Lunds Tekniska Högskola
Institutionen för datavetenskap

EDAP01 (a.k.a. TFRP20)
VT 2022

Artificiell Intelligens
Artificial Intelligence
Tentamen 2022–03–16, 14.00–19.00, Sparta A+B

You can give your answers in English or Swedish.
You are welcome to use a combination of figures and text in your answers.

1 Probabilistic reasoning, BNs (EAT) 9p

You have a set of 5 random variables. You know the following about them:

- Semantically speaking, the phenomenon represented by B is known to cause an effect on what is represented by C, which can then have an effect on what is represented by D and E respectively.
- $P(D|A,B,C,E) = P(D|B)$ and $P(E|A,B,C,D) = P(E|B)$
- $P(A|B,C,D,E) = P(A)$
- $P(D,E) \neq P(D)P(E)$

Answer the following questions (motivate your answers!):

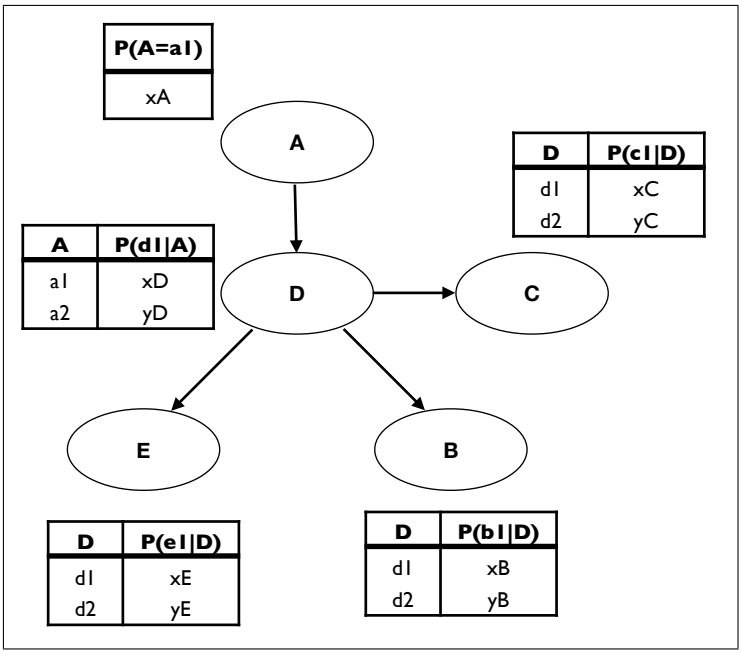
a) When are two random variables independent of each other? When are they conditionally independent? 2 points

b) Which of the networks i), ii), and iii) is / are correct wrt the set of variables described above? 2 points

c) Which network is optimal (if any), and why? 2 points

d) What do the CPTs represent? Explain explicitly the one for variable C in network iii)! 2 points

e) Network iv) represents a Naïve Bayesian Classifier. Why is it called naïve? 1 point



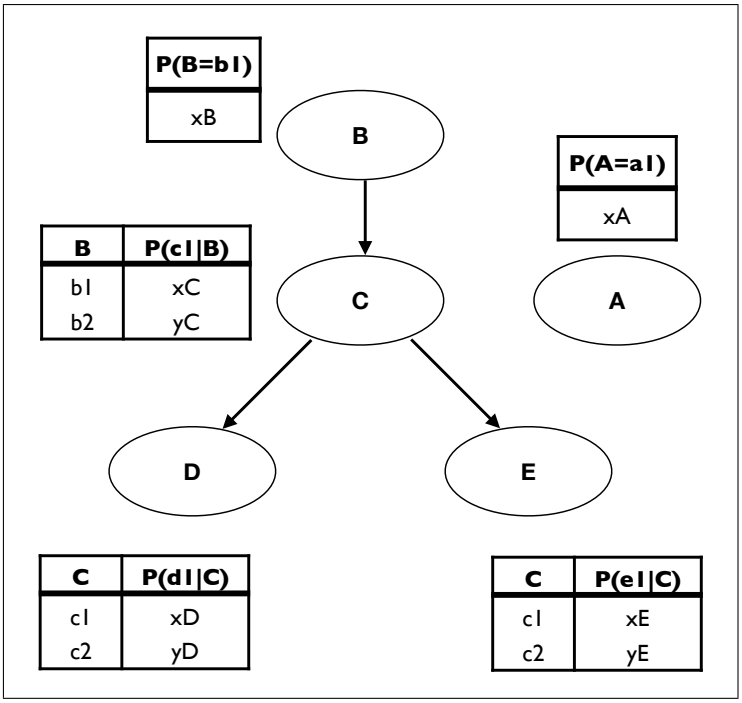
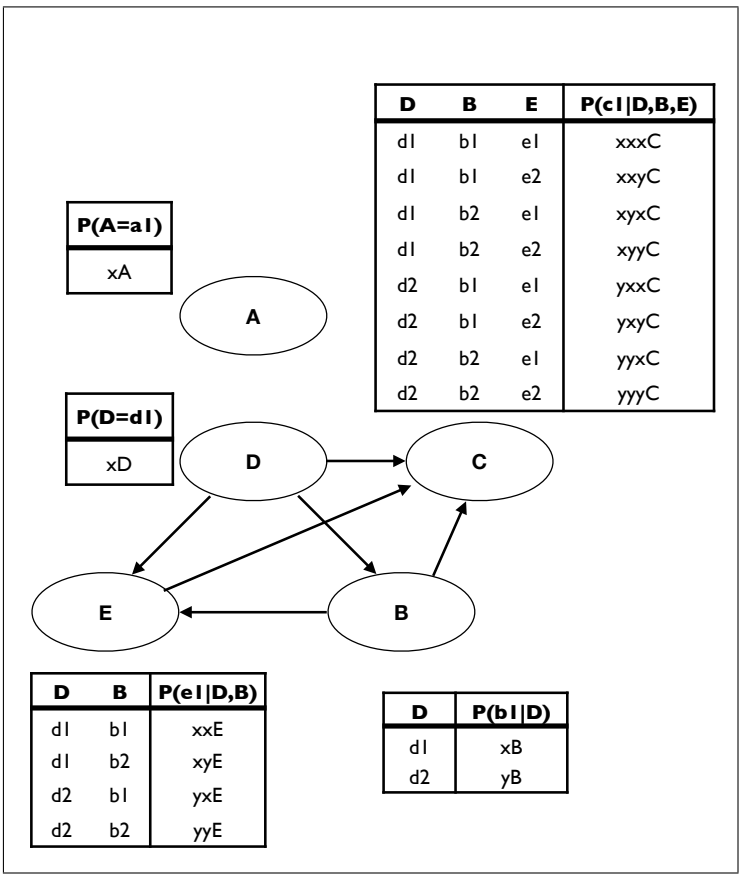


Figure 1: Networks i) (top), ii) (bottom)



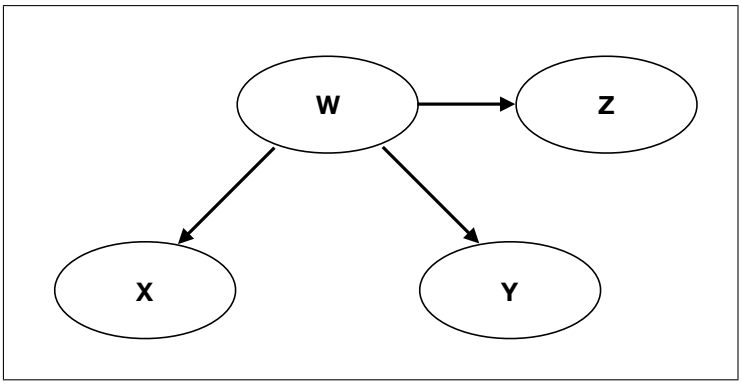


Figure 2: Networks iii) (top) and iv) (bottom

2 Probabilistic (Bayesian) reasoning (over time) / Robotics (EAT) 21p

Similar to what you did before as a homework assignment, you are supposed to localise an agent in a grid world. In this case, the world has obstacles in it, and the states correspond to the number of possible positions for the agent (see below). The agent moves according to the following assumptions: Stay put with probability 0.4, and move to one directly adjacent state (no diagonal moves) with probability 0.6 overall. It reports in every step how many spots it could move to around it (this means, it reports "1", "2", or "3") with probability 0.8, or it tries to trick you and does not report anything with probability 0.2.

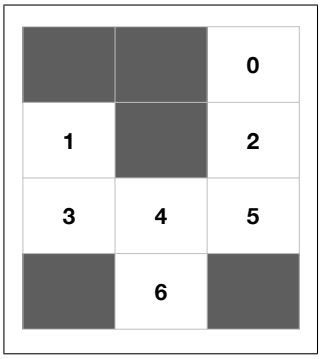


Figure 3: The gridworld for the agent to move in; the numbers in the cells correspond to the state

a) Formulate the problem of localising the agent as an HMM in matrix-vector form, i.e. note down the matrices describing the transition and sensor models. 6 points

b) Assume you start out with not knowing where the agent actually is, but then you get a sensor report of "3". Do you know for sure now, where it is? Why? Explain both intuitively and mathematically! 2 points

c) Assume now to receive the following series of sensor reports AFTER the initial report of "3" discussed above: "2", "nothing", "1". What is/are the *possible* state(s) the agent can be in? Explain intuitively! 2 points

d) Using your models, determine and explain the *most likely* state for the agent after the report series above ("2", "nothing", "1"). 8 points

e) What is the difference between a *static* and a *stationary* world/process model? 1 point

f) Why is an HMM not a good representation for a general localisation problem in robotics? 2 points

3 Machine Learning (PN) 30 p

During the course, we have seen how to apply a logistic regression and find optimal parameters to classify French and English texts. In this examination, we will extend logistic regression and use a neural network with one hidden layer.

This part is essentially a mathematical development and no programming is necessary.

3.1 Dataset

We will consider a dataset DS of q observations, some corresponding to French texts, some to English ones. As in the assignment, each observation $\mathbf{x}_i$ (data point) is defined by a vector of three parameters, where the two last ones are the total number of characters and the number of As:

$$\mathbf{x}_i = (x_{i,1}, x_{i,2}, x_{i,3}),$$

where $x_{i,1} = 1$.

Each observation i belongs to one of two classes: either 0 (English) or 1 (French). The class of $\mathbf{x}_i$ is denoted y_i. The whole dataset is then defined by:

$$DS = \{(x_{i,1}, x_{i,2}, x_{i,3}, y_i) | i : 1..q\},$$

where $x_{i,1} = 1$.

You will suppose that this dataset is available in the form of a matrix, $\mathbf{X}$, for the observation parameters (the features), and a vector, $\mathbf{y}$, for the classes.

3.2 Logistic Regression

You will now suppose you have an optimal weight vector from an optimization procedure (gradient descent), $\mathbf{w}$, to carry out the classification. In the next questions, you will describe its properties and how to use it:

1. What is the number of parameters of $\mathbf{w}$, i.e. the dimension of the vector? 1 point
2. Expand the dot product $\mathbf{w} \cdot \mathbf{x}_i$ with $\mathbf{w}$'s parameters and $\mathbf{x}_i$'s parameters $(x_{i,1}, x_{i,2}, x_{i,2})$. You will number $\mathbf{w}$ parameters from one, $w_1, ...$ 1 point
3. Write the logistic function that will predict the class of a $\mathbf{x}_i$ vector. 2 points
4. Give the threshold that will define the class (French or English). 1 point

To help you, Fig. 4 from the course shows a sketch of logistic regression.

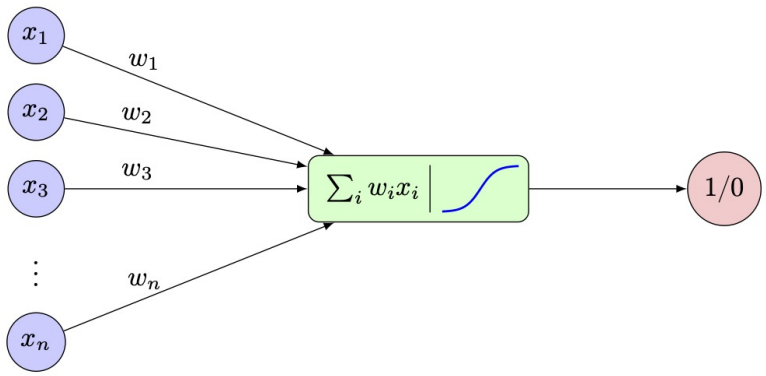


Figure 4: Logistic regression

3.3 Neural Network: The Forward Pass

3.3.1 First Step

In this part, you will use the network in Fig. 5 with one hidden layer:

- The **hidden layer**, see Fig. 5, has no activation: The operations in the nodes are just dot products.
- The **output layer** has a logistic activation: The operation is a dot product followed by a logistic function as in logistic regression.

Answer the questions:

1. Using the same operations as in the previous question, write the value in node $h_1^{(1)}$. This is just a dot product that you will expand as a sum. You will use the weights $w_{1,1}^{(1)}$, $w_{1,2}^{(1)}$, and $w_{1,3}^{(1)}$.
   $h_1^{(1)} = ...$ 1 point
2. Similarly, write the value in node $h_2^{(1)}$. You will use the weights $w_{2,1}^{(1)}$, $w_{2,2}^{(1)}$, and $w_{2,3}^{(1)}$.
   $h_2^{(1)} = ...$ 1 point
3. Write this result compactly as a matrix product below $\mathbf{h}^{(1)} = \mathbf{W}^{(1)}\mathbf{x}$, where you will replace the underscores with the weight parameters.

$$\begin{bmatrix} h_1^{(1)} \\ h_2^{(1)} \end{bmatrix} = \begin{bmatrix} - & - & - \\ - & - & - \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

2 points

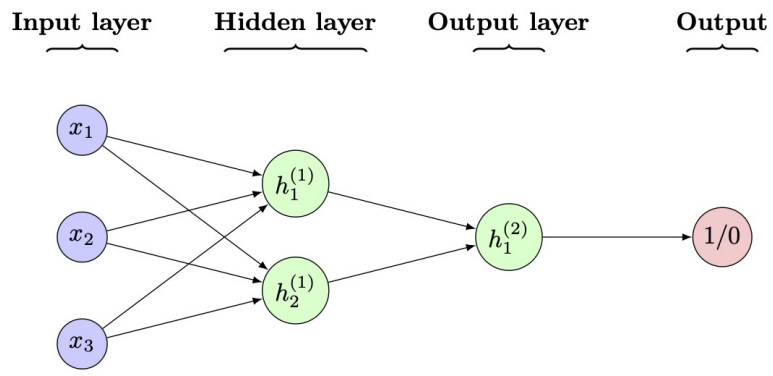


Figure 5: A simple neural network

3.3.2 Second Step

You will now compute the value in $h_1^{(2)}$ in Fig. 5. This time, you will first compute a dot product bewteen the hidden layer output and the weights, and then pass the resulting value to a logistic function as in Fig. 4

1. Using $h_1^{(1)}$ and $h_2^{(1)}$ as inputs, compute the incoming value in the output layer. It is just a dot product, where you will use the weights $w_{1,1}^{(2)}$ and $w_{1,2}^{(2)}$. You will denote this value $z_1^{(2)}$:
   $z_1^{(2)} = ...$; 2 points

2. Write this result compactly as a matrix product below, $\mathbf{z}^{(2)} = \mathbf{W}^{(2)}\mathbf{h}^{(1)}$, where you will replace the underscores with the weight parameters.

$$\begin{bmatrix} z_1^{(2)} \end{bmatrix} = \begin{bmatrix} \_ & \_ \end{bmatrix} \begin{bmatrix} h_1^{(1)} \\ h_2^{(1)} \end{bmatrix}.$$

   1 point

3. Compute the final value in $h_1^{(2)}$. It is just the application of the logistic function to $z_1^{(2)}$:
   $h_1^{(2)} = ...$ 1 point

3.3.3 Full Network

Now that you have completed the two steps, you will compute the output from the inputs. You will set aside the final logistic function.

1. Expand the two steps so that $z_1^{(2)}$ is a function of the input $(x_{i,1}, x_{i,2}, x_{i,2})$. 2 points

2. Write this as a matrix product, $\mathbf{z}^{(2)} = \mathbf{W}^{(2)}\mathbf{W}^{(1)}\mathbf{x}$, where you will replace the underscores with the appropriate weights.

$$\begin{bmatrix} z_1^{(2)} \end{bmatrix} = \begin{bmatrix} \_ & \_ \end{bmatrix} \begin{bmatrix} \_ & \_ & \_ \\ \_ & \_ & \_ \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

2 points

3.4 Weight Optimization for Logistic Regression

Reminder on logistic regression and on the logistic loss you used for the assignment. You will probably not need all what is below, but it contains all you should need for the rest of the examination.

- The loss in logistic regression is defined by

$$L(y, \hat{y}) = -y \ln \hat{y} - (1 - y) \ln(1 - \hat{y}),$$

  where $\hat{y} = \frac{1}{1+e^{-\mathbf{w}.\mathbf{x}}}$.

- The derivative of the loss is:

$$\frac{\mathrm{d}L(y, \hat{y})}{\mathrm{d}\hat{y}} = \frac{y - \hat{y}}{\hat{y}(1 - \hat{y})}.$$

- The derivative of the logistic function is $\frac{1}{1+e^{-x}}$ is $\frac{e^{-x}}{(1+e^{-x})^2}$. It is sometimes useful to factor it as

$$(1 - \frac{1}{1 + e^{-x}}) \cdot \frac{1}{1 + e^{-x}}.$$

- The gradient of the loss is defined by the partial derivatives:

$$\begin{aligned} \frac{\partial L(y, \hat{y})}{\partial w_i} &= \frac{\mathrm{d}L(y, \hat{y})}{\mathrm{d}\hat{y}} \frac{\partial \hat{y}}{\partial w_i}, \\ &= x_i(y - \hat{y}), \\ &= x_i(y - \frac{1}{1 + e^{-\mathbf{w}.\mathbf{x}}}). \end{aligned}$$

- The weight updates of the $\mathbf{w}$ parameter of logistic regression with a gradient descent[1] at step k of the iteration are given by:

$$\mathbf{w}_{(k+1)} = \mathbf{w}_{(k)} - \alpha_{(k)} \nabla_{\mathbf{w}^{(k)}} Loss(\mathbf{w}_{(k)}).$$

  where α is the learning rate and ∇ is the gradient notation.

[1] or ascent if you take the negated value of the loss

3.5 Weight Optimization of the Neural Network

In the logistic regression assignment, you applied a gradient descent to find the optimal weights.

1. Would it be possible to do the same with the network in Fig. 5? Justify your answer. 2 points

2. In a large network, would it be practical to compute the gradient with all the weight parameters? Justify your answer. 1 point

3.6 Backpropagation

In practice, the weight optimization uses the backpropagation algorithm. You will proceed in two steps to recreate it:

- First compute the gradient with respect to the inputs and then
- Apply the chain rule to compute it with respect to the weights.

3.6.1 Gradient with Respect to the Inputs

The loss of the network in Fig. 5 is the same as the logistic loss presented in the course. You will denote the loss gradient with respect to the last node $\nabla_{h_1^{(2)}} L(y, \hat{y})$. Do not try to compute it as it is given in Sect. 3.4. Just use this notation. Reusing the results from Sects. 3.3.1 and 3.3.2:

1. Compute $\nabla_{z_1^{(2)}} h_1^{(2)}$, the gradient of $h_1^{(2)}$ with respect to $z_1^{(2)}$. This is simply the derivative of the logistic function. 1 point

2. Compute $\nabla_{z_1^{(2)}} L(y, \hat{y})$. To help you start, use the chain rule and $\nabla_{h_1^{(2)}} L(y, \hat{y})$:

$$\begin{aligned} \frac{\partial L(y, \hat{y})}{\partial z_1^{(2)}} &= \frac{\partial L(y, \hat{y})}{\partial h_1^{(2)}} \frac{\partial h_1^{(2))}}{\partial z_1^{(2)}} \\ &= \nabla_{h_1^{(2)}} L(y, \hat{y}) \frac{\partial h_1^{(2))}}{\partial z_1^{(2)}} \end{aligned}$$

   and give the value of $\frac{\partial h_1^{(2))}}{\partial z_1^{(2)}}$. As answer, you just need to add one term to the equality below:

   $\nabla_{z_1^{(2)}} L(y, \hat{y}) = \nabla_{h_1^{(2)}} L(y, \hat{y})...$ 1 point

3. Compute $\nabla_{\mathbf{h}^{(1)}} z_1^{(2)}$ that is $(\frac{\partial z_1^{(2)}}{\partial h_1^{(1)}}, \frac{\partial z_1^{(2)}}{\partial h_2^{(1)}})$. This is a simple differentiation. 1 point

4. Compute $\nabla_{\mathbf{h}^{(1)}} L(y, \hat{y})$ that is $(\frac{\partial L(y,\hat{y})}{\partial h_1^{(1)}}, \frac{\partial L(y,\hat{y})}{\partial h_2^{(1)}})$. You just need to apply the chain rule. As answer, fill in the underscores

$$\nabla_{\mathbf{h}^{(1)}} L(y, \hat{y}) = \nabla_{h_1^{(2)}} L(y, \hat{y}) \cdot \_ \begin{bmatrix} - \\ - \end{bmatrix}.$$

2 points

The three items below are not part of the examination, just an explanation on how to finish the backpropagation. To reach $\mathbf{x}$, you would need to compute the terms:

- $\nabla_{\mathbf{x}} h_1^{(1)}$ that is $(\frac{\partial h_1^{(1)}}{\partial x_1}, \frac{\partial h_1^{(1)}}{\partial x_2}, \frac{\partial h_1^{(1)}}{\partial x_3})$.
- $\nabla_{\mathbf{x}} h_2^{(1)}$ that is $(\frac{\partial h_2^{(1)}}{\partial x_1}, \frac{\partial h_2^{(1)}}{\partial x_2}, \frac{\partial h_2^{(1)}}{\partial x_3})$.
- Once you have all these parts, you can compute $\nabla_{\mathbf{x}} L(y, \hat{y})$ that is $(\frac{\partial L(y,\hat{y})}{\partial x_1}, \frac{\partial L(y,\hat{y})}{\partial x_2}, \frac{\partial L(y,\hat{y})}{\partial x_3})$.

3.6.2 Gradient with Respect to the Weights

You will finally compute $\nabla_{\mathbf{w}^{(2)}} L(y, \hat{y})$, where $\mathbf{w}^{(2)} = (w_{1,1}^{(2)}, w_{1,2}^{(2)})$.

1. Using the chain rule, compute $\frac{\partial L(y,\hat{y})}{\partial w_{1,1}^{(2)}}$. This corresponds to: $\frac{\partial L(y,\hat{y})}{\partial z_1^{(2)}} \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(2)}}$, where you will replace the terms by the values you obtained before. 1 point
2. Using the chain rule, compute $\frac{\partial L(y,\hat{y})}{\partial w_{1,2}^{(2)}}$. This corresponds to: $\frac{\partial L(y,\hat{y})}{\partial z_1^{(2)}} \frac{\partial z_1^{(2)}}{\partial w_{1,2}^{(2)}}$, where you will replace the terms by the values you obtained before. 1 point
3. Now that you have the $\nabla_{\mathbf{w}^{(2)}} L(y, \hat{y})$ can you apply the update rule to $\mathbf{w}^{(2)}$. Justify your answer. 2 points
4. How would you proceed to update the weights from the previous layer. Justify your answer. 1 point

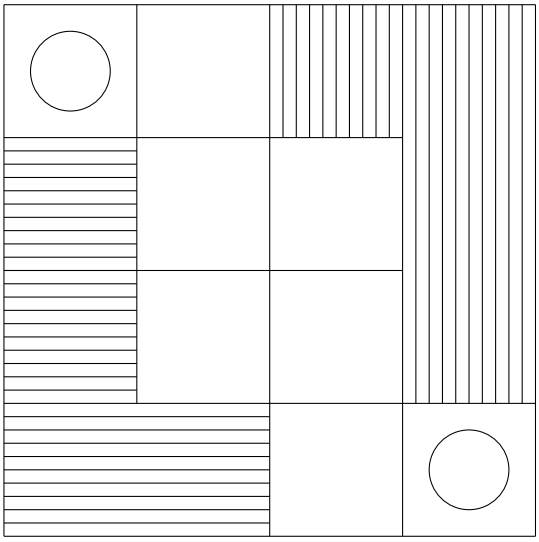
4 Games (JM) 15p

L-game is played by two players on a 4×4 board. Each player has an L-shaped figure which can be turned upside down and/or rotated in all directions. A move consists of two parts:

1. The player lifts her L-figure and puts it down on the board in a different position than before.

2. If she wants, she can also move one of the two neutral pieces[2] to a new position.

The player who cannot move her L to a new position, looses the game. The following picture illustrates the board in the beginning of the game.



Task 1 Assume now that you want to write a program that could play the L-game. Your task consists of representing the problem as an *adversarial search* problem. Describe how state and operators (possible moves) could be represented, how the goal state will be recognized (a *goal-test* function), how possible moves will be generated (a *successor* function), how one of the possible moves will be chosen (a *choose-move* function), etc. 10 points

In order to avoid misunderstanding, some precision is necessary in your answer. Therefore it would be a benefit if you could use e.g. list structures (or whatever you deem appropriate) to define the necessary data types you choose to represent state and operators. You can write your functions using a pseudocode.

Remember that your program is going to play against an opponent. Therefore during the search for the next best move you should take into account the possible moves of the opponent.

Task 2 What is the branching factor of the search space? Is the search space finite? 2 points

How would your answers to these two question change if the neutral pieces were not moved by any of the players, but got random placement after each player's move of her L-piece? I.e., the game would go as follows: (1) first player moves her L-piece, (2) the two neutral pieces get random positions on

[2]A neutral piece, marked by a circle on top of it, covers just one square of the board.

the available space (including their current placement), (3) the other player moves her L-piece, (4) the two neutral pieces get random positions on the available space, (5) do (1) again. 2 points

Task 3 Can you come up with a winning strategy for the player moving first? 1 point

5 Logic (JM) 20p

Imagine the following situation in the wumpus world:

SMELL 1,4	2,4	3,4	4,4
1,3	SMELL 2,3	3,3	4,3
SMELL 1,2	AGENT 2,2	BREEZE 3,2	4,2
1,1	BREEZE 2,1	3,1	BREEZE 4,1

Prove that the position 3,3 is safe, i.e., the agent will not get killed if it moves there (via position 3,2 or position 2,3; other paths may be unsafe, as you know). You need to

- formulate your problem in logic, 4 points
- state all *necessary* laws of the Wumpus world (do not do more than necessary, you don't have time), 6 points
- and finally prove that position 3,3 is safe. Maximal number of points (10) will be given for a resolution proof. 10 points

6 KR (JM) 5 p

Illustrate the concept of *non-monotonic reasoning* using the Wumpus world from the previous question as an example domain. Give a concrete example rather than just describe the concept. 5 points

Good Luck!