An Argumentative Knowledge-Based Model Construction Approach for Bayesian Networks

Michelle Blom

NICTA Victoria Research Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010, Australia

Abstract. In this paper, an argumentative knowledge-based model construction (KBMC) technique for Bayesian networks is presented. This approach allows an agent to collect and instantiate the most accepted subset of an imperfect knowledge base to dynamically construct a Bayesian network. Arguments are constructed to represent active paths through an agent's knowledge base – paths consisting of information that is computationally relevant in the evaluation of a query $Pr(Q|E)$. Argumentation over paths is used to select the valid or most accepted information according to the preferences of the agent. This information is consequently formed into candidate network structures by accrual. This work is presented as an extension of the KBMC approach of Haddawy [5]. The potential of the approach to be used in multi-agent network construction is discussed.

1 Introduction

This paper presents an argumentation-based approach for allowing agents to reason about the probability of claims given a large body of imperfect knowledge. Imperfect knowledge may be uncertain, of varying validity or reliability, and potentially cyclic. The presented technique is an argumentative knowledge-based model construction (KBMC) approach for Bayesian networks [10, 2, 5]. In this approach, argumentation is used to construct a Bayesian network from the most *accepted* subset of an imperfect knowledge base. In doing so, limitations of existing Bayesian network construction techniques when faced with knowledge bases of this kind are addressed. Based on an instantiation of the abstract argumentation framework of Dung [3] and the accrual operator of Prakken [8], this paper provides a mechanism for collecting the relationships that are relevant in inferring the probability of a claim, instantiating them in a given context (a set of observations of the environment state), and combining the most accepted subset to come to a conclusion.

Bayesian networks [7] provide a powerful and well-established technique for reasoning about the probability of a claim or state assignment given relevant evidence concerning an environment state. Using a static Bayesian network to model a domain is sometimes inadequate: reasoning across different contexts

or situations with a network expressing how propositional statements influence each other is not possible; it is a cumbersome task to express *everything* that may influence a statement in a single structure; and we do not need information about an entire domain to make inferences about a small subset of it.

In Bayesian network theory, these problems have been addressed with the introduction of knowledge-based model construction (KBMC) techniques [10, 2]. The principle behind KBMC is to express information about a domain in a general first-order knowledge representation language. A query $Pr(Q|E)$ may be answered by collecting and instantiating relevant pieces of knowledge to dynamically construct and evaluate a Bayesian network. KBMC techniques allow networks to be constructed from relationships over *classes* of objects, rather than object instances, and prevent the inclusion of information that is not relevant in the evaluation of a query. To operate successfully, existing KBMC approaches place several constraints on the knowledge bases used. These knowledge bases must be acyclic, for example, and are assumed to consist of information that is valid across all contexts.

Where the knowledge base of an agent, or the collective knowledge of a set of agents, is perfect – acyclic, reliable, and true independent of context – existing KBMC approaches may be used without modification. Where this is not the case, agents need more control over what knowledge they include in the network construction process. A motivating example is the construction of patient-specific networks. Consider a knowledge base outlining the factors involved in classifying a *Disease*. We may wish to construct a patient-specific network by including only relationships that hold given information about the patient. If the patient is old, we may wish to exclude certain factors which sometimes influence *Disease*.

The aim of this approach is to allow an agent to determine the probabilities of claims given an *imperfect* knowledge base of probabilistic relationships. We allow an agent to argumentatively select a subset of this knowledge that (i) is relevant in evaluating a query, $Pr(Q|E)$, and (ii) consists of the most reliable or preferred information from the perspective of the agent.

Given a query, $Pr(Q|E)$, a process of argumentation followed by voting is used to construct a network for its evaluation. Arguments, AR, are constructed to represent *active* paths (chains of influence) through the knowledge base of an agent. Each active path expresses information that is computationally relevant in the evaluation of $Pr(Q|E)$. Comparison of these arguments is performed to select the subset that is accepted in the context E. This subset represents the grounded extension of a Dung argumentation framework $\langle AR, \mathcal{D} \rangle$ [3] where $\mathcal{D}(A, B, KB, E)$ encodes the ways in which the presence of argument A invalidates argument B in the context E given the knowledge base KB. The concept of accrual, as defined in [8], is used to combine acceptable arguments to form candidate network structures. A voting process is used to select the set of accruals that are as good as, or better than, any other network the agent may construct.

Before delving into the details of the network construction process, Section 2 presents some logical preliminaries – aspects of the abstract argumentation

framework of Dung [3] and formulation of accrual by Prakken [8] that are used in this work. The aspects of Bayesian network theory that are relevant in the presentation of this work are described in Section 3. In Section 4, the knowledge representation used by Haddawy [5], and in this approach, is defined. The construction of arguments and their comparison is outlined in Sections 5 and 6. The formation and comparison of candidate network structures, and the extension of this approach to a multi-agent setting, are described in Section 7. Finally, the soundness and completeness of the approach, and future work planned on the topic, is discussed in Section 8.

2 Logical Preliminaries

In this section, concepts from the formalisations in [3, 8] required in the presentation of this work are described. According to [3], an argumentation framework is defined as follows.

Definition 1 *An argumentation framework is a pair $AF = \langle AR, \mathcal{D} \rangle$ where AR is a set of arguments, and $\mathcal{D} \subseteq AR \times AR$ is a binary relation on AR. When $\mathcal{D}(A,B)$ holds, we say that argument A attacks argument B, or that argument B is attacked by argument A. Given an argumentation framework $AF = \langle AR, \mathcal{D} \rangle$:*

(i) A set of arguments S is conflict free if there are no two arguments A and B in S such that $\mathcal{D}(A,B)$.
(ii) An argument $A \in AR$ is acceptable w.r.t a set of arguments S iff for every $B \in AR$ where $\mathcal{D}(B, A)$, there is an argument $C \in S$ such that $\mathcal{D}(C, B)$.
(iii) A set of arguments S is admissible iff it is conflict free and each argument in S is acceptable w.r.t S.
(iv) The characteristic function, F_C, of the argumentation framework finds the set of arguments in AR that are acceptable w.r.t the set of arguments S: $F_C(S) = \{A \mid A \in AR$ and A is acceptable w.r.t $S\}$.
(v) The grounded extension of an argumentation framework is the least fixed point of F_C and represents the set of acceptable arguments in AR.

In [8], Prakken formalises the accrual of arguments as an inference rule. Given a collection of arguments, all supporting the same conclusion, this inference rule combines these arguments into a new argument called an accrual. Accrual embodies the idea that arguments have a *strength*, and that combining arguments may provide a stronger case for the truth of their common conclusion. Moreover, it allows a reasoner to model the situation where certain collections of arguments may not be believed in conjunction with others, and where certain collections of arguments are preferred over others.

Prakken has outlined three principles that must be satisfied in a treatment of accrual [8]: (i) flawed arguments cannot accrue, (ii) accruals can be weaker than their constituent arguments, and (iii) accruals make their constituent arguments inapplicable. The third principle is included to manage the impact of principle (ii). If an accrual A is attacked by an accrual B, it is undesirable that a proper subset $A' \subset A$ be able to attack B. According to principle (ii), A' may be stronger than both A and B, yet it takes into account *less* information.

3 Bayesian Networks – A Primer

A Bayesian network is a directed acyclic graph (DAG). Each node in the graph is a propositional variable, which may take on one of a finite set of values. Arcs in the graph – edges connecting one variable to another – are a qualitative representation of which variables have an influence on the values of others.

Each node N in a Bayesian network has a possibly empty set of children *Children(N)*, and a possibly empty set of parents *Parents(N)*. A child of node N is any node *child* such that there is a directed arc from N to *child*. A parent of a node N is any node *parent* such that there is a directed arc from *parent* to N. Each node N in a Bayesian network is associated with a conditional probability table (CPT) outlining, for each value the variable represented by N may take on, the probability of that value assignment given the different states of its parents.

A Bayesian network additionally represents conditional *independencies*. Given a Bayesian network, BN, we may determine if a set of variables $X \in BN$ are independent of a set of variables $Y \in BN$ given the states of variables in set $Z \in BN$ are known. X are conditionally independent of Y given Z if Z *d-separates* X and Y. Z d-separates X and Y if every undirected path between a node in X and a node in Y is *blocked* by Z due to one of the following conditions being present: (i) there is a node $N \in Z$ on the undirected path with an arrow entering and an arrow leaving or two arrows leaving (diverging arrows), or (ii) there is a node $N \notin Z$ on the undirected path that has two arrows entering it (converging arrows) and whose every descendant D satisfies $D \notin Z$.

Finally, an approach for managing uncertain observations is described, as it shall be used in the work presented in this paper. In this approach, uncertain observations are viewed as *virtual evidence* [7]. In doing so, a dummy node is added to the network for each observation. A directed arc from each node to each dummy node that represents an observation of its state is also added. Each observation is characterised by a set of probabilities – the probabilities of that observation being made given each possible value assignment to the observed variable. The observation that a is true, for example, is associated with the probabilities $Pr(O(a)|a)$ and $Pr(O(a)|\neg a)$, which represent the probability that a will be observed true if it is actually true, and the probability that a will be observed true if it is actually false, respectively. These probabilities form the CPT for the dummy observation nodes in the network. For each observation node, $O(\cdot)$, $Pr(O(\cdot))$ is set to 1 and the network is evaluated using standard techniques.

4 Knowledge Representation and Constraints

The approach presented in this paper is based on the network construction approach of Haddawy [5]. This section outlines the aspects of [5] that are used in this work – a language for representing knowledge, a set of constraints placed on knowledge to support the network construction process, and a definition of d-separation for knowledge bases.

An appropriate knowledge representation language must be selected to encode the knowledge of an agent. This language must be expressive enough to

describe relationships amongst classes of objects, and provide the information necessary to construct a Bayesian network. We would like to express knowledge in such a way so that we may find a ground subset of the knowledge base that represents a Bayesian network.

The knowledge base needs to represent parent/child relationships (encoding direct influence) amongst variables in the domain, potentially uncertain observations, and unconditional probabilities for variables with no parents. As in [5], the knowledge base of an agent is represented by a collection of rules of the form:

$$\begin{array}{ll} \textbf{Ante:} & f_1(\boldsymbol{x_1})\text{: } \{v_1^1, \ldots, v_k^1\}, \ldots, f_n(\boldsymbol{x_n})\text{: } \{v_1^n, \ldots, v_l^n\} \\ \textbf{Conse:} & g(\boldsymbol{x_0})\text{: } \{v_1^0, \ldots, v_m^0\} \\ \textbf{Matrix:} & |g| \cdot |f_1| \cdot \ldots \cdot |f_n| \end{array}$$

which expresses a set of probabilistic sentences of the form:

$$\forall X P(g(\boldsymbol{x_0}) = v_i^0 | f_1(\boldsymbol{x_1}) = v_j^1 \wedge \ldots \wedge f_n(\boldsymbol{x_n}) = v_k^n) = \alpha$$

where $f_i(\boldsymbol{x_i})$ and $g(\boldsymbol{x_k})$ represent random variables with discrete values v_h^l, $\boldsymbol{x_j}$ is a subset of the variables in X, **Ante** denotes the antecedents of a rule ($f_1(\boldsymbol{x_1})$, $\ldots$, $f_n(\boldsymbol{x_n})$), **Conse** denotes the consequent of a rule ($g(\boldsymbol{x_0})$), and **Matrix** denotes the CPT of a rule.

In [5], each universally quantified probability sentence $Pr(A|B) = \alpha$ is true iff for all ways in which we can substitute domain elements for the quantified variables in A and B, $Pr(A \wedge B) = \alpha \cdot Pr(B)$. To reflect the fact that the knowledge bases considered in this work are imperfect, and the truth of a sentence is dependent on context, each sentence is interpreted to mean that *there exists* a substitution for the quantified variables in A and B such that $Pr(A \wedge B) = \alpha \cdot Pr(B)$.

Using this knowledge representation language, Haddawy [5] defines the following constraints which must be satisfied by a knowledge base so that a ground subset of it is "structurally isomorphic" to a Bayesian network: (**C1**) each rule antecedent must appear as the consequent of a rule; (**C2**) all variables in the antecedents of a rule must appear in the consequent of the rule; (**C3**) the knowledge base does not contain two distinct rules that have ground instances with identical consequents; and (**C4**) there does not exist a set of ground rule instances $R_1, R_2, \ldots, R_n$ in the knowledge base such that $\textbf{Conse}(R_1) \in \textbf{Ante}(R_2)$, $\textbf{Conse}(R_2) \in \textbf{Ante}(R_3)$, $\ldots$, $\textbf{Conse}(R_n) \in \textbf{Ante}(R_1)$.

In argumentative KBMC, these constraints are relaxed, allowing for more expressiveness in the knowledge base. Argumentation is used to select a ground subset of the knowledge base that (i) does satisfy these constraints, (ii) is valid in the given context, and (iii) is computationally relevant to the evaluation of a query $Pr(Q|E)$. It is this constraint relaxation, and the relaxation of the assumption that the knowledge is valid across all contexts, that makes the knowledge bases considered here different to those defined in most existing work [10, 2, 5]. In [4], Glesner and Koller develop a KBMC technique in which constraint C3 is relaxed. In their approach, multiple causes (rules with the same consequent)

can be incorporated in network construction, and combination rules (such as noisy-OR [7]) are used to determine new CPTs for network nodes. Their technique does not consider cyclic knowledge bases, or knowledge bases containing contextually-invalid knowledge.

To determine which knowledge is computationally relevant to a query requires a definition of d-separation for a knowledge base, expressing which variables are conditionally independent of others given a set of evidence variables. We use the definition of d-separation for knowledge bases found in [5], and summarised in Definition 2.

Definition 2 *Given a knowledge base KB of probabalistic rules:*

(i) *There is an edge between two terms F and G if there is a rule $R \in KB$ where $F \in \boldsymbol{Ante}(R)$ and $G = \boldsymbol{Conse}(R)$, or $G \in \boldsymbol{Ante}(R)$ and $F = \boldsymbol{Conse}(R)$.*

(ii) *There is a directed edge between two terms F and G if there is a rule $R \in KB$ where $F \in \boldsymbol{Ante}(R)$ and $G = \boldsymbol{Conse}(R)$.*

(iii) *There is a path between two ground terms F and G if there is a set of ground instances of rules $\boldsymbol{R} \subseteq KB$ and a set of ground terms $\boldsymbol{F} = \{F_1, \ldots, F_n\}$ appearing in $\boldsymbol{R}$ such that there is an edge between F and F_1, F_1 and F_2, F_2 and F_3, $\ldots$, and F_n and G. $\boldsymbol{F}$ are called the ground terms on the path.*

(iv) *A ground term F_i on a path has converging arrows if there are two directed edges on the path pointing to F_i.*

(v) *F is a predecessor of G if there is a path in the knowledge base between F and G along which all edges point toward G.*

(vi) *If $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ are three disjoint sets of ground instances of terms in a knowledge base KB then $\boldsymbol{Z}$ is said to d-separate $\boldsymbol{X}$ from $\boldsymbol{Y}$ in KB, denoted $\langle \boldsymbol{X} | \boldsymbol{Z} | \boldsymbol{Y} \rangle_D^{KB}$, if along every path between a ground term in $\boldsymbol{X}$ and a ground term in $\boldsymbol{Y}$ there is a ground term W satisfying one of the following two conditions: (1) W has converging arrows and none of W or its descendants are in $\boldsymbol{Z}$, or (2) W does not have converging arrows and $W \in \boldsymbol{Z}$.*

(vii) *Given a query, $Pr(Q|E)$, a path is active if each ground term on the path is not d-separated from Q by E. Otherwise, it is blocked.*

Given a query, $Pr(Q|E)$, every piece of knowledge in a knowledge base that can be instantiated with others to form an active path between an evidence node in E and the query node Q is computationally relevant in the evaluation of $Pr(Q|E)$. Every piece of knowledge that can be instantiated to form an active path between a node with an unconditional probability and the node Q, where all ground terms along the path are predecessors of Q, is also computationally relevant in the evaluation of $Pr(Q|E)$.

5 An Algorithm for Constructing Arguments

There are potentially many pieces of knowledge that are computationally relevant in the evaluation of a query, but cannot all together represent a Bayesian network. Relevant ground instances of rules in the knowledge base may form a

cycle, for example, violating the constraint C4 (outlined in Section 4). We assume in this paper that $CPTs$ can be combined, as in [4], using a rule such as noisy-OR [7], and constraint C3 is allowed to be violated.

An agent constructs arguments to represent the active paths found within its knowledge base. Each argument is defined to be a subset of ground rule instances from an agent's knowledge base, and to satisfy the structural properties of a Bayesian network. An argument is constructed such that it satisfies the conditions outlined in Definition 3.

Definition 3 *Given a knowledge base KB of probabalistic rules and a query, $Pr(Q|E)$, an argument:*

(i) is a collection of ground instances of rules in KB,
(ii) contains no nodes that are d-separated from Q by E w.r.t to the knowledge contained within the argument,
(iii) satisfies the constraints C1, C2, and C4 (outlined in Section 4),
(iv) contains all observations in E for each antecedent and consequent appearing in a rule within the argument,
(v) is a minimal subset of KB such that conditions (i) – (iv) are satisfied (minimal in the sense that the removal of a rule results in the violation of one of the conditions (i) – (iv)).

Arguments are defined in this way as: any path that cannot be combined with other information to form a Bayesian network is not worth considering; any path that cannot be combined with other information to form part of an acceptable subset of KB is not worth considering; and all information that is relevant only when a particular path is present in an acceptable argument is not worth considering if no such argument exists. Part (v) of Definition 3 is required to ensure that unnecessary information is not added to an argument to cause a path being discounted for reasons other than those described above. Part (ii) of Definition 3 ensures that if an argument is selected to form the final network on its own, it will still represent an active path w.r.t the knowledge contained in the argument.

A basic algorithm for the construction of arguments satisfying Definition 3 is shown in Figure 1. ConstructArguments represents the main body of the algorithm. First, paths of predecessors that link the query node Q to nodes with prior probabilities are determined by backward chaining from Q through rules in the knowledge base KB. This step is achieved by FindPredecessors. Second, undirected paths linking Q to each evidence node in the query are determined in FindPaths, using a combination of backward and forward chaining through rules in KB. As each of the determined paths does not (necessarily) contain enough information to structurally define a Bayesian network, the paths are expanded using FindExpansions (shown in Figure 2). The expansion of a path finds additional rules to transform it into an argument. In each of the presented algorithms, : denotes set concatenation.

To build an expansion of a path, P, we must consider each of the antecedents A that occur in a rule within it and check whether there is a rule with A as its

Algorithm 1 ConstructArguments

Input: KB, Q, E
Output: *Arguments*
$Arguments \leftarrow [\,]$
$Paths \leftarrow$ FindPredecessors(Q, KB)
for each $e \in E$ **do**
  $Paths^* \leftarrow$ FindPaths(Q, e, KB)
  $Paths \leftarrow Paths : Paths^*$
end for
for each $P \in Paths$ **do**
  $Expansions \leftarrow$ FindExpansions(P, KB, $[\,]$)
  $Arguments \leftarrow Expansions : Arguments$
end for

Fig. 1: The ConstructArguments algorithm for a query $Pr(Q|E)$.

consequent in the path. If there is not, we must find all rules R in the knowledge base KB with $A = \mathbf{Conse}(R)$. For each of these rules, we construct a new rule collection by adding the rule to P. For each of these new collections, we find their possible expansions and add them to our result, ensuring that each antecedent is only visited once in the construction an expansion. The *Visited* set – empty on the first call to FindExpansions – achieves this task by keeping a record of the antecedents considered in each thread of the algorithm. FilterInactive ensures that parts (iii) and (v) of Definition 3 are satisfied for each expansion returned. This occurs as the final step as often we cannot determine if a path is active or not (according to part (ii) of Definition 3) without knowing what information it can be combined with.

To demonstrate the construction of arguments, let's consider a simple example based on the construction of patient specific networks and the Wumpus World [9]. In the Wumpus World game, a player has the task of navigating across a grid where each square may contain gold, a pit, a Wumpus, or nothing. While there are many dangers facing players in the game, often overlooked is the plight of the Wumpii. A Wumpus may break a bone by falling in a pit, get shot by an arrow, or contract a disease from handling germ-ridden gold. In this example, a doctor (D) attempts to diagnose the injury that has befallen a Wumpus (W).

The knowledge base of D contains the following rules, where $+$, $-$, t, s, o, y, h, and l denote *true*, *false*, *tall*, *short*, *old*, *young*, *high*, and *low*, respectively.

Stealing gold leads to gold lurgy
$R1$ **Ante:** $steal(X) : \{+, -\}$
Conse: $lurgy(X) : \{+, -\}$

Gold lurgy typically causes rashes
$R2$ **Ante:** $lurgy(X) : \{+, -\}$
Conse: $rash(X) : \{+, -\}$

A Wumpus with a rash tends to steal gold to pay for treatment
$R3$ **Ante:** $rash(X) : \{+, -\}$
Conse: $steal(X) : \{+, -\}$

Tall Wumpii tend to have high resilience to gold lurgy
$R4$ **Ante:** $height(X) : \{t, s\}$
Conse: $resil(X) : \{h, l\}$

Low resilience leads to gold lurgy
$R5$ **Ante:** $resil(X) : \{h, l\}$
Conse: $lurgy(X) : \{+, -\}$

Old Wumpii have low resilience
$R6$ **Ante:** $age(X) : \{o, y\}$
Conse: $resil(X) : \{h, l\}$

Algorithm 2 FINDEXPANSIONS

Input: *Path*, *KB*, *Visited*
Output: *Expansions*
Result ← [ ]
for each rule $R \in Path$ **do**
  for each $A \in$ **Ante**(R) **do**
    if $A \in Visited \vee [\exists R^*.\ R^* \in Path \wedge A =$ **Conse**$(R^*)]$ **then**
      continue
    else
      $Set \leftarrow \{R' \mid R' \in KB \wedge A =$ **Conse**$(R')\}$
      $Visited \leftarrow A : Visited$
      for each $S \in Set$ **do**
        $P' \leftarrow S : Path$
        $Expansions^* \leftarrow$ FINDEXPANSIONS$(P', KB, Visited)$
        $Result \leftarrow Expansions^* : Result$
      end for
    end if
  end for
end for
if $|Result| = 0$ **then**
  $Result \leftarrow [Path]$
end if
Expansions ← FILTERINACTIVE(*Result*)

Fig. 2: The FINDEXPANSIONS algorithm.

The **Matrix** of each rule has been omitted for brevity. $R6$, for example, may be associated with the **Matrix**:

	$age(X) = o$	$age(X) = y$
$resil(X) = h$	0.15	0.80
$resil(X) = l$	0.85	0.20

D's knowledge base also consists of rules defining: prior probabilities for the states of $lurgy(X)$, $height(X)$, $age(X)$, and $rash(X)$; and observations for $rash(W) = +$, $steal(W) = +$, $height(W) = t$, and $age(W) = o$. A rule expressing the prior probability of a variable v in each of its possible states has an empty **Ante**, variable v as its **Conse**, and the prior probabilities forming its **Matrix**. A rule representing an observation of variable v has v as its **Ante**, the observed state of v as its **Conse**, and the reliability of the observation as its **Matrix**.

Assume D would like to determine the probability that W has gold lurgy given the observations that D has made. This corresponds to the query:

$$Pr(lurgy(W) = +|O(rash(W), +), O(steal(W), +), O(height(W), t), O(age(W), o))$$

where $O(v, s)$ denotes that D has observed v to be in state s. Given this query, D constructs 6 arguments, according to Definition 3 and the algorithm presented above, representing potential paths in the final network. These arguments are shown in Figure 3, where dotted arrows denote rules representing prior probabilities.

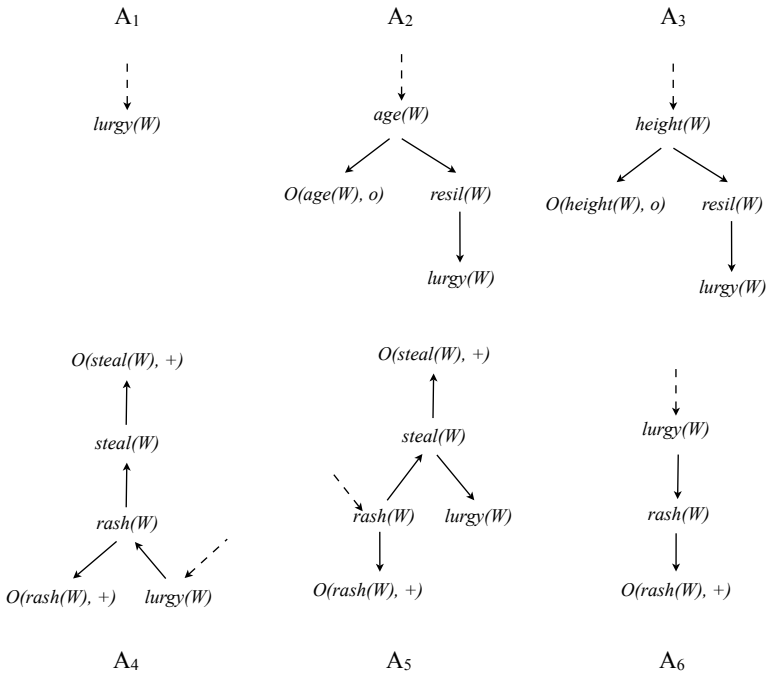

Fig. 3: Arguments constructed in the Wumpus example.

Given a query, $Pr(Q|E)$, not all arguments formed in this section will represent valid, acceptable, or reliable chains of influence in the context E. In the following section, a mechanism for filtering out unacceptable information from consideration is presented.

6 Reasoning about the Validity of Information

Once arguments have been formed, they must be compared to determine which of them should *not* form part of the network being constructed. Given a query, $Pr(Q|E)$, to determine whether an argument A invalidates an argument B in the context E, we use an attack relation $\mathcal{D}(A, B, KB, E)$. $\mathcal{D}(A, B, KB, E)$ holds if argument A invalidates (or *attacks*) argument B given E and knowledge base KB. This attack relation may be based on specificity, or the context-dependence of relationships (amongst other things).

Only *undefeated* arguments are considered in network formation. An argument $A \in AR$ is defeated if there exists an argument $B \in AR$ such that $\mathcal{D}(B, A, KB, E)$, and B is undefeated. An argument $B \in AR$ is undefeated if

there does not exist an argument $C \in AR$ such that $\mathcal{D}(C, B, KB, E)$ or for each argument $C \in AR$ where $\mathcal{D}(C, B, KB, E)$, C is defeated. An argument that may be used to attack or defend itself (directly or indirectly) is *not* undefeated. The set of undefeated arguments represents the grounded extension of a Dung argumentation framework $\langle AR, \mathcal{D} \rangle$ [3] where AR is the set of arguments representing information relevant in the evaluation of a query. This extension represents the subset of AR, $S \subseteq AR$, that are not attacked by any argument, in addition to the set of arguments $T \subseteq AR$ that are defended by arguments in S, in addition to the set of arguments $U \subseteq AR$ that are defended by arguments in T, and so on.

An agent may wish to invalidate a line of reasoning (or flow of information) between two variables given its preferences. Assume in the Wumpus example that D has the following additional knowledge, that (i) where age influences the state of a variable, the variable is completely independent of height if a (sufficiently reliable) observation of old age has been made, and that (ii) if a Wumpus has been observed (with sufficient reliability) to be tall and *height* is considered a factor in contracting lurgy, then the state of *lurgy* is completely independent of gold stealing. In this situation, argument A_2 in Figure 3 attacks argument A_3, and argument A_3 attacks arguments A_4 and A_5. The grounded extension of the resulting argumentation framework consists of all arguments except A_3.

An agent may wish to form a network where, for each node, only the most specific influences are included. Let $a >_s b$ denote that a is more specific than b. It might use the following criteria to determine if an argument $A \in AR$ attacks an argument $B \in AR$: if (i) $R_A \in A$ and $R_B \in B$, where $\mathbf{Conse}(R_A) = \mathbf{Conse}(R_B)$, and (ii) there exists antecedents $a \in \mathbf{Ante}(R_A)$ and $b \in \mathbf{Ante}(R_B)$ such that $a >_s b \in KB$, and (iii) there is a sufficiently reliable observation $O(a, +)$, then A attacks B by specificity.

In [1], Bacchus suggests the use of specificity to transform a knowledge base with multiple causes for an event into one with only the most specific causes. This is equivalent to ensuring that constraint C3 (Section 4) holds in the knowledge base. By considering the context in which these causes influence a query node – the paths on which they exist – much more general preferences over knowledge may be modelled. Argumentation over paths allows rules, which may be less specific than other causes present in the knowledge base, to be included when these more specific causes cannot be used to form an acceptable path. Argumentation over paths, as opposed to argumentation over individual rules, allows both direct and indirect context-dependent relationships – relationships that exist only in certain contexts – to be detected and removed from consideration in network building.

The argumentation stage presented in this section has an $\mathcal{O}(n^2)$ complexity, where n is the number of arguments that are formed. This stage consists of several steps: forming an attack graph with arguments as vertices and attacks as edges ($\mathcal{O}(n^2)$), removing all vertices that participate in an attack cycle ($\mathcal{O}(n^2)$ using depth first search at each vertex), finding the undefeated arguments given an acyclic attack graph ($\mathcal{O}(|n| + |\mathcal{D}|)$ where $|\mathcal{D}|$ is the number of edges) [3]. The number of edges in the attack graph is at most n^2.

7 Finding and Selecting Candidate Networks

In this section an accrual operator [8] that collects arguments and combines them to form a candidate Bayesian network is defined.

Definition 4 *Given a query, $Pr(Q|E)$, a group of arguments G may be combined to form an accrual iff:*

(i) each argument $A \in G$ is valid, and satisfies Definition 3,
(ii) the collection of ground rule instances in G does not violate constraints C1, C2, and C4 (Section 4),
(iii) no argument $O \notin G$ may be added to G such that conditions (i) – (ii) are not violated.

The principles of accrual [8] map nicely to the properties of the networks an agent would like to construct. An agent does not want any unreliable information in the network it constructs (i), and it would like to capture the largest amount of relevant information in each of the networks it considers (iii). The networks are maximal in the sense that no further reliable and relevant information can be added to them without violating the structural properties of a Bayesian network. A network $N \subset BN$ will never be a better model of the agent's environment than BN. Given two distinct accruals, where one is not a subset of the other, it is the importance of the information in each that will then determine which is the better model.

In the Wumpus example, two argument collections may be formed that satisfy the constraints in Definition 4, namely $\{A_1, A_2, A_5\}$ and $\{A_1, A_2, A_4, A_6\}$, assuming that A_3 has been considered unacceptable (Section 6). Each accrual represents a Bayesian network candidate – the union of rule instances in each argument in an accrual forms a network in the same way that an individual argument represents a network. Considering the arguments in Figure 3, it is evident that no extra arguments may be added to either of these collections without creating a cycle.

These accruals may be naively formed by considering every possible unique subset of valid arguments, and checking if it satisfies Definition 4. Given n arguments, there are 2^n of these subsets. A more intelligent algorithm should take advantage of the constraints in Definition 4, reducing the number of subsets that need to be generated and checked. Figure 4 defines such an algorithm, called FindCandidates, where: Difference(A, B) returns the elements in the set B that are not in the set A, SatisfiesConstraint(S) denotes that the set S satisfies constraint (ii) in Definition 4, Covers(S, *Cons*) denotes that S contains one element from each of the sets in *Cons*, and FindCover(*Cons*, N) determines if there is a set (of N or less elements) consisting of one element from each of the sets in *Cons* that satisfies constraint (ii) in Definition 4.

FindCandidates checks the largest subsets first, and ensures that all other subsets checked are not subsets of an accrual that has already been found (line 17). Given a set of n arguments, the complexity of finding accruals by checking all subsets is $\mathcal{O}(2^n)$. Using the above algorithm, the number of sets that must

Algorithm 3 FindCandidates

```
1: Input: AR
2: Output: Accruals
3: Cons ← [ ]
4: Accruals ← [ ]
5: N = |AR|
6: Set ← [{AR}]
7: for each S ∈ Set do
8:    if SatisfiesConstraint(S) then
9:       Accruals ← S : Accruals
10:      Cons ← [Difference(S, AR)] : Cons
11:   end if
12: end for
13: if |Cons| > 0 ∧ ¬FindCover(Cons, N − 1) then
14:   stop
15: end if
16: N ← N − 1
17: Set ← all unique subsets S′ ⊆ AR of size N where Covers(S′, Cons)
18: go to line 7
```

Fig. 4: The FindCandidates algorithm.

be checked is $\binom{n}{n} + \binom{n}{n-1} + \binom{n}{n-2} + \ldots + \binom{n}{n-d}$ where $n-d$ is the size of the smallest accrual that may be formed. This is equivalent to $\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \ldots + \binom{n}{d} = \mathcal{O}(n^d)$.

Given a set of candidates, an agent must compare them to determine which network should be selected as the end result. This end result must be a network that is as good as, or better than, any candidate network the agent may produce. This network may not contain all of the evidence nodes in the query $Pr(Q|E)$ it was designed to evaluate, in which case the query must be approximated with $Pr(Q|E_S)$ where $E_S \subseteq E$ is the subset of E that appears in the network. We assume that an agent has a mechanism for ranking candidates according to its preferences, where the set of available ranks is ordered, and an agent prefers candidates of a rank R to all candidates of rank $R' < R$.

For example, an agent may prefer one candidate to another if it includes a greater number of evidence nodes from E. In the Wumpus example, the two candidate networks include the same number of observations appearing in the query. Utilising this criteria leads an agent to believe that the two candidates are equally good. In a sense, an agent *votes* for a candidate network if it is ranked highest amongst all of its choices. Viewing the candidate selection process as voting provides a good platform for extending the approach to a multi-agent domain.

This paper has presented a tool that allows a *single* agent to argumentatively construct a Bayesian network, given a *single* query. It would not be hard to extend the approach to construct a network for several queries, and there is potential to extend the approach for use by multiple agents. Multiple agents may

collaborate to construct a network for a query or multiple queries, constructing arguments from their respective knowledge bases. If the union of the knowledge bases of each agent does not represent a valid network, argumentation over the active paths present in each knowledge base allows a compromise to be achieved.

In a multi-agent setting, attack amongst arguments must incorporate the knowledge possessed by each agent. For example, one agent may be aware of some situations in which a relationship does not hold, and another agent may be aware of others. Additionally, arguments representing networks can attack each other according to different criteria, and the application of these criteria may be a source of disagreement amongst agents. Multiple agents must be able to argue about whether observing a Wumpus to be old, for example, really does indicate independence of *lurgy* from *height*. The case in which information forming an active path is spread across multiple agents must also be considered in a multi-agent setting.

Once a set of candidate networks is agreed upon, voting theory may be used to manage the preferences of different agents to select the best candidate. In a simple voting process, each agent may cast a vote for the candidates that it most prefers according to its preferences. The candidate with the most votes is selected as the best candidate.

8 Conclusion and Future Work

Unlike traditional knowledge based model construction (KBMC) techniques for Bayesian networks, argumentative KBMC aims to allow an agent to construct a network that maximises the amount of reliable and relevant information incorporated and, from the perspective of the agent, is the most preferred. The correctness of the technique may be determined by showing that the result does in fact represent a network that is as good, or better, than any other network the agent may produce.

Given a query, $Pr(Q|E)$, arguments AR are constructed to represent each active path through the knowledge base of an agent. An argument contains reliable information only if it is undefeated by other arguments that may be formed, assuming that the defeat relation amongst arguments $\mathcal{D}$ encodes all ways in which an argument may become invalid given the current context. The set of acceptable arguments produced after the first stage of argumentation represents the grounded extension of a Dung argumentation framework $\langle AR, \mathcal{D} \rangle$. The accrual process forms all subsets of this extension that represent a valid Bayesian network.

In the single agent case, selection of the best network resolves to selecting the candidate that is ranked highest according to its preferences. In a multi-agent setting, each agent may *vote* for the candidates it most prefers. Votes must then be analysed to determine the most preferred candidate for the collection of agents. Assuming that an agent's preferences are completely expressed in its ranking, the only way in which an agent may select a candidate network that is not one of the best is if there are candidates that have not been constructed. As the approach is designed to find all acceptable arguments, and all valid accruals

based on these arguments, the approach is both sound and complete in the single agent setting.

At present, this approach has been demonstrated with a simple example. As future work, we would like to evaluate the technique with real world examples to demonstrate its usefulness beyond toy problems. Additionally, while we have compared aspects of this work with existing KBMC approaches, we would like to analyse the similarities and differences between our work and other argumentative approaches for constructing Bayesian networks, particularly the work of Nielsen and Parsons in [6].

In [6], a technique for fusing individual and possibly inconsistent Bayesian networks using concepts from the field of argumentation is presented. A finite number of agents, each maintaining a Bayesian network expressing relationships amongst the same variables, engage in a debate to construct a compromise network – a preferred extension of their own networks. Candidate networks are constructed by examining which variables should be linked by a directed edge, and assigned a score by each agent. The candidate with the highest combined score is selected as the compromise network.

In this paper, only a sketch of how our approach may be extended to a multi-agent domain has been provided. In the future, we would like to explore this extension in more detail.

References

1. F. Bacchus. Using First-Order Probability Logic for the Construction of Bayesian Networks. In *UAI*, pages 219–226, 1993.
2. J. S. Breese. Construction of belief and decision models. *Computational Intelligence*, 1992.
3. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n-person games. *Artificial Intelligence*, 77:321–357, 1995.
4. S. Glesner and D. Koller. Constructing flexible dynamic belief networks from first-order probabilistic knowledge bases. In *ECSQARU*, pages 217–226, 1995.
5. P. Haddawy. Generating Bayesian networks from probability logic knowledge bases. In *UAI*, pages 262–269. Morgan Kaufmann, 1994.
6. S. H. Nielsen and S. Parsons. An application of formal argumentation: Fusing Bayesian networks in multi-agent systems. *Artificial Intelligence*, 2007.
7. J. Pearl. *Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.
8. H. Prakken. A study of accrual of arguments, with applications to evidential reasoning. In *ICAIL*, pages 85–94, 2005.
9. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2002.
10. M. P. Wellman, J. S. Breese, and R. P. Goldman. From knowledge bases to decision models. *The Knowledge Engineering Review*, 7:35–53, 1992.