Automated planning of simple persuasion dialogues

Elizabeth Black, Amanda Coles and Sara Bernardini

Department of Informatics, King's College London, UK
firstname.surname@kcl.ac.uk

Abstract. We take a simple form of non-adversarial persuasion dialogue in which one participant (the *persuader*) aims to convince the other (the *responder*) to accept the *topic* of the dialogue by asserting sets of beliefs. The responder replies honestly to indicate whether it finds the topic to be acceptable (we make no prescription as to what formalism and semantics must be used for this, only assuming some function for determining acceptable beliefs from a logical knowledge base). Our persuader has a *model* of the responder, which assigns probabilities to sets of beliefs, representing the likelihood that each set is the responder's actual beliefs. The beliefs the persuader chooses to assert and the order in which it asserts them (i.e. its *strategy*) can impact on the success of the dialogue and the success of a particular strategy cannot generally be guaranteed (because of the uncertainty over the responder's beliefs). We define our persuasion dialogue as a *classical planning problem*, which can then be solved by an automated planner to generate a strategy that maximises the chance of success given the persuader's model of the responder; this allows us to exploit the power of existing automated planners, which have been shown to be efficient in many complex domains. We provide preliminary results that demonstrate how the efficiency of our approach scales with the number of beliefs.[1]

1 Introduction

Argument dialogues are an established agreement technology; they provide a principled way of structuring rational interactions between participants (machine or human) who argue about the validity of certain claims in order to resolve their conflicting information, competing goals, incompatible intentions or opposing views of the world [16]. Such dialogues are typically defined by the *moves* that can be made and rules to determine which moves are permissible at any point in the dialogue. Much existing work in the field focusses on defining argument dialogues that allow achievement of a particular goal; for example, to persuade the other participant to accept some belief [19] or to agree on some action to achieve a shared goal [3]. However, successful achievement of a participant's dialogue goal normally depends on the *strategy* it employs to determine which of the permissible moves to make during the dialogue; the development of effective argument dialogue strategies is thus an important area of active research [23].

We consider a simple non-adversarial persuasion dialogue in which the *persuader* asserts beliefs with the aim of convincing the *responder* to accept the dialogue *topic*.

[1] The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-09764-0_6

Success depends on the beliefs the persuader chooses to assert and the order in which it asserts them (its *strategy*); this is informed by the persuader's (uncertain) *model* of the responder (i.e. its beliefs about the responder's beliefs). Our proposal is general in that it allows for any logical formalism and semantics to be used to determine the acceptability of claims; the beliefs asserted may be logical formulas or abstract arguments.

We define the persuader's choice of beliefs to assert as a *classical planning problem*; this allows us to use an automated planner to search for an optimal strategy given the persuader's model of the responder. Our preliminary results show that a planner can find an optimal strategy for a problem where there are 8 beliefs the persuader can assert and 2^8 possible sets of responder beliefs it considers in 70.22 seconds (32.54 seconds to find the strategy, 37.68 seconds to prove it optimal). We discuss how we might adapt our encoding of the planning problem and the search strategy of the planner to improve the scalability.

2 Simple persuasion dialogues

In our simple persuasion dialogues, the persuader aims to convince the responder to accept the topic of the dialogue by asserting beliefs. We make no prescription as to which semantics the responder must use to reason about the acceptability of beliefs. We assume only a finite logical language $\mathcal{L}$ and some function for determining the set of *acceptable* claims given some knowledge base of $\mathcal{L}$.

Definition 1. *We assume a function* $\mathsf{Acceptable} : \wp(\mathcal{L}) \rightarrow \wp(\mathcal{L})$ *which, for a knowledge base* $\Phi \subseteq \mathcal{L}$*, returns the set of* **acceptable claims** *of* Φ *such that:*

$$\mathsf{Acceptable}(\Phi) = \{\alpha \in \mathcal{L} \mid \alpha \textit{ is acceptable given } \Phi \textit{ under the chosen acceptability semantics}\}$$

The examples in our paper use a simple argumentation formalism with Dung's grounded semantics [5] to determine the acceptability of beliefs (which we define later). There are, however, many formalisms and associated acceptability semantics that may be used to instantiate Definition 1, some examples are: logic-based deductive argumentation [2], abstract argumentation [5], assumption-based argumentation [6], defeasible logic programming [9], ASPIC+ [15], classical logic.

Each dialogue participant has a set of *beliefs*, which is a subset of $\mathcal{L}$. We assume some *common knowledge*, which is a subset of the intersection of the participants' beliefs and is known by the persuader to be part of the responder's beliefs. The persuader has a *model* of the responder, which is a function that assigns a probability to subsets of $\mathcal{L}$, representing how likely the persuader believes it is that the responder's beliefs are that set ([11] considers how such a model might be constructed). Our framework thus allows for the case where the persuader believes the responder has beliefs the persuader itself does not believe, but assumes that the persuader is aware of all the beliefs the responder may hold.

This proposal allows us to capture situations where the persuader is an expert who aims to convince the responder to accept a certain belief. For example, the persuader

may be a medical expert aiming to convince a patient that they ought to give up smoking, where the common knowledge contains the patient-specific information such as their age and medical history and the expert's model of the patient captures the beliefs it has about the patient's preferences and values. Based on this model, the expert must select knowledge to assert to the patient that will convince them to accept that they ought to give up smoking based on the information specific to their circumstances.

We define a *simple persuasion situation* by the persuader's and responder's beliefs, the common knowledge, the persuader's model of the responder and the topic of the dialogue. The set of *possible responder belief sets* refers to those sets of beliefs that the persuader believes may be the responder's beliefs (each of which contain the common knowledge, which is known by the persuader to be part of the responder's beliefs). We assume that the persuader's model is accurate in the sense that it assigns a non-zero probability to the responder's actual set of beliefs.

Definition 2. *A* **simple persuasion situation** *is a tuple* $\langle \Sigma^P, \Sigma^R, \Omega, \mathsf{m}, T \rangle$ *where:*

- $\Sigma^P \subseteq \mathcal{L}$ *is the* **persuader's beliefs***;*
- $\Sigma^R \subseteq \mathcal{L}$ *is the* **responder's beliefs***;*
- $\Omega \subseteq \Sigma^P \cap \Sigma^R$ *is the* **common knowledge***;*
- $\mathsf{m} : \wp(\mathcal{L}) \rightarrow [0, 1]$ *is the* **persuader's model of the responder** *such that*
  (a) $\sum_{\Phi \subseteq \mathcal{L}} \mathsf{m}(\Phi) = 1$,
  (b) for all Φ *such that* $\mathsf{m}(\Phi) > 0$, $\Omega \subseteq \Phi$, *and*
  (c) $\mathsf{m}(\Sigma^R) > 0$*;*
- $T \in \mathcal{L}$ *is the* **topic** *of the dialogue.*

The **possible responder belief sets** *given a particular model of the responder* m *is denoted* $\mathsf{PossRespBels}(\mathsf{m})$ *where:* $\mathsf{PossRespBels}(\mathsf{m}) = \{\Phi \mid \mathsf{m}(\Phi) > 0\}$.

The two participants take it in turn to make moves to one another. The persuader asserts subsets of its beliefs, not asserting beliefs it knows to be part of the common knowledge and not repeating beliefs previously asserted. After each asserting move made by the persuader, the responder replies honestly with a yes or no move, indicating whether it finds the topic of the dialogue to be acceptable given the union of its beliefs and those beliefs that the persuader has asserted thus far in the dialogue. If the responder makes a yes move, then the dialogue terminates successfully. We thus define a *well-formed simple persuasion dialogue* as follows.

Definition 3. *A* **well-formed simple persuasion dialogue** *of a simple persuasion situation* $\langle \Sigma^P, \Sigma^R, \Omega, \mathsf{m}, T \rangle$ *is a sequence of moves* $[P_1, R_1, \ldots, P_n, R_n]$ *such that:*

1. $P_1 = \{T\}$,
2. *for all* i *such that* $1 < i \leq n$*:*
   (a) $P_i \subseteq \Sigma^P \setminus \Omega$,
   (b) for all j *such that* $1 < j < i$, $P_j \cap P_i = \emptyset$*;*
3. *for all* i *such that* $1 \leq i < n$*:*
   (a) $R_i = \mathsf{no}$,
   (b) $T \notin \mathsf{Acceptable}(\Sigma^R \cup P_2 \cup \ldots \cup P_i)$*;*
4. $R_n \in \{\mathsf{yes}, \mathsf{no}\}$,

5. $R_n = \mathsf{yes}$ *iff* $T \in \mathsf{Acceptable}(\Sigma^R \cup P_2 \cup \ldots \cup P_n)$.

If $R_n = \mathsf{yes}$, *the dialogue is* **successful**. *If* $R_n = \mathsf{no}$, *the dialogue is* **unsuccessful**.

The persuader has a choice of beliefs it can assert at each point in the dialogue (determined by its *strategy*), while the responder's moves are determined by its beliefs and those asserted by the persuader. A *strategy* for the persuader is simply a sequence of non-intersecting subsets of its beliefs.

Definition 4. *A* **strategy** *for a persuader with beliefs* Σ^P*, for a dialogue with topic* T *where the common knowledge is* Ω *is a sequence* $[P_1, P_2, \ldots, P_{n-1}, P_n]$ *such that:*

1. $P_1 = \{T\}$,
2. *for all* i *such that* $1 < i \leq n$*:*
   (a) $P_i \subseteq \Sigma^P \setminus \Omega$,
   (b) *for all* j *such that* $1 < j < i$, $P_j \cap P_i = \emptyset$.

A strategy thus corresponds to a sequence of persuader moves in a simple persuasion dialogue. We give some examples in the following section.

2.1 Simple persuasion dialogue examples

To illustrate our simple persuasion dialogues, we must first specify the acceptability semantics with which we instantiate Definition 1; for this, we define an argumentation formalism to which we apply the grounded semantics of Dung [5]. The argumentation formalism we define allows us to concisely present some examples; we make no claims about the appropriateness of its properties. Recall that any semantics for determining the set of acceptable claims given some knowledge base of a logical language can be used with our proposal; this is only one such example and the argumentation formalism we present can be replaced with an established formalism (e.g. [2, 5, 6, 9, 15]).

We use a simple propositional language $\mathcal{L}$ that is constructed from a set of propositional atoms $\{a, b, c, \ldots\}$; α is a strong literal iff α is an atom or of the form $\neg\beta$ where β is an atom and $\neg$ represents strong classical negation; α is a weak literal iff α is of the form $\sim \beta$ where β is a strong literal and $\sim$ represents negation as failure; α is a wff of $\mathcal{L}$ iff α is a strong literal or α takes the form of a rule $\phi_1 \wedge \ldots \wedge \phi_n \rightarrow \psi$ where ψ is a strong literal and each $\phi_1, \ldots, \phi_n$ is either a strong or weak literal.[2]

An *argument* constructed from a knowledge base of $\mathcal{L}$ has a *support* and a *claim* such that: (1) the support is a subset of the knowledge base; (2) the claim is either a strong literal that appears as the support or is the head of a rule in the support; (3) for every rule in the support of the argument, every strong literal that appears in its body is either the head of another rule in the support or is itself a member of the support; (4) the support is consistent; and (5) the support is a minimal set satisfying (1-4).

Definition 5. *An* **argument** *constructed from a knowledge base* $\Delta \subseteq \mathcal{L}$ *is a tuple* (Γ, γ) *where* Γ *is the* **support** *and* γ *is the* **claim** *such that* γ *is a strong literal from* $\mathcal{L}$ *and:*

[2] Note that the symbols $\wedge$ and $\rightarrow$ are not being used here to represent classical conjunction or implication, but rather represent meta-relations between sets of literals.

1. $\Gamma \subseteq \Delta$;
2. *either* $\Gamma = \{\gamma\}$ *or there exists* $\phi_1 \wedge \ldots \wedge \phi_n \rightarrow \gamma \in \Gamma$;
3. *for every* $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta \in \Gamma$, *for every* $i \in \{1, \ldots, n\}$ *such that* α_i *is a strong literal, either* $\alpha_i \in \Gamma$ *or there exists* $\phi_1 \wedge \ldots \wedge \phi_m \rightarrow \alpha_i \in \Gamma$;
4. *if* $\Phi = (\{\psi \in \Gamma \mid \psi \text{ is a strong literal}\} \cup \{\beta \mid \text{there exists } \alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta \in \Gamma\})$, *then*
   (a) $\Phi \nvdash \bot$, *and*
   (b) *if there exists* $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta \in \Gamma$ *such that* $\sim \psi \in \{\alpha_1, \ldots, \alpha_n\}$, *then* $\psi \notin \Phi$;
5. Γ *is minimal under set inclusion.*

We denote **the set of all arguments that can be constructed from** Δ *as* $\mathsf{Args}(\Delta)$.

An argument $A1$ *attacks* an argument $A2$ if and only if either: the claim of $A1$ is the negation of the claim of $A2$, the claim of $A1$ is something that appears as a weak literal in the support of $A2$, or the claim of $A1$ is the negation of something that appears in the body of a rule that is part of the support of $A2$.

Definition 6. *An argument* (Γ_1, γ_1) **attacks** *an argument* (Γ_2, γ_2) *iff either:*

- $\gamma_1 = \neg\gamma_2$,
- *there exists a rule* $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta \in \Gamma_2$ *such that* $\sim \gamma_1 \in \{\alpha_1, \ldots, \alpha_n\}$, *or*
- *there exists a rule* $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta \in \Gamma_2$ *such that* $\neg\gamma_1 \in \{\alpha_1, \ldots, \alpha_n\}$.

The *argument framework* of a particular knowledge base represents the set of all arguments that can be constructed and the attack relations between those arguments.

Definition 7. *The* **argument framework** *of a knowledge base* $\Delta \subseteq \mathcal{L}$, *denoted* $\mathsf{AF}(\Delta)$, *is the tuple* $(\mathcal{A}, \mathcal{R})$ *where:*

1. $\mathcal{A} = \mathsf{Args}(\Delta)$,
2. $\mathcal{R} = \{(A_1, A_2) \mid A_1, A_2 \in \mathcal{A} \text{ and } A_1 \text{ attacks } A_2\}$.

We apply the grounded semantics [5] to determine the acceptable claims of an argument framework. To define the grounded semantics, we follow Caminada's labelling approach [4], which assigns exactly one label from $\{\texttt{in}, \texttt{out}, \texttt{undecided}\}$ to each argument in an argument graph such that the *reinstatement labelling* conditions given in the definition below hold. The *grounded labelling* is the unique labelling that meets the reinstatement labelling conditions and minimises the number of arguments labelled as `in`, which are those arguments that are acceptable under the grounded semantics.

Definition 8. *Let* $(\mathcal{A}, \mathcal{R})$ *be an argument framework. A* **reinstatement labelling** *of* $(\mathcal{A}, \mathcal{R})$ *is an assignment of exactly one label from* $\{\texttt{in}, \texttt{out}, \texttt{undecided}\}$ *to each of the arguments in* $\mathcal{A}$ *such that the following conditions hold.*

1. *An argument is labelled as* `in` *iff every argument that attacks it is labelled as* `out`.
2. *An argument is labelled as* `out` *iff there is no argument that attacks it and is labelled as* `in`.

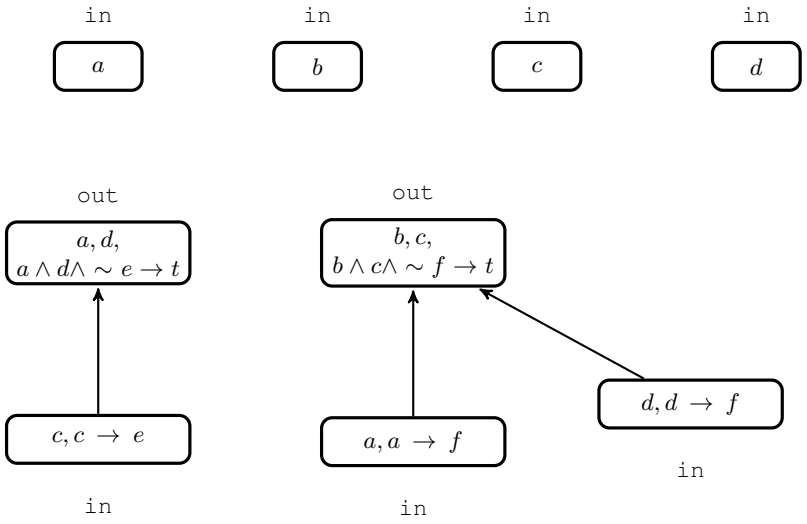

Fig. 1. The argument framework constructed from the knowledge base given in Example 1. For brevity, nodes are labelled only with the support of the argument they correspond to. The directed edges represent the attacks between arguments and nodes are annotated with the grounded labelling.

The **grounded labelling** *of an argument framework* $(\mathcal{A}, \mathcal{R})$ *is the reinstatement labelling of* $(\mathcal{A}, \mathcal{R})$ *that minimises the number of arguments labelled as* `in`*.*

We can now define the acceptable claims of a particular knowledge base Δ as those that appear as the claim of an argument that can be constructed from Δ and are labelled as `in` in the grounded labelling of the argument framework constructed from Δ.

Definition 9. *Let* $\Delta \subseteq \mathcal{L}$ *be a knowledge base such that* $\mathsf{AF}(\Delta) = (\mathcal{A}, \mathcal{R})$*. A wff* $\alpha \in \mathcal{L}$ *is* **acceptable given** Δ *iff there exists an argument* $(\Phi, \alpha) \in \mathcal{A}$ *such that* (Φ, α) *is labelled* `in` *in the grounded labelling of* $(\mathcal{A}, \mathcal{R})$*.*

Example 1. Consider the knowledge base $\Delta = \{a, b, c, d, a \wedge d\wedge \sim e \rightarrow t, b \wedge c\wedge \sim f \rightarrow t, c \rightarrow e, a \rightarrow f, d \rightarrow f\}$. The argument framework $\mathsf{AF}(\Delta)$ is shown in Figure 1. Thus we see that $\mathsf{Acceptable}(\Delta) = \{a, b, c, d, e, f\}$.

Now we have defined a mechanism for determining the acceptable claims from some knowledge base of $\mathcal{L}$, we present some examples of well-formed persuasion dialogues where Definition 1 is instantiated with the definition of acceptable claims from Definition 9.

Example 2. Consider a persuader with beliefs Σ^P, common knowledge Ω, model of the responder m, for a dialogue with topic t where:

- $\Sigma^P = \{a, b, c\} \cup \Omega$;

- $\Omega = \{a \wedge d \wedge \sim e \rightarrow t, b \wedge c \wedge \sim f \rightarrow t, c \rightarrow e, a \rightarrow f, d \rightarrow f\}$;
- $\mathsf{m}(\{c\} \cup \Omega) = \mathsf{m}(\{d\} \cup \Omega) = 0.4$, $\mathsf{m}(\{a, c\} \cup \Omega) = 0.2$ and for all other $\Phi \subseteq \mathcal{L}$, $\mathsf{m}(\Phi) = 0$.

There are three possible responder belief sets to consider: $\{c\} \cup \Omega$, $\{d\} \cup \Omega$, $\{a, c\} \cup \Omega$.

- If $\Sigma^R = \{c\} \cup \Omega$, some examples of well-formed simple persuasion dialogues are: $D1 = [\{t\}, \mathsf{no}, \{a, b\}, \mathsf{no}]$; $D2 = [\{t\}, \mathsf{no}, \{b\}, \mathsf{yes}]$.
- If $\Sigma^R = \{d\} \cup \Omega$, some examples of well-formed simple persuasion dialogues are: $D3 = [\{t\}, \mathsf{no}, \{a, b\}, \mathsf{yes}]$; $D4 = [\{t\}, \mathsf{no}, \{b\}, \mathsf{no}, \{a\}, \mathsf{yes}]$.
- If $\Sigma^R = \{a, c\} \cup \Omega$, some examples of well-formed simple persuasion dialogues are: $D1 = [\{t\}, \mathsf{no}, \{a, b\}, \mathsf{no}]$; $D5 = [\{t\}, \mathsf{no}, \{b\}, \mathsf{no}, \{a\}, \mathsf{no}]$.

We consider two corresponding strategies for this persuader: $S1 = [\{t\}, \{a, b\}]$; $S2 = [\{t\}, \{b\}, \{a\}]$.

- If the persuader follows strategy $S1$ and $\Sigma^R = \{c\} \cup \Omega$ or $\Sigma^R = \{a, c\} \cup \Omega$, dialogue $D1$ will result; if $\Sigma^R = \{d\} \cup \Omega$, dialogue $D3$ will result.
- If the persuader follows strategy $S2$ and $\Sigma^R = \{c\} \cup \Omega$, dialogue $D2$ will result; if $\Sigma^R = \{d\} \cup \Omega$, dialogue $D4$ will result; if $\Sigma^R = \{a, c\} \cup \Omega$, dialogue $D5$ will result.

If the persuader in Example 2 follows strategy $S1$, it will be successful if the responder's beliefs are $\{c\} \cup \Omega$ but not if they are $\{d\} \cup \Omega$ or $\{c, d\} \cup \Omega$. If the persuader follows strategy $S2$, it will be successful if the responder's beliefs are either $\{d\} \cup \Omega$ (as in dialogue $D4$) or $\{c\} \cup \Omega$ (in which case the responder would terminate the dialogue successfully after the persuader moves $\{b\}$, as in dialogue $D2$). Thus the persuader should prefer $S2$ over $S1$.

We see then that there may be multiple possible responder belief sets given which a particular strategy will lead to success. We define the *probability of success* of a strategy (given the persuader's model of the responder) as the sum of the probabilities assigned by the persuader's model to each possible responder belief set under which that strategy leads to success.

Definition 10. *The* **probability of success** *of the strategy* $[P_1, P_2, \ldots, P_{n-1}, P_n]$ *for a persuader whose model of the responder is* m*, for a dialogue with topic* T *and common knowledge* Ω *is* $\sum_{\Psi \in \Phi} \mathsf{m}(\Psi)$ *where:*

$$\Phi = \{\Sigma^R \mid \textit{there exists } m \textit{ such that } 1 \leq m \leq n \textit{ and } [P_1, \mathsf{no}, P_2, \mathsf{no}, \ldots, P_{m-1}, \mathsf{no}, P_m, \mathsf{yes}] \textit{ is a well-formed simple persuasion dialogue of } \langle \Sigma^P, \Sigma^R, \Omega, \mathsf{m}, T \rangle\}$$

Example 3. Continuing Example 2, the probability of success of $S1$ is 0.4, the probability of success of $S2$ is 0.8.

An *optimal strategy* for a persuader (given its model of the responder) is one that maximises the probability of success. In the next section we show how we can represent the persuader's choice of moves to make in a simple persuasion dialogue as a classical planning problem so that we can use an automated planner to search the space of possible strategies to find one that maximises the probability of success.

3 Representing simple persuasion dialogues as a planning problem

In this section we describe how the simple persuasion dialogue can be modelled as a classical planning problem, which can then be solved by an automated planner to generate a strategy for the persuader. A classical planning problem consists of four components: an initial state, I, describing the current state of the world; a desired goal condition, G; an optimisation metric M; and the set of actions, A, which determine the state transitions that can be made. Each action in A has preconditions, which must be true in a state S for it to be applicable, and effects that occur when it is applied allowing the generation of a new state S'. A solution to a planning problem is a *plan*: an ordered sequence of actions (each of which is applicable in sequence) that transforms I into a state that satisfies G.

We formally define our model of the planning problem later, but first we discuss some high-level issues. The major challenge in representing our persuasion dialogue as a classical planning problem is that the initial state is not known: that is, the persuader does not know which of the possible responder belief sets hold. We might desire a plan that will convince the responder regardless of which of the possible belief sets it holds; in the planning literature such a plan is known as a *conformant plan*. Many approaches to solving conformant planning problems have been proposed, the most closely related to our work is that of compiling conformant planning into classical planning and then using a classical planner to solve the problem [1].

Whilst conformant planning is sufficient in the case where there exists a sequence of actions that will achieve the goal no matter which of the possible initial states actually hold, this is not always the case in our simple persuasion dialogues (as we see in Example 2 and later in our experimental analysis). Instead what we seek is the plan that maximises the probability of success. In a sense we are seeking the 'most conformant' plan, the one that is most likely to result in the responder being convinced, but accept that it is not possible to guarantee success. This problem is related to that considered in [22]; whilst [22] considers a general solution to this type of planning problem, our compilation is made more efficient by exploiting particular properties of this problem, specifically that we do not need to consider possible initial states any further once the responder is convinced from these states.

3.1 Overview of model

To aid in understanding the formal model of our planning problem we give a brief overview here, making use of a small example. The actions that the persuader can perform are to `assert` a belief or to pass the turn to the responder (we refer to this action as `responder-turn`). We allow the persuader to make multiple assertions before the responder makes a response; this is simply a way of modelling the fact that the persuader can assert multiple beliefs simultaneously. Since the first move the persuader must make is fixed (it must assert the topic) a plan to determine the persuader's strategy always starts with a `responder-turn` (this action must be included since the persuader does not know what move the responder will make); similarly a plan must always end with a `responder-turn`.

Example 4. Following from Example 2, the strategy $S1$ is captured by the plan

(`responder-turn`) (`assert a`) (`assert b`) (`responder-turn`)

and the strategy $S2$ is captured by the plan

(`responder-turn`) (`assert b`) (`responder-turn`) (`assert a`) (`responder-turn`)

At each state in a simple persuasion plan (i.e. after each action is made), we can consider the formulas the responder is reasoning with to determine whether it finds the topic acceptable, i.e. the union of its beliefs and the beliefs asserted so far by the persuader; we refer to this set as the **responder's reasoning set**. Although the persuader does not know what this set is in a particular state, since we assume that the responder's beliefs are one of the possible responder belief sets, the persuader does know that the responder's reasoning set is a member of the following set (where m is the persuader's model of the responder): $\{\Phi \cup \Psi \mid \Phi$ are the beliefs asserted so far and $\Psi \in \mathsf{PossRespBels}(\mathsf{m})\}$.

The general idea is to monitor in each state during planning, given the beliefs that have been asserted so far and the persuader's model of the responder: (a) the probability that the responder's reasoning set is each of the possible sets that may occur (i.e. $\{\Phi \cup \Psi \mid \Phi \subseteq \Sigma^P$ and $\Psi \in \mathsf{PossRespBels}(\mathsf{m})\}$), and (b) the probability that the responder will terminate the dialogue successfully either in the current state or some previous state of the plan (i.e. the probability of success of the plan).

Table 1 shows how the two strategies discussed in Example 2 are evaluated by our approach. We see that there are 12 possible sets that may occur as the responder's reasoning set. Let us consider strategy $S1$. In the initial state, the responder's reasoning set is simply its beliefs (as the persuader has not yet asserted anything) and so (according to the persuader's model of the responder) there is a 0.4 probability that the responder's reasoning set is $\{c\} \cup \Omega$, a 0.4 probability that it is $\{d\} \cup \Omega$, and a 0.2 probability that it is $\{a, c\} \cup \Omega$. Since none of these sets cause the responder to find the topic acceptable (and so the responder is sure to make a `no` move), after the first (`responder-turn`) action these probabilities stay the same.

After the (`assert a`) action, if the responder's reasoning set had been $\{c\} \cup \Omega$ or $\{a, c\} \cup \Omega$ in the previous state it would now be $\{a, c\} \cup \Omega$, thus the probability assigned to $\{c\} \cup \Omega$ is now 0 and the probability assigned to $\{a, c\} \cup \Omega$ is now 0.6. If the responder's reasoning set had been $\{d\} \cup \Omega$ in the previous state it would now be $\{a, d\} \cup \Omega$, thus the probability assigned to $\{d\} \cup \Omega$ is now 0 and the probability assigned to $\{a, d\} \cup \Omega$ is 0.4.

After the (`assert b`) action, if the responder's reasoning set had been $\{a, c\} \cup \Omega$ in the previous state it would now be $\{a, b, c\} \cup \Omega$, thus the probability assigned to $\{a, c\} \cup \Omega$ is now 0 and the probability assigned to $\{a, b, c\} \cup \Omega$ is now 0.6. If the responder's reasoning set had been $\{a, d\} \cup \Omega$ in the previous state it would now be $\{a, b, d\} \cup \Omega$, thus the probability assigned to $\{a, d\} \cup \Omega$ is now 0 and the probability assigned to $\{a, b, d\} \cup \Omega$ is 0.4.

If the responder's reasoning set is $\{a, b, d\} \cup \Omega$, then it will find the topic acceptable and terminate the dialogue successfully, thus we increase the probability of success in the state after the final (`responder-turn`) action by 0.4 (the probability that the

	Probability responder's reasoning set is:												Probability of success
	$\{c\} \cup \Omega$	$\{p\} \cup \Omega$	$\{a, c\} \cup \Omega$	$\{a, d\} \cup \Omega$	$\{b, c\} \cup \Omega$	$\{b, d\} \cup \Omega$	$\{a, b, c\} \cup \Omega$	$\{c, d\} \cup \Omega$	$\{a, b, d\} \cup \Omega$	$\{a, c, d\} \cup \Omega$	$\{b, c, d\} \cup \Omega$	$\{a, b, c, d\} \cup \Omega$	
Strategy $S1$:													
Initial state	0.4	0.4	0.2	0	0	0	0	0	0	0	0	0	0
(`responder-turn`)	0.4	0.4	0.2	0	0	0	0	0	0	0	0	0	0
(`assert a`)	0	0	0.6	0.4	0	0	0	0	0	0	0	0	0
(`assert b`)	0	0	0	0	0	0	0.6	0	0.4	0	0	0	0
(`responder-turn`)	0	0	0	0	0	0	0.6	0	0	0	0	0	0.4
Strategy $S2$:													
Initial state	0.4	0.4	0.2	0	0	0	0	0	0	0	0	0	0
(`responder-turn`)	0.4	0.4	0.2	0	0	0	0	0	0	0	0	0	0
(`assert b`)	0	0	0	0	0.4	0.4	0.2	0	0	0	0	0	0
(`responder-turn`)	0	0	0	0	0	0.4	0.2	0	0	0	0	0	0.4
(`assert a`)	0	0	0	0	0	0	0.2	0	0.4	0	0	0	0.4
(`responder-turn`)	0	0	0	0	0	0	0.2	0	0	0	0	0	0.8

Table 1. For Example 2, at each state during planning, shows the updates to: (a) the probabilities assigned to the sets that may occur as the responder's reasoning set and (b) the probability of success.

responder's reasoning set in the previous state was $\{a, b, d\} \cup \Omega$) and set the probability assigned to $\{a, b, d\} \cup \Omega$ to 0 (as, if the plan were to continue after this point, the responder could no longer be reasoning with this set, as if it had been it would have ended the dialogue successfully). The probability of success of the plan that captures Strategy $S1$ is thus 0.4.

The effect of an `asssert` action is to update the probabilities assigned to the possible responder's reasoning sets (i.e. those with a non-zero probability), as a consequence of the asserted belief being added to these sets. The effect of a `responder-turn` move is to update the probability of success assigned to the plan when a possible responder reasoning set would cause the topic to be acceptable, and to assign the probability associated with that set to zero. In the following section we formally define the conditions and effects of these two actions and specify how a planning problem instance is constructed from a simple persuasion situation.

3.2 Formal model of the simple persuasion planning problem

To represent our simple persuasion dialogues as a planning problem, we use PDDL2.1 [8], a standard language for encoding the information required by automated planners (i.e. the actions that can be performed, the initial situation, the goal and the optimisation metric). PDDL2.1 allows the use of typed objects, predicates and functions to define the preconditions and effects of actions. We define two types of object: the `wff` type

(`can-assert` ?`w` − `wff`)	True if ?`w` is a persuader's belief that has not yet been asserted
(`acceptable` ?`sow` − `setOfWffs`)	True if the topic is acceptable given ?`sow`
(`add` ?`w` − `wff` ?`sow1` ?`sow2` − `setOfWffs`)	True if the adding ?`w` to the set ?`sow1` gives the new set ?`sow2`
(`belief-asserted`)	Flag to ensure that the persuader asserts at least one belief each turn
(`initial-move`)	Flag to ensure the first action of a plan is a (`responder-turn`)
(`responder-moved`)	Flag to ensure the last action of a plan is a (`responder-turn`)
(`prob-resp-reasoning-with` ?`sow` − `setOfWffs`)	Function that assigns probabilities to the sets that may occur as the responder's reasoning set
(`prob-of-success`)	Function that updates the probability of success of a plan

Table 2. Predicates and functions used to define the actions `assert` and `responder-turn`.

is used to capture wff of $\mathcal{L}$; the `setOfWffs` type is used to capture sets of wff of $\mathcal{L}$. The predicates and functions used to define our actions are given in Table 2 (variables in PDDL2.1 begin with a ? and are annotated with their type). Recall that we take the initial state to be where the persuader has opened the dialogue by asserting the topic (as all simple persuasion dialogues start in this manner), thus a plan must always start with a (`responder-move`) action. A plan must also always end with a (`responder-move`) action, to allow the persuader to consider the responder's possible responses.

We now define how a planning problem instance is constructed for a simple persuasion situation; this determines the initial state that the planner must plan from.

Definition 11. *For a persuader with beliefs Σ^P, for a dialogue with common knowledge Ω, topic T, where its model of the responder is* m *we construct a* **planning problem instance** *as follows:*

- *for every belief $\alpha \in \Sigma^P$, there is an equivalent* `wff` *object: α;*
- *for every set of beliefs $\Upsilon \in \{\Phi \cup \Psi \mid \Phi \subseteq \Sigma^P$ and $\Psi \in \mathsf{PossRespBelSets}(\mathsf{m})\}$ (i.e. for every set that may occur as the responder's reasoning set), there is an equivalent* `setOfWffs` *object: Υ;*
- *for all $\alpha \in \Sigma^P$, (`can-assert` α) is true ;*
- *(`acceptable` Φ) is true if and only if $T \in \mathsf{Acceptable}(\Phi)$;*
- (`belief-asserted`) *is true (since we assume we are in the state where the persuader has asserted the topic);*
- (`initial-move`) *is true;*
- (`responder-moved`) *is not true;*
- *for all $\Upsilon \in \{\Phi \cup \Psi \mid \Phi \subseteq \Sigma^P$ and $\Psi \in \mathsf{PossRespBelSets}(\mathsf{m})\}$:*
  - *if $\Upsilon \in \mathsf{PossRespBelSets}(\mathsf{m})$,*
    (= (`prob-resp-reasoning-with` Υ) x) where $x = \mathsf{m}(\Upsilon)$;

    - *otherwise*
      $(=$ (prob-resp-reasoning-with Υ) $0)$*;*
- $(=$ (prob-of-success) $0)$ *;*
- *the goal is* (responder-moved)*;*
- *the optimisation metric is to maximise* (prob-of-success).

The PDDL2.1 definition of our actions is shown in Figure 2. The persuader can assert a wff object ?w as long as it can assert ?w (i.e. it has not asserted it already) and it is not the initial move. The main effect of an asssert action is to update the probabilities assigned to the sets that can occur as the responder's reasoning set. For each set ?sow1 that was assigned a non-zero probability p in the previous state, asserting the belief ?w has the effect of setting the probability assigned to ?sow1 to zero and increasing the probability assigned to ?sow1 $\cup \{$?w$\}$ (i.e. ?sow2) by p (so that if ?w is already a member of ?sow1, the probability assigned to ?sow1 does not change).

A responder-turn action can be made as long as a belief has been asserted (so two or more responder-turn actions cannot occur in sequence). The main effect of a responder-turn action is to update the probability of success assigned to the plan. For each of the possible responder's reasoning sets ?sow that was assigned a non-zero probability p in the previous state, if that set causes the responder to find the topic acceptable, responder-turn has the effect of increasing the probability of success by p and setting the probability assigned to ?sow to zero.

Given this formal model of our planning problem, a planner can search for a plan that maximises the probability of success. In the following section we present some preliminary results that explore the efficiency of the automated planning process.

3.3 Experimental results

We used the planner Metric-FF [12, 13] to generate plans for our model. The combination of features required by the model means that this is the most appropriate planner: standard ('STRIPS') planners support only conjunctions in preconditions, whereas we require quantification and conditional effects[3]. We also require numeric fluents, the values of prob-resp-reasoning-with, and optimisation based on a metric function. In fact, because the updates to the values of prob-resp-reasoning-with and prob-of-success are state-dependent, Metric-FF is not able to optimise our problem off-the-shelf; however, we implemented a simple wrapper which allows us to use Metric-FF to perform the optimisation as follows[4]. The wrapper first calls Metric-FF to solve the problem with the goal:

```
(and (responder-moved)
     (> (prob-of-success) 0)
)
```

[3] In theory it is possible to compile these away to create a STRIPS representation, with the possibility of using different planners, we leave this to future work.

[4] This wrapper, our formal model of the planning problem and some example problem instances can be downloaded from http://www.inf.kcl.ac.uk/staff/lizblack/automated-planning-simple-persuasion.html.

```
  (:action assert
    :parameters (?w - wff)
    :precondition (and (not (initial-move))
                       (can-assert ?w))
    :effect (and (belief-asserted)
                 (not (responder-moved))
                 (not (can-assert ?w))
                 (forall (?sow1 ?sow2 - setOfWffs)
                         (when (and (> (prob-resp-reasoning-with ?sow1) 0)
                                    (add ?w ?sow1 ?sow2))
                               (and (increase (prob-resp-reasoning-with ?sow2)
                                              (prob-resp-reasoning-with ?sow1))
                                    (assign (prob-resp-reasoning-with ?sow1) 0))
)          )    )       )

(:action responder-turn
  :parameters ()
  :precondition (belief-asserted)
  :effect (and (not (belief-asserted))
               (not (initial-move))
               (responder-moved)
               (forall (?sow - setOfWffs)
                       (when (and (> (prob-resp-reasoning-with ?sow) 0)
                                  (acceptable ?sow))
                             (and (increase (prob-of-success)
                                            (prob-resp-reasoning-with ?sow))
                                  (assign (prob-resp-reasoning-with ?sow) 0)))
)          )    )     )
```

Fig. 2. The PDDL2.1 definition of the two actions used to model simple persuasion dialogues as a planning problem: `assert` and `responder-turn`.

If a plan exists to solve the problem (i.e. if there is a strategy that has a greater than 0 probability of success), Metric-FF will return some such plan. The wrapper then updates the problem instance so that the goal is now

```
(and (responder-moved)
     (> (prob-of-success) X)
)
```

where `X` is the probability of success of the plan returned by Metric-FF in the previous step, and calls Metric-FF to solve the problem again with the new goal. This process continues until the planner reports that the problem is unsolvable with a probability of success higher than the last plan found; we know therefore that the previous plan found by the planner is optimal.

For our experiments we generated problem instances with $\#\texttt{beliefs}$ possible beliefs that the persuader can assert. The possible responder belief sets are all the elements of the power set of the persuader's beliefs, to which we have assigned equal probability (so we assume the persuader believes that the responder's beliefs are some subset of its own but has no *a priori* beliefs about which is more likely and is not aware of any common knowledge). Each possible set of beliefs was randomly determined to make the topic acceptable with probability θ. Our first set of experiments scaled $\#\texttt{beliefs}$, while keeping $\theta = 0.3$. The second set of experiments set $\#\texttt{beliefs}$ to 8 and varied θ. For each parameter setting we generated a single problem instance and recorded: the time taken to find the optimal plan, the cumulative time taken to find the optimal plan *and* prove it optimal, the number of runs of the planner required to find the optimal plan and prove it optimal, the probability of success of the optimal plan. All experiments were run on a 3GHz machine with a memory limit of 27GB. Results of both experiments are shown in Table 3.

Note that what we are evaluating here is the time taken to find an optimal plan, not the time taken to generate the problem instance. Generating the problem instance is not trivial; in particular, determining the sets that make the topic acceptable is typically costly (depending on the acceptability semantics chosen to instantiate Definition 1, an existing implementation such as ASPARTIX [7] could be used for this). We expect that the work done in generating the problem instance can be reused for other problem instances, and will explore this in future work.

Our experiments show that we can optimally solve problems with up to 2^9 possible responder belief sets; problems of this size are very difficult for humans to solve even close to optimally, and this shows real benefits of automation. Scalability does remain a challenge, however; in particular memory usage seems to be the bigger concern than time. The size of the problem, and the search space, grows exponentially with the number of persuader beliefs that can be asserted; the number of possible responder belief sets also grows exponentially with the number of persuader beliefs and reasoning about all of these is challenging. We plan to work on more efficient encodings that will allow significantly greater scalability. We also expect improved performance for problems where there are fewer possible responder belief sets.

Our first experiment shows that planning gets more difficult as the space of possible solutions increases, but the overall time remains reasonable for problems with up to 8

#beliefs	1	2	3	4	5	6	7	8	9			
findOptPlan	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	1.3	32.54	1220.09			
proveOpt	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	0.16	2.46	70.22	2464.46			
#runs	3	3	6	6	8	12	20	30	34			
probSucc	1	1	1	0.81	1	0.94	0.80	0.94	0.97			
%acceptable	49.6	25.8	23.1	17.6	15.6	13.3	11.3	9.4	4.3	3.5	2.7	1.2
findOptPlan	5.76	15.71	48.41	33.65	20.97	20.71	9.26	19.16	5.22	4.77	8.69	3.17
proveOpt	14.69	54.67	73.02	68.7	49.89	49.46	32.61	39.77	8.41	7.47	10.86	4.14
#runs	24	31	28	30	32	25	20	17	11	17	17	11
probSucc	1	0.85	0.72	0.82	0.79	0.80	0.76	0.75	0.24	0.23	0.23	0.26

Table 3. Shows: seconds to find the optimal plan (findOptPlan); seconds to find the optimal plan *and* prove it optimal (proveOpt); number of runs of planner to prove plan optimal (#runs; probability of success of the optimal plan (probSucc). Top part of table shows results for first experiment (where we varied the number #beliefs of persuader beliefs and the probability that a set causes the topic to be acceptable was fixed as 0.3); bottom part of table shows results for second experiment (where we varied the probability that a set causes the topic to be acceptable and the number #beliefs of persuader beliefs was fixed as 8) and gives the percentage %acceptable of the 2^8 possible responder belief sets that were determined to make the topic acceptable as we varied the probability θ that a set causes the topic to be acceptable.

persuader beliefs. In general the most time consuming step is the final run to prove the solution optimal; finding the optimal solution often takes less than half as long as proving it optimal, a future direction is to scale to larger problems by finding solutions that have a certain probability of success but are not necessarily optimal. The difficulty of the problem clearly depends on *which* sets cause the topic to be acceptable; this is currently assigned randomly by our problem generator, we control only the proportion of such sets. We intend to run experiments where acceptability status of sets is determined from the underlying logic, to investigate whether this improves performance.

Results of our second experiment support what might be expected. If a large percentage of the possible sets make the topic acceptable then planning is relatively quick because solutions are abundant; so, although the planner might run many times, it is relatively easy to find a solution that improves on the previous one quickly. As the number of belief sets that make the topic acceptable becomes low, it is harder to find solutions to the planning problem as there are fewer, but (because the number is so low) search space pruning is more powerful as the planner can recognise early during plan generation that a plan cannot lead to a better state and prune it without further exploring. There is somewhat of a phase transition between these two extremes, at approximately 10-25% of belief sets acceptable, where neither of these advantages prevails. Variation in the results appears because of the random assignment of acceptable sets by our problem generator, which impacts on the the difficulty of the problem.

4 Related work

Recent works on argument dialogue strategy [3, 10, 21] also use a *model* of the other participant. The dialogue of [3] allows participants to agree on some action to achieve a shared goal. The authors provide a strategy that requires a certain model of the other participant's preferences and depends on a particular argumentation formalism, whilst a strategy generated by our approach maximises the chance of success taking into account the uncertainty over the responder's beliefs and we allow for any reasoning mechanism.

Different tactics for making concessions in argumentation-based negotiation are presented in [10], which use a model of the other participant's (perhaps distinct) defeat relation over the arguments that can be used to support or attack offers; here we instead use a model of the responder's beliefs, and assume its mechanism for determining the acceptability of claims is known to the persuader. In case this mechanism is argument based, our approach can account for the construction by the responder of new arguments by combining its existing beliefs with those asserted by the persuader; this is not possible in [10], which does not consider the structure of arguments.

In [21], a variation of the minimax algorithm is used with a recursive model of the opponent to determine dialogue strategy in an adversarial abstract persuasion setting. Uncertainty over the opponent model is also allowed for in [21]. The authors present results regarding the effectiveness of their approach (i.e. whether the strategy leads to success) but do not present results regarding the efficiency of their algorithm; we consider the time taken to find a guaranteed optimal strategy (albeit in a simpler non-adversarial setting). The experiments in [21] assume 10 arguments distributed between the two agents, which is comparable to the size of problem we have shown our approach to be efficient for. Whilst we assume that the persuader is aware of all beliefs the responder may believe, [21] uses *virtual arguments* to allow the for the case where the responder has beliefs that the persuader is unaware of; this is something we will consider adopting in our model.

The application of the minimax algorithm to dialogical argumentation is also considered in [14], which proposes a general framework for specifying argument dialogue systems using propositional executable logic; this allows a finite state machine to be generated, which represents all possible dialogues from a particular initial state. Such a finite state machine can then be analysed with the minimax algorithm to determine an optimal strategy for a participant, although this requires certain knowledge of the other participant's private state. Efficiency results are also given in [14]; these are better than the results we achieve here but we are considering a set of possible initial states (i.e. the different possible responder belief sets) while [14] considers a situation in which the persuader and responder beliefs are known. It will be interesting to explore more closely the relationship between our approach and [14].

In [17] and in [18], argument-based negotiation is considered as a planning problem. Each of these proposals allow plans of arguments specific to negotiation to be generated, where the arguments that can be generated are specified by the domain; our approach is more general than this, since our domain does not depend on a particular argumentation formalism, and so could also be used to determine a persuasive line of argument within a negotiation context.

Finally, [20] also considers the generation of a persuasive line of argument as a planning problem. The focus of that work is on generating natural language discourse; it is concerned with eloquence and style of language, as well as the logical structure of arguments, whilst we consider only the acceptability of logical formulas.

5 Discussion

Our proposal allows an automated planner to find an optimal strategy for a simple persuasion dialogue; a key advantage is that it is general in the sense that it does not prescribe the logical formalism and semantics for determining acceptability of claims, allowing for both abstract and structured argumentation, as well as other non-argumentation based formalisms. Our preliminary results show that the efficiency of the planner does not scale well beyond 8 persuader beliefs; however, we expect a significant improvement when we reduce the number of possible responder belief sets considered.

The major obstacle to scalability in our current approach is that we are not exploiting any knowledge about the responder belief sets that we know are either not possible or not likely. Whilst the current model allows reasoning with these as zero probability states, it does not reduce the size of the task. We intend to explore more efficient encodings of the planning problem to allow exploitation of such knowledge, and also to consider exclusion of unlikely possible responder belief states to further improve scalability. Better exploitation of the native ability of planners to handle sets, through the explicit reasoning over beliefs as individual entities rather than as black-box sets, is also an avenue to improve performance. Finally, we intend to investigate how the search algorithms and heuristics of the planner itself can be modified to allow better performance in this particular domain, or indeed what inspiration we can gain from this problem for improving general planning strategies across different types of problems.

Our optimisation metric currently only considers the success of the dialogue. We could also consider that the persuader may have some preferences regarding the beliefs it shares with the responder. By assigning values to each belief that represent how willing the persuader is to make it known to the responder, we could adapt our optimisation metric to also take into account the beliefs the persuader has had to share.

We intend to model more complex types of argument dialogue as planning problems. In particular, we are interested in the case where both participants are making assertions with the aim of achieving their individual (and potentially conflicting) dialogue goals. The dynamic and uncertain nature of these dialogues presents interesting challenges for classical planning; nevertheless, we believe this work demonstrates the feasibility of using automated planners to generate strategies for such dialogues.

References

1. A. Albore, H. Palacios, and H. Geffner. Compiling uncertainty away in non-deterministic conformant planning. In *Proc. of 19th European Conf. on Artificial Intelligence*, pages 465–470, 2010.
2. P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128(12):203 – 235, 2001.

3. E. Black and K. Atkinson. Choosing persuasive arguments for action. In *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems*, pages 905–912, 2011.
4. M. Caminada. On the issue of reinstatement in argumentation. In *Proc. of the 10th European Conf. on Logics in Artificial Intelligence*, pages 111–123, 2006.
5. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intellegence*, 77:321–357, 1995.
6. P. M. Dung, R. A. Kowalski, and F. Toni. Dialectic proof procedures for assumption-based, admissible argumentation. *Artificial Intelligence*, 170(2):114–159, 2006.
7. U. Egly, S. A. Gaggl, and S. Woltran. ASPARTIX: Implementing argumentation frameworks using answer-set programming. In *Proc. of 24th Int. Conf. on Logic Programming*, pages 734–738, 2008.
8. M. Fox and D. Long. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *J. of Artificial Intelligence Research*, 20:61–124, 2003.
9. A. J. García and G. R. Simari. Defeasible logic programming an argumentative approach. *Theory and Practice of Logic Programming*, 4(1–2):95–138, 2004.
10. N. Hadidi, Y. Dimopoulos, and P. Moraitis. Tactics and concessions for argumentation-based negotiation. In *Proc. of the 4th Int. Conf. on Computational Models of Argument*, pages 285–296, 2012.
11. C. Hadjinikolis, Y. Siantos, S. Modgil, E. Black, and P. McBurney. Opponent modelling in persuasion dialogues. In *Proc. of the 23rd int. Joint Conf. on Artificial Intelligence*, pages 164–170, 2013.
12. J. Hoffmann. The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. *J. of Artificial Intelligence Research*, 20:291–341, 2003.
13. J. Hoffmann and B. Nebel. The FF planning system: Fast plan generation through heuristic search. *J. of Artificial Intelligence Research*, 14:253–302, 2001.
14. A. Hunter. Analysis of dialogical argumentation via finite state machines. In *Proc. of the 7th Int. Conf. on Scalable Uncertainty Management*, pages 1–14, 2013.
15. S. Modgil and H. Prakken. A general account of argumentation with preferences. *Artificial Intelligence*, 195(0):361 – 397, 2013.
16. S. Modgil, F. Toni, F. Bex, I. Bratko, C. I. C. nevar, W. Dvořák, M. A. Falappa, X. Fan, S. A. Gaggl, A. J. García, M. P. González, T. F. Gordon, J. ao Leite, M. Možina, C. Reed, G. R. Simari, S. Szeider, P. Torroni, and S. Woltran. The added value of argumentation. In S. Ossowski, editor, *Agreement Technologies*, pages 357–403. Springer Netherlands, 2013.
17. A. Monteserin and A. Amandi. Argumentation–based negotiation planning for autonomous agents. *Decision Support Systems*, 51(3):532–548, 2011.
18. A. R. Panisson, G. Farias, A. Freitas, F. Meneguzzi, R. Vieira, and R. H. Bordini. Planning interactions for agents in argumentation-based negotiation. In *Proc. of 11th Int. Workshop on Argumentation in Multi-Agent Systems*, 2014.
19. H. Prakken. Formal systems for persuasion dialogue. *The Knowledge Engineering Review*, 21(02):163–188, 2006.
20. C. Reed, D. Long, and M. Fox. An architecture for argumentative dialogue planning. In *Proc. of 1st Int. Conf on Formal and Applied Practical Reasoning*, pages 555–566, 1996.
21. T. Rienstra, M. Thimm, and N. Oren. Opponent models with uncertainty for strategic argumentation. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence*, 2013.
22. R. Taig and R. I. Brafman. Compiling conformant probabilistic planning problems into classical planning. In *Proc. of 23rd Int. Conf. on Automated Planning and Scheduling*, 2013.
23. M. Thimm. Strategic argumentation in multi-agent systems. *Künstliche Intelligenz, Special Issue on Multi-Agent Decision Making*, 2014. (In press.).