Implicative Bayesian Networks

Combining propositional logic and Bayesian Networks to create a more expressive representation model

Elijah Roussos
RSSELI007
Department of Computer Science
University of Cape Town
elijah.rou@gmail.com

ABSTRACT

In this paper a model is proposed that combines Bayesian Networks and Propositional Logic, called an Implicative Bayesian Network (IBN). The goal of the project was to create a more expressive way to represent inferential systems, while maintaining the ability to use the well-established algorithms synonymous with both classical reasoning and Bayesian reasoning. The model was then extended with defeasible structures, which allows the model to perform non-monotonic reasoning. This paper also provides background information in the relevant fields, including an overview of logical implication and entailment in propositional logic, non-monotonic reasoning and Bayesian Networks. The paper then describes how to integrate propositional logic into a Bayesian Network, and includes new definitions, proofs, descriptions of the various relationships and properties that exist in an IBN, as well as the algorithm to transform a Bayesian Network into an IBN via a propositional logic knowledge base. To extend the IBN model to be able to perform non-monotonic reasoning, details are outlined about how the model should transform in structure. Descriptions of the new formalisms and procedures required for the extension are also provided. It is identified that further research into the properties of the construction could improve the model and its versatility. In conclusion, the basis for the model is sound, and provides a new and intuitive way to model intelligent systems.

Keywords

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

1. INTRODUCTION

Within the field of artificial intelligence, researchers are always trying to produce new ways to present and reason with knowledge. Knowledge representation and reasoning systems are often very powerful and are able to solve complex decision problems. However, these systems' ability to reason often comes at the cost of the expressivity of the representation language in question. In other words, the more one is able to express in one's knowledge representation language, the more difficult it becomes solve decision problems effectively. Two well established representation languages exist in the form of propositional logic and Bayesian Networks.

Propositional logic is a basic language that reasons explicitly with logical boolean values and the relationships between those values without quantification, and as such, is used to solve basic logical decision problems. Relationships in this language are usually described with logical implication, a notion that describes how the existence of one thing immediately implies the existence of another. Typical queries to a propositional logic system will involve asking the system whether a particular statement is true within the system. These systems may also be extended with the idea of defeasibility, that is, allowing arguments that are contingent and therefore able to deal with exceptions in an effective manner.

A Bayesian Network on the other hand, is a model used reason with probabilistic information in the context of different relationships between events that are dependent on each other. As multiple algorithms exist that exploit the structure of the model, querying the system for probabilistic information is typically computationally efficient.

The goal of this project is to merge these two reasoning systems effectively in order to create a new, more expressive reasoning structure. This involves the construction of a new model, the Implicative Bayesian Network, created by supplementing a Bayesian Network with a logical knowledge base. We speculate this can be done because Bayesian Network system variables are largely independent of one another, and therefore additional relations can be specified between such variables. This paper will cover the necessary background knowledge of the domain and will describe the specific theoretical definitions, proofs and algorithms needed to create such a construction. Extensions to the model will then be discussed, including introducing negation capable implication and incorporating defeasibility into an Implicative Bayesian Network.

The second part of this project is to develop and describe a working implementation of this model in the form of a software tool. This aspect of the project is presented by L, Neville in [1], and is not the topic of this paper.

The development of a model such as the Implicative Bayesian Network will allow modellers to specify more specific relationships between entities in a system, thereby enabling the modellers to articulate certain systems in more detail and with greater accuracy. Furthermore, additional relationships between system variables can be specified after the network has been constructed, without the need to recreate the network. Finally, as the model is built on the existing frameworks of propositional logic and Bayesian Networks, querying an Implicative Bayesian Network can be accomplished with established algorithms, which are both computationally efficient and familiar to end users. As this is the case, Implicative Bayesian Networks may prove to be a stepping stone in the way researchers in Artificial Intelligence model intelligent systems.

2. LITERATURE REVIEW & BACKGROUND

2.1 Propositional Logic

The content of this section(2.1) cites Darwiche, A et al [2].

Formally, propositional logic is a mathematical language, used to reason about claims that cannot be further decomposed and have a value of either true or false. To construct a propositional sentence, compositions of boolean variables are formed. A boolean variable may also be referred to as an atom. While an atom is itself a propositional sentence, more complicated sentences may be formed altering atoms or stating specific relationships between atoms using logical operators. A sentence is said to be *declarative* should it correspond to a specific truth value. For any propositional sentence, a *world* is defined as a particular declarative combination of the atoms in the system. In any propositional system there always exist 2^n different worlds, where n is the number of atoms in the system.

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 $true$. It should be noted that logical implication is transitive. 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. An inference ϕ can be entailed from some knowledge base KB if ϕ is known to be true in every world. Entailment is denoted by $\models$, such that if KB entails ϕ it is represented as:.

$$K \models \phi$$

As an example, consider a knowledge base KB, comprised of the following propositional sentences:

$$\begin{array}{|c|}\hline A \rightarrow B \\ A \rightarrow C \\ C \rightarrow E \\ \hline \end{array}$$

Figure 1: A knowledge base

As a trivial demonstration of entailment, it can be seen that $KB \models A \rightarrow E$.

2.1.3 Reasoning in Propositional Logic

When given a propositional sentence, the truth value of the sentence can always be computed when supplied with the values of the atoms in the sentence. It is for this reason that propositional logic is decidable, and therefore given a knowledge base of propositional statements, any possible combination of atoms will definitively result in some value. In other words, any world is decidable.

Due to this decidability, knowledge bases can be queried by checking if a given propositional sentence is consistent with all worlds that the knowledge base expresses. As discussed in 2.1.2, this defines entailment in propositional logic, and gives means to reason with knowledge bases.

2.2 Non-Monotonic Reasoning

Defeasible arguments are contingent: they allow people to draw conclusions from sentences that do not imply a specific answer and are contextual rather than definitive. If for instance someone attempts to cross a street, we could infer that *typically* they would succeed in doing so. However, it is not definite. The individual may be having a particularly unlucky day and get hit by a truck, and therefore would not succeed in crossing the street [3]. Hence, our original inference was defeasible.

This feature of defeasibility is due to the fact that it is *non-monotonic*, that is upon learning new information about a system certain conclusions may be withdrawn. This is in contrast to monotonic logic, such as propositional logic, where the addition of new logical axioms to a knowledge base KB may never decrease the conclusions that can be drawn from KB [4].

2.2.1 Defeasible Implication

The work of Kraus, Lehman and Magidor (KLM) in [5] proposed a set of natural properties of non-monotonic reasoning. Conditional entailment, denoted in this paper with the symbol $\rightsquigarrow$, was introduced to describe plausible inferences. For instance, if atom A typically implies another atom B, it is given by:

$$A \rightsquigarrow B$$

In this paper conditional entailment will be referred to as defeasible implication, and will be treated as an operator similar to propositional implication.

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** [6], is the logic system used to define the $\rightsquigarrow$ operator in this problem. For **R**, authors Lehmann and Magidor outlined 7 key properties of defeasible implication sets, presented in the form of inference rules:

Reflexivity:
Conditional inference should imply itself
$A \rightsquigarrow A$

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

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

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

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

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

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

2.2.3 The Ranking Algorithm and Rational Closure

The rational closure of **R** is an algorithm that performs entailment over defeasible knowledge bases. In order to conduct the rational closure algorithm, every statement in the defeasible knowledge base in question should be assigned a ranking. From a high-level view, the ranking 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, that is a ranked list R of propositional statements where a higher rank indicates that the sentence is deemed to be more exceptional in the context of the knowledge base [7]. The lowest rank of R is R_0, and contains every non-defeasible implication statement. The highest rank of R is referred to as R_∞.

Once a ranking of K has been determined, the rational closure algorithm can be used for entailment checking with respect to K. The way this works is as follows: A formula $\phi = X \rightsquigarrow Y$ is converted to the propositional statement $X \Rightarrow Y$ and is iteratively checked with R to ensure that it is a non-exceptional statement. This is done by checking if the negation of the antecedent $\neg X$ holds in the current ranked interpretation, starting with R_∞. If it does not hold, the check is repeated with the next rank R_{i-1}. This is repeated until $\neg X$ holds in some R_i. An entailment check is then performed with the interpretation R_i and ϕ as if it was a normal propositional logic entailment check.

2.3 Bayesian Networks

A Bayesian Network is a visual construct that represents relationships between probabilistic events in a structured manner. It consists of a graph depicting the relationships between random variables, and a set of conditional probabilities associated with those variables [10]. The random variables in a network are boolean in nature. They are either true or false. Suppose a model for the way rain interacts with being sad. Two random variables would be defined, $rain$ which represents if it is raining, and sad which represents the emotion of sadness. If rain has a known influence on being sad, this could represented by the following Bayesian Network:

$rain$	$P(rain)$
true	0.1

$rain$	sad	$Pr(sad\|rain)$
true	true	0.6
false	true	0.5

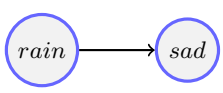


Figure 2: A Bayesian Network and its CPTs

Note that in both probability tables specified in Figure 2, the probabilities for $sad = false$ are not specified. This is as $Pr(sad|X) + Pr(\overline{sad}|X) = 1$ on some evidence X, and therefore trivially $Pr(\overline{sad}|X) = 1 - Pr(sad|X)$.

The construction is formatted in such a way that making inferences about an event in a system arranged as a Bayesian Network is efficient. This is due to a variety of probabilistic reasoning algorithms that have been developed which take advantage of the simple structure of a Bayesian Network.

2.3.1 Bayes Theorem

Bayesian Networks rely on **Bayes Rule**, which describes the notion of *conditional probability* [11].

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 B given any number of conditions A_i as :

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

2.3.2 The structure of a Bayesian Network

The content of the following section(2.3.2) cites Darwiche, A et al [2]

Formally, a Bayesian Network BN is pair $\langle DAG, CPT \rangle$, where DAG is a directed acyclic graph and CPT is a set of conditional probability tables. DAG nodes represent random variables, with the edges that connect those nodes representing a dependency relation. Variables are deemed *independent* of each other should no edge directly connect them. The CPT of BN specifies the probability distributions of each of the variables and their respective parents in the modelled system. There can only ever be one such CPT specifying a network, and as such any Bayesian Network BN has completeness and consistency guarantees.

The power in Bayesian Networks relies on the notion of variable independence. Given some variable X in BN, if the values of the parents of X are known, then X is also independent from its children. Due to this independence, in any computation only probabilities associated with the variable in question need to be considered, decreasing computation time.

2.3.3 Reasoning within Bayesian Networks

The content of the following section(2.3.3) cites Darwiche, A et al [2]

Without a method of evaluating or reasoning with a Bayesian network, the formalism would be useless. The structure should be able to answer queries about the state of its variables in an efficient and accurate manner.

The simplest and most intuitive query that can be made on a network is the probability-of-evidence query. This is a query for the probability that certain events occurred in the system, *i.e.* to ask $Pr(A)$, where A is some variable in the network. Queries may also be asked for a subset of variables in the network, $Pr(A, B)$. Given some subset of variables, $X_1...X_i$, the query $Pr(X_1...X_i)$ 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, $Pr(A, B)$, 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. THE PROJECT: INTEGRATING LOGIC INTO BAYESIAN NETWORKS

This section details the the construction of the proposed new model, the Implicative Bayesian Network. The model is created by supplementing a Bayesian Network with a propositional logic knowledge base, causing implicative statements to alter the Bayesian Network.

3.1 Relationships between events

This section details the definitions for the two types of relationships in an Implicative Bayesian Network, influences and implications.

3.1.1 Influences

The influence relationship describes whether or not the probability of an event occurring is directly affected by another. Simply put, if some event A affects the probability of another event occurring, B, then A is an influence on B. Denoted $A \rightarrow B$, an influence of A on B is formally defined as follows:

Definition 3.1. *$A \rightarrow B$ if $Pr(B|A, X) \neq Pr(B|X)$ on some evidence X*

By definition, an influence is equivalent to an edge in a Bayesian Network.

3.1.2 Implications

An implication describes a relationship in which the observation of one event immediately implies the existence of another, i.e. if some event B immediately follows upon observing some event A, then A implies B. It then follows that if this is the case, the probability of observing event B given A must be equal to 1. Similarly, if there exist two events A and B such that $Pr(B|A) = 1$ and $Pr(B|\overline{A})$ exists, it must be the case the B logically follows from A, and hence A implies B. As such, an implication of A on B, denoted $A \Rightarrow B$, is formally defined as follows:

Definition 3.2. *$A \Rightarrow B$ if and only if $Pr(B|A, X) = 1$ on some evidence X*

An implication relationship is equivalent to logical implication as found in propositional logic, and obeys all the properties of the connective such as transitivity. Implications in this model may only be specified by positive atoms. Namely, the implications of $\overline{A} \Rightarrow B$, $A \Rightarrow \overline{B}$ and $\overline{A} \Rightarrow \overline{B}$ are not yet defined.

3.2 The IBN model

An Implicative Bayesian Network IBN is a 3-tuple:

$$(DAG_{\perp}, CPT_{\perp}, KB),$$

where $DAG_{\perp}$ is an input directed acyclic graph connected via influences, $CPT_{\perp}$ is a set of conditional probability tables associated with $DAG_{\perp}$ and KB is a set of implications.

The KB set represents the propositional logic knowledge base associated with the system, while the $DAG_{\perp}$ and $CPT_{\perp}$ sets form the bottom Bayesian Network, $B_{\perp}$. The bottom Bayesian Network represents the system when it is unaffected by the inferences made in KB.

Using the elements from the KB set, $DAG_{\top}$ is constructed by creating or modifying existing influences in $DAG_{\perp}$ such that the all the statements in KB are represented as influences in $DAG_{\top}$. Implications are represented with a double line in the newly constructed graph, while regular influences remain the same. Then, the accompanying probability tables for $DAG_{\top}$, $CPT_{\top}$, are created by modifying the probability tables in $CPT_{\perp}$ such that every implication relationship specified by KB holds (covered in section 3.3). Implications will always take precedence of regular influence relationships. $DAG_{\top}$ and $CPT_{\top}$ form the top Bayesian Network $B_{\top}$ of IBN, which represents the system once the inference rules in KB have been applied.

When the top Bayesian Network has been determined, an IBN can then be queried as if it was a normal Bayesian Network. This is done by using satisfiability checking to determine whether the top or bottom Bayesian Network should be used, and then using algorithms such as variable elimination on the correct network to answer the query. An example of a bottom Bayesian Network and the associated top Bayesian Network excluding the probability tables is given below.

$$KB = \boxed{\begin{array}{c} A \Rightarrow B \\ A \Rightarrow C \\ D \Rightarrow E \end{array}}$$

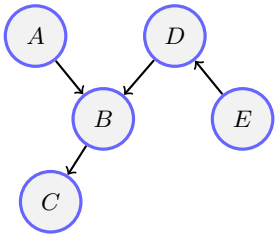


Figure 3: The graph of a bottom Bayesian Network $B_{\perp}$ and the corresponding knowledge base

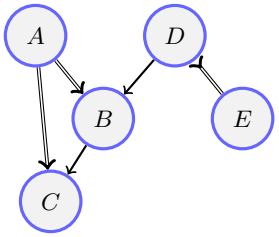


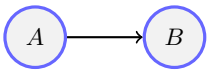
Figure 4: The graph of the top Bayesian Network $B_{\top}$ converted from Figure 3

3.3 Implicative Relationships

Implications specified between various events can take different forms within an Implicative Bayesian Network. This section details the specific transformations to $DAG_{\perp}$ and $CPT_{\perp}$ that occur in the various scenarios of node pairings to create $DAG_{\top}$ and $CPT_{\top}$.

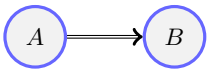
3.3.1 Direct Relation

A direct relation is one in which an implication $A \Rightarrow B$ maps to a specified influence $A \rightarrow B$ in the network (two nodes connected via an edge). In this case, the probability table of the subsequent event B in the relationship is modified such that in every case where the antecedent event A is $true$, $Pr(B|A, X)$ becomes 1.



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0.5

$A \Rightarrow B$

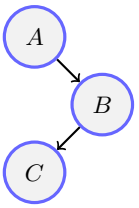


A	B	$Pr(B\|A)$
true	true	1
false	true	0.5

Figure 5: Using a direct relation

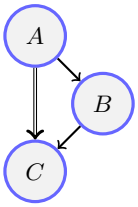
3.3.2 None Relation

A none relation is one in which an implication $A \Rightarrow B$ maps to events in the network that do not directly influence each other such that $A \nrightarrow B$ (i.e. they are independent). In this case an influence is created from the antecedent event A in the implication to the subsequent event B such that $A \rightarrow B$. As A becomes a parent of B, the probability table associated with B is modified to include rows for A. All the rows where A is $false$ contain the probabilities from the previous iteration of the table, while the new rows where A is $true$ have their probability set to be equal to 1.



B	C	$Pr(C\|B)$
true	true	0.7
false	true	0.3

$A \Rightarrow C$



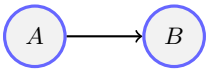
A	B	C	$Pr(B\|A)$
false	true	true	0.7
false	false	true	0.5
true	true	true	1
true	false	true	1

Figure 6: Using a none relation

None relations have the potential to cause cycles in certain situations, which violate the properties of a Bayesian Network. This issue is addressed later with the notion of a *clash* in Section 3.4 .

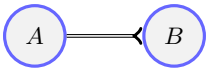
3.3.3 Reverse Relation

A reverse relation is one in which an implication $B \Rightarrow A$ maps to the reverse of a specified influence $A \rightarrow B$. In this case, the direction of the original influence does not change. However, the probability table of the subsequent event B is modified such that in every case where the antecedent event A is $false$, $Pr(B|A, X)$ becomes 0. The Bayesian Network representation for a reverse relation is slightly different in that arrowhead of the edge representing the implication in the graph is reversed.



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0.5

$B \Rightarrow A$



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0

Figure 7: Using a reverse relation

3.4 Implicative Clashes

Allowing for implication relationships that can create cycles via none relations causes problems in the Bayesian Network model. Hence, any implication that causes such a phenomenon is not allowed to modify the network, and will not be used in querying the structure. A clash will be used to identify such implications, and is defined as follows:

Definition 3.3. *A clash occurs should an implication cause a cycle with at least two influences.*

It should be noted that if an implication causes a clash, it does not mean it is thrown out of the knowledge base. Rather, it is distinguished from other implications as potentially causing an *Implicative Cycle*.

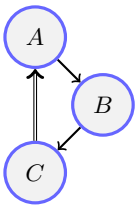


Figure 8: An example of an implication $C \Rightarrow A$ causing a cycle in a network

3.5 Implicative Cycles and Event Equivalence

Implications specified in KB may themselves cause cycles within an IBN. However in this case, the events can be considered equal in outcome. In particular for a set of events $C_1 ... C_n$:

Theorem 3.1. *Given events $C_1 ... C_n$ with $n > 1$, if $C_1 \Rightarrow C_2$, $C_2 \Rightarrow C_3, ..., C_{n-1} \Rightarrow C_n, C_n \Rightarrow C_1$ then $C_1 \equiv C_2 \equiv ... \equiv C_n$*

Consider the formula $C = C_1 \Rightarrow C_2 \wedge X_2 \Rightarrow C_3 \wedge ... \wedge C_{n-1} \Rightarrow C_n \wedge C_n \Rightarrow C_1$, where $n > 1$ is the number of atoms in the formula. Assume some conjunct in C, $\phi_i = C_i \Rightarrow C_{i+1}$, is $false$ in an assignment that makes $C = true$ for some i. For C to be true, the conjunct of $\phi_i \wedge \phi_{i+1}$ should evaluate to true. However, as $\phi_i = false$ there exists no assignment for ϕ_{i+1} such that $\phi_i \wedge \phi_{i+1} = true$. Therefore, ϕ_i must be true, and via the transitivity property of implication, all ϕ_1 to ϕ_n must be $true$. As this is the case, there can exist only two possible assignments for any ϕ_i: $C_i = true, C_{i+1} = true$ or $C_i = false, C_{i+1} = false$. As these are the only possible assignments, all of C_i are logically equivalent, and $C_1 \equiv C_2 \equiv ... \equiv C_n$.

Therefore, events in an implicative cycle can treated as a single event in which all the events in the cycle occur.

Corollary. *Given n events $C_1..C_n$, if $C_1...C_n$ form an implicative cycle C, the event $C = C_1 \cap C_2 \cap, ..., \cap C_n$. It follows that $Pr(C|X) = Pr(C_1, C_2, ..., C_n|X)$ on some evidence X.*

There are now two cases to consider, *Case 1*: influences to an implicative cycle from another event, and *Case 2*: influences from an implicative cycle to another event. To describe what happens in either scenario, consider a Bayesian Network consisting of an implicative cycle C consisting of events C_1, C_2, C_3 and C_4, together with events A and B in the following configuration:

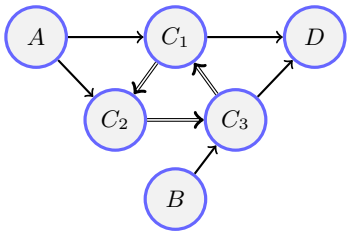


Figure 9: A Bayesian Network with an implicative cycle

3.5.1 Probability to a cycle

For case 1, consider event A and event B with cycle C. $Pr(C|A, B)$ must be equal to $Pr(C_1, C_2, C_3|A, B)$. Via Bayes Theorem, it follows:

$$Pr(C_1, C_2, C_3|A, B) = \frac{Pr(C_1, C_2, C_3, A, B)}{Pr(A, B)}$$
$$= \frac{Pr(C_1, C_2, C_3, A, B)}{Pr(C_2, C_3, A, B)} \cdot \frac{Pr(C_2, C_3, A, B)}{Pr(A, B)}$$
$$= Pr(C_1|C_2, C_3, A, B) \cdot Pr(C_2, C_3|A, B)$$
$$= Pr(C_1|C_2, C_3, A, B) \cdot \frac{Pr(C_2, C_3, A, B)}{Pr(A, B)}$$
$$= Pr(C_1|C_2, C_3, A, B) \cdot \frac{Pr(C_2, C_3, A, B)}{Pr(C_3, A, B)} \cdot \frac{Pr(C_3, A, B)}{Pr(A, B)}$$
$$= Pr(C_1|C_2, C_3, A, B) \cdot Pr(C_2|C_3, A, B) \cdot Pr(C_3|A, B)$$

This expansion can be generalised to several variables and is summarised accordingly:

Theorem 3.2. *Given an event A and implicative cycle $C = C_1 \Rightarrow C_2, ..., C_n \Rightarrow C_1$ consisting of n events. If $A \rightarrow C_1, A \rightarrow C_2, ..., A \rightarrow C_i$ for some $i <= n$, then:*

$$Pr(C|A) =$$
$$Pr(C_1|C_2, ..., C_i, A) \cdot Pr(C_2|C_3, ..., C_i, A) \cdot ... \cdot Pr(C_i|A)$$

The newly calculated probability becomes an entry in the conditional probability table for the new node C. In this example, the same calculation would be done for $P(C|\overline{A}, B)$, $P(C|A, \overline{B})$ and $P(C|\overline{A}, \overline{B})$.

3.5.2 Probability from a cycle

For case 2, consider event D with cycle X. Due to the logical equivalence amongst events in a cycle, there can only be two possible cases of observation: Either all the events are observed to be $true$, or all the events are observed to be $false$. Therefore, the probability table associated with any event which has an influence leaving the cycle is truncated to included only these cases. In this case, every entry in the probability table for D where $C_1 \neq C_2$ is removed.

From figure 7, we obtain the following concatenated Bayesian Network:

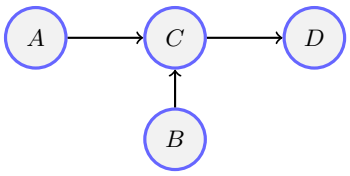


Figure 10: The concatenation of the network in Figure 9

3.6 Querying an IBN

Queries posed to an IBN structure will involve determining the probability of an event occurring in the system given some logical observations. Suppose a query to find the probability of some event A occurring given a set of of n negatable atoms $N = \{N_1, ..., N_n\}$ and a set of m implications $I = \{I_1 \Rightarrow I_2, ..., I_{m-1} \Rightarrow I_m\}$. This is given by:

$$Pr(A|N_1, ..., N_n, I_1 \Rightarrow I_2, ..., I_{m-1} \Rightarrow I_m)$$

In order to perform this query, it must be determined whether the top or bottom Bayesian Network should be used to calculate the required probability. This can be done using entailment checking with respect to KB on every element in both N and I. If every statement specified by N and I is consistent with KB, the top Bayesian Network $B_\top = \langle DAG_\top, CPT_\top \rangle$ is used. However if there exists an element in either N or I that is not consistent with KB, the system is treated as if none of the implications in KB have been applied, and the bottom Bayesian Network $B_\bot = \langle DAG_\bot, CPT_\bot \rangle$ is used.

Upon determining the correct network to conduct the query with, the query is treated as if it was a normal Bayesian Network query without the observed implications as such:

$$Pr(A|N_1, ..., N_n)$$

This can be calculated easily via algorithms such as variable elimination.

3.7 Transformation from a Bayesian Network to an IBN

This section details the algorithms that transform the bottom Bayesian Network $BN_\bot = \langle DAG_\bot CPT_\bot \rangle$ into the top Bayesian

Network $BN_\top = \langle DAG_\top CPT_\top \rangle$ using the knowledge base of implications KB, and the analysis of their computational complexity in terms of Big-O notation.

Algorithm 1, applyImplication, applies a transformation to the conditional probability table(CPT) of the subsequent event in a specified implication $X \Rightarrow Y$. The algorithm performs at least $n/2$ operations, where n is the number of entries in the CPT, and is therefore $O(n)$.

Algorithm 1: Transforming the probability tables of node Y according to the type of implicative relationship specified by X and Y.

```
procedure applyImplication (G, X, Y);
Input : Graph G, Antecedent node X, Consequent node Y
if relation(X,Y) = DIRECT OR relation(X,Y) = NONE then
    if relation(X,Y) = NONE then
        Add edge from X to Y in G
        Extend Y.cpt by length(B.cpt) rows
        Add column for X in Y.cpt
    for entry ∈ Y.cpt do
        if entry.X = true then
            entry.Probability ← 1
else
    foreach entry ∈ Y.cpt do
        if entry.X = true then
            entry.Probability ← 0
```

Algorithm 2, createTop, copies the bottom Bayesian Network BN into a new network BN' and modifies BN' to create the top Bayesian Network. This algorithm relies on a function that finds cycles in graphs, *cycleCheck*, which is assumed to be based on the *Bellman-Ford* algorithm [12]. As this is the case, cycleCheck is assumed to run with complexity $O(|V| \cdot |E|)$, where $|V|$ is the number of vertices in the graph and $|E|$ is the number of edges. Another function is required to find implicative cycles. This can be done trivially via the transitivity properties of implication, and in this algorithm will be represented by the function *findICycles*. The algorithm performs at least k operations in findICycles, and further k operations of cycleCheck and applyImplication, where k is the number of implications in the Knowledge Base KB. Therefore, the algorithm has a worst case time complexity of:

$$O(k \cdot |V| \cdot |E| \cdot n)$$

Therefore, creating a top Bayesian Network from a bottom Bayesian Network can be done in polynomial time.

Algorithm 2: Applying each implication in K to the Bottom Bayesian Network.

```
function applyKnowledge (BN, K);
Input : Bayesian Network BN, Knowledge base KB
Output: Bayesian Network BN'
copy(BN', BN) // copy BN into BN'
iCycles ← findICycles(KB)
for i ∈ iCycles do
    concatenate Cycle i and add to BN
for k ∈ KB do
    if !cycleCheck(BN', k) then
        applyImplication(BN'.graph, k.antecedent,
        k.subsequent)
```

4. EXTENDING THE IBN MODEL WITH NEGATION CAPABLE IMPLICATION

While an Implicative Bayesian Network adds a useful layer of expressivity over a typical Bayesian Network, there is a significant drawback in the model. Namely, the definition of implication does not describe how to deal with statements in which negation is applied to the events. This not only limits how much the system can depict, but in order to describe any defeasible extension, it is required that such properties be in the representation language. To this end, the definition of implication is extended to describe different configurations of negatable events and their effects on conditional probability tables.

4.1 Representing Negation

In order to keep the representation of implication consistent across both the knowledge base and the Bayesian Network components of an IBN, new notation will be introduced that represents different configurations of implication coupled with the negation of events. In particular, the overline in a negated event such as $\overline{A}$ is replaced with a $\circ$ in an implication statement. If for instance, *not* A implies B, this would be written as $A \circ\!\!\Rightarrow B$, and represented by the following Bayesian Network:

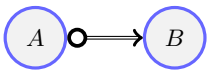


Figure 11: A Bayesian Network representation of $A \circ\!\!\Rightarrow B$

Similarly, if A implies *not* B then $A \Rightarrow\!\circ B$, and if *not* A implies *not* B, then $A \circ\!\!\Rightarrow\!\circ B$. This notation allows the representation to remain the same for both the Bayesian Network and the propositional logic knowledge base.

As with implication in propositional logic, any statement $A \Rightarrow B$ is equivalent to *not* B implies *not* A. Therefore:

$$A \Rightarrow B \equiv B \circ\!\!\Rightarrow\!\circ A$$
$$A \circ\!\!\Rightarrow B \equiv B \circ\!\!\Rightarrow A$$
$$A \Rightarrow\!\circ B \equiv B \Rightarrow\!\circ A$$

Given this, in this extension the reverse relation as specified in Section 3.3.3 is removed, as all reverse relationships can be expressed as direct relationships with negation. The remaining two relations, the direct and none relations, remain in the model. However, the way in which the probability changes is now dependent on the type of implication specified by the relation as well. It should be noted that the notion of a clash still holds in this extension, as well as the notion of implicative cycles.

4.2 Defining negation capable implication

Implication is considered negation capable under the following definition:

Definition 4.1. Following the definition of implication as outlined in Section 3.1.2, implication is considered negation capable if,

$$A \Rightarrow B \textit{ if and only if } Pr(B|A, X) = 1,$$
$$A \circ\!\!\Rightarrow B \textit{ if and only if } Pr(B|\overline{A}, X) = 1,$$
$$A \Rightarrow\!\circ B \textit{ if and only if } Pr(B|A, X) = 0,$$
$$A \circ\!\!\Rightarrow\!\circ B \textit{ if and only if } Pr(B|\overline{A}, X) = 0,$$
$$\textit{on some evidence } X$$

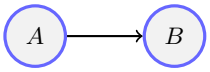
4.3 Type 1 Implication

A type 1 implication is any implication that results in a value of $false$ when some event $A = true$ and some event $B = false$, and

therefore $Pr(B|A) = 1$. These include $A \Rightarrow B$ and the equivalent reverse statement $B \circ\!\!\Rightarrow\!\!\circ A$. A type 1 implication invokes the same transformation to conditional probability tables as the direct and none relations did in Section 3.3 (refer to Figure 5 and Figure 6).

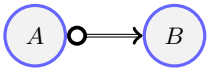
4.4 Type 2 Implication

A type 2 implication is any implication that results in a value of $false$ when some event $A = false$ and some event $B = false$, and therefore $Pr(B|\overline{A}) = 1$. These include $A \circ\!\!\Rightarrow B$ and the equivalent reverse statement $B \circ\!\!\Rightarrow A$. In the example below, all the rows in the probability table for B where A is $false$ have their probability set to be equal to 1.



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0.5

$A \circ\!\!\Rightarrow B$

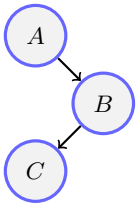


A	B	$Pr(B\|A)$
true	true	0.6
false	true	1

Figure 12: Using a type 2 direct relation

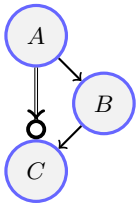
4.5 Type 3 Implication

A type 3 implication is any implication that results in a value of $false$ when some event $A = true$ and some event $B = true$, and therefore $Pr(B|A) = 0$. This includes $A \Rightarrow\!\!\circ B$ and the equivalent reverse statement $B \Rightarrow\!\!\circ A$. In the example below, all the rows in the probability table for C where A is $true$ have their probability set to be equal to 0.



B	C	$Pr(C\|B)$
true	true	0.7
false	true	0.3

$A \Rightarrow\!\!\circ C$

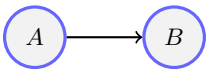


A	B	C	$Pr(B\|A)$
false	true	true	0.7
false	false	true	0.5
true	true	true	0
true	false	true	0

Figure 13: Using a type 3 none relation

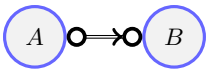
4.6 Type 4 Implication

A type 4 implication is any implication that results in a value of $false$ when some event $A = false$ and some event $B = true$, and therefore $Pr(B|\overline{A}) = 0$. This includes $A \circ\!\!\Rightarrow\!\!\circ B$ and the equivalent reverse statement $B \Rightarrow A$. In the example below, all the rows in the probability table for B where A is $false$ have their probability set to be equal to 0.



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0.5

$A \circ\!\!\Rightarrow\!\!\circ B$



A	B	$Pr(B\|A)$
true	true	0.6
false	true	0

Figure 14: Using a type 4 direct relation

4.7 Amending the algorithm to apply implications

This section details the change to the algorithm applyImplication when using negation capable implication. The algorithm performs at least $n/2$ operations as with Algorithm 1, where n is the number of entries in the CPT, and is therefore $O(n)$.

Algorithm 3: Transforming the probability tables of node Y according to the type of implication by X and Y.

```
procedure applyImplication (G, X, Y);
Input : Graph G, Antecedent node X, Consequent node Y
if relation(X,Y) = NONE then
    Add edge from X to Y in G
    Extend Y.cpt by length(B.cpt) rows
    Add column for X in Y.cpt
if type(X,Y) = TYPE_1 then
    for entry ∈ Y.cpt do
        if entry.X = true then
            entry.Probability ← 1
else if type(X,Y) = TYPE_2 then
    foreach entry ∈ Y.cpt do
        if entry.X = false then
            entry.Probability ← 1
else if type(X,Y) = TYPE_3 then
    foreach entry ∈ Y.cpt do
        if entry.X = true then
            entry.Probability ← 0
else
    foreach entry ∈ Y.cpt do
        if entry.X = false then
            entry.Probability ← 0
```

5. EXTENDING THE IBN MODEL WITH DEFEASIBLITY

This section details the construction of a defeasible IBN from a negation capable IBN, which would allow a multi-state system to be represented by the model.

5.1 Alternations

An alternation describes a relationship in which the observation of one event *typically* implies the existence of another. Specifically, if some event B follows from A *and* $\overline{A}$ is true, then A alternates B. Therefore in the case in which we observe A and $\overline{A}$ holds, the probability of B given A must be equal to 1. As such, an alternation of A on B, denoted $A \rightsquigarrow B$, is formally defined as follows:

Definition 5.1. *$A \rightsquigarrow B$ if $A \Rightarrow B$ when $\overline{B}$ holds true and $\Rightarrow$ is negation capable*

An alternation is equivalent to defeasible implication as found in non-monotonic propositional logic.

5.2 The Defeasible IBN model

A Defeasible Implicative Bayesian Network is a 3-tuple:

$$(DAG_{\perp}, CPT_{\perp}, KB),$$

where $DAG_{\perp}$ is an input directed acyclic graph connected via influences, $CPT_{\perp}$ is a set of conditional probability tables associated with $DAG_{\perp}$ and KB is a set of implications and alternations.

The KB set represents the non-monotonic logic knowledge base associated with the system, while the $DAG_{\perp}$ and $CPT_{\perp}$ sets form the bottom Bayesian Network as with a typical IBN. Now however, KB is used to build an ordered set of Bayesian Networks B, each associated with a rank as determined by the defeasible ranking algorithm such that each rank R_i maps to a Implicative Bayesian Network B_i. The bottom Bayesian Network $B_{\perp}$ is associated with rank 0, R_0, while the top Bayesian Network $B_{\top}$ is associated with rank ∞, R_{∞}. The entire system of networks B can be represented as a single network where the edges for alternations are squiggly lines. An example of a bottom Bayesian Network and its associated knowledge base is shown below, together with the system view graph for B.

$$KB = \begin{array}{|c|}\hline A \rightsquigarrow B \\ A \rightsquigarrow\!\!\circ\, C \\ B \Rightarrow C \\ E \circ\!\!\Rightarrow\!\!\circ\, B \\ \hline \end{array}$$

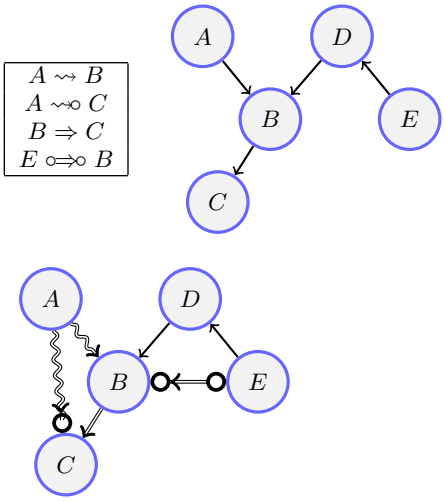


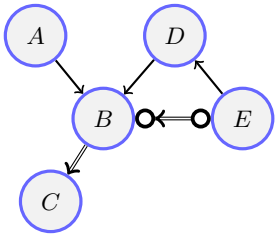
Figure 15: A bottom Bayesian Network and the corresponding defeasible knowledge base, as well as the overall system view graph

Using the ranking algorithm (refer to Section 2.2.3) on alternations found in KB, a ranking R of implications is constructed with $R_0 = \{\emptyset\}$. Then, a series of Bayesian Networks constructed by applying the logical implication statements in R_{i+1} to the corresponding network B_i to create a new Bayesian Network B_{i+1} from $i = 0$. This is done in the same manner as described in Section 3.2. This is repeated until the statements associated with R_{∞} are used to create the top Bayesian Network $B_{\top}$, a network that contains all the statements associated with KB.

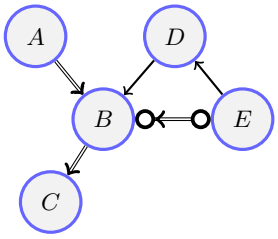
The graph of each individual Bayesian Network B_i is represented in exactly the same manner as specified in Section 3.1, with implication edges as double lines. An example of such a construction is presented below in Figure 16.

$$R = \begin{array}{|c|c|}\hline R_{\infty} \mapsto B_{\top} & A \Rightarrow\!\!\circ\, C \\ \hline R_2 \mapsto B_2 & A \Rightarrow B \\ \hline R_1 \mapsto B_1 & B \Rightarrow C, E \circ\!\!\Rightarrow\!\!\circ\, B \\ \hline R_0 \mapsto B_{\perp} & \emptyset \\ \hline \end{array}$$

Apply R_1 to $B_{\perp}$ to create B_1:



Apply R_2 to B_1 to create B_2:



Apply R_{∞} to B_2 to create $B_{\top}$:

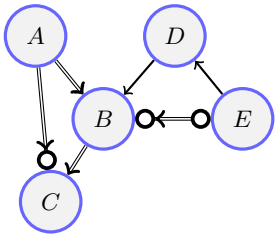


Figure 16: The ranking of KB from Figure 15, R, and the corresponding Bayesian Networks

When the top Bayesian Network has been determined, a defeasible IBN can then be queried for probabilistic information. Queries

are similar to that of a regular IBN. However, rational closure is now used for entailment checking to determine which Bayesian Network in B should be utilised to calculate the probability of the query.

5.3 Querying a Defeasible IBN

Queries posed to a defeasible IBN structure will involve determining the probability of an event occurring in the system given some defeasible observations. Suppose a query to find the probability of some event A occurring given a set of n negatable atoms $N = \{N_1, ..., N_n\}$, a set of m implications $I = \{I_1 \Rightarrow I_2, ..., I_{m-1} \Rightarrow I_m\}$ and j alternations $T = \{T_1 \rightsquigarrow T_2, ..., T_{j-1} \rightsquigarrow T_j\}$. This is given by:

$$Pr(A|N_1, ..., N_n, I_1 \Rightarrow I_2, ..., I_{m-1} \Rightarrow I_m, T_1 \rightsquigarrow T_2, ..., T_{j-1} \rightsquigarrow T_j)$$

Performing this query involves using the rational closure algorithm to determine if all alternations $t \in T$ are entailed by KB. Upon running the rational closure algorithm, ranks will slowly be eliminated from R. This occurs until the algorithm can decide whether t is entailed by KB. If t is not entailed by KB, the algorithm halts and the bottom Bayesian Network B_0 is used for the query. Otherwise, the index of the rank at which this occurs is noted as i, and the Bayesian Network $B_i \in B$ is used for the query.

Upon determining the correct network to conduct the query with, the query is treated as if it was a normal Bayesian Network query without the observed implications as such:

$$Pr(A|N_1, ..., N_n)$$

5.4 Transformation from a Bayesian Network to a Defeasible IBN

This section details the algorithm required to create a Defeasible IBN. The algorithm performs at least r operations, where r is the number of rankings given by performing the ranking algorithm on KB. Together with the run-time complexity of Algorithm 2, Algorithm 4 has a worst case time complexity of:

$$O(k \cdot |V| \cdot |E| \cdot n \cdot r)$$

Algorithm 4: Applying each implication in a ranking of K to create a set of Bayesian Networks.

function createBN (BN, R);
Input : Bottom Bayesian Network BN, Defeasible knowledge base KB
Output: Bayesian Network set B
$R \leftarrow$ rankingAlgorithm(KB)
B.push(BN)
for $i \leftarrow 0$ **to** *R.length* **do**
  b $\leftarrow$ applyKnowledge(B.i, R.i)
  B.push(b)

6. CONCLUSIONS

As seen from the successful construction of the Implicative Bayesian Network model, it is clear that the model is both sound and complete, and is able to reason with a system of knowledge that contains both logical and probabilistic information. In addition, from the graphical representation illustrated in Section 3 it is evident that the model has a thorough visualisation mechanism, and therefore provides a suitable basis for a language of knowledge representation. It is further noted that enhancing the model with defeasibility in Section 5, and by extension negation, also results in a sound and complete construction with an apt visualisation mechanism.

As this is the case, it may be inferred that both the regular and defeasible constructions of an IBN offer a level of expressivity that exceeds that of typical Bayesian Networks. This is due to the ability of an IBN to construct two or more Bayesian Networks, which consequently may be used to represent different system states. This can be achieved with little overhead over a regular Bayesian Network.

While the the construction is correct, the model may be improved by further research. In particular, if implications are discovered to be influences via an if and only if relationship, a method for accurately calculating the associated probabilities of the relationship would need to be determined. Similarly, the properties of implicative cycles could be explored further, and may lead to additional characteristics that could enhance the expressivity of the model further.

In conclusion, the Implicative Bayesian Network model is a successful construction which combines the two reasoning mechanisms of propositional logic and Bayesian Network. This is done while retaining the ability to utilise the existing algorithms associated with the said reasoning structures. Hence the IBN construction provides a new and intuitive way to model intelligent systems without a significant compromise in performance.

7. ACKNOWLEDGEMENTS

First and foremost, I would like to thank Professor Thomas Meyer in suggesting this interesting topic of research and agreeing to supervise the project. Your input and insight into the topic had an immeasurable impact in the outcome of this work. I would like to extend this thanks to my project partner Luke Neville for spending a myriad of long, hardworking hours on the practical component of this project, as well as helping me brainstorm certain topics in the development of the theory. Also, thanks for coming up with the name of the model. Thank you to Professor Deshen Moodley for the invaluable intermediate project evaluation, which helped both Luke and I to finish the project timely. Finally, thank you to my mother, Professor DK Glencross, who provided a great deal of support throughout my journey in writing this paper, and rightly badgered me about writing a scientific paper in first person.

8. REFERENCES

[1] Neville, L (2018) Inferential Bayesian Reasoner, University of Cape Town

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

[3] Koons, & Robert, (2005) Defeasible Reasoning, The Stanford Encyclopedia of Philosophy (Winter 2017 Edition), Edward N. Zalta (ed.)

[4] Bundy A., Wallen L. (1984) Non-Monotonic Reasoning. In: Bundy A., Wallen L. (eds) Catalogue of Artificial Intelligence Tools. Symbolic Computation (Artificial Intelligence). Springer, Berlin, Heidelberg

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

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

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

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

[9] 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).

[10] Darwiche, A. (2010). Bayesian networks. Communications of the ACM, 53(12), 80–90. doi:10.1145/1859204.1859227

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

[12] Addison-Wesley (2010). Data Structures and Problem Solving Using Java 4th Edition, MA Weiss