Evaluating Bayesian Networks from Agent-Based Simulations

Ludi van Leeuwen

July 15, 2022

Abstract

There are probabilistic methods for reasoning with evidence in court cases. Bayesian Networks have been suggested as one of these methods, but we do not know how suitable they would be in the domain of law, which is not traditionally a statistical field. This work investigates Bayesian Networks for criminal cases by creating multi-agent simulations of crimes, collecting frequency data on these simulations, using the K2 algorithm to generate Bayesian Networks from this data, and then evaluating the generated Bayesian Networks on structural, performance and human factor criteria. This is done for 3 simple simulations: one non-spatial simulation of a murder, one spatial simulation of a home-robbery with evidence, and one spatial simulation of a street robbery in a non-trivial environment.

We find that the Bayesian Networks are generally accurate and perform well even if we do not know the exact probabilities as generated by the simulation. However, it is implausible that human modellers could replicate the cpt's of the networks. Apart from that, private knowledge, conditioning on implicit factors and the effects of non-ideal operationalisation are factors that need to be solved before Bayesian Networks can be responsibly applied in court.

Contents

Chapter 1

Introduction

When we find evidence for a hypothesis that we have held in the back of our minds, our belief in the hypothesis increases. 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 its connection to hypotheses is as well - how can we be sure that some evidence supports some hypothesis? 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 hypotheses?

We have a way of expressing how we should use evidence to update our beliefs in hypotheses. We can do this with the use of probabilities. When we reason probabilistically with evidence, we want to use the gold standard - Bayes Law. Let's say that we find some piece of evidence e, we want to know how our finding e affects our belief in some hypothesis h. We can express how much we believe h given e, using Bayes Law:

$$P(h|e) = \frac{P(e|h) \cdot P(h)}{P(e)}$$

This is the simplest form of Bayesian reasoning, one piece of evidence, and one hypothesis. But real life is not so simple, and we are constantly reasoning with many pieces of evidence. Small hypotheses can serve as stepping stones to greater hypotheses. One way of handling the complexity of interacting pieces of evidence and hypotheses is by using Bayesian Networks.

A Bayesian Network is a directed, acyclic graph that represents the joint probability distribution over a set of events (Pearl, 1988). BNs have nodes and arcs. Formally, the nodes are random variables that represent events. In case of our Bayesian Networks, these random variables represent hypotheses and pieces of evidence. An arc represent a conditional relationships between two nodes. Conditional probability is expressed in the conditional probability table (cpt) of every node. A node that is not conditionally dependent on any other node (has no incoming arcs) is independent.

An example of a Bayesian Network can be seen in Figure 1.1. It expresses a simple reasoning step: how should our belief in the hypothesis that our train will be late in Groningen, change once we learn about the train station renovations in Zwolle?

In the figure, the node 'train_on_time_in_groningen' is a hypothesis. Nodes with one or more incoming arcs are conditionally dependent on their parent node(s). In our example network, the node 'station_renovation_zwolle' is conditionally dependent on its parent. The links between the nodes are links of conditional probability, and do not correspond to real-world causality, although they are sometimes interpreted as being causal (Dawid, 2008).

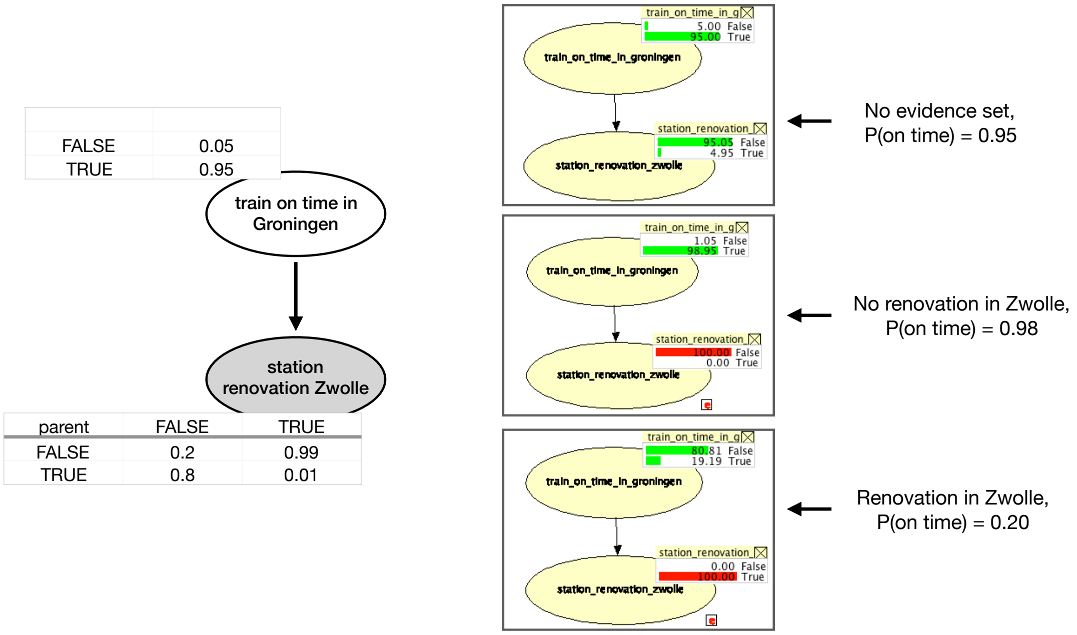


Figure 1.1: An example of a Bayesian Network, with and without evidence set.

We can reason with Bayesian Networks by setting evidence on one or more nodes that we can observe - our evidence. This means that we say whether some observable node is true or false, and use that information to update the network. We can now find a new posterior probability for a node that we cannot observe directly (yet). In Figure 1.1, this means that when we learn that there are station renovations going on at Zwolle, the our belief that our train will be late in Groningen increases from a probability of 0.01, to a probability of 0.8.

Using Bayesian Networks, we can make explicit how we think that we should reason about hypotheses given pieces of evidence. In our toy example, this is not very consequential, it is a very simple network. However, Bayesian reasoning and Bayesian Networks specifically, can be applied to many domains. One of which is AI and law, specifically, Bayesian Networks might be tools that could be useful in court cases, by organising evidence and hypotheses. A Bayesian Network might make explicit how reasoners should update their beliefs based on the evidence, which might increase transparency and fairness. In the ideal case, a Bayesian Network might stop a judge from making a reasoning error!

However, creating Bayesian Networks is very complex, especially in domains where probabilistic data is sparse. We also do not have a good idea if our methods for creating Bayesian Networks in criminal situations work, since it is difficult to test these networks empirically - you cannot ask a murderer to murder again, for the statistics. This is why we need a grounding to investigate whether our methods for building Bayesian Networks actually work.

This is why this project proposes to ground Bayesian Networks for crimes in multi-agent simulations of crimes. We can model crime cases using simulations to a level of realism we desire. We can see what probabilities emerge out of these simulations by running them multiple times, and see if we can capture this probabilistic information into Bayesian Networks. We can collect exactly the frequency information that we need. This means that we can focus on assessing the Bayesian Networks, since we know what frequency information they should replicate.

Our Research Questions:

- **Can we create Bayesian Networks that correctly reflect multi-agent simulations of crimes?**
- **If we can, under what assumptions do these networks function?**

Chapter 2 is an overview of the state of the art, Chapter 3 proposes a pipeline for creating simulations and automatic Bayesian Networks, and evaluation criteria that state when a Bayesian Network correctly reflects multi-agent simulations of crimes. Chapter 4, 5 and 6 are applications of this pipeline in three separate simulations: first a simple, non-spatial simulation based on (Vlek et al., 2015), a simple spatial simulation of a home-robbery, and finally a spatial simulation of a street-robbery. For each of these simulations, some simple experiments are done to investigate several problematic aspects of Bayesian Networks. These simulations are then each evaluated based on the evaluation criteria set out in Chapter 3. Finally, we draw conclusions in Chapter 7.

You can interact with the simulation in Chapter 6 on https://shielded-journey-34533.herokuapp.com.

Code for this project can be found at https://github.com/aludi/simulationTest.

Chapter 2

Background

We want to create grounded Bayesian Networks for reasoning with evidence. This can be done by creating multi-agent simulations, and building Bayesian Networks based on the events in these simulations. In this section, the relevant background is laid out on reasoning with evidence, Bayesian Networks, and multi-agent simulations.

2.1 Reasoning with evidence

We consider three main directions in the field of reasoning with evidence within law (Verheij et al., 2015). The first direction is via argumentation approaches, where hypotheses and evidence are represented as propositions that attack or support each other. The second direction is via scenario approaches, where more-or-less coherent hypotheses are combined into stories (Pennington and Hastie, 1993), (Wagenaar et al., 1993), which are supported with evidence. The third direction is via probabilistic approaches, where hypotheses and evidence are assigned probabilities, and the relation between hypothesis and evidence is represented with conditional probability. There are also be hybrids that combine or synthesise aspects of each approach, see, for example: (Bex, 2010) and (Timmer, 2016). This project does not consider the argumentative approach, and is mainly based on the work of Vlek (Vlek et al., 2015), (Vlek, 2016) and Fenton (Fenton et al., 2012), (Fenton et al., 2019). The networks in these papers describe whole criminal situations (scenarios), but use Bayesian Networks and hence have a probabilistic component.

2.2 Bayes and Bayesian Networks

We have already seen the power of Bayes's law in the introduction. It tells us how much we have to change our belief in a hypothesis once we find a piece of evidence. To do this, we need to have probabilistic or frequentist information about the likelihood ratio and the prior.

Bayesian reasoning, without Bayesian Networks, have been used by Dahlman in 'event trees', specifying different branches of combinations and their probabilities (Dahlman, 2020).

However, just using Bayes law is very difficult, because sometimes we want to condition on multiple pieces of evidence. Doing all the calculations is tedious, hence we use the computational tools called Bayesian Networks.

Bayesian Networks can represent aspects of a criminal case or can attempt a scenario-like hybrid and represent the entire case - modelling actual crime cases (J Kadane, 1996), (Fenton et al., 2019), cases from fiction (Fenton et al., 2012) or fictionalized crime cases (van Leeuwen, 2019). In situations where the network represents aspects, the nets model DNA or blood-spatter evidence, for methods see (Ronald Meester, 2021).

Bayesian Networks can be built by hand, by experts or academics, in (proprietary) software like AgenaRisk or Hugin. They can also be built by hand in PyAgrum (Ducamp et al., 2020), a free Python software package, which does not have a GUI but has everything else. In this project, PyAgrum was used. Alternatively, Bayesian Networks can be automatically constructed from large datasets - PyAgrum also offers the opportunity for that.

Bayesian Networks are a promising tool, but there are a lot of open questions:

1. The use of Bayesian Networks is not straightforward, neither for the builder or for the interpreter. From de Koeijer et al. (2020): it is complex, time-consuming, hard to explain, and, the 'repeatability [...] leaves much to be desired. Node definitions and model structures are often directed by personal habits, resulting in different models for the same problem, depending on the expert'.

2. The granularity of the network - how do we know which nodes hypotheses and pieces of evidence to include? Ideally we would model as detailed as possible, but as we increase the number of nodes, we increase the complexity, which increases the probability of mistakes, and the time spend on the network.

3. The links: how do we know which events depend on each other?

4. The numbers - there's not just subjectivity in selecting the nodes, and drawing the links, but the probabilities that we have to fill in into the cpt themselves are the most obvious stumbling block. We can identify that there has to be some sort of correlation between 'smoke' and 'fire', but how to express this in numbers? Fenton argues that we're just making something explicit that we would otherwise have left implicit. But what are the consequences of making something explicit, but doing it wrongly? How robust is the network against imprecise or wrong frequencies? If we

use frequencies (or subjective probabilities based on frequencies), how do we decide what set of events is included when we start to count (this is the problem of the reference class, see (Allen and Pardo, 2007), (Colyvan et al., 2001)). Probabilities can be pure frequencies, or can be subjectively elicited from experts (Renooij, 2001), (Druzdzel and van der Gaag, 2000).

2.2.1 Evaluating Bayesian Networks

There is a precedent for building Bayesian Networks to model reasoning with evidence in criminal cases, although these methods have not been used in court. However, even though there are methods for building these networks, it is unclear how to determine how we can validate these networks.

In more data-rich domains, where we train Bayesian Networks on large amounts of frequency data, we can estimate the accuracy of the networks by cross-validation. We construct the network based on some subset of the data and then use the rest of the data as a test, to see whether the prediction of the network is correct, given some input (a split of 80/20 for train/test is common) (good practice Chen) This approach prevents overfitting, where the network might perform well on some subset of input, but perform worse on some other, unrepresented subset.

However, this approach towards BN validity through accuracy is implausible, given the circumstances of court cases. The court cases involve one-of-a-kind events (schum, 1982), which means that it is unlikely that we can find frequency data on most of the nodes present in our Bayesian Network. This is why the probabilities in these Bayesian Networks in court cases are subjective probabilities, either elicited from experts or estimated by the modeller (sometimes based on real data).

This does not mean that there are no methods of evaluating these Bayesian Networks, however. Evaluation of the Bayesian Networks for criminal cases as found in the existing literature confine themselves to two approaches:

1. Sensitivity analysis

   Sensitivity analysis tests the effect of small parameter changes on the outcome of the network. This is useful if the outcome of the network hinges on elicited or subjectively estimated probabilities.

2. Cumulative evidence on one set of evidence

   In this approach, the modeller selects all evidence nodes, and start to enter the evidence that was established during the trial. This results in a cumulative posterior probability on the outcome node. In this process, we can see how much each piece of evidence contributes to our overall final belief in the posterior.

However, what we see with this cumulative evidence approach, is that we are only testing one set of evidence. Both Vlek and Fenton only test the cumulative probability of the output node by turning on the evidence nodes that the court has found. However, this is a risky move, since this might lead to overfitting the probabilities in the CPTs, so that the network seems to reflect the right 'evidence strength' on the selected set of evidence, but misrepresenting other combinations of evidence. An example of this overfitting can be found in (van Leeuwen, 2019), which is biased towards the scenario that the suspect is guilty. In Figure 2.1, we see a situation where a certain set of evidence is entered into the network. The outcome of the network is, that the suspect is guilty. But the evidence that is entered into this network, would not rationally support this conclusion: there is no body, no signs of violence, no car with bloodstains, to evidence of existing conflict... The only evidence that the prosecution has, are some phone calls, and perhaps some testimony of the suspect that does not match medical reality. Even with this scant set of evidence, the Bayesian Network would decide to convict.

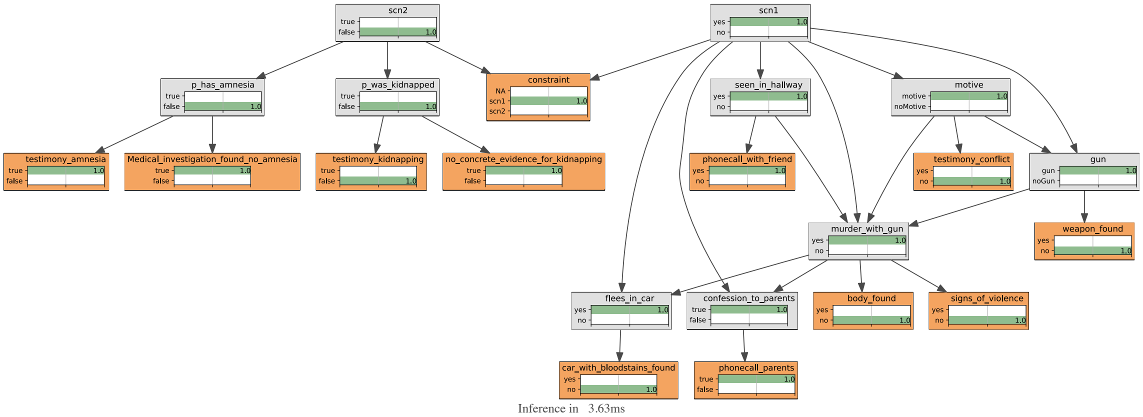


Figure 2.1: Example of subjective probability estimation resulting in overfitting

This illustration shows that it is necessary to consider all possible combinations of evidence. If we have tunnel vision and only focus on the set of evidence that is found in court, we might create a network that overfits. If we want to take the premise behind BNs seriously, we have to create a network that gives sensible results for all possible combinations of evidence valuations - and not just the valuation of the evidence that are established in court.

2.3 Agent simulations

We can create such a data-generating environment for criminal cases by the use of multi-agent simulations. In a multi-agent simulation, agents observe and interact with their environment. The environment and the agent behaviour is fully controlled by its programming and all randomness can be accounted for. This means that we know exactly with which frequencies fully-specified events occur. Running the simulation multiple times generates a lot of data, which will serve as input to automatically build a Bayesian Network from data. This means we use a multi-agent simulations as a ground for our theory-testing. In this project, the simulations will be programmed in Python using the MESA framework (Kazil et al., 2020).

Multi-agent simulations have been used to investigate the criminal domain and to test out sociological theories. By creating spatially explicit simulations, the complex interactions of agents with their locations can be represented better than in traditional models - these agent-based models can model criminal hotspots (Bosse and Gerritsen, 2008), theories of behaviour (Gerritsen, 2015), and police strategies in urban crime (Haojie Zhu, 2021). Weaknesses identified with multi-agent models for criminological research are insufficiently complex models, unclarity on the effect of temporal resolution, and lack of a systemic approach (Haojie Zhu, 2021).

2.4 Conclusion

The two main ideas are Bayesian reasoning, specifically as implemented in Bayesian Networks, and computational simulations of agents. The simulation is supposed to be the grounding for the Bayesian Networks.

But why do we want to use Bayesian Networks in the first place? In the best possible case, a Bayesian Network would help us to make the correct reasoning steps, telling us exactly how to weigh each piece of evidence in the grand scale of the simulated crime. This is a normative approach - it's telling us how to reason, but it is not (and not meant to be) an empirical approach. Bayesian networks are not a reflection of 'how people actually reason' - If you open up our minds, you won't see Bayesian Networks in there. One of the problems for normative models in decision making (Colyvan, 2013), is that we can only test how well our normative models work, if we already know what a good outcome would be. In a sense, we're doing experimental model testing for normative Bayesian Networks in this project.

Chapter 3

Method

This chapter lays out the general method for creating agent-based simulations with automatic Bayesian Networks, as well as how to evaluate these Bayesian Networks.

We can consider an approach in four stages. First, we need a scenario, or set of alternative scenarios, that are to be modelled. These scenarios are written description of crime scenarios that contain the hypothetical events and evidence for these events, as postulated by the prosecutor and defence.

Second, based on this written description of the scenario(s) to be modelled, an agent-based simulation is created. In the simulation, all events of the scenario need to represented. The simulation can be as granular as desired. The simulation can be run multiple times, so that we can gain an understanding of possible underlying frequency information on the events in the scenario. The frequencies by which events occur in the simulation are collected.

Third, the collected frequencies are then used to automatically build a Bayesian Network on the simulation, using the K2 algorithm. Finally, the generated Bayesian Network is then evaluated based the criteria set out at the end of this chapter.

This four-step process will be explained in this chapter, and illustrated with a running example of a stabbing.

3.1 Scenario

We start our process with one or more written scenarios. These scenarios can, in the first instance, be obtained from (abridged) court case descriptions. The scenarios should contain all and only those hypothesised events, and the evidence for such events, that are relevant to the case. Both the prosecution and the defence should be able to select relevant events and their evidence.

Example 1 *Here we will introduce our running example, loosely based on the first scenario in (Vlek et al., 2015): Mark and Jane live together. One day, a neighbour overhears that they've started to fight in the kitchen. Jane grabs a knife, and stabs Mark. Mark dies of the stab wounds, as confirmed by the forensic scientist at the scene. Jane's fingerprints are on the knife.*

3.2 Simulation

We need a way to transform the written text descriptions of the events in the scenario to observable events in the simulation. First we identify all relevant actors, objects and places in the scenario. Then, we divide all all events into three classes: either an event is 'evidence', a 'hypothesis', or an ultimate 'outcome'. The 'outcome' is what we are ultimately interested in, it represents the central fact of the case. Apart from it being the node of ultimate interest, it otherwise functions the same as a 'hypothesis' node.

Then we select all relevant events from the written description and attempt to operationalise them in the simulation. This means that we create a way to measure if the events take place within the simulation. This is done by means of 'reporters', or random variables. These reporters report whether a given event has taken place within the simulation.

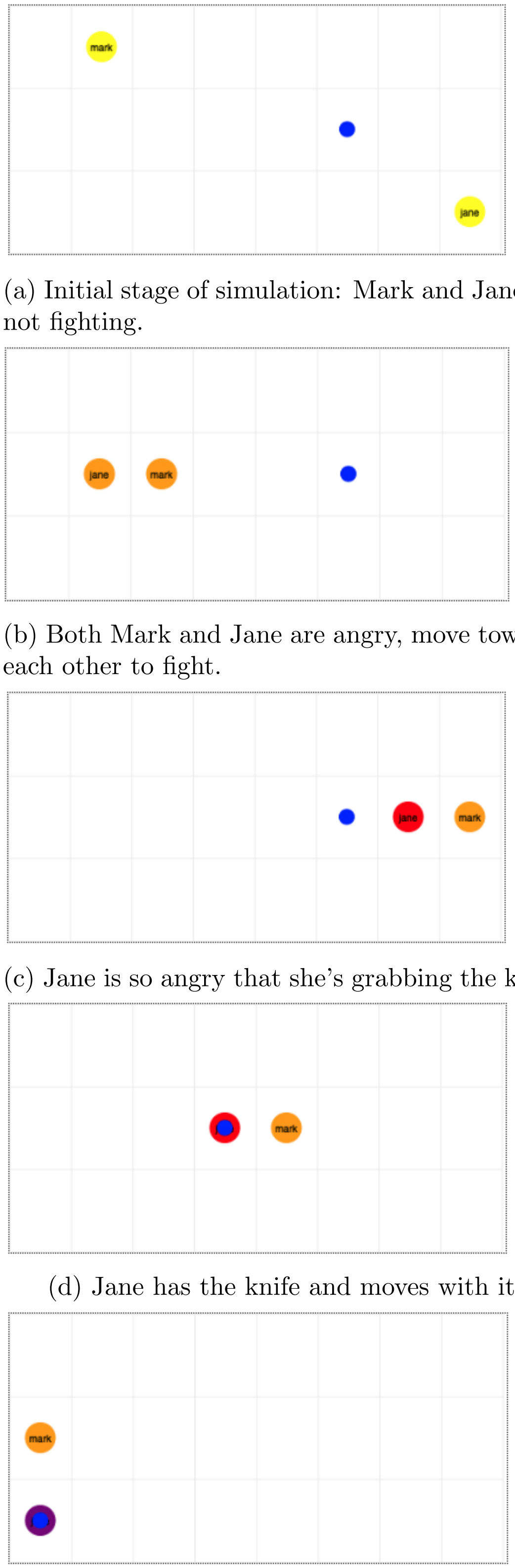
The process of operationalisation is very subjective and depends entirely on the modeller's assumptions and their time, knowledge or skill constraints. The behaviour of agents, as reflections of actual people, can be modelled in a way that reflect real sociological theories of behaviour (such as in Gerritsen (2015)), or with simple rules and a random number generator (such as here). The environment of the simulation can reflect a real-world geography with different affordances, or can be simplified. How a given event is operationalised is essential for the validity of the network: if it is unclear, or disputed, when an event is true in a simulation, the network should not be used.

Example 2 *We have modelled Jane and Mark as two agents in a simulation (Figure* **??***), with as relevant object a knife (blue dot). The only 'psychological' aspect that these agents have, is an anger level, which has 4 escalating states = {not angry, fighting, grabbing knife, stabbing}. Jane is quicker to increase to the next anger level.*

jane_and_mark_fight *Both Mark and Jane have an anger level that is higher than 'not angry'*

jane_has_knife *Jane has an anger level of 'grabbing knife' and moved to the location of the knife*

jane_stabs_mark_with_knife *Jane has an anger level of 'stabbing', has the knife, and moved to Mark's position*



(a) Initial stage of simulation: Mark and Jane are not fighting.

(b) Both Mark and Jane are angry, move towards each other to fight.

(c) Jane is so angry that she's grabbing the knife.

(d) Jane has the knife and moves with it.

(e) Jane's anger increases by a level, and she stabs Mark.

Figure 3.1: Simulation stages.

mark_dies *Jane has stabbed Mark, and if a random number generator (uniform, 0, 100) produces a number > 60*

E_neighbour *Either Mark and Jane have an anger level that is higher than 'not angry'*

E_prints *True when Jane has knife*

E_stab_wounds *True when Jane stabs Mark*

E_forensic *True when Mark dies*

Then, the operationalisation itself: In the simulation, certain events can be brought about. In our example, this could be the events of 'jane_stabs_mark', or 'E_forensic'. We need a way to observe these states: this is where reporters come in. A reporter is a random variable that is reports the outcome of a relevant event 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) (for a further explanation of Random Variables, see Chapter 8). In short, a random variable maps an event (e) to a truth value:

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

Over which events we define reporters, depends on which events we deem relevant. We could, in theory, create a combinatorial explosive number of reporters for any possible event, like $jane\_in\_position\_(0, 0)$, $jane\_in\_position\_(1, 0)$, etc, but pragmatically, these very granular reporters will not help us in modelling the case. This means that we need to find a balance between over-specification and under-specification.

A reporter's value is 0 by default, but maps an event to 1 whenever it happens in the simulation. When we run the simulation in our example, we expect to see that, whenever Jane and Mark are fighting, R : 'jane_and_mark_fight' $\to 1$. At a later time-step, if Jane stabs Mark, we would see: R : 'jane_stabs_mark' $\to 1$.

No matter how many reporters we define, we can combine all reporters at the end of one run of the simulation into a global state G.

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

or, for n reporters:

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

A global state that would represent that every one of the postulated events and evidence happened, except that Mark did not die of the stabbing, and that there was no forensic scientist to determine that he died of the stab wounds, would be represented as (reporters in the same order as in the listing above):

$$1, 1, 1, 0, 1, 1, 1, 0$$

Then, we collect these global states over the number of runs that we do for each experiment, which results in 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

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. The reporters in the simulation are random variables. These reporters become the nodes in the Bayesian Network.

The Bayesian Network is generated automatically, using the automated BN learner method as implemented in pyAgrum. There are several learners implemented in pyAgrum.[1] The algorithm used in this experiment to structure the Bayesian Network from the simulation data is the K2 algorithm (Cooper and Herskovits, 1992).

The K2 algorithm is a greedy search algorithm that attempts to maximise the posterior probability of the network by correctly connecting parent nodes to child nodes. It takes an ordering on the nodes as input. The first node in the ordering X_0 does not have any parents. For two nodes X_i and X_j, X_j cannot be the parent node of X_i if $j > i$: a node can only have a parent that is earlier in the ordering. The algorithm processes each node X_i by adding possible parent nodes to X_i (as constrained by the ordering), and maximising the score of the network. The algorithm stops if there are no parents to add, no parents improve the score, or the node has reached the maximal number of parents (Chen et al., 2008).

Due to this ordering constraint, the K2 algorithm is efficient. However, finding the ordering of the nodes as input for the network is not trivial. The ordering should be meaningful, to reflect causal or temporal information present in the domain. Through our simulation, we can find the temporal ordering of the nodes by collecting the temporal order in which hypothesis events occur. The evidence nodes are added at the end, to ensure that evidence nodes are never parents to hypothesis nodes.

[1] An introduction to structure learning using PyAgrum: http://webia.lip6.fr/~phw/aGrUM/docs/last/notebooks/structuralLearning.ipynb.html

Example 3 *The temporal ordering of events in the simulation would be:*

{'E_neighbour', 'jane_and_mark_fight', 'jane_has_knife', 'E_prints', 'jane_stabs_mark_with_knife', 'E_stab_wounds', 'mark_dies', 'E_forensic' }

The ordering we want to give to K2 to save the temporal structure and the evidence-idiom Fenton et al. (2012):

{ 'jane_and_mark_fight', 'jane_has_knife', 'jane_stabs_mark_with_knife', 'mark_dies', 'E_neighbour', 'E_prints', 'E_stab_wounds', 'E_forensic' }

The ordering is calculated as follows: at every run, we note down which hypothesis events happen in which order. We only note down the order, not the time-step at which an event occurs. We count how often each set of events occurs. We select the set of events that contains the most events, as this is likely the set that represents the entire 'causal' chain, or an entire scenario. If there are two sets of events of equal length, we pick the one that occurs most frequently. We add this set first to the ordering. We then look at the number of hypothesis nodes we have left, and add the largest set we can, until we have added all the hypothesis nodes to the ordering. Then we add the evidence nodes to the ordering.

For example, let's say we have 4 hypothesis nodes, $X_1, ...X_4$. In Run 1 we find X_2, X_3, X_4, in Run 2 we find X_1, in Run 3 X_2, X_3, X_4 again, but Run 4 is X_1, X_2, X_3. We select the longest sets first, and since X_2, X_3, X_4 occurs more often than the other one, we add it to the ordering first. The ordering is now X_2, X_3, X_4. We have 4 hypothesis nodes, so we need to add a set of length 1, which is X_1, which results in an ordering of X_2, X_3, X_4, X_1 that we add evidence to, and apply to the K2 algorithm.

Example 4 *We give the collection of global states O from our example to K2, as well as the temporal ordering as laid out in the previous example. This results in the a Bayesian Network that is shown in Figure 3.2.*

After the network is generated, we need to manually change a thing about it. The algorithm does not know how to handle impossible states, which are cases where combinations of valuations for parent nodes are impossible. For example, in Figure 3.2, the value of 'E_forensic'. 'E_forensic' has two parents: 'mark_dies', as well as 'E_neighbour'. In the simulation, it can never be the case that Mark dies, but the neighbour does not hear and report on fighting, or the combination of events $`mark\_dies' \rightarrow 1 \wedge `E\_neighbour' \rightarrow 1$. However, the event for the thief stealing the object, can have both 'motive' and 'knowing about object' as parent nodes. The algorithm does not know how to deal with these situations and the resulting network will crash. This is why the probability for 'stealing' given the impossible combination of the parent state values, will be manually set to 0.

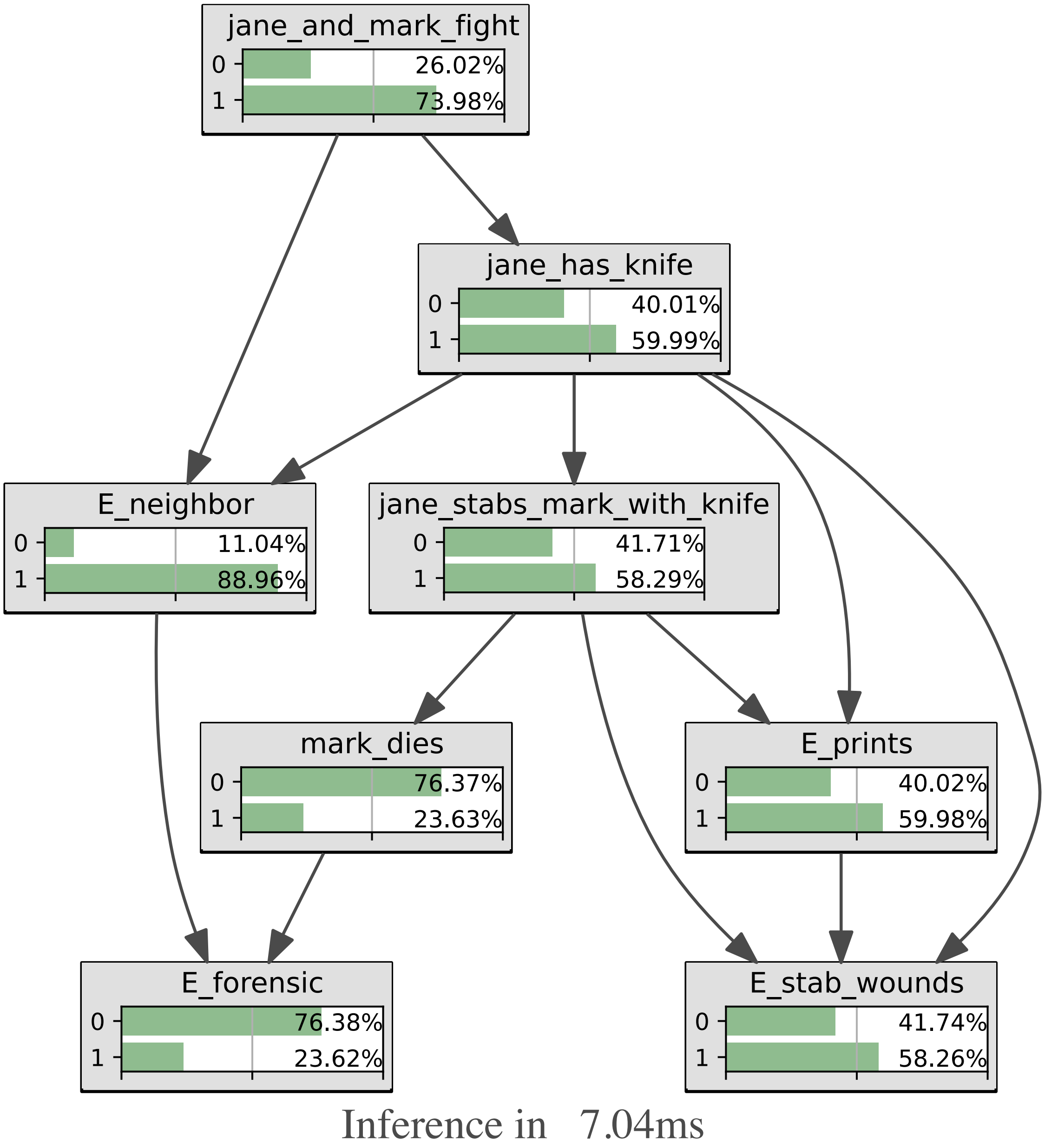


Figure 3.2: Example network.

3.4 Evaluation

Now we know that we can build Bayesian Networks automatically using the K2 algorithm based on data we generated in our simulation, we need to evaluate different aspects of the Bayesian Network: numerical, structural and performance criteria.

3.4.1 Numbers

1. **The conditional frequencies in the BN correspond to the conditional frequencies in the simulation**

   We take the collection of global states as the output of the simulation. For every event in the global state, we calculate the base-frequency at which they occur, so if event A occurs 40 times and $\neg A$ occurs 60 times in 100 runs, we find that the frequency of $A = 0.4$. Then, we compare these frequencies with the prior probabilities of the events in the BN.

   Given below (Table 3.1) are the correspondence between the frequencies in the simulation and in the BN. There are differences between the probabilities in the BN and frequencies in the simulation, but these are smaller than 0.01.

Conclusion	Frequency P(event)	BN P(event)
jane_and_mark_fight	0.74	0.7398
jane_has_knife	0.6	0.5999
jane_stabs_mark_with_knife	0.583	0.5829
mark_dies	0.236	0.2363
E_neighbour	0.89	0.8896
E_prints	0.6	0.5998
E_stab_wounds	0.583	0.5826
E_forensic	0.236	0.2352

   Table 3.1: Correspondences between simulation frequency and BN

2. **The values in the conditional probability tables are elicitable from human modellers**

   In real life, we do not have access to the true frequencies of events in real life. This means that we would have to discover them in some other ways. The CPT in question here, means including the parent node as determined by the K2 algorithm. Relevant questions to ask about the elicitability of CPTs of nodes in the BN are:

   (a) Could we discover the frequencies in the CPT using forensic-science methods?

(b) Could we discover the frequencies in the CPT using population statistics? [2]

If we have to answer both of these questions with 'no', then the CPTs are not elicitable at all. However, answering Question 1 or 2 with a 'yes' is not straightforward - we have to deal with the problem of the reference class.

To show what is meant by that problem, let's apply these questions to our example. We have the node 'jane_has_knife', given parent node 'jane_and_mark_fight', and we want to assign it some probability. However, we cannot find a 'population statistic' about 'jane_has_knife'. Are we going to the CBS and ask about all knife owners named Jane? This is not a meaningful population statistic. Hence, we need to generalise Jane to some class of person that we can find population statistics for. If we generalise 'Jane' to the reference class 'a random adult woman in the Netherlands', we could probably find population statistics about knife possession. However, if domestic violence is a pattern in Mark and Jane's relationship (as evidenced by the parent node 'jane_and_mark_fight'), we cannot just consider Jane to be 'a random adult woman in the Netherlands', but instead consider her 'an adult woman in a violent relationship in the Netherlands'. If we find more risk factors and learn more about Jane (high aggression, previous offences, etc.), the subset that we have to find a population statistic over gets smaller and smaller, until we have reached a subjective probability estimation. This is the case for the first question as well.

This problem is unavoidable and it is unclear how we can resolve it. This is a question for future research. However, in this idealised situation, let us always take the most generous interpretation of a node - meaning that we interpret a node as if there might be a reference class. As example, let us divide the nodes into the relevant categories (Table 3.2).

Node	Forensic science	Population Statistics
jane_and_mark_fight		x
jane_has_knife		x
jane_stabs_mark_with_knife		x
mark_dies	x	
E_neighbour		x
E_prints	x	
E_stab_wounds	x	
E_forensic	x	

Table 3.2: Theoretical elicitability

An important thing to note is that these are **not** assessments about the actual prob-

[2] Guessing.

ability values in the CPTs in the generated BN, but instead only generous theoretical assessments about the deemed possibility of elicitation. The actual probability values in the generated BN will not line up with the forensic or statistical probabilities that we could or would find in real life, since the simulation is very simplified.

3. **The BN is robust against imprecision in the conditional probability tables**

   We can reduce the precision of the values inside the CPTs to investigate the robustness of the BN (Oniśko and Druzdzel, 2013). We generate one Bayesian Network, with the data we collected from the simulation. Then, we create many new networks, with this network as a basis. We do not change the number of nodes, nor the structure of the Bayesian network. We only change the values in each node's CPT. In this case, we round the values in the CPT, such that the BN becomes increasingly less precise. The intuition behind this is, that when expert users are going to elicit the probabilities, we do not know how robust the network is against smaller and larger imprecisions in the elicited probabilities. By simulating such imprecisions, we can compare the predictive power of the more imprecise networks to the ground truth of the 'real' network.

   We round every value in the cpts c to each of

   $$i, i \in \{\text{no disturbance}, 0.01, 0.05, 0.1, 0.125, 0.2, 0.25, 0.33, 0.5\}$$

   according to

   $$floor(\frac{c}{i} + 0.5) \cdot i.$$

   We also add some imprecision into these networks - so that we never have the values of 0 and 1 in our cpts, as this blocks the propagation of evidence throughout the network. Instead, where we would round to 0 or 1, we round to $1 = \epsilon$ and ϵ, where $\epsilon = 0.01$.

3.4.2 Structural

1. **The BN represents all events of the scenario**

   We have to define relevance in a meaningful way. The scenario can be split into many different (implicit and explicit) events. The simulation should attempt to model the level of granularity as given in the scenario description, with more detail given in areas where there is conflict between reasoners.

   In our example, we modelled every fact as described by the written scenario description. All nodes in the network reflect an event that happened in the scenario. However, the rules that underlie the agent's behaviour (increasing anger, goal-directed movement) are not represented explicitly in the scenario. Using more or less complex behaviour would result in different BNs. Testing different models of criminal behaviour lies outside the scope for this project.

2. **The BN has temporal ordering of the hypothesis nodes**

   We want the BN to follow the temporal structure of the simulation and original scenario. The temporal ordering given to the K2 algorithm should ensure this - however, sometimes there are events that can happen in arbitrary order, or contradict each other with no clear way of determining which events happens first (as they happen exclusively).

   One example of contradictory events, which are not represented in the example network: the simulated 'jane_stabs_mark_with_knife', compared to the non-simulated 'mark_stabbed_himself'. Apart from the contradictory events or arbitrary temporal events, the hypothesis nodes in the example network are ordered chronologically, in the same order as in the simulation.

3. **The BN follows the evidence-idiom**

   We want to reason from evidence to hypotheses, or, we can observe evidence in real life (or in the simulation), and from there, infer the posterior probability of its parent-hypothesis node. We do not want to do this the other way around, setting a value in a hypothesis node, and looking at the resulting posterior probability of the parent-evidence. This is why evidence must always be the child node of a hypothesis. This construction is also called the evidence idiom (Fenton et al., 2012). Evidence can be the parent of other evidence, as evidence can be dependent on each other (**?**).

   For example, in the network that was generated for our example, we see that the network fulfils the criterium: There is not a hypothesis node that has an evidence node as a parent.

4. **The BN represents multiple alternative scenarios**

   If we want to model both the scenario as postulated by the defence, and by the prosecution, and let the collective evidence speak, then we need to be able to represent multiple conflicting stories within one BN. Representing different scenarios is not just setting some hypothesis node to false, but instead, it is creating a separate hypothesis node, that contradicts the first node.

   In our example, this would look like, instead of 'jane_stabs_mark_with_knife', the defence would say that 'mark_stabbed_himself'. These alternative scenarios are not represented in the scenario, simulation or network in this chapter.

3.4.3 Predictive

1. **The BN only predicts high probability of the output node given relevant evidence truth values**

   We want to use the BN to give a probability estimation of the ultimate output node. We want the probability of the output node to be high if the given evidence supports it, and the probability of this node to be low, when the evidence does not support it. We can test this: given a certain set of evidence, vary the assigned truth values of these evidence nodes, and switch the evidence nodes on in chronological order. We can plot the results, showing a progression of the posterior given a certain set of true or false evidence. Examples of these progression plots are shown in Figure 3.3.

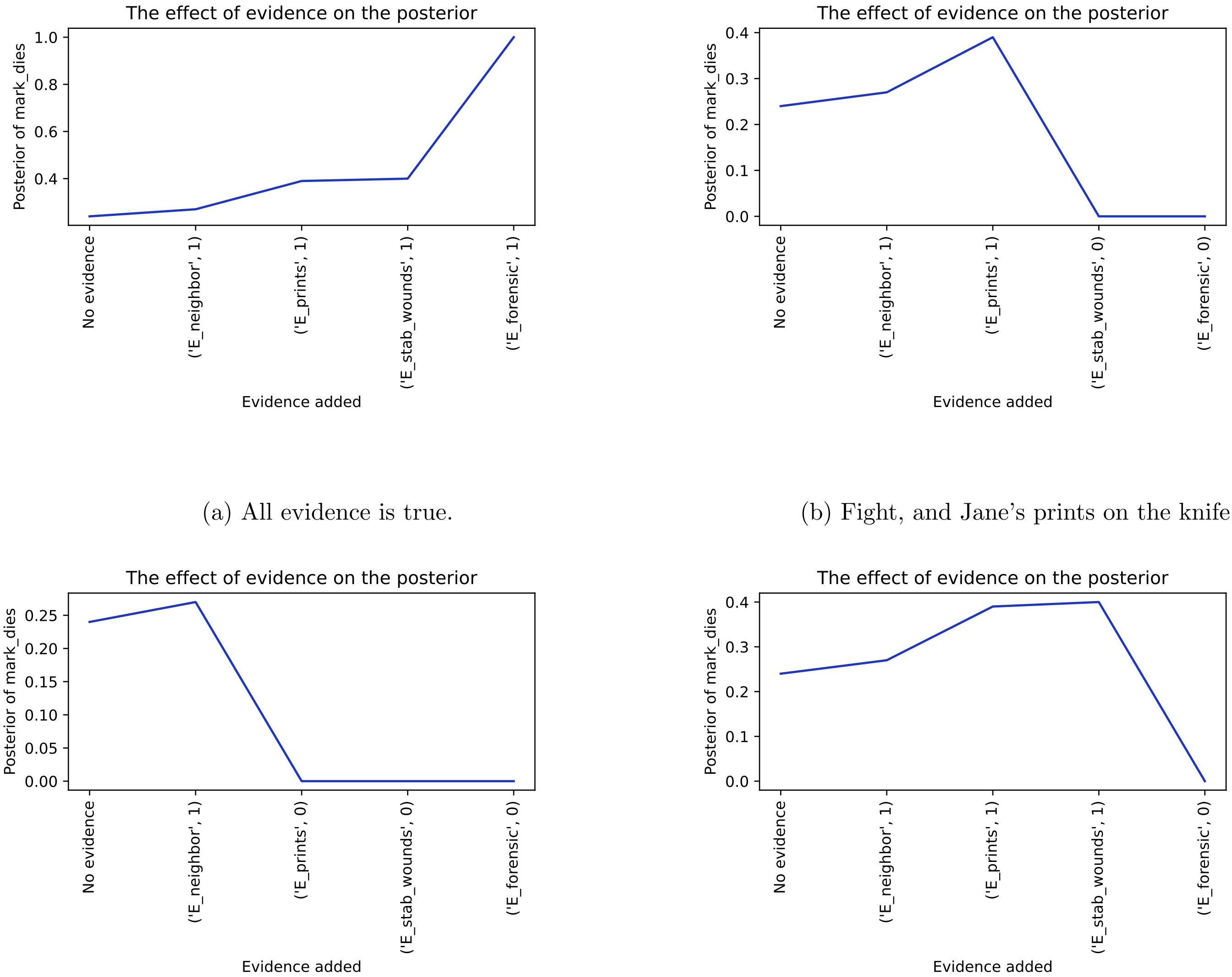
   From these plots, we can see that we can only find a convincing posterior probability for 'mark_dies' when we have the forensic report, the rest of the evidence increases the posterior probability of the node, but never higher than 0.4. This is the expected behaviour - given the operationalisation fo the 'E_forensic' evidence, we expect it (and only it) to weight heavily on the posterior for 'mark_dies'. The other evidence is weaker.

2. **We can know when the BN predicts the wrong outcome**

   We investigate all possible permutations of evidence values. There are also impossible permutations of evidence, these are states of evidence that are not allowed by constraints of the simulation, such as the forensic report stating that Mark is dead, but Mark not having stab wounds. These impossible states are not tested. Over the possible evidence states, we show for which evidence states, the network predicts the wrong value for the output node. This means, that in the simulation, given some evidence, the output would happen, but in the BN, given that same set of evidence, the output does not happen. The states for which this occurs are states where the BN is failing. In our example, this is never the case: for all possible evidence states, the network predicts the correct response in the output node.

3.5 Conclusion

In this chapter, we have explained the method for creating and evaluating Bayesian Networks from a simulation based on a written scenario.



(a) All evidence is true.

(b) Fight, and Jane's prints on the knife.

(c) Only the neighbour hears them fight.

(d) Only the death report is false.

Figure 3.3: Effect of evidence on posterior.

Chapter 4

Simulation of Grote Markt

In the previous chapter, we have laid out the process from creating a Bayesian Network from a written scenario of a case, and laid out criteria on how to evaluate that network. In this chapter, the method is applied to a simulation of a robbery at the Grote Markt, the main square of Groningen.

By placing agents in a ‘real’ spatial environment, they are constrained by geography in some manner - they cannot see through buildings, and cannot move through them. These affordances result in more interesting behaviours for the agents, and are a first step towards using this approach to less abstracted societal issues.

4.1 Scenario

There are two agents walking around the area of Grote Markt. One of the agents is old and carries a valuable object. The other agent is young, and might potentially steal the object. If the potential thief sees the potential victim, it decides whether the potential victim is vulnerable enough to steal from, and the object is valuable enough. If both of these conditions are fulfilled, the agent becomes a potential thief, and now has a motive to steal from the old agent. The agent will attempt to sneak up on the victim, and steal the object from the old agent. At some point, the old agent realises that their valuable object is gone. This is the first scenario. In the second scenario, the old agent simply dropped the valuable object, and after a while notices that it is gone. As evidence, we have a psychological report that estimates whether the young agent is capable of the crime, we might have video footage of the thief stealing the object, and we have the fact that the potential thief shows up on camera.

Additionally, outside the described scenario, we also have access to the ‘mental’ state of the potential thief, so we know whether they actually consider the old agent vulnerable, and whether they find the object itself valuable. We also know if the thief is actually intending to sneak up on the other agent.

4.2 Simulation

We extract the relevant agents from the scenario, we need two of them: a potential thief, and a potential victim. Every agent has an age, to determine whether they are vulnerable or not - old people are considered more vulnerable. Every agent also has an object of a certain value, the thief's object has a value of -1, and the other agent's object has a value of 1000, to make it a tempting target. An agent decides if it wants to steal something by making a very simple risk-calculation based on their risk threshold: if the object is more expensive than their risk threshold, they will attempt to steal it (contributing to 'motive'). Every agent also has a goal state, this is the location at the edge of the map. When they enter their goal state, they are essentially removed from the simulation, as they leave the relevant area. Every simulation was run for 100 timesteps, or until both agents are in their goal states. The simulation itself was ran 500 times. The behaviour of the agents is shown in Figure 4.1. Agents are placed randomly on the map.

The operationalisation of the random variables is described below. The output node is the node 'stealing_1_0':

seen_1_0 if the victim agent is in the line of sight of the thief agent.

know_valuable_1_0 if the value is higher than the risk threshold.

know_vulnerable_1_0 if the victim's age is older than the thief's threshold age.

motive_1_0 if the object is more valuable than the risk threshold, the victim's age is older than the thief's threshold age, and the thief is not already stealing from someone else.

sneak_1_0 if the thief has targeted the victim (has a motive) , but is not yet in the same position.

stealing_1_0 if the thief has targeted the victim, the object's value is greater than the risk threshold, and the thief's position is the same as the victim's position.

object_dropped_accidentally_0 at every epoch, there is a 1/500 probability that the victim agent will drop the object by accident.

E_vulnerable_1_0 the thief decides that the target is vulnerable (know_vulnerable_1_0 is true).

E_valuable_1_0 the thief decides that the object is valuable (know_valuable_1_0 is true).

E_sneak_1_0 the agent sneaks up on the target (sneak_1_0 is true).

E_camera_1 the thief is seen on any one of the cameras.

E_psych_report_1_0 if the thief has a motive, we draw an estimated risk threshold and an estimated age threshold from two normal distributions (mean = victim age, sd = 20), (mean = value good, sd = 100). If the victim is older than the thief's minimal age threshold, and the object more valuable, we estimate that the psych profile states that the victim fits the thief's profile, else not.

E_camera_1 the thief is seen on any one of the cameras.

E_object_gone_0 if the object is dropped accidentally, or if the object has been stolen.

E_camera_seen_stealing_1_0 if any camera sees the thief during the state in which it is stealing.

The environment for the agents was created by converting a map image into an agent-readable world. This was done by writing a method to convert screenshots of maps into an agent-readable environment. The maps were screenshotted from http://maps.stamen.com/terrain/#18/53.21618/6.57225 and converted into greyscale. Then, they were transformed into a grid of a given size. The average greyscale value of each cell in the grid was taken and coded as either 'accessible' or 'non-accessible'. On the greyscale map, the color of the buildings was in the range of (189, 199) - cells within this range were coded as 'inaccessible', since agents cannot walk through buildings. All other cells were 'accessible'. This resulted in a map shared by all agents that constrained their movements. There are 5 cameras placed randomly on the 'accessible' cells on the map, each with a visual radius of 8. Additionally, we used this map to calculate the sight lines of both cameras and agents. An agent or a camera cannot see another agent if there is an 'inaccessible' grid cell on the sight line between the two.

4.3 Creating the Bayesian Network

The network is shown in Figure 4.3.

4.4 Evaluation

4.4.1 Numbers

1. **The conditional frequencies in the BN correspond to the conditional frequencies in the simulation**

   The frequency of events in the network (without evidence set) correspond to the frequency of events in the simulation within ±0.05, see Table 4.1.

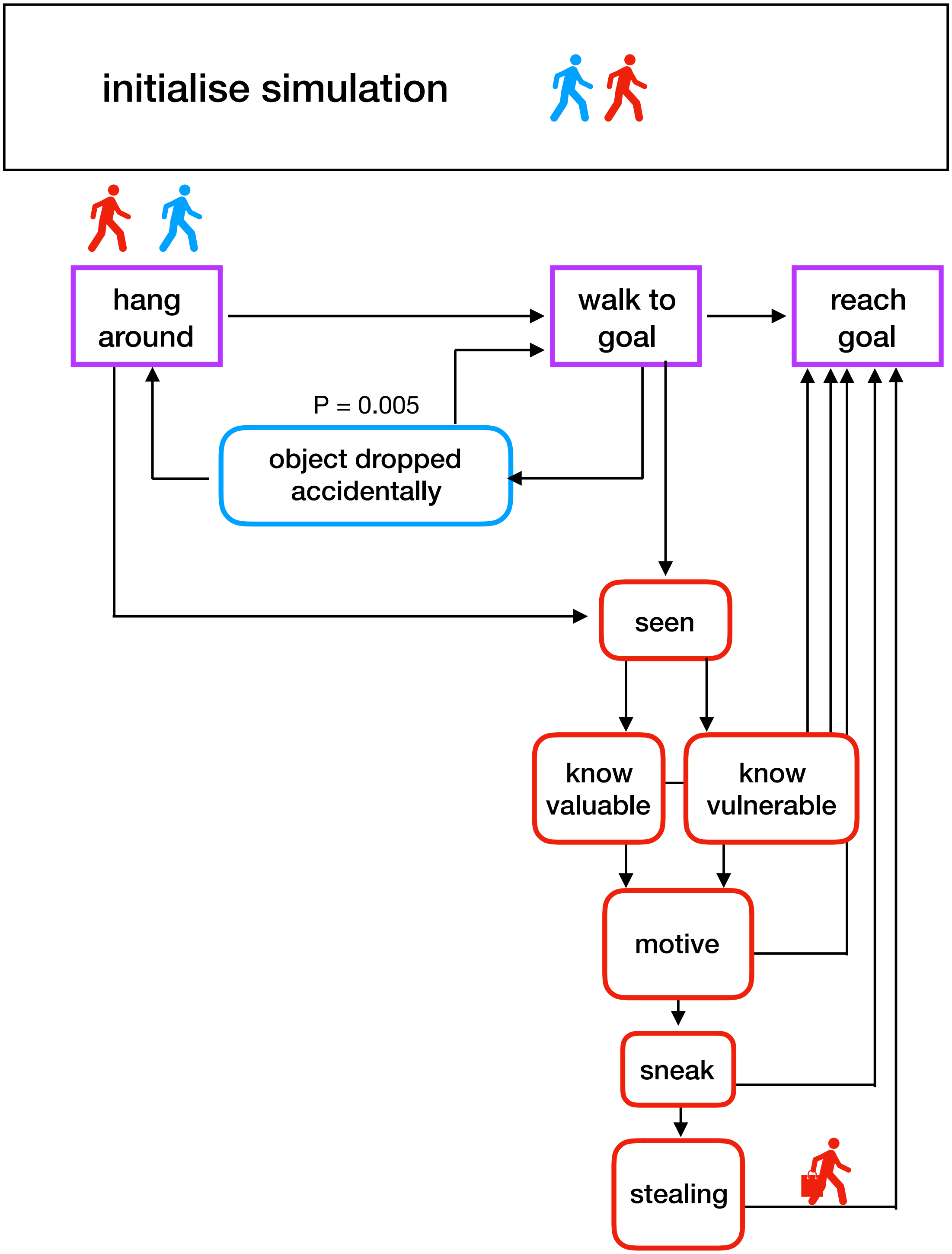
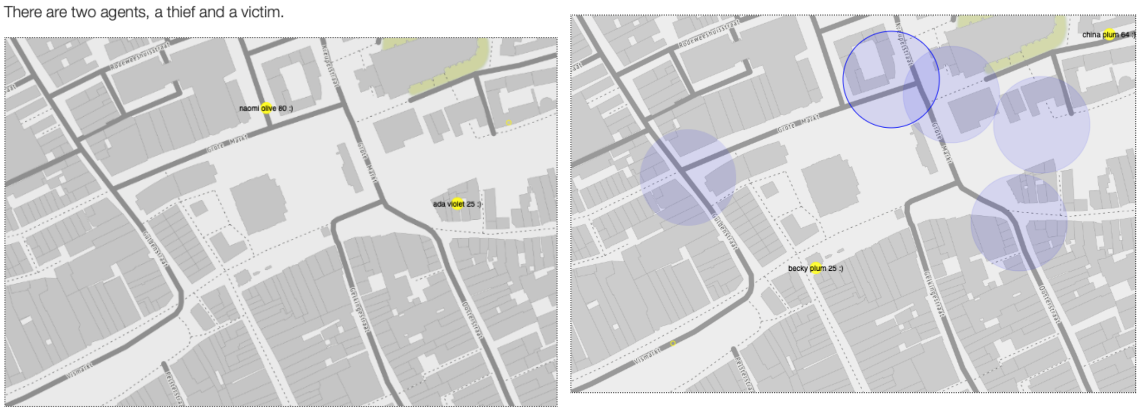


Figure 4.1: The behaviour of the agents. The nodes with rounded edges correspond to the nodes in the Bayesian Network.



(a) map of environment - 2 agents

(b) Camera locations are randomly initialized

Figure 4.2: The Grote Markt environment

Conclusion	Frequency P(event)	BN P(event)
seen_1_0	0.52	0.52
know_valuable_1_0	0.046	0.469
know_vulnerable_1_0	0.238	0.2385
motive_1_0	0.012	0.0186
sneak_1_0	0.012	0.0182
stealing_1_0	0.012	0.0177
object_dropped_accidentally_0	0.186	0.187
E_valuable_1_0	0.046	0.046
E_vulnerable_1_0	0.238	0.237
E_psych_report_1_0	0.006	0.002
E_camera_1	0.778	0.777
E_sneak_1_0	0.012	0.018
E_camera_seen_stealing_1_0	0.008	0.0116
E_object_gone_0	0.198	0.197

Table 4.1: Correspondence

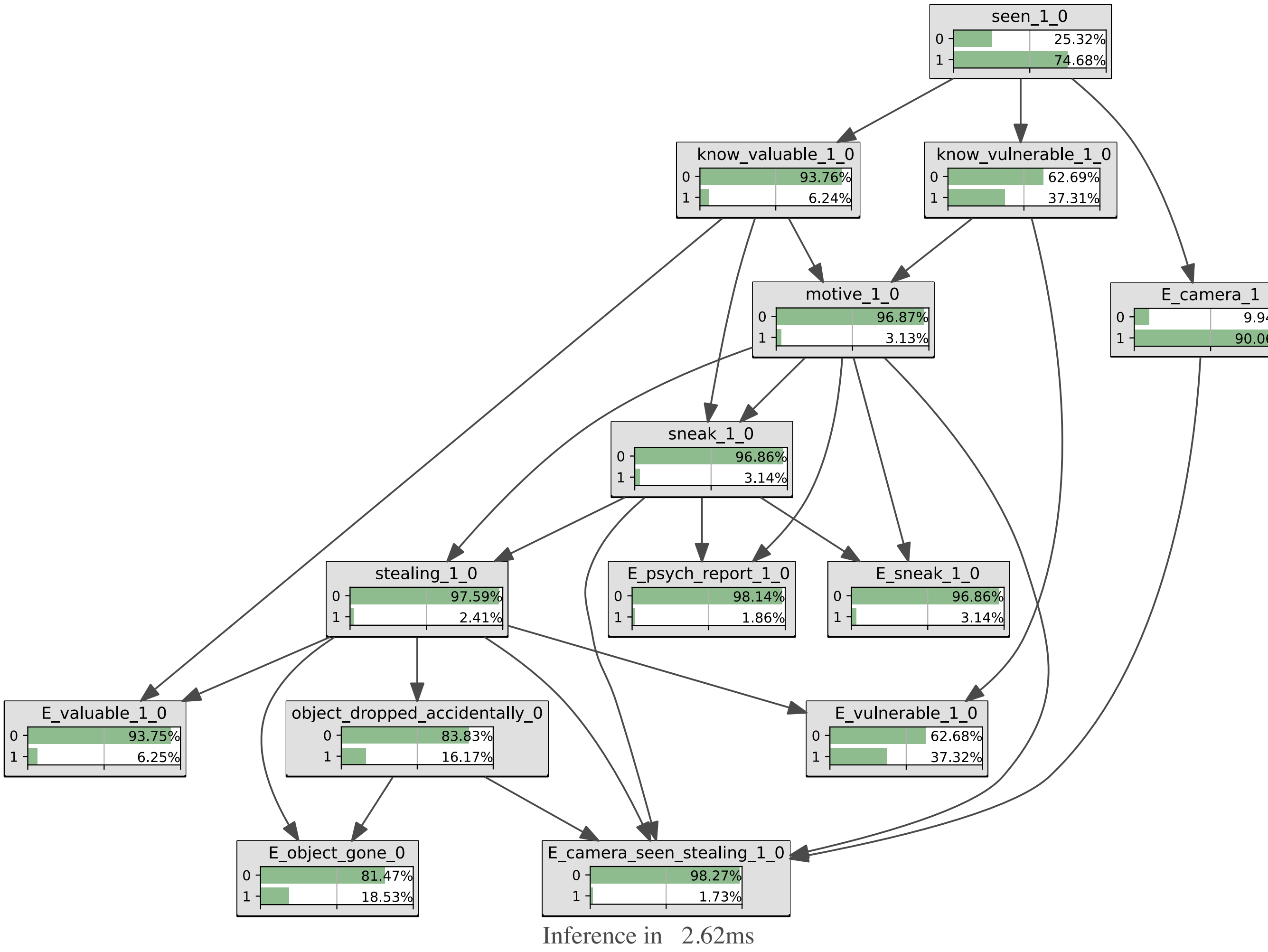
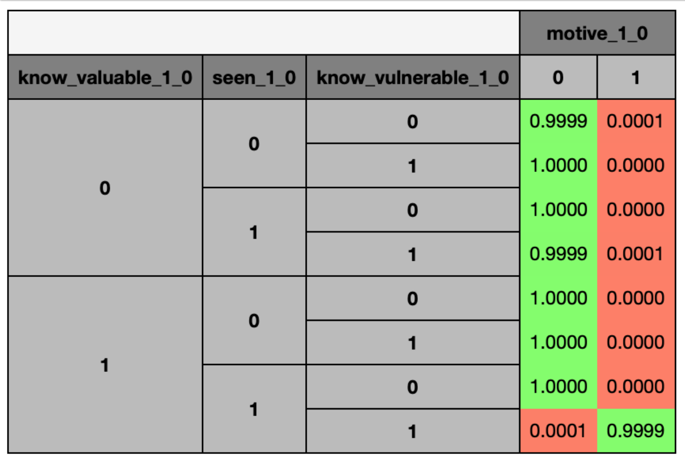


Figure 4.3: Network of Grote Markt simulation.

2. **The values in the conditional probability tables are elicitable from human modellers** (todo)

   On the other hand, nodes like ‘motive’ are almost constructed logically (Figure 4.4), which implies that we might not need to elicit probabilistic information about all hypotheses nodes - if we find appropriate priors for nodes without parents, and their children can be constructed out of logical combinations of their parents, we might only need to elicit probabilistic information for the parents. However, most of the hypothesis nodes are not such logical combinations, and hence need effortful and almost impossible-to-find information about frequencies.



			motive_1_0	
know_valuable_1_0	seen_1_0	know_vulnerable_1_0	0	1
0	0	0	0.9999	0.0001
		1	1.0000	0.0000
	1	0	1.0000	0.0000
		1	0.9999	0.0001
1	0	0	1.0000	0.0000
		1	1.0000	0.0000
	1	0	1.0000	0.0000
		1	0.0001	0.9999

Figure 4.4: Conditional probability table for ‘motive_1_0’. A logical approach.

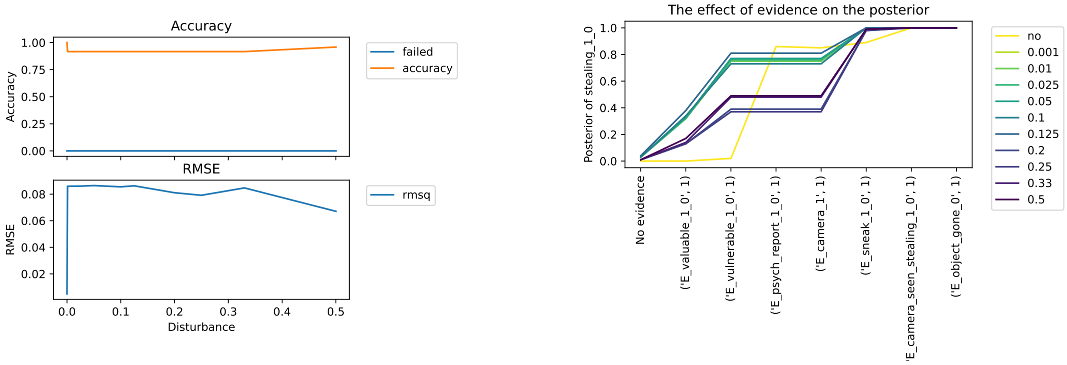
3. **The BN is robust against imprecision in the conditional probability tables**

   We see again that the networks hold up well against loss of precision (Figure 4.5), which is promising for human builders, as long as they do not assign 1 and 0 but instead $1 - \epsilon$ and ϵ.

4.4.2 Structural

1. **The BN represents all events of the scenario** No irrelevant events are in the network. There are some connections between nodes that seem irrelevant or over-specified, though.

2. **The BN has temporal ordering of the hypothesis nodes** Due to the ordering as presented to the K2 algorithm, we see that most of the hypotheses nodes from the main scenario are ordered temporally. The node ‘object_dropped_accidentally_0’ is the one exception. This node represents the alternative scenario where the thief does



(a) Network Under Disturbance.

(b) Progression of evidence resulting in changing the posterior

Figure 4.5: Loss of precision in networks.

not steal, but the victim agent drops their object by accident. When the two scenarios are merged when the ordering is presented to the K2 algorithm, the second scenario is added after the first scenario, but before the evidence, because the second scenario occurs less frequently. It is correct that the node is connected to the 'motive_1_0' and 'stealing_1_0' nodes, because the thief cannot have a motive or steal an object of the victim if the victim does not possess the object anymore. Ideally, we would want this node to have no parents. That this does not happen is a flaw in the naive calculation of the temporal ordering - we should not just look at ordering, but at the time-step of the event instead to produce correct temporal ordering of merged scenarios.

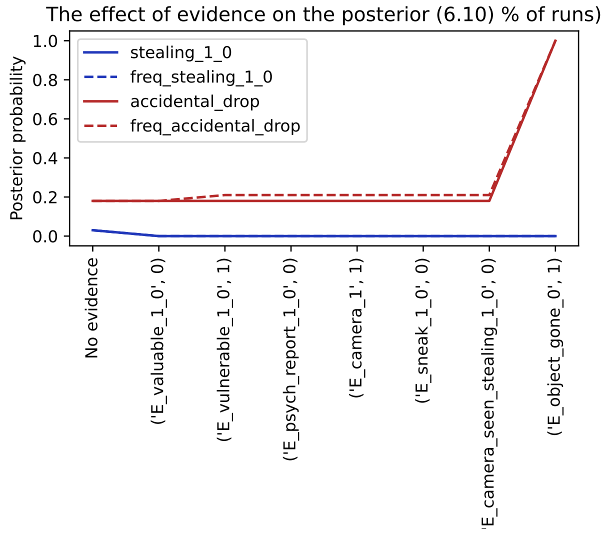
3. **The BN follows the evidence-idiom** Every piece of evidence has at least one hypothesis node as a parent, and no piece of evidence is the parent of a hypothesis node. The network satisfies this constraint, although not every hypothesis node has evidence applied to it.

4. **The BN represents multiple alternative scenarios** (todo) We see that the conflicting scenarios are reflected in the BN.

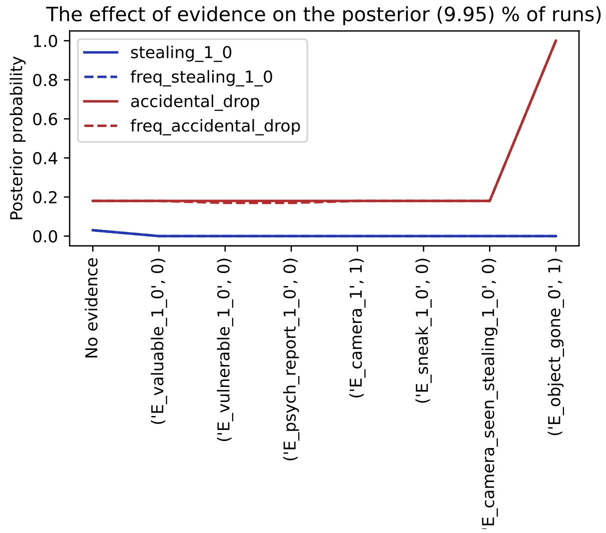
4.4.3 Predictive

1. **The BN only predicts high probability of the output node given relevant evidence truth values**

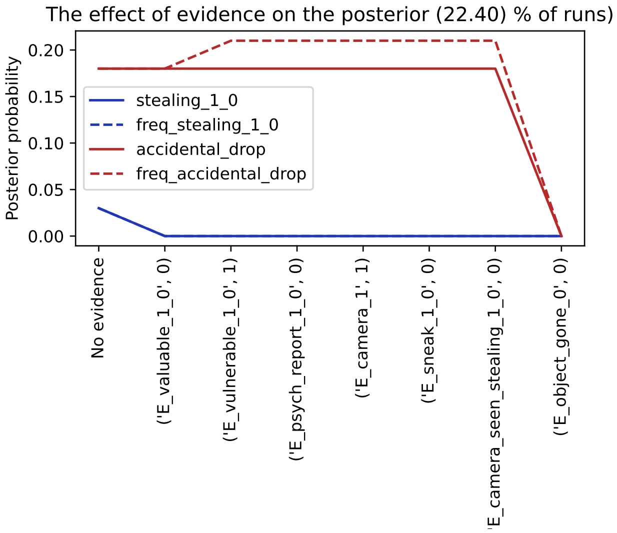
   We see here that the posterior probability of the 'stealing_1_0' node reflects exculpatory evidence well - as soon as we find evidence that would make it impossible for the agent to have stolen (such as not finding the other agent vulnerable, or the object


The effect of evidence on the posterior (6.10) % of runs)
Posterior probability
1.0
0.8
0.6
0.4
0.2
0.0
stealing_1_0
freq_stealing_1_0
accidental_drop
freq_accidental_drop
No evidence
('E_valuable_1_0', 0)
('E_vulnerable_1_0', 1)
('E_psych_report_1_0', 0)
('E_camera_1', 1)
('E_sneak_1_0', 0)
'E_camera_seen_stealing_1_0', 0)
('E_object_gone_0', 1)

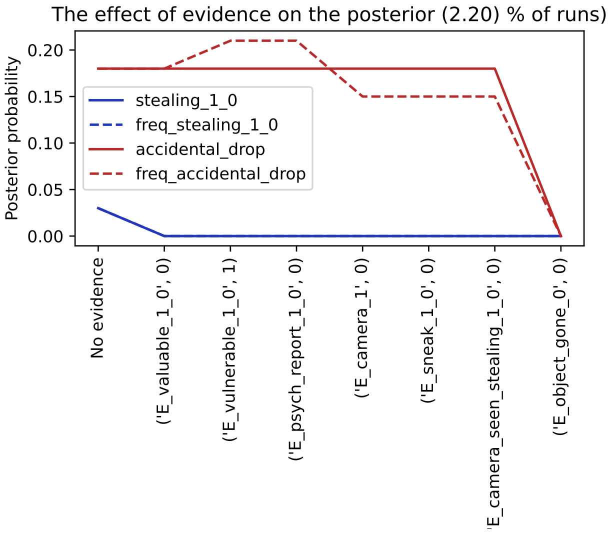
(a)


The effect of evidence on the posterior (9.95) % of runs)
Posterior probability
1.0
0.8
0.6
0.4
0.2
0.0
stealing_1_0
freq_stealing_1_0
accidental_drop
freq_accidental_drop
No evidence
('E_valuable_1_0', 0)
('E_vulnerable_1_0', 0)
('E_psych_report_1_0', 0)
('E_camera_1', 1)
('E_sneak_1_0', 0)
'E_camera_seen_stealing_1_0', 0)
('E_object_gone_0', 1)

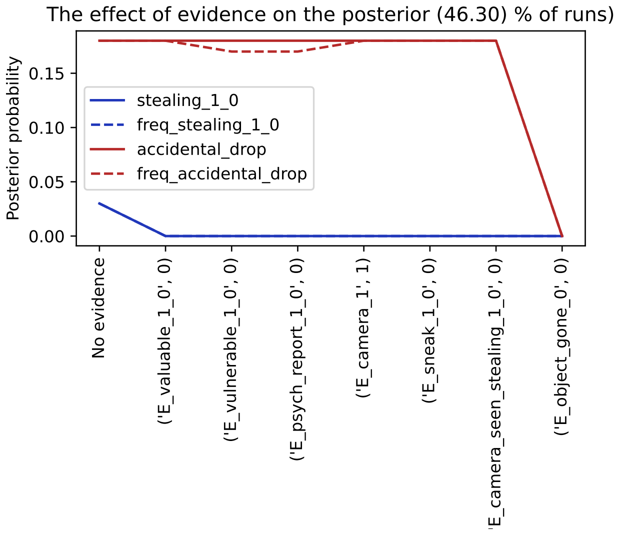
(b)


The effect of evidence on the posterior (22.40) % of runs)
Posterior probability
0.20
0.15
0.10
0.05
0.00
stealing_1_0
freq_stealing_1_0
accidental_drop
freq_accidental_drop
No evidence
('E_valuable_1_0', 0)
('E_vulnerable_1_0', 1)
('E_psych_report_1_0', 0)
('E_camera_1', 1)
('E_sneak_1_0', 0)
'E_camera_seen_stealing_1_0', 0)
('E_object_gone_0', 0)

(c)


The effect of evidence on the posterior (2.20) % of runs)
Posterior probability
0.20
0.15
0.10
0.05
0.00
stealing_1_0
freq_stealing_1_0
accidental_drop
freq_accidental_drop
No evidence
('E_valuable_1_0', 0)
('E_vulnerable_1_0', 1)
('E_psych_report_1_0', 0)
('E_camera_1', 0)
('E_sneak_1_0', 0)
'E_camera_seen_stealing_1_0', 0)
('E_object_gone_0', 0)

(d)


The effect of evidence on the posterior (46.30) % of runs)
Posterior probability
0.15
0.10
0.05
0.00
stealing_1_0
freq_stealing_1_0
accidental_drop
freq_accidental_drop
No evidence
('E_valuable_1_0', 0)
('E_vulnerable_1_0', 0)
('E_psych_report_1_0', 0)
('E_camera_1', 1)
('E_sneak_1_0', 0)
'E_camera_seen_stealing_1_0', 0)
('E_object_gone_0', 0)

(e)

valuable), the posterior for stealing immediately turns to 0.

However, if we look at the progression of evidence in 4.4(a), where all the evidence is true, we see a posterior > 0.9 for the 'stealing_1_0' node as soon as we entered that the agent has a motive ('E_psych_report'). Depending on a given guilt threshold, this might be enough to convict. But this is very weak evidence in real life - just because the agent might have a motive, does not mean that it is going to steal. We see a small increase in the posterior when 'E_camera_seen_stealing_1_0' is set to true, when this should really be the strongest piece of evidence in the entire set. This shows that if we have access to the private knowledge of the agent (its intention to steal), we can make good predictions.

2. **We can know when the BN predicts the wrong outcome** (todo)

4.5 Discussion

We have shown in this chapter that we can generate satisfactory Bayesian Networks from simple agent-simulations with non-trivial environments. These networks can be used to reason towards a conclusion about a posterior probability, given a set of evidence. The network is accurate and is robust under imprecision. However, there are too many arcs in the network, and it is unclear how we should elicit the correct probabilities.

4.5.1 Limitations of the simulation.

This simulation should be seen as a preliminary for further research, as it is lacking in several aspects: the environment is still too simple, the agent-behaviour is too static, and there are only two agents in the simulation. These are the same weaknesses as identified in (Haojie Zhu, 2021). Expanding the simulation to include or improve these aspects is useful for future research.

The non-trivial environment is still not reflecting the non-trivialness of real environments. The environment of a city offers different affordances to different people, but here the map is shared by all agents. Additionally, shops and buildings can actually be entered.

There is a lack of dynamic behaviour in the agents. They can move around and decide to steal from someone, but they do so based on simple factors. This does not meaningfully reflect the real dynamics of street crime - for example, the vulnerable agent walks around with their valuable object in plain sight, why would they do that if they know that there are thieves on the loose?

One of the purposes of modelling a 'real' location is to model the people within that real location. There are many people around on the Grote Markt in real life, modelling just two of them (and not modelling interactions with the crowd) is sufficient to show that automated Bayesian Networks might work in this situation. However, situations where Bayesian Networks might rely on statistical data from real samples (that might be taken from real locations, like the Island Prior), are not modelled in this simulation.

4.5.2 Problems

1. **Does the BN not depend on private knowledge?**

   We can think about the process of a criminal investigation and a trial as a systemic way of gathering, sharing and finally agreeing on knowledge. Initially, only the criminal will know the entire story, victims know parts of it, and the police and judges know even less. An ideal trial results in common knowledge of the entire situation.

   Of course, it is not in the interest of everyone to freely share their private knowledge all the time. Police might not want to talk about the limitations of their evidence (or their ways of procuring it), witnesses and victims might not want to talk due to incrimination, shame, or relationships to the people involved. Suspects might not want to freely share their murder plans (obviously). There is an abundance of private knowledge.

   The Bayesian Network generated up till now, have not taken this into consideration. We have the full set of nodes - even nodes that a realistic investigation cannot be sure about, like the internal considerations of the criminal agent (like the its assessment of the victim or its tendency to steal). If we were real investigators, we cannot actually know this behaviour and hence cannot model it in our network before the agent admits to the behaviour in court.

   This would imply that we can only create Bayesian Networks after the fact: only when the court case is over, and all the facts and their probabilities have been established, can we start building them. Otherwise, we risk missing out on essential psychological or internal information. This would not be ideal. This leads us to ask the question: do our networks still work if we lose information about the internal intentions of agents?

   We created a second network, based on the first scenarios, only now without the private knowledge. This means that the random variables E_vulnerable_1_0, E_valuable_1_0 , E_sneak_1_0 are removed from the list of total random variables that are passed to the K2 algorithm. The resulting network is shown in Figure 4.7.

   We find that this network has fewer nodes, the same temporal ordering as the original network, and might even be more easy to interpret due to these factors. If we look at the response of the output node to evidence valuations (Figure 4.8), we see that the network results in the same pitfalls as the previous network: if all the evidence is true, we see that the most heavy evidence, which is the fact that the camera sees

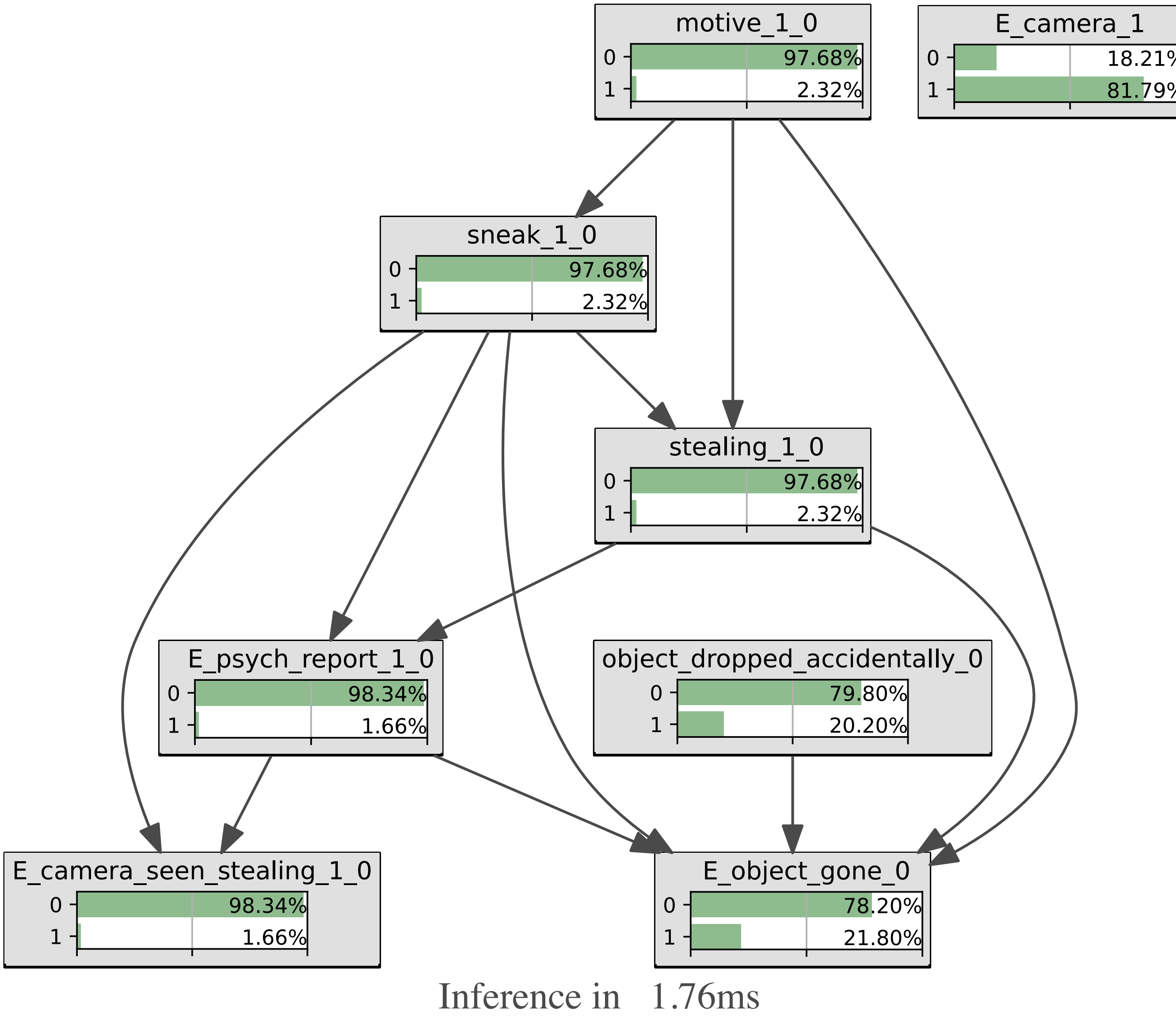


Figure 4.7: Network of Grote Markt simulation.

the agent steal, is 1, this does not affect the posterior belief in theft, which is high from the initial psychological report. This would not be strong evidence in a real-life court case.

2. **Could we apply this BN to a different location?**

   We used one non-trivial environment to create this network. However, the underlying geometry of the simulation affects the probabilities that can be found in the simulation. We cannot just assume one prior probability for 'seen_1_0', since the probability whether some agent sees the other, depends on the structure of the environment. We can show this by using the exact same agent behaviour, but placing the agents on a different map.

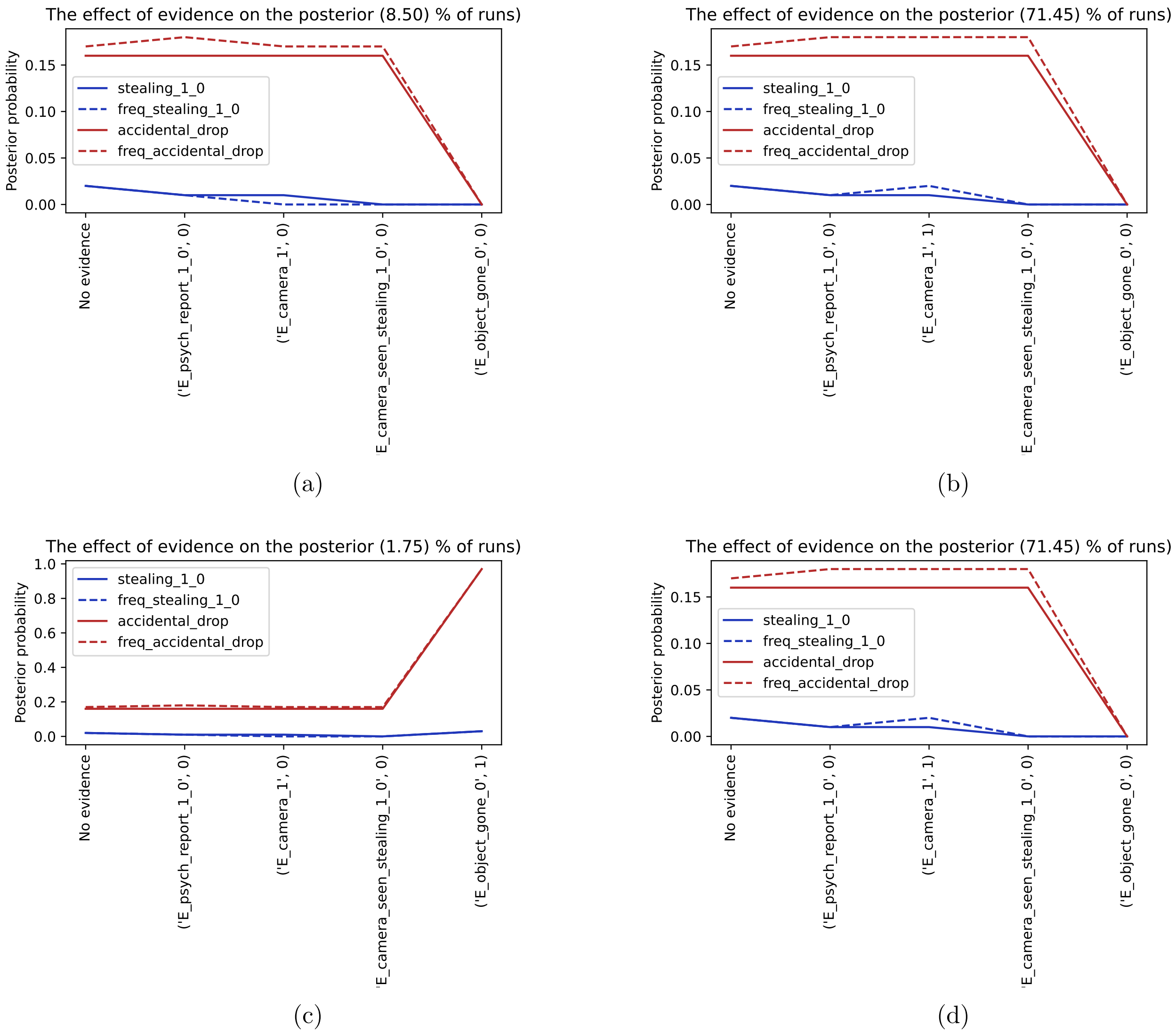
   Instead of the Grote Markt, we selected 5 different parts of Groningen (Figure 4.9), converted them into maps according to the method, and then let the agents loose in them to rob each other.

   We look at the difference in the cpt tables for 'seen_1_0', which represents the event that agent 1 (the thief), sees agent 0 (Table 4.2). If we did not need to condition on underlying geometry of the simulation, we would expect that this probability of 'seen_1_0' would be the same, regardless of map. However, we find that the probability of the thief seeing the potential victim, depends on the underlying map.

   This means that we actually cannot speak of just one 'global' probability for 'seen_1_0'. The probability of the thief seeing the victim depends on a variable that is not included in the Bayesian Network: the map. Instead of a 'global' probability for 'seen_1_0', that applies to all maps, we can only speak of the probability of the agent seeing the victim as conditioned on a specific map.

   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. Additionally, there's no way to predict how the map that we're using affects the probability of 'seen_1_0', this probability emerges from the interaction of the agents with the map. This has implications for the real world, because it means that we can't depend on some generic "probability of getting robbed", we need to condition on spatial conditions, and background world assumptions.

3. **Identity and guilt?** We have avoided the problem of identity in this network, because we only have two agents, and one of the agents is always the thief, the other the victim. As soon as we create a more plausible simulation of crime, where everyone could, in principle, rob everyone, given the right incentives, we would need a different kind of reporter. Even if we created the same simulation, but now the victim agent might also be able to rob the thief-agent, we would need twice the amount of nodes (we need 'stealing_0_1', and the rest as well). Three agents that can



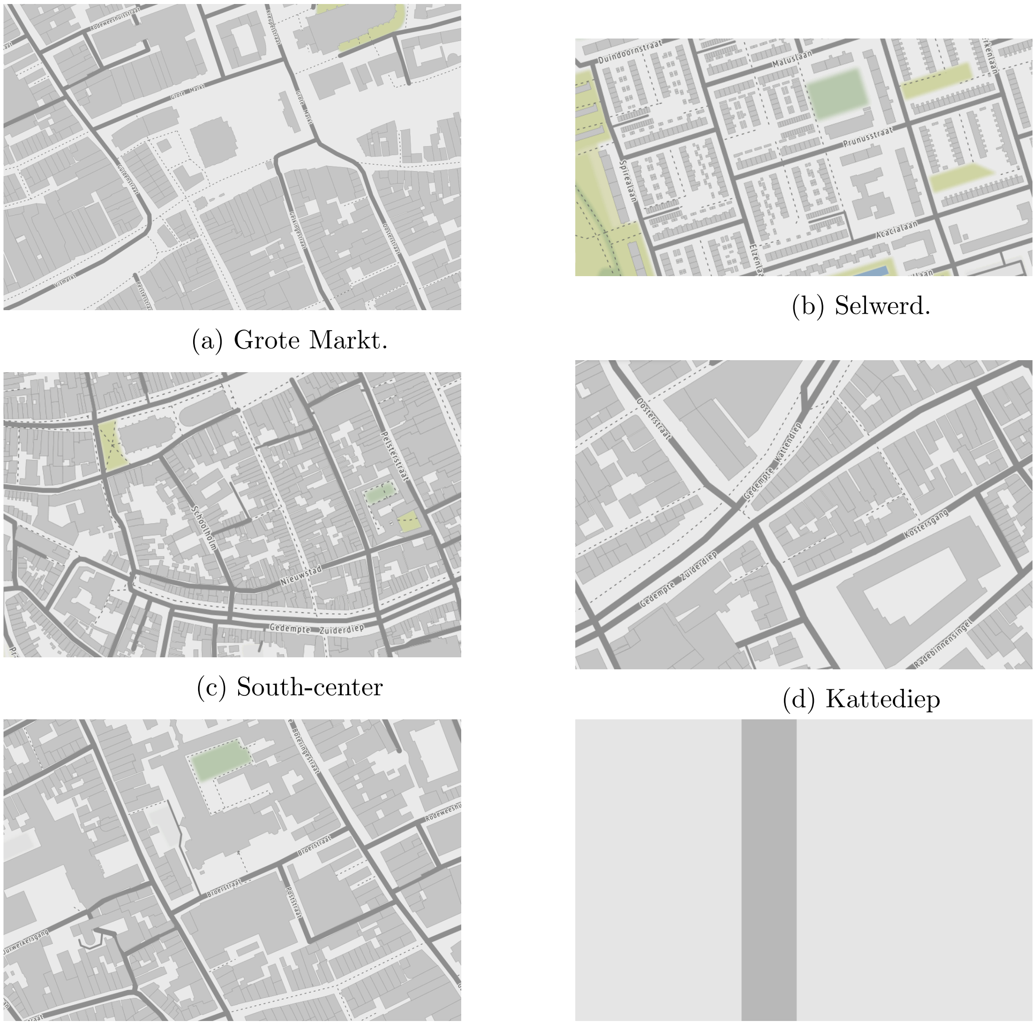
(a)

(b)

(c)

(d)

Figure 4.8: Effect of evidence on posterior



(a) Grote Markt.

(b) Selwerd.

(c) South-center

(d) Kattediep

(e) Street with the academy building

(f) Not Groningen, just a wall

Figure 4.9: Maps.

map	cpt of 'seen_1_0' True
selwerd	0.471
academy	.951
groteMarkt	0.534
kattediep	0.534
wall	0.512
zuidCentrum	0.509

Table 4.2: Difference in cpt depending only on difference in underlying map, no further difference in agent behaviour!

each rob each other, and we have 6 times as many nodes, resulting in for n agents, $n(n-1)$ number of nodes if we use the same structure. This is very implausible due to complexity constraints. However, generalising the nodes (so using 'stealing', instead of 'stealing_0_1') means that we have to represent the identity of the suspect in some other way. It is unclear how that is to be done.

4.5.3 Possible Legal Interpretations.

As we found in the network in the previous chapter, the problem of operationalisation continues to haunt us. The simulation has very clear operationalisation, as represented by the always-correct reporters. However, it is unclear what shape these reporters should take outside of our simulation. Lawyers and judges would have to ask themselves the following questions, to be applied to every node:

- **By what method can we find out whether this event happened or not?**

  This question can be answered relatively easily for evidence nodes. For instance, we would know that 'E_camera_1' would be true in real life, if we saw agent 1 on a relevant camera. For our psychological report, we would know that this event did not happen if the psychiatrist decided that the victim did not fit the suspect's profile.

  However, this becomes very complex for hypothesis nodes. How would you find out of 'stealing_1_0' is true in real life? In our simulation, we just set the event to 1 if it happened. In real life, how would you operationalise this? It is unclear. Should we decide that 'stealing_1_0' is true if the victim says that their object was stolen? That's perhaps a part of an operationalisation, but it is not a completely valid one, since we know that people can lie about their stuff being stolen. If we say, the victim should say that their object was stolen, and we should find the object on the thief, we are still in trouble if the thief decides to sell the object before they can be apprehended. What would be the method to find out that the state of the real world is such that you could say that 'stealing_1_0' is true, or false?

In the real world, we do not need an operationalisation for these kinds of facts, because we all 'sort of know what we mean'. However, in Bayesian Networks we need an operationalisation, because Bayesian Networks and random variables are mathematical objects, and need to map events to truth values. This cannot be done without operationalisation.

- **If we have a method, how do we decide which events we want to include in our frequency calculation?**

  This is the problem of the reference class. To take the example of camera, if we include all cameras in the inner city, the probability that the thief would be seen is great, and hence would not be very relevant to proving the thief's guilt. However, if we decide that we will only use cameras that were on the victim's trajectory, we have limited the amount of cameras, and the probability of the thief showing up on one of these 'coincidentally' is lower, hence it would be a stronger piece of evidence if we found that it was true (and more exculpatory if we found that it was false).

  Even though the reference class is a fundamental problem, as long as every legal participant is aware of it, and can argue about why some reference class is better than another class, or decides on the appropriate classes together, this should not be a limiting factor for the Bayesian Network in itself. However, it requires a great deal of work that would not be done if Bayesian Networks were not used, and it is unsure if it would actually result in better outcomes.

These essential questions should come before talking about Bayesian Network accuracy, precision or structure, especially if these networks are to be used for practical applications.

Chapter 5

Conclusion

Our research questions, and answers:

- **Can we create Bayesian Networks that correctly reflect multi-agent simulations of crimes?**

  We can create Bayesian Networks that correctly reflect multi-agent simulations of crimes. For different types of simulations, the Bayesian Networks that were generated using the pipeline had high accuracy, low RMSE, generally fulfilled the structural criteria we set out. The probabilities in these networks reflected the probabilities we found in the simulation.

  The networks were even robust under imprecision, which means that even though experts might disagree on elicited probability values, or there are imprecisions in the measuring tools, the performance of the network and the conclusion that it gets to in the end, might remain correct.

  We can also show how we should update our posterior probability of an event given a set of evidence and hence show that different pieces of evidence have different strengths. The effect of evidence on the posterior lines up with the evidence as reported in the simulation, as well as with modeller's intuition about the evidence. The visualisation of the posterior probability of the outcome node is a useful interpretative tool for reasoning. Different sets of evidence can also be compared with each other, so that we could compare the effect of different evidence and different scenarios on the posterior probability of the ultimate output node.

  However, the networks that we generated using the pipelines did have some features that are hard on modellers, some nodes had too many parents to easily fill a cpt. We might want to trade-off high network scores for user ease, and restrict the number of parents that a node can have. Additionally, the probabilities that were used in the network would be difficult for a human modeller to elicit or discover. However, since we have our robustness to imprecision, this might not be a large practical problem. Additionally, the temporal ordering of the nodes is not correct when two alternative scenarios are represented within one network.

- **If we can, under what assumptions do these networks function?**

  The assumptions under which the networks are correct, are too strict for this approach to be applied one-on-one to the real world. There are three problems that were identified in this work. These are the meaning of the reporters, private knowledge, and implicit conditioning. These problems are all abundantly present in the real world and it is not clear how to we should fix them. This means that Bayesian Networks might remain a good tool for seeing how evidence affects hypotheses, but might not be ready for the real world.

Reporters and random variables

Fundamentally, the problem is that the nodes in Bayesian Networks are random variables. They are not pragmatic, dialogical, argumentation, or logical sentences, but mathematical objects. They are random variables, and a random variable implies an observation procedure that maps a world state to a truth value. 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. Hence, Bayesian Networks might work for subsections of reality that have a clear observation procedure. For example, reasoning with DNA evidence or other valid forensic evidence. In these cases, we know exactly what it means for the node to be true, since we know the measurement associated with the random variable.

However, for many of the events that we encounter in our simulation, or in Bayesian Networks in the state of the art, we do not know how we are determining that the node is true or not. We do not know which operationalisation is used to determine the truth value, and if that operationalisation is correct or not. This information has to be included with the networks, otherwise we cannot interpret what a node actually means.

Private knowledge

This problem is very clear: in our simulation, we know when private information is true or false, even if we have no way of knowing this private knowledge in real life.

If we remove this knowledge from our network, we find that this effects the posterior probability of the outcome node to such an extent that we fall below a 'comfortable' threshold of guilt. This implies that we can only generate a network that correctly and legally reasons about evidence if everyone in the process would help. This is not feasible.

Conditioning on implicit parameters

We find that if we change the parameters of the simulations, such as using a different map, the probabilities for events change. It means that there is outside influence - things that are not nodes, can influence the cpt's of the network, as if there is an invisible 'environment' node that is the parent of all the nodes in the network. This means that if we see a Bayesian Network and we do not know exactly the context in which it was created, we cannot assume that it generalises to a different environment, even if on the face of it, the nodes would be the same. We cannot take parts of that network (with probabilities) and put it in a different one, because the Bayesian Network is implicitly defined over the environment for which it was originally created, and not any other.

5.1 Future Work

It is difficult to assess the human criteria for Bayesian Networks. We need formal methods to investigate whether a person could really 'think' of a dependence relationship between two nodes, or estimate the correct probabilities. The ad-hoc, subjective way that the human criteria were assessed in this thesis are not very satisfactory. New standards need to be invented that correspond to objective facts about people's ability to estimate probability and investigate conditional relations between nodes.

The simulations as tested here are still very simplified - they are only appropriate as a baseline for testing Bayesian Networks, and for nothing else. Testing Bayesian Networks on more complex simulations would likely result in a reduced accuracy due to inherent and irreducible uncertainty, and lay bare many more problems that do not emerge from these simple simulations. Hence, testing on more complex models is necessary to fully understand the limitations of Bayesian Networks.

The values of ϵ can be reduced from 0.01 to smaller probabilities, to discover how this would influence accuracy and precision of networks, especially in situations that are more complex than the simulations we used here. If there are inherent features to the simulation that makes the frequency of some event smaller than 0.01, it is important to know how that would interact with an ϵ of 0.01. Hence, this should be investigated further.

As the K2 algorithm used depends on a given node ordering, the node-ordering algorithm used in this project resulted generally in successfully replicating the temporal structure of scenarios, however the merging of scenarios caused problems, and resulted in a missing temporal structure. This can be fixed relatively easily, by taking the average time-step at which an event occurs into account, instead of only the ordering relative to other events.

While we discussed the problem of operationalisation in the discussion sections of these chapters, this was not investigated directly. This makes it unclear how big the effect of inefficient, insufficient or invalid operationalisation on the structural, performance and human criteria of the network would be. This problem can be investigated in simulation in simple ways - for instance, by changing the accuracy of reporting, such that a reporter might sometimes report an event wrongly. Alternative operationalisations for the ‘same’ random variables or events should also be investigated. Investigating the effects of non-ideal operationalisation is essential for knowing if we can generalise our Bayesian Networks beyond the bounds of simulation.

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.

Bibliography

Allen, R. and Pardo, M. (2007). The problematic value of mathematical models of evidence. *The Journal of Legal Studies*, 36(1):107–140. 7

Bex, F. (2010). *Arguments, Stories and Criminal Evidence.* Springer Science and Business Media LLC. 5

Bosse, T. and Gerritsen, C. (2008). Agent-based simulation of the spatial dynamics of crime: On the interplay between criminal hot spots and reputation. In *Proc. of the 7th Int. Conf. on Autonomous Agents and Multiagent Systems.* 9

Chen, X.-W., Anantha, G., and Lin, X. (2008). Improving bayesian network structure learning with mutual information-based node ordering in the k2 algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):628–640. 15

Colyvan, M. (2013). Idealisations in normative models. *Synthese*, 190(8):1337–1350. 9

Colyvan, M., Regan, H. M., and Ferson, S. (2001). Is it a crime to belong to a reference class. *Journal of Political Philosophy*, 9(2):168–181. 7

Cooper, G. and Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347. 15

Dahlman, C. (2020). De-biasing legal fact-finders with bayesian thinking. *Topics in Cognitive Science*, 12(4):1115–1131. 6

Dawid, A. P. (2008). Beware of the dag! In *JMLR: Workshop and Conference Proceedings 6: 59–86.* 2

de Koeijer, J. A., Sjerps, M. J., Vergeer, P., and Berger, C. E. (2020). Combining evidence in complex cases-a practical approach to interdisciplinary casework. *Science & Justice*, 60(1):20–29. 6

Druzdzel, M. J. and van der Gaag, L. C. (2000). Building probabilistic networks: awhere do the numbers come from? *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING.* 7

Ducamp, G., Gonzales, C., and Wuillemin, P.-H. (2020). aGrUM/pyAgrum : a Toolbox to Build Models and Algorithms for Probabilistic Graphical Models in Python. In *10th International Conference on Probabilistic Graphical Models*, volume 138 of *Proceedings of Machine Learning Research*, pages 609–612, Skørping, Denmark. 6

Fenton, N., Neil, M., and Lagnado, D. A. (2012). A general structure for legal arguments about evidence using bayesian networks. *Cognitive Science*, 37(1):61–102. 5, 6, 16, 21

Fenton, N., Neil, M., Yet, B., and Lagnado, D. (2019). Analyzing the simonshaven case using bayesian networks. *Topics in Cognitive Science*, 12(4):1092–1114. 5, 6

Gerritsen, C. (2015). Agent-based modelling as a research tool for criminological research. *Crime Science*, 4(1). 9, 12

Haojie Zhu, F. W. (2021). An agent-based model for simulating urban crime with improved daily routines. *Computers, Environment and Urban Systems*. 9, 34

J Kadane, D. S. (1996). *A Probabilistic Analysis of the Sacco and Vanzetti Evidence*. John Wiley and Sons. 6

Kazil, J., Masad, D., and Crooks, A. (2020). Utilizing python for agent-based modeling: The mesa framework. In Thomson, R., Bisgin, H., Dancy, C., Hyder, A., and Hussain, M., editors, *Social, Cultural, and Behavioral Modeling*, pages 308–317, Cham. Springer International Publishing. 9

Oniśko, A. and Druzdzel, M. J. (2013). Impact of precision of bayesian network parameters on accuracy of medical diagnostic systems. *Artificial Intelligence in Medicine*. 20

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann. 2

Pennington, N. and Hastie, R. (1993). 5

Renooij, S. (2001). Probability elicitation for belief networks: issues to consider. *The Knowledge Engineering Review*, 16(3):255–269. 7

Ronald Meester, K. S. (2021). *Probability and Forensic Evidence*. Cambridge University Press. 6

Timmer, S. (2016). *Designing and Understanding Forensic Bayesian Networks using Argumentation*. PhD thesis, University of Utrecht. 5

van Leeuwen, L. (2019). A comparison of two hybrid methods for analysing evidential reasoning. In *JURIX*. 6, 8

Verheij, B., Bex, F., Timmer, S. T., Vlek, C. S., Meyer, J.-J. C., Renooij, S., and Prakken, H. (2015). Arguments, scenarios and probabilities: connections between three normative frameworks for evidential reasoning. *Law, Probability and Risk*, 15(1):35–70. 5

Vlek, C. (2016). *When stories and numbers meet in court.* PhD thesis, University of Groningen. 5

Vlek, C. S., Prakken, H., Renooij, S., and Verheij, B. (2015). Representing the quality of crime scenarios in a bayesian network. In *JURIX*. JURIX. 3, 5, 12

Wagenaar, W. A., Van Koppen, P. J., and Crombag, H. F. (1993). *Anchored narratives: The psychology of criminal evidence.* St Martin's Press. 5