Introduction to Artificial Intelligence Exercises (WS 25/26)

Assignment 5: Probabilistic Reasoning and Making Decisions

Prof. Dr. Mathias Niepert, mathias.niepert@ki.uni-stuttgart.de
Aneesh Barthakur, aneesh.barthakur@simtech.uni-stuttgart.de
Machine Learning for Simulation Science, KI, Universität Stuttgart

Submit your solution in ILIAS as PDF. Submit either a (LaTeX) typeset PDF or legibly done handwritten solution. Please DO NOT forget to write your full names and matriculation numbers of each group member. Please name your *.pdf* submission files using the following format containing your
Groupnumber_Lastname1_Lastname2_Assignmentnumber.pdf
e.g. G1_Manolache_Lukashina_Tong_Assignment1.pdf
If there are any questions, please do not hesitate to ask them in the exercise forum on ILIAS.

1 Probabilistic Reasoning — 50 Points

1.1 From Logical to Probabilistic Reasoning — 20 Points

Probability theory provides a formalism that extends propositional logic in that it assigns probabilities to propositions. For example, the expression $P(b|a) \geq P(b)$ indicates that knowing a being true makes b become more probable, which is the probabilistic analogue of the logical implication $a \Rightarrow b$. The aim of this exercise is to use the rules of probability theory and Bayes' theorem to convince ourselves that probabilistic reasoning is indeed a generalization of propositional logic.

a. Consider three binary random variables A, B, and C corresponding to the propositions "*Frost occurs*", "*The road is slippery*" and "*An accident happens*". Each of these variables have domain $\{True, False\}$. As in the lecture, we use the statements a and $A = True$ equivalently. Provide English-language interpretations of the following probabilistic statements.

   a) $P(b|\neg a) \leq P(b)$
   b) $P(c|b) \geq P(c)$
   c) $P(a|b) \geq P(a)$
   d) $P(\neg a|\neg b) \geq P(\neg a)$ — 4 Points

b. Using probability theory, the special case of *logical implication* ($a \Rightarrow b$) can be expressed as $P(b|a) = 1$. Assume that $P(b|a) = 1$. Show that

   a) $P(b|\neg a) \leq P(b)$

b) $P(a|b) \geq P(a)$

c) $P(\neg a|\neg b) = 1$ 6 Points

c. The Bayesian Network in Figure 1 represents the joint probability distribution over the binary random variables A, B, and C. Recall that a, b, and c are shorthand for $A = true, B = true$, and $C = true$. We are given that $P(a) = 0.6$, $P(b|a) = 0.7$, $P(b|\neg a) = 0.5$, $P(c|b) = 0.6$ and $P(c|\neg b) = 0.3$. Compute the following values using marginalization

a) $P(b)$

b) $P(c)$

c) $P(a|c)$ 10 Points

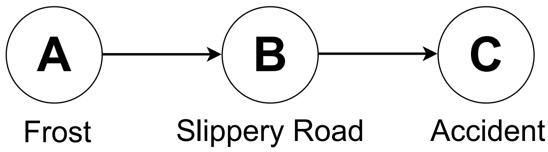

Figure 1: An exemplary Bayesian Network

1.2 Semantics of Bayesian Networks 30 Points

Commuting to and from university can often be delayed, either due to unexpected weather conditions or due to the maintenance work on public transport. Consider the Bayesian Network shown in Figure 2 for such scenarios, where the nodes indicate binary random variables.

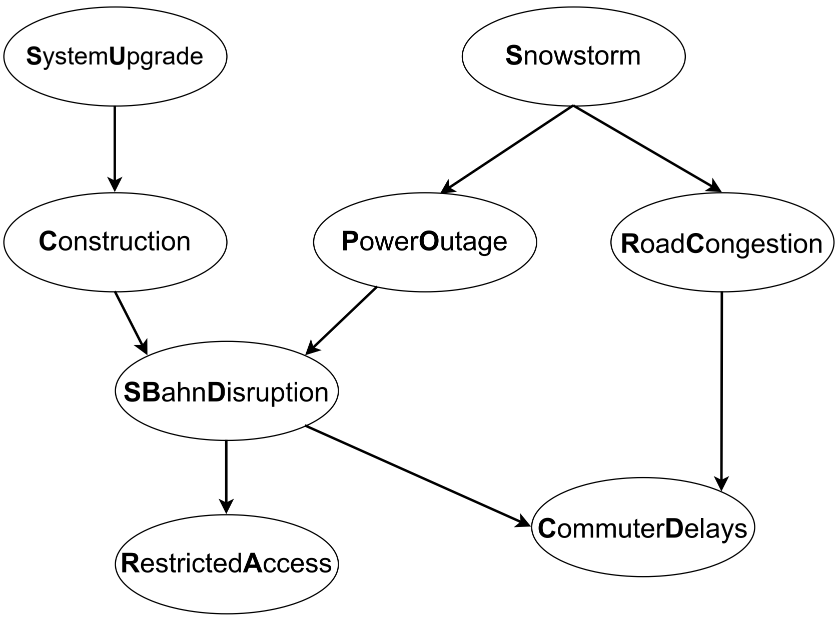

Figure 2: A Bayesian Network for Commute Delays

a. List all nodes in the Markov Blanket of the following nodes -
   a) $PowerOutage$
   b) $SBahnDisruption$

15 Points

b. Consider the conditional probabilities for the Bayesian Network (Table 1). Compute the value of $P(CommuterDelays|snowstorm, \neg construction)$, that is $P(CD|S = True, C = False)$ 15 Points

$P(PO\|S)$		$Snowstorm$ True	False
$PowerOutage$	True	0.3	0.02

$P(RC\|S)$		$Snowstorm$ True	False
$RoadCongestion$	True	0.1	0.01

$P(SBD = True\|C, PO)$		$Construction$ True	False
$PowerOutage$	True	0.9	0.8
	False	0.2	0.01

$P(CD = True\|SBD, RC)$		$SBahnDisruption$ True	False
$RoadCongestion$	True	0.9	0.8
	False	0.7	0.005

Table 1: Conditional Probabilities for Bayesian Network on Commute Delays (Fig 2)

2 Making Decisions 50 Points

2.1 Utility Theory (1) 10 Points

Say $A \succeq C \succeq B$. And the utilities of the outcome are:

$U(A) = 455$

$U(B) = -150$

$U(C) = 50$

What is the lottery over A and B that will make us indifferent between the lottery and C?

2.2 Utility Theory (2) 10 Points

Suppose that for a utility function U over three outcomes A, B, and C:

$U(A) = 5$

$U(B) = 20$

$U(C) = 0$

We are provided a choice between a lottery that give us a 50% probability of B and a 50% probability of C and a lottery that guarantees A. Compute the preferred lottery.

2.3 Markov Decision Processes (MDPs)

2.3.1 Computing discount factor γ **15 Points**

Suppose we have a Markov Decision Process (MDP) consisting of five states, $s_1, ..., s_5$, and two actions,

$$\begin{aligned} a_S &\rightarrow \quad stay \\ a_C &\rightarrow \quad continue \end{aligned}$$

And we have the following transition model and reward function:

$$\begin{aligned}
T(s_i|s_i, a_S) &= 1 & & for\ i \in \{1, 2, 3, 4\} \\
T(s_{i+1}|s_i, a_C) &= 1 & & for\ i \in \{1, 2, 3, 4\} \\
T(s_5|s_5, a) &= 1 & & \forall\ actions\ a \\
R(s_i, a) &= 0 & & for\ i \in \{1, 2, 3, 5\}\ and\ \forall\ actions\ a \\
R(S_4, a_S) &= 0 & & \\
R(S_4, a_C) &= 10 & &
\end{aligned}$$

What is the discount factor γ if the optimal value $U^*(s_1) = 1$?
Note: s_5 is the terminal state.
Hint: $U^*(.)$ is the optimal value obtained if we follow the optimal policy π^*.

2.3.2 Markov Decision Process - Value Iteration **15 Points**

The goal in this task is to apply the *value iteration* algorithm to a simple grid world that is shown below.

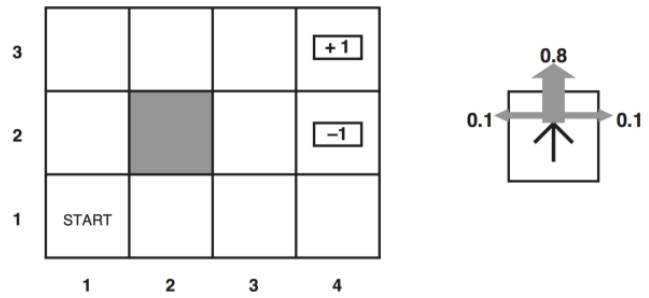

Figure 3: A 4x3 grid world with a 'living penalty' $R = \mathbf{-0.2}$.

In *value iteration*, we set all initial values as 0, except for the terminal states (3,4) and (2,4) which have fixed utilities of +1 and -1 respectively. Apply the *value iteration* algorithm to the above grid world for **2 iterations** and compute the utility (i.e., the value) of the cell (3,3). Please note that you also have to compute the value of all the cells along the trajectory leading up to the said state. You can assume the discounting factor γ to be **0.9**. Remember that the **Bellman update** is defined as

$$U_{i+1}(s) = \max_{a \in A(s)} \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma U_i(s')]$$

where $U_i(s)$ is the utility value for state s at the i^{th} iteration.

For more details on the *value iteration* algorithm, we encourage you to refer section **16.2.1 Value Iteration** in the book (4^{th} Global Edition).