Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2015-10-27
Sal (3)	G33 TER1 TER2
Tid	14-18
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	Mariusz Wzorek
Telefon under skrivtiden	0703887122
Besöker salen ca klockan	ja
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	

Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2015-10-27
Sal (3)	G33 TER1 TER2
Tid	14-18
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	Mariusz Wzorek
Telefon under skrivtiden	0703887122
Besöker salen ca klockan	ja
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	

Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2015-10-27
Sal (3)	G33 TER1 TER2
Tid	14-18
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	Mariusz Wzorek
Telefon under skrivtiden	0703887122
Besöker salen ca klockan	ja
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
27 october 2015 kl. 14-18

Points:

The exam consists of exercises worth 38 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 1-3.
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: Mariusz Wzorek, 0703887122. Mariusz will arrive for questions around 16.00.

1. Based on the material in Chapter 2 of the course book, define succinctly in your own words, the following terms:
   (a) Agent, Agent Function, Agent Program **[1p]**
   (b) Performance Measure, Rationality, Autonomy **[1p]**
   (c) Reflex Agent. Also provide a schematic diagram of such an agent. **[1p]**
   (d) Model-based Agent. Also provide a schematic diagram of such an agent. **[1p]**
   (e) Goal-based Agent. Also provide a schematic diagram of such an agent. **[1p]**

2. 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 does it consist of? **[2p]**
   (b) What does the Physical Symbol System Hypothesis state? **[1p]**
   (c) What does the Heuristic Search Hypothesis state? **[1p]**
   (d) Do you think the Physical Symbol System Hypothesis is true or false or somewhere in between? Provide reasonable justifications for your opinion. **[1p]**

3. Consider the following logical theory about elephants (where x and y are variables and clyde, oscar and sam are constants):

$$Pink(\mathsf{sam}) \tag{1}$$

$$Gray(\mathsf{clyde}) \tag{2}$$

$$Likes(\mathsf{clyde}, \mathsf{oscar}) \tag{3}$$

$$Likes(\mathsf{oscar}, \mathsf{sam}) \tag{4}$$

$$\neg(Pink(\mathsf{oscar}) \wedge Gray(\mathsf{oscar})) \tag{5}$$

$$\neg(\neg Pink(\mathsf{oscar}) \wedge \neg Gray(\mathsf{oscar})) \tag{6}$$

   We would like to show using resolution that a pink elephant likes a gray elephant. 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 $\exists x \exists y\,(Gray(x) \wedge Pink(y) \wedge Likes(x, y))$ is a logical consequence of (1) - (6) using the resolution proof procedure. **[3p]**
      - Your answer should be structured using a resolution refutation tree (as used in the book or course slides).
      - Since the unifications are trivial, it suffices to simply show the binding lists at each resolution step. Don't forget to substitute as you resolve each step.

4. Constraint satisfaction problems consist of a set of variables, a value domain for each variable and a set of constraints. Figure 1 depicts a constraint graph with variables A, B, C, D, E, each with a value domain $\{1, 2, 3, 4\}$. Binary constraints are associated with each arc and domain constraints are associated with each node. The graph is domain consistent but not arc-consistent. Appendix 3 provides the AC3 algorithm which makes a constraint graph arc-consistent.

   A solution to a CS problem is a consistent set of bindings to the variables that satisfy the constraints. A standard backtracking search algorithm can be used to find solutions to CS problems. Constraint propagation is the general term for propagating constraints on one variable onto other variables. Constraint propagation may be integrated with a backtracking algorithm or used for preprocessing.

   Describe the following:

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

   (b) What is arc consistency? Provide a definition. **[1p]**

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

   (d) Make the constraint graph in figure 1 arc consistent using the AC3 algorithm in Appendix 3. For the answer, only the resulting consistent bindings for each variable in the constraint graph output by the AC3 algorithm are required. **[2p]**

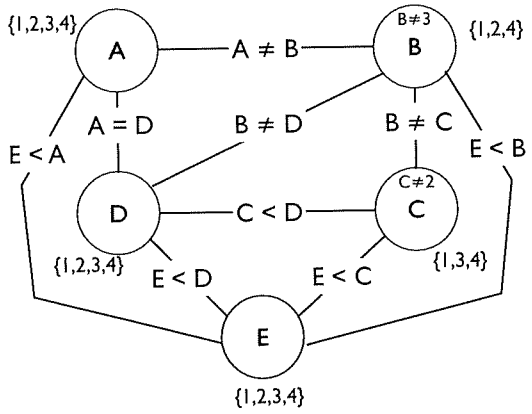

Figure 1: Constraint graph that is domain consistent but not arc consistent

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) Given the 8-puzzle problem:

   - let $h_1(n)$ be the Hamming distance of a board configuration. The Hamming distance computes the number of misplaced tiles in a board configuration where a misplaced tile is one that is not in the correct position.
   - Let $h_2(n)$ be the sum of Manhattan distances to the correct tile positions for each of the tiles in a board configuration.

   Explain whether each of $h_1(n)$ and $h_2(n)$ are admissible heuristics for the 8-puzzle problem and justify your answer explicitly. Which of $h_1(n)$ and $h_2(n)$ is the better heuristic? Justify your answer. **[3p]**

   (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 automated planning.

   (a) A partial order causal link (POCL) planner begins with an initial plan and proceeds by resolving *flaws* in this plan. For example, there can be *threats* that need to be resolved.

   What is a *threat*? Remember that there is a specific definition of threats for POCL planning, related to the structure of a plan that is under construction but has not yet been finished.

   How can we *resolve* a threat? That is, how does a POCL planner modify or extend the plan so that the threat no longer exists? **[2p]**

   (b) Recall that the optimal delete relaxation heuristic $h^+(s)$ applies delete relaxation to a planning problem and then solves the resulting relaxed problem optimally starting in state s, taking the cost of the resulting plan as the heuristic value. As this requires optimal planning, h^+ is too slow for use in planning, but we could construct other heuristics such as h_1 that build on h^+.

   What is the main idea behind h_1 that allows it to avoid "combinatorial explosions" and to be computed far more quickly than h^+?

   What is the difference between the two very similar heuristic functions h_1 and h_{add} (in terms of how the functions are defined and computed)? **[2p]**

   (c) What is *satisficing* planning? Does this type of planning typically require *admissble* heuristics? Why or why not? **[2p]**

7. The following questions pertain to machine learning. Give detailed answers to the questions below.

   (a) As formally as possible, define *supervised learning*. **[1p]**

   (b) Explain the purpose of each of the terms $Q()$, $R()$, α and γ in the Q-learning update below. **[2p]**

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

   (c) Explain in more detail *the curse of dimensionality* in the context of reinforcement learning. **[1p]**

8. Use the Bayesian network in Figure 2 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 $P(a, \neg b, c, \neg d, e)$? **[1p]**

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

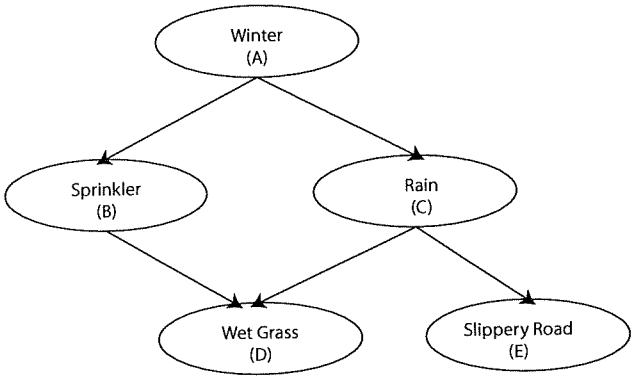

Figure 2: Bayesian Network Example

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

Appendix 1

Converting arbitrary wffs to CNF form

1. Eliminate implication signs.
2. Reduce scopes of negation signs.
3. Standardize variables within the scopes of quantifiers (Each quantifier should have its own unique variable).
4. Eliminate existential quantifiers. This may involve introduction of Skolem constants or functions.
5. Convert to prenex form by moving all remaining quantifiers to the front of the formula.
6. Put the matrix into conjunctive normal form. Two useful rules are:
   - $\omega_1 \vee (\omega_2 \wedge \omega_3) \equiv (\omega_1 \vee \omega_2) \wedge (\omega_1 \vee \omega_3)$
   - $\omega_1 \wedge (\omega_2 \vee \omega_3) \equiv (\omega_1 \wedge \omega_2) \vee (\omega_1 \wedge \omega_3)$
7. Eliminate universal quantifiers.
8. Eliminate $\wedge$ symbols.
9. Rename variables so that no variable symbol appears in more than one clause.

Skolemization

Two specific examples. One can of course generalize the technique.
$\exists x P(x)$:
Skolemized: $P(\mathsf{c})$ where c is a fresh constant name.
$\forall x_1, \ldots, x_k, \exists y P(y)$:
Skolemized: $P(\mathsf{f}(x_1, \ldots, x_k))$, where f is a fresh function name.

Useful Equivalences

$\neg\forall x\, \omega(x) \equiv \exists x\, \neg\omega(x)$
$\neg\exists x\, \omega(x) \equiv \forall x\, \neg\omega(x)$

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:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid x_{i-1}, \ldots, x_1) \tag{7}$$

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

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

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:

$$\mathbf{P}(X \mid Y) = \frac{\mathbf{P}(X \wedge Y)}{\mathbf{P}(Y)} \tag{9}$$

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:

$$\mathbf{P}(X \mid \mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y}) \tag{10}$$

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:

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

Appendix 3: Arc-Consistency algorithm

```
function AC-3( csp) returns false if an inconsistency is found and true otherwise
  inputs: csp, a binary CSP with components (X, D, C)
  local variables: queue, a queue of arcs, initially all the arcs in csp

  while queue is not empty do
    (X_i, X_j) ← REMOVE-FIRST(queue)
    if REVISE(csp, X_i, X_j) then
      if size of D_i = 0 then return false
      for each X_k in X_i.NEIGHBORS - {X_j} do
        add (X_k, X_i) to queue
  return true
```

```
function REVISE( csp, X_i, X_j) returns true iff we revise the domain of X_i
  revised ← false
  for each x in D_i do
    if no value y in D_j allows (x,y) to satisfy the constraint between X_i and X_j then
      delete x from D_i
      revised ← true
  return revised
```

Figure 3: AC3 Arc Consistency Algorithm