Försättsblad till skriftlig tentamen vid Linköpings Universitet

Datum för tentamen	2014-10-28
Sal (3)	TER1 <u>TER2</u> TERE
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	16.00
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Anna Grabska Eklund, ankn. 2362, anna.grabska.eklund@liu.se
Tillåtna hjälpmedel	Miniräknare/ Hand calculators.
Övrigt	
Antal exemplar i påsen	

Linköpings Universitet
Institutionen för Datavetenskap
Patrick Doherty

Tentamen
TDDC17 Artificial Intelligence
28 october 2014 kl. 14-18

Points:

The exam consists of exercises worth 36 points.
To pass the exam you need 18 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. Alan Turing proposed the Turing Test as an operational definition of intelligence.
   (a) Describe the Turing Test using your own diagram and explanations.**[2p]**
   (b) Do you believe this is an adequate test for machine intelligence? Justify your answer.**[1p]**

3. Consider the following logical theory (where x and y are variables and a, b and $27, 28$ are constants):

$$\forall x \forall y\ [(Package(x) \land Package(y) \land Inroom(x, 27) \land Inroom(y, 28)) \Rightarrow Smaller(x, y)] \tag{1}$$

$$Package(\mathsf{a}) \tag{2}$$

$$Package(\mathsf{b}) \tag{3}$$

$$Inroom(\mathsf{a}, 27) \lor Inroom(\mathsf{a}, 28) \tag{4}$$

$$Inroom(\mathsf{b}, 27) \tag{5}$$

$$\neg Smaller(\mathsf{b}, \mathsf{a}) \tag{6}$$

   We would like to show using resolution that package a is in room 27. To do this, answer the following questions:
   (a) Convert formulas (1) - (6) into conjunctive normal form (CNF) with the help of appendix 1. **[2p]**
   (b) Prove that $Inroom(\mathsf{a}, 27)$ is a logical consequence of (1) - (6) using the resolution proof procedure. **[2p]**
   - 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.

   (c) Convert the following formula into a formula in conjunctive normal form (CNF). **[1p]**

$$\forall x\ [\neg P(x) \lor [\forall y\ [\neg P(y) \lor P(\mathsf{f}(x, y))] \land \exists w\ [Q(x, w) \land \neg P(w)]]]$$

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 are required. **[2p]**

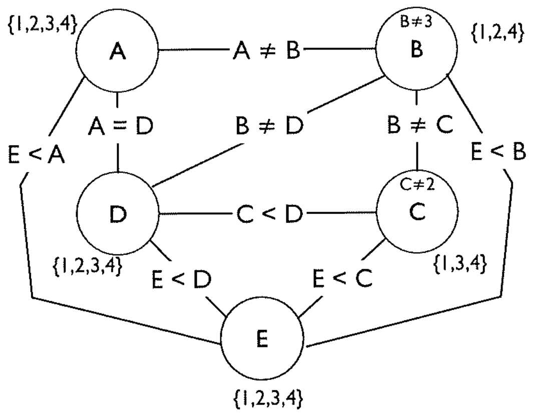

Figure 1: Constraint graph that is domain consistent but not arc consistent

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) 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 automated planning.

   (a) What is *satisficing* planning? **[1p]**

   (b) Informally, a *relaxation* to a planning problem "removes constraints" from the problem. The lectures (and lecture notes) provided a more formal definition, allowing us to show more directly that optimal solutions to a relaxed problem can be used to define admissible heuristics. Complete the following definition:

      Given two classical planning problems P and P', we know that P' is a relaxation of P iff ... **[1p]**

   (c) A partial order causal link (POCL) planner begins with an initial plan and proceeds by resolving *flaws* in this plan. One typically distinguishes between two specific types of flaw.

      Which flaw types exist?

      How is each flaw type *recognized* in a plan? That is, what characterizes a flaw of each type?

      How can each flaw type be resolved? **[3p]**

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

   (a) Explain the difference between *supervised learning* and *reinforcement learning*. **[1p]**

   (b) What is the intuition behind the *information gain* selection strategy in decision tree learning (i.e. what does it do)? **[1p]**

   (c) What is the formula below used for? Explain what happens if you decrease γ? **[2p]**

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

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]**

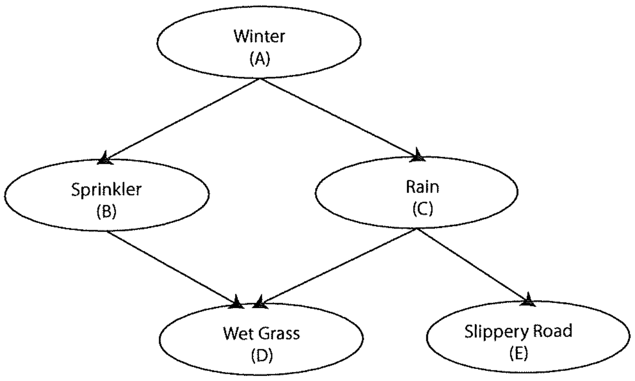

Figure 2: Bayesian Network Example

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.

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
    $(X_i, X_j) \leftarrow$ 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 $X_i$
  $revised \leftarrow false$
  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 $D_i$
      $revised \leftarrow true$
  return $revised$
```

Figure 3: AC3 Arc Consistency Algorithm