Computational Models of Argument
F. Toni et al. (Eds.)

doi:10.3233/FAIA220151

Automated Reasoning with Epistemic Graphs Using SAT Solvers

Anthony HUNTER
Department of Computer Science,
University College London, London, UK
(anthony.hunter@ucl.ac.uk)

Abstract. Epistemic graphs have been developed for modelling an agent's degree of belief in an argument and how belief in one argument may influence the belief in other arguments. These beliefs are represented by constraints on probability distributions. In this paper, we present a framework for reasoning with epistemic graphs that allows for beliefs for individual arguments to be determined given beliefs in some of the other arguments. We present and evaluate algorithms based on SAT solvers.

Keywords. Probabilistic argumentation; Argumentation algorithms; Bipolar argumentation.

1. Introduction

Epistemic graphs are a generalization of the epistemic approach to probabilistic argumentation [1]. In epistemic graphs, the graph is augmented with a set of constraints on probability distributions. These constraints restrict the belief we have in each argument and they capture how beliefs in arguments influence each other. The aim is that a set of constraints captures the subjective, and possibly imperfect way, that an agent views the beliefs in the arguments and their interactions. The graphs can model both attack and support (see for example Figure 1) as well as relations that are neither positive nor negative (see for example Figure 3) with the label denoting the type of influence (e.g. positive (supporting), negative (attacking), and mixed). Both the label and the constraints provide information about the argumentation. In this paper, we focus on the constraints.

There are some similarities between epistemic graphs and graded and ranking–based semantics proposed for a number of argumentation frameworks [2,3,4,5,6,7,8,9,10,11,12] but there are also substantial differences. Most assign a value in the unit interval to arguments without further clarification of the meaning of the number. Furthermore, many of the postulates in these approaches are not really applicable in the epistemic approach, even though they can be perfectly suitable in other scenarios (e.g. in the epistemic approach, an increase or decrease in beliefs in attackers (or supporters) does not necessarily invoke an decrease or increase in the belief of the target argument).

Epistemic graphs have some similarities with abstract dialectical frameworks (ADFs) [13] and weighted ADFs (WADFs) [14]. However, differences include epistemic graphs allow for a finer-grained probabilistic evaluation of arguments, allowing unattacked arguments to be disbelieved, and long-distance effects between arguments that do not have

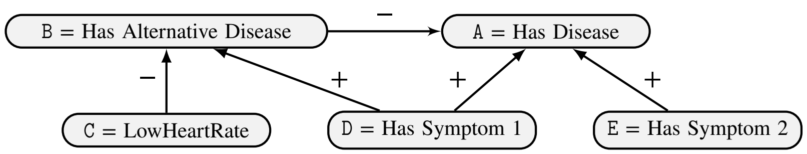

Figure 1. Example of an epistemic graph concerning diagnosis of a disease based on belief in symptoms and a differential diagnosis which in turn is based on a symptom and a test. The + (resp. −) label denote support (resp. attack) relations. Assume that if B is strongly believed, and D or E is strongly disbelieved, then A is strongly disbelieved, whereas if B is believed, and D or E is disbelieved, then A is disbelieved. Furthermore, if D and E are believed, then A is believed. These constraints could be reflected by the following formulae: $\varphi_1 : p(\mathtt{B}) > 0.8 \wedge p(\mathtt{D} \vee \mathtt{E}) < 0.2 \Rightarrow p(\mathtt{A}) < 0.8$; $\varphi_2 : p(\mathtt{B}) > 0.5 \wedge p(\mathtt{D} \vee \mathtt{E}) < 0.5 \Rightarrow p(\mathtt{A}) < 0.5$; $\varphi_3 : p(\mathtt{D} \wedge \mathtt{E}) > 0.5 \Rightarrow p(\mathtt{A}) > 0.5$; $\varphi_4 : p(\mathtt{C}) > 0.5 \Rightarrow p(\mathtt{B}) \leq 0.5$; And $\varphi_5 : p(\mathtt{C}) \leq 0.5 \Rightarrow p(\mathtt{B}) > 0.5$.

an arc connecting them. For more detailed comparison of ADFs with epigraphs, see [1]. Also see [1] for coverage of substantial differences with Bayesian networks.

In previous work, we presented a model-based theorem prover which can be used to check whether one constraint entails another that was based on enumerating all the models [15], and an approach based on calculating probability distributions satisfying the constraints using numerical optimization methods [16]. These methods only work for small numbers of arguments. Yet, there is a need for a scalable theorem prover that allows us to query the constraints of an epistemic graph in order to draw inferences about the belief in specific arguments. To address this need, we present a new proposal in this paper for taking a knowledgebase of constraints and optionally further assumptions, and drawing inferences from them. The approach involves representing the constraints as clauses, and then uses an off-the-shelf SAT solver (see [17] for an introduction to SAT solvers). We do this by defining a set of axioms, which we call a completion, of an epistemic graph which we add to the constraints when querying the SAT solver. By assuming these axioms, we can obtain a sound and complete inferencing algorithm.

2. Epistemic Graphs: A Simplified Version

In this paper, we present a simpler version of epistemic graphs than presented in [1]. Let $\mathcal{G}$ denote a graph where $\mathsf{Nodes}(\mathcal{G})$ be the set of nodes in $\mathcal{G}$, and $\mathsf{Arcs}(\mathcal{G})$ be the set of arcs in $\mathcal{G}$. We consider a probability distribution $P : \wp(\mathsf{Nodes}(G)) \rightarrow [0,1]$ as being a probability assignment to each subset of the set of arguments such that this sums to 1 (i.e. $\sum_{\Gamma \subseteq \mathsf{Nodes}(G)} P(\Gamma) = 1$). We denote the set of all probability distributions on $\mathsf{Nodes}(\mathcal{G})$ by $\mathsf{Dist}(\mathcal{G})$. The constraints restrict the set of probability distributions that satisfy the arguments (as we explain in the rest of this subsection).

Based on a given graph, we can now define the epistemic language. In this paper, we will only consider a sublanguage of that defined in [1]. The **simplified epistemic language** based on graph $\mathcal{G}$ is defined as follows: an **epistemic atom** is of the form $p(A)\#x$ where $\# \in \{<, \leq, =, \geq, >\}$, $x \in [0,1]$ and $A \in \mathsf{Nodes}(\mathcal{G})$; and an **epistemic formula** is a Boolean combination of epistemic atoms. For example, from the epistemic atoms $p(\mathtt{A}) \leq 0.5$ and $p(\mathtt{B}) \geq 0.5$, an epistemic formula is $p(\mathtt{A}) \leq 0.5 \rightarrow p(\mathtt{B}) \geq 0.5$.

The semantics for constraints come from probability distributions $P \in \mathsf{Dist}(\mathcal{G})$. Each $\Gamma \subseteq \mathsf{Nodes}(\mathcal{G})$ corresponds to a possible world where the arguments in Γ are true. The **probability of an argument** being acceptable is defined as the sum of the probabilities of

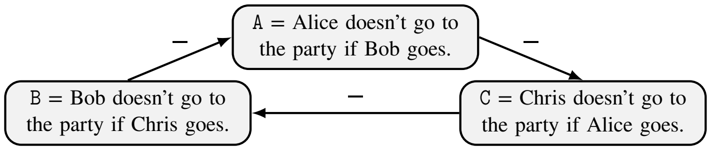
A = Alice doesn't go to the party if Bob goes.
B = Bob doesn't go to the party if Chris goes.
C = Chris doesn't go to the party if Alice goes.
−
−
−

Figure 2. The epistemic graph has constraints $\{p(\mathtt{B}) > 0.5 \Rightarrow p(\mathtt{A}) \leq 0.5,\ p(\mathtt{C}) > 0.5 \Rightarrow p(\mathtt{B}) \leq 0.5,\ p(\mathtt{A}) > 0.5 \Rightarrow p(\mathtt{C}) \leq 0.5\}$. Given these constraints, we see that at most one argument can be believed.

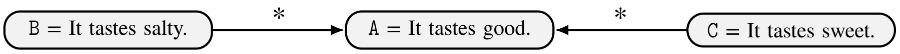

Figure 3. The epistemic graph has constraints $\{p(\mathtt{B}) > 0.5 \land p(\mathtt{C}) \leq 0.5 \Rightarrow p(\mathtt{A}) > 0.5,\ p(\mathtt{C}) > 0.5 \land p(\mathtt{B}) \leq 0.5 \Rightarrow p(\mathtt{A}) > 0.5,\ p(\mathtt{B}) > 0.5 \land p(\mathtt{C}) > 0.5 \Rightarrow p(\mathtt{A}) \leq 0.5\}$. Given these constraints, the influence of B and C on A is not simply a positive or a negative one. Consider some an item of food. If it is believed to be tasting salty and not believed to be tasting sweet, or it is not believed to be tasting salty and believed to be tasting sweet, then it is believed to be good tasting, and if it is believed to be tasting salty and believed to be tasting sweet, then it is not believed to be good tasting.

the worlds containing it: $P(A) = \sum_{\Gamma \subseteq \mathsf{Nodes}(\mathcal{G}) \text{ s.t. } A \in \Gamma} P(\Gamma)$. We say that an agent believes an argument A to be acceptable if $P(A) > 0.5$, disbelieves A to be acceptable if $P(A) < 0.5$, and neither believes nor disbelieves A to be acceptable when $P(A) = 0.5$.

For an epistemic atom $p(A)\#v$, where $\# \in \{<, \leq, =, \geq, >\}$, the **satisfying distributions**, or equivalently **models**, of $p(A)\#v$ are defined as $\mathsf{Sat}(p(A)\#v) = \{P' \in \mathsf{Dist}(\mathcal{G}) \mid P'(A)\#v\}$. The set of satisfying distributions for a given epistemic formula is as follows where ϕ and ψ are epistemic formulae: $\mathsf{Sat}(\phi \land \psi) = \mathsf{Sat}(\phi) \cap \mathsf{Sat}(\psi)$; $\mathsf{Sat}(\phi \lor \psi) = \mathsf{Sat}(\phi) \cup \mathsf{Sat}(\psi)$; and $\mathsf{Sat}(\neg\phi) = \mathsf{Sat}(\top) \setminus \mathsf{Sat}(\phi)$. For a set of epistemic formulae $\Phi = \{\phi_1, \ldots, \phi_n\}$, the set of satisfying distributions is $\mathsf{Sat}(\Phi) = \mathsf{Sat}(\phi_1) \cap \ldots \cap \mathsf{Sat}(\phi_n)$. A set of epistemic formulae is consistent iff its set of models is non-empty.

Example 1. *Consider the set of formulae* $\{p(\mathtt{A}) > 0.5 \rightarrow \neg(p(\mathtt{B}) > 0.5), p(\mathtt{A}) = 0 \lor p(\mathtt{A}) = 0.5 \lor p(\mathtt{A}) = 1, p(\mathtt{B}) = 0 \lor p(\mathtt{B}) = 0.5 \lor p(\mathtt{B}) = 1\}$. *Examples of probability distributions that satisfy the set include* P_1 *s.t.* $P_1(\emptyset) = 1$, P_2 *s.t.* $P_2(\emptyset) = P_2(\{\mathtt{A}\}) = 0.5$, P_3 *s.t.* $P_3(\{\mathtt{A}\}) = 1$, *or* P_4 *s.t.* $P_4(\{\mathtt{A}\}) = P_3(\{\mathtt{A},\mathtt{B}\}) = 0.5$ *(omitted sets are assigned* 0*). The probability distribution* P_5 *s.t.* $P_5(\{\mathtt{A},\mathtt{B}\}) = 1$ *does not satisfy the formula.*

For the arguments in graph $\mathcal{G}$, and probability function P, an **epistemic extension** is the set $\{A \in \mathsf{Nodes}(\mathcal{G}) \mid P(A) > 0.5\}$. So the extension is determined from the probability function rather the structure of the graph. For example, for Figure 2, if $P(\mathtt{A}) = 0.1$, $P(\mathtt{B}) = 0.9$, and $P(\mathtt{C}) = 0.1$, then the epistemic extension is $\{\mathtt{B}\}$.

We define an **entailment relation**, denoted $\models$, as follows, where Γ is a set of epistemic formulae, and ϕ is an epistemic formula: $\Gamma \models \phi$ iff $\mathsf{Sat}(\phi) \subseteq \mathsf{Sat}(\Gamma)$

Example 2. *Let* $\Gamma = \{p(\mathtt{C}) > 0.9, p(\mathtt{B}) = 0.3, p(\mathtt{C}) \geq 0.8 \land p(\mathtt{B}) < 0.6 \rightarrow p(\mathtt{A}) > 0.5\}$. *Hence,* $\mathsf{Sat}(p(\mathtt{A}) \geq 0.5) \subseteq \mathsf{Sat}(\Gamma)$*, and so* $\Gamma \models p(\mathtt{A}) \geq 0.5$.

The simplified epistemic language does not incorporate features of the full epistemic language (as presented in [1]) such as terms that are Boolean combinations of arguments (e.g. $P(\mathtt{B} \lor \mathtt{C}) > 0.6$ which says that the probability argument B or argument C is greater than 0.6) or summation of probability values (such as $P(\mathtt{A}) + P(\mathtt{B}) \leq 1$ which says that the sum of probability A and probability B is less than or equal to 1). Nonetheless, the restricted epistemic language is a useful sublanguage and it simplifies the presentation and evaluation in this paper.

An **epistemic constraint** is an epistemic formula $\psi \in \mathsf{Formulae}(\mathcal{G})$. An **epistemic graph** is a tuple $(\mathcal{G}, \mathcal{L}, \mathcal{C})$ where $(\mathcal{G}, \mathcal{L})$ is a labelled graph, and $\mathcal{C} \subseteq \mathsf{Formulae}(\mathcal{G})$ is a set of epistemic constraints associated with the graph.

In general, the graph (and its labellings) is not necessarily induced by the constraints and therefore it contains additional information. The actual direction of the edges in the graph is also not necessarily derivable from $\mathcal{C}$. For example, if we have two arguments A and B connected by an edge, a constraint of the form $p(\mathtt{A}) < 0.5 \vee p(\mathtt{B}) < 0.5$ would not tell us the direction of this edge. The constraints may also involve unrelated arguments, similar to [18], e.g. $\neg p(\mathtt{C}) > 0.5 \vee \neg p(\mathtt{D}) > 0.5$ when there is no arc between C and D. So the assignment of a label to an arc (by the $\mathcal{L}$ function) is an extra piece of information. The assignment is intended to denote the kind of influence of the source node on the target node. If we use the labels $\{+, *, -\}$, then the assignment of $+$ is intended to denote a form of support, the assignment of $-$ is intended to denote a form of attack, and $*$ is intended to denote an influence that is neither support nor attack. So $*$ could denote that under some conditions behaves as an attack and under some conditions behaves as a support as illustrated in the arc in Example 3. As investigated in [1], there are various ways that we can formalize the relationships between labels and constraints. We will not consider labels further in this paper, and we will focus on the constraints.

For this paper, we also require the notion of an **observation** which is an epistemic formula. The difference between constraints and observations is that we assume the constraints always hold, whereas observations only hold in some situations or for some periods. For example, if a debater uses an epistemic graph to model what opponents believe, the observations would be specific beliefs for a specific opponent.

Example 3. *Returning to Figure 2, suppose we have the observation* $p(\mathtt{C}) \geq 0.8$*, then we want to draw the conclusions* $p(\mathtt{B}) \leq 0.5$ *and* $p(\mathtt{A}) \leq 0.5$*.*

Example 4. *Returning to Figure 3, suppose we have the observations* $p(\mathtt{B}) = 0.7$ *and* $p(\mathtt{C}) = 0.2$*, then we want to draw the conclusion* $p(\mathtt{A}) > 0.5$*. Or suppose we have the observations* $p(\mathtt{B}) > 0.7$ *and* $p(\mathtt{C}) \geq 0.8$*, then we want to draw the conclusion* $p(\mathtt{A}) \leq 0.5$*.*

In the following, we will use the term **knowledgebase**, denoted $\mathcal{K}$, to refer to the union of a set of constraints and a set of observations.

3. Reasoning with Epistemic Graphs

In this paper, our approach to inference with constraints and observations is to use SAT solvers. So we will need to represent constraints and observations as clauses (i.e. a disjunction of literals). Any formula of propositional logic (and similarly any epistemic formula) can be rewritten in conjunction normal form, and then conjunction elimination applied, to obtain a set of clauses that are logically equivalent to the original epistemic formula. So we do not lose any expressibility if we represent our epistemic formulae as clauses. Note, clauses can be rewritten as implications. So $\beta_1 \vee \ldots \vee \beta_{n-1} \vee \beta_n$ can be represented as $\neg\beta_1 \wedge \ldots \wedge \neg\beta_{n-1} \rightarrow \beta_n$.

We will also restrict the probability values that the formulae can take by using a **restricted value set**, denoted Π, which is a subset of the unit interval such that $0,1 \in \Pi$, and for all $x,y \in \Pi$, if $x+y \in [0,1]$, then $x+y \in \Pi$, and if $x-y \in [0,1]$, then $x-y \in \Pi$. For example, $\{0,0.5,1\}$ and $\{0,0.1,0.2,\ldots,0.9,1\}$ are restricted value sets. In this paper, we will assume $\Pi = \{0,0.1,0.2,\ldots,0.9,1\}$ unless explicitly stated otherwise.

Definition 1. *The* **restricted language** *based on graph $\mathcal{G}$ and a restricted value set Π is defined as follows: a* **restricted atom** *of the form $p(A)\#x$ where $\# \in \{<,\leq,=,\geq,>\}$, $x \in \Pi$ and $A \in \mathsf{Nodes}(\mathcal{G})$; a* **restricted clause** *of the form $\beta_1 \vee \ldots \vee \beta_n \vee \beta_{n+1}$ where each β_i in $\{\beta_1,\ldots,\beta_n,\beta_{n+1}\}$ is a* **restricted literal** *(i.e. a restricted atom, or its negation).*

Example 5. *Let $\Pi = \{0,0.5,1\}$. In the restricted language w.r.t. Π, we can only have atoms of the form $p(\mathtt{A})\#0$, $p(\mathtt{A})\#0.5$, and $p(\mathtt{A})\#1$, where $\mathtt{A} \in \mathsf{Nodes}(\mathcal{G})$ and $\# \in \{<,\leq,=,\geq,>\}$. From these atoms we compose epistemic formulae, using the Boolean connectives, such as $p(\mathtt{A}) \leq 0.5 \rightarrow \neg(p(\mathtt{B}) \geq 0.5)$.*

We also require some subsidiary definitions. Literals ϕ and ψ are **logically complementary** iff ϕ is $\neg\psi$ or ψ is $\neg\phi$. (e.g. the literals $P(\mathtt{A}) > 0.8$ and $\neg(P(\mathtt{A}) > 0.8)$ are logically complementary); And the literals ϕ and ψ are **probabilistically complementary** iff $\mathsf{Sat}(\phi) \cap \mathsf{Sat}(\psi) = \emptyset$ and ϕ and ψ are not logically complementary (e.g. $P(\mathtt{A}) > 0.8$ and $P(\mathtt{A}) < 0.8$ are probabilistically complementary, and when $\Pi = \{0,0.1,\ldots,0.9,1.0\}$, $P(\mathtt{A}) > 0.9$ and $\neg(P(\mathtt{A}) = 1)$ are probabilistically complementary).

To reason with a knowledgebase (i.e. a set of constraints and observations), we propose a proof theoretic approach based on adding extra axioms to the knowledgebase to capture the implicit probabilistic information that is required. For this we introduce the notion of equality completion to reduce our knowledgebase and query to disjunctions involving only equality and the restricted value set Π. For example the atom $p(A) > 0.6$ implies $p(A)$ is one of 0.7, 0.8, 0.9, or 1 as captured by the following clause.

$$\neg(p(\mathtt{A}) > 0.6) \vee p(\mathtt{A}) = 0.7 \vee p(\mathtt{A}) = 0.8 \vee p(\mathtt{A}) = 0.9 \vee p(\mathtt{A}) = 1.0$$

In the following definition of completion, we also include the constraint that an argument cannot have two values. So for all arguments $\mathtt{A}$, for all $x,y \in \{0,0.1,0.2,\ldots,0.9,1\}$, s.t. $x \neq y$, $\neg(p(\mathtt{A}) = x) \vee \neg(p(\mathtt{A}) = y)$.

Definition 2. *For a graph $\mathcal{G}$, the set of* **completion clauses** *is the following set of clauses*

$$\mathsf{Complete}(\mathcal{G}) = \bigcup_{A \in \mathsf{Nodes}(\mathcal{G})} \left(\left(\bigcup_{k \in \{1,\ldots,8\}} \mathsf{C}_k(A) \right) \cup \mathsf{Exclusion}(A) \right)$$

where $\mathsf{Exclusion}(A) = \{\neg(p(A) = x) \vee \neg(p(A) = y) \mid x \neq y\}$ *and*

$$\begin{aligned}
\mathsf{C}_1(A) &= \{p(A) > x \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x \leq y_1,\ldots,y_n\}\\
\mathsf{C}_2(A) &= \{\neg(p(A) > x) \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x > y_1,\ldots,y_n\}\\
\mathsf{C}_3(A) &= \{p(A) < x \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x \geq y_1,\ldots,y_n\}\\
\mathsf{C}_4(A) &= \{\neg(p(A) < x) \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x < y_1,\ldots,y_n\}\\
\mathsf{C}_5(A) &= \{\neg(p(A) \leq x) \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x \leq y_1,\ldots,y_n\}\\
\mathsf{C}_6(A) &= \{p(A) \leq x \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x > y_1,\ldots,y_n\}\\
\mathsf{C}_7(A) &= \{\neg(p(A) \geq x) \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x \geq y_1,\ldots,y_n\}\\
\mathsf{C}_8(A) &= \{p(A) \geq x \vee p(A) = y_1 \vee \ldots \vee p(A) = y_n \mid x < y_1,\ldots,y_n\}
\end{aligned}$$

The size of $\mathsf{Complete}(\mathcal{G})$ is a linear function of the number of arguments in the graph $\mathcal{G}$, as there are 198 axioms in $\mathsf{Complete}(\mathcal{G})$ per argument.

Proposition 1. *If* $|\mathsf{Nodes}(\mathcal{G})| = n$, *then* $|\mathsf{Complete}(\mathcal{G})| = 198n$.

Proof. For each argument in $A \in \mathsf{Nodes}(\mathcal{G})$, there is 11 axioms for each of Com_1 to Com_8 (there are 11 axioms since there is one axiom per value of x), and there are 110 exclusion axioms (since, for $\neg(p(A) = x) \vee \neg(p(A) = y)$, there are 11 choices for x and therefore 10 choices for y, which is 110 choices), giving a total of 198 axioms per argument. □

The axioms given in the completion are sound. In other words, they are satisfied by all probability distributions, and are therefore entailed by any knowledgebase.

In the following definition, we present the resolution proof rule as part of the resolution proof relation. This proof rule takes a pair of clauses where one has a disjunct, and the other has a disjunct that is its negation, and returns a clause where the disjuncts are all the disjuncts from the original clauses except the disjunct in the first clauses that is negated in the second clause.

Definition 3. *Let ϕ and ϕ' be clauses where ϕ is of the form $\alpha \vee \beta$ and ϕ' is of the form $\gamma \vee \delta$, and α and γ are logically complementary literals (i.e. α is $\neg\gamma$ or $\neg\alpha$ is γ), then $\beta \vee \delta$ is a* **resolvent** *of ϕ and ϕ'. The* **resolution proof relation**, *denoted $\vdash_{\mathsf{resolution}}$, is defined as follows where Δ is a set of clauses and ψ is a clause where the proof rules are: (1) Resolution; (2) Reflexivity; (3) Associativity; and (4) Contradiction.*

1 $\Delta \vdash_{\mathsf{resolution}} \beta \vee \delta$ *if* $\Delta \vdash_{\mathsf{resolution}} \alpha \vee \beta$ & $\Delta \vdash_{\mathsf{resolution}} \gamma \vee \delta$ & α *is* $\neg\gamma$
2 $\Delta \vdash_{\mathsf{resolution}} \phi$ *if* $\phi \in \Delta$
3 $\Delta \vdash_{\mathsf{resolution}} \alpha_1 \vee \ldots \vee \alpha_m$ *if* $\Delta \vdash_{\mathsf{resolution}} \beta_1 \vee \ldots \vee \beta_n$ & $\{\alpha_1, \ldots, \alpha_m\} = \{\beta_1, \ldots, \beta_n\}$
4 $\Delta \vdash_{\mathsf{resolution}} \bot$ *if* $\Delta \vdash_{\mathsf{resolution}} \phi$ & $\Delta \vdash_{\mathsf{resolution}} \neg\phi$

We now consider resolution with a knowledgebase and completion. Consider two clauses and two literals (one in each clause) that are either logically complementary or probabilistically complementary. For entailment, there is no probability distribution that satisfies both literals, and so the inference follows. In contrast, the resolution proof rule only deals with logically complementary literals, and so the completion is required to treat probabilistically complementary literals as logically complementary literals, and thereby obtain the inference. We illustrate this in the following example.

Example 6. *Consider $\phi_1 = p(\mathtt{A}) > 0.8 \vee p(\mathtt{B}) > 0.5$ and $\phi_2 = p(\mathtt{A}) < 0.2 \vee p(\mathtt{B}) > 0.5$. Clearly, $p(\mathtt{A}) > 0.8$ and $p(\mathtt{A}) < 0.2$ are probabilistically complementary literals, and that $\{\phi_1, \phi_2\} \models p(\mathtt{B}) > 0.5$ holds. The following axioms are from the completion.*

$\pi_1 = \neg(p(\mathtt{A}) > 0.8) \vee p(\mathtt{A}) = 0.9 \vee p(\mathtt{A}) = 1$ $\qquad$ $\pi_4 = \neg(p(\mathtt{A}) = 1) \vee \neg(p(\mathtt{A}) = 0)$
$\pi_2 = \neg(p(\mathtt{A}) < 0.2) \vee p(\mathtt{A}) = 0 \vee p(\mathtt{A}) = 0.1$ $\qquad$ $\pi_5 = \neg(p(\mathtt{A}) = 0.9) \vee \neg(p(\mathtt{A}) = 0.1)$
$\pi_3 = \neg(p(\mathtt{A}) = 0.9) \vee \neg(p(\mathtt{A}) = 0)$ $\qquad$ $\pi_6 = \neg(p(\mathtt{A}) = 1) \vee \neg(p(\mathtt{A}) = 0.1)$

We now show that $p(\mathtt{B}) > 0.5$ can be obtained using the resolution proof relation with the completion of the knowledge. We use the names of clauses rather than the clauses in the premises to save space. The name of each clause generated by resolution is given on the right after the clause.

1 $\{\phi_1, \pi_1\} \vdash_{\mathsf{resolution}}$	$P(\mathtt{A}) = 0.9 \lor P(\mathtt{A}) = 1 \lor P(\mathtt{B}) > 0.5$	(ω_1)
2 $\{\phi_2, \pi_2\} \vdash_{\mathsf{resolution}}$	$P(\mathtt{A}) = 0 \lor P(\mathtt{A}) = 0.1 \lor P(\mathtt{B}) > 0.5$	(ω_2)
3 $\{\omega_1, \pi_3\} \vdash_{\mathsf{resolution}}$	$\neg(P(\mathtt{A}) = 0) \lor P(\mathtt{A}) = 1 \lor P(\mathtt{B}) > 0.5$	(ω_3)
4 $\{\omega_3, \pi_4\} \vdash_{\mathsf{resolution}}$	$\neg(P(\mathtt{A}) = 0) \lor P(\mathtt{B}) > 0.5$	(ω_4)
5 $\{\omega_1, \pi_5\} \vdash_{\mathsf{resolution}}$	$\neg(P(\mathtt{A}) = 0.1) \lor P(\mathtt{A}) = 1 \lor P(\mathtt{B}) > 0.5$	(ω_5)
6 $\{\omega_5, \pi_6\} \vdash_{\mathsf{resolution}}$	$\neg(P(\mathtt{A}) = 0.1) \lor P(\mathtt{B}) > 0.5$	(ω_6)
7 $\{\omega_2, \omega_4\} \vdash_{\mathsf{resolution}}$	$P(\mathtt{A}) = 0.1 \lor P(\mathtt{B}) > 0.5$	(ω_7)
8 $\{\omega_6, \omega_7\} \vdash_{\mathsf{resolution}}$	$P(\mathtt{B}) > 0.5$	(ω_8)

In the following lemma, we generalize the above example by showing that if a clause is entailed by a pair of clauses, then that inference can be obtained from the completion of the clauses using only the resolution proof rule.

Lemma 1. *For graph $\mathcal{G}$, if ϕ, ϕ', ψ are clauses where ϕ is of the form $\alpha_1 \lor \ldots \lor \alpha_n$, ϕ' is of the form $\beta_1 \lor \ldots \lor \beta_m$, ψ is of the form $\alpha_1 \lor \ldots \lor \alpha_{n-1} \lor \beta_1 \lor \ldots \lor \beta_{m-1}$, and $\{\phi, \phi'\} \models \psi$, then $\{\phi, \phi'\} \cup \mathsf{Complete}(\mathcal{G}) \vdash_{\mathsf{resolution}} \psi$.*

Proof. Assume $\{\phi, \phi'\} \models \psi$. So for all $P \in \mathsf{Sat}(\{\phi, \phi'\})$, $P \not\models \alpha_n$ or $P \not\models \beta_m$. So either α_n and β_m are logically complementary literals (i.e. syntactically, α_n is $\neg\beta_m$ or $\neg\alpha_n$ is β_m) or α_n and β_m are probabilistically complementary literals (i.e. α_n is of the form $p(A_1)\#_1 v_1$ and β_m is of the form $p(A_2)\#_2 v_2$ and there is no assignment for w_1 and w_2 where $P(A_1) = w_1$ and $P(A_2) = w_2$ that would satisfy α_n and β_m). In the case that α_n and β_m are logically complementary literals, then $\{\phi, \phi'\} \vdash_{\mathsf{resolution}} \psi$ holds, and hence $\{\phi, \phi'\} \cup \mathsf{Complete}(\mathcal{G}) \vdash_{\mathsf{resolution}} \psi$ holds. In the case that α_n and β_m are probabilistically complementary literals, then the disjunct α_n in ϕ is resolved with a completion axiom and so exchanged for a disjunction of $p(A_1) = y_1 \lor \ldots \lor p(A_1) = y_n$, and the disjunct β_n in ϕ' is resolved with a completion axiom and so exchanged for a disjunction of $p(A_2) = y'_1 \lor \ldots \lor p(A_2) = y'_n$. So together with the exclusion axioms, there is no assignment for w_1 and w_2 in $p(A_1) = w_1$ and $p(A_2) = w_2$ that would satisfy $p(A_1) = y_1 \lor \ldots \lor p(A_1) = y_n$ and $p(A_2) = y'_1 \lor \ldots \lor p(A_2) = y'_n$. So each of these incompatible assignments is removed by resolution until none of them remain. So via a number of resolution steps, $\{\phi, \phi'\} \cup \mathsf{Complete}(\mathcal{G}) \vdash_{\mathsf{resolution}} \psi$. □

The following correctness result shows that a literal α is entailed if and only if the negation of the query together with the knowledgebase and completion results in a contradiction using the resolution consequence relation

Proposition 2. *For all epistemic graphs $(\mathcal{G}, \mathcal{L}, \mathcal{C})$, and literals α, $\mathcal{C} \models \alpha$ iff $\mathcal{C} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\} \vdash_{\mathsf{resolution}} \bot$.*

Proof. ($\Rightarrow$) Assume $\mathcal{C} \models \alpha$. Therefore $\mathsf{Sat}(\mathcal{C} \cup \{\neg\alpha\}) = \emptyset$. Therefore there is a subset $\Gamma \subseteq \mathcal{C} \cup \{\neg\alpha\}$ such that $\mathsf{Sat}(\Gamma) = \emptyset$ and for all $\Gamma' \subseteq \Gamma$, $\mathsf{Sat}(\Gamma') \neq \emptyset$. So for all $\phi \in \Gamma$, and for all $\alpha \in \mathsf{Disjuncts}(\phi)$, $\Gamma \setminus \{\phi\} \models \neg\alpha$. Moreover, for all $\phi, \phi' \in \Gamma$, and for all ψ such that ψ is a resolvent of ϕ and ϕ', $\Gamma \vdash \psi$, and by Lemma 1, $\Gamma \cup \mathsf{Complete}(\mathcal{G}) \vdash_{\mathsf{resolution}} \psi$. Since $\mathsf{Sat}(\Gamma) = \emptyset$, $\Gamma \vdash \bot$, and by Lemma 1, $\Gamma \cup \mathsf{Complete}(\mathcal{G}) \vdash_{\mathsf{resolution}} \bot$. So $\mathcal{C} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\} \vdash_{\mathsf{resolution}} \bot$. ($\Rightarrow$) Assume $\mathcal{C} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\} \vdash_{\mathsf{resolution}} \bot$. So $\mathsf{Sat}(\mathcal{C} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\}) = \emptyset$. Since all $\delta \in \mathsf{Complete}(\mathcal{G})$ are satisfied by all $P \in \mathsf{Dist}(\mathcal{G})$ (i.e. for all $P \in \mathsf{Dist}(\mathcal{G})$, $P \models \delta$), we have $\mathsf{Sat}(\mathsf{Complete}(\mathcal{G})) = \mathsf{Dist}(\mathcal{G})$. Therefore, $\mathsf{Sat}(\mathcal{C} \cup \{\neg\alpha\}) = \emptyset$. Hence, $\mathcal{C} \models \alpha$ holds. □

Algorithm 1 Clausal inference for knowledgebase $\mathcal{K}$, query α, and graph $\mathcal{G}$

```
function INFERENCE($\mathcal{K}$,$\alpha$,$\mathcal{G}$ )
    if $\alpha$ is a positive literal then
        return NOT SAT($\mathcal{K} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\}$)
    else
        return NOT SAT($\mathcal{K} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\beta\}$) where $\alpha$ is of the form $\neg\beta$
```

Algorithm 2 Bounds for argument A w.r.t knowledgebase $\mathcal{K}$, graph $\mathcal{G}$, and increments μ.

```
function TIGHTINFERENCE($\mathcal{K}$,$A$,$\mu$,$\mathcal{G}$)
    $n = 0$
    while INFERENCE($\mathcal{K}, p(A) \geq n), \mathcal{G}$ do
        $n = n + \mu$
    $m = 1$
    while INFERENCE($\mathcal{K}, p(A) \leq m), \mathcal{G}$ do
        $m = m - \mu$
    return $(n, m)$
```

4. Algorithms

The inference algorithm (Algorithm 1) calls the SAT solver with a knowledgebase, and its completion, plus the negation of the query. If the SAT solver returns True, then the set of formulae is consistent, and hence the query does not follow from the premises, whereas if the SAT solver returns False, then the set of formulae is inconsistent, and hence the query does follow from the premises.

Proposition 3. *For a knowledgebase $\mathcal{K}$, and restricted literal α,* INFERENCE($\mathcal{K},\alpha,\mathcal{G}$) = True *iff* $\mathcal{K} \models \alpha$.

Proof. INFERENCE($\mathcal{K},\alpha,\mathcal{G}$) = True iff $\mathcal{K} \cup \mathsf{Complete}(\mathcal{G}) \cup \{\neg\alpha\} \vdash_{SAT} \perp$ iff $\mathcal{K} \models \alpha$. □

We also give an algorithm for obtaining bounds on a query (Algorithm 2). It obtains the tightest bounds $n, m \in \Pi$ such that $\mathcal{K} \models p(A) \geq n$ and $\mathcal{K} \models p(A) \leq m$ hold. The parameter μ specifies the restricted value set. For example, $\mu = 0.5$ when $\Pi = \{0, 0.5, 1\}$ and $\mu = 0.1$ when $\Pi = \{0, 0.1, 0.2, \ldots, 0.9, 1\}$.

Example 7. *Given the constraints* $\{p(\mathtt{A}) \geq 0.4), p(\mathtt{A}) < 0.7 \vee p(\mathtt{B}) < 0.5\}$ *and the observations* $\{p(\mathtt{B}) \geq 0.5\}$*, we obtain* $(0.4, 0.6)$ *from Algorithm 2 (i.e.* 0.4 *as the lower bound for* $p(\mathtt{A})$ *and* 0.6 *as the upper bound for* $p(\mathtt{A})$*).*

The algorithms (i.e. Algorithms 1 and 2) were implemented on Python. The implementation[1] uses the PySAT implementation [19] that incorporates SAT solvers such as Glucose3. The implementation includes code to randomly generate sets of epistemic constraints and queries. For a given number of arguments, and an upper limit on the number of disjuncts in each clause, the code randomly selects the argument, comparator and probability value for each atom in the clause. Each query is generated in the same way.

[1] http://www0.cs.ucl.ac.uk/staff/a.hunter/papers/episat.zip

	(2,10)	(2,100)	(4,10)	(4,100)	(6,10)	(6,100)
25	0.22	0.27	0.44	0.18	0.18	0.45
50	0.40	0.43	0.87	0.38	0.39	0.82
75	0.65	0.62	0.88	0.63	0.65	0.88
100	0.92	0.87	0.96	0.93	0.90	0.94
125	1.19	1.17	1.17	1.19	1.17	1.28
150	1.56	1.52	1.50	1.42	1.52	1.56
175	1.82	2.15	2.38	1.87	1.81	3.05
200	2.10	1.98	3.76	2.19	2.12	2.67
225	2.46	2.48	2.52	2.52	2.55	2.59
250	3.02	4.17	3.03	3.04	2.82	3.04

Table 1. Experiments with the INFERENCE algorithm (Algorithm 1). Each column is for a pair (d, c) where d is the upper limit of disjuncts (taking the value 2, 4, or 6 disjuncts) and c is cardinality of knowledgebase (taking the value of 10 or 100 clauses). Each row is the number of arguments in the range 25 to 250. For each combination of column and row, we obtained the average time taken (seconds) obtained over 10 runs.

Combination	$(a = 10, c = 10)$	$(a = 20, c = 20)$	$(a = 30, c = 30)$	$(a = 40, c = 40)$
Average time	1.47	3.25	5.61	7.55

Table 2. For each combination, where a is the number of arguments, and c is the number of clauses, we obtained the average time taken (seconds) obtained over 20 runs for each number of arguments.

The main purpose of the evaluation was to determine how the inference algorithm performs with the number of arguments (propositional letters), disjuncts per clauses, and clauses per knowledgebase. We considered the values 2, 4, and 6 for the number of disjuncts as this reflects what might be common values in applications, we considered 10 and 100 for the number of clauses, and similarly between 25 and 250 arguments, as they represent the numbers that might be found in small and larger applications. Table 1 shows that for each row, the time taken was similar for each column. So increasing the number of disjuncts per clause (i.e. d), or increasing the number of clauses (i.e. c), does not substantially affect the time taken. In contrast, the number of arguments does substantially increase the time taken. This can be clearly seen in each column.

The algorithm for bounds involves more computation since repeated queries are made to the inference algorithm. As a result the average time to obtain bounds were slower than for entailment as indicated by the results in Table 2. A simple improvement to the algorithm to decrease the time would be to only form the completion once (rather than form the completion each time the inference algorithm is called) and then use this completion each time the SAT solver is called.

The conclusion that we draw from the evaluations is that by basing the algorithms on off-the-shelf SAT solvers, we are able to have scalable reasoning with epistemic graphs. Given a set of constraints for an epistemic graph together with a set of observations, we are able to quickly determine the belief in any of the arguments. In other words, the belief on some arguments can be efficiently propagated through the graph to determine the belief in the others. We can claim that this is scalable because we see that even with 100s of arguments with clauses of up to 6 disjuncts, and a set of constraints plus observation of 100 clauses, the time taken is a few seconds. For instance, with 200 arguments, a maximum of 5 disjuncts, and 500 clauses in the knowledgebase, the average time is 2.71 seconds.

5. An Application of Automated Reasoning

We now consider an extended example (which has been adapted from [1]) to illustrate how we can use the automated reasoning as part of an automated persuasion system. Assume we have the graph presented in Figure 4 and that through, for instance, crowdsourcing data, we have learned which constraints should be associated with a given user profile. So now we assume we are dealing with a user of an automated persuasion system whose profile leads to the selection of the following constraints in order to predict his or her attitudes.

(1) $p(\mathtt{B}) > 0.5 \land p(\mathtt{C}) < 0.5 \land p(\mathtt{D}) < 0.5 \rightarrow p(\mathtt{A}) > 0.5$
(2) $p(\mathtt{B}) > 0.7 \land p(\mathtt{C}) < 0.5 \land p(\mathtt{D}) < 0.5 \rightarrow p(\mathtt{A}) > 0.8$
(3) $p(\mathtt{B}) > 0.9 \land p(\mathtt{C}) < 0.5 \land p(\mathtt{D}) < 0.5 \rightarrow p(\mathtt{A}) > 0.9$
(4) $p(\mathtt{C}) \geq 0.9 \rightarrow p(\mathtt{A}) < 0.25$
(5) $p(\mathtt{D}) \leq 0.5 \rightarrow p(\mathtt{A}) \geq 0.25$
(6) $p(\mathtt{D}) > 0.75 \rightarrow p(\mathtt{A}) < 0.75$
(7) $p(\mathtt{E}) > 0.9 \rightarrow p(\mathtt{B}) < 0.5$
(8) $p(\mathtt{E}) \leq 0.5 \land p(\mathtt{F}) > 0.5 \rightarrow p(\mathtt{B}) > 0.5$
(9) $p(\mathtt{G}) > 0.5 \rightarrow p(\mathtt{C}) < 0.5$
(10) $p(\mathtt{H}) > 0.5 \rightarrow p(\mathtt{D}) < 0.5$
(11) $p(\mathtt{I}) > 0.75 \rightarrow p(\mathtt{D}) \leq 0.5$
(12) $p(\mathtt{J}) > 0.5 \rightarrow p(\mathtt{E}) < 0.5$
(13) $p(\mathtt{J}) > 0.5 \rightarrow p(\mathtt{B}) > 0.5$
(14) $p(\mathtt{J}) > 0.5 \land p(\mathtt{F}) > 0.5 \rightarrow p(\mathtt{B}) > 0.9$

We explain these constraints as follows: (1) If B is believed, and C and D are disbelieved, then A is believed; (2) This refines above so if B is strongly believed, then A is strongly believed; (3) This refines above so if B is very strongly believed, then A is very strongly believed; (4) If C is very strongly believed, then A is strongly disbelieved; (5) If D is not believed, then A is not strongly disbelieved; (6) If D is strongly believed, then A is not strongly believed; (7) If E is strongly believed, then B is disbelieved; (8) If E is not believed, and F is believed, then B is believed; (9) If G is believed, then C is disbelieved; (10) If H is believed, then D is disbelieved; (11) If I is strongly believed, then D is not believed; (12) If J is believed, then E is disbelieved; (13) If J is believed, then B is believed; And (14) If J is believed, and F is believed, then B is strongly believed;

We can use these constraints together with any specific observations we have about an individual (perhaps a lapsed patient at a dental surgery) to predict the belief in the persuasion goal (i.e. argument A)). For instance, if we know that a given individual strongly believes F and G, e.g. $p(\mathtt{F}) = 0.8$ and $p(\mathtt{G}) = 0.8$, then we can infer that C is disbelieved (i.e. $p(\mathtt{C}) < 0.5$). However, it is not possible to infer whether the individual believes or disbelieves the persuasion goal.

Next, we could consider presenting an argument to the individual in order to see whether (according to the epistemic graph) the persuasion goal is believed or even strongly believed. For instance, if we present H and J, we may assume that the patient believes the arguments (i.e. $p(\mathtt{H}) > 0.5$ and $p(\mathtt{J}) > 0.5$). This assumption could be based on analyzing the crowdsourced data to see which arguments are believed after being presented. Then from $p(\mathtt{H}) > 0.5$ and $p(\mathtt{J}) > 0.5$, together with the original information about

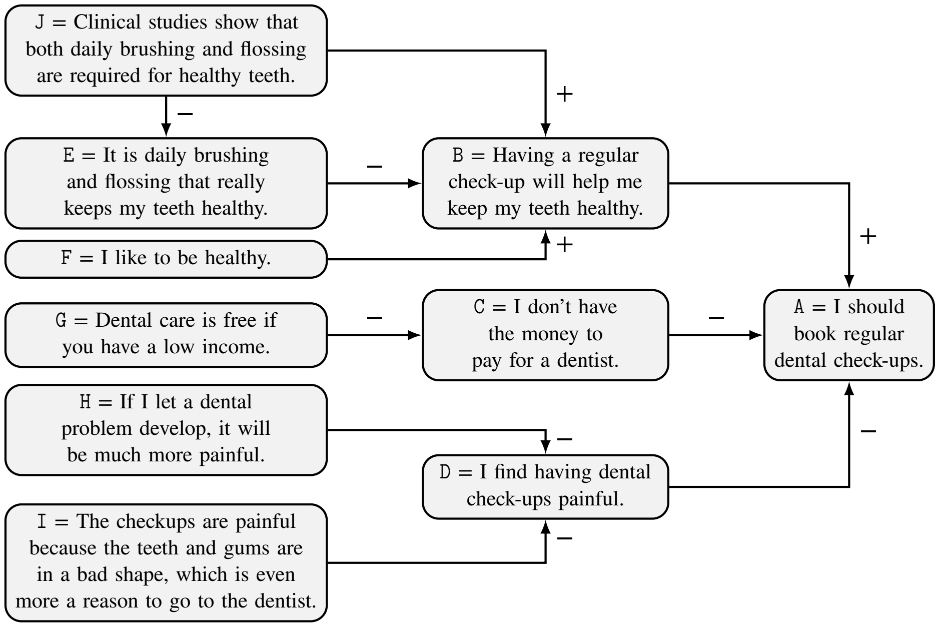
J = Clinical studies show that both daily brushing and flossing are required for healthy teeth.
E = It is daily brushing and flossing that really keeps my teeth healthy.
F = I like to be healthy.
G = Dental care is free if you have a low income.
H = If I let a dental problem develop, it will be much more painful.
I = The checkups are painful because the teeth and gums are in a bad shape, which is even more a reason to go to the dentist.
B = Having a regular check-up will help me keep my teeth healthy.
C = I don't have the money to pay for a dentist.
D = I find having dental check-ups painful.
A = I should book regular dental check-ups.
+
−

Figure 4. Epistemic graph (adapted from [1]) for the domain model for a case study on encouraging people to take regular dental check-ups.

the patient (i.e. $p(\mathtt{F}) = 0.8$ and $p(\mathtt{G}) = 0.8$), we can infer B and A are very strongly believed (i.e. $p(\mathtt{B}) > 0.9$ and $p(\mathtt{A}) > 0.9$).

Since it is possible to acquire substantial amounts of crowdsourced data, and apply machine learning to generate constraints [20], we can easily acquire large numbers of constraints on a topic that can be harnessed for user models in automated persuasion. The above example only involved 14 constraints, and so the inferences can be made by hand, but if we have 100s of constraints (which can easily arise if we have an argument graph with 100 arguments), then we need automated reasoning such as the approach presented in this paper (which was shown in the previous section to scale to 100s of clauses with 200 arguments) to be able to identify the implications of specific options for presenting arguments.

6. Discussion

Epistemic graphs offer a rich and flexible formalism for modelling argumentation. The approach provides subjective reasoning by allowing different agents to be modelled by a different set of constraints (which can be useful in complex problem analysis where different perspectives and the associated unncertainty is captured). This may be useful for modelling how different decision makers make their decisions based on their beliefs in the relevant arguments by each presenting an epistemic graph. Epistemic graphs also allow for better modelling of imperfect agents, which can be important in multi–agent application with dialogical argumentation (e.g. persuasion, negotiation, etc.).

The benefit of the work presented in this paper is that we can use the automated reasoning system to allow us to draw inferences about a situation modelled by an epistemic graph, or about what inferences another agent would draw based on what we assume about their epistemic graph. Off-the-shelf SAT solvers (which are available for a range of programming languages) allow the reasoning to scale to large epistemic graphs, and this allows us to deal with much larger numbers of arguments than possible with previous proposals for automated reasoning with epistemic graphs [15,16]. The approach of using the completion clauses can be adapted to a range of automated reasoning tasks. We will explore these in future work. We will also consider generalizing the algorithms to handle the general version of epistemic graphs that was presented in [1].

References

[1] Hunter A, Polberg S, Thimm M. Epistemic graphs for representing and reasoning with positive and negative influences of arguments. Artificial Intelligence. 2020;281:103236.

[2] Amgoud L, Ben-Naim J. Ranking-Based Semantics for Argumentation Frameworks. In: Proceedings of SUM'13. vol. 8078 of LNCS. Springer; 2013. p. 134-47.

[3] Amgoud L, Ben-Naim J, Doder D, Vesic S. Acceptability Semantics for Weighted Argumentation Frameworks. In: Proceedings of IJCAI'17. IJCAI; 2017. p. 56-62.

[4] Bonzon E, Delobelle J, Konieczny S, Maudet N. A Comparative Study of Ranking-Based Semantics for Abstract Argumentation. In: Proceedings of AAAI'16. AAAI Press; 2016. p. 914-20.

[5] Cayrol C, Lagasquie-Schiex M. Graduality in Argumentation. Journal of Artificial Intelligence Research. 2005;23:245-97.

[6] Leite J, Martins J. Social Abstract Argumentation. In: Proceedings of IJCAI'11. AAAI Press; 2011. p. 2287-92.

[7] Rago A, Toni F, Aurisicchio M, Baroni P. Discontinuity-Free Decision Support with Quantitative Argumentation Debates. In: Proceedings of KR'16. AAAI Press; 2016. p. 63-73.

[8] da Costa Pereira C, Tettamanzi A, Villata S. Changing One's Mind: Erase or Rewind? Possibilistic Belief Revision with Fuzzy Argumentation Based on Trust. In: Proceedings of IJCAI'11. AAAI Press; 2011. p. 164-71.

[9] Baroni P, Romano M, Toni F, Aurisicchio M, Bertanza G. Automatic evaluation of design alternatives with quantitative argumentation. Argument & Computation. 2015;6(1):24-49.

[10] Potyka N. Continuous Dynamical Systems for Weighted Bipolar Argumentation. In: Proceedings of KR'18. AAAI Press; 2018. p. 148-57.

[11] Pu F, Luo J, Zhang Y, Luo G. Argument Ranking with Categoriser Function. In: Proceedings of KSEM'14. vol. 8793 of LNCS. Springer; 2014. p. 290-301.

[12] Pu F, Luo J, Zhang Y, Luo G. Attacker and Defender Counting Approach for Abstract Argumentation. In: Proceedings of CogSci'15. cognitivesciencesociety.org; 2015. p. 1.

[13] Brewka G, Woltran S. Abstract Dialectical Frameworks. In: Proceedings of KR'10. AAAI Press; 2010. p. 102-11.

[14] Brewka G, Strass H, Wallner J, Woltran S. Weighted Abstract Dialectical Frameworks. In: Proceedings of AAAI'18. AAAI Press; 2018. p. 1779-86.

[15] Hunter A, Polberg S. A Model-Based Theorem Prover for Epistemic Graphs for Argumentation. In: Proceedings of ECSQARU'19. vol. 11726 of LNCSe. Springer; 2019. p. 50-61.

[16] Hunter A, Polberg S, Potyka N. Updating Belief in Arguments in Epistemic Graphs. In: Proceedings of KR'18. AAAI Press; 2018. p. 138-47.

[17] Vizel Y, Weissenbacher G, Malik S. Boolean Satisfiability Solvers and Their Applications in Model Checking. Proceedings of the IEEE. 2015;103(11):2021-35.

[18] Coste-Marquis S, Devred C, Marquis P. Constrained Argumentation Frameworks. In: Proceedings of KR'06. AAAI Press; 2006. p. 112-22.

[19] Ignatiev A, Morgado A, Marques-Silva J. PySAT: A Python Toolkit for Prototyping with SAT Oracles. In: Proceedings of SAT'18. vol. 10929 of LNCS. Springer; 2018. p. 428-37.

[20] Hunter A. Learning Constraints for the Epistemic Graphs Approach to Argumentation. In: Proceedings of COMMA'20. vol. 326. IOS Press; 2020. p. 239-50.