Sample Book

Author

May 31, 2022

Abstract

Here I write the abstract

Contents

Chapter 1

Introduction

Reasoning with evidence in law is cool!

The main research question for this Master's Thesis is: can we automatically create good Bayesian Networks that reflect the ground truth of simulations? If so, can we use this simulation + Bayesian Network setup to investigate BN idioms and methods for law more generally, to see how well the probabilistic approach holds up?

Chapter 2

State of the Art

2.1 The problem with evidence

When we find evidence for a hypothesis that we have held in the back of our minds, we then find the hypothesis more likely. This is the basic idea behind reasoning with evidence. In a constellation of hypotheses and pieces of evidence, we want to construct a network that will lead us to believe as many true hypotheses as possible, given the evidence that we have.

However, evidence itself is elusive, and it's connection to hypotheses is as well - how can we be sure that some evidence supports some hypothesis, and even if we know that it does, how can we express how strong the piece of evidence is? Some evidence is very weak, and only after a tedious process of ruling out other factors and careful investigation and collection of other pieces of evidence, we can come to a conclusion about a hypothesis. On the other hand, some evidence is so strong that it leaves no room for doubt.

We all have intuitions about evidence strength - but can we make these intuitions precise? Additionally, can we manage the complex realities of weak evidence for many different hypothesis? These are questions that have guided this state of the art section.

In this section I will briefly discuss methods for reasoning with evidence, then transition to probabilistic reasoning with evidence, Bayesian Networks. I will discuss some problems with Bayesian Networks.

2.2 Reasoning with evidence

Argumentation graphs and their semantics. Scenario theory. Bayes Law. Hybrid Approach.

2.3 Bayes!

Probabilistic reasoning (Dahlman, simple). Bayesian Networks Fenton and Vlek - combining scenarios, idioms in Bayesian Networks.

2.3.1 Bayesian Networks

DAG. Pearl. Random variables nodes, "causal" but actually conditional.

2.3.2 (Automatic) Bayesian Networks Builders

Standard toolsets. Pyagrum, agenarisk and hugin. We build Bayesian Networks by hand, perhaps guided by a scenario or by argumentation. However, the links that we put between nodes must be done by ourselves.

The standard way of building Bayesian Networks in law, is for now, based on human intuition. Usually, in other domains, we can build Bayesian Networks automatically from large datasets using algorithms such as K2. Automated Bayesian Network building is not plausible in the legal-evidence domain, because the data that we need is notoriously sparse - we have information about the number of crimes, from police departments, but for the subaspects for each scenario, it is hard to find frequencies (reference class problem, and others).

However, since we're using simulations to investigate the Bayesian Approach, we can generate a near infinite amount of information, and then we use this information to build the network, such as we might use data on health markers to predict kidney failure (medical domain - pretty sure this is the standard example). So automated building tools come in handy. In this project I only used the K2 algorithm, because you can add temporal information to it.

Explain K2 algorithm here.

Automated Bayesian Network building might prove useful in other, more data-and-information-rich legal subdomains - such as pure forensics (DNA evidence), or case-outcome prediction.

2.4 Problems with the Bayesian Approach

There are many problems with the Bayesian Approach.

The most obvious is the problem of the numbers: where do we get them? We can get some of them from statistics, but we need to many numbers that we have to make some numbers up. This is not necessarily a bad thing - we put our (betting) money where our mouth is and assign numerical precise probabilities to situations that we preciously only had vague intuitions for. However, this brings about the veil of objectivity. By giving a probability to your intuitions, you have made your intuitions more precise and you can now reason with them, update on evidence using Bayes Law, and everything's great. In some domains, this is obviously okay. If you want to bet cents on world events and walk that fine line between calibration and discrimination for fun and profit, that's no problem. After all, there are incentives to abstain from the unclear, the stuff that might not have a specified answer, the vague. But when we are talking about using evidence in law, we are talking about exactly that domain - we're not making predictions about the price of oil in 6 months, or the outcome of the French election, with clear outcomes, clear procedures for measurement. Instead, we're trying to make predictions about crimes and crime scenarios, which are a lot vaguer, and strangely unobservable at times - things like motives, or behaviours that happen under specific circumstances, interlocking stuff with complex dependencies. We all have intuitions about evidence strengths in vague situations, but they are more difficult to make precise than the traditional 'forecasting' events. So trying to assign probabilities without a clear method of calibration, makes that they will be imprecise.

Level of granularity.

Independence relations. Selecting events.

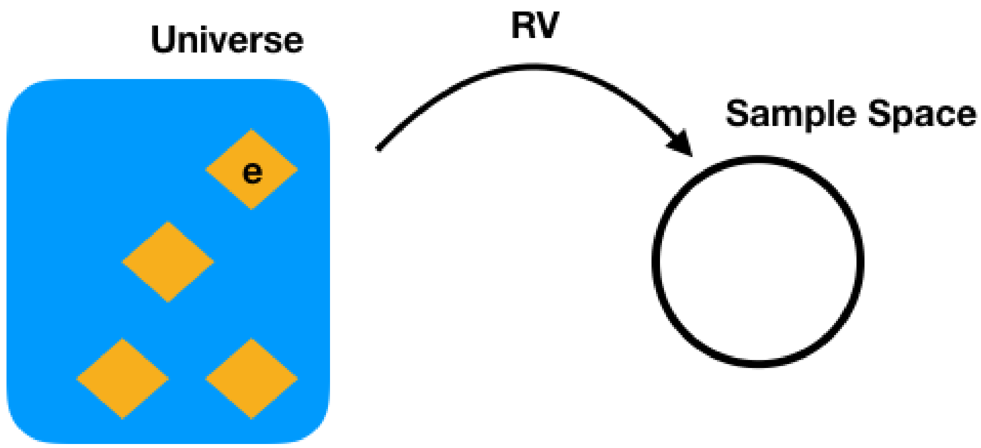
Technical problems with Bayesian Networks. Precision. Evaluation. Sensitivity Analysis.

1. The problem of the reference class, which is also a problem within a 'clear' domain - but this is a fundamental philosophical problem that I will not discuss further.
2. We do not know the probabilities (frequency) of our variables in the first place.
3. We do not know how robust the network is to imprecise or wrong frequencies.
4. We do not know if the assumption of independence between any two variables holds. The assumption of independence is necessary in Bayesian Networks, otherwise the complexity of the network becomes unmanageable.

2.5 Background on probabilities - the meaning of random variables

Bayesian Networks consists of nodes, and arcs between those nodes. The nodes in a Bayesian Network are random variables. But what are random variables?

Well, a random variable (RV) is not just a natural language statement, but it's a mathematical object. It is a function or mapping of the form $X : \Omega \to S$, where Ω is the universe, and S is the sample space. The RV maps some elementary event in the universe, to a specific output in sample space (Figure).



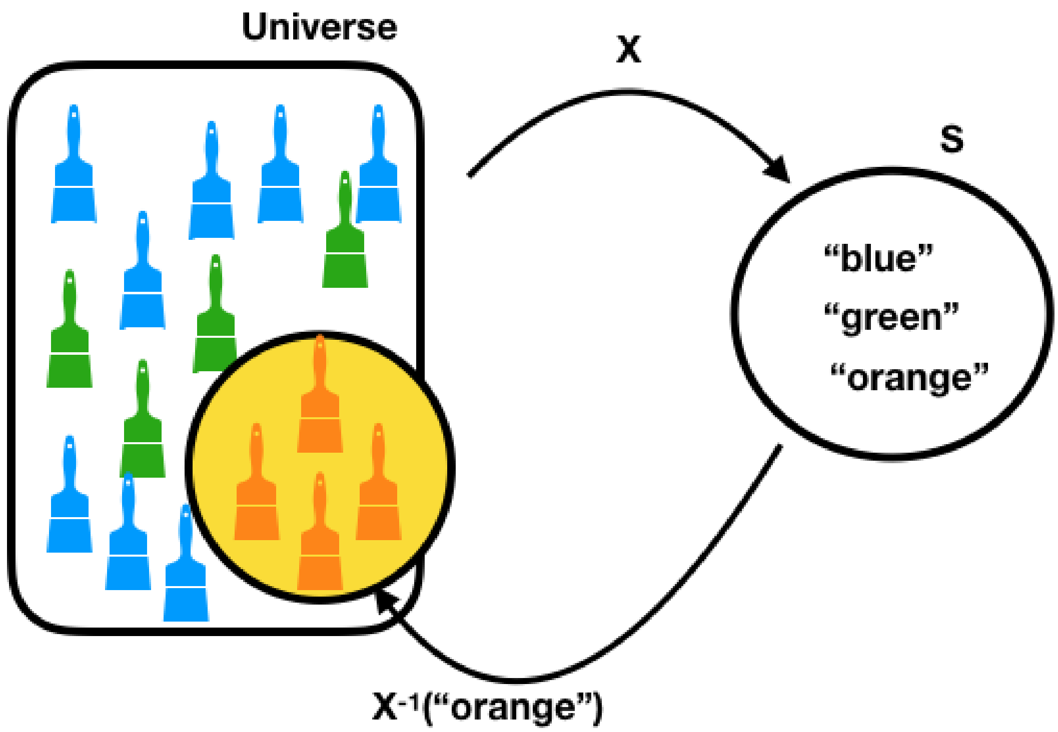
This is not really an explanation, because 1) what is a universe? 4) what is a sample space? 3) mapping how? 2) what is an elementary event? and so on.

The universe can be super abstract. It is the place where the events happen that we're interested in. It can be our actual universe, a simulation, or a subset or our actual universe or simulation. Elementary events are the events that happen within our universe - they are the elements of the set that is the universe. A random variable is a procedure, with a method, that can map an event to a certain value. All the possible values that an event can take, are collected in the sample space S. [1]

To illustrate this, we have a simple example (Example 1).

Example 1 *Let's say that we are observing a painter who wants to put down the first layer of paint on their canvas. The painter has to pick the color of the background layer.* Ω *is the universe, and contains all events where this painter is picking a color for a background for an oil painting.* $\mathcal{S}$ *is the sample space, and is the set of all colors that the painter can pick. The RV* X *maps events to values in sample space. For this initial simple example,* $S = \{green, blue, orange\}$*. Then* $X^{-1}(orange)$ *refers to the set of all events where the painter picked "orange" as the background color.*

[1] This is how far I'll go with this explanation, if you want to know about σ-fields I don't have time to understand and explain all of that. And I don't think its necessary for the problems with bayesian networks.



In our example, we say that the RV X maps events to values in sample space (and inversely, X^{-1} maps a value in sample space to a set of events). However, how does this mapping work? The mapping itself is not a mathematical object (eg, we don't do $X(\omega) = 2 * \omega^2 = "orange")$. Instead, we observe a certain event w in Ω, and measure in some way, to find out what value w takes in S. For things like dice-rolls, it is clear how we observe the event (we just look at it), just like in our painting example. However, there are several ways that we can observe and map this situation, operationalise it, which all respond to different RV's:

1. **RGB Color-picker**: we take a picture of the canvas, and analyse it in the computer. We have selected certain ranges of RGB values that correspond to orange, blue and green, respectively. The average color of the canvas within one of the RGB-color bins is how we know what color it is.
2. **Taste**: we have a super-taster who specialises in paint pigments [2]. When she blind-tastes a bit of the paint, she can tell us what color it is, because the pigment that causes the color orange tastes different from the green, which tastes different from blue.

[2] Not the healthiest of occupations.

3. **Subjective taste**: I have my own opinions on whether some paint color is orange, blue or green. I see the canvas, and I decide [3].

4. Think of something else ridiculous.

Of these operationalisations of the RV you can think many things. Some might be more valid than others. We can only know if we trust the final probability assignment (I will get there in a second) if we can agree with the method of mapping - which is the random variable. The operationalisation of the random variable is a big deal, and to know if it is valid/accurate, we need to know exactly how it was mapped, because then we can argue about it if we don't like it.

Now, let's assume we have some sort of way of determining the color of the canvas (really does not matter which one, as long as we pick one). Then, we can talk about probability! Probability maps an event to a real number $[0, 1]$. The 'event' here is not the same as an elementary event. Instead, 'event' means that a RV takes some value in the sample space - so an event is the set of elementary events in the universe, for which the random variable on that event takes a specific value - $\{\omega \in \Omega | X(\omega) \in orange\}$ is the set of cases where the color on the canvas is orange.

Then, we can assign a probability value to this event, simply $P(X \in A) = p$. There's a whole debate on what p should actually mean - is it a degree of belief, or a frequency, or something else? The frequentist view is that we repeat measurements - eg, we apply the RV a lot (every time the painter is starting a new painting), and count how often the canvas is orange, and then divide that by the total amount of new canvases started, and this should be the value of p. In the subjectivist view, p can be any value as long as that value reflects our belief on how many times the painter paints orange vs blue vs green backgrounds. Then, we have our probability!

A different problem is for our universe. Are we only counting oil paintings, or are we also counting acrylics. Or did we say 'painting' when we meant 'artwork' and should we also take the painter's sketches and pastel drawings into account? This is the second problem, and it is known as the problem of the reference class. Different reference classes will result in different frequencies (also in different degrees of belief). That's why it is not just important to specify operationalisations for every random variable, but also specify exactly what part of the universe you're investigating (eg, what is and isn't in Ω).

2.6 Agent simulations

Why are simulations a good tool for investigating problematic aspects of the application of Bayesian Networks in law? Primarily, simulations are useful because they are a completely

[3] I might be colorblind

determined environment. This means that we can know exactly with what frequencies events occur, generating lots of data for our K2 algorithm to build networks with.

But are agent simulations even used in crime? Is this purely an academic exercise? Actually yes, but maybe also no. Bosse and Gerritsen and friends.

Chapter 3

Creating Simulations to Evaluate Bayesian Networks.

3.1 Introduction

In this part, I explain the general method for creating simulations with automatic Bayesian Networks, as well as how to evaluate these Bayesian Networks. The general process of creating simulations to evaluate Bayesian Networks is illustrated in Figure ??. We start by defining a simulation with agents, and then have reporters report on that simulation in the 'Experiment' stage. Relevant events in simulations are collected by the reporters in each run. The collection of runs is used by an automatic Bayesian Network constructor algorithm to construct a Bayesian Network automatically in the 'Building BN' stage. Finally, the constructed Bayesian Network is evaluated with respect to the criteria in the 'Evaluate BN' stage. These three stages are explained in this section. Specific instances of simulations and networks I created with this process are the subject of the next chapters.

3.2 Setting-up an Experiment

An experiment consists of a simulation and reporters. Reporters are defined separately from the simulation because they are not inherent to it - they are defined with respect to what we want to know about the simulation.

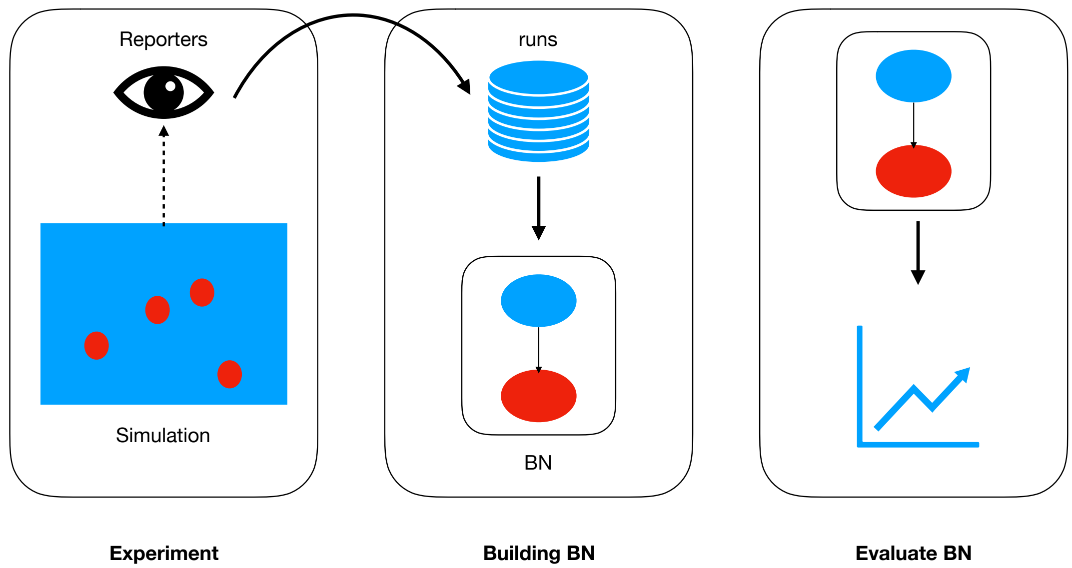


Figure 3.1: Method for evaluating automatically generated Bayesian Networks from simulations.

3.2.1 Simulation

Structure of a simulation

In the simulation, we are simulating some sort of criminal scenario - a theft, usually. Or we are simulating purely for the theoretical things. The simulation can be as precise as necessary, but there are certain things that need to be present: we need to have states that happen, there needs to be evidence for those states as well. The granularity of the simulation and its complexity depend on the modeller and her requirements.

Spatial and Non-spatial simulations.

I'm discussing two types of simulations: spatial simulations and non-spatial simulations. In a spatial simulation, an agent's behaviour is mediated with respect to their environment - eg, agents cannot pass through buildings, or cannot see agents that are far away, or can only steal from another agent when that agent is nearby. In a non-spatial simulations, agents can behave and interact with each other, but this is happening without any environment, hence we are simulating an abstraction. In concrete terms, you can think about a non-spatial simulation as a communication game.

For this project, that means that spatial simulations are more complex and interesting than non-spatial simulations, as there are more possibilities for variety.

The simulations were programmed in MESA, a python package that is made for the creation of Multi-agent systems simulations. We define an environment that the agents can interact with, as well as agents that perform some behaviours. Specifics of simulations and agent behaviour are described in the following chapters.

Predictability and randomness

Where does the interest of the simulation come from? In one part, agents sometimes do things because they are commanded to do so by the computer [1]. This means that, at the start, a random number generator might 'decide' that some agent has a motive, since the random number generator generated a 1 instead of a 0. On the other hand, some randomness arises from interactions between agents, or between agents and their environments. This is where spatial simulations bring additional value compared to non-spatial simulations.

In non-spatial simulations, all agent states are essentially brought about by a combination of randomly generated numbers, and reasoning rules. For example, if an agent has a tendency to lie (randomly generated), and it has the opportunity for lying (brought about due to the current non-spatial simulation), then it will lie. Hence a combination of behavioural rules and randomly generated numbers results in a state of 'agent lies' of 1.

However, in spatial simulations, interactions and behavioural rules and randomly generated numbers are all brought together: if an agent is near another agent (chance interaction), and it has a tendency to steal (randomly generated), then it will attempt (but might not succeed) in stealing. Here the behavioural rule might lead to a more interesting/complex/complicated outcome than in the non-spatial simulation.

3.2.2 Reporters (as Random Variables)

In the simulation, certain states can be brought about. For example, an agent can succeed in stealing an object, or in lying, or in having a motivation (or in not having those things). We need a way to observe these states: this is where reporters come in. A reporter reports the outcome of a relevant event or state in the simulation, and is embedded in the code. If an event happens (or does not happen), the reporter reports that the event is true (or false). In essence, the reporter (R) is a random variable (RV) that maps an event (e) to a truth value:

[1] rephrase, this is always true lol.

$$R : e \to \{0, 1\}$$

Not all the states in the simulation have a reporter associated with it - otherwise I could build infinitely many reporters. I could have reporters for names, for $agent\_Q\_at\_x\_1\_y\_200$. Hence, I only created reporters for states I deemed relevant for the scenario that I am investigating. Here is a subjectivity gap. I can imagine that in my simulation of the Grote Markt (see later chapter), there is some part of the simulation that by chance geometry, lends itself to an easier job for the thief than another part of the simulation. If the thief and victim spawn near this point, then the probability of the thief succeeding will be higher. Increasing the granularity of the reporters might help us determine if there is a spot like this. However, I did not implement this level of granularity in the simulation (yet), because that is a local and specific part, and does not fit into the more global scenario description (the scenario of theft is spatially-free).

The global state of one run of the simulation, is the combination of all reporters.

$$G = (e_0 \to \{0, 1\} \times e_1 \to \{0, 1\} \times ... \times e_n \to \{0, 1\})$$

or,

$$G = R_1 \times R_2 \times ... \times R_n$$

for n reporters.

Then, we collect these global states over the number of runs that we do for each experiment, which results finally that the output O of this stage of the method, is a series of global states, one for each run:

$$O = (G_0, G_1, ...G_{runs})$$

3.3 Creating a Bayesian Network from a Simulation Automatically

The output of an experiment is the collection of runs O, where each run is the global state G of the simulation, as measured by the random variables R. Semantically, it fits that reporters are random variables, as the reporters become the nodes in the Bayesian Network.

Once we have a collection of states and runs, we can give this to an automated Bayesian Network learner, such as those implemented in pyAgrum. These learners can interpolate a Bayesian Network using algorithms, such as Greedy Hillclimbing and K2. This results in automatic generation of the Bayesian Network which is solely based on the data that is collected in O.

For more detail on these algorithms watch this space.

3.4 Evaluating the Bayesian Network

We want to evaluate different aspects of the Bayesian Network: structural criteria, performance criteria and human criteria.

3.4.1 Structural Criteria

1. (temp) Events are ordered temporally - scenario-like.
2. (con) Evidence connects to hypotheses.
3. (exc) All events that are irrelevant are not included in the scenario BN.
4. (exh) All events that are relevant are included in the scenario BN.
5. (ind) Independent events are not connected to each other.

3.4.2 Performance Criteria

1. The accuracy of the network is not lower than the inherent randomness of the simulation.
2. The strong view: probabilities in the network correspond exactly to probabilities in the simulation.
3. The weak view: Updates on evidence in the network go in the same direction as updates on evidence in the simulation.
4. Sensitivity analysis shows that no simulation-irrelevant event influences some output [2]

3.4.3 Human Criteria

1. Do we think that a human can find these numbers?
   (a) How robust is the network against disturbances around the mean (problem with precision)?
   (b) How robust is the network around rounding to arbitrary intervals?

[2] rephrase

Now that we've established how the automatically generated Bayesian Networks are going to be judged, we can show cautiously in the next chapter how well they are holding up!

Chapter 4

A Simple Stabbing - Non-Spatial Simulations and their Bayesian Networks

4.1 Introduction

Here I talk about the credibility game and Charlotte Vlek's Stabbing Bayesian Network example.

In these non-spatial simulations, there are agents, and they interact with each other, but there is no environment for them to interact in. So the simulation is a pure combination of the probabilities assigned to each state transition (or something along those lines). In a sense, the probability for an agent to 'stab with knife' purely depends on the probability of 'motive' and 'opportunity'. Compared to a spatial simulation, where 'opportunity' is more complex, as it involves proximity to victim which arises from the simulation and not from some random number generator.

These non-spatial simulations are boring, but they are necessary first steps: after all, if we cannot make BNs out of these predetermined (the probabilities in each run are not predetermined, but the distributions that they are drawn from, and their thresholds, are), then the rest of our endeavours will be fruitless. On the other hand, if the process for creating and evaluating these simple simulations work well, then we can proceed to modelling more complex, spatial simulations.

In this part, I have two experiments. One simple experiment meant to be a replication of Charlotte Vlek's Bayesian Network in xxx [1], and another simple experiment mainly

[1] rephrase

meant to test the evaluation criteria for the Bayesian Networks, as outlined in the previous chapter.

4.2 Method

Take the Vlek networks from Vlek Jurix 2015.

The general story is: Jane and Mark had a fight, but Jane had a knife. Mark died.

Then, there are two specific scenarios that can explain why Mark died. In scenario one, Jane stabbed Mark, and then he died. In scenario 2, Jane threatened Mark with the knife, Mark hit Jane, Jane dropped the knife, Mark fell on the knife, and Mark died by accident. There are two separate networks for these scenarios.

I'm going to create two separate networks, and then also see if I can merge them, by creating a Jane-and-Mark-knife simulation, assigning some random probabilities that correspond to the story, and see what the K2 algorithm makes of it. Then I will also merge the two networks to see if the K2 can deal with mutually exclusive nodes (eg: 'Mark died by accident' should rule out 'Mark died by stabbing').

So, I created some logical rules, either atoms or rules, and we can process in the simulation these using standard forward chaining inference. Every atom has a prior probability, and every conclusion of a rule has a probability given the F/T state of the premises. At every step, the simulation checks which new sentences are true, applies a rule with a given probability, and counts the outcome.

This does mean that rules at the end are not triggered as often as rules in the beginning, even if they have the same trigger probability (they are on other sides of the chain, more needs to be have happened to conclude that Mark died).

4.2.1 Behavioural rules for the simulation

This was done using forward chaining, with a time index, which means that a rule could only be triggered at one time (otherwise the probabilities get messed up). So we have a forward chaining rule, which means that if the premises of some rule are true, there are no excluding facts true, the timestep is correct, and the random number generator generated a number that is lower than the probability threshold, we find that the conclusion is true, and add it to our found facts. We keep doing this.

Vlek's paper contains no probabilities, so I'm just making some up. The logical sentences are reporters in the simulation, and then I let the simulation run for 10,000 times.

Premise	Conclusion	P(conclusion) given premises
	Jane and Mark fight	20
	Jane has knife	70
Jane and Mark fight, Jane has a knife	Jane stabs Mark with knife	1
Jane and Mark fight, Jane has a knife	Jane threatens Mark with knife	3
Jane threatens Mark with knife	Mark hits Jane	90
Mark hits Jane	Jane drops knife	50
Jane drops knife	Mark falls on knife	10
Mark falls on knife	Mark dies by accident	60
Jane stabs Mark with knife	Mark dies	70
Mark dies by accident	Mark dies	100

Table 4.1: For combined scenarios: blue rules belong to scenario 1, red rules belong to scenario 2, and black rules belong to both.

4.3 Results

We succeeded in generating three different BNs: One for scenario 1, one for scenario 2, and one for the combined network (Figure 4.1, Figure 4.3, Figure 4.5).

4.4 Discussion

How well do these networks evaluate compared to the criteria we set out initially, and how well do they compare to the Vlek networks, and the criteria set out in Vlek 2015?

4.4.1 Structural Criteria

1. (temp) Events are ordered temporally - scenario-like.

   In the forward chaining knowledge base, we have premises and conclusions. If we take the event in a conclusion, then we know that the premise of that rule, is the parent of that node in the Bayesian Network. (again, rephrase this is so vague. Maybe make it a rule or smth). The two "premiseless" events Jane and Mark fight, and Jane has a knife, are also parentless in the simulation. In this sense, the structure of the Bayesian Network reflects the structure of the inference rules that we set out.

   We see that in both the separate networks, the temporal ordering of the nodes is correct - every node has as a parent a node that represents an event that would have happened beforehand. In the combined network, this is not the case: mostly the temporal links are represented ok (eg: the chain of events from "Jane threatening Mark" to "Mark dies by accident" is represented correctly), however, this chain of nodes all has as a parent "Mark dies", which would, temporally, occur only after

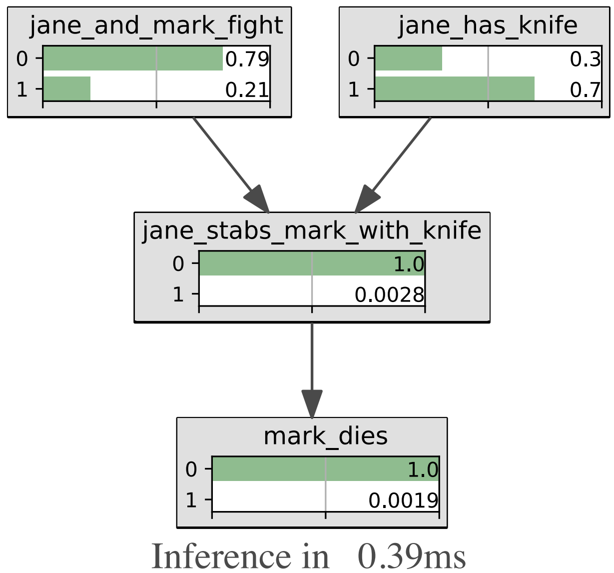


Figure 4.1: Automatically generated BN with K2 and the above forward chaining rules, scenario 1.

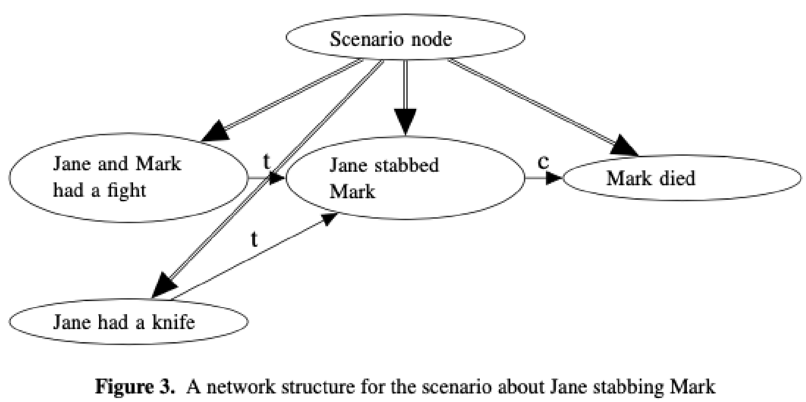


Figure 4.2: Vlek BN.

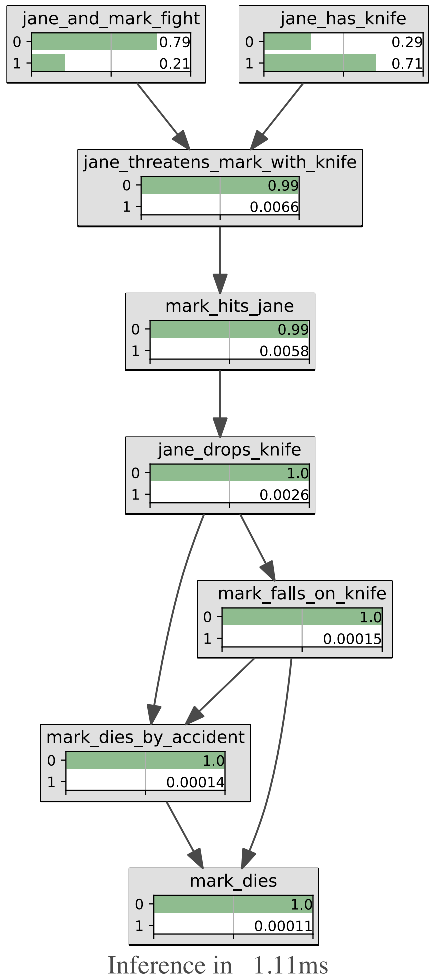


Figure 4.3: Automatic generated BN with K2 and above forward chaining rules, scenario 2

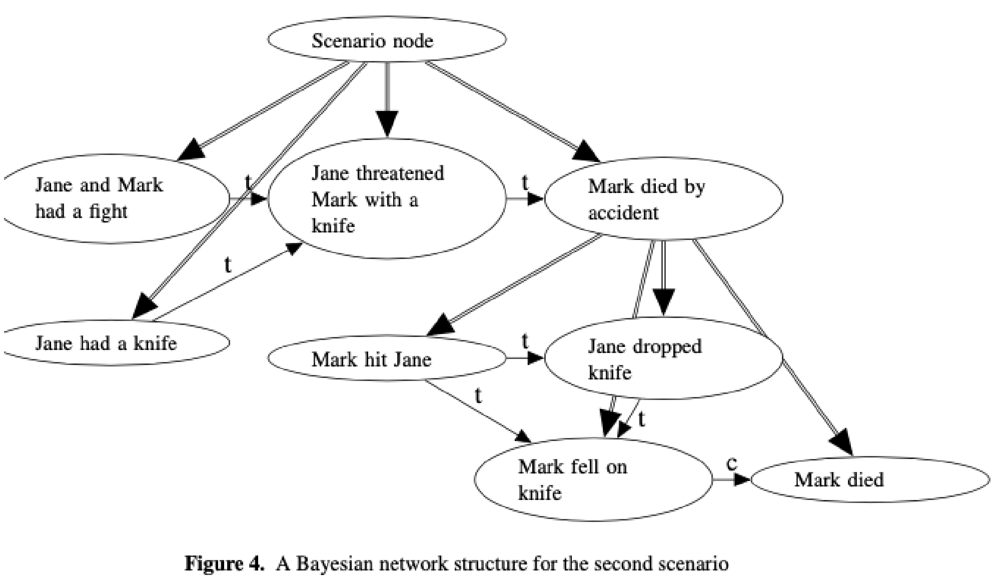


Figure 4. A Bayesian network structure for the second scenario

Figure 4.4: Vlek BN.

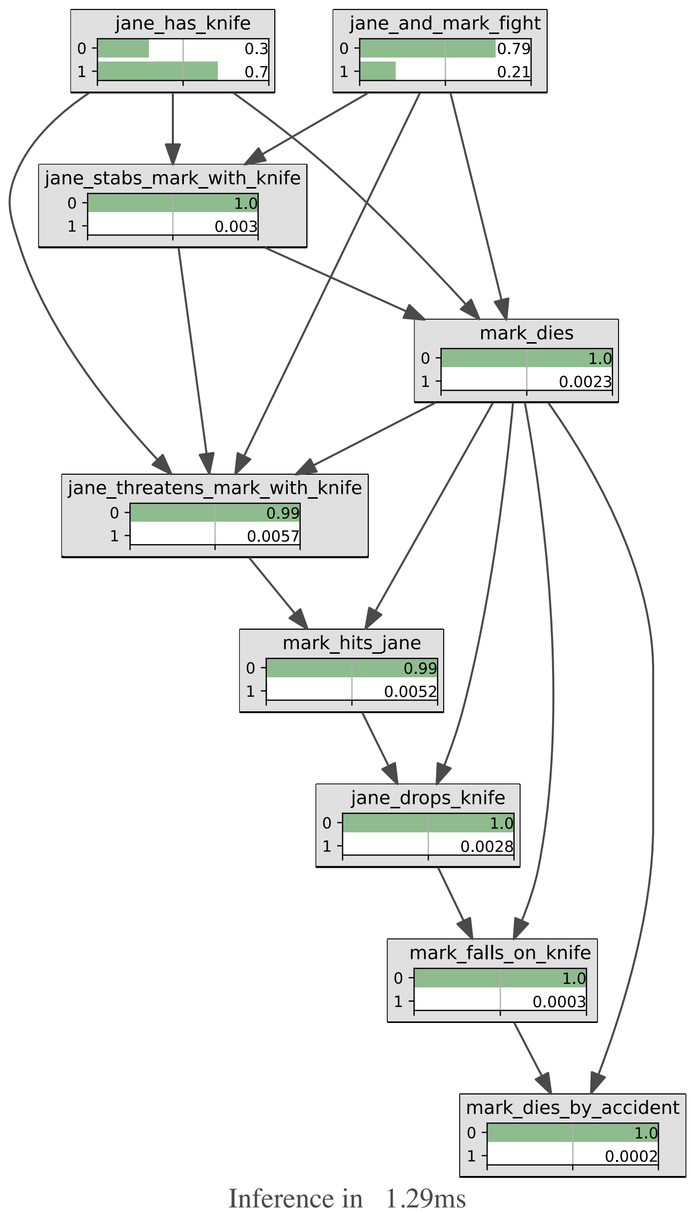


Figure 4.5: Automatically generated BN with rules from both scenarios included.

"Mark falls on knife" occurs. This means that in the combined Bayesian Network, we cannot satisfy the temporal ordering constraint due to the conditional probabilities between the events (eg: if Jane stabs Mark, then she cannot threaten him after, which is a relation that 'inhibits' the 'Jane threatening Mark' node - eg: there is a conditional relation between the two nodes, but not a temporal or causal one). As a conclusion about this criteria: Maybe temporal ordering should just hold within scenarios and not between scenarios, because we get into these sorts of causal messes. Or we should use the scenario node to organise different scenarios together. Idea: scenario node could cause temporal ordering, but what in cases with much shared evidence?

2. (con) Evidence connects to hypotheses.

   In the original network, we have no evidence, so it is not included into this one. However, I might add some evidence to test the Performance Criteria.

3. (exc) All events that are irrelevant are not included in the scenario BN.

   This criteria, and the next one, are freebies, because we are just copying a network that already exist - our set of relevant events is the same as that in the original network by Vlek. We do not have any irrelevant event because we can see that all events are connected to each other, and all events reflect an underlying 'decision' by an agent.

4. (exh) All events that are relevant are included in the scenario BN.

   We have included all relevant events - all the events in the knowledge base are also in the network.

5. (ind) Independent events are not connected to each other.

   This is not entirely correct in the uncombined networks (and also not correct in the combined network) - If Mark dies by accident, this is only dependent on Mark falling on the knife, and not also on Jane dropping the knife (since all cases where Mark dies by accident are also cases where Mark falls on the knife, we don't need a separate condition for Jane dropping the knife). This is probably due to an insufficient number of runs (eg: the probability that this happens is very low and we might not have enough data to correctly estimate it).

4.4.2 Performance Criteria

1. The accuracy of the network is not lower than the inherent randomness of the simulation.

   We see that as the disturbance increases (eg we round to larger intervals), accuracy obviously decreases, and RMS increases. But these decreases and increases are not

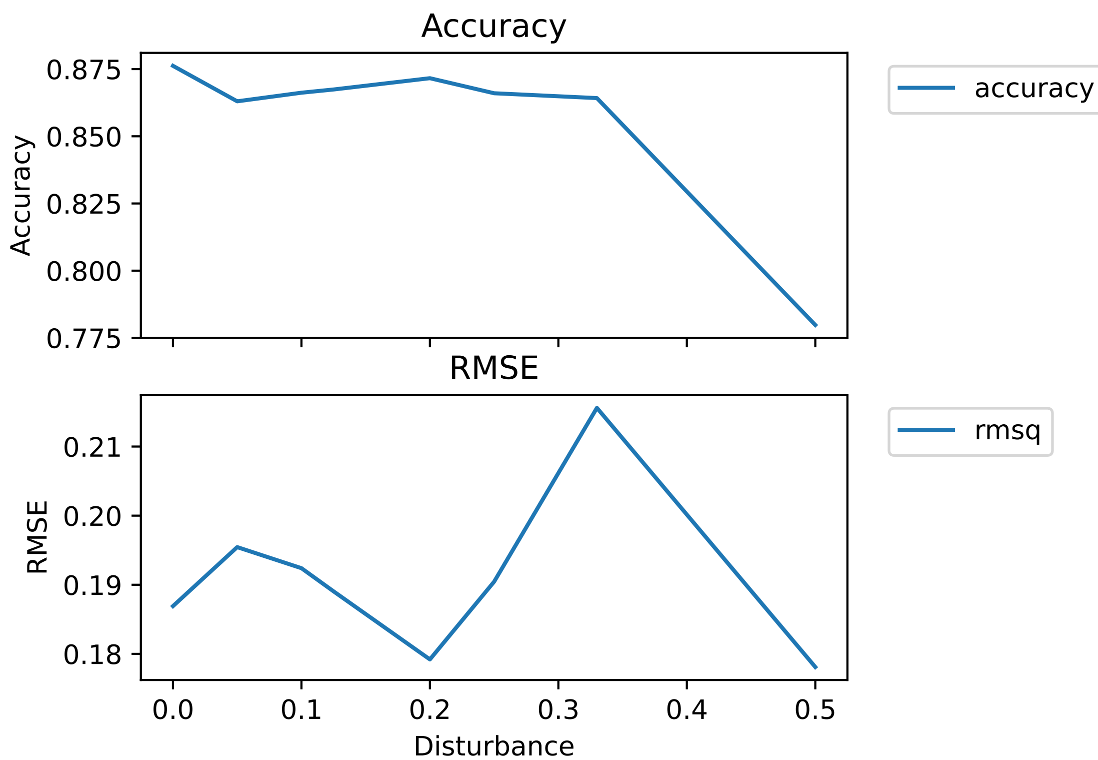


Figure 4.6: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in network 1.

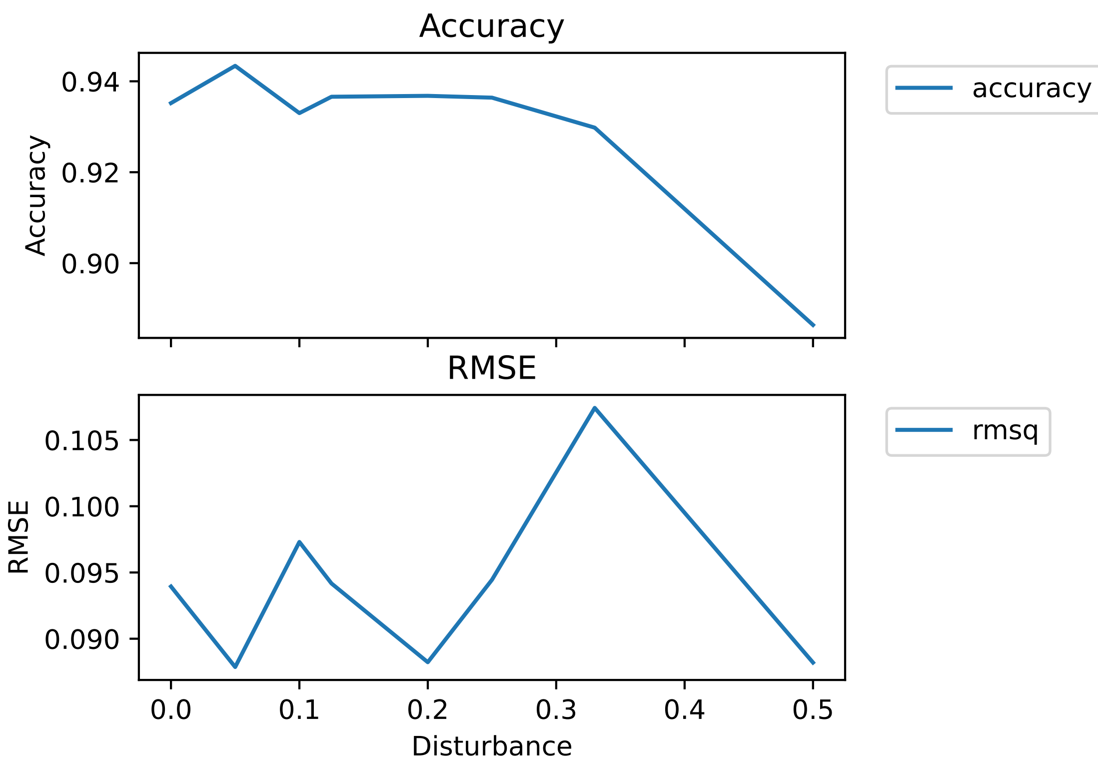


Figure 4.7: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in network 2.

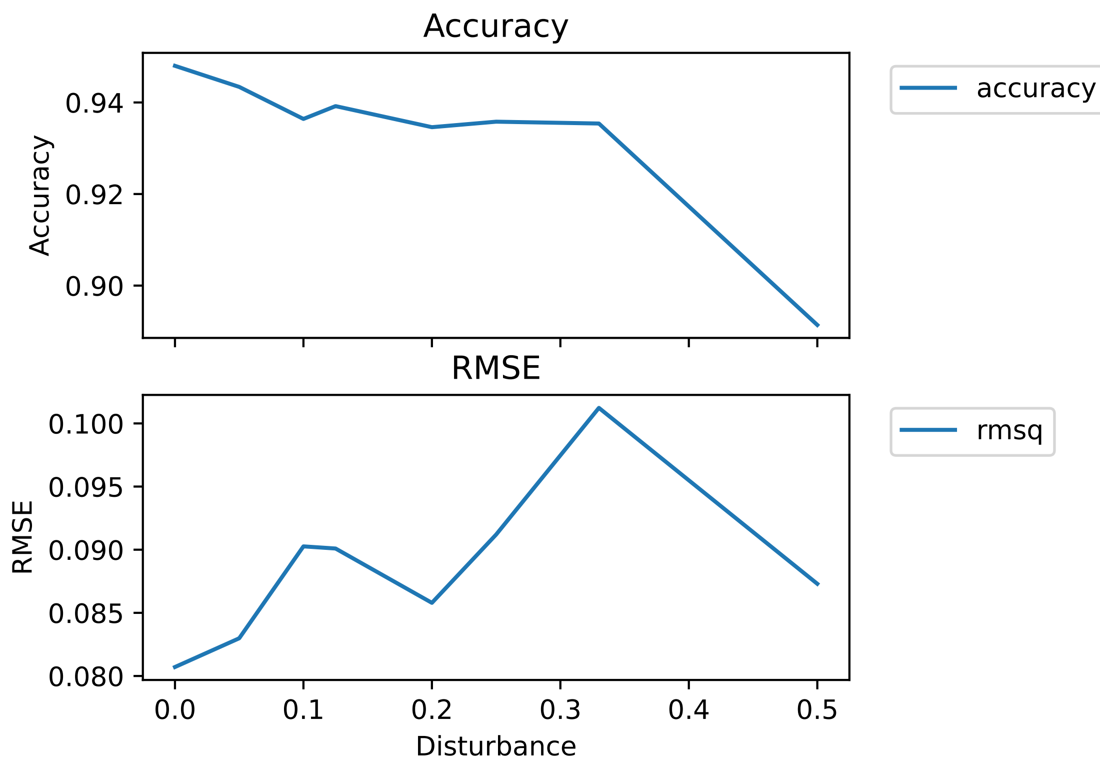


Figure 4.8: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in the full network.

Conclusion	P(conclusion) given premises	P(event) given premises
Jane and Mark fight	20	20.87
Jane has knife	70	70.23
Jane stabs Mark with knife	1	2.05
Mark dies	70	75.46

Table 4.2: For only scenario 1.

Conclusion	P(conclusion) given premises	P(event) given premises
Jane and Mark fight	20	20.87
Jane has knife	70	70.23
Jane threatens Mark with knife	3	3.88
Mark hits Jane	90	90.21
Jane drops knife	50	50.09
Mark falls on knife	10	11.13
Mark dies by accident	60	61.25
Mark dies	100	99.79

Table 4.3: For only scenario 2

that drastic, and overall performance is still pretty good (eg even for the worst, accuracy is still near 0.88)

2. The strong and the weak view.

   The strong view: probabilities in the network correspond exactly to probabilities in the simulation. The weak view: Updates on evidence in the network go in the same direction as updates on evidence in the simulation.

   For both scenarios, the probabilities in the network are very similar to the probabilities in the ground truth (simulation). There is a loss of precision, which is the worst in case of 'Mark dies' in the first scenario, which is estimated at 75% by the network, but in fact in the simulation is only 70% probable. I don't know where this comes from. Apart from that, all divergences for the original probabilities are within 2%. This means the strong view does not hold, but the weak view does (updating on evidence gets us with probabilities that are closer to the simulation probabilities than before).

   The probabilities of the conclusions in the combined network given the premises, also look relatively similar on a human scale - I assume that a human probability elicitor would not be able to distinguish between a probability of 11.13, 10.92 or 10.00 for the conclusion "Mark falls on knife". On the other hand, if we round all the probabilities in the network to 10, a human probability elicitor might be able to say that the

Conclusion	P(conclusion) given premises	P(event) given premises
Jane and Mark fight	20	20.87
Jane has knife	70	70.17
Jane stabs Mark with knife	1	2.06
Jane threatens Mark with knife	3	3.88
Mark hits Jane	90	91.50
Jane drops knife	50	52.97
Mark falls on knife	10	10.92
Mark dies by accident	60	66.33
Mark dies (premise: Jane stabs Mark)	70	68.68
Mark dies (premise: Mark dies by accident)	100	98.77

Table 4.4: For combined scenarios

probability of 'Mark falls on knife" is 0.1 (or 10%), to exactly that level of precision (so not 0.10, or 0.100). The resulting probabilities might be elicitable, while still meaningfully reflecting the ground truth.

3. Sensitivity analysis shows that no simulation-irrelevant event influences some output [2]

4.4.3 Human Criteria

1. Do we think that a human can find these numbers?

   The conditional probability tables as displayed in these networks are not great.

   (a) How robust is the network against disturbances around the mean (problem with precision)?

   (b) How robust is the network around rounding to arbitrary intervals?

2. Can a human find this network & reason with it? All the evidence points in the correct direction, in the weak and strong views we're ok, even for relatively large intervals - we see a sharp drop in accuracy and increase in RMS error at around intervals of 0.33. This would mean, that for these networks, the elicitor should only need to specify probabilities that are 0, 0.33, 0.66, and 1 (eg, round every probability that is not one of these, to one of these). Note, that this is likely different dependent on the complexity of the networks (in some way, lol). This sounds definitely plausible as a lower level. If someone has strong opinions on the CPT then we can always increase precision in certain nodes. Hence, this seems pretty great.

[2] rephrase

4.4.4 Comparison to Vlek 2015

If we only look at the ordering of the Bayesian Network, we can see that for Figure 4.1, the sub-scenario structure is the same as in Vlek 2015 Figure 3, our Figure 4.2. There's no scenario node constraining the network, all the information is contained in the network, no scenario node needed (todo: make the nodes the exact same names).

However, there are differences between Vlek 2015 Figure 4, and the automatically generated BN here. Figure 4 in Vlek 2015 is replicated below in Figure 4.4 (todo: ask permission?? or just remove). The automatically generated BN is very linear and can be interpreted in a purely temporal way: mark hits jane, jane drops the knife, and due to jane dropping the knife, and mark falling on it, mark dies by accident - and if mark dies by accident, mark dies. The probabilities for all these events (from jane dropping the knife on), are ridiculously low, and don't really make sense (should interpret the small probabilities as e, and not as actual numbers I guess, due to underflow?).

In Vlek's paper, we see a subscenario: we have the subscenario "Mark died by accident", which contains events such like mark hit jane, which leads to jane dropping the knife, and mark falling on the knife, and then dying. The coparents of Mark died are the same in this network as in the automatically generated BN, however, we once-again miss the scenario-like construction where mark hitting jane, jane dropping the knife, and mark falling on the knife are connected as part of a subscenario, rather than their "own" nodes.

Why to choose for subscenarios when it is not required? Tomorrow I will look up why the scenario construction was used and I will see if it does something that I miss? Coherence? But that also travels up the chain? But something with d-separation probably. I'll read the Vlek 2015 again i guess.

In the full final network, we see that the two mutually exclusive scenarios (stabbing vs falling on knife) are excluding each other even without the use of a scenario node - so we can make networks that combine two different stories. This is nice.

4.4.5 Quality of scenario

- Completeness - have all the parts and not one more. Not a big deal for this simulation, as I've taken all the nodes from Vlek 2015, there's no discussion about granularity.
- Consistent - no internal contradictions. There is no explicit structure in the final BN that ensures consistency. The probabilities inside the BN need to ensure that. For instance, the only thing that is contradictory in the KB is that jane threatening and jane stabbing, cannot both happen. In the final network (Figure 4.5), there are nodes for both threatening and stabbing. However, the relations of these nodes are such, that when one node becomes more likely (eg, we set evidence on it), the other node becomes less likely (as we can see in the figure below:

figure here of Δ P stabbing for increasing values of P threatening. If we know threatening is true, stabbing goes to 0, and vice-versa - and all the consequences of the threatening scenario also go to 0, so it just works. This means that several different scenarios can be combined into one network, and still be mutually exclusive. Both scenarios share some events (as some scenarios can do in real life), but the different scenarios themselves are not separated from each other by structure or construction (as is the case in Vlek 2015).

Do we need the constraint node? Or do we only use that if we have knowledge but no data? what. 11 in Vlek.

We can just straight-up draw a line between two events in order for them to become mutually exclusive, like we do with Jane stabs mark and jane threatens mark. Fenton et al advice against that - we draw a relation between two causes to ensure mutual exclusivity. Fenton doesn't like this even though it satisfies the axioms because 1) the parent cause becomes part of the causal pathway leading to the child cause, which means that you have to involve many more numbers ("meaningless columns"). Fortunately, this is not a problem in the automatically generated BNs, because the algorithm fixes this for us :). Not sure if an extra node is good for the computational complexity of BNs either. Uh. Anyway, even if this is not the case, there are lots of unnecessary numbers in Bayesian Networks anyway (show image here), because there's many combinations of events that just do not happen (eg: mark dies but jane doesn't have a knife, never happens, but is in the table anyway. Unnecessary complexity? Probably a rounding error!!). Anyway, that's a thing. So it doesn't really matter.

Uh. The second objection is that you have to arbitrarily decide which cause is the parent - which doesn't make sense if you interpret the networks causally. Fortunately. there's no causality in this part, its just frequencies so it doesn't matter, we can just pick one and it's fine. So turns out we don't need the constraint node anyway :D

- Plausible - scenario should correspond to the modellers knowledge about the world. Support can help implausible scenarios to become plausible.

4.5 Important take-aways

In this section, I summarise some important take-aways, that will be necessary for the rest of this project. Things that we learn here, will be useful in evaluating the rest.

4.5.1 Simulations work

Even this very simple 'forward chaining' simulation (eg: forward chaining with some probabilities attached), can be used to construct meaningful Bayesian Networks. The Bayesian

Networks have generally ok accuracy and RMS error, and reflect the structure of the simulation (dependent on the number of runs, that's true). Even when we disturb the CPTs in the table, we get relatively ok accuracy and RMS until

4.5.2 We don't need the scenario node for mutual exclusivity

Both of Fenton's objections for the direct node connection are not problems for us (yet) - the first one, specifying too many numbers, might become problematic in the future, and the second, about the interpretation of the arcs between nodes is not a problem, because we interpret these links as solely conditional, and not causal. Background information about causal relations between events might help us to order the network, so that it can become more efficient, but this should not affect the "reasoning" aspect - eg, even if the structure of the network is different, updating on evidence should produce the same results as before.

4.5.3 Resolution and number of runs

The accuracy of the Bayesian Network improves as the number of runs improves. This is an obvious fact. It has implications for the rest of the network. For example, if we run the network too few times (let's say 10 times), then the network will not be able to estimate the probability of certain states - eg: the probability that Mark hits Jane can be calculated as: $0.7 \cdot 0.2 \cdot 0.03 \cdot 0.9 = 0.00379$, which means that there's a 0.378% chance that it happens, eg, if we run the simulation a 1000 times, Mark will hit Jane in 4 of them (or, actually, the sentence "Mark hits Jane" is true in 4 of them). If we go all the way down the chain of unreasonable facts to "Mark dies accidentally", we get a probability of 0.0001134, which is 1 every 10.000 runs. If we run the simulation 10.000 times, that means that we cannot estimate probabilities that are smaller than 0.0001 - either they occur 'accidentally', eg, by chance, and the network estimates that they happen once every 10.000 runs, or they don't happen within 10.000 runs, and the network estimates that these events never happen at all - when in fact, they might happen, but just once every 20.000 runs.

What are the practical implications of this? On this side, that we need enough data to get an accurate prediction. But, if we're thinking about elicitation, this means that if you decide to add an event to a scenario, or a node to a network, you really have to think about the probability that you assign it - a probability of 1 in 10.000 would really mean that this event would only happen once every 10.000 situations (that you are considering - the reference class is never far away).

4.6 Conclusion

With this part, I have shown that the pipeline works. We can make a simulation using forward chaining, that generates world-states, and those world states can be translated into Bayesian Networks by means of the K2 algorithm, with reasonable accuracy even under rounding. The K2 manages to replicate constraints and merge two different possible scenarios together. So if the real world worked like a simple forward chaining inference method where we knew exactly when a proposition was true (which is, if the state is true, we set the proposition to true in every epoch), then BNs would work flawlessly even if we weren't very great at estimating the correct frequencies precisely.

Sad that the real world doesn't work that way.

Chapter 5

A Simple Robbery - Spatial Simulations and their Bayesian Networks

Three experiments: 1) getting the bayesian networks with disturbances (). 2) testing the effect of a simulation parameter change on the BN (how different does it become. 3) investigating the effect of hidden information/private knowledge on the network. Summary:

The problem is that in the previous chapter, we have established that we have a method to convert a simple, forward chaining simulation, to a collection of data, which then in turn can be used to create a Bayesian Network, that represents the situation in the simulation relatively well (some problems none-withstanding, accuracy and RMS error are generally ok even in increments that people might plausibly be able to estimate intervals of 0.33/0.25-ish). However, this was totally 100% determined by deterministic forward chaining rules. The real world does not run on deterministic forward chaining rules (as far as we know). At least, we are spatially situated, which means that there are probabilities that will arise out of interactions between agents and their environment. These probabilities are not 'set' in the same ways as the probabilities are set in the previous chapter, they arise organically from interactions: eg: an agent has to be located at a door to break in, an agent can be seen by a camera only in some locations of the simulation. We do not know these probabilities a-priori (although we can probably calculate them, I don't know).

New idea: We create spatially situated simulations that are more complex, and see if they work the same way/are as accurate & rms as the previous. Then we will also use these simple spatial simulations to test two hypotheses about 1) the effect of private knowledge and 2) the effect of parameters.

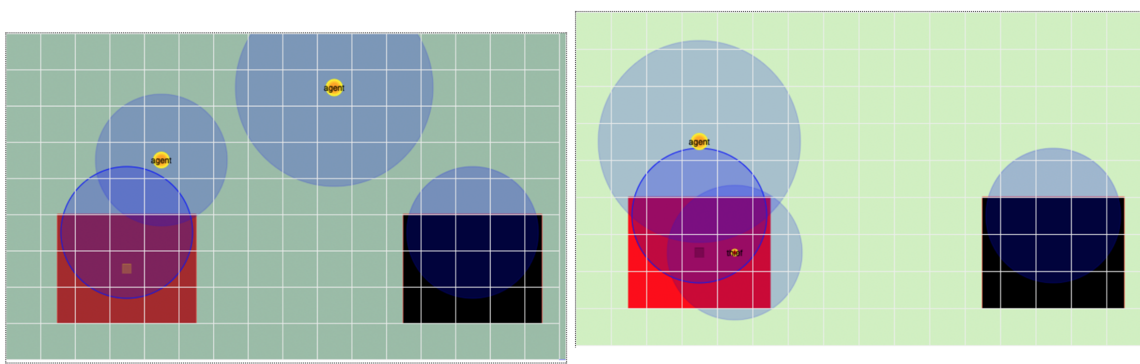
RV	Operationalization
lost_object	A random number generator (0, 1) generates a number. If the number ≤ 0.2, the object is lost.
curtains	A random number generator (0, 1) generates a number. If the number ≤ 0.8, the house has no curtains.
raining	A random number generator (0, 1) generates a number. If the number ≤ 0.5, it is raining.
know_object	if we see an object in our vision that is not our own, and we are not already targeting something else
target_object	if we know the object exists, and we consider the value of the object higher than our risk threshold.
motive	if we have a target
compromise_house	if we are adjacent to the target's house's door, and we have a breaking and entering skill of greater than 5.
flees_startled	if we see another agent and we haven't been observed yet
successful_stolen	if we're not in someone's house anymore and we have the object in our possession
E_s_spotted_by_house	
E_disturbed_house	
E_object_is_gone	
E_broken_lock	
E_s_spotted_with_goodie	
E_private	

Table 5.1: This is so tedious.

5.1 Experiment 1: Generating BNs with disturbances

5.1.1 Introduction

Here I talk about the robbery example with the evidence.



(a) The simulation with the 2 agents.

(b) The agent attempts to steal something but flees when it thinks that it is noticed.

5.1.2 Methods

Same as before.

table where I define all the variables here.

5.1.3 Results

image of network.

See Figure 5.3

Overall pretty accurate. This shows that small disturbances in rounding - and even larger disturbances in rounding, such as to 0.25 or even 0.33, does not matter very much for the accuracy of this specific network.

5.1.4 Discussion

The network itself makes sense. The spurious node 'raining' is not connected to any other node, because the rain should not affect what happens.

The fact that these networks can be rounded without losing accuracy, means that even with a lack of precision in the ctps of the network, the network is still able to accurately predict outputs - eg, whether a node is going to be true or false given a set of evidence. Changing the precision of the network does not change the network structure itself, this is kept constant - it is generated first, and then the cpts are rounded to arbitrary intervals.

This shows that even when this network's cpt's only contain probability values, [0, 0.33, 0.66, 0.99], we still get the reflection of the evidence in the network.

To note - network structure is not trivial. Can we place restrictions on network structure (such as evidence not connecting to each other) and preserve accuracy? Keep the network temporal? The network structure is not temporal right now.

How would a judge/lawyer interpret this?

5.2 Experiment 2: Investigating the evidence strength depending on parameters of the network.

5.2.1 Introduction

A funny thing is that this simulation we can test how the evidence strength is dependent on certain parameters in the network. For a simple example, see the radius of camera vision. If we have the Reporter "agent seen near house"/"agent spotted by camera", and by that we mean, that if the agent is visible in the camera placed near the house, then the range of vision of the camera becomes relevant for the investigation. If the camera is super good and can see the agent even when he's not near the house, then the effect of being-seen-on-camera should decrease: it becomes less relevant that the agent was seen by the camera, because they usually are. On the other hand, if the camera is pointed only at the door, then being seen by the camera is relevant, the agent is only by the door when he's trying to break in. Below you see a plot of camera vision range vs effect on posterior when the node is turned on (given the same structure of the BN).

5.2.2 Methods

I changed the parameter of vision of the camera for the simulation between 1 and 15, for the rest the simulation was the exact same. I didn't apply any disturbances to the network.

5.2.3 Results

5.2.4 Discussion

We see here, that the name of the variable is actually incorrect. It shouldn't be called "agent seen near house", because the reporter is not actually reporting that the agent is near the house - instead it is reporting whether the agent is within the vision of the camera. So properly it should be called "agent is seen in camera", or we can rewrite the reporter to measure whether the agent is actually near the house.

But, this means that the BN is not responsive to the effect of relevance in the real world.

How would a judge/lawyer interpret this?

5.3 Experiment 3: Investigating the effect of private knowledge

5.3.1 Introduction

5.3.2 Method

Drop column Private knowledge and evidence for it, see the effect on the posterior. See the effect on the disturbances.

5.3.3 Results

New network. New accuracy and RMS plots.

5.3.4 Discussion

Seems to be effective, posterior never goes to 1. Lack of private knowledge hence will affect the end results, even if it will not affect the trajectory of the evidence progression. Overall accuracy and rms performs worse as well. How would a judge/lawyer interpret this?

5.4 Take aways from the three experiments

Simulating works, accuracy and RMS fine.

Table of accuracies and rms for the three experiments for comparisons.

We run into problems when we use words like ‘near’ in our node names. We’re lucky that we know what we mean (because our reporter forces us to make this explicit). We do not just ground the probabilities in our network, but we actually also ground our random variables - we have a measure of exactly what events we are interested in, and which we aren’t.

We are in trouble with private knowledge, dropping this column from our table (which means that we don’t know it) when we make our BN, we’re drastically reducing our uh. Accuracy, and increasing our RMS. This implies that we do seem to need a full picture of everything that’s happening, otherwise our BN will be kind of shit, and be less accurate :(.

Hence, for a simulation to work we 1) need to know exactly what it is that we’re measuring since evidence strength depends on it [1], and 2) private information is necessary to create the BN because it influences fleeing behaviour. If we don’t have this information our BN will be less effective.

[1] explain this more

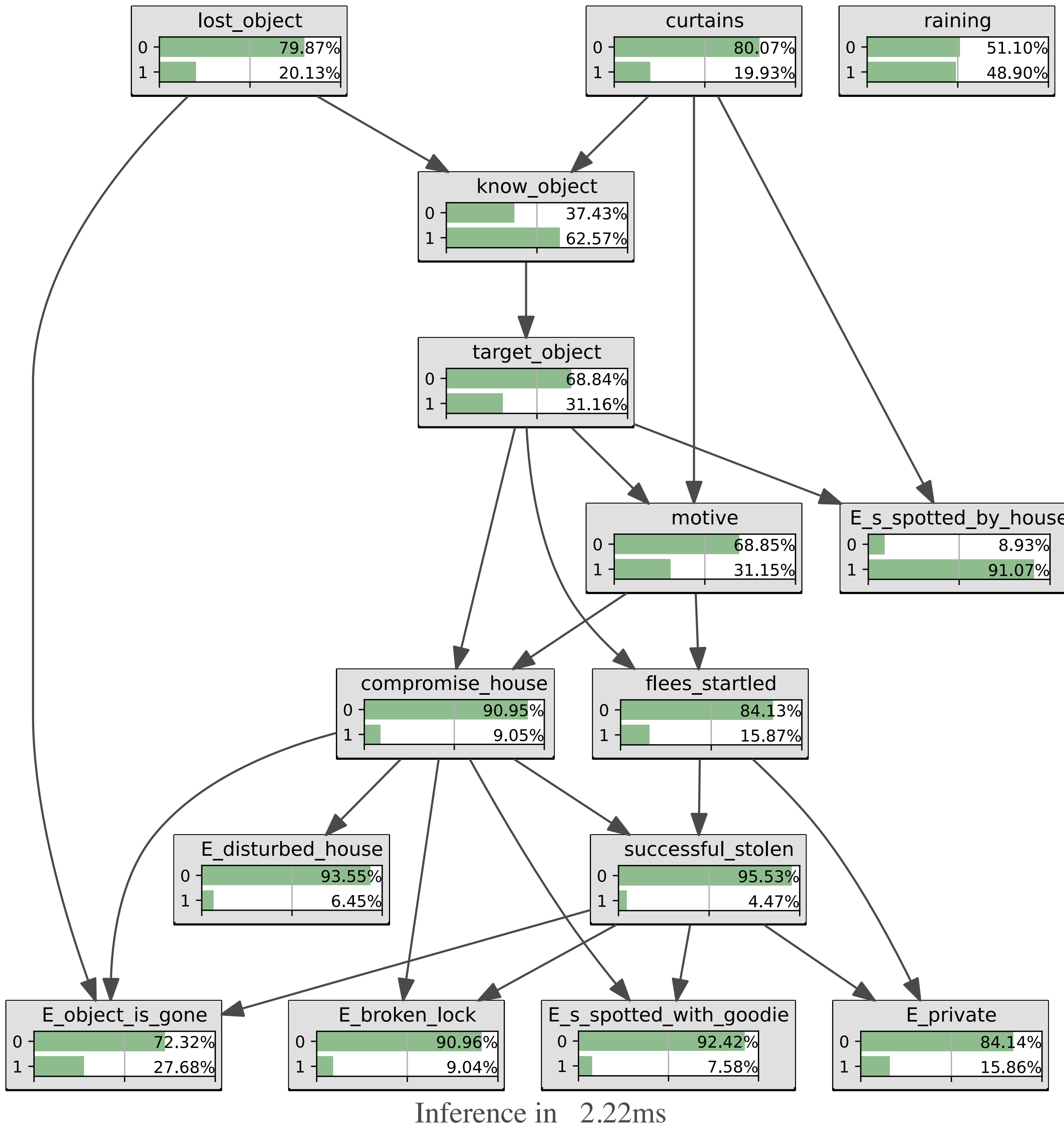


Figure 5.2: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in stolenLaptop network and simulation

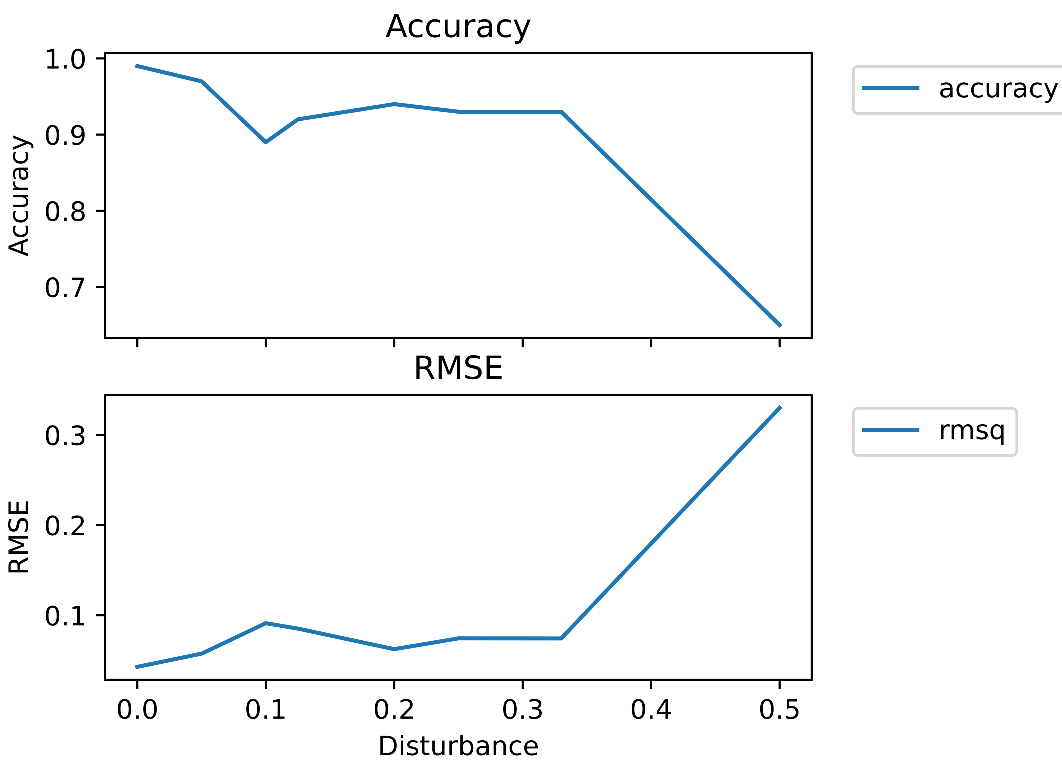


Figure 5.3: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in stolenLaptop network and simulation

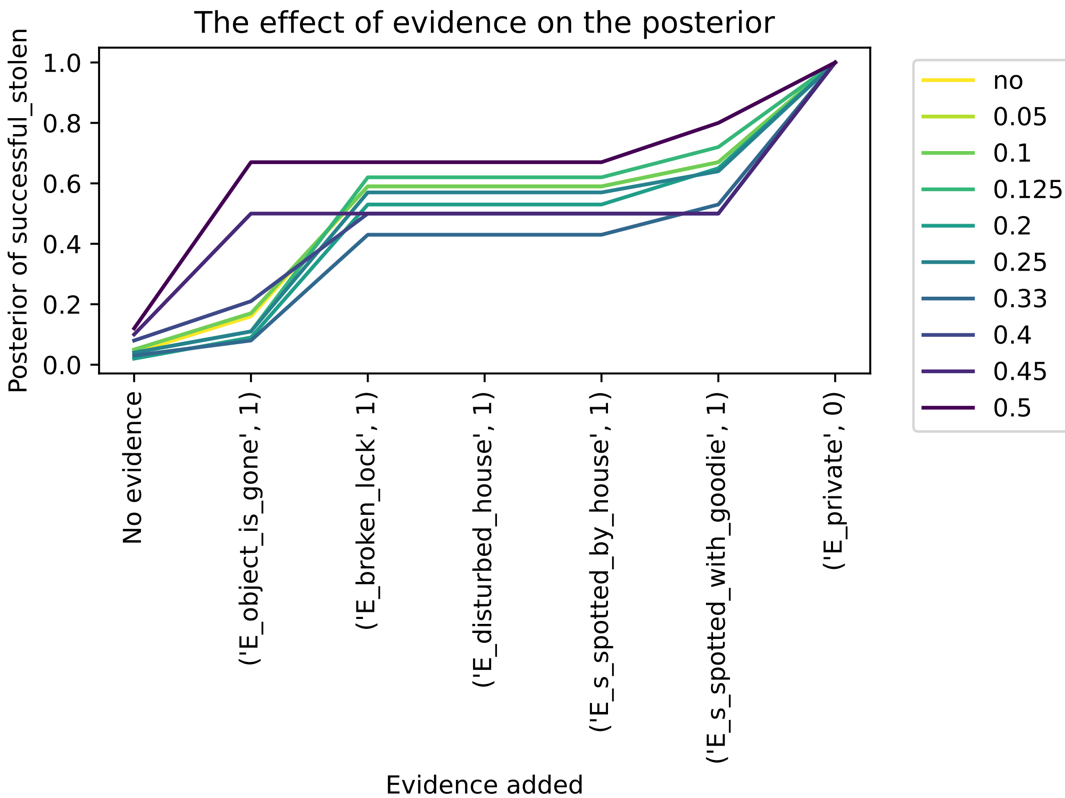


Figure 5.4: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in stolenLaptop network and simulation

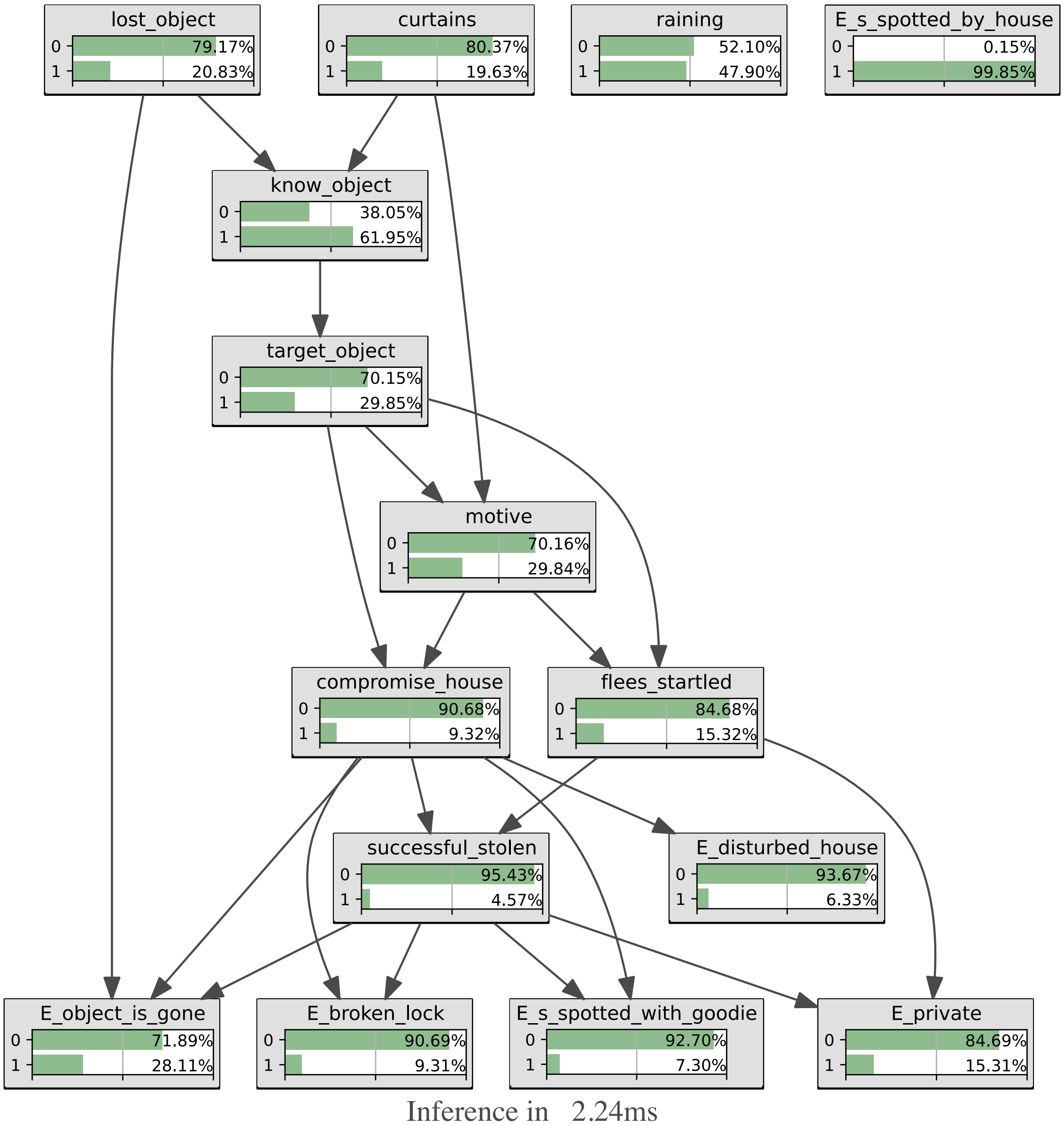


Figure 5.5: Vision. Doesn't matter.

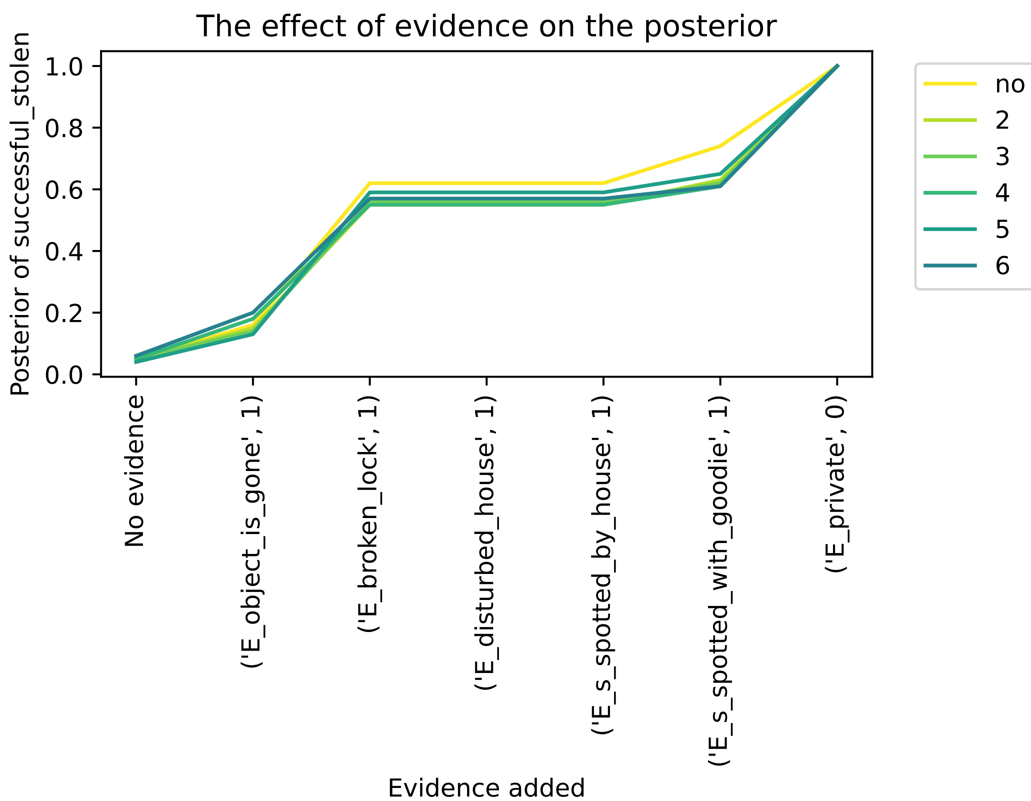


Figure 5.6: Vision. Doesn't matter.

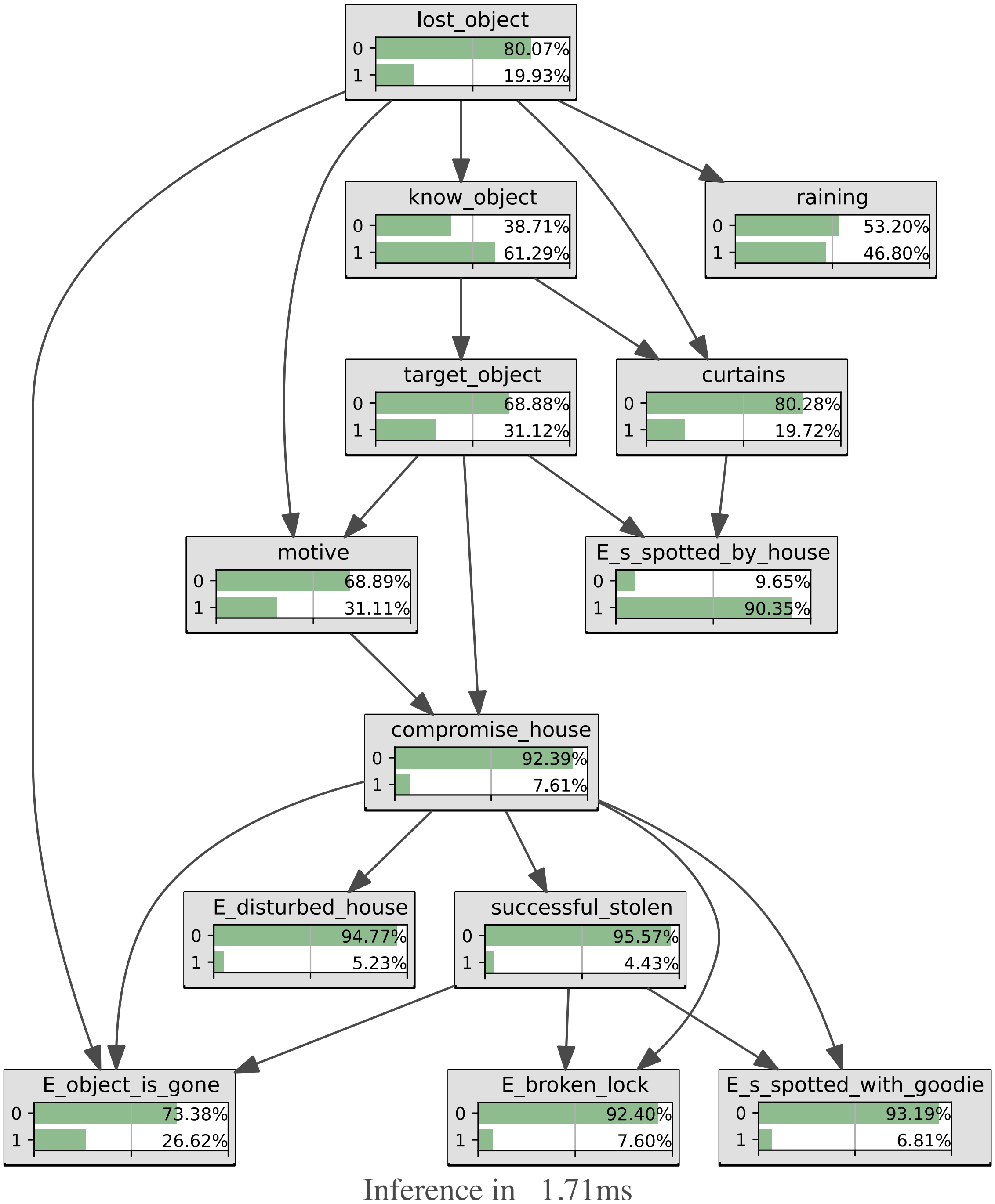


Figure 5.7: Private evidence

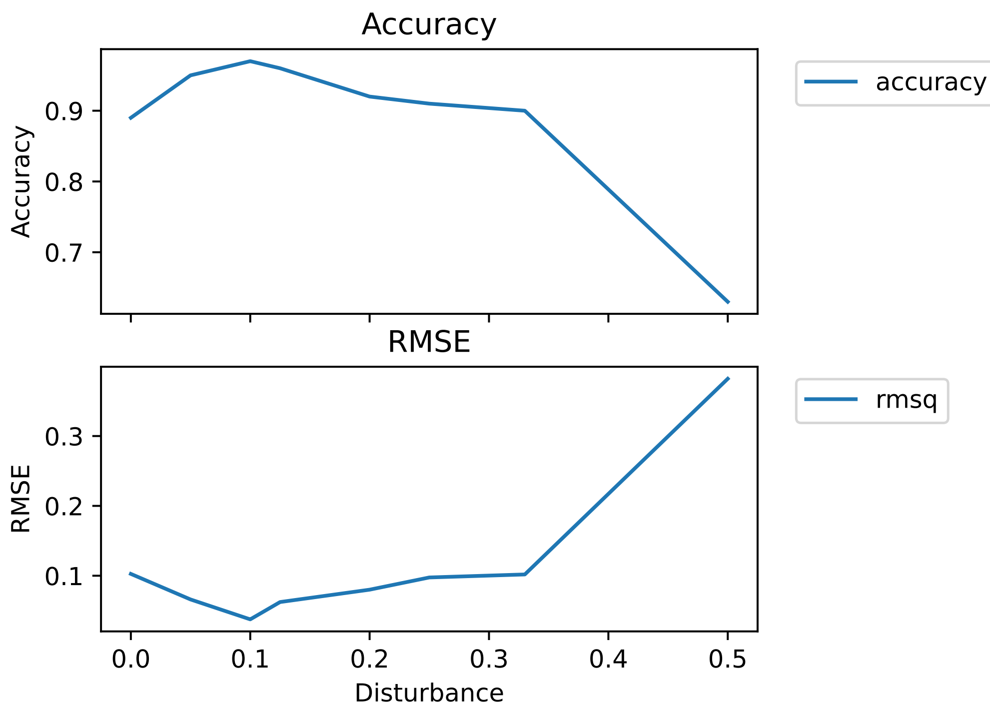


Figure 5.8: Private evidence

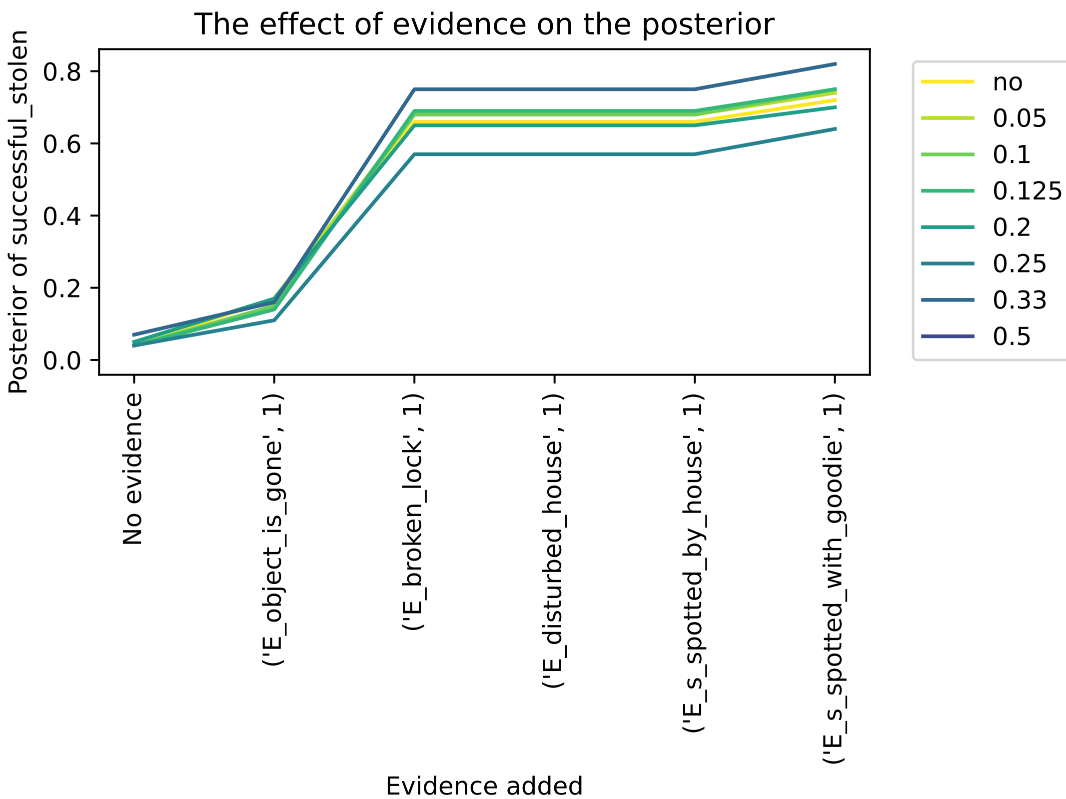


Figure 5.9: Private evidence

Chapter 6

Robberies

Why now go to a larger map, if we have already identified some problems? WEll, there can always be more problems :D

New idea: A larger, more interesting spatial situation. we know we can have agents walking around a simple flat simulation, but now we want a more life-like situation, a town, and they're going to rob shit.

6.1 General information

Agent behavioural loop, etc. Look at the code its online.

6.2 Experiment 1: General - can we create BNs of a robbery at Grote Markt?

6.2.1 Introduction

Here I talk about the Grote Markt and the more complex simulations that I created for it, including the island prior.

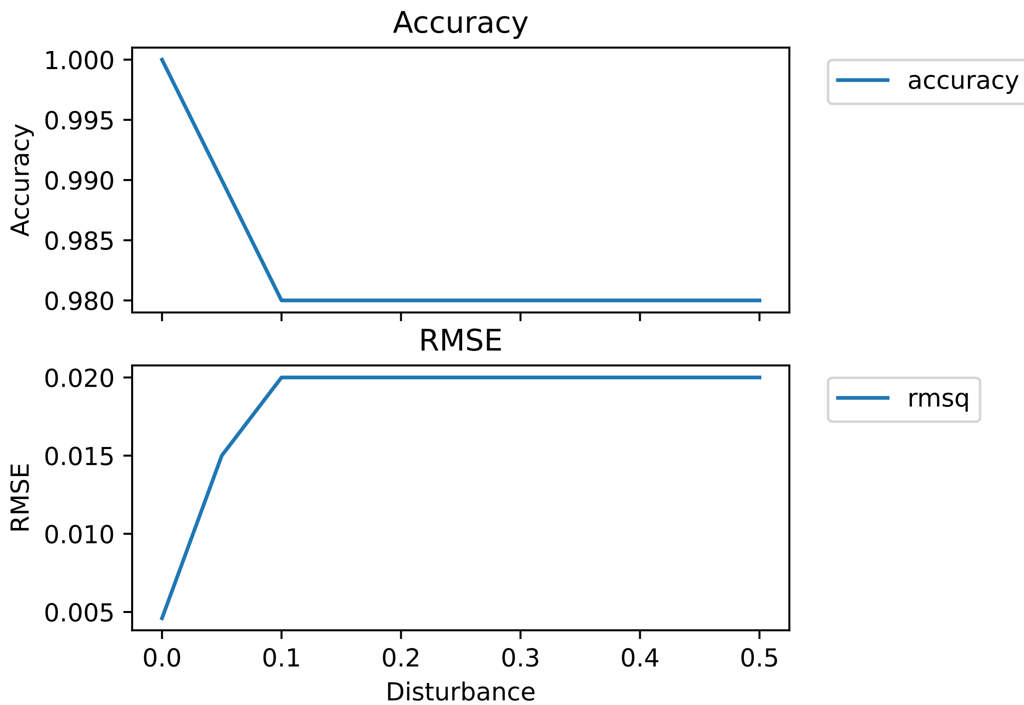


Figure 6.1: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in GroteMarkt network and simulation

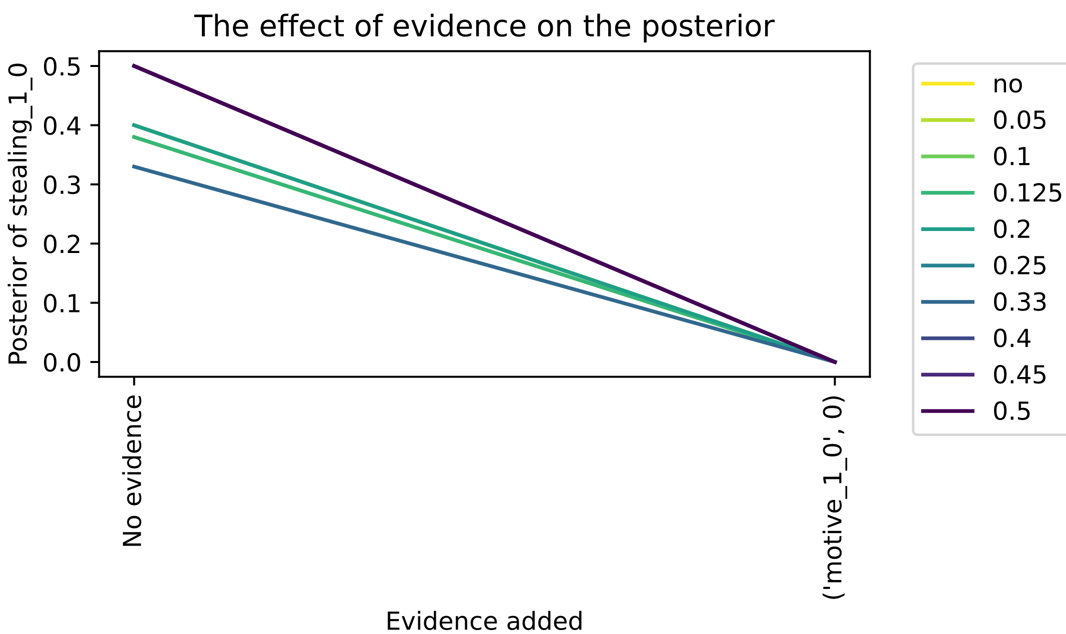


Figure 6.2: Posterior

6.2.2 Methods

6.2.3 Results

6.2.4 Discussion

6.3 Experiment 2: Swapping out the maps.

6.3.1 Introduction

What happens if we have the exact same agent logic but we place them in a different spatial configuration? Instead of the Grote Markt, I now put some other part of Groningen in the simulation. We have the same nodes in the network. The only thing that's different is the underlying map.

I selected 5 different parts of Groningen, screenshotted then, converted them to maps like before, and then let the agents loose in them to rob each other. Then I also made one part just completely empty, and one map random.

6.3.2 Methods

6.3.3 maps

Image maps compilation here.

6.3.4 Results

6.3.5 Discussion

Implications of this is that we need to condition explicitly on maps for our networks to work, because it does meaningfully change the probabilities that we find, and there's no way to predict how the map that we're using affects the probabilities. This has implications for the real world, because it means that. ugh. we can't depend on some generic "probability of getting robbed", we need to condition on spatial conditions/background world assumptions.

And these are not even very good maps - agents can either go somewhere, or not, and can see through buildings. Affordances in the real world are very different [1]. So it's likely that the actual probabilities in the real world are even worse...

[1] parkour!

(a) Grote Markt.

(b) Selwerd.

(c) South-center

Figure 6.3: Maps.

Figure 6.4: Accuracy (100% to 0%) and Root Mean Square error (1 - 0) for rounding to different intervals in GroteMarkt network and simulation

Figure 6.5: Posterior

6.4 Experiment 3: Reporters, random variables and the reference class problem

6.4.1 Introduction

A reporter is a random variable. A random variable maps a sample space to value. Let's make the random variable problematic. What sort of events are we talking about in our sample space? There can be many instantiations of random variables from one natural language string. Here I don't want to talk about the reference class, which is the problem of selecting which sample space we're actually interested in. Instead, it's the awareness that a natural language string (like a node name) by itself does not offer sufficient information to know what events we are actually interested in. We need to select a subset of all the events in the world, that we're actually interested in, and this is underspecified in most bayesian networks.

Some simple & clear problems

1. Vlek thesis "often".

2. Fenton simonshaven.

What counts as a relevant event?

For some things its not a problem because in those cases its clear what exactly we're measuring - think DNA evidence, forensic whatever, the node names there are not just natural language strings but they're RVs, a mathematical beast, with an associated method for selecting which events, and the method of knowing whether they're true. We can argue about reference classes and Pardo's barns and Nigerian Smuggler's all day long, but there we have actually established our terms. Here we're talking the step before that. Making the class of interesting events & the way of knowing that they're true, explicit. They just don't do that in the old research!!! Why not!!!

This has deep implications because if we want BNs to be accepted, everyone relevant in the discussion has to agree exactly on 1) what we're using the nodes 'natural language' to mean in RV language, 2) how we're measuring whether an RV is true or not, and then 3) the probability of the RV. I agree with Fenton that subjectivity or slight imprecision in the CPTs in part 3) is ok, fine, as long as everyone knows which parts are subjective. But the greater problems (or at least implicit problems, are 1) and 2), and those need to be made explicit as well.

6.4.2 Methods

I'm going to create many reporters that can all point back to the same natural language meaning. The easiest version is with something inherently subjective like 'near', or different definitions of ‘motive’.

6.4.3 Results

6.4.4 Discussion

6.5 Experiment 4: Investigating the island prior - Fenton.

6.5.1 Introduction

6.5.2 Methods

6.5.3 Results

6.5.4 Discussion

6.6 Take aways - Conclusion

Chapter 7

Conclusion

7.1 Good things

We can have reasonable accuracy and rms in the bayesian networks in the simulation. This is due to reporters.

7.2 Fundamental Problems with Bayesian Networks

There are some fundamental problems with BNs as used for scenario-like whole legal Bayesian Networks.

Robustness is not the problem, even if we round to arbitrary intervals, losing a lot of precision, we have networks with a reasonable accuracy/rms error, that generally reach the right (enough) conclusion and respond correctly to evidence. Granularity is also whatever. The real problem is with the translation from human language to mathematical language. A random variable maps a sample world to a value, according to a certain procedure that is described in natural language. This means that for every node in the BN, we need to describe a procedure through which we can know whether or not it is true. This should be explicit, and everyone in court should agree with this - there can be no 'personal interpretation' of this, no 'probably close enough to count', because then we have shifted from one RV to a different one. This is not necessarily problematic, perhaps the robustness of BNs does not just hold for estimating a probability for a given reporter, but also holds for estimating a probability over an estimated set of reporters. However, we have to make this explicit - if we change reporters (RVs) without communicating about it, we're in trouble.

However, in court this would imply that we would have a discussion about every measuring procedure for every node of every BN, and there would be no 'idiom database', because as soom as we try to app

7.2.1 Reporters, Reference Classes and Random Variables.

Fundamentally, the problem is that the nodes in Bayesian Networks have a specific meaning. They are not pragmatic/dialogical/argumentation sentences, but have intricate statistical meaning [1] They are random variables, and a random variable implies an observation procedure (a mapping from a world state to a number). This means, that a node implies that we know how to measure if it is true or not in the real world.

In our simulation, this is really not a problem. We have an observation procedure: if a certain state occurs, we have the reporter in the same place as when the state change occurs, and the reporter reports exactly and only that. In experiment XXX we saw what happens when this goes wrong: if we have an imprecise/contradictory/smaller/larger reporters, we see that the probabilities can change a lot, might even change the structure of the network.

Hence, BNs might work for subsections of reality that have a clear observation procedure (eg reasoning with DNA evidence or other forensic stuff). We know exactly what it means for the node to be true (eg: we know the measurement associated with the random variable). However, for many of the node events that we encounter in scenario BNs/argumentation BNs, we do not know how we are determining that the node is true or not. So either we spell out exactly when a node is true or not, and by exactly I mean exactly, and we lose all generality/chance at DBs. Or we do not spell it out, interpret BN nodes not as random variables but as some sort of fuzzy conditional logic operation, but then that's fine, but we're not really building Bayesian Networks, we're doing something else.

Experiment outcomes: best possible case would be that the structure of the BN becomes different (not just the probabilities) based on different reporters used.

- I want one random variable that is a 'wide' interpretation: a combination of all possible reporters for some thing. - I want some random variables that are the most narrow possible interpretation. - alternative.

This is not the same as the problem of the reference class, but it is related. The problem of the reference class is the discussion which definition/reporter is appropriate for a given situation. This problem is the step before that, which is - how do we specify what reference class we are talking about? This is not described in the scenario-like BNs I've discussed in my introduction. The forensic BNs come closer, because there we at least know what method is used to determine if an event happened or not...

[1] secret fourth good sherlock episode.

7.2.2 Dependence on implicit parameters and background knowledge.

Put the exact same agents with the same behavioural loops on a different map, and the probabilities for events change (see XXX). This is not a problem - this means that environment/world knowledge/context, which is not a node explicitly in the Bayesian Network, affects the output of networks. This can be remedied in a sense, by adding a context node. So what's the problem? That if we see a BN and we do not know exactly what the context was in which it was created, we cannot assume that it generalises to a different environment. We cannot take parts of that network (with probabilities) and put it in a different one, because implicitly it is only defined over the environment for which it was originally created.

The networks are also dependent on implicit parameters, like the radius of the vision of the camera. It's neat that changing this parameter will influence the evidential strength of it. Not necessarily a flaw but it does mean that its like. We have the nodes in the networks are we need to specify exactly what we mean, over all relevant parameters. And we don't know what these parameters are.

7.2.3 Private knowledge

Here we run into a paradox: in our simulation we know exactly when something is true, even when that thing is private knowledge. If we leave out the private knowledge (drop from our table, basically), we have a problem and the accuracy of the network drops. This means that we can only generate a BN that uses private knowledge if we have all knowledge, requiring everyone in the process to help along. Feasible? No.

7.3 Conclusion

Bayesian Networks are a good tool if you know exactly what you want to investigate and have methods for it as well. This is the case in simulation, obviously.

7.4 Future Work

Testing or generating Bayesian Network idioms from simulations. The dream is to get "plug and play" Bayesian Network idioms - preconnected structures (perhaps even with some probabilities attached) that you can add evidence to and adapt and combine if necessary. Using simulations, we can test the granularity of these possible idioms, to simulate a crime at larger and smaller resolution (more or fewer events) to see how well the idioms can capture it.

Investigate the effect of granularity, expecially in relation to the complexity of the network. Size, num of arrows, etc.