TR-01-06: Fusing Bayes Nets using Formal Argumentation in Multi-agent Systems

Søren Holbech Nielsen
Department of Computer Science
Aalborg University, Aalborg
Denmark
holbech@cs.aau.dk

and

Simon Parsons
Department of Computer and Information Science
Brooklyn College, City University of New York
Brooklyn, 11210 NY, USA
parsons@sci.brooklyn.cuny.edu

June 26, 2006

Abstract

We consider a multi-agent system where each agent is equipped with a Bayesian network, and present an open framework for the agents to compromise on a possible consensus network. The framework builds on formal argumentation, and unlike previous solutions on graphical consensus belief, it is sufficiently general to allow for a wide range of compromises to be identified.

1 Introduction

In the wake of the digital revolution of the last few decades, sophisticated electronic devices have been installed everywhere in society, and with the introduction of cheap networking equipment, communication between these devices has become widespread. Consequently research in distributed systems has intensified, and within this wide field, the crossover area of multi-agents systems (MAS), that incorporates parts of artificial intelligence research, is a heavily researched field. However, not all areas of artificial intelligence research have seen an equal degree of employment in the setting of MAS. Bayesian networks (BNs), in particular, have so far mostly been treated as isolated reasoning machines for the individual agents, with the notable exception of the work of Xiang (e.g. (Xiang and Lesser 2003)), who considers a series of BNs held by individual agents to be a single BN. Contrarily, formal logics have had a significant impact on theory of MAS. Failure in face of inconsistent information has traditionally been one of formal logic's weak points, but recent development of non-monotonic approaches to logic has pretty much eliminated that weakness. One non-monotonic approach that is inherently applicable to MAS is the theory of argumentation systems, which is based on human dialectic inquiry.

In this text we investigate how BNs can be used in a multi-agent setting with the help of argumentation theory. Previously the two methodologies have mainly been studied together with a view to incorporate the efficiency and precision of BNs into argumentation theory (e.g. (Saha and Sen 2004)), or as an exercise in converting models of one theory into models of the other (e.g. (Vreeswijk 1999) and (Vreeswijk 2004)). Here, we instead try to exploit strong points of both reasoning methods: BNs constitute an elegant framework for drawing diagnostic or causal inferences from observations to hypothesis variables, even in the face of noisy observations, and they can be constructed automatically from observation data. Argumentation, on the other hand, provides a transparent system for extracting a consistent "truth" from a set of conflicting and/or overlapping ones, and furthermore has strong roots in dialectics, which makes distributed implementations natural.

Playing along with these strengths, we envision equipping each agent in a MAS with a BN, as a model of the domain it is situated in, and aim at providing a framework built on formal argumentation principles in which the agents, taking outset in their individual domain models, can conclude on a single network representing their joint domain knowledge. More concretely, the problem we address take outset in a MAS containing a finite number of cooperating agents — cooperation entailing that agents do not lie to each other nor withhold requested information or information that they deem to be relevant. We assume that agents in the MAS have access to perfect communication channels, and that all agents share a common language and agree on the exact semantics of this language.

The agents are continually observing their surroundings, and have a priori been asked to focus on a set of domain variables. As it operates, each agent is constructing and maintaining a probabilistic model over these variables in the form of a BN. At some point in time a set of the agents meet up/decide to communicate, and the task is to pool their knowledge, as represented by the BNs, and end up with a new BN that is a compromise or distillate of the original networks. We establish a formal argumentation framework where such a compromise network can be discussed.

The task of fusing several BNs into one compromise BN has previously been addressed with an a priori specified view to what constitutes a compromise ((Matzkevich and Abramson 1992), (Pennock and Wellman 1999), (II and Chajewska 2001), (Richardson and Domingos 2003), (Del Sagrado and Moral 2003)), with no apparent consensus on the goal of network fusion among the authors, and has mainly been considered a centralized operation. Matzkevich and Abramson (1992) disregards the strength of independency statements in the input networks, and seek to obtain a graph that contains all arcs from the input networks or their reverses. Del Sagrado and Moral (2003) similarly disregards the quantitative parts of the input networks, and gives rules for constructing graphs that imply either all independency statements implied by at least one of the input BNs, or only those independency statements implied by all input BNs. Richardson and Domingos (2003) also disregards the quantitative parts of the input networks, and construct a prior distribution over all graph structures from the input BNs. This prior is then used as basis for a standard greedy search based on a separate database. Finally, Pennock and Wellman (1999) derives a series of impossibility results for the general problem of combining probability distributions, and II and Chajewska (2001) uses some of the results from (Pennock and Wellman 1999) to adapt standard greedy search score functions to use the input BNs, rather than a database, as their basis. The differing objectives of these papers stems from differences in interpretations of BNs: They are seen as either specifying flow of information, specifying independencies between variables, representing expert experience, or summarising data. In this paper, we do not commit ourselves to any specific compromise objective. Rather, we establish a general framework in which any kind of compromise can be reached, with the exact nature of this specified by a compromise score function and possibly a heuristic for walking search trees. The advantages of our approach include that a general purpose argumentation engine can be implemented, and reused in contexts with different definitions of compromise; efficient distributed implementations are natural; in cases where agents almost agree a priori, little information need to be shared among the agents; and anytime compromises can be achieved.

The text is structured as follows: In Section 2 we briefly cover the methodology that we need. In particular, we need some concepts of graph theory, the main theory of BNs and their equivalence classes, and finally theory of an argumentation framework. Following this, in Section 3 we specify the problem that we wish to solve more formally, and present the strategy for doing so. The actual results then follows. First, Section 4 describes how we encode BNs in our framework. Then Section 5 presents a formal argumentation system whose preferred extensions correspond to proper BNs, and Section 6 contains the actual debating guide-lines that must be followed by agents. We end with a discussion of these results in Section 7.

2 Preliminaries

Before we proceed with the theory needed for the results later in the paper, we briefly clarify the notation: As a general rule, sets are printed in boldface type ($\boldsymbol{S}, \mathbf{pa}(V), \ldots$), and individual entities are printed in plain letters ($A, x_i, \ldots$), except data structures, which are printed in calligraphic letters ($\mathcal{G}, \mathcal{A}, \ldots$), and general classes, which are printed in fractured type ($\mathfrak{C}$). As convention we use $\equiv$ to mean "is defined to be" or "defined as", and decline to use { and } when listing singleton sets. The material is necessarily heavy on definitions, and to help the reader refer back, each term is *emphasized* where it is defined.

2.1 Graphs

Here we briefly introduce the graph theory that is used in the remainder of the text. For more elaboration on the concepts introduced, see e.g. (Lauritzen 1996).

A *graph* is a pair $\mathcal{G} \equiv (\boldsymbol{X}, \boldsymbol{E})$, where $\boldsymbol{X}$ is the finite set of *nodes* of the graph, and $\boldsymbol{E} \subseteq (\boldsymbol{X} \times \boldsymbol{X}) \setminus \{(X, X) \ : \ X \in \boldsymbol{X}\}$ is the set of *edges* of the graph. For any two nodes X and Y, if both (X, Y) and (Y, X) is in $\boldsymbol{E}$ we say that there is an *undirected link* (or just a *link*) between X and Y and that they are *neighbours*. If only (X, Y) is in $\boldsymbol{E}$, then we say that there is an *arc from* X to Y, that X is a *parent* of Y, and that Y is a *child* of X. The set of parents of a node X is denoted $\mathbf{pa}(X)$. If X is a neighbour, parent, or child of Y, we say that X and Y are *adjacent*. A triple of nodes (X, Y, Z) is said to constitute a *v-structure*, if Y is a child of both X and Z, and X and Z are not adjacent. When depicting a graph we use circles for nodes, lines for links, and arrows for arcs, as shown in Figure 1. From the figure it can be seen that $\mathbf{pa}(C) = G$, that C is a neighbour of both B and D, that C has no children, and that (A, B, F) is the only v-structure in the graph.

For two nodes X_1 and X_k, we say that there is a *path* from X_1 to X_k, if either (X_1, X_k) is in $\boldsymbol{E}$, or there are distinct nodes $X_2, \ldots, X_{k-1}$ different from X_1 and X_k, such that (X_{i-1}, X_i) is in $\boldsymbol{E}$ for all $2 \leq i \leq k$. We denote

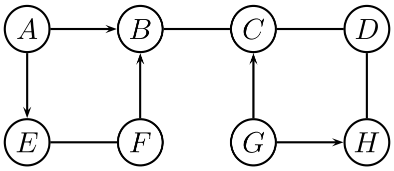


Figure 1: *A simple graph.*

the path $(X_1, \ldots, X_k)$ and say that its *length* is $k-1$. If, for one of the edges (X, Y) in a path, (Y, X) is not in $\boldsymbol{E}$ then we say that the path is *directed*, if not it is *undirected*. A path $(X, \ldots, Y)$, where X and Y are the same node, we call a *cycle*. Given a cycle $(X_1, \ldots, X_{k-1}, X_1)$ of length $k \geq 4$, we say that the cycle has a *chord* if there is a pair of nodes X_i and X_j, where $|i-j| \geq 2$ and X_i and X_j are not the pair X_1 and X_{k-1}, such that X_i and X_j are adjacent. A cycle of length $k \geq 4$ with no chord is called *chordless*. If there is a directed path from a node X to a node Y, then we say that Y is a descendant of X. The set of descendants of X is denoted by $\mathbf{desc}(X)$. If, for any two nodes X and Y in a set $\boldsymbol{Y}$, either X is the same node as Y, or there is an undirected path from X to Y, and this holds for no other set $\boldsymbol{Z} \supset \boldsymbol{Y}$, then we call $\boldsymbol{Y}$ a *chain component*. For examples, consider the graph in Figure 1: There are no cycles in the graph, $\mathbf{desc}(A) = \{B, C, D, E, F, H\}$, and $\{B, C, D, H\}$ is a chain component.

Given a graph $\mathcal{G} \equiv (\boldsymbol{X}, \boldsymbol{E})$, we introduce some auxiliary definitions. Let $\boldsymbol{Y} \subseteq \boldsymbol{X}$, then we say that $\mathcal{G}_{\boldsymbol{Y}} \equiv (\boldsymbol{Y}, \boldsymbol{E} \cap (\boldsymbol{Y} \times \boldsymbol{Y}))$ is the *subgraph induced* by $\boldsymbol{Y}$. If all the edges in $\mathcal{G}$ are arcs, we say that $\mathcal{G}$ is *directed*. If all the edges are links, we say that $\mathcal{G}$ is *undirected*. A graph, which is neither directed nor undirected, is a *partially directed graph*. An undirected graph containing no chordless cycles is called *decomposable*. A graph with no directed cycles is called a *chain graph*. A directed chain graph is also called an *acyclic directed graph*, traditionally abbreviated *DAG*. The graph $\mathcal{G}^u \equiv (\boldsymbol{X}, \{(X, Y) : (X, Y) \in \boldsymbol{E} \text{ or } (Y, X) \in \boldsymbol{E}\})$, obtained by replacing each arc in $\mathcal{G}$ with a link, is called the *skeleton* of $\mathcal{G}$.

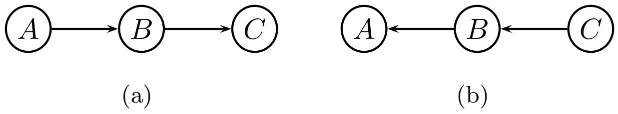


Figure 2: *Two BNs with the same independence properties.*

2.2 Bayesian Networks

A *Bayesian network* (BN), over a set of discrete[1] domain variables $\boldsymbol{V}$, is a pair $(\mathcal{G}, \boldsymbol{\Phi})$, where $\mathcal{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ is a DAG, and $\boldsymbol{\Phi}$ is a set of conditional probability distributions:

$$\boldsymbol{\Phi} \equiv \{P(V|\mathbf{pa}(V)) \ : \ V \in \boldsymbol{V}\}.$$

In the sequel we assume $\boldsymbol{V}$ to be fixed and implicitly constitute the basis for the nodes of all graphs. For more information on BNs in general, see (Jensen 2001).

A BN provides a probabilistic model P' of the domain $\boldsymbol{V}$, such that for any configuration $\boldsymbol{v} \equiv (v_1, \ldots, v_n)$ over $\boldsymbol{V}$,

$$P'(\boldsymbol{v}) \equiv \prod_{\boldsymbol{\Phi}} P(V_i = v_i | \mathbf{pa}(V_i) = \boldsymbol{v}_{\mathbf{pa}(V_i)}),$$

with $\boldsymbol{v}_{\mathbf{pa}(V_i)}$ denoting the entries in the configuration $\boldsymbol{v}$ that corresponds to the variables in the set $\mathbf{pa}(V_i)$. For any BN, the limitations on P' imposed by the structure of $\mathcal{G}$ are summarised as a set of independence constraints:

$$\big\{V \perp\!\!\!\perp \boldsymbol{V} \setminus (\mathbf{desc}(V) \cup \mathbf{pa}(V) \cup V) \mid \mathbf{pa}(V) \ : \ V \in \boldsymbol{V}\big\},$$

where the notation $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$, for three disjoint sets of variables $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$, means that the variables in $\boldsymbol{X}$ are probabilistically conditionally independent of those in $\boldsymbol{Y}$ given those in $\boldsymbol{Z}$, i.e.

$$P'(\boldsymbol{X} \cup \boldsymbol{Y}|\boldsymbol{Z}) = P'(\boldsymbol{X}|\boldsymbol{Z}) \cdot P'(\boldsymbol{Y}|\boldsymbol{Z}),$$

whenever $P'(\boldsymbol{Z}) > 0$. In other words, the set of probability models that can be associated to some DAG to constitute a BN, all need to obey the independence constraints dictated by the structure (Pearl 1988).

[1] "Discrete" meaning that a variable can be in one of only a finite number of states. Other equivalent terms include "cardinal" and "ordinal".

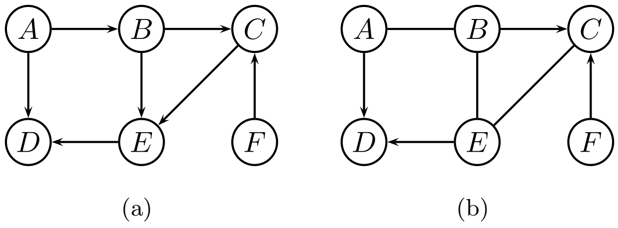


Figure 3: *(a) A graph and (b) its pattern.*

Two distinct DAGs may give rise to the same independence constraints as can be seen from the two graphs in Figures 2(a) and 2(b): They imply the independence statements $A \perp\!\!\!\perp C \mid B$ and $C \perp\!\!\!\perp A \mid B$, respectively, which by the commutativity of standard multiplication are the same. Verma and Pearl (1990) showed that any two DAGs imply the same independence constraints if and only if (iff) they contain the same v-structures and have the same skeleton.

We can thus define an equivalence relation over DAGs, such that $\mathcal{G}$ and $\mathcal{H}$ are equivalent iff the independence constraints implied by $\mathcal{G}$ are exactly those implied by $\mathcal{H}$. Thanks to the result of (Verma and Pearl 1990), we can represent the equivalence class of a DAG $\mathcal{G}$ as the graph having the same skeleton as $\mathcal{G}$, the same v-structures, and all edges not participating in defining a v-structure being undirected links. Following Verma and Pearl (1990) we call this graph the *pattern* of the equivalence class. As an example, the pattern of the equivalence class of the DAG in Figure 3(a) is shown in Figure 3(b).

Once a pattern has been determined for a DAG all the members of its equivalence class can be constructed by exchanging links for arcs in all possible ways under the constraints that

1. the resulting graphs contains no directed cycles, and
2. no other v-structures than those of the pattern emerge.

These two constraints sometimes cause one or more of the arcs replacing links in the pattern to have the same orientation in the resulting DAGs, no matter in what order or direction other links are converted. For instance, the link between C and E in the pattern in Figure 3(b) cannot be exchanged with an arc going from E to C, as that would create a new v-structure (E, C, F). Hence, it must be exchanged for an arc from C to E in all DAGs

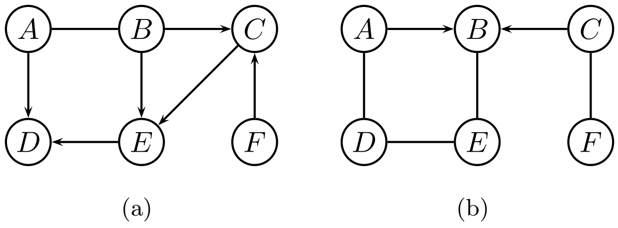


Figure 4: *The completed pattern of the graph in Figure 3(a) and a graph with no consistent extension.*

created from that pattern. The arcs arising from replacing such links, and arcs participating in defining the v-structures in the pattern, we collectively call *compelled* arcs. The graph that results from replacing links for compelled arcs wherever possible is called the *completed pattern.* Notice that by definition this graph must necessarily be unique for each equivalence class. The completed pattern of the DAG in Figure 3(b) is shown in Figure 4(a).

Now, consider the graph in Figure 4(b). No matter how we try to exchange the links in the graph with arcs we end up either creating a directed cycle or a v-structure not present in the original graph. Thus, this graph cannot be a pattern for any equivalence class of DAGs nor be a result of extending a pattern while respecting Constraints 1 and 2, and we are therefore dealing with two classes of graphs: Those graphs that can be turned into at least one DAG, by exchanging links for arcs while respecting Constraints 1 and 2, we say *admit a consistent extension*, whereas those that cannot *do not admit a consistent extension* (terminology originally introduced by Chickering (2002)). We denote the class of graphs admitting a consistent extension as $\mathfrak{C}$. As a DAG has itself as a consistent extension, it follows that any DAG $\mathcal{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ is a member of $\mathfrak{C}$. Furthermore, the pattern $\mathcal{G}^p \equiv (\boldsymbol{V}, \boldsymbol{E}^p)$ of $\mathcal{G}$ must necessarily be a member of $\mathfrak{C}$, as must any graph $\mathcal{G}' \equiv (\boldsymbol{V}, \boldsymbol{E}')$, where $\boldsymbol{E} \subseteq \boldsymbol{E}' \subseteq \boldsymbol{E}^p$, including the completed pattern of $\mathcal{G}$. We shall also distinguish completed patterns from other members of $\mathfrak{C}$, i.e. $\mathfrak{C}^{cp} \subset \mathfrak{C}$ is the class of graphs that are completed patterns of at least one DAG.

Unfortunately, we are unaware of any simple characterization of members of $\mathfrak{C}$. That is, potentially trying out all possible directions of undirected links in the hope of obtaining a consistent extension, besides requiring global investigation of the graph rather than a series of local investigations, is not simple, but a computationally horrible procedure, potentially calling for a lot of back tracking. However, Andersson, Madigan, and Perlman (1997) provides the following result on members of $\mathfrak{C}^{cp}$:

Theorem 1. *A graph $\mathcal{G}$ is a member of $\mathfrak{C}^{cp}$ iff*

1. *$\mathcal{G}$ is a chain graph,*
2. *$\mathcal{G}_{\boldsymbol{C}}$ is decomposable for all chain components $\boldsymbol{C}$ of $\mathcal{G}$,*
3. *if X is a parent of Y, and Z is a neighbour of Y, then X and Z are adjacent (i.e. the configuration in Figure 5(a) does not exist as a subgraph of $\mathcal{G}$), and*
4. *each arc in $\mathcal{G}$ is* strongly protected *(defined below).*

An arc from a node X to a node Y in a graph $\mathcal{G}$ is strongly protected if either

- *there is a node Z that is a parent of X and not adjacent to Y (see Figure 5(b)),*
- *there is a node Z that is a parent of Y and not adjacent to X (see Figure 5(c)),*
- *there is a node Z that is a parent of Y and a child of X (see Figure 5(d)), or*
- *there are non-adjacent nodes Z and W, such that both Z and W are parents of Y and neighbours to X (see Figure 5(e)).*

The bullets of the theorem all have pretty intuitive justifications. In short, Bullets 1, 2, and 3 ensure that all compelled arcs are arcs, whereas Bullet 4 ensure than nothing but compelled arcs are arcs. In particular, Bullet 1 calls for the graph to be a chain graph, which is reasonable, since any graph with a fully directed cycle cannot be extended to a DAG, and any graph with a directed cycle still containing undirected links cannot be extended to a DAG without creating a new v-structure, unless the cycle is of length 3, in which case the arc(s) participating in the cycle cannot be strongly protected, and hence would be forbidden by Bullet 4.

Bullet 2 is reasonable, since any graph with a chordless undirected cycle cannot be extended into a DAG without creating a new v-structure.

A graph failing to satisfy Bullet 3 cannot be extended into a DAG without creating a new v-structure, unless the undirected link between Y and Z is directed away from Y. But then this arc is compelled, and it should have been an arc if the graph was to be a completed pattern.

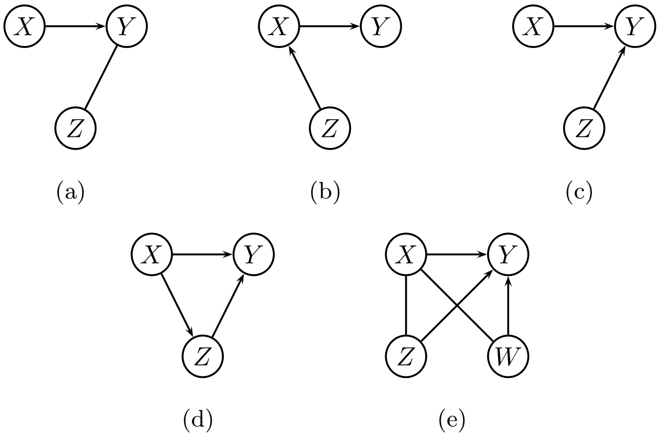


Figure 5: *Configurations described in Theorem 1.*

As for Bullet 4, consider that for any arc in a completed pattern, it holds by definition that this arc cannot be reversed without changing the set of v-structures or introducing a directed cycle. Going through each case of the definition of strongly protected, it is easy to see that changing the direction of these arcs does result in a cycle, a new v-structure, or the destruction of a v-structure.

Clearly, both Bullets 3 and 4 are easily checked locally, and Bullets 1 and 2 can be checked without any backtracking.

2.3 Argumentation Systems

Argumentation is a relatively new approach to extracting consistent knowledge from a possibly inconsistent knowledge base while possibly respecting some assumptions on the relative epistemological worth of statements in the knowledge base. The approach can be seen as an alternative to/generalization of formal non-monotonic and defeasible logics (Dung 1995). As the research area of argumentation systems is only just beginning to take form, no single methodology has yet to stand out as the main approach, and a lot of problems are still not solved. However, the methods show great potential as reasoning engines, which is why we use it here.

Due to the lack of an established framework for argumentation, it has been necessary to pick one from a large pool of these (e.g. (Simari and Loui 1992), (Elvang-Goransson and Hunter 1995), (Krause, Ambler, Elvang-

Goeransson, and Fox 1995), (Pollock 1995), and (Vreeswijk 1997)). The framework we have picked for our purpose is the framework of (Nielsen and Parsons 2006b) (which generalizes that of (Dung 1995)), as this is an abstract framework, which leaves the underlying language and reasoning unspecified.

An *argumentation system* is a pair $\mathcal{A} \equiv (\boldsymbol{A}, \rhd)$, where $\boldsymbol{A}$ is a set of *arguments*, and $\rhd \subseteq (2^{\boldsymbol{A}} \setminus \{\varnothing\}) \times \boldsymbol{A}$ is an *attack relation*. The exact nature of an argument is left unspecified, but examples built on a language akin to propositional logic could be:

- "The sun is shining, so it is not raining",
- "I saw a pigeon yesterday, so it is not raining", and
- "It is not raining".

For two sets of arguments $\boldsymbol{S} \subseteq \boldsymbol{A}$ and $\boldsymbol{S}' \subseteq \boldsymbol{S}$ and an argument A, if $\boldsymbol{S}' \rhd A$ then $\boldsymbol{S}$ is said to *attack* A, and A is said to *be attacked by* $\boldsymbol{S}$. If no proper subset of $\boldsymbol{S}'$ attacks A, then $\boldsymbol{S}'$ is called a *minimal attack* on A. An example of an attack that would intuitively make sense is

"The sun is shining"$\rhd$"It is raining".

A *semantics* of an argumentation framework is a definition of which arguments in the framework that should be accepted by a rational individual. (Dung 1995) and (Nielsen and Parsons 2006b) work with a wide range of semantics, but we only introduce those we need here. First of all, we define a set of arguments $\boldsymbol{S} \subseteq \boldsymbol{A}$ as being *conflict-free*, if there is no argument A in $\boldsymbol{S}$ such that $\boldsymbol{S}$ attacks A. Intuitively, claiming a conflict-free set of arguments as your beliefs implies that you are not contradicting yourself. We further define a single argument A as being *acceptable with respect to a set of arguments* $\boldsymbol{S}$, if for each set of arguments $\boldsymbol{T} \subseteq \boldsymbol{A}$, such that $\boldsymbol{T} \rhd A$, there is an argument B in $\boldsymbol{T}$, such that $\boldsymbol{S}$ attacks B. In that case we also say that $\boldsymbol{S}$ *defends* A. A conflict-free set $\boldsymbol{S}$, where all arguments in $\boldsymbol{S}$ are acceptable with respect to $\boldsymbol{S}$, is called *admissible*. Thus, claiming an admissible set of arguments as your beliefs means that you can defend your beliefs against all possible counter arguments.

A very skeptical attitude towards arguments is captured by a semantics called the *grounded extension*. The grounded extension of an argumentation framework is the least fixpoint of the function $F : 2^{\boldsymbol{A}} \to 2^{\boldsymbol{A}}$, defined as

$$F(\boldsymbol{S}) = \{A \in \boldsymbol{A} \ : \ A \text{ is acceptable wrt. } \boldsymbol{S}\}.$$

Thus the grounded extension consists of all the arguments that have no counter arguments, along with the arguments they defend, along with those defended by these arguments, and so on. A less skeptical semantics is that of a *preferred extension*, which is an admissible set that is maximal with respect to set inclusion. Finally, an admissible set $\boldsymbol{S}$ is said to be a *stable extension*, if it attacks all arguments in $\boldsymbol{A} \setminus \boldsymbol{S}$. Clearly, a stable extension is a preferred extension as well.

In general it is hard to compute a preferred extension (Dimopoulos, Nebel, and Toni 1999), but Doutre and Mengin (2001) present a method that enumerates preferred extensions for an abstract argumentation system as presented in (Dung 1995). Furthermore, Vreeswijk and Prakken (2000) and Cayrol, Doutre, and Mengin (2001) present methods for answering whether a specific argument is in at least one preferred extension, or if it is in all preferred extensions. In (Nielsen and Parsons 2006a), we have adapted the technique of (Doutre and Mengin 2001) to the problem of enumerating preferred extensions for argumentation systems, where sets of arguments attack other arguments. We present only the essentials of this technique here, and refer the interested reader to (Nielsen and Parsons 2006a) for the details:

Given an argumentation system $\mathcal{A} \equiv (\boldsymbol{A}, \triangleright)$, we define an $\mathcal{A}$-*candidate* as a triple $\mathcal{C} \equiv (\boldsymbol{I}, \boldsymbol{O}, \boldsymbol{U} \equiv \boldsymbol{A} \setminus (\boldsymbol{I} \cup \boldsymbol{U}))$ where

- $\boldsymbol{I} \cap \boldsymbol{O} = \varnothing$,
- every argument that is attacked by $\boldsymbol{I}$ is in $\boldsymbol{O}$, and
- every argument A, for which there exists $\boldsymbol{S} \subseteq \boldsymbol{I}$ and $B \in \boldsymbol{I}$, such that $\boldsymbol{S} \cup A \triangleright B$, is in $\boldsymbol{O}$.

Given an $\mathcal{A}$-candidate $\mathcal{C} \equiv (\boldsymbol{I}, \boldsymbol{O}, \boldsymbol{U})$ and an argument $A \in \boldsymbol{U}$ the triples $\mathcal{C}{-}A \equiv (\boldsymbol{I}_{-A}, \boldsymbol{O}_{-A}, \boldsymbol{U}_{-A} \equiv \boldsymbol{A} \setminus (\boldsymbol{I}_{-A} \cup \boldsymbol{O}_{-A}))$ and $\mathcal{C}{+}A \equiv (\boldsymbol{I}_{+A}, \boldsymbol{O}_{+A}, \boldsymbol{U}_{+A} \equiv \boldsymbol{A} \setminus (\boldsymbol{I}_{+A} \cup \boldsymbol{O}_{+A}))$ are given by:

$$\begin{aligned} &\boldsymbol{I}_{-A} \equiv \boldsymbol{I}, \qquad \boldsymbol{O}_{-A} \equiv \boldsymbol{O} \cup A, \\ &\boldsymbol{I}_{+A} \equiv \boldsymbol{I} \cup A, \text{ and } \quad \boldsymbol{O}_{+A} \equiv \boldsymbol{O} \cup \boldsymbol{\Delta}_{\mathcal{C}+A}, \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{\Delta}_{\mathcal{C}+A} \equiv \{B \in \boldsymbol{U} \ : \ & \exists \boldsymbol{S} \subseteq \boldsymbol{I},\ C \in \boldsymbol{I} \\ & \text{s.t. } \boldsymbol{S} \cup A \triangleright B \ \vee \ \boldsymbol{S} \cup B \triangleright A \\ & \quad \vee\ \boldsymbol{S} \cup \{A, B\} \triangleright C \ \vee \ \boldsymbol{S} \cup \{A, B\} \triangleright A\}. \end{aligned}$$

If A does not participate in a minimal attack on itself (which is the case for all arguments of the argumentation system we construct in this paper), then both $\mathcal{C}-A$ and $\mathcal{C}+A$ are $\mathcal{A}$-candidates themselves, and we can thus construct *candidate trees* where each node is an $\mathcal{A}$-candidate: Each $\mathcal{A}$-candidate $\mathcal{C}$ has two children $\mathcal{C} - A$ and $\mathcal{C} + A$, for some arbitrary chosen A in $\boldsymbol{U}$, except those candidates where $\boldsymbol{U} = \varnothing$, which act as leaves in the tree. A candidate tree having candidate $\mathcal{C}$ as root, is called a *$\mathcal{C}$-tree.*

It can be proved that if $\boldsymbol{S}^*$ is a preferred extension of $\mathcal{A}$, then there is a leaf $\mathcal{C} \equiv (\boldsymbol{I}, \boldsymbol{O}, \varnothing)$ of any $(\varnothing, \varnothing, \boldsymbol{A})$-tree such that $\boldsymbol{I} = \boldsymbol{S}^*$. Conversely, for any leaf in a $(\varnothing, \varnothing, \boldsymbol{A})$-tree, where $\boldsymbol{I}$ defends itself, $\boldsymbol{I}$ is admissible. It follows that, by constructing an arbitrary $(\varnothing, \varnothing, \boldsymbol{A})$-tree, all preferred extensions can be enumerated.

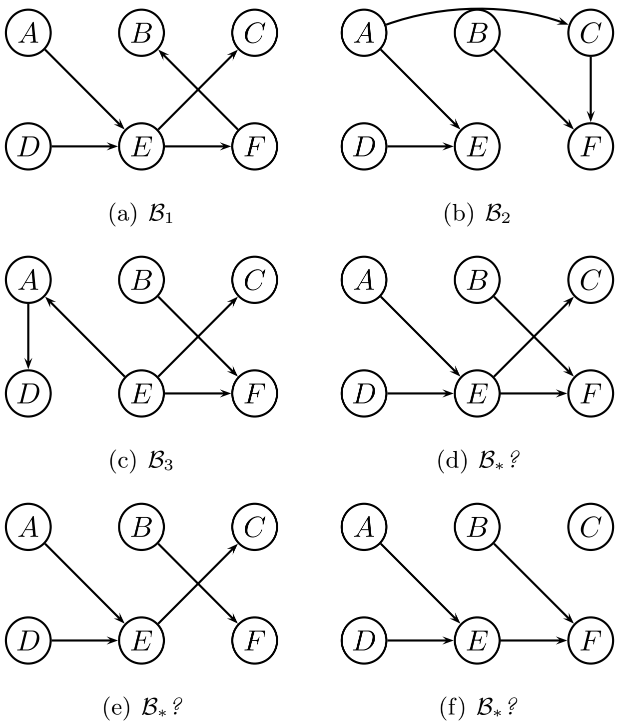
Nielsen and Parsons (2006a) gives a number of pruning rules for candidate trees, which can be used during construction to cut off branches, which cannot contain leaves with preferred extensions.

3 Compromising On Bayesian Networks

The problem we are posing arises in a MAS containing a finite number of cooperating agents. We assume an ordering over the agents exists, so that we can refer to them by integers. Each agent i has a BN $\mathcal{B}_i$ over a common set of domain variables $\boldsymbol{V}$, which we assume to be implicit in the remainder of the text. For ease of exposition, we furthermore assume that an arbitrary but fixed total ordering $\rightsquigarrow$ over the variables is known by all agents a priori. At some point agents 1 to k decide to pool their knowledge, as represented by $\mathcal{B}_1$ to $\mathcal{B}_k$, into a new BN $\mathcal{B}_*$, the *compromise* BN. Facilitating this task is the problem addressed here. We expect $\mathcal{B}_1$ to $\mathcal{B}_k$ to be large but somewhat similar (as each describe relationships among the same variables), and therefore that having each agent communicate its entire model to each other agent is inefficient.

We shall focus solely on the graphical structure of $\mathcal{B}_*$ (although we allow for non-structural aspects to act as guidelines in the construction of the structure). Therefore, the outcome of debate can be the full DAG of $\mathcal{B}_*$, its pattern, or its completed pattern. These should all be equivalent. However, as the next example shows, this is not entirely true.

Example 1 (Agreeing On A Compromise BN): *Consider three agents with BNs $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ portrayed in Figures 6(a) to 6(c). If the compromise is constructed by simple majority voting on the possible connections between each pair of nodes, the result is the structure in Figure 6(d). If the same is*



(a) $\mathcal{B}_1$ (b) $\mathcal{B}_2$

(c) $\mathcal{B}_3$ (d) $\mathcal{B}_*$?

(e) $\mathcal{B}_*$? (f) $\mathcal{B}_*$?

Figure 6: *Three BNs and three different compromise graphs.*

done for the patterns of $\mathcal{B}_1$ to $\mathcal{B}_3$ instead, the BN structure in Figure 6(e) is obtained. For the completed patterns the result is the one in Figure 6(f).

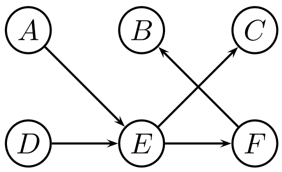
As can be seen, evaluating each feature in isolation leads to vastly different results, even though the starting points are the same. This suggests that this simple approach is problematic, and should be ruled out, or that we should settle for one representation only. We chose the latter route, as we see no simple characterization of those combination methodologies that guarantee that equivalent results are obtained.

There seems to be little epistemological justification for choosing to compromise on a full graph, as most/all learning algorithms are unable to differentiate between two graphs belonging to the same equivalence class, and compromising on a DAG therefore seems to involve compromising on too much. Therefore, we choose to have agents compromise on a pattern or completed pattern. Unfortunately, we are unaware of any simple characterization of graphs that are patterns, so we choose completed patterns out of necessity. In summary, we shall aim for having the agents compromise on the completed pattern $\mathcal{G}_* \in \mathfrak{C}^{cp}$ of $\mathcal{B}_*$.

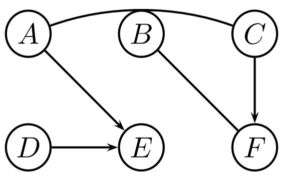
To establish whether a graph is a good compromise for the agents, we need a measure for how well such graphs match each of $\mathcal{B}_1$ to $\mathcal{B}_k$. Furthermore, as we plan to build this compromise gradually, we wish for this measure to be relative to an already agreed *partial compromise*. The need for this should be clear from the following example.

Example 2 (The Need for Partial Compromises): *Consider again the setting from Example 1, and refer to the completed patterns of $\mathcal{B}_1$ to $\mathcal{B}_3$ in Figures 7(a) to 7(c). It may be the case that agents have already agreed upon the connections in the graph in Figure 7(d), and now Agent 2 is asked to evaluate the compromise where a link is added between A and C. Obviously, this addition is consistent with Agent 2's beliefs, and thus must be valued highly. Consider then another situation, where the agents have agreed upon the connections in Figure 7(e), and Agent 2 is again asked to evaluate the addition of a link between A and C. Now the addition is not as important, since the connection between A and C would have to be directed into C (to avoid a directed cycle), and thus create a v-structure (A, C, E) not present in $\mathcal{B}_2$. Moreover, A and C are already connected (albeit through D and E) and the relationship between the two might be sufficiently represented by this connection.*

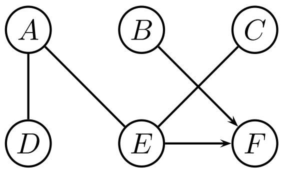
In general, we cannot assume that a partially specified graph is suitable as representation of a partial compromise, as this might include agreements



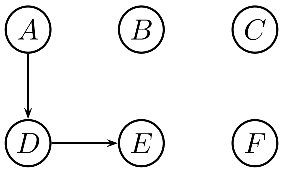
(a) *Completed pattern of* $\mathcal{B}_1$



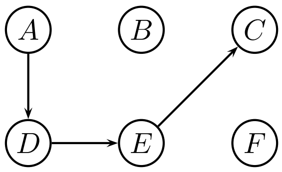
(b) *Completed pattern of* $\mathcal{B}_2$



(c) *Completed pattern of* $\mathcal{B}_3$



(d) *Partial compromise*



(e) *Partial compromise*

Figure 7: *Three completed patterns and two partial compromises.*

on what should *not* be part of the final compromise. Therefore, we take a partial compromise $\mathcal{P} \equiv (\boldsymbol{P}_+, \boldsymbol{P}_-)$ to be two sets of sentences in some language, where $\boldsymbol{P}_+$ describe aspects that should be true of the final compromise, and $\boldsymbol{P}_-$ describe aspects that cannot be true.

For any three partial compromises $\mathcal{P}$, $\mathcal{P}^a$, and $\mathcal{P}^b$, where $\boldsymbol{P}_+ \subseteq \boldsymbol{P}^a_+$, $\boldsymbol{P}_- \subseteq \boldsymbol{P}^a_-$, $\boldsymbol{P}_+ \subseteq \boldsymbol{P}^b_+$ and $\boldsymbol{P}_- \subseteq \boldsymbol{P}^b_-$, we assume that each agent i can compute its *compromise scores* $s_i(\mathcal{P}, \mathcal{P}^a)$ and $s_i(\mathcal{P}, \mathcal{P}^b)$ such that $s_i(\mathcal{P}, \mathcal{P}^a) > s_i(\mathcal{P}, \mathcal{P}^b)$ iff $\mathcal{P}^a$ describes $\mathcal{B}_i$ better than $\mathcal{P}^b$, given that $\mathcal{P}$ has already been accepted as being descriptive of $\mathcal{B}_i$. We will assume s_i to be additive, i.e. for any three partial compromises $\mathcal{P}^0$, $\mathcal{P}^1$, and $\mathcal{P}^2$, where $\boldsymbol{P}^0_+ \subseteq \boldsymbol{P}^1_+ \subseteq \boldsymbol{P}^2_+$ and $\boldsymbol{P}^0_- \subseteq \boldsymbol{P}^1_- \subseteq \boldsymbol{P}^2_-$, it is the case that $s_i(\mathcal{P}^0, \mathcal{P}^2) = s_i(\mathcal{P}^0, \mathcal{P}^1) + s_i(\mathcal{P}^1, \mathcal{P}^2)$.

Example 3 (Compromise Scoring Functions): *Consider the partial compromises* $\mathcal{P}^a \equiv (\boldsymbol{P}^a_+, \boldsymbol{P}^a_- = \varnothing)$ *and* $\mathcal{P}^b \equiv (\boldsymbol{P}^b_+, \boldsymbol{P}^b_- = \varnothing)$ *where the sentences in* $\boldsymbol{P}^a_+$ *and* $\boldsymbol{P}^b_+$ *represent the graphs in Figures 7(d) and 7(e), respectively. A simple example of* $s_2(\mathcal{P}^a, \mathcal{P}^b)$ *could be the number of features described in* $\boldsymbol{P}^b_+ \setminus \boldsymbol{P}^a_+$*, which are consistent with* $\mathcal{B}_2$*, minus those that are not. Specifically,* $s_2(\mathcal{P}^a, \mathcal{P}^b)$ *would equal* -1.

A more complex score could weigh each of these described features according to the empirical evidence Agent 2 has in favor of or against them: We may have that Agent 2 is employing a sensor for measuring the E *variable, and that this sensor is known to be of poor quality. Therefore,* $s_2(\mathcal{P}^a, \mathcal{P}^b)$ *would be equal to* -0.3*, as an indication that the addition of the arc is against Agent 2's observations, but that those observations could easily be flawed.*

Yet another score could take into account the ramifications of the sentences in $\boldsymbol{P}^b_+ \setminus \boldsymbol{P}^a_+$*. For instance, the addition of the arc from* E *to* C *might in itself not be much at odds with Agent 2's observations, but the addition forces it to either give up the idea of a link between* A *and* C*, or accept the existence of v-structure* (A, C, E)*, both of which it may have strong evidence for/against. Therefore, it has* $s_2(\mathcal{P}^a, \mathcal{P}^b) = -10$*. Of course, the challenge is for Agent 2 to actually be able to survey the impact of the sentences in* $\boldsymbol{P}^b_+ \setminus \boldsymbol{P}^a_+$.

Had $\boldsymbol{P}^a_-$ *not been empty, but rather included an agreement that there was not to be any connection between* A *and* C*, then the last given example of a compromise score, would not have been so low, as the dreaded consequence of the addition of the arc from* E *to* C *would already be a consequence of* $\mathcal{P}^a$*. On the other hand, had this agreement been a part of* $\boldsymbol{P}^b_- \setminus \boldsymbol{P}^a_-$*, then the first two examples of scores would have been lower:* -2 *for the first, and something less than* -0.3*, depending on Agent 2's evidence, for the second.*

Notice, that with this open definition, we do not attempt to define what it means to be a "better description", since we believe that this issue can be dependent on the actual setting in which the framework is to be used, as stated in Section 1.

In addition to the compromise score, we also assume that the agents know the *combination function* $c : \mathbb{R}^k \rightarrow \mathbb{R}$, indicating how much trust should be put into the individual agents' models. Differences in trust can be justified by differences in experiences, sensor accuracies, etc. Formally, we define c as follows: Let $\mathcal{P}$, $\mathcal{P}^a$, and $\mathcal{P}^b$ be partial compromises. c is the combination function for agents i to k, if

$$c(s_1(\mathcal{P}, \mathcal{P}^a), \ldots, s_k(\mathcal{P}, \mathcal{P}^a)) > c(s_1(\mathcal{P}, \mathcal{P}^b), \ldots, s_k(\mathcal{P}, \mathcal{P}^b)),$$

whenever $\mathcal{P}^a$ is a better compromise than $\mathcal{P}^b$ for the group of agents 1 to k, given that they have already agreed on $\mathcal{P}$. An obvious choice for c would be a linear combination of its inputs, and in fact we shall assume that this is the case in this text. We refer to $c(s_1(\mathcal{P}, \mathcal{P}^a), \ldots, s_k(\mathcal{P}, \mathcal{P}^a))$ as the *joint compromise score* of $\mathcal{P}^a$ given $\mathcal{P}$.

With this notation in place, we can restate the task more formally, as that of finding a partial compromise $\mathcal{P}$, which uniquely identifies some graph $\mathcal{G}_* \in \mathfrak{C}^{cp}$, such that

$$c(s_1((\varnothing, \varnothing), \mathcal{P}), \ldots, s_k((\varnothing, \varnothing), \mathcal{P})) \geq c(s_1((\varnothing, \varnothing), \mathcal{P}'), \ldots, s_k((\varnothing, \varnothing), \mathcal{P}')),$$

for all other partial compromises $\mathcal{P}'$, which uniquely identifies a graph $\mathcal{G}' \in \mathfrak{C}$.

As presented here, it is clear that the problem is not of a simple binary nature, as we are not trying to establish whether some proposition is true or not, and that we are furthermore dealing with a setting in which more than two agents may interact. Consequently, we cannot utilize the vast literature on dialectic proof theories directly. Rather, the problem we are trying to solve is a distributed maximization over a super exponential hypothesis space ($\mathfrak{C}$). Furthermore, as the worth of (partial) compromises are specified in relation to already agreed upon compromises, the problem is of a highly dynamic nature.

Our solution to the problems is divided into three parts. First, we create a finite language with which graphs and some essential properties of these can be expressed; second and most importantly, we construct an argumentation system with which the agents can reason about consequences of committing to partial compromises; and thirdly, we create an agora in which the agents can reach compromise graphs in an anytime fashion.

4 Encoding Graphs

For the agents to compromise on $\mathcal{G}_*$, a formal language $\boldsymbol{L}$ for expressing graphs and properties of graphs must be defined. For efficiency reasons we aim to make this language as simple as possible, while ensuring that it is still sufficiently powerful to describe any graph and its membership status of $\mathfrak{C}^{cp}$. By simple, we mean preferably finite and as small as possible.

First of all, we introduce a simple language $\boldsymbol{L}^g$ for encoding of graphs only:

Definition 2 (Simple Graph Language)**.** *The language $\boldsymbol{L}^g$ is the set such that `Arc(X,Y)`, `Arc(Y,X)`, `Link(X,Y)`, and `NonAdjacent(X,Y)` are members of $\boldsymbol{L}^g$ iff X and Y ($X \rightsquigarrow Y$) are distinct variables.*

A *graph knowledge base* we define to be a set $\boldsymbol{\Sigma}^g \subseteq \boldsymbol{L}^g$. Further:

Definition 3 (Consistent and Closed Graph Knowledgebases)**.** *Given a graph knowledge base $\boldsymbol{\Sigma}^g$, if it holds that for all pairs of variables X and Y, where $X \rightsquigarrow Y$, a maximum of one of `Arc(X,Y)`, `Arc(Y,X)`, `Link(X,Y)`, and `NonAdjacent(X,Y)` is in $\boldsymbol{\Sigma}^g$, then we call $\boldsymbol{\Sigma}^g$ a* consistent graph knowledgebase (CGK)*.*

Furthermore, if it holds that for any two variables X and Y, where $X \rightsquigarrow Y$, exactly one of `Arc(X,Y)`, `Arc(Y,X)`, `Link(X,Y)`, and `NonAdjacent(X,Y)` is in $\boldsymbol{\Sigma}^g$, then we call $\boldsymbol{\Sigma}^g$ a closed graph knowledgebase (CLGK)*.*

The *graph encoded by* a CGK $\boldsymbol{\Sigma}^g$ is the graph $\mathcal{G}[\boldsymbol{\Sigma}^g]$ resulting from starting with the graph with no edges, and then for any two nodes X and Y ($X \rightsquigarrow Y$) adding an arc from X to Y if `Arc(X,Y)` is in $\boldsymbol{\Sigma}^g$, an arc from Y to X if `Arc(Y,X)` is in $\boldsymbol{\Sigma}^g$, or an undirected edge if `Link(X,Y)` is in $\boldsymbol{\Sigma}^g$. It is easy to see that graph encoded by a CGK is well-defined. Furthermore, given a graph $\mathcal{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ there exists a unique CLGK $\boldsymbol{\Sigma}^g[\mathcal{G}]$, for which $\mathcal{G}$ is the encoded graph. Clearly, $\boldsymbol{\Sigma}^g[\mathcal{G}]$ can be constructed in time $O(|\boldsymbol{V}|^2)$.

Example 4 (Graph Encoding): *Consider the graph knowledgebase $\boldsymbol{\Sigma}^g_e$ comprised of `Arc(A,D)`, `Arc(D,E)`, and `Arc(E,C)`. $\boldsymbol{\Sigma}^g_e$ is a CGK, albeit not a closed one, as the connections among several pairs of variables, such as A and E are not specified. Assuming that $\boldsymbol{L}^g$ is built from variables $\{A, B, C, D, E, F\}$, $\mathcal{G}[\boldsymbol{\Sigma}^g_e]$ corresponds to the graph in Figure 7(e). If furthermore all of*

`NonAdjacent(A,B)`, `NonAdjacent(A,C)`, `NonAdjacent(A,E)`,
`NonAdjacent(A,F)`, `NonAdjacent(B,C)`, `NonAdjacent(B,D)`,
`NonAdjacent(B,E)`, `NonAdjacent(B,F)`, `NonAdjacent(C,D)`,
`NonAdjacent(C,F)`, `NonAdjacent(D,F)`, *and* `NonAdjacent(E,F)`

were added to $\mathbf{\Sigma}^g_e$, $\mathbf{\Sigma}^g_e$ would be the encoded version of the graph in Figure 7(e), and would therefore be a CLGK. If any further sentence was added to $\mathbf{\Sigma}^g_e$, the set would cease to be a CGK.

We thus have that any graph can be efficiently encoded by a unique CLGK, and Definition 3 allows us to distinguish the graph knowledge bases, which can be interpreted as graphs, from those that cannot. Next, we extend $\boldsymbol{L}^g$ into a language powerful enough for building a reasoning engine about graphs and their membership status of $\mathfrak{C}^{cp}$ on top:

Definition 4 (Graph Language)**.** *The graph language $\boldsymbol{L}$ is the set consisting of all sentences in $\boldsymbol{L}^g$ and*

- *`ArcNotAllowed(X,Y)`,*
- *`DirectedPath(X,Y)`,*
- *`UndirectedPath(X,Y)`,*
- *`UndirectedPath(X,Y)Excluding(Z,W)`,*
- *¬`DirectedPath(X,Y)`,*
- *¬`UndirectedPath(X,Y)`, and*
- *¬`UndirectedPath(X,Y)Excluding(Z,W)`,*

for any choice of distinct variables X, Y, Z, and W[2] *($Z \rightsquigarrow W$). Sentences like `DirectedPath(X,Y)`, `UndirectedPath(X,Y)`, and `Undirected-Path(X,Y)Excluding(Z,W)` will be referred to as* path *sentences, and sentences like ¬`DirectedPath(X,Y)`, ¬`UndirectedPath(X,Y)`, and ¬`Undirected-Path(X,Y)Excluding(Z,W)` will be referred to as* negative path *sentences.*

Intuitively, the sentences just introduced are supposed to be used as descriptors of attributes of the graphs encoded by CGKs: `ArcNotAllowed-(X,Y)` states that an arc from X to Y would not be strongly protected; `DirectedPath(X,Y)` states that there is a directed path from X to Y; `UndirectedPath(X,Y)` states that there is an undirected path between X and Y; `UndirectedPath(X,Y)Excluding(Z,W)` states that there is an undirected path not comprising Z nor W between X and Y; ¬`DirectedPath-(X,Y)` states that there is no directed path from X and Y; ¬`Undirected-Path(X,Y)` states that there is no undirected path between X and Y; and

[2]Throughout the text we assume that the implicit set of variables has at least five members. This assumption can be lifted, albeit with a more complex notation to follow.

$\neg$`UndirectedPath(X,Y)Excluding(Z,W)` states that there is no undirected path between X and Y, or that any such path necessarily contain either Z or W.

As $\boldsymbol{L}^g$ is a subset of $\boldsymbol{L}$, it follows that a graph knowledge base is a set of sentences in $\boldsymbol{L}$ as well, and in particular that Definition 3 still makes sense. Given a set $\boldsymbol{\Sigma}$ of sentences of $\boldsymbol{L}$, we denote by $\boldsymbol{\Sigma}^g$ the set $\boldsymbol{\Sigma} \cap \boldsymbol{L}^g$.

5 Graph Argumentation System

Building on the language $\boldsymbol{L}$ introduced above, we define an argumentation system for distinguishing completed patterns that could be compromises for the agents. The system that we construct enjoys the properties that a graph is a member of $\mathfrak{C}$ iff there is a preferred extension of the system, such that the extension encodes this graph.

Definition 5 (Graph Argumentation System)**.** *The* graph argumentation system $\mathcal{A}^g$ *is the tuple* $(\boldsymbol{L}, \triangleright^g \subseteq (2^{\boldsymbol{L}} \times \boldsymbol{L}))$*, where* $\triangleright^g$ *is defined as follows (*`[A,B]` *is short-hand for any one of* `(A,B)` *and* `(B,A)`*):*

1. *`Arc(X,Y)`$\triangleright^g$ `Arc(Y,X)`*
2. *`Arc(X,Y)`$\triangleright^g$ `Link[X,Y]`*
3. *`Arc(X,Y)`$\triangleright^g$ `NonAdjacent[X,Y]`*
4. *`Link(X,Y)`$\triangleright^g$ `Arc[X,Y]`*
5. *`Link(X,Y)`$\triangleright^g$ `NonAdjacent[X,Y]`*
6. *`NonAdjacent(X,Y)`$\triangleright^g$ `Arc[X,Y]`*
7. *`NonAdjacent(X,Y)`$\triangleright^g$ `Link[X,Y]`*

8. *$\neg$`DirectedPath(X,Y)`$\triangleright^g$ `DirectedPath(X,Y)`*
9. *$\neg$`UndirectedPath(X,Y)`$\triangleright^g$ `UndirectedPath(X,Y)`*
10. *$\neg$`UndirectedPath(X,Y)Excluding(Z,W)`$\triangleright^g$ `UndirectedPath(X,Y)Excluding-(Z,W)`*
11. *`Arc(X,Y)`$\triangleright^g$ $\neg$`DirectedPath(X,Y)`*
12. *`Link(X,Y)`$\triangleright^g$ $\neg$`UndirectedPath[X,Y]`*

13. *Link(X,Y)*$\rhd^g$ $\neg$*UndirectedPath[X,Y]Excluding(Z,W)*
14. $\{$*DirectedPath(X,Y), DirectedPath(Y,Z)*$\}\rhd^g$ $\neg$*DirectedPath(X,Z)*
15. $\{$*DirectedPath(X,Y), UndirectedPath[Y,Z]*$\}\rhd^g$ $\neg$*DirectedPath(X,Z)*
16. $\{$*UndirectedPath[X,Y], DirectedPath(Y,Z)*$\}\rhd^g$ $\neg$*DirectedPath(X,Z)*
17. $\{$*UndirectedPath[X,Y], UndirectedPath[Y,Z]*$\}\rhd^g$ $\neg$*UndirectedPath-[X,Z]*
18. $\{$*UndirectedPath[X,Y]Excluding(Z,W), UndirectedPath[Y,U]Excluding-(Z,W)*$\}\rhd^g$ $\neg$*UndirectedPath[X,U]Excluding(Z,W)*

19. *DirectedPath(X,Y)*$\rhd^g$ *Arc(Y,X)*
20. *DirectedPath(X,Y)*$\rhd^g$ *Link[X,Y]*
21. *UndirectedPath[X,Y]*$\rhd^g$ *Arc(X,Y)*

22. $\{$*UndirectedPath[X,Y]Excluding(W,Z), Link[X,W], Link[Y,Z], NonAdjacent[X,Z], NonAdjacent[Y,W]*$\}\rhd^g$ *Link[W,Z]*

23. $\{$*Arc(X,Y), NonAdjacent[X,Z]*$\}\rhd^g$ *Link[Y,Z]*

24. *ArcNotAllowed(X,Y)*$\rhd^g$ *Arc(X,Y)*
25. $\{$*Arc(Z,X), NonAdjacent[Z,Y]*$\}\rhd^g$ *ArcNotAllowed(X,Y)*
26. $\{$*Arc(Z,Y), NonAdjacent[Z,X]*$\}\rhd^g$ *ArcNotAllowed(X,Y)*
27. $\{$*Arc(X,Z), Arc(Z,Y)*$\}\rhd^g$ *ArcNotAllowed(X,Y)*
28. $\{$*Link[X,Z], Arc(Z,Y), Link[X,W], Arc(W,Y), NonAdjacent[Z,W]*$\}\rhd^g$ *ArcNotAllowed(X,Y)*

for all choices of distinct variables X, Y, Z, W, and U where the sentences obtained are in $\boldsymbol{L}$.

Loosely speaking, if $\mathbf{\Sigma}$ is a conflict free set wrt. $\triangleright^g$, then Bullets 1–7 ensure that $\mathbf{\Sigma}^g$ is a CGK; Bullets 8–18 make sure that the path and negative path sentences in $\mathbf{\Sigma} \setminus \mathbf{\Sigma}^g$ are correct wrt. the graph that is encoded by $\mathbf{\Sigma}^g$; Bullets 19–21 ensure that $\mathbf{\Sigma}^g$ encodes a chain graph; Bullet 22 ensures that the graph encoded by $\mathbf{\Sigma}^g$ has decomposable chain components; Bullet 23 ensures that that graph also respects Bullet 3 of Theorem 1; and Bullets 24–28 guarantee that all arcs in the graph are strongly protected.

Example 5 (Argumentative Reasoning About Graphs): *We return to the CGK $\mathbf{\Sigma}_e^g$ presented in Example 4 joined with sentences `DirectedPath(A,D)`, `DirectedPath(D,E)`, `DirectedPath(E,C)`, `DirectedPath(D,C)`, and `DirectedPath(A,C)` to constitute the set of sentences $\mathbf{\Sigma}_e$. Clearly, $\mathbf{\Sigma}_e$ is conflict-free. From Bullet 19 we can conclude that `Arc(C,A)` cannot be added to $\mathbf{\Sigma}_e$ without destroying its property of being conflict-free wrt. $\mathcal{A}^g$. Bullet 20 yields the same result for `Link(A,C)`, and the only options for putting a description of the connection between A and C into $\mathbf{\Sigma}_e$ are thus `NonAdjacent(A,C)` and `Arc(A,C)`. The argumentation system thus allows for reasoning about such consequences that were treated informally in Examples 2 and 3.*

In what follows we provide a formal treatment of the above outlined intuitions. More specifically we prove two important results, namely that any preferred extension of $\mathcal{A}^g$ is a member of $\mathfrak{C}^{cp}$, and that any member of $\mathfrak{C}^{cp}$ has a corresponding stable extension of $\mathcal{A}^g$. These results are important since they guarantee that, if agents seek compromises under the restrictions specified by $\mathcal{A}^g$, they can be sure that their result is a completed pattern and that they are not restricted from agreeing on any model a priori by the relations of $\mathcal{A}^g$.

Lemma 6. *Let $\mathbf{\Sigma}$ be conflict free wrt. $\mathcal{A}^g$. Then $\mathbf{\Sigma}^g$ is a CGK.*

Proof. Obvious, given Bullets 1– 7. ☐

Lemma 7. *Let $\mathbf{\Sigma}$ be a preferred extension of $\mathcal{A}^g$. Then $\mathbf{\Sigma}^g$ is a CLGK.*

Proof. We need to prove that if X and Y ($X \rightsquigarrow Y$) are two nodes that are non-adjacent in $\mathcal{G}[\mathbf{\Sigma}^g]$ then `NonAdjacent(X,Y)` is in $\mathbf{\Sigma}$. We prove this by contradiction: We assume that `NonAdjacent(X,Y)` is not in $\mathbf{\Sigma}$ and then prove that $\mathbf{\Sigma}^* = \mathbf{\Sigma} \cup$`NonAdjacent(X,Y)` is an admissible set, which contradicts that $\mathbf{\Sigma}$ is a preferred extension.

First we prove that $\mathbf{\Sigma}^*$ is conflict-free: Since X and Y are non-adjacent in $\mathcal{G}[\mathbf{\Sigma}^g]$ it follows that none of `Arc(X,Y)`, `Arc(Y,X)`, and `Link(X,Y)` are in

$\mathbf{\Sigma}$, and therefore that $\mathbf{\Sigma}$ cannot attack `NonAdjacent(X,Y)`. Thus, if there is a set $\boldsymbol{S} \subseteq \mathbf{\Sigma}^*$ and an argument $A \in \mathbf{\Sigma}^*$, such that $\boldsymbol{S} \rhd^g A$, then A cannot be `NonAdjacent(X,Y)` and hence must be in $\mathbf{\Sigma}$. As $\mathbf{\Sigma}$ is admissible, it follows that $\mathbf{\Sigma}$ attacks at least one argument B in $\boldsymbol{S}$, and as $\mathbf{\Sigma}$ does not attack `NonAdjacent(X,Y)`, B must be a member of $\boldsymbol{S}\backslash$`NonAdjacent(X,Y)`. But that means that $\mathbf{\Sigma}$ attacks $\mathbf{\Sigma}$, meaning that it is not a conflict-free set, contradicting the assumption that $\mathbf{\Sigma}$ is a preferred extension. So we conclude that $\mathbf{\Sigma}^*$ must be conflict-free.

It is easy to see that $\mathbf{\Sigma}^*$ attacks all sets of arguments that attack $\mathbf{\Sigma}^*$: By assumption, $\mathbf{\Sigma}$ attacks all sets of arguments that attack an argument in $\mathbf{\Sigma}$, and by Bullets 1– 7, `NonAdjacent(X,Y)` attacks all sets of arguments that attack itself. Consequently, $\mathbf{\Sigma}^* = \mathbf{\Sigma} \cup$`NonAdjacent(X,Y)` attacks all arguments that attack some argument in itself, and the lemma follows. ☐

Lemma 8. *Let $\mathbf{\Sigma}$ be a preferred extension, and let A an argument defended by $\mathbf{\Sigma}$. Then A is in $\mathbf{\Sigma}$.*

Proof. We show that $\mathbf{\Sigma}^* \equiv \mathbf{\Sigma} \cup A$ is an admissible set. As $\mathbf{\Sigma}$ is a preferred extension, it is obvious that it defend all arguments in itself, and by assumption it also defends A. We thus only have to show that $\mathbf{\Sigma}^*$ is conflict-free. Therefore, let $\boldsymbol{S} \subseteq \mathbf{\Sigma}^*$ and $B \in \mathbf{\Sigma}^*$ such that $\boldsymbol{S} \rhd B$. As $\mathbf{\Sigma}$ defends both itself and A, it follows that $\mathbf{\Sigma}$ attacks an argument C in $\boldsymbol{S}$. However, as $\mathbf{\Sigma}$ is conflict-free this C has to be A. But then $\mathbf{\Sigma}$ attacks A, and since it defends A, also itself, which contradicts that it is conflict-free. ☐

Lemma 9. *Let $\mathbf{\Sigma}$ be a preferred extension of $\mathcal{A}^g$ and X, Y, Z, and W ($Z \rightsquigarrow W$) be variables. Then*

1. *if there is a directed path from X to Y in $\mathcal{G}[\mathbf{\Sigma}^g]$, then `DirectedPath-(X,Y)` is in $\mathbf{\Sigma}$,*

2. *if there is an undirected path from X to Y in $\mathcal{G}[\mathbf{\Sigma}^g]$, then `Undirected-Path(X,Y)` is in $\mathbf{\Sigma}$, and*

3. *if there is an undirected path from X to Y in $\mathcal{G}[\mathbf{\Sigma}^g]$ not comprising any of the variables Z and W, then `UndirectedPath(X,Y)Excluding-(Z,W)` is in $\mathbf{\Sigma}$.*

Proof. Induction on the length l of the path between X and Y. We first consider the base case where l is 1. If the path is a directed path, this means that there is an arc from X to Y and consequently that `Arc(X,Y)` is in $\mathbf{\Sigma}$. By Bullet 11 this means that $\mathbf{\Sigma}$ attacks ¬`DirectedPath(X,Y)`, which is the only argument that attacks `DirectedPath(X,Y)`. Lemma 8 then guarantees that `DirectedPath(X,Y)` is in $\mathbf{\Sigma}$. Similar arguments establish the base cases for 2 and 3.

For the induction step, assume that the result is valid for all paths of length $l \leq n$ and consider a path $\pi = (X = X_1, U = X_2, \ldots, X_{n+2} = Y)$ of length $n+1$. If π is an undirected path, then we know from the induction hypothesis that both `UndirectedPath(X,U)` and `UndirectedPath(U,Y)` must be in $\mathbf{\Sigma}$. The two sentences collectively attack ¬`UndirectedPath(X,Y)`, which is the only argument attacking `UndirectedPath(X,Y)`. Since $\mathbf{\Sigma}$ is a preferred extension, Lemma 8 then gives that `UndirectedPath(X,Y)` is a member of $\mathbf{\Sigma}$. The proof for `UndirectedPath(X,Y)Excluding(Z,W)` is the same.

Assume next that π is a directed path. Then either there is an arc from X to U or a link between them. We consider the two cases in turn.

In the first case, an argument similar to the one used for the base case gives us that `DirectedPath(X,U)` must be in $\mathbf{\Sigma}$. Next, if $(U = X_2, \ldots, X_{n+2} = Y)$ is a directed path, then induction hypothesis gives us that `Directed-Path(U,Y)` is in $\mathbf{\Sigma}$. If $(U = X_2, \ldots, X_{n+1} = Y)$ is undirected, we get that `UndirectedPath(U,Y)` is in $\mathbf{\Sigma}$. Either way, the two sentences obtained collectively attack ¬`DirectedPath(X,Y)`, which is the only argument attacking `DirectedPath(X,Y)`. Since $\mathbf{\Sigma}$ is a preferred extension, Lemma 8 then states that `DirectedPath(X,Y)` must be a member of $\mathbf{\Sigma}$.

If there is a link between X and U, there must be a directed path from U to Y, for π to be a directed path. As before the presence of `Link(X,U)` or `Link(U,X)` yields that `UndirectedPath(X,U)` is in $\mathbf{\Sigma}$, and the induction hypothesis that `DirectedPath(U,Y)` is in $\mathbf{\Sigma}$. Together the two attack ¬`DirectedPath(X,Y)`, which is the only argument that attacks `Directed-Path(X,Y)`. Once again Lemma 8 guarantees that `DirectedPath(X,Y)` is in $\mathbf{\Sigma}$. □

The converse of Lemma 9 is not true, which can be seen from the following example:

Example 6: *Let $\mathbf{\Sigma}^g$ consist of `NonAdjacent(A,B)`, `NonAdjacent(B,C)`, and `NonAdjacent(A,C)`, thus encoding the empty graph over the set of variables $\boldsymbol{V}_e = \{A, B, C\}$. Furthermore, let $\mathbf{\Sigma} \setminus \mathbf{\Sigma}^g$ be*

$$\begin{aligned} &\{\mathtt{ArcNotAllowed(X,Y)} \ : \ X, Y \in \boldsymbol{V}_e\} \\ \cup&\{\mathtt{UndirectedPath(X,Y)} \ : \ X, Y \in \boldsymbol{V}_e\} \\ \cup&\{\mathtt{DirectedPath(X,Y)} \ : \ X, Y \in \boldsymbol{V}_e\}. \end{aligned}$$

Then $\mathbf{\Sigma}$ is conflict free and attacks all other sentences in $\boldsymbol{L}$, meaning that it is stable and hence a preferred extension, yet clearly $\mathcal{G}[\mathbf{\Sigma}^g]$ does not reflect the meaning of the path sentences in $\mathbf{\Sigma}$.

Theorem 10. *Let $\mathbf{\Sigma}$ be a preferred extension of $\mathcal{A}^g$. Then $\mathcal{G}[\mathbf{\Sigma}^g]$ is in $\mathfrak{C}^{cp}$.*

Proof. Assume not. This means that $\mathcal{G}[\mathbf{\Sigma}^g]$ fails to satisfy one of the bullets of Theorem 1. We prove that this cannot be the case, one bullet at the time:

Assume $\mathcal{G}[\mathbf{\Sigma}^g]$ does not satisfy Bullet 1, namely that it is not a chain graph. Then there is a directed cycle $(X, \ldots, Y, X)$ in $\mathcal{G}[\mathbf{\Sigma}^g]$ containing at least one arc. Without loss of generality, assume this arc to be going from Y to X, and consequently that `Arc(Y,X)` is in $\mathbf{\Sigma}$. The rest of the cycle then constitutes either a directed path from X to Y, or an undirected path between them. Either way, Lemma 9 allow us to conclude that one of `DirectedPath(X,Y)` and `UndirectedPath[X,Y]` must be in $\mathbf{\Sigma}$. But that is impossible, since either one of them attacks `Arc(Y,X)` and $\mathbf{\Sigma}$ is conflict-free.

Assume then that $\mathcal{G}[\mathbf{\Sigma}^g]$ does not satisfy Bullet 2, namely that it contains a chain component that is not decomposable. This implies that the there is a chordless cycle in $\mathcal{G}[\mathbf{\Sigma}^g]$. Let this cycle be $(X, Z, \ldots, W, Y, X)$, which implies that `Link[X,Z]`, `Link[W,Y]`, and `Link[Y,X]` are in $\mathbf{\Sigma}$. Furthermore, as the cycle is chordless, both X and W and Y and Z must be pairs of non-adjacent nodes in $\mathcal{G}[\mathbf{\Sigma}^g]$. By Lemma 7 `NonAdjacent[X,W]` and `NonAdjacent[Y,Z]` must then be in $\mathbf{\Sigma}$. Additionally, the path $(Z, \ldots, W)$ and Lemma 9 collectively allow us to conclude that `UndirectedPath[Z,W]Excluding(X,Y)` must be in $\mathbf{\Sigma}$ as well. However, `UndirectedPath[Z,W]Excluding(X,Y)` in conjunction with `Link[Z,X]`, `Link[W,Y]`, `NonAdjacent[X,W]`, and `NonAdjacent[Y,Z]` attacks `Link[X,Y]`, which is impossible since $\mathbf{\Sigma}$ is conflict-free.

Next, assume that $\mathcal{G}[\mathbf{\Sigma}^g]$ does not satisfy Bullet 3, which means that there are three variables X, Y, and Z, such that X is a parent of Y, X and Z are not adjacent, and Z is a neighbour of Y. According to Lemma 7, $\mathbf{\Sigma}$ would then contain all of `Arc(X,Y)`, `NonAdjacent[X,Z]`, and `Link[Y,Z]`. But the first two attack the last one, and since $\mathbf{\Sigma}$ is conflict-free, this is impossible.

Finally, assume that $\mathcal{G}[\mathbf{\Sigma}^g]$ does not satisfy Bullet 4, which means that there is some arc from a node X to a node Y that is not strongly protected in $\mathcal{G}[\mathbf{\Sigma}^g]$. Thus `Arc(X,Y)` is in $\mathbf{\Sigma}$, and as `Arc(X,Y)` is attacked by `ArcNotAllowed(X,Y)` and $\mathbf{\Sigma}$ is an admissible set, it follows that $\mathbf{\Sigma}$ attacks `ArcNotAllowed(X,Y)`. The only ways $\mathbf{\Sigma}$ can attack `ArcNotAllowed(X,Y)` are by containing either

- `Arc(Z,X)` and `NonAdjacent[Z,Y]` for some variable Z,

- `Arc(Z,Y)` and `NonAdjacent[Z,X]` for some variable Z,
- `Arc(X,Z)` and `Arc(Z,Y)` for some variable Z, or
- `Link[X,Z]`, `Link[X,W]`, `Arc(Z,Y)`, `Arc(W,Y)`, and `NonAdjacent(Z,W)` for some variables Z and W.

We deal only with the first case as the others are similar. As $\mathbf{\Sigma}$ contains `Arc(Z,X)` and `NonAdjacent[Z,Y]` and $\mathbf{\Sigma}^g$ is a CGK, it follows that there is an arc from Z to X in $\mathcal{G}[\mathbf{\Sigma}^g]$ and that Z and Y are not connected by a link nor an arc. But then the arc from X to Y is protected, which yields a contradiction with the assumption. As the other three cases yield similar contradictions, the assumption must be false, and the theorem follows. □

Theorem 11. *If $\mathcal{G}$ is in $\mathfrak{C}^{cp}$, then there is a stable extension $\mathbf{\Sigma}$ of $\mathcal{A}^g$, such that $\mathcal{G}[\mathbf{\Sigma}^g] = \mathcal{G}$.*

Proof. We prove this by construction of a stable extension $\mathbf{\Sigma}$ such that $\mathcal{G}[\mathbf{\Sigma}^g] = \mathcal{G}$. First let $\mathbf{\Sigma}^g$ be $\mathbf{\Sigma}^g[\mathcal{G}]$, which ensures that $\mathcal{G}[\mathbf{\Sigma}^g] = \mathcal{G}$, no matter what sentences are in $\mathbf{\Sigma} \setminus \mathbf{\Sigma}^g$.

Next, for any four variables X, Y, Z, and W, let $\mathbf{\Sigma} \setminus \mathbf{\Sigma}^g$ contain

- `ArcNotAllowed(X,Y)` iff there is no arc between X and Y and such an arc would not be strongly protected,
- `DirectedPath(X,Y)` iff there is a directed path from X to Y, and ¬`DirectedPath(X,Y)` otherwise,
- `UndirectedPath(X,Y)` iff there is an undirected path between X and Y, and ¬`UndirectedPath(X,Y)` otherwise, and
- `UndirectedPath(X,Y)Excluding(Z,W)` iff there is an undirected path not comprising Z nor W between X and Y, and ¬`UndirectedPath-(X,Y)Excluding(Z,W)` otherwise.

Then $\mathbf{\Sigma}$ is the sought stable extension. To prove it, we show (i) that $\mathbf{\Sigma}$ is conflict-free and (ii) that $\mathbf{\Sigma}$ attacks all sentences in $\boldsymbol{L} \setminus \mathbf{\Sigma}$.

i) We consider each bullet of the definition of $\triangleright^g$ in Definition 5.

Bullets 1 to 7: As $\mathbf{\Sigma}^g$ was defined to be $\mathbf{\Sigma}^g[\mathcal{G}]$, which is a CLGK, there can be no two variables X and Y ($X \rightsquigarrow Y$), for which more than one of `Arc(X,Y)`, `Arc(Y,X)`, `Link(X,Y)`, and `Link(X,Y)` are in $\mathbf{\Sigma}$. Consequently, these bullets do not give rise to any conflicts.

Bullets 8 to 13: These bullets cannot give rise to any conflict. The first because by definition of $\mathbf{\Sigma}$, for each pair of variables X and Y, only one of `DirectedPath(X,Y)` and ¬`DirectedPath(X,Y)` can be in $\mathbf{\Sigma}$. The same reasoning prevents the next two bullets from giving rise to a conflict. Bullet 11 cannot give rise to a conflict as the presence of `Arc(X,Y)` in $\mathbf{\Sigma}$ means that there is an arc from X to Y in $\mathcal{G}$ and thus a directed path from X to Y, which by definition of $\mathbf{\Sigma}$ means that it does not contain ¬`DirectedPath(X,Y)`. The same reasoning goes for the two next bullets.

Bullets 14 to 18 cannot give rise to conflicts. If Bullet 14 was the reason for a conflict, then both `DirectedPath(X,Y)` and `DirectedPath(Y,Z)` would have to be in $\mathbf{\Sigma}$, which by definition of $\mathbf{\Sigma}$ would mean that there is a directed path $(X, \ldots, Y)$ from X to Y and a directed path $(Y, \ldots, Z)$ from Y to Z in $\mathcal{G}$. But then there is also a directed path $(X, \ldots, Y, \ldots, Z)$ from X to Z meaning that ¬`DirectedPath(X,Z)` by definition is not in $\mathbf{\Sigma}$. Similar arguments show that the other bullets cannot give rise to conflicts.

Bullet 19 cannot give rise to a conflict as that would mean that both `DirectedPath(X,Y)` and `Arc(Y,X)` would be in $\mathbf{\Sigma}$, and consequently that there would be a directed path from X to Y and an arc from Y to X in $\mathcal{G}$, which would mean that $\mathcal{G}$ is not a chain graph, and by Theorem 1 not a completed pattern. The same goes for Bullets 20 to 21, and a similar argument, substituting the chain graph requirement of Theorem 1 with the decomposability one, yields that Bullet 22 cannot be cause of conflicts either.

Bullet 23 cannot give rise to a conflict, as if it did that would mean that all of `Arc(X,Y)`, `NonAdjacent[X,Z]`, and `Link[Y,Z]` would be in $\mathbf{\Sigma}^g$, which in turn would mean that X would be a parent of Y in $\mathcal{G}$, that X and Z would not be adjacent, and that Y and Z would be connected with an undirected link. This would violate Bullet 3 of Theorem 1, meaning that $\mathcal{G}$ is not a completed pattern, which is a contradiction.

Bullet 24 can clearly not give rise to any conflicts, as `ArcNotAllowed(X,Y)` are only in $\mathbf{\Sigma}$ for nodes not connected by an arc, ruling out that `Arc(X,Y)` is also in $\mathbf{\Sigma}$, which would be needed for a conflict.

Bullets 25 to 28 cannot give rise to conflicts due to similar reasons, wherefore only Bullet 25 is treated here. If this bullet caused a conflict, it means that `Arc(Z,X)`, `NonAdjacent[Z,Y]`, and `ArcNotAllowed(X,Y)` are all in $\mathbf{\Sigma}$, implying that Z is a parent of X and not adjacent to Y in $\mathcal{G}$, and that there is no arc between X and Y, and had there been such an arc, it would not be strongly protected. But the definition of strongly protected clearly states that an arc from X to Y with Z being a parent of X and not adjacent to Y *is* strongly protected, yielding a contradiction.

ii) Next, we prove that $\mathbf{\Sigma}$ attacks every sentence not in $\mathbf{\Sigma}$, and thus that it is stable and therefore a preferred extension. We consider each argument A of $\boldsymbol{L}$ in turn, and show that it is either a member of $\mathbf{\Sigma}$ or attacked by it.

First, let A be `Arc(X,Y)` for some variables X and Y ($X \rightsquigarrow Y$). If `Arc(X,Y)` is not in $\mathbf{\Sigma}^g$ then, by definition of a CLGK either `Arc(Y,X)`, `Link(X,Y)`, or `NonAdjacent(X,Y)` must be in $\mathbf{\Sigma}^g$ and $\mathbf{\Sigma}$ thus attacks `Arc(X,Y)` by one of Bullets 1 to 7. A similar argument applies if A is `Arc(Y,X)`, `Link(X,Y)`, or `NonAdjacent(X,Y)`.

Assume then that A is `ArcNotAllowed(X,Y)` for some pair of variables X and Y, and that $\mathbf{\Sigma}$ does not attack `ArcNotAllowed(X,Y)`. That means that none of the left hand sides of Bullets 25 to 28 are satisfied by members of $\mathbf{\Sigma}$. This in turn means that the graph encoded by $\mathbf{\Sigma}^g$, viz. $\mathcal{G}$, does not give strong protection to an arc from X to Y. As $\mathcal{G}$ is a completed pattern, it follows from Theorem 1 that there cannot be an arc from X to Y. However, by definition of $\mathbf{\Sigma}$ this means that `ArcNotAllowed(X,Y)` must be in $\mathbf{\Sigma}$.

Then assume that A is `DirectedPath(X,Y)` and that it is not attacked by $\mathbf{\Sigma}$. This means that ¬`DirectedPath(X,Y)` is not in $\mathbf{\Sigma}$, and by definition of $\mathbf{\Sigma}$, `DirectedPath(X,Y)` must therefore be in $\mathbf{\Sigma}$. Similar reasoning establishes that both `UndirectedPath(X,Y)` and `UndirectedPath(X,Y)-Excluding(Z,W)` are in $\mathbf{\Sigma}$ iff they are not attacked by $\mathbf{\Sigma}$.

Finally, assume that A is ¬`UndirectedPath(X,Y)`, and that it is not in $\mathbf{\Sigma}$. By definition of $\mathbf{\Sigma}$ this means that there is an undirected path $(X, \ldots, Y)$ between X and Y in $\mathcal{G}$. If the length of $(X, \ldots, Y)$ is 1, then there is an undirected link between the two variables X and Y, and consequently `Link[X,Y]` is in $\mathbf{\Sigma}^g$ and $\mathbf{\Sigma}$ therefore attacks ¬`UndirectedPath(X,Y)` by Bullet 12. If the length of $(X, \ldots, Y)$ is more than 1, there must be at least one variable Z on this path such that both $(X, \ldots, Z)$ and $(Z, \ldots, Y)$ are undirected paths in $\mathcal{G}$. It follows by the definition of $\mathbf{\Sigma}$ that both `UndirectedPath(X,Z)` and `UndirectedPath(Z,Y)` are in $\mathbf{\Sigma}$, and $\mathbf{\Sigma}$ then attacks ¬`UndirectedPath-(X,Y)` by Bullet 17. Similar (but more elaborate) arguments establish the result for arguments A of the form ¬`DirectedPath(X,Y)` and ¬`Undirected-Path(X,Y)Excluding(Z,W)`. □

Given these results it is thus clear that the result $\mathbf{\Sigma}$ of a debate, if undertaken respecting $\mathcal{A}^g$, is a completed pattern if it is a preferred extension of $\mathcal{A}^g$, and that $\mathbf{\Sigma}$ can represent any completed pattern. However, checking whether a set of arguments constitute a preferred extension is complex. It involves checks for both admissibility and maximality. We end this section with a result that yields a computationally efficient way of testing whether a set of arguments of $\mathcal{A}^g$ is a preferred extension.

Lemma 12. *Let $\mathbf{\Sigma}$ be a preferred extension of $\mathcal{A}^g$, and A a path sentence not in $\mathbf{\Sigma}$ and not attacked by it. Then the negative path sentence corresponding to A is not in $\mathbf{\Sigma}$ nor attacked by it.*

Proof. We show only the case where the path sentence is `UndirectedPath(X,Y)` for two variables X and Y, as the others cases are similar. Since `UndirectedPath(X,Y)` is not in $\mathbf{\Sigma}$ and not attacked by $\mathbf{\Sigma}$, it follows from Bullet 9, and the fact that this bullet represents the only attack on `UndirectedPath(X,Y)`, that ¬`UndirectedPath(X,Y)` cannot be in $\mathbf{\Sigma}$ either. Furthermore, as $\mathbf{\Sigma}$ is a preferred extension and does not attack `UndirectedPath(X,Y)`, it follows that $\mathbf{\Sigma}$ cannot attack ¬`UndirectedPath(X,Y)` either, as that would imply that `UndirectedPath(X,Y)` was defended by $\mathbf{\Sigma}$, which according to Lemma 8 would mean that it should have been a member of $\mathbf{\Sigma}$. ☐

Theorem 13. *Let $\mathbf{\Sigma}$ be a preferred extension of $\mathcal{A}^g$. Then $\mathbf{\Sigma}$ is a stable extension.*

Proof. We need to show that $\mathbf{\Sigma}$ attacks each argument A in $\boldsymbol{A} \setminus \mathbf{\Sigma}$. We do so by considering each possible A:

Assume first that A is one of `Arc(X,Y)`, `Arc(Y,X)`, `Link(X,Y)`, or `NonAdjacent(X,Y)`, for some variables X and Y. As $\mathbf{\Sigma}$ is a preferred extension, Lemma 7 guarantees that one of the other three arguments is in $\mathbf{\Sigma}$, and Bullets 1 to 7 then ensures that $\mathbf{\Sigma}$ attacks A.

Assume then that A is `ArcNotAllowed(X,Y)`, for some X and Y, and that it is not attacked by $\mathbf{\Sigma}$. This means that for each of the left-hand sides of Bullets 25 to 28 there is at least one element that is not in $\mathbf{\Sigma}$. Furthermore, as $\mathbf{\Sigma}$ does not defend `ArcNotAllowed(X,Y)`, there is at least one of these bullets, whose left-hand side is not attacked by $\mathbf{\Sigma}$. Assume that this bullet is Bullet 25. That means that either `Arc(Z,X)` or `NonAdjacent[Z,Y]` or both are neither in $\mathbf{\Sigma}$ nor attacked by it. But this is impossible, as was proved above. A similar contradiction arise for the remaining three bullets. It follows that `ArcNotAllowed(X,Y)` must be attacked by $\mathbf{\Sigma}$.

Now, assume that A is ¬`UndirectedPath(X,Y)`, and that $\mathbf{\Sigma}$ does not attack ¬`UndirectedPath(X,Y)`. As $\mathbf{\Sigma}$ does not attack ¬`UndirectedPath(X,Y)`, it follows from Bullet 17 that for all variables Z, one of `UndirectedPath[X,Z]` and `UndirectedPath[Z,Y]` cannot be in $\mathbf{\Sigma}$ either. Furthermore, Bullet 12 implies that `Link[X,Y]` cannot be in $\mathbf{\Sigma}$, and since it is a preferred extension, and by Lemma 7 thus a CLGK, it must attack `Link[X,Y]`. Since by Lemma 8 $\mathbf{\Sigma}$ fails to defend ¬`UndirectedPath(X,Y)`, then it must fail to attack both of `UndirectedPath[X,Z]` and `UndirectedPath[Z,Y]` for some variable Z. Without loss of generality, assume that `UndirectedPath[Z,Y]` is the path sentence which is both outside of $\mathbf{\Sigma}$ and not attacked by it. Lemma 12 then guarantees that ¬`UndirectedPath[Z,Y]` must also be outside of $\mathbf{\Sigma}$ and not attacked by $\mathbf{\Sigma}$. But ¬`UndirectedPath[Z,Y]` has the exact same form as the A we started out exploring, and we can therefore apply the same argument again, obtaining yet another negative path sentence that must be outside $\mathbf{\Sigma}$ and not be attacked by it. This process will never end, and since $\boldsymbol{L}$ is a finite language, it thus follows that there is a set $\boldsymbol{N}$ of negative path sentences, which are all outside $\mathbf{\Sigma}$ and not attacked by $\mathbf{\Sigma}$ solely because of other negative path sentences that are in $\boldsymbol{N}$. From the definition of $\boldsymbol{N}$ it is obvious that the set $\boldsymbol{P}$ of path sentences, corresponding to the negative ones in $\boldsymbol{N}$, attacks each of the sentences in $\boldsymbol{N}$, and furthermore that $\boldsymbol{N}$ contains all the sentences that attack $\boldsymbol{P}$. As $\mathbf{\Sigma}$ defends itself, it follows that the set $\mathbf{\Sigma}^* \equiv \mathbf{\Sigma} \cup \boldsymbol{P}$ defends itself against all attacks. If we can also show that $\mathbf{\Sigma}^*$ is conflict-free, then it contradicts that $\mathbf{\Sigma}$ is a preferred extension, and hence implies that the original claim, that ¬`UndirectedPath(X,Y)` is not attacked by $\mathbf{\Sigma}$, is false. To show that $\mathbf{\Sigma}^*$ is conflict-free, we assume otherwise, and let $\boldsymbol{S} \subseteq \mathbf{\Sigma}^*$ attack some B in $\mathbf{\Sigma}^*$. Assume first that B is in $\mathbf{\Sigma}$. As $\mathbf{\Sigma}$ defends itself, it must attack some argument in $\boldsymbol{S}$. But this is impossible since $\mathbf{\Sigma}$ is conflict-free and by construction $\boldsymbol{P}$ is not attacked by $\mathbf{\Sigma}$. Thus, B must be in $\boldsymbol{P}$. But this is also impossible, because the only arguments that attack $\boldsymbol{P}$ are the ones in $\boldsymbol{N}$, which by definition is not part of $\mathbf{\Sigma}^*$. It follows that $\mathbf{\Sigma}^*$ is conflict-free and thus that ¬`UndirectedPath(X,Y)` must be attacked by $\mathbf{\Sigma}$.

If A is `UndirectedPath(X,Y)` and is not attacked, then Lemma 12 guarantees that ¬`UndirectedPath(X,Y)` is also not in $\mathbf{\Sigma}$ and not attacked by it, which as we just saw, is impossible. Thus `UndirectedPath(X,Y)` must be attacked by $\mathbf{\Sigma}$.

Proving that `UndirectedPath(X,Y)Excluding(Z,W)` and ¬`UndirectedPath(X,Y)Excluding(Z,W)` must be attacked by $\mathbf{\Sigma}$ if they are not in $\mathbf{\Sigma}$ is proved in the same way as for `UndirectedPath(X,Y)` and ¬`UndirectedPath(X,Y)`. The case of ¬`DirectedPath(X,Y)` can be proved in a similar manner to ¬`UndirectedPath(X,Y)`, except that instead of only Bullet 17 to establish the existence of the set $\boldsymbol{N}$, we need to see that in any case at least one of Bullets 14 to 16 implies the existence of another ¬`DirectedPath(Z,W)` not in $\mathbf{\Sigma}$ and not attacked by $\mathbf{\Sigma}$. That `DirectedPath(X,Y)` is attacked then once again follows from Lemma 12. □

6 Fusing Agoras

We now address the problem of having agents agree on a preferred extension of $\mathcal{A}^g$, given that each of them has its own prior beliefs, as expressed by the compromise score function s_i, and that each know the combination function c. There has not been a lot of work done in dialectics for more than two agents, where the simple proponent/opponent dualism does not suffice. The solution that we propose here is inspired by the Risk Agoras of (McBurney and Parsons 2000) and (McBurney and Parsons 2001) and the traditional blackboard architecture of MAS of cooperating agents, without being an actual instantiation of any of them. We construct a *fusing agora*, which is a framework in which the agents can debate. The agora has the property that, if agents are allowed to run the debate to conclusion, they end up with the best possible compromise according to their joint compromise score, and that throughout the debate they maintain a compromise, which improves as the debate progresses.

In the agora we shall take a $\mathcal{A}^g$-candidate $(\boldsymbol{I}, \boldsymbol{O}, \boldsymbol{U})$ as a unique representative of a partial compromise $(\boldsymbol{I}, \boldsymbol{O})$. This is possible since $\boldsymbol{I}$ and $\boldsymbol{O}$ are subsets of $\boldsymbol{L}$, and thus both contain sentences describing aspects of a compromise graph as required, and since $\boldsymbol{U}$ is determined by $\boldsymbol{I}$ and $\boldsymbol{O}$. Any leaf candidate representing a preferred extension then uniquely identifies a completed pattern, as guaranteed by Theorem 10. Agents can explore all compromises by examining a $(\varnothing, \varnothing, \boldsymbol{L})$-tree. Continually the agents take it upon themselves to explore sub-trees of this tree, and mark other sub-trees as open for investigation by other agents. The heuristics guiding the agents' choices for exploration, in addition to $s_1, \ldots, s_k$ and c, then determine the outcome.

The agora can work in a variety of ways, depending on the behavior of the individual agents (a vanilla algorithm for an individual agent is provided later in Algorithm 1), but basically builds on two elements, which we assume each agent can access in a synchronized fashion only: A *pool of candidates* $\boldsymbol{C}$ and a *current best result* $\langle \boldsymbol{I}_*, s_{\boldsymbol{I}_*} \rangle$. $\boldsymbol{C}$ consists of pairs $\langle \mathcal{C}, s \rangle$, where $\mathcal{C}$ is an $\mathcal{A}^g$-candidate and thus a sub-tree of a $(\varnothing, \varnothing, \boldsymbol{L})$-tree, and s is a real value. $\boldsymbol{I}_*$ is either the empty set or a preferred extension of $\mathcal{A}^g$, and $s_{\boldsymbol{I}_*}$ is a real value. Initially, $\boldsymbol{C}$ contains only one element $\langle (\varnothing, \varnothing, \boldsymbol{L}), 0 \rangle$, and $\langle \boldsymbol{I}_*, s_{\boldsymbol{I}_*} \rangle$ is $\langle \varnothing, -\infty \rangle$.

Each agent i can utter the following locutions:

- *ExploreFromPool*$_i(\langle \mathcal{C}, s \rangle)$ — where $\langle \mathcal{C}, s \rangle$ is a member of $\boldsymbol{C}$. The meaning of the locution is that agent i takes upon itself the responsibility to investigate the preferred extensions in a $\mathcal{C}$-tree, assuming that $\mathcal{C}$ has a joint compromise score of s.
- *PutInPool*$_i(\langle \mathcal{C}, s \rangle)$ — where $\mathcal{C}$ is an $\mathcal{A}^g$-candidate, and s is a real value. The meaning of the locution is that agent i wants someone else to

investigate the preferred extensions in a $\mathcal{C}$-tree, and that $\mathcal{C}$ has a joint compromise score of s.

- $\mathit{UpdateBest}_i(\langle \boldsymbol{I}, s\rangle)$ — where $\boldsymbol{I}$ is a subset of $\boldsymbol{L}$, and s is a real value. The meaning of the locution is that agent i has identified a preferred extension $\boldsymbol{I}$ with a joint compromise score s higher than $s_{\boldsymbol{I}_*}$.
- $\mathit{AskOpinion}_i(\mathcal{C}_1, \mathcal{C}_2)$ — where $\mathcal{C}_1$ and $\mathcal{C}_2$ are $\mathcal{A}^g$-candidates. The meaning of the locution is that agent i needs to know $s_j(\mathcal{C}_1, \mathcal{C}_2)$ for all other agents j.
- $\mathit{StateOpinion}_i(\mathcal{C}_1, \mathcal{C}_2, s_\delta)$ — where $\mathcal{C}_1$ and $\mathcal{C}_2$ are $\mathcal{A}^g$-candidates, and s_δ is a real value. The meaning of the locution is that $s_i(\mathcal{C}_1, \mathcal{C}_2)$ is s_δ.

The rules governing which locutions individual agents can utter, as well as their effects, we present as a set of pre and post conditions:

- $\mathit{ExploreFromPool}_i(\langle \mathcal{C}, s\rangle)$
  - Pre: $\langle \mathcal{C}, s\rangle$ is in $\boldsymbol{C}$.
  - Post: $\langle \mathcal{C}, s\rangle$ is removed from $\boldsymbol{C}$
- $\mathit{PutInPool}_i(\langle \mathcal{C}, s\rangle)$
  - Pre: There is no $\langle \mathcal{C}', s'\rangle$ in $\boldsymbol{C}$ such that $\mathcal{C}$ is a sub-tree of some $\mathcal{C}'$-tree.
  - Post: $\langle \mathcal{C}, s\rangle$ is in $\boldsymbol{C}$.
- $\mathit{UpdateBest}_i(\langle \boldsymbol{I}, s\rangle)$
  - Pre: $s > s_{\boldsymbol{I}_*}$.
  - Post: $\langle \boldsymbol{I}_*, s_{\boldsymbol{I}_*}\rangle$ is set to $\langle \boldsymbol{I}, s\rangle$.

Locutions $\mathit{AskOpinion}_i()$ and $\mathit{StateOpinion}_i()$ have no pre or post conditions attached.

The basic algorithm in Algorithm 1 corresponds to an exhaustive search, if it is followed by all agents. The search is gradual in two senses: One, the longer the search goes on, the more elements the average candidate in $\boldsymbol{C}$ will have in its $\boldsymbol{I}$ and $\boldsymbol{O}$ sets, and thus the closer it will be to describing a full compromise. Two, the current compromise held in $\boldsymbol{I}_*$ will have an increasingly higher score.

In order for the search to be a success, each agent i would of course need to keep an eye out for $\mathit{AskOpinion}_j(\cdot)$'s uttered by other agents, and reply

Algorithm 1 *Vanilla algorithm for agent i*

$\langle \mathcal{C}, s \rangle \leftarrow$ SelectCandidate($\boldsymbol{C}$)
ExploreFromPool$_i(\langle \mathcal{C}, s \rangle)$
$\mathcal{C}' \triangleq (\boldsymbol{I}', \boldsymbol{O}', \boldsymbol{U}') \leftarrow$ Prune($\mathcal{C}$)
if $\boldsymbol{U}' = \varnothing$ **then**
  if PreferredExtension($\boldsymbol{I}'$) **then**
    AskOpinion$_i(\mathcal{C}, \mathcal{C}')$
    $s_i \leftarrow s_i(\mathcal{C}, \mathcal{C}')$
    wait for *StateOpinion*$_j(\mathcal{C}, \mathcal{C}', s_j) \forall j \neq i$
    $s' \leftarrow c(s_1, \ldots, s_k) + s$
    if $s' > s_{\boldsymbol{I}_*}$ **then**
      UpdateBest$_i(\langle \mathcal{C}', s' \rangle)$
    end if
  end if
  go to 1
else
  $A \leftarrow$ SelectArgument($\mathcal{C}'$)
  AskOpinion$_i(\mathcal{C}, \mathcal{C}' + A)$
  AskOpinion$_i(\mathcal{C}, \mathcal{C}' - A)$
  $s_i^+ \leftarrow s_i(\mathcal{C}, \mathcal{C}' + A)$
  $s_i^- \leftarrow s_i(\mathcal{C}, \mathcal{C}' - A)$
  wait for *StateOpinion*$_j(\mathcal{C}, \mathcal{C}' + A, s_j^+)$ **and** *StateOpinion*$_j(\mathcal{C}, \mathcal{C}' - A, s_j^-) \forall j \neq i$
  $s^+ \leftarrow c(s_1^+, \ldots, s_k^+)$
  $s^- \leftarrow c(s_1^-, \ldots, s_k^-)$
  if $s^+ > s^-$ **then**
    PutInPool$_i(\langle \mathcal{C}' - A, s + s^- \rangle)$
    $\mathcal{C} \leftarrow \mathcal{C}' + A$
    $s \leftarrow s + s^+$
  else
    PutInPool$_i(\langle \mathcal{C}' + A, s + s^+ \rangle)$
    $\mathcal{C} \leftarrow \mathcal{C}' - A$
    $s \leftarrow s + s^-$
  end if
  go to 3
end if

to these with $StateOpinion_i(\cdot)$. It is relatively easy to verify that agents using Algorithm 1 are uttering locutions in accordance with the pre and post conditions of the fusing agora. Algorithm 1 uses a series of helper functions, which are described below.

$\textsc{Prune}(\mathcal{C} \equiv (\boldsymbol{I}, \boldsymbol{O}, \boldsymbol{U}))$ uses pruning rules to investigate whether there is an argument A in $\boldsymbol{U}$ such that either $\mathcal{C} + A$ or $\mathcal{C} - A$ contains no leaves with preferred extensions. If this is the case, the method invokes itself recursively on the sub-tree that did not get pruned away, until no further branches can be pruned. Some general pruning rules are given in (Nielsen and Parsons 2006a), and more can be established for the specific case of $\mathcal{A}^g$. For instance, it is known that any preferred extension of $\mathcal{A}^g$ is stable, so whenever $\boldsymbol{I} \cup \boldsymbol{U}$ does not attack an argument $A \in \boldsymbol{U}$, then the branch corresponding to $\mathcal{C} - A$ cannot contain any preferred extensions.

$\textsc{SelectCandidate}(\boldsymbol{C})$ picks a promising candidate from $\boldsymbol{C}$. A promising candidate could be one with a high score annotated, since these encode good partial compromises, or candidates with small $\boldsymbol{U}$ sets, as these represent partial compromises that are nearly complete. If all agents use the same criteria for picking promising candidates, this selection can be sped up by implementing the pool as a sorted list. $\textsc{SelectCandidate}(\cdot)$ is one of the areas where heuristics limiting the search space can be implemented. For instance, it makes sense to allow agents to abstain from exploring the sub-tree rooted at a candidate if it cannot contain compromises that are consistent with their own BN. This would mean that in cases where agents agree on all or most of the aspects of $\mathcal{G}_*$ only few candidates would need to be explored.

$\textsc{PreferredExtension}(\boldsymbol{I})$ is a Boolean valued function that returns true if the conflict-free set $\boldsymbol{I}$ is a preferred extension of $\mathcal{A}^g$. The task of answering this is simplified by Theorem 13, as it states that $\boldsymbol{I}$ is a preferred extension iff $\boldsymbol{I}$ attacks each argument in $\boldsymbol{L} \setminus \boldsymbol{I}$.

$\textsc{SelectArgument}(\mathcal{C} \equiv (\boldsymbol{I}, \boldsymbol{O}, \boldsymbol{U}))$ simply selects an element A of $\boldsymbol{U}$. This selection can be based on the agent's own score increase going from $\mathcal{C}$ to $\mathcal{C} + A$ or $\mathcal{C} - A$, or it might involve negotiations or argumentation with other agents.

We illustrate the basics of the fusing agora and Algorithm 1 with an example.

Example 7: *Refer back to Examples 1 and 2, and consider the simple case where the compromise scoring function of the three agents is defined as the number of features that correspond to the completed pattern of their model minus those that do not, as exemplified in Example 3. We shall follow* *the initial actions in the agora, where $\boldsymbol{C}$ consists of only one candidate, $\langle \mathcal{C} \equiv (\varnothing, \varnothing, \boldsymbol{L}), 0\rangle$. We assume that Agent 1 is first to make a move.*

Agent 1's first choice for exploration is simple, as $\boldsymbol{C}$ only contains one element, and the agent thus utters ExploreFromPool$_1$($\langle \mathcal{C}, 0\rangle$). This candidate cannot be pruned, and since the third set in it is not empty, Agent 1 selects an argument to add to one of the other two sets, say, `Arc(A,D)`*. Agent 1 then constructs the two candidates (the third set of the candidates are left out, as it is simply $\boldsymbol{L}$ minus the first two sets):*

$$\begin{aligned}\mathcal{C} + \texttt{Arc(A,D)} =&(\texttt{Arc(A,D)}, \{\texttt{Arc(D,A)}, \texttt{Link(A,D)},\\ &\texttt{NonAdjacent(A,D)}, \texttt{ArcNotAllowed(A,D)},\\ &\neg\texttt{DirectedPath(A,D)}, \texttt{DirectedPath(D,A)},\\ &\texttt{UndirectedPath(A,D)}, \texttt{UndirectedPath(D,A)}\})\end{aligned}$$

and

$$\mathcal{C} - \texttt{Arc(A,D)} = (\varnothing, \{\texttt{Arc(A,D)}\}).$$

Agent 1 ask the other two agents their opinion by AskOpinion$_1$($\mathcal{C}, \mathcal{C} +$ `Arc(A,E)`) and AskOpinion$_1$($\mathcal{C}, \mathcal{C} -$ `Arc(A,E)`), and receives four answers -1, -1, 1, and 1 (as none of the other agents agree with `Arc(A,D)`*). Added to its own perception, Agent 1 gets the scores $s^+ = -3$ and $s^- = 3$, and it thus utters PutInPool$_i$($\langle \mathcal{C}+$`Arc(A,E)`$, -3\rangle$), and continues exploring $\mathcal{C}-$`Arc(A,E)`.*

Let us switch to Agent 2, who at this point utters ExploreFromPool$_2$($\langle \mathcal{C}+$`Arc(A,E)` $\equiv (\boldsymbol{I}_+, \boldsymbol{O}_+, \boldsymbol{U}_+), -3\rangle$), and takes on the responsibility of examining the sub-tree rooted at $\mathcal{C}+$`Arc(A,E)`. First, Agent 2 prunes the candidate, by adding `DirectedPath(A,D)` *to $\boldsymbol{I}_+$, and then chooses a new argument to investigate (seeing that $\boldsymbol{U}_+$ is not empty). For sake of coherence with previous examples, assume this argument is* `Arc(D,E)`*. Agent 2 creates the candidates*

$$\begin{aligned}&\mathcal{C}+\texttt{Arc(A,D)} + \texttt{Arc(D,E)} = (\boldsymbol{I}_+ \cup \{\texttt{DirectedPath(A,D)}, \texttt{Arc(D,E)}\},\\ &\quad\boldsymbol{O}_+ \cup \{\texttt{Arc(E,D)}, \texttt{Link(D,E)}, \texttt{NonAdjacent(D,E)},\\ &\quad\texttt{ArcNotAllowed(D,E)}, \neg\texttt{DirectedPath(D,E)},\\ &\quad\texttt{DirectedPath(E,D)}, \texttt{UndirectedPath(D,E)},\\ &\quad\texttt{UndirectedPath(E,D)}, \texttt{ArcNotAllowed(A,E)}\})\end{aligned}$$

and

$$\mathcal{C} + \texttt{Arc(A,D)} - \texttt{Arc(D,E)} = (\boldsymbol{I}_+, \boldsymbol{O}_+ \cup \{\texttt{Arc(D,E)}\}).$$

and gets an evaluation from Agents 1 and 3 using AskOpinion$_2$($\cdot$). These evaluations are $+1$, -1, -1, and $+1$, and taken together with $+1$ and -1

derived from Agent 2's own beliefs, it gets $s^+ = 1$ *and* $s^- = -1$*, so it puts* $\mathcal{C} + \texttt{Arc(A,D)} - \texttt{Arc(D,E)}$ *in* $\boldsymbol{C}$ *and continues exploring* $\mathcal{C} + \texttt{Arc(A,D)} + \texttt{Arc(D,E)}$*.*

Of course, the debate in the agora can be stopped at any time, and $\mathcal{G}[\boldsymbol{I}_*^g]$ will then be the best compromise encountered so far, as it is only ever replaced by compromises having a higher joint compromise score. More specifically we have that:

Proposition 14. *Let agents* 1 *to* k *argue in a fusing agora. If*

- *all agents have used Algorithm 1, and*
- *each agent* i *have replied to all* $\mathrm{AskOpinion}_j(\cdot)$*'s uttered by other agents, with a* $\mathrm{StateOpinion}_i(\cdot)$ *consistent with* s_i*,*

then all locutions uttered by agents are in accordance with the pre and post conditions of the fusing agora, and $(\boldsymbol{I}_*, \boldsymbol{L} \setminus \boldsymbol{I}_*, \varnothing)$ *has a higher joint compromise score* s_i *than all other explored leaf candidates* $(\boldsymbol{I}, \boldsymbol{L} \setminus \boldsymbol{I}, \varnothing)$*, where* $\boldsymbol{I}$ *is a preferred extension of* $\mathcal{A}^g$*.*

If we furthermore have that

- *all agents have completed processing,*
- *the pool of candidates has not been thinned along the way, and*
- *the pool is empty now,*

then $\mathcal{G}[\boldsymbol{I}_*^g] = \mathcal{G}_*$*.*

It is worth stressing that Algorithm 1 is a vanilla algorithm, and that the agora is open for more aggressive behaviour. One such behaviour could be to have agents skip the asking for opinions part in Lines 14 to 22 for most additions of arguments (and basing the decision only on the agents own beliefs), and only ask when the agent itself is indifferent. Another behaviour could be to never perform Lines 23 and 27, which would correspond to a myopic greedy construction of the compromise. Alternatively, these two lines could be carried out only when the difference between s^+ and s^- is very small. We could even have setups where the agents show different behaviours, or where individual agents change behaviour during debate depending on their available resources and utility of a good compromise. Moreover, the agora does not require that agents wait for a candidate to be in the pool, before somebody can start exploring this candidate; so even when one agent is pursuing an aggressive strategy and fails to leave candidates for others to explore, other agents can still decide to explore these. The point is, that no matter what behaviour is required, the basics of the agora and the agents remains the same, and can be reused.

7 Discussion

We have introduced a problem which we believe is a challenging one for the argumentation community, due to its mix of complexity and conditional decomposability as well as its origin in conflicting knowledge bases. Our own solution enables agents to judge the possible compromises resulting from a partial compromise, by constructing a candidate tree rooted in this partial compromise, and the agora we have proposed ensures that such exploration can take place in a distributed fashion. As an aside, we note that the method given here provides for a distributed solution to the problem of enumerating all equivalence classes of BNs (**?**).

One problem with the vanilla algorithm we have given, is that agents exploring a branch of a candidate-tree can end up putting a lot of candidates into the pool of annotated candidates. The space requirements for storing the pool of annotated candidates can be prohibitive, so it might be required that the candidates in the pool are defined in relation to each other, which imposes restrictions on which candidates an agent can choose to explore, as these are removed from the pool. Furthermore, it might be necessary to construct heuristics for thinning the pool of annotated candidates. These issues, as well as finding good heuristics for selecting candidates to explore are challenging topics for future research.

References

Andersson, S. A., D. Madigan, and M. D. Perlman (1997). A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics 25*(2), 505–541.

Cayrol, C., S. Doutre, and J. Mengin (2001). Dialectical proof theories for the credulous preferred semantics of argumentation frameworks. In *ECSQARU 6*, pp. 668–679. Springer.

Chickering, D. M. (2002). Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research 2*, 445–498.

Del Sagrado, J. and S. Moral (2003). Qualitative combination of bayesian networks. *International Journal of Intelligent Systems 18*(2), 237–249.

Dimopoulos, Y., B. Nebel, and F. Toni (1999). Preferred arguments are harder to compute than stable extension. In *IJCAI 16*, pp. 36–43. Morgan Kaufmann.

Doutre, S. and J. Mengin (2001). Preferred extensions of argumentation frameworks: Query, answering and computation. In *IJCAR*, Volume 2083, pp. 272–288. Springer.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence 77*(2), 321–358.

Elvang-Goransson, M. and A. Hunter (1995). Argumentative logics: Reasoning with classically inconsistent information. *Data Knowledge Engineering 16*(2), 125–145.

II, P. M.-R. and U. Chajewska (2001). Aggregating learned probabilistic beliefs. In *UAI 17*, pp. 354–361. Morgan Kaufmann.

Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs*. Springer.

Krause, P., S. Ambler, M. Elvang-Goeransson, and J. Fox (1995). A logic of argumentation for reasoning under uncertainty. *Computational Intelligence 11*(1), 113–131.

Lauritzen, S. L. (1996). *Graphical Models*. Oxford Statistical Science. Oxford.

Matzkevich, I. and B. Abramson (1992). The topological fusion of bayes nets. In *UAI 8*, pp. 191–198. Morgan Kaufmann.

McBurney, P. and S. Parsons (2000). Risk agoras: Dialectical argumentation for scientific reasoning. In *UAI 16*. Stanford.

McBurney, P. and S. Parsons (2001). Chance discovery using dialectical argumentation. In *Workshop on Chance Discovery, JSAI 15*. Matsue.

Nielsen, S. H. and S. Parsons (2006a). Computing preferred extensions for argumentation systems with sets of attacking arguments. Submitted For Publication.

Nielsen, S. H. and S. Parsons (2006b). A generalization of Dung's abstract framework for argumentation. In *ArgMAS 3*. Springer.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Representation & Reasoning. Morgan Kaufmann.

Pennock, D. M. and M. P. Wellman (1999). Graphical representations of consensus belief. In *UAI 15*, pp. 531–540. Morgan Kaufmann.

Pollock, J. L. (1995). *Cognitive Carpentry: A Blueprint for How to Build a Person*. MIT Press.

Richardson, M. and P. Domingos (2003). Learning with knowledge from multiple experts. In *ICML 20*, pp. 624–631. AAAI Press.

Saha, S. and S. Sen (2004). A Bayes net approach to argumentation based negotiation. In *ArgMAS 1*, pp. 208–222. Springer.

Simari, G. R. and R. P. Loui (1992). A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence 53*, 125–157.

Verma, T. and J. Pearl (1990). Equivalence and synthesis of causal models. In *UAI 6*, pp. 220–227. Elsevier Science Publishing Company, Inc.

Vreeswijk, G. (1997). Abstract argumentation systems. *Artificial Intelligence 90*, 225–279.

Vreeswijk, G. (1999). Bayesian inference and defeasible reasoning: suggestions for a unified framework. Working paper, Department of Computer Science, University of Utrecht.

Vreeswijk, G. and H. Prakken (2000). Credulous and sceptical argument games for preferred semantics. In *JELIA Proceedings*, Volume 1919, pp. 239–253. Springer.

Vreeswijk, G. A. W. (2004). Argumentation in Bayesian belief networks. In *ArgMAS 1*, Lecture Notes in Artificial Intelligence, pp. 111–129. Springer.

Xiang, Y. and V. Lesser (2003). On the role of multiply sectioned Bayesian networks for cooperative multiagent systems. *IEEE Systems, Man, and Cybernetics (Part A) 33*(4), 489–501.