Study Manual

Probabilistic Reasoning

2023 – 2024

Silja Renooij

July 2023

General information

This study manual was designed to help guide your self studies. As such, it does not include material that is not already present in the mandatory literature and course slides or discussed in class. Hopefully, it will help you take a deeper dive into the subjects and enhance your understanding.

This manual enumerates some details on the knowledge and abilities students enrolling in this course are expected to learn. In addition, this manual lists for each chapter from the *syllabus "Probabilistic Reasoning with Bayesian networks"* a number of questions that can help to guide your self-study. The answer to each question is in the associated chapter, but trying to reproduce it by formulating your own answer will reveal what you did and did not grasp from the material covered in the chapter, and indicates to what extent you need to go through the chapter again. This type of question will **not** be part of the written exam.

Exercises to practice with the material, which are representative of exam questions (mostly former exam questions), can be found in the syllabus, along with answers and/or hints (for exercises indicated with a *). Additional examples of previous exams are available through the course web-site `https://www.cs.uu.nl/docs/vakken/prob/practicum.html`.

Examination

The INFOPROB course is a 7.5 ECTS course, which means that an average student is expected to spend 210 hours of work to complete and pass the course. In order to pass the course, you are expected to have knowledge of, insight in and, most importantly, understanding of the subjects presented in the mandatory literature and (sometimes more detailed) in the course slides. This also means that you have practised with applying different procedures and algorithms and can correctly do so in little time. Moreover, you are expected to be able to use your knowledge and understanding to solve problems and apply contexts and new concepts that have not been discussed in the course.

The course is graded based on a number of practical assignments and a written exam. A detailed grading scheme is available on `https://www.cs.uu.nl/docs/vakken/prob/beoordeling.html`. **In general we can say for both the exam and the assignments that convincingly demonstrating your understanding is what is most important: explanations will always be worth far more points than just an outcome or simple answer.**

The practical assignments serve two purposes: 1) to enhance your understanding of the treated material, and 2) as an early acquaintance with more specialized topics treated later on in the course. *The probabilistic programming assignment (Assignment C) serves the former purpose and the associ-*

ated learning goals are examined only through the assignment.

The final exam covers the subjects from the syllabus and the associated course slides. In general, we expect at least the following for the final exam:

- you know all preliminaries by heart, as well as any formula that can be easily reconstructed from understanding the idea behind it;
- you understand the origin of the components of all compound and message parameters from Pearl's algorithm (Syllabus, Chapter 4), but you need not memorise the exact way they are computed (the lemmas);
- you understand the entropy formula (Syllabus, Chapter 5) and its effects, but you are not required to reproduce it;
- you know what different symbols and notations mean, how different concepts are defined, and you are able to apply the different methods and algorithms discussed; (NB make sure you practice with Pearl's algorithm; this will save you at lot of time during the exam!!)
- you have enough understanding of all subjects treated to be able to identify advantages and drawbacks of different concepts and algorithms, and to use these insights in generating or understanding variations on what was treated during the course;
- ...

More specifically:

- Syllabus, Chapter 2 + course slides: you know all definitions, propositions and theorems by heart;
- Syllabus, Chapter 3 + course slides: you know all definitions, theorem, corollaries and lemmas by heart, except those from Section 3.2.1 (Undirected graphs[1]); you can apply d-separation;
- Syllabus, Chapter 4 + course slides:
  - you know by heart: all definitions, corollaries, proposition 4.1.3 ('chain rule' for Bayesian networks), lemmas 4.2.1 (basics of data fusion), 4.2.8 (useful property), all algorithms (test your inference answers with one of the Bayesian network software packages (see links on the course website));
  - you will be provided during the exam (see example sheet on course website): lemmas 4.2.5, 4.2.7, 4.2.13, 4.2.14, for computing compound parameters and message parameters in a singly connected graph (recall that the formulas for trees are just a special case of these).

[1] The lectures and the examination focus on directed graphical models. However, the concept of separation in undirected graphs is easier than the concept of d-separation in directed graphs, so you may want to study undirected graphs first. Both course syllabus and slides (with accompanying videos) provide the necessary information and exercises for that.

  - you can apply Pearl's algorithm with and without loop cutset conditioning, and the Suermondt & Cooper heuristic.

- Syllabus, Chapter 5 + course slides:
  - you know by heart: all definitions (except 5.2.2), all lemmas, corollaries and algorithms, all formulas for the (leaky) noisy-or gate; you can apply the formulas of the (leaky) noisy-or gate;
  - if necessary, you will be given during the exam: definition 5.2.2 (quality measure; from this you should be able to derive definition 5.2.3 for node quality);
  - you can compute the MDL measure and apply the B search heuristic.

- Syllabus, Chapter 6 + course slides: you know all concepts and formulas in 6.1 and 6.2 by heart; you have read 6.3 and 6.4 and have global understanding of the presented ideas; you have read 6.4.

- You can apply the analytic approach to establishing sensitivity functions, can compute the sensitivity value for one-way sensitivity functions, and can determine sensitivity sets;

An important note on applying procedures and algorithms: there is a difference between solving a problem and applying an algorithm that solves the problem. If an exercise or exam asks you to solve a problem, you can use whatever method you think is most suitable to do the job. If, however, you are required to solve a problem using a certain algorithm, then you are basically asked to illustrate how the algorithm can be used to solve the problem and you should execute the steps prescribed by the algorithm yourself, and write these down in such a way that we are convinced of your understanding. For example, if you are asked to compute a certain probability from a set of given probability, you may use whatever rules from probability theory you can apply. If you are asked to compute a probability using Pearl's algorithm (see Chapter 4), then you should perform exactly those computations prescribed by the algorithm, even if the example is such that there are more efficient ways of solving the problem by hand! Basically, if you are asked to apply an algorithm, we are testing whether or not you understand how the algorithm works, not if you are able to solve the problem for which the algorithm is designed.

Selfstudy Questions

The following sections list various questions for each chapter in the course syllabus. These questions are meant to guide you selfstudies and allow you to monitor your progress.

Chapter 1

The first chapter gives some historical background about the use of probability theory and other uncertainty formalisms for reasons of decision support. It briefly motivates the emergence of Bayesian networks and the reason why Bayesian network applications historically have often concerned the medical domain.

Questions

- What is mutually exclusive?
- What is collectively exhaustive?
- Which assumption is made in a naive Bayes approach for reasons of *time* complexity, what is the reduction achieved and why?
- Which assumption is made in a naive Bayes approach for reasons of *space* complexity, what is the reduction achieved and why?
- Why is a naive (or idiot) Bayes approach naive? Is it still used?
- What is the relation between a naive Bayes approach and GLADYS?

Chapter 2

The second chapter refreshes the necessary concepts from graph- and probability theory that play a central role in the Bayesian network framework. These concepts have already been discussed in other courses and can be found in any textbook on graph theory and probability theory. The chapter also introduces the notation that is used throughout the syllabus and which may differ from what you encountered previously.

Two important things to note here are the following:

- In this course we often address paths in graphs; unless stated otherwise, these paths are assumed to be *simple* paths.
- A lot of formulas in the course material contain a summation of the following form:
  $$\sum_{c_{\mathbf{V}}} f(c_{\mathbf{V}})$$
  for some expression $f(c_{\mathbf{V}})$ depending on $c_{\mathbf{V}}$. Note that this summation is a summation over the set of *configurations* $c_{\mathbf{V}}$ of a set $\mathbf{V}$, not over the elements of set $\mathbf{V}$ itself. If this set $\mathbf{V}$ is empty, this therefore does not mean the summation is undefined! Rather, it means the summation is over the single element $c_{\mathbf{V}} = c_{\emptyset} = \mathsf{T}$ (true). In that case the summation reduces to the single term $f(\mathsf{T})$.

Questions

- Is there a limit to the number of times a certain vertex may be included in a path?
- What is the difference between a walk, a path, and a simple path?
- What is the difference between a path and a chain in a directed graph?
- What is the difference between a loop and a cycle in a directed graph?
- What is the difference between a tree, a singly connected digraph and a multiply connected digraph?
- When can we use the operators $\cap$ and $\cup$?
- When can we use the operators $\wedge$ and $\vee$?
- What is the difference between the proposition True and a value *true* for a variable?
- What is the difference between a *joint* probability distribution, a *conditional* probability distribution, and a *marginal* probability distribution?
- What is the difference between a *value assignment*, a *configuration* and a *configuration template*?
- Why is it the case that for two sets of variables X and Y $C_{X \cup Y} = C_X \wedge C_Y$ is the only correct way of expressing the configuration template over both sets X and Y in terms of the separate configuration templates?
- What is the relation between the *marginalisation* and the *conditioning* property?
- What is the difference between independence of *propositions* and independence of *variables*?
- Suppose a (not necessarily binary) random variable V can take on the values $v_1, \ldots, v_n$. Why does the definition of probability distribution imply that $\sum_{i=1}^{n} \Pr(v_i) = 1$?
- What is the difference between (in)dependence and conditional (in)dependence?

Chapter 3

This chapter formalises two types of independence relation. The first, $I_{\Pr}$, is the type of relation that can be captured by a probability distribution. These independence relations form a proper subset of a more general type of independence relation I that abstracts away from probability distributions. The chapter also discusses different representations of independence relations, most notably (in)directed graphs. We will mostly focus on the use of directed graphs. However, some concepts are easier for undirected graphs, so you may want to study these prior to moving on to directed graphs. An important notion introduced in this chapter is the concept of d-separation.

Independence relations and their representation are still an area of ongoing research, see for example the work by Milan Studený from Prague. This chapter, however, describes the full body of research as far as required for our discussion of Bayesian networks in the subsequent chapters.

Questions

- Intuitively, what is the difference between independence relations I_{Pr} and I?
- What is the difference between the properties satisfied by I_{Pr} compared to those satisfied by I?
- What are possible encodings of an independence relation?
- Why does an I-map represent independencies and a D-map dependencies?
- Why is it unfortunate that not every independence relation has an (un)directed P-map?
- Why is an independence relation for which a P-map exists termed *graph-isomorphic*?
- Does a graph-isomorphic independence relation have a *unique* P-map?
- Does the (d-)separation criterion require a graph to be acyclic? Why/why not?
- Why does the following statement hold for directed (and undirected) graphs: If a set Z blocks a path, then $Z \cup Y$ blocks the path for any Y?
- Why does the following statement for directed graphs only hold if $Z \neq \emptyset$ and at least one $Z_i \in Z$ is on the chain under consideration: If a set Z blocks a chain, then $Z \cup Y$ blocks the chain for any Y?
- If you studied undirected graphs: why does the following statement hold: If a set Z separates two other sets, then $Z \cup Y$ separates the sets for any Y?)
- Why does the following statement *not* hold for directed graphs: If a set Z d-separates two other sets, then $Z \cup Y$ d-separates the sets for any Y?
- Directed I-maps can be transformed into undirected I-maps. If you studied both: how can we do this transformation? Is minimality inherited?

Chapter 4

This chapter defines the concept of Bayesian network and reconstructs the algorithm for exact probabilistic inference as designed by Judea Pearl; for ease of exposition only binary variables are considered. Other, and often more efficient, algorithms for exact inference exist such as *jointree propagation* (S.L. Lauritzen, D.J. Spiegelhalter, F.V. Jensen, P.P Shenoy, G. Shafer) and *variable elimination* (R. Dechter, G.F. Cooper) methods. The reason for discussing Pearl's algorithm is that it explicitly exploits the graphical structure of the network without transforming it, and therefore seems the one easiest to explain and comprehend.

Current research focuses on finding still more efficient algorithms, both for exact and approximate inference, and also for dealing with networks that include continuous variables. Also, now and again, researchers look into better algorithms for loop cutset conditioning (A. Becker, D. Geiger), the extension to Pearl's algorithm that is required to do inference in multiply connected graphs.

To gain more insight in inference and the effects of loop cutset conditioning try one of the free software packages available through the course website.

Questions

The network as graphical model:

- Why is a Bayesian network often called a *belief network*, or *causal network*?
- What is the independence relation I modelled by a Bayesian network?
- Why have people chosen the graph of a Bayesian network to represent a (minimal) I-map of the independence relation as opposed to a D-map?
- Why have people chosen a directed graph to capture the independence relation for a Bayesian network?

The probability distribution:

- Why is the digraph of a Bayesian network assumed to be acyclic?
- What is the difference between an assessment function for a vertex and a probability distribution for that vertex?
- Which piece(s) of information is/are necessary to determine the number of assessments functions required for specifying a Bayesian network?
- Which pieces of information are necessary to determine the number of (conditional) probabilities required for specifying all assessments functions?
- How does the joint probability distribution Pr *defined* by a Bayesian network exploit the fact that the digraph of the network is an I-map of the independence relation of Pr?

- How do constraints for space requirements for Bayesian networks compare to the naive Bayes approach?

Probabilistic Inference – general:

- How do constraints for time requirements for Bayesian networks compare to the naive Bayes approach?
- What is the difference between a prior and a posterior probability?
- Is there actually a difference between $\Pr(v_i \mid v_j)$ and $\Pr^{v_j}(v_i)$?
- What is the difference between $\Pr(v_i)$ and $\Pr^{v_i}(v_i)$?
- How can a set of instantiated or observed vertices be related to the *blocking set* introduced in Chapter 3?
- What is the relation between *configuration*, *instantiation*, and *partial configuration*?
- What is the computational time complexity of probabilistic inference?

Pearl's algorithm – definitions:

- Why does the definition of V_i^- differ across different types of graph?
- How do the messages of Pearl's algorithm exploit the fact that the digraph is an I-map of the independence relation of Pr?
- Why do the formulas for computing messages differ across different types of graph?
- Which lemma represents the first step of Pearl's algorithm?
- In Pearl's algorithm: what is the difference between a compound parameter and a message parameter?
- What is the difference between a compound causal parameter and a compound diagnostic parameter?
- What is the difference between a causal message parameter and a diagnostic message parameter?
- What are the differences and similarities between causal/diagnostic parameters and probabilities?
- Why do the *compound* causal parameters for a vertex sum to 1? Why do the causal parameters *sent* by a vertex sum to 1?
- Why do the compound diagnostic parameters for a vertex in general *not* sum to 1? Why do the diagnostic parameters *sent* by a vertex in general *not* sum to 1?

- Why is it the case that if the compound diagnostic parameter of a vertex equals 1, that all diagnostic message parameters sent by that vertex are equivalent (in a directed tree: equal to 1)?
- When is a vertex ready to send a causal message parameter?
- When is a vertex ready to send a diagnostic message parameter?
- When is a vertex ready to calculate its (posterior) probability?
- To calculate *prior* probabilities, do you require causal (compound/message) parameters, diagnostic (compound/message) parameters, or both?
- To calculate *posterior* probabilities, do you require causal (compound/message) parameters, diagnostic (compound/message) parameters, or both?
- Why do the formulas in Lemmas 4.2.5 and 4.2.7 and similar formulas for singly connected digraphs assume V_i to be an uninstantiated vertex?
- Why do observations have to be entered by means of *dummy nodes*?
- Why does the *dummy node* approach enable us to use lemmas 4.2.5, 4.2.7 etc. for instantiated vertices as well?
- Why can Pearl's algorithm lead to incorrect results when applied to multiply connected graphs?

Pearl's algorithm – normalisation:

- What is the use of a normalisation constant α?
- Why does the normalisation constant α differ across messages?
- Is normalisation always used for message parameters that sum to 1, or could they sum to a different constant? (Compare α in the formula for diagnostic message parameters in singly connected graphs to the other α's).
- Why can normalisation be postponed to the final step of Pearl's algorithm?
- When does it come in handy to not postpone normalisation?
- Why is normalisation not actually required when computing prior probabilities? (Can be used as correctness check)

Loop Cutsets:

- Why doesn't Pearl's algorithm as presented in 4.2.1 and 4.2.2 work in multiply connected graphs?
- What is the idea behind the method of loop cutset conditioning?

- Which of the following statements is correct (the difference is rather subtle):

  I Loop cutset conditioning is a method for probabilistic inference in multiply connected networks that uses Pearl's algorithm for computing some of the probabilities is requires.

  II Loop cutset conditioning is a method that is used in combination with Pearl's algorithm in order to enable its correct application to multiply connected networks.

  III Loop cutset conditioning is a method that is used inside Pearl's algorithm in order to enable its correct application to multiply connected networks.

- Consider a Bayesian network with loops in its digraph G; can the loop cutset for G be the empty set $\emptyset$?

- Given the definition of a loop cutset, can a vertex with two incoming arcs in G be included in a loop cutset for G?

- For a graph G, will the Suermondt & Cooper algorithm return a loop cutset that includes a vertex with two or more incoming arcs in G?

- Why does every loop in a Bayesian network's digraph have at least one vertex with at most one incoming arc?

- Is a loop cutset a property of a graph or a property of a Bayesian network?

- Does a loop cutset change after entering evidence in a Bayesian network?

- Does the probability distribution over configurations of the loop cutset change after entering evidence into a Bayesian network?

- Why can the prior probability distribution over configurations of the loop cutset *not* be computed using Pearl's algorithm, but has to be computed by marginalisation from the joint distribution instead?

- In loop cutset conditioning, in general, which probabilities can and which probabilities cannot be computed using Pearl's algorithm (as a "black box")?

- What is the use of a *global supervisor*?

- What is the first step in performing loop cutset conditioning (not: in finding a loop cutset!)?

- Why is loop cutset conditioning computationally expensive?

- Is finding a loop cutset computationally expensive?

- Is a minimal loop cutset always optimal? Is an optimal loop cutset always minimal?

- Why should vertices with a degree of one or less not be included in a loop cutset?

- Why are vertices with an *in*degree of at most one selected as candidates for the loop cutset by the heuristic Suermondt & Cooper algorithm? What are possible effects of this choice?
- Why does the heuristic Suermondt & Cooper algorithm for finding a loop cutset choose to add a candidate with highest degree to the loop cutset? What if there are several candidates with the highest degree?
- What are properties of the graph and the loop cutset before and after performing the algorithm for finding a loop cutset?

Chapter 5

This chapter and the next one differ from previous chapters in the sense that the latter describe more or less finished research, whereas the following chapters discuss topics that are subject of ongoing research. As a result, the chapters tell a number of short stories with lots of open endings. We provide the basics of various algorithms and methodologies that still underlie, and are necessary for understanding, state of the art results. The assumption that we consider binary variables only is now lifted, although employed in some specific situations.

This chapter discusses the construction of a Bayesian network: how do we get the graphical structure (by hand, or automatically?) and where do the probabilities come from (experts, data, ...?). Try and construct a (small) Bayesian network yourself using one of the numerous software packages or online tools developed for constructing and reasoning with Bayesian networks (see the course website)

Remark: in this chapter you will find formulas including a term of the form $a \cdot \log b$, where a and/or b can be zero. A standard convention is that $0 \log 0 = 0$.

Questions

Construction in general

- What are typical trade-offs made when constructing a Bayesian network for a real application domain?
- What is the difference between domain-variables and Bayesian network variables?
- What is the difference between single-valued variable, multi-valued variables and random variables?
- Given the definition of a Bayesian network in Chapter 4, why is it, in general, not possible to allow continuous random variables?

- What are advantages and disadvantages of the different ways of modelling a multi-valued domain variable in a Bayesian network?

Construction by hand

- What is the problem with using the notion of causality for constructing the digraph?
- Why is it important to distinguish between direct and indirect relations?
- Why do digraphs which are constructed by hand often contain cycles at some stage of the construction?
- Why does the assumption of a disjunctive interaction simplify probability assessment? Is the assumption realistic?
- Why is a noisy-or gate called noisy? Why is its leaky version called leaky?
- This chapter formulates the noisy-or gate for binary variables. Is it easy to generalise to non-binary variables?
- Why does the inhibitor probability in the leaky noisy-or gate differ semantically from the inhibitor probability in the noisy-or gate?
- Why is it difficult to assess probabilities from literature?
- Why is it difficult to use domain experts for assessing probabilities?
- What is a typical trade-off when eliciting probabilities from experts?
- What is the use of sensitivity analysis?

Automated construction

- Are the assumptions required of a database in order to use it for learning a network realistic?
- What are possible solutions to and the effects of handling *missing values* in a database?
- $N(c_X)$ gives the number of cases in a database for which variable X has the given configuration. Why can we correctly assume that $N(c_\varnothing) = N$?
- Is frequency counting an efficient way of probability estimation?
- What is the difference between a conditional independence learning algorithm and a metric learning algorithm?
- Does the order of variables used for conditional independence learning affect the resulting graph?
- Why are a quality measure and a search heuristic necessary ingredients for a metric learning algorithm?

- What is the function of the different ingredients of the MDL quality measure?
- Why does the MDL measure as stated assume variables to be binary? Can this restriction be lifted?
- What are the ranges of the different terms in the MDL measure?
- Why is the $\log P$ term a constant if you assume a uniform prior distribution over possible digraphs?
- You could say "Entropy $\equiv$ Chaos $\equiv$ Uncertainty". Why does lower entropy go hand in hand with denser digraphs?
- Why do denser digraphs go hand in hand with less reliable probability estimates? What solution to this problem is present in the MDL measure?
- What are the benefits and drawbacks of using a search *heuristic*?
- When employing the search heuristic as described in Chapter 5 it is sufficient to consider only *gain* in quality. Why?
- For the search heuristic described: why does a difference in *vertex quality* for a single vertex correspond to a difference in *graph quality*?
- When the search heuristic adds an arc (V_i, V_j), why does the quality of V_j change and not the quality of V_i?
- If adding a certain arc *decreases* the quality of the graph, will that arc *ever* be added?
- If adding a certain arc *increases* the quality of the graph, will that arc *always* be added?
- Does the choice of an arc to add affect the set of arcs that may be subsequently added?
- Does the procedure of adding arcs that give the largest increase in quality result in a graph with maximum quality?
- Is the MDL measure a suitable measure?
- When should you learn networks of restricted topology?
- What are possible effects of learning a network from a database that does not obey the required assumptions?

Chapter 6

This chapter tries to bridge the gap between construction and actual use of a Bayesian network application. It first discusses possible methods for analysing and evaluating the behaviour of your network; then some example applications where a Bayesian network is used as a component in a decision support process are discussed, followed by a brief note on explanation.

Questions

Sensitivity analysis:

- What is the use of sensitivity analysis?
- Compare the following terms: ’sensitivity’, ’robustness’, and ’correctness’.
- Is it computationally feasible to perform a sensitivity analysis?
- Why can’t a CPT-parameter have a non-linear influence on a prior probability of interest?
- Is the sensitivity set a property of a Bayesian network’s digraph?
- Why can variables in the sensitivity set be excluded from a sensitivity analysis?
- What do sensitivity functions look like in general?
- Why are sensitivity functions for a prior joint probability linear?
- What is the amount of data you get from a sensitivity analysis?
- How can you select CPT-parameters of interest?
- What are advantages and disadvantages of performing a two-way sensitivity analysis as opposed to a one-way analysis?

Evaluation:

- In evaluating a Bayesian network against a standard of validity, why would you ideally use a *gold standard*? Why is this not always possible?
- What information does and doesn’t a percentage correct convey?
- What is an acceptable percentage correct?
- What information does and doesn’t the Brier score convey?
- What is an acceptable Brier score?

Application:

- What is the use of a two-layer architecture?
- In the threshold model for patient management, are the utilities patient-specific? What about the threshold probabilities?
- Can the threshold model for patient management be applied to a series of tests?
- Why are utilities in the threshold model said to subjective and in diagnostic problem solving objective?

- In diagnostic problem solving utilities are defined for values of binary variables only. Can this be generalised to non-binary variables?
- In diagnostic problem solving utilities represent the shift in probability of an hypothesis as a result of evidence. Why are large shifts in *either* direction awarded large utilities?
- In diagnostic problem solving, is the utility of a variable's value a fixed number? What about the expected utility of a node?
- In selective evidence gathering, what is the use of a *stopping criterion*?
- In selective evidence gathering, what are the effects of the *single-disorder* and *myopic* assumptions?

Explanation:

- What can or should be explained in Bayesian networks?
- What should be explained to whom?