Introducing Bayesian Argumentation Networks

D. M. Gabbay
Department of Informatics, King's College London,
Ashkelon Academic College, Israel,
Bar Ilan University, Ramat Gan, Israel
University of Luxembourg, Luxembourg.
http: // /www. inf. kcl. ac. uk/ staff/ dg
dov.gabbay@kcl.ac.uk

O. Rodrigues
Department of Informatics, King's College London, The Strand, London, WC2R 2LS, UK,
http: // www. inf. kcl. ac. uk/ staff/ odinaldo
odinaldo.rodrigues@kcl.ac.uk

Abstract

We give a faithful interpretation of Bayesian networks into a version of numerical argumentation networks based on Łukasiewicz infinite-valued logic with product conjunction. The advantages of such a translation, beyond the theoretical aspects of it, are hopefully threefold: 1) importing updating algorithms into argumentation networks; 2) importing the handling of loops into cyclic Bayesian networks; and 3) importing logical proof theory into Bayesian networks.

Keywords: Bayesian Networks, Argumentation Theory, Numerical Networks

1 Introduction

In this paper, we compare probabilistic argumentation with Bayesian networks and motivate the new definition of Bayesian Argumentation Networks. We examine what extra features are needed to extend traditional abstract argumentation frameworks to enable the extended frameworks to simulate Bayesian networks. Once we identify such features, then we can call the extended argumentation frameworks by the name

Bayesian Argumentation Networks. We shall see later that the extra features are all well-known features existing in the literature in various contexts.

In order to illustrate these ideas, consider the network $\langle S, R\rangle$ of Fig. 1. In this figure, the arrows just indicate parenthood. If the arrows are considered as attacks, then $\langle S, R\rangle$ is a traditional abstract argumentation framework (henceforth a "Dung network"), and there is only one complete extension $E = \{X, Y, U, W\}$ (with $A =$ "out").

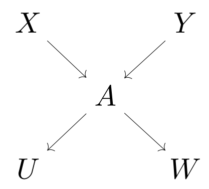


Figure 1: A sample argumentation network.

The operating assumptions (which are violated in Bayesian networks) are:

1. Since there is no connection (i.e., attack) going into X and into Y, then X and Y are "in" (intuitively meaning $X = Y = \top$).
2. Since U and V have the same parent A, we treat U and V in the same way.
3. We do not mind having cycles, i.e., R need not be acyclic.

There are other assumptions in the case of argumentation networks, but let us concentrate only on the ones above.

Bayesian networks do not allow for cycles (R must be acyclic) and they do not determine the values of nodes without parents, such as X and Y. Moreover, they are not committed to treating nodes with the same parents (such as U and W) the same way. Such a view is not new to argumentation. In fact such a view is shared by Abstract Dialectical Frameworks (ADFs) [6]. In an ADF, each node α depends on its parents, say $\{\beta_1, \ldots, \beta_n\}$ via a Boolean formula Ψ_α specific to α. Thus, we have that α depends on $\Psi_\alpha(\beta_1, \ldots, \beta_n)$ in the ADF case and we want that

$$\alpha \leftrightarrow \Psi_\alpha(\beta_1, \ldots, \beta_n).$$

In Dung's networks, the same constraint Ψ_α is imposed on all nodes α, namely

$$\alpha \leftrightarrow \bigwedge_{i=1}^{n} \neg\beta_i$$

where "$\wedge\neg$" is the Peirce-Quine connective

$$\downarrow\{E_i\} = \wedge_{i=1}^{n}\neg E_i{}^{1}$$

Still, we do not have many options when we treat the network of Fig. 1 as an ADF. Since X and Y depend on the empty set, then each can be either $\top$ or $\bot$. A depends on X and Y, and these being $\top$ or $\bot$ allow for A to be either $\top$ or $\bot$ and similarly for U and W. So depending on the Boolean functions Ψ_α employed, we can get all possible distributions of $\{\bot, \top\}$ among the nodes in Fig. 1.[2]

If we allow source nodes such as X and Y to have arbitrary given values in $[0, 1]$ and are able to describe the desired dependencies between node values in an argumentation context, then we can bridge the gap between the Bayesian and argumentation representations and hence analyse the properties of the former under the perspective of the latter. This can bring benefits to both areas which we will discuss later.

We find that the probabilistic approach to argumentation is the nearest we can get to Bayesian networks. We identify that what is missing in the probabilistic approach is a representation of conditional probabilities, a feature which is central in Bayesian networks. We further realise that if we define new argumentation networks based on joint attacks defined numerically using Łukasiewicz infinite-valued logics, we will have what we need. The integration of conditional probabilities and joint attacks is one of the objectives of this paper.

The rest of the paper is structured as follows. We start with a description of the probabilistic approach to argumentation in Section 2. The section is written in a Socratic manner, leading the reader to our conclusions using examples and semi-formal definitions. Section 3 gives the formal definitions in a systematic manner. Section 4 contains a comprehensive example illustrating what we have done. Section 5 deals with complexity issues. Section 6 discusses related literature [3, 4, 9, 16, 5, 19, 20, 8, 21, 22, 24, 25, 27]. In Section 7, we conclude with a discussion and directions for future research.

[1]Notice that the other boolean connectives can be defined in terms of $\downarrow$, for instance, $\neg P \equiv P \downarrow P$.

[2]As pointed out by one of the referees, the reader might think that this is a shortcoming of ADFs in the sense that initial arguments, being dependent on the empty set, can only have a fixed value. However, there is also the possibility to consider all initial arguments A as self-looping with acceptance condition A. In this case most semantics then yield a "guessing" value for A. We should however be cautious in not allowing too many modifications. It is known that enough modifications can reduce ADFs to traditional argumentation systems (see [14]).

2 Background Discussion

This section develops in a Socratic manner the features we need to reach a proper representation of a Bayesian network as an extension of an argumentation network. We therefore turn to the probabilistic approach to argumentation, it being the nearest to Bayesian networks. We need some notation before we describe it. As a starting point, we consider the elements of $S = \{X, Y, A, U, V\}$ as classical propositional atoms capable of getting the values $\alpha = \top$ (corresponding to α is "in") and $\alpha = \bot$ (corresponding to α is "out"). Let us understand the term *full conjunction of literals* to mean a conjunction containing for each atom α of the language (which is assumed to be finite) either α or $\neg\alpha$. Any full conjunction of literals of the form

$$\boldsymbol{e} = \wedge_i \alpha_i^{\pm}$$

can be considered a classical model $m(\boldsymbol{e})$. We have

$$m(\boldsymbol{e}) \models \alpha \text{ iff } \boldsymbol{e} \vdash \alpha$$

We can also associate with $\boldsymbol{e}$ a subset $S_{\boldsymbol{e}}$ of S, $S_{\boldsymbol{e}} = \{\alpha \in S \mid \boldsymbol{e} \vdash \alpha\}$ (remember that the elements of S are atoms without negation). So if we assign probability distributions π on the models $m(\boldsymbol{e})$ or on the set of full conjunctions of literals $\{\boldsymbol{e}\}$, we get a traditional probability function π on the space $\Omega = 2^{2^S}$ = families of models = $2^{\{\boldsymbol{e}\}}$ = the set of all propositional well-formed formulae (wffs) built-up from the atoms of S.

In our example, $S = \{X, Y, A, U, V\}$. 2^S = all subsets of S = all models of the language S. $\Omega = 2^{2^S}$ = all possible sets of models. So π gives a value $0 \leq \pi(m) \leq 1$, for each model m of S. We have that $\sum_m \pi(m) = 1$.

According to [13], the probability $\pi(m)$ for a typical conjunctive model $m(\boldsymbol{e})$ where $\boldsymbol{e} = \wedge_{\alpha \in S} \alpha^{\pm}$ can be given in two main ways.

i) The semantic way, which gives values $\pi(m(\boldsymbol{e}))$ directly for each $\boldsymbol{e}$.

ii) The syntactic way, which gives values $\pi(\alpha)$, for each $\alpha \in S$, and then $\pi(m(\boldsymbol{e}))$ is defined as the product

$$\prod_{\alpha} \pi^{\pm}(\alpha)$$

where $\pi^{+}(\alpha) = \pi(\alpha)$ and $\pi^{-}(\alpha) = 1 - \pi(\alpha)$.

So for example in Fig. 1, we either give probability directly to each model, e.g., to $\boldsymbol{e} = X \wedge Y \wedge \neg A \wedge U \wedge \neg W$ or give probabilities to each of X, Y, A, U and W, and then the probability of $\boldsymbol{e}$ can be calculated as

$$\pi(X) \cdot \pi(Y) \cdot (1 - \pi(A)) \cdot \pi(U) \cdot (1 - \pi(W))^3$$

The version of the probabilistic approach to both Dung or ADF which can be compared with the Bayesian approach is the syntactical one, *ii)* above, the one which assigns probabilities to nodes (not the one that assigns probabilities to the subnetworks). Thus each of the nodes X, Y, A, U and V is assigned a probability value.

The most general case of getting such probability is to regard the arguments as atoms in a space (i.e., $S = \{X, Y, A, U, V\}$) and assign probabilities to the subsets of S. This is a traditional probability distribution. Note that the subsets $E \subseteq \Omega = 2^{2^S}$ can also be identified with sets of models of formulas built-up using atoms from S.

We now show a connection of probabilistic argumentation with Bayesian argumentation. Both Dung and ADFs would read Fig. 1 as follows. The figure suggests the probability space being the family $\Omega = 2^{2^S}$ of all subsets of $S = \{X, Y, A, U, V\}$ and the connections (arrows) in the figure suggest restriction on the probability π on Ω. We want to consider only those probabilities which satisfy for every $\alpha \in S$ with parents $\{\beta_1, \ldots, \beta_n\}$ the following:

$$\pi(\alpha) = \pi(\psi_\alpha(\beta_1, \ldots, \beta_n))$$

Remember that each wff defines a set of models in which it holds, and π is a probability on sets of models. Thus π gives a number $0 \leq \pi(m) \leq 1$ to each model m of the language of S with $\sum_m \pi(m) = 1$.

Bayesian argumentation looks at the elements of S as random variables capable of getting $\top$ or $\bot$, and regards all probability functions $P(\alpha_1, \ldots, \alpha_n)$ where $\{\alpha_i\} = S$.

In our case we have probability functions $\boldsymbol{P}(X, Y, A, U, V)$. Thus for each combination of values of $\top$ or $\bot$ to the variables in S, $\boldsymbol{P}$ will give a probability.

Such a combination can also be viewed as a model m for the language of S, and so $\boldsymbol{P}$ gives probability to models. This is the same as π, but the restrictions on $\boldsymbol{P}$ and the manipulation of $\boldsymbol{P}$ are different in this case. We have, according to the Bayesian view, that Fig. 1 gives the dependencies of the variables on each other. Let $\{\beta_1, \ldots, \beta_n\}$ be all the parents of α and $\boldsymbol{P}(\alpha|Z)$ denote the conditional probability of α given Z. We have the following equations:

$$\boldsymbol{P}(\alpha) = \boldsymbol{P}(\alpha \mid \beta_1, \ldots, \beta_n) \cdot \boldsymbol{P}(\beta_1, \ldots, \beta_n)$$

[3]See [13].

If α has no parents, then $\boldsymbol{P}(\alpha)$ must be given. If α does have parents $\{\beta_1, \ldots, \beta_n\}$, then $\boldsymbol{P}(\alpha \mid \beta_1, \ldots, \beta_n)$ must be given. $\boldsymbol{P}(\alpha \mid \beta_1, \ldots, \beta_n)$ can be given as a function giving a value in $[0, 1]$ for every choice of $\top, \bot$ to each β_i.

Thus the Bayesian approach specifies a syntactical type probability on the atoms $\alpha \in S$, by using the graph of the network and giving conditional probabilities for the dependencies of the graph.

So for the network of Fig. 1 we need the following values to specify a specific Bayesian distribution $\boldsymbol{P}$:

- Values of the probabilities of the source nodes X and Y, i.e., $\boldsymbol{P}(X)$ and $\boldsymbol{P}(Y)$.
- A table of values v, describing the coefficients of the function $\boldsymbol{P}(A|X, Y)$. We use the notation $F_{A|11}$ for the case $A|X \wedge Y$, $F_{A|10}$ for the case $A|X \wedge \neg Y$, etc. We denote the transmission coefficient for each case as $e_{F_{A|11}}$, $e_{F_{A|10}}$, and so on:

X	Y	v
$\top$	$\top$	$e_{F_{A\|11}}$
$\top$	$\bot$	$e_{F_{A\|10}}$
$\bot$	$\top$	$e_{F_{A\|01}}$
$\bot$	$\bot$	$e_{F_{A\|00}}$

- A table for $\boldsymbol{P}(U|A)$:

A	v
$\top$	$e_{F_{U\|1}}$
$\bot$	$e_{F_{U\|0}}$

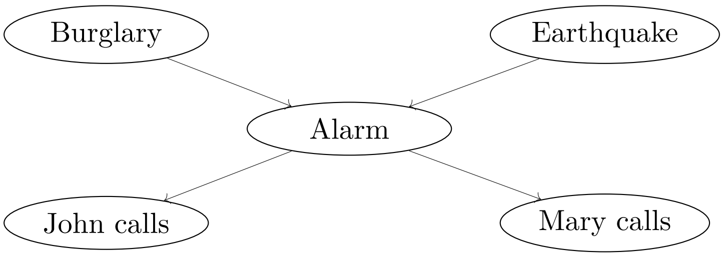
- A table for $\boldsymbol{P}(W|A)$:

A	v
$\top$	$e_{F_{W\|1}}$
$\bot$	$e_{F_{W\|0}}$

Note that the network of Fig. 1 actually depicts Pearl's famous Earthquake example [20] as described in the book "Bayesian Artificial Intelligence" by Korb and Nicholson [17]:

"You have a new burglar alarm installed. It reliably detects burglary, but also responds to minor earthquakes. Two neighbors, John and Mary, promise to call the police when they hear the alarm. John always calls when he hears the alarm, but sometimes confuses the alarm with the phone ringing and calls then also. On the other hand, Mary likes loud music and sometimes doesn't hear the alarm. Given evidence about who has and hasn't called, you'd like to estimate the probability of a burglary."

Replacing X with "Burglary", Y with "Earthquake", A with "Alarm", U with "John calls", and W with "Mary calls", gives the Bayesian network:



For simplicity, we will continue to use the letters X, Y, A, U and W.

Let us assume the following values. $\boldsymbol{P}(X) = 0.01$, giving $\boldsymbol{P}(X = \top) = 0.01$, and $\boldsymbol{P}(X = \bot) = 0.99$; $\boldsymbol{P}(Y = \top) = 0.02$, giving $\boldsymbol{P}(Y = \bot) = 0.98$; and the values given by the tables below:

X	Y	v
$\top$	$\top$	$e_{F_{A\|11}} = 0.95$
$\top$	$\bot$	$e_{F_{A\|10}} = 0.94$
$\bot$	$\top$	$e_{F_{A\|01}} = 0.29$
$\bot$	$\bot$	$e_{F_{A\|00}} = 0.001$

A	v
$\top$	$e_{F_{U\|1}} = 0.9$
$\bot$	$e_{F_{U\|0}} = 0.05$

A	v
$\top$	$e_{F_{W\|1}} = 0.70$
$\bot$	$e_{F_{W\|0}} = 0.01$

We can now compute the probabilities $\boldsymbol{P}(A)$, $\boldsymbol{P}(U)$ and $\boldsymbol{P}(W)$.

$$\begin{aligned}
\boldsymbol{P}(A) &= e_{F_{A|11}} \times \boldsymbol{P}(X) \times \boldsymbol{P}(Y) + \\
&\quad e_{F_{A|10}} \times \boldsymbol{P}(X) \times (1 - \boldsymbol{P}(Y)) + \\
&\quad e_{F_{A|01}} \times (1 - \boldsymbol{P}(X)) \times \boldsymbol{P}(Y) + \\
&\quad e_{F_{A|00}} \times (1 - \boldsymbol{P}(X)) \times (1 - \boldsymbol{P}(Y)). \\
\boldsymbol{P}(W) &= e_{F_{W|1}} \times \boldsymbol{P}(A) + e_{F_{W|0}} \times (1 - \boldsymbol{P}(A)). \\
\boldsymbol{P}(U) &= e_{F_{U|1}} \times \boldsymbol{P}(A) + e_{F_{U|0}} \times (1 - \boldsymbol{P}(A)).
\end{aligned}$$

So

$$\begin{aligned}
\boldsymbol{P}(A) &= 0.95 \times 0.01 \times 0.02 + \\
&\quad 0.94 \times 0.01 \times 0.98 + \\
&\quad 0.29 \times 0.99 \times 0.02 + \\
&\quad 0.001 \times 0.99 \times 0.98. \\
&= 0.00019 + 0.009212 + 0.005742 + 0.0009702 \\
&= 0.0161142 \approx 0.016
\end{aligned}$$

So $\boldsymbol{P}(\neg A) \approx 0.984$. Now for U and W, we get

$$\begin{aligned}
\boldsymbol{P}(W) &= (0.7 \times 0.016) + (0.01 \times 0.984) = 0.0112 + 0.00984 \approx 0.021. \\
\boldsymbol{P}(U) &= (0.9 \times 0.016) + (0.05 \times 0.984) = 0.014 + 0.0492 \approx 0.063.
\end{aligned}$$

Thus we can see that we have a syntactical probability distribution on S.

2.1 A Common Ground for Bayesian, Argumentation and Abstract Dialectical Frameworks

To compare Bayesian networks with say ADFs or with traditional Dung networks, we need to go to a common ground. First we note that with any formal system, whether it be a logic such as classic or intuitionistic logic, or whether it be a Bayesian, ADF or traditional argumentation network there are always two components. The first one is the intended meaning of the system. The second is the formal mathematical representation of the system and the mathematical machinery of handling it. When we compare two such systems we can compare them in regard to their formal machinery or we can compare them in their intended meaning. It may be that two systems have the same formal machineries but completely different meanings. This happens a lot in modal logics with possible world semantics. In the case of a network $\langle S, R\rangle$, the intended meaning may impose some restrictions on the graph. In an argumentation network the arrows mean attack; the variables get values "in", "out" and "undecided"; and unattacked nodes must get value "in". In Bayesian networks the arrows represent dependencies and there is the requirement of the network being acyclic. In addition, nodes with the same parents in Bayesian networks can behave differently, which is not the case in the traditional argumentation but is the case in ADFs. In ADFs the arrows represent dependencies and there is no requirement of being acyclic. Having said all that let us now compare the systems on the basis of their mathematical machinery which can be captured by the two points below.

a. Since Bayesian networks allow for points without parents to have an arbitrary probability assigned to them, Bayesian networks can agree to limit such assignment for the sake of common grounds with argumentation and assign probability 1, we can assume a similar sacrifice and the same property for ADFs. If, on the other hand, we want to leave Bayesian networks as they are (not ask them to make any limitations) but we still want to have common ground with respect to this property with ordinary Dung networks, we can modify Dung's networks, and add for each node α a new node called $\neg\alpha$, with α and $\neg\alpha$ attacking each other. This will allow any node α which was originally unattacked to get any value in the modified network, because it will be part of the cycle $\{\alpha, \neg\alpha\}$.

   We can also assume, for Bayesian networks, the sacrifice limitation that nodes with the same parents behave the same (we can call these Bayesian networks *BNA nets*.[4] We can also add this requirement (that nodes with the same parents behave the same way) as an additional assumption on ADFs to make them more in line with traditional Dung networks.

b. Bayesian networks are acyclic and since ADFs and traditional Dung networks can also be acyclic, let us accept this additional restriction on them.

So we compare acyclic probabilistic ADFs with BNA nets and see what else we need to add to argumentation networks to be able to implement Bayesian networks in them.

The above discussion outlined several possibilities for finding common grounds between Bayesian networks and Dung's networks. We made what we think is the best choice/approach, which we now proceed to explain.

[4]The letters "NA" stand for "nice-to-argumentation".

Looking again at Fig. 1, we see that what is missing in order to do a proper comparison is the fact that Bayesian networks have the conditional probabilities. Let us look at Fig. 2, which is the top part of Fig. 1.

$$\begin{tikzcd}
X \arrow[dr] & & Y \arrow[dl] \\
& A &
\end{tikzcd}$$

Figure 2: The top part of the network in Fig. 1.

What the Bayesian approach does is to give arbitrary values $\boldsymbol{P}(X)$, $\boldsymbol{P}(Y)$ (so we have syntactical probabilities for X and Y), but to get $\boldsymbol{P}(A)$, it uses transmission values as given in Fig. 3. $e_{F_{A|ij}}$ are transmission coefficients in the sense of [1, 2] and $\boldsymbol{F}_{ij}$ is an attack formation in the sense of [12], to be explained below and formally defined in Section 3.

$$\begin{tikzcd}
\boldsymbol{F}_{11} : X \wedge Y \arrow[drr, "e_{F_{A|11}}"'] & \boldsymbol{F}_{10} : X \wedge \neg Y \arrow[dr, "e_{F_{A|10}}"] & & \boldsymbol{F}_{01} : \neg X \wedge Y \arrow[dl, "e_{F_{A|01}}"'] & \boldsymbol{F}_{00} : \neg X \wedge \neg Y \arrow[dll, "e_{F_{A|00}}"] \\
& & A & &
\end{tikzcd}$$

We have $\boldsymbol{P}(A) = \sum e_{ij}\boldsymbol{P}(\boldsymbol{F}_{ij})$.

Figure 3: An argumentation network with attack formations, transmission coefficients and joint attacks.

So what we need to accommodate Bayesian networks are argumentation networks with attack formations, transmission coefficients, and joint attacks obeying the attack formula of Fig. 3. The semantics of such networks is best given using the *equational approach* [15].[5]

[5]There are several different interpretations of basic argumentation notions. In traditional Dung networks arcs represent attacks while in ADFs they represent dependencies It is natural then to think that ADFs are closer in meaning to Bayesian networks because certainly arcs in Bayesian networks are not attacks but dependencies. Our reader may therefore be puzzled at our translation of Bayesian networks into argumentation where arcs represent attacks. We even use joint attacks. We remind the reader that ADFs can be translated into traditional argumentation networks using joint attacks and additional nodes. The additional nodes are used to help simulate the boolean dependencies (see [14]). It may be possible to translate Bayesian networks into ADFs, but we would need additional points and some kind of fuzzy propagation. It therefore makes more sense to translate directly into traditional networks with attacks. Note also that numerical values associated with nodes can have several interpretations: 1) a fuzzy truth-value; 2) a probability value expressing uncertainty about argument acceptance; 3) a value obtained in the context of the equational

We now explain the components needed.

a. Attack formations

Consider Fig. 4 (L) and Fig. 4 (R). In Fig. 4 (L), we have that α attacks β. We have

(a) If $\alpha =$ "in", then $\beta =$ "out"

(b) If $\alpha =$ "out", then $\beta =$ "in" (unless β is attacked by something else that is not "out")

(c) If $\alpha =$ "undecided", then $\beta =$ "undecided" (unless it is attacked by something else that is "in", in which case β has to be "out")

$$\begin{tikzcd}
\alpha \arrow[d] & \alpha \arrow[d] \\
\beta & X_{\alpha,\beta} \arrow[d] \\
 & Y_{\alpha,\beta} \arrow[d] \\
 & \beta \\
\text{(L)} & \text{(R)}
\end{tikzcd}$$

Figure 4

In Fig. 4 (R), we have two intermediary points unique to the pair (α, β). We view this as an *attack formation*. It does the job of Fig. 4 (L). (a), (b) and (c) still hold for Fig. 4 (R). Fig. 4 (R) is a general replacement for Fig. 4 (L), used by Gabbay in [14]. It allows for the implementation of higher level attacks.[6]

approach as the result of some calculation. 4) a Bayesian probability value. The conditions when two of such values coincide need to be investigated. For example, we do know that for the Eq_{inv} equational approach the numerical values can be viewed as probabilistic values where the arguments are mutually independent [13].

[6]Higher level attacks are attacks on attacks. So for example, an academic professional argument β put forward in favour of promoting three members of staff to the rank of full professor by virtue of

For example the attack $\gamma \to (\alpha \to \beta)$ can be implemented as $\gamma \to Y_{\alpha,\beta}$. The two additional dummy points $X_{\alpha,\beta}$ and $Y_{\alpha,\beta}$ are just two "transmitting points", which allow the node γ to attack the transmission by attacking the point $Y_{\alpha,\beta}$.

We do not need higher level attacks in this paper but it is useful to know how useful attack formations are in translations/implementations. We use the notation $\boldsymbol{F}[\alpha, \beta]$ as in Fig. 5.

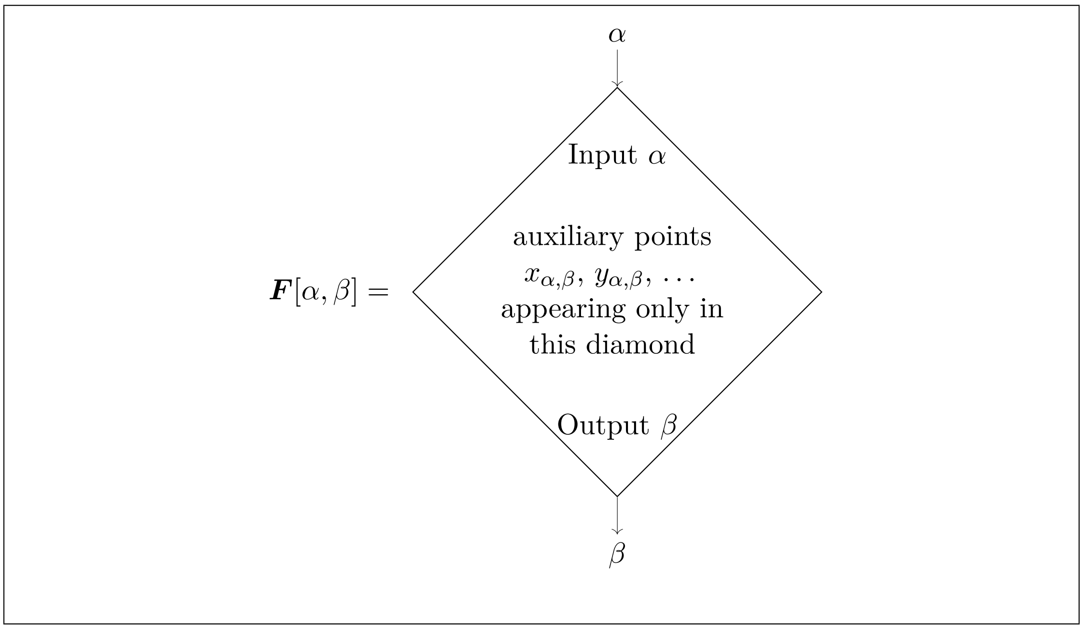


Figure 5

Attack formations can be single arguments as in Fig. 6. The argument A is both the input and the output point of the formation.

Attack formations can attack each other, as in Fig. 7. The output point of formation $\boldsymbol{F}_1$ attacks the input point of formation $\boldsymbol{F}_2$.

b. Transmission coefficients

their brilliant performance may be attacked by an argument α which says that there is not enough money to pay their higher salaries and benefits if indeed promoted. If the claims of both arguments are true (i.e., the individuals did perform well and indeed there is not enough money to pay for the extra expenditure caused by the promotion, there is nothing to say except to put forward an argument γ which says that budgetary considerations should not be arguments against promotion. Here γ is a higher level attack on the very attack arrow $\alpha \to \beta$. We write this as $\gamma \to (\alpha \to \beta)$.

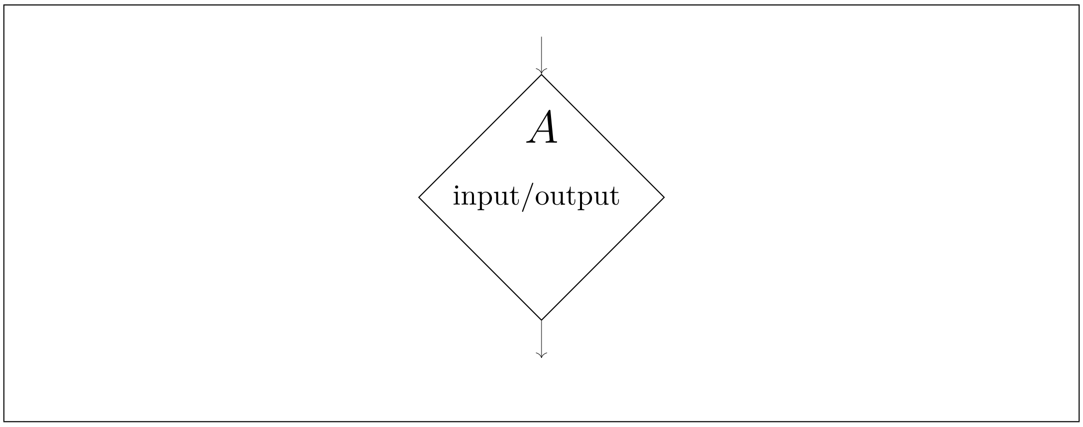


Figure 6

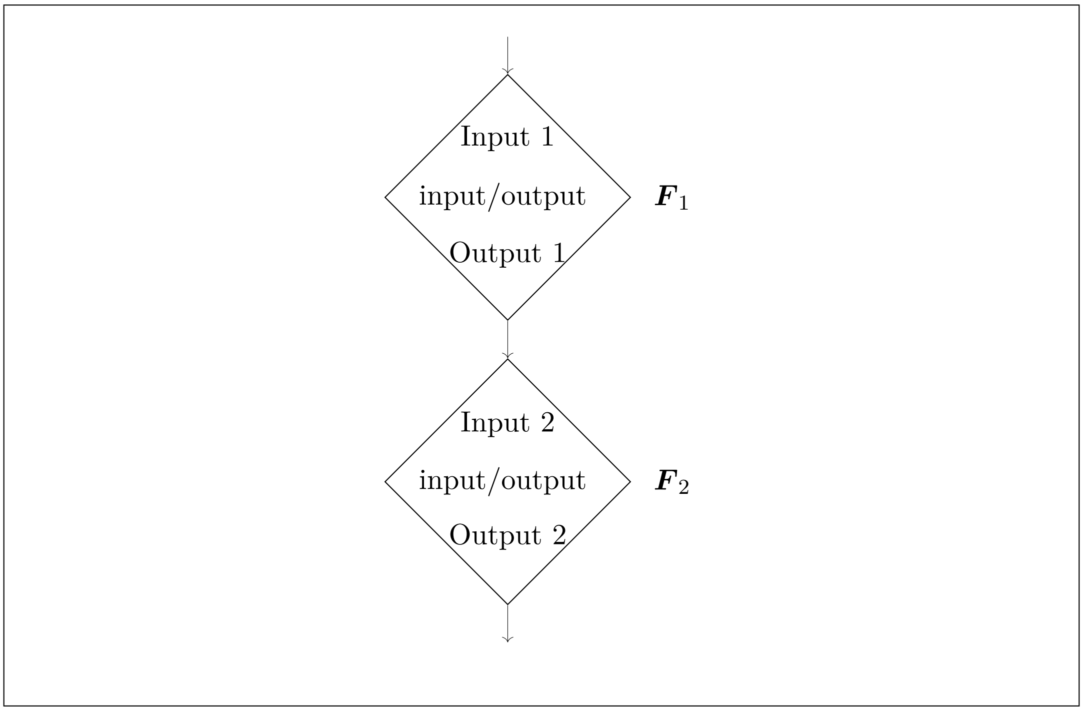


Figure 7: An attack formation attacking another attack formation.

$$\begin{tikzcd}
\beta_1 \arrow[drr] & \beta_2 \arrow[dr] & \cdots & & \beta_k \arrow[dll] \\
& & \alpha & &
\end{tikzcd}$$

Figure 8: A typical attack configuration in an argumentation network.

Consider Fig. 8, under the equational approach of [15]. Let $\beta_1, \ldots, \beta_k$ be all of the attackers of node α. Each node has a numerical value, $f(\alpha)$, $f(\beta_1)$, $\ldots$, $f(\beta_k)$. The equational approach (under the Eq_{inv} Equational semantics) requires that

$$f(\alpha) = \prod_{i=1}^{k}(1 - f(\beta_i)) \tag{1}$$

The equational approach obtains f as a solution to the equations of type (1), for every $\alpha \in S$, and *at least* all the preferred extensions (in Dung's sense) are obtained via the correspondence:

(a) $\alpha =$ "in", if $f(\alpha) = 1$

(b) $\alpha =$ "out", if $f(\alpha) = 0$

(c) $\alpha =$ "undecided", if $0 < f(\alpha) < 1$

Such solutions also can be seen as syntactical probability distributions for the nodes as shown in [13]. So, if we implement Bayesian networks in argumentation networks with the Eq_{inv} interpretation, we hope to get the Bayesian probabilities as preferred extensions.

When we have a transmission coefficient e_i between the attacker β_i and α, the strength of the attack from β_i is adjusted by the coefficient e_i, and hence its value is only $e_i \times \beta_i$. Thus, we get for Fig. 9 of attacks with transmission coefficients that

$$f(\alpha) = \prod_{i=1}^{k}(1 - e_i \times f(\beta_i)) \tag{2}$$

It is worth noting that the transmission coefficient does not change the expressive power of Eq_{inv}, since the effects of the coefficients can be implemented through additional nodes in Eq_{inv}.

c. Joint attacks

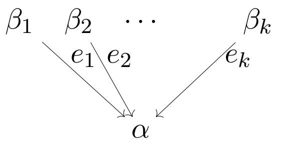


Figure 9: A typical attack configuration in an argumentation network with transmission coefficients.

In [14], Gabbay et. al. used the notation of Fig. 10 in the context of Fibring networks, joint attacks and disjunctive attacks. The idea of joint attacks on its own was earlier introduced in [18]. The intended meaning of a joint attack is that α is "out", if all β_i are "in". In a numerical context (i.e., under an equational approach), we can write the equation

$$f(\alpha) = 1 - \prod_i f(\beta_i)$$

Clearly α is "out" exactly when all of β_i are "in".

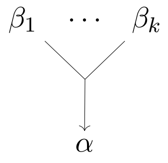


Figure 10: A joint attack from β_1,...,β_k to α.

Remark 1. *Note that if we have such joint attacks (as given in [18]) under the equational approach, we can implement products (π-attacks) with the help of auxiliary points. For example, the value of α as the product of β_1 and β_2 in Fig. 11 can be implemented as Fig. 12 and vice-versa. In Fig. 12 we have that $f(x) = 1 - \beta_1 \cdot \beta_2$ and $f(\alpha) = 1 - x = \beta_1 \cdot \beta_2$.*

Fig. 13 can be implemented as Fig. 14. In Fig. 13, we have that $f(\alpha) = 1 - \beta_1 \cdot \beta_2$. In Fig. 14, $f(y) = \beta_1 \cdot \beta_2$ and $f(\alpha) = 1 - y = 1 - \beta_1 \cdot \beta_2$.

For implementing Bayesian networks we need a different understanding for joint attacks, yielding a different equation. What we need is the following equation:

$$f(\alpha) = \min(1, \textstyle\sum_j (1 - f(\beta_j))) \tag{3}$$

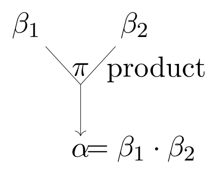


Figure 11

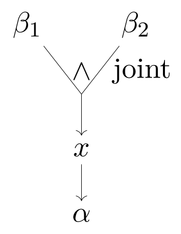


Figure 12

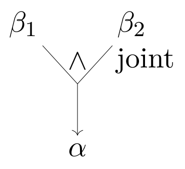


Figure 13

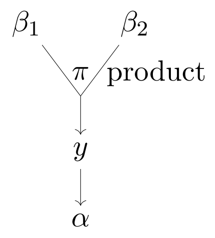


Figure 14

From (3), one can see that only if all $f(\beta_j) = 1$, do we get $f(\alpha) = 0$.

So we are using the equation below for the joint attacks of Fig. 10.

$$\alpha = \min(1, \textstyle\sum_j 1 - \beta_j)$$

We make some comments about this equation.

(a) First, note that this equational truth-table is definable in Łukasiewicz logic [23]. In this logic, propositions get value in $[0, 1]$, 1 represents truth and 0 represents falsity. The truth-tables for $\neg$ and $\to$ are:

$$\begin{array}{rcl} \neg X & = & 1 - X \\ X \to Y & = & \min(1, 1 - X + Y) \end{array}$$

Therefore,

$$X \to \neg Y = \min(1, 1 - X + 1 - Y)$$

Let φ be a new connective operating on a non-empty list $[X_1, \ldots, X_n]$[7] defined as follows.

$$\begin{array}{rcl} \varphi([X_1]) & = & \neg X_1 \\ \varphi([X_1, X_2]) & = & X_2 \to \varphi([X_1]) \end{array}$$

By induction, assume $\varphi(X_1, \ldots, X_n)$ is definable and satisfies

$$\varphi(X_1, \ldots, X_n) = min(1, \sum_i (1 - X_i))$$

Then

$$\begin{array}{rcl} X_{n+1} \to \varphi(X_1, \ldots, X_n) & = & \min(1, 1 - X_{n+1} + \varphi(X_1, \ldots, X_n)) \\ & = & \min(1, \sum_{i=1}^{n+1} (1 - X_i)) \\ & = & \varphi(X_1, \ldots, X_{n+1}) \end{array}$$

Note that $\varphi(X_1) = \min(1, 1 - X_1) = \neg X_1$. So we have

$$\begin{array}{rcl} \varphi(X_1) & = & \neg X_1 \\ \varphi(X_1, \ldots, X_n, X_{n+1}) & = & X_{n+1} \to \varphi(X_1, \ldots, X_n) \end{array}$$

(b) Also note that argumentation networks with a formula of the type of φ just defined for joint attacks are not definable through the traditional Dung semantics. This gives new semantics. Consider Fig. 15.

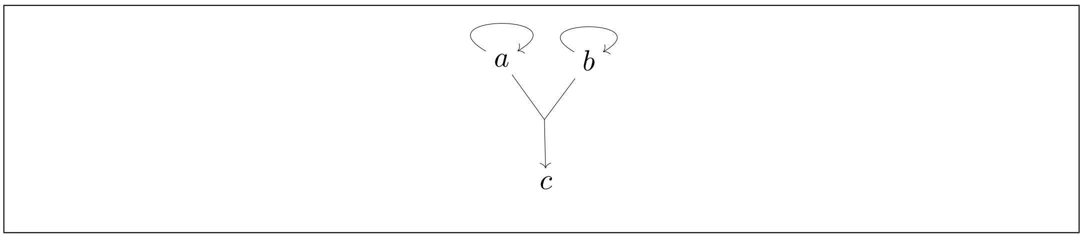
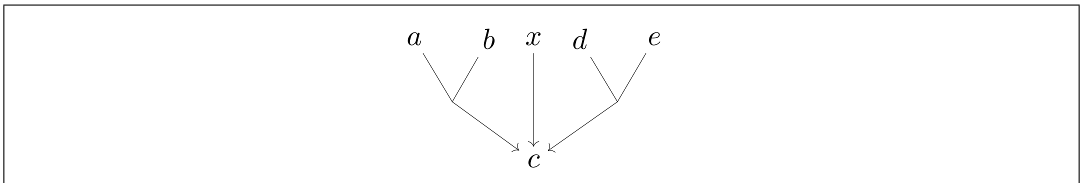


Figure 15



$$\begin{aligned} c &= \min(1, 1-a+1-b) \times \min(1, 1-x) \times \min(1, 1-d+1-e) \\ &= \min(1, 1-a+1-b) \times (1-x) \times \min(1, 1-d+1-e) \end{aligned}$$

Figure 16

Use only φ. We get $a = \frac{1}{2}$, $b = \frac{1}{2}$, $c = \min(1, \frac{1}{2} + \frac{1}{2}) = 1$. So $c =$ "in".

The rationale behind this semantics is as follows.

We reject c if there are joint attacks on c where all the attackers are "in". If some attackers are "undecided", then so is c. However, if too many attackers are "undecided" (and remember this is a consortium joint attack where too many members of the consortium are undecided), then we disregard the attack and let $c =$ "in".

To get a better idea of how Eq_{inv} with Łukasiewicz joint attacks works, compare Figures 16 and 17.

The equation in Fig. 17 is given by Eq_{inv}, for β_1 being the joint attack of $\{a, b\}$, β_2 being the attack of x, and β_3 being the joint attack of $\{d, e\}$. The joint attacks of β_1 and β_3 are calculated each according to Equation 3 (see page 15). So going back to Fig. 16, under the above considerations we get

$$c = \min(1, 1-a+1-b) \times (1-x) \times \min(1, 1-d+1-e).$$

[7]For example conjunction is an operator that can be seen to be operating on a list, where $\bigwedge([X]) = X$ and $\bigwedge([X_1, \ldots, X_n]) = X_n \bigwedge([X_1, \ldots, X_{n-1}])$.

$$\begin{tikzcd}
\beta_1 \arrow[dr] & \beta_2 \arrow[d] & \beta_3 \arrow[dl] \\
 & c &
\end{tikzcd}$$

$$\begin{aligned}
c &= \min(1, 1-\beta_1) \times \min(1, 1-\beta_2) \times \min(1, 1-\beta_3) \\
&= (1-\beta_1) \times (1-\beta_2) \times (1-\beta_3) \\
&= \prod_i (1-\beta_i)
\end{aligned}$$

Figure 17

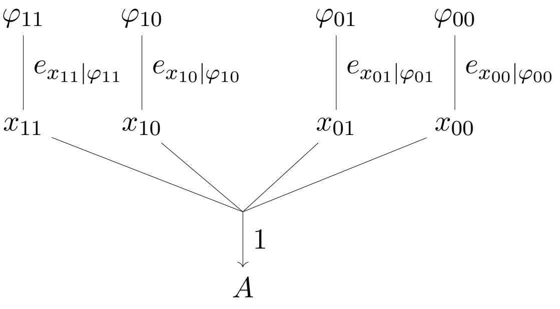

φ_{11}
φ_{10}
φ_{01}
φ_{00}
$e_{x_{11}|\varphi_{11}}$
$e_{x_{10}|\varphi_{10}}$
$e_{x_{01}|\varphi_{01}}$
$e_{x_{00}|\varphi_{00}}$
x_{11}
x_{10}
x_{01}
x_{00}
1
A

Figure 18: A complex configuration of joint attacks with transmission factors.

2.2 Combining It All

Consider Fig. 18.

The equational approach will give a solution f to this configuration as

$$f(x_{ij}) = 1 - e_{ij} \times f(\varphi_{ij})$$

and

$$\begin{aligned}
f(A) &= \min\left(1, \sum(1 - f(x_{ij}))\right) \\
f(A) &= \min\left(1, \sum(1 - (1 - e_{ij} \times f(\varphi_{ij})))\right) \\
f(A) &= \min\left(1, \sum e_{ij} \times f(\varphi_{ij})\right)
\end{aligned}$$

Now look again at Fig. 3. If we can instantiate φ_{ij} by an appropriate attack

$$\boldsymbol{F}_{00}[\neg X, \neg Y] = \left\{ \begin{tikzcd} \neg X \arrow[r, leftrightarrow] & X \arrow[dr, "1"'] & & Y \arrow[dl, "1"] \arrow[r, leftrightarrow] & \neg Y \\ & & c_{00} & & \end{tikzcd} \right.$$

$$c_{00} = (1 - \boldsymbol{P}(X)) \cdot (1 - \boldsymbol{P}(Y))$$

$$\boldsymbol{F}_{01}[\neg X, Y] = \left\{ \begin{tikzcd} \neg X \arrow[r, leftrightarrow] & X \arrow[dr, "1"'] & & Y \arrow[r, leftrightarrow] & \neg Y \arrow[dll, "1"] \\ & & c_{01} & & \end{tikzcd} \right.$$

$$c_{01} = (1 - \boldsymbol{P}(X)) \cdot (1 - 1 + \boldsymbol{P}(Y)) = (1 - \boldsymbol{P}(X)) \cdot \boldsymbol{P}(Y)$$

$$\boldsymbol{F}_{10}[X, \neg Y] = \left\{ \begin{tikzcd} \neg X \arrow[r, leftrightarrow] \arrow[drr, "1"'] & X & & Y \arrow[dl, "1"] \arrow[r, leftrightarrow] & \neg Y \\ & & c_{10} & & \end{tikzcd} \right.$$

$$c_{10} = (1 - 1 + \boldsymbol{P}(X)) \cdot (1 - \boldsymbol{P}(Y)) = \boldsymbol{P}(X) \cdot (1 - \boldsymbol{P}(Y))$$

$$\boldsymbol{F}_{11}[X, Y] = \left\{ \begin{tikzcd} \neg X \arrow[r, leftrightarrow] \arrow[drr, "1"'] & X & & Y \arrow[r, leftrightarrow] & \neg Y \arrow[dll, "1"] \\ & & c_{11} & & \end{tikzcd} \right.$$

$$c_{11} = (1 - 1 + \boldsymbol{P}(X)) \cdot (1 - 1 + \boldsymbol{P}(Y)) = \boldsymbol{P}(X) \cdot \boldsymbol{P}(Y)$$

Figure 19

formation $\boldsymbol{F}_{ij}$ such that

$$f(\varphi_{ij}) = f(\boldsymbol{F}_{ij}) = \boldsymbol{P}^{\pm}(X) \times \boldsymbol{P}^{\pm}(Y)$$

then we have implemented that figure as Fig. 18.

This is easy to do. Look at Fig. 19 and remember that since X, $\neg X$, Y and $\neg Y$ are end points attacking each other respectively, we can give them arbitrary probabilities! In Fig. 19, c_{ij} is the output point (realising $f(\varphi_{ij})$), and X and Y are given probabilities $\boldsymbol{P}(X)$ and $\boldsymbol{P}(Y)$, respectively. This gives probabilities $\boldsymbol{P}(\neg X) = 1 - \boldsymbol{P}(X)$ and $\boldsymbol{P}(\neg Y) = 1 - \boldsymbol{P}(Y)$.

Remark 2. *Note that the above implementation required two types of attacks. The product attack of Fig. 11 and the new joint attack of Fig. 10, namely*

$$\alpha = \min(1, \textstyle\sum_j (1 - \beta_j)).$$

We know that the new joint attack can be expressed in Łukasiewicz infinite-valued logic. Therefore in the extension of this logic with product conjunctions, namely with the additional connective $*$:[8]

$$x * y = x \cdot y$$

we can implement Bayesian networks!

3 Formal Definitions

This section will describe the formal machinery of Bayesian Argumentation Networks (BANs) and the mechanism to translate a Bayesian network into a BAN.

Definition 1 (Bayesian Argumentation Network (BAN))**.**

a. *A BAN has the form $\mathbb{B} = \langle S, R, \boldsymbol{e} \rangle$, where S is a non-empty set of arguments, $R \subseteq (2^S - \varnothing) \times S$ is the attack relation between non-empty subsets of S and an element of S. For each pair $(H, x) \in R$, such that $H \subseteq S$ and $x \in S$, $\boldsymbol{e}$ is a transmission function giving each h in the pair (H, x) a real value $\boldsymbol{e}(H, x, h) \in [0, 1]$.*

We can describe this situation in Fig. 20, where $(H, x) \in R$, $H = \{h_1, \ldots, h_k\}$ and $e_i = \boldsymbol{e}(H, x, h_i)$.

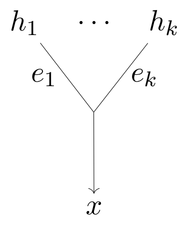
The general attack configuration of a node is depicted in Fig. 21, where $H^1 = \{h_1^1, \ldots, h_{k_1}^1\}, \ldots, H^i, \ldots, H^m = \{h_1^m, \ldots, h_{k_m}^m\}$ are all attackers of the node x. In such configuration $e_{ij} = \boldsymbol{e}(H^i, x, h_{ij})$.

b. *Let f be a function from S into $[0, 1]$. The equation associated with f and the configuration of Fig. 21 is*

$$f(x) = \prod_{j=1}^{m} \min(1, \sum_{i=1}^{k_j} (1 - e_{ij} \cdot f(h_i^j))) \tag{4}$$

c. *A solution to equation (4) of item b. is called an extension to $\mathbb{B}$.*

[8]See [10] for details.



$$\begin{aligned}
&H = \{h_1, \ldots, h_k\}\\
&(H, x, h_1) = e_1\\
&(H, x, h_2) = e_2\\
&\quad\vdots\\
&(H, x, h_k) = e_k
\end{aligned}$$

Figure 20

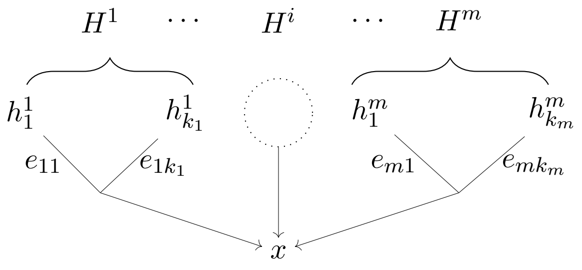


Figure 21

Example 1. *Consider Fig. 22. In this figure we have:*

$$\begin{aligned}
S &= \{a, b, c, d, e, x\}\\
R &= \{(\{a, b\}, c), (\{x\}, c), (\{d, e\}, c)\}\\
\boldsymbol{e}(\{a, b\}, c, a) &= e_1\\
\boldsymbol{e}(\{a, b\}, c, b) &= e_2\\
\boldsymbol{e}(\{x\}, c, x) &= e_3\\
\boldsymbol{e}(\{d, e\}, c, d) &= e_4\\
\boldsymbol{e}(\{d, e\}, c, e) &= e_5
\end{aligned}$$

The equation for c is

$$\begin{aligned}
f(c) = {} & \min(1, 1 - e_1 \cdot f(a) + 1 - e_2 \cdot f(b)) \times\\
& \min(1 - e_3 \cdot f(x)) \times\\
& \min(1, 1 - e_4 \cdot f(d) + 1 - e_5 \cdot f(e))
\end{aligned}$$

Take $e_i = 1$ and compare this with Fig. 16.

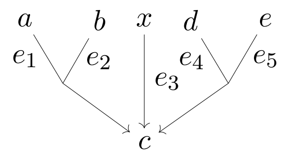


Figure 22

Definition 2. *A Bayesian network $\mathcal{N}$ has the form $\langle S, \varrho, \boldsymbol{P}\rangle$ where S is a set of nodes, $\varrho \subset S \times S$ is an* **acyclic** *dependence relation, and $\boldsymbol{P}$ gives probability distributions based on (S, ϱ) defined below. The symbol Ψ_x will denote the set of parents of the node x in $\mathcal{N}$, i.e., $\Psi_x = \{y \mid (y, x) \in \varrho\}$.*

- *The elements of S are considered Boolean variables which can be in only one of two states either true $= \top = 1$, or false $= \bot = 0$.*
- *If $x \in S$ is a source node, i.e., $\Psi_x = \varnothing$, then $\boldsymbol{P}(x) \in [0, 1]$ denotes the probability that $x = \top$. $\boldsymbol{P}(\neg x) = 1 - \boldsymbol{P}(x)$ denotes the probability that $X = \bot$.*
- *If $\Psi_x \neq \varnothing$, then $\boldsymbol{P}$ gives the conditional probability of x on $\Psi_x = \{y_1, \ldots, y_k\}$, denoted by $\boldsymbol{P}(x|\Psi_x)$. This means the following:*
  - a. *First for each $q \in S$, consider a new atom letter denoted by $\neg q$. Read $q^0 \overset{def}{=} \neg q$ and $q^1 \overset{def}{=} q$.*
  - b. *For each $\varepsilon \in 2^k$ (a vector of numbers $\varepsilon(i)$, $1 \leq i \leq k$ from {0,1}), we look at the option*
  $$\overrightarrow{y}(\varepsilon) = \wedge_i y_i^{\varepsilon(i)}$$
  where we read y_i^0 as $\neg y_i$ or $y_i = \bot$ and y_i^1 as y_i or $y_i = \top$.
  - c. *We can now state what $\boldsymbol{P}(x|\Psi_x)$ is. $\boldsymbol{P}(x|\Psi_x)$ gives values $\boldsymbol{P}(x, \Psi_x, \varepsilon) \in [0, 1]$, for each $\varepsilon \in 2^k$.*
- *Let x be a node such that $\Psi_x \neq \varnothing$. Assuming that the probabilities $\boldsymbol{P}(y_i)$, for $y_i \in \Psi_x$, are known, the probability $\boldsymbol{P}(x)$ of x can be calculated as*
  $$\boldsymbol{P}(x) = \sum_{\varepsilon} \boldsymbol{P}(x, \Psi_x, \varepsilon) \times \boldsymbol{P}(\overrightarrow{y}(\varepsilon))$$
  where $\boldsymbol{P}(\overrightarrow{y}(\varepsilon))$ is $\prod_i \boldsymbol{P}(y_i)^{\varepsilon(i)}$ and $\boldsymbol{P}(y_i)^0 = 1 - \boldsymbol{P}(y_i)$ and $\boldsymbol{P}(y_i)^1 = \boldsymbol{P}(y_i)$. Once we know $\boldsymbol{P}(x)$, we also know $\boldsymbol{P}(\neg x) = 1 - \boldsymbol{P}(x)$.

Remark 3. *Note that certainly $\boldsymbol{P}(x) \geq 0$, but also that $\boldsymbol{P}(x) \leq \sum_{\varepsilon} \boldsymbol{P}(\overrightarrow{y}(\varepsilon)) = 1$, since all $\boldsymbol{P}(y_i)$ are probabilities.*

Remark 4. *Note that Definition 2 is restricted to Boolean values. Variables can be either in state $1 = true$ or in state $0 = false$. So any conditional probability for a variable x depending on a variable y needs to give real numbers for each state of y, see condition c. of Definition 2. This is similar to Pearl's definition in [19]. Note however that we propagate values in the direction of the arrows, i.e., towards descendant nodes. Pearl allows for updating of probabilities in both directions (ancestors and descendant nodes) at any point in the network (see Section 2.2.3 of [19]). As we are dealing with argumentation networks, the restriction to boolean variables is more natural and we need not be concerned about it. However, the propagation of updates in both directions is important and should be investigated not only in order to be more faithful in translating Bayesian networks into argumentation networks but even without any connection with Bayesian networks. Just by looking at traditional Dung networks we may wish to insist on a certain status for an argument (i.e., "in", "out" or "undecided") and propagate this result in both directions. Our work on Bayesian networks may give us ideas on how to do that, not only in the case of Bayesian Argumentation Networks (i.e., reflecting their behaviour) but perhaps by also taking advantage of the argumentation environment in developing an update theory applicable in argumentation in general, and not just restricted to the context of translated Bayesian networks. This requires extensive research and we leave it as future work.*

Example 2. *In order to illustrate Definition 2, we consider the network in Fig. 1 once more, with the probability values given in Section 2. We have $\boldsymbol{P}(X) = 0.01$, so $\boldsymbol{P}(\neg X) = 0.99$. $\boldsymbol{P}(Y) = 0.02$, so $\boldsymbol{P}(\neg Y) = 0.98$.*

$\Psi_A = \{X, Y\}$. We have that $\boldsymbol{P}(A, \Psi_A, X \wedge Y) = 0.95$; $\boldsymbol{P}(A, \Psi_A, X \wedge \neg Y) = 0.94$; $\boldsymbol{P}(A, \Psi_A, \neg X \wedge Y) = 0.29$; and $\boldsymbol{P}(A, \Psi_A, \neg X \wedge \neg Y) = 0.001$. According to

Definition 2:

$$\begin{aligned}
\boldsymbol{P}(A) &= \boldsymbol{P}(A, \Psi_A, X \wedge Y) \times \boldsymbol{P}(X) \times \boldsymbol{P}(Y) + \\
&\quad \boldsymbol{P}(A, \Psi_A, X \wedge \neg Y) \times \boldsymbol{P}(X) \times \boldsymbol{P}(\neg Y) + \\
&\quad \boldsymbol{P}(A, \Psi_A, \neg X \wedge Y) \times \boldsymbol{P}(\neg X) \times \boldsymbol{P}(Y) + \\
&\quad \boldsymbol{P}(A, \Psi_A, \neg X \wedge \neg Y) \times \boldsymbol{P}(\neg X) \times \boldsymbol{P}(\neg Y) \\
\boldsymbol{P}(A) &= 0.95 \times 0.01 \times 0.02 + \\
&\quad 0.94 \times 0.01 \times 0.98 + \\
&\quad 0.29 \times 0.99 \times 0.02 + \\
&\quad 0.001 \times 0.99 \times 0.98 \\
\boldsymbol{P}(A) &\approx 0.016
\end{aligned}$$

We then get $\boldsymbol{P}(\neg A) = 1 - \boldsymbol{P}(A) = 0.984$. The values of $\boldsymbol{P}(U)$ (resp., $\boldsymbol{P}(\neg U)$) and $\boldsymbol{P}(W)$ (resp., $\boldsymbol{P}(\neg W)$) are calculated in a similar way.

Definition 3. *We now translate any Bayesian network $\mathcal{N} = \langle S, \varrho, \boldsymbol{P}\rangle$ into a Bayesian Argumentation Network $\langle A, R, \boldsymbol{e}\rangle$.*

a. Assume the elements of S to be positive atoms of the form $\{q_i\}$. Let $\bar{S}$ be a new set of atoms of the form $\bar{S} = \{\bar{q} \mid q \in S\}$ and let $A_0 = S \cup \bar{S}$.

b. For any $x \in S$ such that Ψ_x has k elements, define two sets of new atoms

$$\begin{aligned}
C(x, \Psi_x) &= \{c(x, \Psi_x, \varepsilon) \mid \varepsilon \in 2^k\} \text{ and} \\
D(x, \Psi_x) &= \{d(x, \Psi_x, \varepsilon) \mid \varepsilon \in 2^k\}
\end{aligned}$$

Let $A = A_0 \cup \bigcup_{x \in S} (C(x, \Psi_x) \cup D(x, \Psi_x))$.

c. We now define R on A.

(a) Have $(\{\bar{q}\}, q)$ and $(\{q\}, \bar{q})$ be in R.

(b) For any $x \in S$ such that Ψ_x has k elements, let $(\{c(x, \Psi_x, \varepsilon)\}, d(x, \Psi_x, \varepsilon))$ be in R and let $(\{y_i^{1-\varepsilon(i)}\}, c(x, \Psi_x, \varepsilon))$ be in R.

(c) Let $(D(x, \Psi_x), x)$ be in R.

d. We now define $\boldsymbol{e}$. We let

$$\boldsymbol{e}(\{c(x, \Psi_x, \varepsilon)\}, d(x, \Psi_x, \varepsilon), c(x, \Psi_x, \varepsilon)) = \boldsymbol{P}(x, \Psi_x, \varepsilon)$$

Theorem 1. *Let $\mathcal{N} = \langle S, \varrho, \boldsymbol{P}\rangle$ be a Bayesian network and let $\langle A, R, \boldsymbol{e}\rangle$ be its (argumentation) translation. Let the source nodes of $\mathcal{N}$ be the set $\Omega \subseteq S$ and let f be a solution extension of $(A, R, \boldsymbol{e})$ such that $f(w) = \boldsymbol{P}(w)$, for $w \in \Omega$. Then for every $s \in S$, $f(s) = \boldsymbol{P}(s)$ (remember that $S \subseteq A$).*

Proof. The proof is done by induction on the distance (level) of nodes from Ω.

Level 0: nodes $w \in \Omega$.

Level $n + 1$: nodes s such that all nodes in Ψ_s are of level up to n with at least one of them being of level n.

Every node in S has a unique level because (S, ϱ) is acyclic. So we prove by induction on the level of a node s that $f(s) = \boldsymbol{P}(s)$.

Consider a node s of level $n + 1$, such as the one in Fig. 23. Its translation into $(A, R, \boldsymbol{e})$ is Fig. 24. The computation of $\boldsymbol{P}(s)$ in the Bayesian network is

$$\boldsymbol{P}(s) = \sum_{\varepsilon \in 2^k} \boldsymbol{P}(s, y_s, \varepsilon) \times \prod_i \boldsymbol{P}(y_i)^{\varepsilon(i)}$$

Our inductive assumption is that $\boldsymbol{P}(y_i) = f(y_i)$. We want to show that $\boldsymbol{P}(s) = f(s)$. Let us calculate $f(s)$ from Fig. 24. First we have

$$\begin{aligned} f(y_i) &= \boldsymbol{P}(y_i) \\ f(\neg y_i) &= f(y_i^0) = 1 - f(y_i) \end{aligned}$$

Thus,

$$\begin{aligned} f(c(s_, y_s, \varepsilon) &= \prod_i (1 - f(y_i)^{1-\varepsilon(i)}) \\ &= \prod_i f(y_i^{\varepsilon(i)}) = \prod_i \boldsymbol{P}(y_i)^{\varepsilon(i)} \end{aligned}$$

So

$$\begin{aligned} f(d(s_, y_s, \varepsilon) &= 1 - \boldsymbol{e}(s, y_s, c(s, y_s, \varepsilon)) \times f(c(s, y_s, \varepsilon)) \\ &= 1 - \boldsymbol{P}(s, y_s, \varepsilon) \times \prod_i \boldsymbol{P}(y_i)^{\varepsilon(i)} \end{aligned}$$

Therefore,

$$\begin{aligned} f(s) &= \min(1, \sum_\varepsilon (1 - f(d(s, y_s, \varepsilon)) \\ &= \min(1, \sum_\varepsilon \boldsymbol{P}(s, y_s, \varepsilon) \times \prod_i \boldsymbol{P}(y_i)^\varepsilon(i)) \\ &= \boldsymbol{P}(s) \end{aligned}$$

□

Remark 5. *Note that the translation of a Bayesian network uses only a restricted fragment of Definition 1, where each node s satisfies for all y_s^i and for all y_s^j:*

$$(y_s^i, s) \in R \textit{ and } (y_s^j, s) \in R \textit{ implies either } y_s^i \textit{ and } y_s^j \textit{ are singleton sets or } y_s^i = y_s^j$$

This means that translated Bayesian networks do not have the combination of joint attacks and single attacks such as the one in Fig. 22. Nodes either have a unique joint attack or zero or more attacks by individual nodes.

$$\begin{tikzcd}
y_s : & y_1 \arrow[dr] & \cdots & y_k \arrow[dl] \\
 & & s &
\end{tikzcd}$$

Figure 23: A node of level $n + 1$.

$$\begin{tikzcd}
\neg y_1 \arrow[r, leftrightarrow] \arrow[drrr] & y_1 & \cdots & y_i^{1-\varepsilon(i)} \arrow[d] & \cdots & \neg y_k \arrow[r, leftrightarrow] & y_k \arrow[dlll] \\
 & & & c(s, y_s, \varepsilon) \arrow[d, "\boldsymbol{P}(s, y_s, \varepsilon) = \boldsymbol{e}(s, y_s, c(s, y_s, \varepsilon))"] & & & \\
 & & \cdots \arrow[dr, dash] & d(s, y_s, \varepsilon) \arrow[d] & \cdots \arrow[dl, dash] & & \\
 & & & s & & &
\end{tikzcd}$$

Figure 24: A node of level $n + 1$.

Remark 6. *a. Note that the class of Bayesian Argumentation Networks contains more networks than just translated images of original Bayesian networks. These are indeed a type of argumentation networks inspired by looking at Bayesian networks. Note also that images of Bayesian networks are faithful and can translated back into Bayesian networks.*

b. Another important point to observe is that we are not imposing an argumentation structure on top of a Bayesian network, thus analysing the Bayesian network from an argumentation point of view. Such methods are common in the argumentation community. This will be discussed further in Section 6.

4 A Comprehensive Translation Example

In this section, we show in detail how a Bayesian network can be translated into a Bayesian Argumentation Network. For this, we will use the network of Fig. 1 as an example. The translation starts with the source nodes of the Bayesian network.

Consider a node α and its attackers $Att(\alpha) = \{\beta_1, \ldots, \beta_k\}$. Assume that the nodes in $Att(\alpha)$ are all source nodes, without any attackers. When β_i is a source node, it is given an initial probability $\boldsymbol{P}(\beta_i)$. We model this in a BAN by adding a new node $\neg\beta_i$ for each β_i such that β_i and $\neg\beta_i$ attack each other. Now, in the new network, β_i can assume any initial probability $\boldsymbol{P}(\beta_i)$ we want, with $\neg\beta_i$ obtaining $\boldsymbol{P}(\neg\beta_i) = 1 - \boldsymbol{P}(\beta_i)$. In order to represent all possible conjunctive expressions of the form

$$\boldsymbol{e}_{j=0}^{2^k-1} = \wedge_{i_j} \beta_{i_j}^{\pm}$$

where $\beta_{i_j}^{+} = \beta_{i_j}$ and $\beta_{i_j}^{-} = \neg\beta_{i_j}$, we need to create 2^k intermediate points c_j, whose attackers are all β_i, $\neg\beta_i$, such that $\boldsymbol{e}_j \models \beta_i$, and $\boldsymbol{e}_j \models \neg\beta_i$, respectively. Note that the value of each c_j will now correspond to a particular product

$$\boldsymbol{P}(c_j) = \prod_{i_j} p_{i_j}$$

where

$$p_{i_j} = \begin{cases} 1 - \boldsymbol{P}(\beta_i), \text{ if } \boldsymbol{e}_j \models \beta_i \\ \boldsymbol{P}(\beta_i), \text{ if } \boldsymbol{e}_j \models \neg\beta_j \end{cases}$$

Because we also want to represent transmission values, we now need to duplicate each intermediate point c_j with a corresponding x_j and have c_j attack x_j with transmission factor e_j.

In order to illustrate this, let us recall the Bayesian network of Fig. 1.

$$\begin{tikzcd}
X \arrow[dr] & & Y \arrow[dl] \\
& A \arrow[dl] \arrow[dr] & \\
U & & W
\end{tikzcd}$$

Consider the node A, with attackers X and Y. We first add points $\neg X$ and $\neg Y$ such that each of X and $\neg X$ and Y and $\neg Y$ attack each other. If we give value $\boldsymbol{P}(X)$ to X and value $\boldsymbol{P}(Y)$ to Y, then $\neg X$ will get value $1 - \boldsymbol{P}(X)$ and $\neg Y$ will get value $1 - \boldsymbol{P}(Y)$.

We now add points c_{00}, c_{01}, c_{10} and c_{11} corresponding to the conjunctive expressions

$$\begin{array}{rcl} e_0 : c_{00} & = & X \wedge Y \\ e_1 : c_{01} & = & X \wedge \neg Y \\ e_2 : c_{10} & = & \neg X \wedge Y \\ e_3 : c_{11} & = & \neg X \wedge \neg Y \end{array}$$

Note that $e_0 \models X$ and $e_0 \models Y$, so we add attacks from X and Y into c_{00} and do the same for e_1–e_3.

$$\begin{tikzcd}
X \arrow[r, leftrightarrow] \arrow[drr] \arrow[drrr] & \neg X \arrow[dl] \arrow[d] & Y \arrow[r, leftrightarrow] \arrow[dl] \arrow[dr] & \neg Y \arrow[dlll] \arrow[dl] \\
c_{11} & c_{10} & c_{01} & c_{00}
\end{tikzcd}$$

We can now see that the probabilities of c_{ij} are given as follows

$$\begin{array}{rcl} \boldsymbol{P}(c_{00}) & = & (1 - \boldsymbol{P}(X)) \cdot (1 - \boldsymbol{P}(Y)) \\ \boldsymbol{P}(c_{01}) & = & (1 - \boldsymbol{P}(X)) \cdot (1 - 1 + \boldsymbol{P}(Y)) = (1 - \boldsymbol{P}(X)) \cdot \boldsymbol{P}(Y) \\ \boldsymbol{P}(c_{10}) & = & (1 - 1 + \boldsymbol{P}(X)) \cdot (1 - \boldsymbol{P}(Y)) = \boldsymbol{P}(X) \cdot (1 - \boldsymbol{P}(Y)) \\ \boldsymbol{P}(c_{11}) & = & (1 - 1 + \boldsymbol{P}(X)) \cdot (1 - 1 + \boldsymbol{P}(Y)) = \boldsymbol{P}(X) \cdot \boldsymbol{P}(Y) \end{array}$$

Remember the initial parameters given for this network:

$$\begin{array}{rcl} \boldsymbol{P}(X) & = & 0.01 \\ \boldsymbol{P}(\neg X) & = & 0.99 \\ \boldsymbol{P}(Y) & = & 0.02 \\ \boldsymbol{P}(\neg Y) & = & 0.98 \end{array}$$

This gives us

$$\begin{array}{rcl} \boldsymbol{P}(c_{00}) & = & 0.99 \times 0.98 = 0.9702 \\ \boldsymbol{P}(c_{01}) & = & 0.99 \times 0.02 = 0.0198 \\ \boldsymbol{P}(c_{10}) & = & 0.01 \times 0.98 = 0.0098 \\ \boldsymbol{P}(c_{11}) & = & 0.01 \times 0.02 = 0.0002 \end{array}$$

Now for each node c_{ij}, we add an additional node x_{ij}, so that we can incorporate the transmission factors e_{ij}.

$$\begin{tikzcd}
X \arrow[r, leftrightarrow] \arrow[drr] \arrow[drrr] & \neg X \arrow[dl] \arrow[d] & Y \arrow[r, leftrightarrow] \arrow[dl] \arrow[dr] & \neg Y \arrow[dlll] \arrow[dl] \\
c_{11} \arrow[d, "e_{x_{11}|\varphi_{11}}"] & c_{10} \arrow[d, "e_{x_{10}|\varphi_{10}}"] & c_{01} \arrow[d, "e_{x_{01}|\varphi_{01}}"] & c_{00} \arrow[d, "e_{x_{00}|\varphi_{00}}"] \\
x_{11} & x_{10} & x_{01} & x_{00}
\end{tikzcd}$$

The resulting probabilities of the nodes x_{ij} are then given by the equations

$$\begin{aligned}
\boldsymbol{P}(x_{00}) &= 1 - e_{x_{00}|\varphi_{00}} \cdot c_{00} \\
\boldsymbol{P}(x_{01}) &= 1 - e_{x_{01}|\varphi_{01}} \cdot c_{01} \\
\boldsymbol{P}(x_{10}) &= 1 - e_{x_{10}|\varphi_{10}} \cdot c_{10} \\
\boldsymbol{P}(x_{11}) &= 1 - e_{x_{11}|\varphi_{11}} \cdot c_{11}
\end{aligned}$$

With our initial parameters, we have that

$$\begin{aligned}
e_{x_{00}|\varphi_{00}} &= 0.001 \\
e_{x_{01}|\varphi_{01}} &= 0.29 \\
e_{x_{10}|\varphi_{10}} &= 0.94 \\
e_{x_{11}|\varphi_{11}} &= 0.95
\end{aligned}$$

and hence

$$\begin{aligned}
\boldsymbol{P}(x_{00}) &= 1 - 0.001 \times 0.9702 = 0.9990 \\
\boldsymbol{P}(x_{01}) &= 1 - 0.29 \times 0.0198 = 0.9942 \\
\boldsymbol{P}(x_{10}) &= 1 - 0.94 \times 0.0098 = 0.9907 \\
\boldsymbol{P}(x_{11}) &= 1 - 0.95 \times 0.0002 = 0.9998
\end{aligned}$$

The x_{ij} jointly attack A:

$$\begin{tikzcd}
X \arrow[r, leftrightarrow] \arrow[drr] \arrow[drrr] & \neg X \arrow[dl] \arrow[d] & Y \arrow[r, leftrightarrow] \arrow[dl] \arrow[dr] & \neg Y \arrow[dlll] \arrow[dl] \\
c_{11} \arrow[d, "e_{x_{11}|\varphi_{11}}"] & c_{10} \arrow[d, "e_{x_{10}|\varphi_{10}}"] & c_{01} \arrow[d, "e_{x_{01}|\varphi_{01}}"] & c_{00} \arrow[d, "e_{x_{00}|\varphi_{00}}"] \\
x_{11} \arrow[dr] & x_{10} \arrow[d, "1"] & x_{01} \arrow[dl] & x_{00} \arrow[dll] \\
& A & &
\end{tikzcd}$$

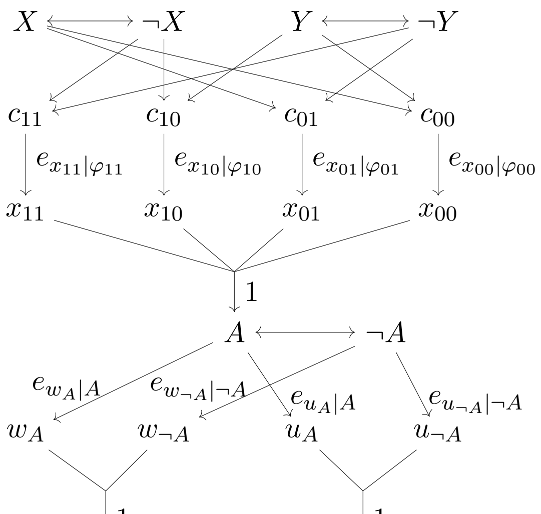
This finally gives us the value of the probability of node A as

$$\begin{aligned}
\boldsymbol{P}(A) &= \min\{1, \sum_{ij}(1 - \boldsymbol{P}(x_{ij}))\} \\
&= \min\{1, 0.001 + 0.0058 + 0.0093 + 0.0002\} \\
&\approx 0.016
\end{aligned}$$

as before. Therefore, $\boldsymbol{P}(\neg A) \approx 0.984$.

Now to proceed to the next level all we need to do is to add a complementary node to A, $\neg A$, such that A and $\neg A$ attack each other. This will give us $\boldsymbol{P}(\neg A) = 1 - \boldsymbol{P}(A)$. As before, we also need new intermediate nodes w_A, $w_{\neg A}$, u_A and $u_{\neg A}$ to incorporate transmission factors.

The original attack of A on W is then realised by the joint attack of the new intermediate nodes $w_{\neg A}$ and w_A and the attack of $\neg A$ on U is realised by the joint attack of the new intermediate nodes $u_{\neg A}$ and u_A.



The values of w_A, $w_{\neg A}$, u_A and $u_{\neg A}$ are calculated as follows.

$$\begin{aligned}
\boldsymbol{P}(w_A) &= 1 - e_{w_A|A} \cdot \boldsymbol{P}(A) \\
\boldsymbol{P}(w_{\neg A}) &= 1 - e_{w_{\neg A}|\neg A} \cdot \boldsymbol{P}(\neg A) \\
\boldsymbol{P}(u_A) &= 1 - e_{u_A|A} \cdot \boldsymbol{P}(A) \\
\boldsymbol{P}(u_{\neg A}) &= 1 - e_{u_{\neg A}|\neg A} \cdot \boldsymbol{P}(\neg A)
\end{aligned}$$

Since

$$\begin{aligned} e_{w_A|A} &= 0.7 \\ e_{w_{\neg A}|\neg A} &= 0.01 \end{aligned}$$

$$\begin{aligned} e_{u_A|A} &= 0.9 \\ e_{u_{\neg A}|\neg A} &= 0.05 \end{aligned}$$

We get

$$\begin{aligned} \boldsymbol{P}(w_A) &= 1 - 0.7 \times 0.016 \approx 0.988 \\ \boldsymbol{P}(w_{\neg A}) &= 1 - 0.01 \times 0.984 \approx 0.990 \\ \boldsymbol{P}(u_A) &= 1 - 0.9 \times 0.016 \approx 0.985 \\ \boldsymbol{P}(u_{\neg A}) &= 1 - 0.05 \times 0.984 \approx 0.950 \end{aligned}$$

We can finally calculate the values of W and U.

$$\begin{aligned} \boldsymbol{P}(W) &= \min\{1, 1 - \boldsymbol{P}(w_{\neg A}) + 1 - \boldsymbol{P}(w_A)\} = \min\{1, 0.009 + 0.0112\} \approx 0.021 \\ \boldsymbol{P}(U) &= \min\{1, 1 - \boldsymbol{P}(u_{\neg A}) + 1 - \boldsymbol{P}(u_A)\} = \min\{1, 0.049 + 0.0144\} \approx 0.063 \end{aligned}$$

These are exactly the same values we had before.

5 Complexity Discussion

We need to consider two aspects involved in the complexity arising from the translation of a Bayesian network into a Bayesian Argumentation Network. The first one is related to the translation itself whereas the second is related to the actual computation of the node values.

It is easy to see that the translation of a Bayesian network into a Bayesian Argumentation Network according to Definition 3 results in the creation of many new nodes. This addition of nodes is linear on the number of nodes of the original Bayesian network (item a. of Definition 3) and exponential on the number of ancestors of each node of the original Bayesian network (item b. of Definition 3).

The effect of the increase in the number of nodes in the actual computation of node values is deceptive.

Although the translation does incur in the addition of many new nodes, the complexity of the actual calculation of the node values remains basically the same.

The nodes added in a Bayesian Argumentation Network simply encode explicitly the intermediate calculations that would otherwise be implicitly done in the original Bayesian network.

In conclusion, a Bayesian Argumentation Network simply explicitly encodes as nodes the calculations that would have to be done anyway in the original Bayesian network. The new nodes name key values when using the conditional probability tables in the original calculations of the Bayesian network. Therefore there is no significant additional cost.

6 Related Work

Let us compare with several related papers. We start with Vreeswijk's "Argumentation in bayesian belief networks" [27]. An important point to observe is that we are not using a meta-level device of extracting/identifying arguments and a notion of attack on the arguments from the Bayesian network and thus obtaining an argumentation network as is done in [27].

Vreeswijk's approach does not give us a direct connection/translation between Bayesian networks and argumentation networks. It is more akin to imposing an argumentation structure on top of a Bayesian network, thus analysing the Bayesian network from an argumentation point of view. Indeed Vreeswijk sees his approach [27] as

> "a proposal to look at Bayesian belief networks from the perspective of argumentation. More specifically, I propose an algorithm that enables users to start an argumentation process within the context of an existing Bayesian belief network. ... with some imagination, the CPTs[9] of the above Bayesian network can be translated into the rule-base and evidence ... A next step towards argumentation is to chain rules into arguments. ... What remains to be done to obtain a full-fledged argument system, is to define an attack relation between pairs of arguments. To this end, I choose to define the notion of attack on the basis of two notions that are more elementary and (therefore) fall beyond the scope of a Dung-type argument system, viz. the notion of counterargument and the notion of strength of an argument. First I will discuss counter-arguments, and then I will discuss argument strength. ...
>
> **Definition 4 (Attack).** We say that argument a is attacked by argument b, written $a \leftarrow b$, if it satisfies the following two conditions:

[9]Conditional Probability Tables

1. Argument b is a counterargument of a sub-argument a' of a.
2. Argument b is stronger than argument a'."

Such methods are common in the argumentation community. For example, taking a logic program, forming arguments using this logic program and thus generating an argumentation network, and then proving equivalence of the logic program with the argumentation network (see for example [7, 28]).

The next related work we consider is the translation of various kinds of argumentation networks into Bayesian networks, as is done for example in [16]. The idea is beautifully described by the authors:

> "This paper presents a technique with which instances of argument structures in the Carneades model can be given a probabilistic semantics by translating them into Bayesian networks. The propagation of argument applicability and statement acceptability can be expressed through conditional probability tables. This translation suggests a way to extend Carneades to improve its utility for decision support in the presence of uncertainty."

Note that [27] identifies some argumentation structure in Bayesian networks whilst [16] uses Bayesian networks to implement certain kinds of argumentation networks. There is similarity but the approaches are different. On the other hand, our approach is to faithfully represent Bayesian networks inside a newly motivated numerical argumentation network.

Finally, the approach used in [25, 24] is similar in spirit to the one used in [27]. The idea is to provide explanations of Bayesian networks in legal or medical domains using support graphs. The nodes of the support graph can be labelled to provide an argumentative interpretation of the Bayesian network.

7 Conclusions and Future Work

The perceptive reader from the Bayesian community may take no interest in the translation of Bayesian networks into argumentation. They will point out justifiably, that we are just translating the Bayesian algorithms into argumentation and then using these same algorithms under the guise of argumentation. This may be of interest from the abstract mathematical expressive power point of view, but that is all.

We would like to show that there are indeed other benefits to this translation both to the argumentation community and to the Bayesian community.

a. By interpreting Bayesian networks in some extension of Dung's networks and vice-versa, the argumentation community, who also deals with updates and change, stands to benefit from the updating algorithms in Bayesian networks. We can add nodes and change values of nodes and use Bayesian propagation to propagate the changes.

   This needs to be studied further.

b. Bayesian networks are acyclic, they have difficulties with loops (see [26]). The equational approach in argumentation can deal with loops without any problems (see for instance the forthcoming special issue in the Journal of Logic and Computation on the Handling of Loops in Argumentation Networks and see [11]). We notice that Bayesian networks can adopt the equational approach [15] in case of loops and simply solve the equations which arise on the probabilities. A solution always exists by Brouwer's fixed-point theorem. We are in fact surprised that this approach has never been taken (to the best of our knowledge) by the Bayesian community. One can then use loop handling techniques from argumentation to propagate updates and changes in the (cyclic) Bayesian network by simply solving equations. By implementing Bayesian networks in argumentation under the equational approach we can allow for loops in Bayesian networks and still hopefully see a way to obtain results, again, this requires further research.

c. A third benefit to Bayesian networks is the possibility to develop proof theory for Bayesian networks. The extension of argumentation networks which hosts the translation of Bayesian networks is implemented in Łukasiewicz infinite-valued logic with product.[10] This Łukasiewicz logic has a proof theory (see [10]). We can therefore hope for the same for Bayesian networks. Again this needs to be studied.

In addition, the following needs to be investigated in detail:

a. Take an example of a cyclic Bayesian network, recognised as interesting in the Bayesian community, and translate it into argumentation. See how argumentation handles the loops and try to find suitable algorithms for handling cycles in Bayesian networks. These algorithms should now be independent of argumentation.

b. Develop algorithms of updating argumentation networks by looking at the examples of updating used in the Bayesian domain and the way they implement

[10]This is actually shown in the current paper (see Remark 2).

the updates. There are important papers investigating updates and revision of argumentation networks by central figures in the argumentation community, including [4, 22, 21, 5, 8, 3, 9]. Addressing these papers will be done in future work.

c. Identify proof theoretic queries in Bayesian networks and import proof theory from Łukasiewicz logic to model them.

References

[1] H. Barringer, D. M. Gabbay, and J. Woods. Temporal dynamics of support and attack networks. In D. Hutter and W. Stephan, editors, *Mechanizing Mathematical Reasoning*, 2005. LNCS, vol. 2605.

[2] Howard Barringer, Dov M. Gabbay, and John Woods. Temporal, numerical and meta-level dynamics in argumentation networks. *Argument & Computation*, 3(2-3):143–202, 2012.

[3] R. Baumann and G. Brewka. AGM meets abstract argumentation: Expansion and revision for dung frameworks. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2734–2740, 2015.

[4] G. Brewka, S. Ellmauthaler, H. Strass, J. P. Wallner, and S. Woltran. Abstract dialectical frameworks revisited. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 803–809. AAAI Press, 2013.

[5] G. Brewka and S. Woltran. GRAPPA: A semantical framework for graph-based argument processing. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, pages 153–158, 2014.

[6] Gerhard Brewka and Stefan Woltran. Abstract dialectical frameworks. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010*. AAAI Press, 2010.

[7] Martin Caminada, Samy Sá, Jo ao Alcântara, and Wolfgang DvoÅŹák. On the equivalence between logic programming semantics and argumentation semantics. *International Journal of Approximate Reasoning*, 58:87 – 111, 2015. Special Issue of the Twelfth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2013).

[8] S. Coste-Marquis, S. Konieczny, J-G. Mailly, and P. Marquis. On the revision of argumentation systems: Minimal change of arguments statuses. In *14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14)*, Vienna, July 2014.

[9] M. Diller, A. Haret, T. Linsbichler, S. Rümmele, and S. Woltran. An extension-based approach to belief revision in abstract argumentation. In *Proceedings of the 24th Inter-*

national Conference on Artificial Intelligence, IJCAI'15, pages 2926–2932. AAAI Press, 2015.

[10] Francesc Esteva, Lluís Godo, and Enrico Marchioni. *Fuzzy Logics with Enriched Language*, chapter VIII, pages 627 – 711. Number 38 in Vol. 2. College Publications, London, 2011.

[11] D. M. Gabbay. The handling of loops in argumentation networks. *Journal of Logic and Computation*, 2014.

[12] D. M. Gabbay. Theory of semi-instantiation in abstract argumentation. *Logica Universalis*, To appear.

[13] D. M. Gabbay and O. Rodrigues. Probabilistic argumentation: An equational approach. *Logica Universalis*, pages 1–38, 2015.

[14] Dov M. Gabbay. Fibring argumentation frames. *Studia Logica*, 93(2-3):231–295, 2009.

[15] Dov M. Gabbay. Equational approach to argumentation networks. *Argument & Computation*, 3(2-3):87–142, 2012.

[16] M. Grabmair, T. F. Gordon, and D. Walton. Probabilistic semantics for the carneades argument model using bayesian networks. In *Proceedings of the 2010 Conference on Computational Models of Argument: Proceedings of COMMA 2010*, pages 255–266, Amsterdam, The Netherlands, The Netherlands, 2010. IOS Press.

[17] Kevin B. Korb and Ann E. Nicholson. *Bayesian Artificial Intelligence, Second Edition.* CRC Press, Inc., Boca Raton, FL, USA, 2nd edition, 2010.

[18] Søren Holbech Nielsen and Simon Parsons. A generalization of dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In Nicolas Maudet, Simon Parsons, and Iyad Rahwan, editors, *ArgMAS*, volume 4766 of *Lecture Notes in Computer Science*, pages 54–73. Springer, 2006.

[19] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29(3):241–288, September 1986.

[20] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[21] S. Polberg and D. Doder. Probabilistic abstract dialectical frameworks. In Eduardo Fermé and João Leite, editors, *Logics in Artificial Intelligence: 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings*, pages 591–599, Cham, 2014. Springer International Publishing.

[22] J. Pührer. Realizability of three-valued semantics for abstract dialectical frameworks. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 3171–3177. AAAI Press, 2015.

[23] A. Rose. Formalisations of further $\aleph_0$-valued Łukasiewicz propositional calculi. *Journal of Symbolic Logic*, 43:207–210, 6 1978.

[24] S. T. Timmer, J-J Ch. Meyer, H. Prakken, S. Renooij, and B. Verheij. Explaining bayesian networks using argumentation. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 13th European Conference, ECSQARU 2015, Compiègne, France, July 15-17, 2015. Proceedings*, pages 83–92, 2015.

[25] S. T. Timmer, J-J Ch. Meyer, H. Prakken, S. Renooij, and B. Verheij. A structure-guided approach to capturing bayesian reasoning about legal evidence in argumentation. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, ICAIL '15, pages 109–118, New York, NY, USA, 2015. ACM.

[26] A. L. Tulupyev and S.I. Nikolenko. Directed cycles in bayesian belief networks: Probabilistic semantics and consistency checking complexity. In A. Gelbukh, A. de Albornoz, and H. Terashima, editors, *MICAI 2005, Lectune notes in Artificial Intelligence 3789*, pages 214–223. Springer-Verlag Berlin Heidelberg, 2005.

[27] G. A. W. Vreeswijk. Argumentation in bayesian belief networks. In *Proceedings of the First International Conference on Argumentation in Multi-Agent Systems*, ArgMAS'04, pages 111–129, Berlin, Heidelberg, 2005. Springer-Verlag.

[28] Yining Wu, Martin Caminada, and Dov M. Gabbay. Complete extensions in argumentation coincide with 3-valued stable models in logic programming. *Studia Logica*, 93(2):383–403, 2009.

Received September 2015