Defeasible Bayesian Reasoning

Introducing Logical Reasoning into Bayesian Networks

-

Project Proposal

Luke Neville
Computer Science Honours
University of Cape Town
Cape Town, South Africa
nvlluk001@myuct.ac.za

Elijah Roussos
Computer Science Honours
University of Cape Town
Cape Town, South Africa
elijah.rou@gmail.com

ABSTRACT

Bayesian Networks are a class of graphical models used to represent relationships between entities and events in a system, and can be queried for probabilistic information pertaining to those events. This paper proposes a project that would examine the viability of supplementing a Bayesian Network with a non-monotonic knowledge base, that is a collection of defeasible logical rules. The aim of this project is to increase the expressive power of a Bayesian Network and allow additional knowledge to be added to a model in a straightforward manner. This will take the form of finding an algorithm that attempts to draw additional relationships between nodes in a Bayesian Network, according to the propositional rules of some knowledge base, which can then be queried. This algorithm would then be adapted to handle defeasible information in a supplementing knowledge base, and implemented in a reasoner. The paper provides background information in the relevant fields, which includes a brief overview of implication and entailment in propositional logic, non-monotonic reasoning and the properties thereof, as well as the foundations of Bayesian reasoning. Furthermore, this paper highlights the key management strategies of the project. We address risk by categorising the various risks associated with this project and determining the corresponding mitigation strategies. Additionally we assess project success by outlining important milestones over a timeline. Overall, we conclude that the basis for the project is sound, as a Bayesian Network model that works in conjunction with logical inferential rules would allow for more complex reasoning and would be beneficial to many fields in artificial intelligence.

Keywords

Bayesian Networks; Formal Logic; Knowledge Representation; Propositional Logic; Defeasible Reasoning; Non-Monotonic Logic; Artificial Intelligence

1. PROJECT DESCRIPTION

1.1 The Problem

This project aims to introduce limited forms of logical implication, such as those from classical and defeasible reasoning, between variables in a Bayesian Network. This can be done as the nodes in a Bayesian Network can be viewed as propositional atoms holding a true or false value, albeit with some probability attached to the relationship.

Given some Bayesian Network, we will seek an approach to introduce the logical statements $A \rightarrow B$ and $A \mathrel{|\!\sim} B$ into the knowledge base, where A and B are variables in the Bayesian Network. As variables in a Bayesian Network are largely independent, this will produce additional dependencies between variables in the Bayesian Network, or modify already existing dependencies. We will attempt to answer such questions has *"How does this change the network and the algorithms associated with it?"*, and *"How do we reason with this network?"*.

Initially, we will only attempt to introduce classical implication, $(A \rightarrow B)$. Once this is successful, we will introduce defeasible implication $(A \mathrel{|\!\sim} B)$. This will make up the theory portion of the project, which will look at how the Bayesian Network algorithms need to be modified in order to accommodate these dependencies. The implementation of these modified algorithms forms the other half of the project. The goal of this implementation is to build a tool that allows users to query a Bayesian network based on this new model, given some classical or defeasible implication statements.

1.2 Why it is important

The addition of logical statements into a Bayesian Network knowledge base would give network designers finer control over the network. It would allow additional dependencies to be added to a network after construction that increase the way dependencies between variables can be expressed. If more knowledge becomes available over time, this can be added to a network without the need to completely reconstruct the network.

In addition, solving this problem is important as it has the potential to decrease the computational complexity of querying a Bayesian Network. This is as the network may condense or simplify due to a reduction in the number of nodes. This results in fewer causal dependencies between nodes, making Bayesian Network algorithms such as variable elimination simpler to compute.

Bayesian Networks find their primary application in artificial intelligence, and as such, any technique that reduces the amount of computational resources needed to process Bayesian Network queries would be greatly beneficial to the field. In addition, the theory underlying our proposed model

has ramifications about the way we treat causality in logical systems. Specifically, the results of our project may provide a way representing causality in knowledge bases, which is more expressive than the current model which only considers the notion of implication.

1.3 Possible issues and difficulties

1.3.1 Theory

- There is no previous literature detailing implementing logical implication in Bayesian Networks. Thus, there is no clear method of determining these algorithms.
- It is difficult to estimate the time required to form an algorithm and prove its correctness.
- Introducing classical implication into Bayesian Networks turns out to be non-trivial, and there is little time left to introduce defeasible reasoning.

1.3.2 Implementation

- A suitable Bayesian Network framework that a wrapper can be built around to implement the logical implication is not found.
- Building a completely new tool from scratch could involve a lot of work and would present additional errors and complications on top of the implementation of the algorithms.

2. BACKGROUND

2.1 Classical Reasoning

2.1.1 Logical Implication

Logical implication represents the **if-then** operation, and is denoted with the symbol $\rightarrow$. This connective describes the relationship of logical consequence. That is, given two atoms A and B, if A implies B then whenever A is known to be true, B will also be *true* [2]. The truth table for $A \rightarrow B$ is given by:

world	A	B	$A \rightarrow B$
w_1	true	true	true
w_2	true	false	false
w_3	false	true	true
w_4	false	false	true

Table 1: Truth Table depicting $A \rightarrow B$

2.1.2 Logical Entailment

A Propositional logic knowledge base is comprised of a finite set of logical rules, in the form of propositional statements. The rules of a knowledge base can be interpreted as a large conjunction [2]. An inference ϕ can be entailed from some knowledge base K if ϕ is known to be true in every world. Entailment is denoted by $\models$, such that if K entails ϕ it is represented as:.

$$K \models \phi$$

2.2 Non-Monotonic Reasoning

2.2.1 Defeasible Entailment

The work of Kraus, Lehman and Magidor (KLM) in [1] proposed a set of natural properties of non-monotonic reasoning. Conditional entailment, denoted with the symbol $\mid\sim$, was introduced to describe plausible inferences. For instance, if atom A typically implies another atom B, it is given by:

$$A \mid\sim B$$

2.2.2 Rational Consequence and the R Logic

KLM organised the essential characteristics of non-monotonic reasoning into a hierarchy of systems. KLM rational logic **R** [3], is the logic system used to define the $\mid\sim$ operator in this problem. For **R**, authors Lehmann and Magidor outlined 7 key properties of conditional entailment sets, presented in the form of inference rules:

Reflexivity:
Conditional inference should imply itself
$A \mid\sim A$

Left Logical Equivalence (LLE):
Logically equivalent formulas should entail exactly the same consequences
$\models A \leftrightarrow B$ then $(A \mid\sim C) \rightarrow (B \mid\sim C)$

Right Weakening (RW):
All plausible consequences that potentially exist should be accepted
$\models A \rightarrow B$ then $(C \mid\sim A) \rightarrow (C \mid\sim B)$

Cautious Monotonicity (CM):
Learning a new fact, the truth of which can be plausibly concluded, should not nullify previous inferences
$[(A \mid\sim B) \wedge (A \mid\sim C)] \rightarrow (A \wedge B \mid\sim C)$

Conjunction (And):
Conditional inference should obey propositional conjunction
$[(A \mid\sim B) \wedge (A \mid\sim C)] \rightarrow (A \mid\sim B \wedge C)$

Disjunction (Or):
Conditional inference should obey propositional disjunction
$[(A \mid\sim C) \wedge (B \mid\sim C)] \rightarrow (A \vee B \mid\sim C)$

Rational Monotonicity (RM):
Only additional information, the negation of which was expected, should force us to withdraw plausible conclusions previously inferred
$[(A \mid\sim B) \wedge \neg(A \mid\sim \neg C)] \rightarrow [(A \wedge C) \mid\sim B]$

2.2.3 Rational Closure

The Rational closure of **R** is an algorithm allows us to perform entailment over defeasible knowledge bases. From a high-level view, the algorithm works as follows. All defeasible statements in a knowledge base K are converted into their corresponding classical implication forms. The knowledge base now contains only classical propositional statements. These statements are then ranked by initially assuming all statements are equally valid in any given world. Then, each statement is checked to ensure that it does not

cause conflict within the knowledge base. If it does, it is pushed 'up' into a new rank of exceptionality. This process is repeated until a ranked interpretation is formed - a ranked list of propositional statements where a higher rank indicates that the sentence is deemed to be more exceptional in the context of the knowledge base [4]. This ranked interpretation can then be queried for entailment as per a standard classical propositional knowledge base.

Given this, a defeasible formula $\phi = A \mid\sim B$ is said to be in the *rational closure* of K should the ranked interpretation of A be strictly less than the ranked interpretation of $A \wedge \neg B$ or where A has no rank [5] [6].

2.3 Bayesian Reasoning

2.3.1 Bayes Theorem

Bayesian Networks rely on **Bayes Rule**, which describes the notion of *conditional probability* [7].

Let A and B be two events in some sample space S. Then, the conditional probability of event B given that event A has occurred is denoted with $P(B|A)$, and is given by:

$$P(B|A) = \frac{P(A,B)}{P(A)} = \frac{P(A|B) \cdot P(B)}{P(A)}$$

Using this formula, we can calculate the probability of any event A given any number of conditions B_i as :

$$P(B|A_1, A_2, ..., A_n)$$

2.3.2 Bayesian Inference

Without a method of evaluating, or reasoning with, a Bayesian network the formalism would be useless. There needs to exist a way of querying the network about the state of its variables and getting some accurate knowledge in return. There are many types of queries that can be used to reason with a Bayesian network. These are outlined by Darwiche in Modelling and Reasoning with Bayesian Networks, from where the following work is taken [2].

The simplest and most intuitive query that can be made on a network is the probability-of-evidence query. This allows us to query the network for the probability that certain events occurred in the system, *i.e.* to ask $P(a)$, where A is some variable in the network. We could also query for a subset of variables in the network, $P(a, b)$. Given some subset of variables, the evidence variables $X_1...X_i$, the query $P(x_1...x_i)$ will return the probability associated with events $X_1...X_i$ taking place. This query is computed using a relatively simple algorithm known as *Variable Elimination*.

Variable Elimination is the process of sequentially removing variables from the network that are not associated with the query, while still maintaining the ability to answer queries on the remaining variables. This elimination removes the need to calculate the probability of all variables in the network. Given a query, $P(a, b)$, variable elimination removes all other variables from the network while embedding the probabilities associated with those variables in the remaining variables, A and B.

3. PROBLEM STATEMENT

3.1 Aims

This project aims to:

- define a suitable meaning for propositional implication between two variables in Bayesian Network
- describe an algorithm for reasoning with a Bayesian Network that contains logical implication in its knowledge base.
- define what defeasible implication between two variables in a Bayesian Network means.
- find an algorithm that allows for defeasible reasoning to be added to a Bayesian Network.
- prove the correctness of all algorithms.
- implement a standalone Bayesian reasoner which allows users to add logical implication statements to the network and reason with it

3.2 Research Questions

The following questions summarize what the project aims to answer:

- Can a provably correct algorithm be defined that allows classical and defeasible implication to be added to a Bayesian Network knowledge base and still be reasoned with?
- How can this algorithm be implemented in a tool that, by making calls to a Bayesian reasoner, allows a user to add classical and defeasible implication to a Bayesian Network?

4. PROCEDURES AND METHODS

The problem is divided into two logical sections - a theoretical component and an implementational component. Most sections in this proposal are split up into these two sections in order to highlight the separation of work.

4.1 Approach

The project can be broken into two distinct components: theory and implementation.

4.1.1 Theory

In researching the relationship between casual links in a Bayesian Network and logical implication, we seek to create an algorithm that will effectively lower the complexity of a Bayesian Network if the network is supplemented with a logical knowledge base. We will then attempt to adapt this algorithm to handle defeasibility.

We conjecture that the addition of logical rules on top of a Bayesian Network will shrink the network by concatenating various nodes, and therefore it must be shown that if such a concatenation occurs that the probabilities of events in the network are preserved. Furthermore, it must be shown that if this is the case, then defeasibility may also be introduced into the network. In particular, we must show that if rational closure is used to determine the ranking of non-monotonic statements in the accompanying knowledge base, a Bayesian Network can be created for each ranking using the aforementioned algorithm.

Once proven, both algorithms will analysed according to their computational complexity.

4.1.2 Implementation

The aim of the implementation component of the project is to build a tool that allows a user to input a Bayesian Network, and then make queries on the network (such as *"What is the probability of variable A?"*) given some classical or defeasible inference statements added to the knowledge base. This will allow the user to make simple queries on the network, as well as specifying additional dependencies between the variables in the network and still maintaining the ability to query the network. The tool will be an implementation of the algorithm defined in the theoretical section of the project.

Ideally, the tool will be built as a wrapper around an already existing Bayesian reasoner. This wrapper will apply the theoretical knowledge gained in the theory component, and then make calls to the Bayesian reasoner. Thus, the wrapper will involve simplifying the given Bayesian Network in the case of classical implication, and applying the defeasible reasoning algorithm in order to simplify the network in the case of defeasible implication.

Failing the implementation of a tool that acts as a wrapper around a pre-existing Bayesian reasoner, a stand alone tool will be built that performs the execution of the algorithm as well as the Bayesian reasoning. The implementation of a wrapper may fail if no suitable Bayesian reasoner can be found. This is a less desirable approach, as time would be spent unnecessarily building a tool for a solved problem. This is however not seen as a major issue as there are many Bayesian Network libraries and packages which can be used.

An example 'test-case' Bayesian Network will be designed and used on the the wrapper which will be used to demonstrate that the wrapper functions correctly. It will also be used to test the theoretical algorithms developed in the theory section of this project. The Asia Network is a likely candidate of this test-case Bayesian Network [18]. Once the application of the algorithm is complete, the additional task of building a user interface and performing some basic user testing will be undertaken. This is not a main goal of the project however, and will only be taken on if the underlying wrapper is complete.

4.2 Testing Results

4.2.1 Theory

There is no explicit testing methodology for the theoretical component of the project. Rather, the correctness of any algorithm developed will be given by formal proof, and independently verified. The results pertaining the computational complexity of the algorithm are analytical in nature, and will also be verified.

4.2.2 Implementation

In order to ensure the software tool works as expected unit and integration tests will be written. These will test each component of the software, as well as ensure that each component works in conjunction with the others.

As this is new work in this field, there is little data to test the software against. Throughout development we will have to design tests that accurately and suitably test the software to ensure that it is producing the correct results.

If the user interface is developed as main feature of the software, user tests will be conducted to ensure that the user interface works as expected and is intuitive to use.

4.3 Measurements of Success

In each section of the project - theory and implementation - specific successes will be measured as follows:

4.3.1 Theory

Any theorem resulting from our research should satisfy the following properties:

1. Bayes theorem must hold under concatenation.
2. any concatenation of a Bayesian Network should maintain previous probabilities.
3. propositional implication must hold when concatenating nodes.
4. the properties of KLM logic **R** must hold.
5. rational closure must be satisfied in Bayesian Networks supplemented with a defeasible knowledge base.

Any reasoning algorithm created should satisfy the following criteria:

1. the algorithm is complies with the outlined theory.
2. the algorithm works.
3. the algorithm successfully concatenates a Bayesian Network and/or reduces its complexity.
4. the computational complexity of the algorithm has been determined.
5. the defeasible version of the algorithm complies with rational closure.

4.3.2 Implementation

Successful completion of the implementation of the defeasible Bayesian reasoner will include a software package that:

1. has all tests - both unit and integration - passed successfully.
2. has the ability to import and export Bayesian networks.
3. allows the user to easily query the Bayesian network and receive accurate results.
4. allows the user to add classical and defeasible inference statements to a knowledge base, and still receive accurate results to queries.
5. has at least a partially complete and user friendly user interface.

5. ETHICAL, LEGAL AND PROFESSIONAL ISSUES

5.1 Ethical Issues

There are no foreseeable ethical issues with the project. The project does not involve strict user testing where ethical treatment of users would need to be taken into account.

5.2 Legal Issues

All software and packages used in the project will be open-source to prevent any legal copy-write infringements. Software produced will be open-source and will take on the licences of any packages used.

5.3 Professional Issues

This project poses no professional issues. The work conducted in the project provides a basis for further research and does not aim to discredit any prior research.

6. RELATED WORK

6.1 Theory

Existing methods that integrate propositional logic into Bayesian networks are well established. The work of Pearl describes various mechanisms for probabilistic reasoning in intelligent systems [14]. Apart from propositional logic, Cozman and Mauá et al 2016 [15] describe how description logics can be used to represent Bayesian Networks.

Furthermore, there may be other ways to deal with defeasibility in a Bayesian Network. Most notably is the construct of fuzzy sets and degrees of truth in Fuzzy Logic (a logic in which logical sentences are assigned probabilities of occurring), which may aid in determining the degree of typicality implied by a non-monotonic statement such as $A \nsim B$ [16]. Non-monotonicity can then also be introduced in fuzzy logic, as described by Castro, Trillas and Zurita et al 1995 [17], which can be used to deal with exceptions in much the same way as propositional non-monotonic logic.

6.2 Implementation

The work of *Costa, et al.* [12], describes the implementation of an open-source tool, UmBBayes-MEMB, that can be used for modeling, learning and reasoning upon probabilistic networks [13]. The application allows users to model probabilistic ontologies by implementing the PR-OWL probabilistic ontology language, which is a probabilistic extension of the OWL based on Multi-Entity Bayesian Networks [12]. This results in a combination of First-Order logic and Bayesian Networks, that can be used for probabilistic ontologies. This is closely related to what we are seeking to implement, however looks at another field in Knowledge Representation.

7. ANTICIPATED OUTCOME

7.1 Theory

We conjecture that a Bayesian Network supplemented with a propositional logic base will have its complexity reduced. This is may be in the form of node concatenation in the Bayesian Network, where any number of subsets of nodes in the network will each reduce to one. However, the introduction of logical rules to the network is likely to affect actual probabilities of events, even if node concatenation does not hold. Should these properties hold, an algorithm could be created with the ability to query the new model.

We further conjecture that if such a model is viable, it may be possible to introduce defeasibility into the knowledge base of the Bayesian Network. This may done trivially by creating multiple propositional Bayesian Networks using the new model, each corresponding to a rank in a minimal ranked interpretation of the knowledge base in question. Any query posed to this model must contain the level of typicality desired, most likely in the form of a logical statement.

It may also be done in a non-trivial manner by modifying the existing query algorithm of a Bayesian Network to incorporate a form of rational closure, though it is less obvious what the outcome of this method would be.

7.2 Implementation

We expect the completed tool be used as a basis for further practical implementations of introducing formal logic to Bayesian networks. The system should be able to provide insight into how this works for implication, potentially paving the way for further expansion of the tool.

This project will be deemed a success if at least some form of implication logic is introduced into a Bayesian network successfully, and the implemented tool allows for this to be used in an easy to use, practical manner.

8. PROJECT PLAN

8.1 Risks

Table 2 in Appendix A shows a risk matrix detailing potential risks that could arise through the course of the project. We detail each risk and the associated probability and impact of the risk taking place. The risk factor is the product of the probability and the Impact. The higher the risk factor, the threat the risk poses to the project. This shows that not completing the theoretical component of the project, and missing deadlines pose the highest risk to the project. However, the biggest risk to this project is that the dependency between the theoretical and practical components of the project. Ensuring this does not cause the project will be a focal point of risk mitigation. Specific emphasis will be placed on mitigating and monitoring the highest impact risks.

The mitigation column shows steps that will be put in place to reduce the likely-hood and impact of the risk happening. The monitoring step details how we will keep track of how probable a risk is, while the management step explains how we will deal with the risk should it take place.

8.2 Timeline

As previously stated, the project is split between two main components, a theory deliverable and an implementation deliverable. Milestones for both components are given in section 9.5 . There are two dependencies in the project. Namely, the implementation of both the classical and defeasible algorithm is critically dependent on their completion. The project timeline is given by the Gantt chart in Appendix B.

8.3 Resources required

For the implementation component of the project, the following will be required:

- A **Bayesian Reasoner** will be required for use by the tool to make calls to compute any given query. This should be open source to allow for direct access to the code in order for the tool to plug directly into the software. It should also be licensed to allow for free use of the software. There are a number of potential

options that will be explored as a first step in this project. Some examples of FOSS Bayesian Network Software include Banjo [8], Bayesian Network tools in Java (BNJ) [9] and JavaBayes [10].

- A **programming language and compiler** will be needed to produce the tool. The majority of open source Bayesian Network tools are produced in Java, and such such this is the most likely choice of language for construction of the tool. This would include the use of the JDK [11]. If the Bayesian reasoner that is chosen is not produced in Java however, the tool will be built the same language as that of the Bayesian reasoner.

- The ontology editing software **Protégé** will be used as reference in production of the tool. The interface will be used as inspiration in development of the tool. The source code will be used as a guide in development of a tool that can accept and manipulate propositional statements.

8.4 Deliverables

The theoretical side of the project will only yield one deliverable - the algorithm for condensing a Bayesian Network given some classical or defeasible inference statements. The implementation aspect of the project will produce a number iterations on the tool as deliverables. Initially, a working prototype of the tool will be produced, followed by any potential iterative steps in the tool, and finally the completed tool. As a final deliverable, the project final paper will be completed detailing the complete process - theoretical and practical - of the project.

8.5 Milestones

The milestones for the theoretical component of the project are:

1. Define the difference between implication and causality.

2. Determine the effect of logical implication within a Bayesian Network with regards to probability propagation in the network.

3. Investigate the effect of node concatenation within a Bayesian Network.

4. Attempt to concatenate the Bayesian Network using logical rules and prove that such a concatenation holds.

5. Should concatenation hold, determine the algorithm for concatenation and calculate its computational complexity.

6. Determine the algorithm for querying the new model and calculate its computational complexity.

7. Investigate the effect of introducing defeasibility into the new model.

8. Prove that any defeasible modification to the algorithm satisfies the KLM **R** properties.

9. Calculate the computational complexity for the defeasible version of the algorithm.

The milestones for the implementation component of the project are:

1. Experiment with and get to know the existing Bayesian reasoner.

2. Implement a basic prototype wrapper around the Bayesian reasoner that can communicate with and query the Bayesian network.

3. Implement the algorithm for introducing classical implication into a Bayesian network into the software.

4. Implement the algorithm for introducing defeasible reasoning into the Bayesian network.

5. Complete a working wrapper program around the Bayesian reasoner that allows queries to be made that include classical and defeasible implication between variables.

8.6 Work Allocation

As mentioned previously, the project can be separated into two defined sections - theory and implementation. Elijah Roussos will complete the theoretical component of the project, which involves determining the relationship between logical entailment and causality, attempting Bayesian Network concatenation using logical rules and developing an algorithm for any model that is created and determining its complexity. The correctness of the aforementioned will also be verified via formal proof.

Luke Neville will complete the implementation section of the project. This includes the designing, implementing and testing the software package. This is heavily based on implementing the algorithm Elijah defines on top of a Bayesian reasoner, allowing a user to reason with a Bayesian network given some classical and defeasible implication statements.

9. CONCLUSIONS

Both classical and non-monotonic logic are well established logic formalisms for reasoning. Specifically, they are able to represent casual and typical relationships between variables, and do so in an efficient manner. We are interested in limited forms of implication in propositional logic and defeasible reasoning. That is, how can classical ($P \rightarrow Q$) and defeasible ($P \mathrel{|\!\sim}$ implication be introduced into another knowledge representation system.

Bayesian Networks have been extensively researched - many practical uses and methods for constructing and evaluating queries can be found in literature [2]. The key insight into their value and performance is due to the assumption of independence between variables that are not directly connected in the network. This greatly simplifies the process of evaluating probabilistic relationships and makes this sort of reasoning feasible in a fast and efficient manner.

We propose that the union of these two reasoning mechanisms would result in more expressive, and potentially more compact, model of reasoning. This would be done by viewing the variables in the network as propositional atoms, and then using the knowledge base to draw causality between these atoms. We conjecture that in drawing logical implication between these variables, they may combine into a single variable with a single probability.

This proposal has outlined the plan for a project which seeks to define an algorithm for introducing classical and

defeasible implication into a Bayesian Network, and the construction of a tool which allows a user to reason with such a model. Overall, we conclude that the project would impact the work of Bayesian Networks by increasing the number of ways relationships between variables in the network can be expressed. It would allow for network size to potentially be reduced, resulting in improvements in processing time in reasoning with a Bayesian network. This could have impacts into the field of Artificial Intelligence as Bayesian networks are increasingly used to model real-world scenarios in this field.

10. REFERENCES

[1] Kraus, S., Lehmann, D., & Magidor, M. (1990). Nonmonotonic reasoning, preferential models and cumulative logics. Artificial intelligence, 44(1-2), 167-207.

[2] Darwiche, A. Modeling and Reasoning with Bayesian Networks. Cambridge University Press, Cambridge, 2009.

[3] Lehmann, D., & Magidor, M. (1992). What does a conditional knowledge base entail?. Artificial intelligence, 55(1), 1-60.

[4] Giordano, L., Gliozzi, V., Olivetti, N., & Pozzato, G. L. (2013). A Semantics for Rational Closure: Preliminary Results. In CILC (pp. 99-113).

[5] Strasser, Christian & Antonelli, G. Aldo, (2001) Non-monotonic Logic, The Stanford Encyclopedia of Philosophy (Summer 2018 Edition), Edward N. Zalta (ed.)

[6] Booth, R., Casini, G., Meyer, T. A., & Varzinczak, I. J. (2015, June). On the Entailment Problem for a Logic of Typicality. In IJCAI (pp. 2805-2811).

[7] Underhill, L., Bradfield D. (2014). INTROSTAT, Department of Statistical Sciences, University of Cape Town

[8] Hartemink, A. (2018). Banjo: Bayesian Network Inference with Java Objects. [online] Users.cs.duke.edu. Available at: https://users.cs.duke.edu/ amink/software/banjo/ [Accessed 19 May 2018].

[9] Bnj.sourceforge.net. (2018). Bayesian Network tools in Java (BNJ) - Kansas State University Lab for Knowledge Discovery in Databases. [online] Available at: http://bnj.sourceforge.net/ [Accessed 19 May 2018].

[10] Cs.cmu.edu. (2018). JavaBayes 0.346. [online] Available at: http://www.cs.cmu.edu/ javabayes/Home/ [Accessed 19 May 2018].

[11] Oracle.com. (2018). Java SE | Oracle Technology Network | Oracle. [online] Available at: http://www.oracle.com/technetwork/java/javase/overview/index.html [Accessed 19 May 2018].

[12] Costa, P.C., Ladeira, M., Carvalho, R.N., Laskey, K.B., Santos, L.L. and Matsumoto, S., 2008, May. A first-order Bayesian tool for probabilistic ontologies. In Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference (pp. 631-636).

[13] Unbbayes.sourceforge.net. (2018). UnBBayes - The UnBBayes Site. [online] Available at: http://unbbayes.sourceforge.net/index.html [Accessed 20 May 2018].

[14] Pearl, J. (2014). Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier.

[15] Cozman, F., & Mauá, D. (2016). The Complexity of Bayesian Networks Specified by Propositional and Relational Languages.

[16] Bergmann, M. (2008). An introduction to many-valued and fuzzy logic: semantics, algebras, and derivation systems. Cambridge University Press.

[17] Castro, J. L., Trillas, E., & Zurita, J. M. (1998). Non-monotonic fuzzy reasoning. Fuzzy Sets and Systems, 94(2), 217-225.

[18] Lauritzen, S., & Spiegelhalter, D. (1988). Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. Journal of the Royal Statistical Society. Series B (Methodological), 50(2), 157–224.

APPENDIX

Appendix A

Table 2: Risk matrix including mitigation, monitoring and management of each risk

Risk	Probability*	Impact*	Risk Factor	Mitigation	Monitoring	Management
Inability to find an algorithm for introducing logical statements into a Bayesian Network - Theory section uncompleted	3	9	36	Actively find resources and plan meetings with supervisor to ensure progress is made	Stick to project timeline and check progress regularly	Work with supervisor to find a solution for implementation
Implementation section uncompleted	3	2	5	Begin prototyping early and plan regular meetings with supervisor to ensure progress is made	Stick to project timeline and check progress regularly	Attempt at least a basic prototype implementation to show proof of concept
Partner dropping out	2	6	12	Separate work out as much as possible	Check on partners status often	Work with supervisor to fill in missing information
Poor time management leading to not meeting project deadline	4	7	28	Stick to strict project deadlines and work actively throughout project	Check progress against project timeline	Hand in basic prototypes and any work completed
Not finding a suitable Bayesian Reasoner to use as a framework on which to build the wrapper	5	5	25	Start early and experiment with many Bayesian Reasoners in order to find a suitable candidate	Check progress against project timeline and decide early if a Bayesian Reasoner cannot be found.	Begin implementing custom Bayesian Reasoner
Conflict/ disagreement on path forward within group	3	7	21	Put in place strict conduct rules, and foster an environment where group members can approach each other to discuss issues	Be open and willing to discuss issues and progress made in the group	Attempt to resolve conflict, bringing in supervisor if necessary.

* rated on a scale of 1 - 10; 1 being low and 10 being high.

Appendix B

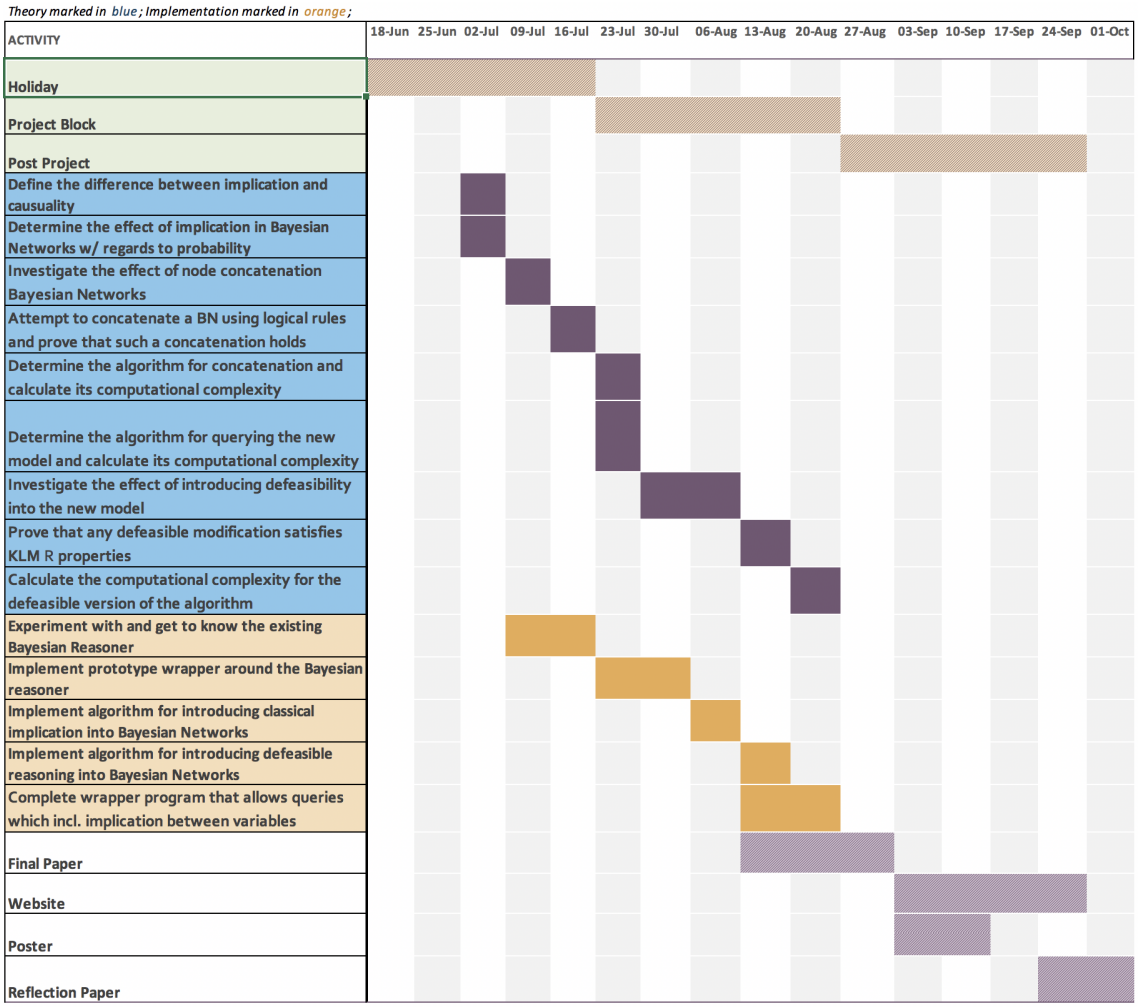

Figure 1: Proposed Gantt Chart for the project