning domain, and a relaxed version P', which could be generated through delete relaxation or any other relaxation technique.)
      - Assume we are applying some kind of state space planner to the original problem P and that this planner has generated a specific state s.
      - The planner now wants to compute an admissible heuristic estimate $h(s)$ using the relaxed problem P', using the *standard* method discussed during the planning lectures. What does the planner do, step by step, to calculate the numeric value of $h(s)$?

   Note again that we are talking about general definitions and methods, applicable to arbitrary problem instances and arbitrary relaxations. We are not interested in specific heuristic computation methods such as those that are only applicable to the 8-puzzle. **[2p]**

(b) Explain the main ideas underlying pattern database heuristics. Here we are not interested in the *database* aspect, which is simply a way of optimizing the computations involved. Instead, we are interested in the use of *patterns* to define the heuristic value $h(s)$ for a given state s in a planning problem P.

Hints: What is a pattern? How are patterns used to define subproblems? Given subproblems, how is $h(s)$ defined and computed? **[3p]**

Appendix 1

Converting arbitrary wffs to CNF form: (Propositional/grounded 1st-order formula case)

1. Eliminate implication signs using the equivalence: $\alpha \to \beta \equiv \neg\alpha \vee \beta$.
2. Reduce scopes of negation signs using De Morgan's Laws:
   - $\neg(\omega_1 \vee \omega_2) \equiv \neg\omega_1 \wedge \neg\omega_2$
   - $\neg(\omega_1 \wedge \omega_2) \equiv \neg\omega_1 \vee \neg\omega_2$
3. Remove double negations using the equivalence: $\neg\neg\alpha \equiv \alpha$.
4. Put the remaining formula into conjunctive normal form. Two useful rules are:
   - $\omega_1 \vee (\omega_2 \wedge \omega_3) \equiv (\omega_1 \vee \omega_2) \wedge (\omega_1 \vee \omega_3)$
   - $\omega_1 \wedge (\omega_2 \vee \omega_3) \equiv (\omega_1 \wedge \omega_2) \vee (\omega_1 \wedge \omega_3)$
5. Eliminate $\wedge$ symbols so only clauses remain.

Appendix 2

A generic entry in a joint probability distribution is the probability of a conjunction of particular assignments to each variable, such as $P(X_1 = x_1 \wedge \ldots \wedge X_n = x_n)$. The notation $P(x_1, \ldots, x_n)$ can be used as an abbreviation for this.

The chain rule states that any entry in the full joint distribution can be represented as a product of conditional probabilities:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid x_{i-1}, \ldots, x_1) \tag{8}$$

Given the independence assumptions implicit in a Bayesian network a more efficient representation of entries in the full joint distribution may be defined as

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid \mathit{parents}(X_i)), \tag{9}$$

where $\mathit{parents}(X_i)$ denotes the specific values of the variables in $\mathit{Parents}(X_i)$.

Recall the following definition of a conditional probability:

$$\mathbf{P}(X \mid Y) = \frac{\mathbf{P}(X \wedge Y)}{\mathbf{P}(Y)} \tag{10}$$

The following is a useful general inference procedure:

Let X be the query variable, let $\mathbf{E}$ be the set of evidence variables, let $\mathbf{e}$ be the observed values for them, let $\mathbf{Y}$ be the remaining unobserved variables and let α be the normalization constant:

$$\mathbf{P}(X \mid \mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y}) \tag{11}$$

where the summation is over all possible $\mathbf{y}$'s (i.e. all possible combinations of values of the unobserved variables $\mathbf{Y}$).

Equivalently, without the normalization constant:

$$\mathbf{P}(X \mid \mathbf{e}) = \frac{\mathbf{P}(X, \mathbf{e})}{\mathbf{P}(\mathbf{e})} = \frac{\sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})}{\sum_{\mathbf{x}} \sum_{\mathbf{y}} \mathbf{P}(\mathbf{x}, \mathbf{e}, \mathbf{y})} \tag{12}$$