Dynamic Argument Systems: A Formal Model of Argumentation Processes Based on Situation Calculus

Gerhard Brewka, University of Leipzig
Dep. of Computer Science, Augustusplatz 10-11, 04109 Leipzig, Germany

May 25, 2000

Abstract

We present a formal model of argumentation based on situation calculus which captures both the logical and the procedural aspects of argumentation processes. The logic is used to determine what is accepted by each agent participating in the discussion and by the group as a whole, on the basis of the speech acts performed during argumentation. Argumentation protocols, also called rules of order, describe declaratively which speech acts are legal in a particular state of the argumentation. We first discuss argumentation with fixed rules of order. Our model tolerates protocol violations but makes it possible to object to illegal actions. In realistic settings the rules of order themselves can at any time become the topic of the debate. We show how meta level argumentation of this kind can be modelled in what we call dynamic argument systems. To illustrate the notions introduced in the paper we present a reconstruction of Rescher's theory of formal disputation and a dynamic argument system with three levels which we use to discuss a murder case.

1 Introduction

In this paper we present a formal model of argumentation. Argumentation here means the process by which agents try to convince other agents that certain propositions hold. This form of communication is sometimes also called critical discussion [44] or disputation [33].

Argumentation plays a central role in the communication of human and artificial agents and is an ubiquitous task in professional and every day life. Argumentation may take place in rather informal settings between two agents, but there are also highly structured forms of argumentation with different roles for the participants and predefined procedures for exchanging arguments and reaching a fair decision. Probably the best example for such a highly structured form is legal argumentation where defendant, prosecutor, judge, etc. obviously play different roles and where legal rules govern the way arguments are brought forward. The goal of the legal procedure is to guarantee that fair decisions can be reached effectively.

The study of argumentation has a long tradition in philosophy reaching back at least to Aristotle. With the breakthrough of formal logic in the 20th century semantical and proof theoretical questions became dominating issues in logic. Nevertheless, a number of philosophers like Toulmin [37], Lorenzen [18], Hamblin [12], and more recently Walton and Krabbe [44, 43], van Eemeren and Grootendorst [41] and many others have emphasized the dialectical aspects of reasoning and studied, formally or informally, various aspects of argumentation.

In recent years the formal study of argumentation became a hot topic in Artificial Intelligence, in particular in the area of multi-agent-systems and in nonmonotonic/uncertain reasoning.

The focus or much of the argumentation research in multi-agent-systems (for a few representative examples see [22, 35, 29]) was the role of argumentation for negotiation and reaching agreement. In nonmonotonic and uncertain reasoning argument systems were used to define inference systems for existing nonmonotonic logics (e.g. [10]), or to define a nonstandard (most often nonmonotonic) consequence relation for a particular logic based on some notion of argument [24, 25, 9, 15, 5, 7, 1, 27]. Although these papers differ in technical detail and the underlying formal languages all of them use notions of undercutting and defeat among arguments to define criteria for acceptability of arguments, respectively propositions supported by arguments. [28] and [6] provide excellent overviews.

The focus of this paper is different. We do not intend to propose a specific new logic, particularly well-suited for argumentation (although we present a logic which we use throughout the paper in Section 2). We doubt that one single logic of this kind exists and believe that picking the right logic heavily depends on the particular area and type of debate. In fact, our general definition of argument systems will not depend on any particular logic of disputation. Such a logic (be it nonmonotonic or not, defined in terms of arguments or not) is one of the parameters that need to be instantiated when specific argument systems are built.

Our major interest is to capture, in a single framework, both the logical and the procedural aspects of argumentation. Our formal account describes which propositions are accepted, by a particular agent or by the group as a whole, in a particular state of the argumentation. But it also formalizes the speech acts that can be performed by members of the group as well as the "rules of order" used to establish whether a speech act can legally be performed in a state of the debate. It is this combination of logic and procedure that distinguishes our approach from the work on nonmonotonic argument systems mentioned earlier and puts us in the tradition of researchers like Rescher, Loui, Gordon, Prakken, Vreeswijk and others who take not only the logical, but also the procedural character of argumentation seriously and who view argumentation as the process of jointly constructing a commonly accepted theory.

Our approach is influenced by Rescher's theory of formal disputation, as developed in his 1977 book *Dialectics: A Controversy-Oriented Approach to the Theory of Knowledge* [33]. Rescher views formal disputation as a three player game involving a proponent, an opponent, and a determiner whose role it is to decide who the winner of the debate is, based on the plausibility of arguments put forward. Rescher's account remains informal to a large extent. In [2] a logical reconstruction of Rescher's theory based on a version of Reiter's default logic [30] with specificity was given. The major goal of the present paper is to abstract from (the formal reconstruction of) Rescher's theory a general framework for argumentation. To define specific argument systems as instances of the framework one has, among other things, to fix a logic of debate, a set of speech acts, and the rules of order. All argument systems defined that way exhibit, as we believe, the spirit of Rescher style disputations.

Another important source of inspiration for the work presented here was Gordon's formalization of legal pleading [11]. In the spirit of Rescher, Gordon combines non-monotonic reasoning with game theoretic notions of argumentation. Legal pleading is conceived as a particular language game. Gordon identifies 10 rather complex moves proponent and opponent can make in this game. The applicability conditions of these moves refer to an underlying nonmonotonic logic, namely Geffner and Pearl's conditional entailment [10]. Also Gordon's pleadings game can be viewed as an instance of the framework to be developed in this paper.

Formal models of argumentation must be highly flexible and adjustable, otherwise they simply will be unable to capture important aspects of realistic argumentation processes. The following properties seem essential:

1. *Declarative representation of the argumentation protocol*
   Protocols describe which actions are allowed, respectively forbidden, in a particular state of the argumentation. It is obvious that deontic notions play an important role here (although a full fledged deontic logic may not be needed to represent a protocol). Declarative representations of the protocol not only make it easier to understand them but are a necessary condition for reasoning about and possibly changing the protocol (see 3.).

2. *Protocol violations*
   Protocols describe normative standards. People often deviate from the ideal behaviour and violate norms. Strictly enforcing normative behaviour (this is sometimes called regimentation, see for instance [16]), although possible in principle, is not an adequate way of dealing with this fact. An overly strict system runs the risk of not being accepted by potential users. What is needed is a much more flexible view where protocols weakly constrain potential courses of argumentation without strictly enforcing normative behaviour. In particular, protocol violations should be possible if nobody objects. In our approach we admit argumentation states obtained through illegal actions. However, every group member has the right to object to such actions and to "undo" their consequences.

3. *Dynamic changes of protocol*
   In realistic settings the rules of order themselves can, at any time, become the current topic of the debate. It is common, for instance, that in a trial about a certain murder case lawyers disagree as to whether making a certain claim is legal or not. In that situation a debate about the accepted rules of order may start, rules may be changed and new rules invented until agreement about the rules of order is reached. This form of meta debate can lead to a meta meta debate and so on.

   The need to formalize argumentation about the rules of argumentation has been discussed, for instance by Loui [19] who writes: "*Locutions pertain not only* *to the substance of dispute, but also to the protocol. Arguments can advance claims about how to define or alter the protocol*". Vreeswijk [42] has formalized restricted forms of meta debates of this kind. We will discuss his approach in more detail in the concluding section. Our goal is to formalize meta debates in a much more flexible way, not restricted to the first meta level.

Our formalization is based on situation calculus, one of the major AI approaches to reasoning about action and modeling dynamic systems [20, 17, 32]. More precisely, we use the dialect of situation calculus developed in Toronto by Ray Reiter and his colleagues. In particular, our representation of argumentation records - which are used to keep track of the history and current state of argumentation - corresponds to the representation of situations in situation calculus. Basically, situations are histories of actions, argumentation records histories of speech acts performed during argumentation. The effects of speech acts are described by Reiter style successor state axioms [31, 32].

Let us conclude this introduction with some remarks about the role of formalizations of the kind developed in this paper. As pointed out by Wooldridge and Jennings [45] formal theories can at least serve two purposes: they can be contributions to *formal philosophy* attempting to capture certain human activities in a precise way. They also can serve as *specifications* for future computer systems. We hope that our formalization can be useful in both respects.

The rest of the paper is organized as follows: In Section 2 we introduce preferential default logic, the logic we use as the logic of disputation in all examples of this paper. Readers satisfied with an intuitive understanding of this logic may skip this section. Section 3 reviews the basic notions of situation calculus. Section 4 investigates argumentation with fixed rules of order and introduces the notion of a static argument system. Section 5 presents, as a first case study, a new reconstruction of Rescher's theory of formal disputation. Section 6 investigates argumentation with dynamic rules of order and introduces dynamic argument systems in which meta level debates are possible. Section 7 discusses a particular dynamic argument system based on three levels and illustrates its behaviour using a debate about a murder case. Section 8 discusses related work and concludes.

2 A logic for argumentation

In this section we present preferential default logic, the logic used in this paper to define specific argument systems. The ideas underlying this logic were first developed in the context of logic programming [4] and later extended to full default logic in [3]. As mentioned earlier, we do not want to claim that this logic is the single best available logic for this purpose. The choice of the most adequate logic depends on the type of debate, and a number of good alternatives exist. Nevertheless, preferential default logic has a number of properties which seem relevant for many kinds of argumentation:

1. *Nonmonotonicity*
   Arguments are often defeasible, that is, additional information may invalidate what counted as acceptable argument before. The propositions accepted on the basis of given arguments thus do not grow monotonically with the available information.

2. *Explicit representation of preferences*
   Preferences among arguments play an important role in many settings. It should be possible to represent them explicitly.

3. *Reasoning about preferences*
   Preferences often are context dependent. Moreover, debates about preferences are very common. Preferences should therefore be represented in the logical language, as all other information relevant to the debate.

We first recall the basic definitions underlying Reiter's default logic [30]. A *default theory* is a pair $\Delta = (D, W)$ of a theory W containing first-order sentences and a set of defaults D. Each default is of the form $a : b_1, \ldots, b_n/c$, $n \geq 1$, where a, b_i, and c are first-order formulas. The intuitive meaning of the default is: if a is derived and the b_i are separately consistent with what is derived, then infer c. Formula a is called the *prerequisite*, each b_i a *justification*, and c the *consequent* of the default. For a default d we use $pre(d)$, $just(d)$, and $cons(d)$ to denote the prerequisite, the set of justifications, and the consequent of d, respectively; $\neg just(d)$ denotes $\{\neg a \mid a \in just(d)\}$. We say a default is defeated in a set of formulas S iff $\neg just(d) \cap S \neq \emptyset$.

As usual, we assume that open defaults, i.e., defaults with free variables, represent the sets of their ground instances over the Herbrand universe; a default theory with open defaults is closed by replacing open defaults with their ground instances. In what follows, we implicitly assume that default theories are closed before extensions are considered.

A (closed) default theory generates extensions which represent acceptable sets of beliefs a reasoner might adopt based on the given default theory (D, W). Extensions are defined as fixed points of an operator Γ_Δ. Γ_Δ maps an arbitrary set of formulas S to the smallest deductively closed set S' that contains W and satisfies the condition: if $a : b_1, \ldots, b_n/c \in D$, $a \in S'$ and $\neg b_i \notin S$ then $c \in S'$. Intuitively, an extension is a set of beliefs containing W such that 1) as many defaults as consistently possible have been applied, and 2) only formulas possessing a noncircular derivation from W using defaults in D are contained. The consequences of a default theory are the formulas contained in all extensions.

Prioritized default theories extend default theories with a strict partial order $<$ (i.e., $d \not< d$ and $d < d'$, $d' < d''$ implies $d < d''$) on the default rules. A default d will be considered preferred over default d', whenever $d < d'$ holds.

Definition 1 A prioritized default theory is a triple $\Delta = (D, W, <)$ where (D, W) is a default theory and $<$ is a strict partial order on D. Δ is called fully prioritized iff $<$ is a well-ordering.

Recall that a partial order is a well-ordering iff every subset of the elements has the least element. Any well-ordering is a total ordering.

Conclusions of prioritized default theories are defined in terms of preferred extensions, which are a subset of their extensions. The definition of preferred extensions for fully prioritized default theories in [3] is based on an additional fixpoint condition imposed on extensions. To avoid additional technical detail we use here a characterization based on the following proposition [3]:

Proposition 2 Let $\Delta = (W, D, <)$ be a fully prioritized ground default theory, and let E be a classical extension of Δ. Then, E is a preferred extension of Δ, if and only if for each default $d \in D$ such that $pre(d) \in E$ and $cons(d) \notin E$, there exists a set of defaults $K_d \subseteq \{d' \in GD(D,E) \mid d' < d\}$ such that d is defeated in $Th(W \cup cons(K_d))$.

Here $GD(D,E)$ denotes the set of all defaults from D which are generating in E. A default d is called *generating* in a set of formulas S, if $pre(d) \in S$ and $\neg just(d) \cap S = \emptyset$. The proposition basically says that in preferred extensions defaults which are not applied must be defeated by defaults with higher priority.

The general case of arbitrary prioritized default theories can then be reduced to this case as follows: Let $\Delta = (D, W, <)$ be a prioritized default theory. E is a prioritized extension of Δ iff E is a prioritized extension of a fully prioritized default theory $\Delta' = (D, W, <')$ such that $d < d'$ implies $d <' d'$.

In [3] we investigated two natural principles for preference handling in rule based nonmonotonic system. Principle I says that if the generating defaults for two extensions differ in exactly one default, and the differing default used to generate E_2 is less preferred than the one used to generate E_1, then E_2 should not be a preferred extension. Principle II says that the preferences of rules whose prerequisites are not in an extension should not be used to rule out this extension (see [3] for precise definitions). We have the following proposition [3]:

Proposition 3 Prioritized default logic satisfies both Principles I and II.

It turns out that most of the alternative approaches to preference handling in default logic described in the literature do not satisfy these principles.

To make reasoning *about* default priorities possible we must be able to refer to defaults explicitly, and we must introduce a special predicate symbol representing default preferences. We, therefore, extend our logical language in two respects.

1. We introduce a distinct set of rule names N. A naming function assigns a unique name to default rules. Formally, default names are simply ground terms in the underlying language. For a default d with $name(d) = n$ we use the notation $n :: d$.

2. We use the reserved two-place infix predicate symbol $\prec$ to represent default priority. For instance, if d_1 and d_2 are default names, then $d_1 \prec d_2$ is a formula with the intended meaning: d_1 has priority over d_2.

Definition 4 A preferential default theory is a triple $\Delta = (D, W, name)$ where (D, W) is a default theory, $name : D \to N$ is an injective function, and W contains axioms guaranteeing that $\prec$ is a strict partial order.

An extension of a preferential default theory $\Delta = (D, W, name)$ is just a classical extension of (D, W).

The preference information now is contained in the extensions of Δ. What we need is a way to eliminate an extension if it contains priority information which is in conflict with the way the extension was generated. Basically, an extension E of a preferential default theory is preferred iff E is a preferred extension of a fully prioritized default theory $(D, W, <)$ such that $<$ is compatible with the preference information in E. Compatibility is tested by generating a syntactic description of $<$ in terms of $\prec$ and checking whether this description is consistent with E.

Definition 5 Let $\Delta = (D, W, name)$ be a preferential default theory. Then, a set of formulas E is a preferred extension of Δ if and only if E is a preferred extension of some fully prioritized default theory $(D, W, <)$ such that $<$ is compatible with E. We say $<$ is compatible with E if and only if $E \cup \{d_i \prec d_k \mid r_i < r_k, name(r_i) = d_i, name(r_k) = d_k\}$ is consistent.

For further discussion of this approach and motivating examples see [4, 3]. An in depth understanding of preferential default logic will not be needed to follow our examples in the rest of this paper.

3 Situation calculus

The situation calculus was originally proposed by John McCarthy and Pat Hayes [20, 21] in the sixties as a means of representing actions and change in classical logic. Intuitively, properties of the world which may change through actions (so-called fluents) are indexed with the situation in which they hold. In the original proposal a situation is a snapshot of the state of the world. Reiter proposed to view situations as histories of actions. We follow in this paper Reiter's view and use his variant of the situation calculus. Reiter's book draft [32] provides an excellent introduction.

The language of the situation calculus is a second order language with sorts for actions and situations. Other sorts may be added depending on the application. The constant S_0 denotes the initial situation, $do(a, s)$ denotes the situation obtained by performing action a in situation s. $s \sqsubset s'$ means situation s is a proper subhistory of situation s'. $poss(a, s)$ says that it is possible to perform action a in situation s.

Reiter uses the following foundational axioms for situation calculus

$$
\begin{aligned}
&\neg s \sqsubset S_0 \\
&s \sqsubset do(a, s') \equiv s \sqsubseteq s' \\
&do(a_1, s_1) = do(a_2, s_2) \rightarrow a_1 = a_2 \wedge s_1 = s_2 \\
&(\forall P).P(S_0) \wedge (\forall a, s)[P(s) \rightarrow P(do(a, s))] \rightarrow (\forall s)P(s)
\end{aligned}
$$

The last axiom is a second order induction axiom which is needed if one wants to derive properties of all situations.

A possibility axiom is a sentence of the form

$$poss(A(x_1, \ldots, x_n), s) \equiv \Pi_A(x_1, \ldots, x_n, s)$$

where $\Pi_A(x_1, \ldots, x_n, s)$ is a formula whose free variables are in $x_1, \ldots, x_n, s$, and which is uniform in s. The latter condition, intuitively, means that the formula does not mention $poss$ and $\sqsubset$, it does not quantify over situation variables, it does not mention equality on situations, and the only situation it refers to is s.

A successor state axiom for a (relational) fluent[1] F is a sentence of the form

$$F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F(x_1, \ldots, x_n, a, s)$$

where $\phi_F(x_1, \ldots, x_n, a, s)$ is a formula whose free variables are in $x_1, \ldots, x_n, a, s$, and which is uniform in s.

[1] We do not discuss functional fluents [32] here.

Successor state axioms provide a (somewhat restricted) solution to the frame problem: how to describe what does not change when an event occurs. They can be derived automatically from effect axioms for actions. An effect axiom for action a describes the changes which result from performing action a [32].

A basic action theory $\mathcal{D}$ is a collection of axioms of the form

$$\mathcal{D} = \Sigma \cup \mathcal{D}_{ss} \cup \mathcal{D}_{poss} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$$

Here Σ is the set of foundational axioms, $\mathcal{D}_{ss}$ the set of successor state axioms, $\mathcal{D}_{poss}$ the set of possibility axioms, $\mathcal{D}_{una}$ unique names axioms for actions saying that different action terms denote different objects, and $\mathcal{D}_{S_0}$ the description of the initial state which must consist of sentences uniform in S_0.

In the rest of this paper we will make use of the notions developed above. In particular, we represent states of argumentation as situations which in our context will be called argumentation records. We use a single fluent $premise(g, q, r)$ to express that in situation r agent g accepts q as a premise. However, we will also extend "standard" situation calculus whenever necessary. In particular, it turns out that the uniformity requirement for successor state axioms is somewhat too restrictive for our purposes. Uniformity guarantees what is often called the Markov property: a successor state depends only on the immediately preceding state. In our context we sometimes need to refer to earlier states. For instance, we will use an *object* action which undoes the effects of illegal actions and restores the state before the illegal action. We therefore need slightly more general successor state axioms. Furthermore, we will admit nonmonotonic logics as the underlying inference mechanisms, and we will need deontic notions to represent protocols.

4 Argumentation with fixed protocol

4.1 Components of argument systems

Before we introduce our formalization we would like to discuss in this subsection informally what needs to be represented in our model of argumentation.

First of all, we need a group description, that is we have to fix the participants of the discussion, or, in other words, the members of the group engaged in the disputation. Moreover, different members of the group may have different roles which may come with different rights and/or obligations (e.g. chairperson vs. ordinary group members; defendent vs. prosecutor vs. judge; proponent vs. opponent; etc.).

Next, the possible speech acts that might be performed need to be specified, together with information about when they are possible, and what their effects are. We assume that each argument system has at least actions for claiming, retracting and conceding propositions. Moreover, it should be possible to object to illegal actions. Other actions (like proposing a new rule of order or voting) are optional.

We also need a language to express the propositions which are claimed, retracted etc. together with a corresponding consequence relation, that is, we need a logic which we call the logic of disputation. In many cases this logic will be nonmonotonic, that is, it will be possible to use statements representing default knowledge in the argumentation.

We need to keep track of the history and current state of the argumentation. We will use so-called argumentation records for this purpose. These correspond to situations in situation calculus. It must be possible to determine what is accepted by each group member and by the group as a wholerepresenting default knowledge in a particular state. To find out whether a group member, say g, accepts a proposition q, it must be possible to determine the premises currently accepted by g, and whether q follows from these in the logic of disputation.

The argumentation protocol is used to determine whether a person p is allowed to perform an action a in the current state of debate. We want this protocol to be described declaratively, by specifying the rules of order used in the discussion. More precisely, the rules of order are expressed in a logic L' which has terms for actions, L-propositions, persons and argumentation records. A binary predicate $legal$ is used to determine whether it is legal to perform action a given an argumentation record r. We call L' the logic of procedure, or the meta logic. Often L and L' will not differ wrt the underlying inference mechanisms, however they differ in the language (unless L is fully self-referential and has a quoting mechanism, see [23]).

Intuitively, an argumentation about a proposition q proceeds as follows: let r be the current argumentation record; a group member g can perform a certain speech act a if the speech act is possible in r. If a is not legal according to the argumentation protocol, then another member of the group can object to a, that is, insist that the effects of a are taken back. In any case, it is recorded in the argumentation record that action a was performed by p, and so on.

Argumentation about q may terminate with agreement because q is decided, that is either q or $\neg q$ is accepted by the group, or without agreement, for instance due to an external termination condition (time, deadline, etc.), or simply because no group member proposes a new move. In the latter case q is considered open. We will not say anything in this paper how to proceed in such a situation. Note, however, that decision criteria (e.g. voting) can be made part of the framework by specifying adequate actions.

In the next subsections we will formalize the components identified in the preceding discussion.

4.2 The logics

We have not much to say about L, the logic of disputation. This logic depends on the context in which argumentation is carried out. For the logic of procedure things are different. This logic is used to reason about the state of argumentation, to determine whether an action is legal, to determine whether a participant of the discussion is committed to a specific premise, etc. The logic of procedure therefore has to satisfy a number of conditions, namely, it has to be an L-meta logic.

Definition 6 Let L be a logic (of disputation). A logic L' is an L-meta logic iff

1. L' extends sorted first order logic,
2. among the sorts of L' are sorts for agents, actions, argumentation records and L-propositions,

3. L' has a term for each proposition in L, a term S_0 of sort argumentation record (the initial record), and a function symbol *do* of sort

$$\mathit{action} \times \mathit{argumentation\ record} \to \mathit{argumentation\ record},$$

4. L' has a unary predicate *member* (applied to agents), a binary predicate *legal* (applied to an action and an argumentation record) and a ternary predicate *premise* (applied to an agent, a proposition, and an argumentation record),

5. L' has a ternary predicate *accepts* (applied to an agent, a proposition, and an argumentation record). This predicate has the following standard interpretation:

$$\mathit{accepts}(g, q, r) \text{ iff } \{p \mid \mathit{premise}(g, p, r)\} \vdash_L q.$$

The use of the fluents *premise* and *legal* will be explained later. Let us first comment on the predicate *accepts*. Intuitively, $\mathit{accepts}(g, p, r)$ says that p follows in L from the premises of g in r. The predicate thus provides a link between meta-logic and logic of disputation. Since its definition refers to the inference relation of L, this definition cannot be expressed within L' (unless we assume that L is axiomatized in L'). We therefore simply require that *accepts* is interpreted the right way. Semantically, this means that we disregard models which do not satisfy its standard interpretation. From the implementation point of view it means that one needs access to a theorem prover for L. Note that if L is not decidable then it will not be decidable whether an L' interpretation is a model of a set of formulas.

Although we work in a multi-agent setting the function *do* has two arguments only, an action and an argumentation record. We will represent the agent performing the action as an additional argument of the action. As it turns out this leads to more concise formalizations.

The fluent *legal* is used in the description of the protocol. This obviously is a deontic notion, and one might consider using a full fledged deontic logic. It turns out, however, that for many protocols this is not necessary. The protocols we will present in this paper are not based on deontic logic.

The inference relation of L' is denoted $\vdash_{L'}$. In the rest of the paper g (possibly with index) is variable of sort agent, a of sort action, p, q are of sort L-proposition, and r, s of sort argumentation record.

Throughout the paper we assume uniqueness of names, that is unless explicitly specified otherwise different terms denote different objects.

Finally, we introduce the following abbreviations which will turn out to be useful later:

$$\begin{array}{ll} \mathit{accepted}(q, s): & \forall x.\ \mathit{member}(x) \to \mathit{accepts}(x, q, s) \\ \mathit{prevPremise}(g, q, r): & \exists a, s. r = \mathit{do}(a, s) \wedge \mathit{premise}(g, q, s) \\ \mathit{in}(a, s): & \exists s'.\ \mathit{do}(a, s') \sqsubseteq s \end{array}$$

where $\sqsubseteq$ is as described in Sect. 3.

4.3 The group description

Describing the participants and their roles in the debate is straightforward. All we need to do is name the group members and describe their roles using unary predicates.

Definition 7 A group description $\mathcal{D}_{group}$ consists of a universally quantified formula of the form

$$member(x) \equiv x = t_1 \vee \ldots \vee x = t_n$$

where the t_i are constants of sort agent, together with a (possibly empty) set of atomic formulas describing the roles of the group members.

For example, in a discussion involving proponent Jack, opponent John, and determiner Mary we may have $\mathcal{D}_{group} =$

$$\{member(x) \equiv x = Jack \vee x = John \vee x = Mary, pro(Jack), opp(John), det(Mary)\}.$$

Note that, although we have provided the possibility to use atomic formulas to describe the different roles of participants in the argumentation, this will be unnecessary in our examples in the rest of the paper. We will choose constants as names of participants which can be used to identify their roles (proponent, opponent, judge, ...). In more elaborate settings, e.g. if different persons have the same role, this is insufficient and additional formulas are needed.

4.4 The action description

Action descriptions must specify preconditions and effects of actions. In our context, actions typically have propositions as arguments, and they refer to and change the state of argumentation. Such states will be represented as argumentation records. These will be discussed in the next subsection. For the time being an intuitive understanding of argumentation records will be sufficient.

We use an additional argument for actions to represent the agent performing the action. For instance, $claim(g, q)$ represents that agent g claims proposition q.

Preconditions of an action specify under which conditions the action is possible. Note that this is different from when an action is legal (which will be specified in the argumentation protocol). For instance, retracting a proposition is possible in a situation s only if the proposition is accepted in s. This is not a consequence of the chosen protocol. Rather, it follows from the meaning of "retract".

As usual in situation calculus we use expressions of the form $poss(a, s)$ to represent that it is possible to perform action a in situation s. A possibility axiom for action a is a (universally quantified) first order formula of the form

$$poss(a, s) \equiv F(a, s)$$

where the free variables in $F(a, s)$ are among those in $poss(a, s)$.

The effects of actions are represented using formulas which describe what is true after the action was performed. If s is a situation, then $do(a, s)$ describes the situation which is obtained when action a is performed in s. The effects of action a are then simply described by specifying what is true in $do(a, s)$.

The effects of speech acts in argumentation concern the premises a group member is committed to. We use $premise(g, q, s)$ to express that in situation s agent g is committed to accepting q, where q is a formula in the logic of disputation. An effect axiom for action a is a (universally quantified) first order formula of the form

$$premise(g, q, do(a, s)) \leftarrow F(g, q, a, s)$$

or

$$\neg premise(g, q, do(a, s)) \leftarrow F(g, q, a, s)$$

where $F(g, q, a, s)$ is a formula whose free variables are among the free variables of $premise(g, q, do(a, s))$.

We are now in a position to define action descriptions.

Definition 8 An action description $\mathcal{D}_{action}$ is a set of possibility axioms and effect axioms.

To clarify these notions let us specify possibility and effect axioms for some actions one typically finds in argumentation contexts. All formulas are implicitly universally quantified.

- Claiming a proposition q:
  $poss(claim(g, q), s)$
  $premise(g, q, do(claim(g', q'), s)) \leftarrow g = g' \wedge q = q'$
- Conceding a proposition q:
  $poss(concede(g, q), s) \equiv \exists g'.g' \neq g \wedge premise(g', q, s)$
  $premise(g, q, do(concede(g', q'), s)) \leftarrow g = g' \wedge q = q'$
- Retracting a proposition q:
  $poss(retract(g, q), s) \equiv premise(g, q, s)$
  $\neg premise(g, q, do(retract(g', q'), s)) \leftarrow g = g' \wedge q = q'$
- Objecting to an illegal action:
  $poss(object(g), do(a, s)) \equiv \neg legal(a, s)$
  $premise(g, q, do(object(g'), s))) \leftarrow prevPremise(g, q, s)$
  $\neg premise(g, q, do(object(g'), s))) \leftarrow \neg prevPremise(g, q, s)$

The precise meaning of the predicate symbols *premise* and *legal* will be defined in the next subsections. We assume that the actions specified here are standard in all argumentation systems. Further actions can be defined as needed. The *object* action is needed for our solution of the regimentation problem: we admit illegal actions, but only if none of the participants objects. We assume that objecting to an illegal action amounts to "undoing" its effects. Note, however, that objections to illegal actions in our model have to be raised immediately after the action.

4.5 Argumentation records

Argumentation records keep track of the state of argumentation including the argumentation history. We use S_0 to denote the initial argumentation record. Additionally, $do(a, s)$ is the argumentation record representing the situation which is obtained when action a is performed in s. We say r' is a subrecord of r iff $r' \sqsubseteq r$ where $\sqsubseteq$ is as described in Sect. 3.

As mentioned earlier, the following predicates having argumentation records as arguments will be needed (AR stands for argumentation records):

$$\begin{array}{lll} premise: & \mathcal{P} \times \mathcal{L} \times AR & \rightarrow boolean \\ accepts: & \mathcal{P} \times \mathcal{L} \times AR & \rightarrow boolean \end{array}$$

These two predicates can be read as queries: *premise* checks whether an L-proposition is a premise an agent is committed to in a state; *accepts* checks whether an L-proposition is accepted by an agent.

We assume a complete description of the initial record in the following sense:

Definition 9 A set of formulas $\mathcal{D}_{S_0}$ is an initial state description if the only argumentation record term appearing in $\mathcal{D}_{S_0}$ is S_0.

$\mathcal{D}_{S_0}$ is *premise*-complete if for all ground atoms of the form $premise(g, a, S_0)$ we have $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \vdash premise(g, a, S_0)$ or $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \vdash \neg premise(g, a, S_0)$.

Here $\mathcal{D}_{una}$ is the set of unique names axioms for actions, agents and propositions. In many cases $premise(g, q, S_0)$ will simply be false, for all g and q. $\mathcal{D}_{S_0}$ may also contain state independent information, for instance useful abbreviations, or information about the topic of the debate (see our formalization of Rescher's theory of disputation, Sect. 5).

For the other argumentation records we can determine whether fluent *premise* holds through an update axiom.

Definition 10 An update axiom for fluent *premise* is a universally quantified first order formula of the form

$$premise(g, q, do(a, s)) \equiv F(g, q, a, s).$$

where the free variables of $F(g, q, a, s)$ are among those of $premise(g, q, do(a, s))$. We call the singleton set containing the update axiom D_{up}.
D_{up} is called compatible with an action description $\mathcal{D}_{action}$ iff $\mathcal{D}_{action} \cup \mathcal{D}_{up}$ is consistent.
D_{up} is called complete wrt. an initial state description $\mathcal{D}_{S_0}$ iff $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup D_{up} \vdash premise(g, q, s)$ or $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup D_{up} \vdash \neg premise(g, q, s)$ for all ground atoms of the form $premise(g, q, s)$.

As an example consider an update condition for the set of actions described in Section 4.4:[2]

$$\begin{aligned}
&premise(g, q, do(a, r)) \equiv \\
&\quad a = claim(g, q) \lor \\
&\quad a = concede(g, q) \lor \\
&\quad \exists g'.\ [a = object(g') \land prevPremise(g, q, r)] \lor \\
&\quad premise(g, q, r) \land \\
&\qquad \forall g'.\ [(a = retract(g', q) \rightarrow g \neq g') \land (a = object(g') \rightarrow prevPremise(g, q, r))]
\end{aligned}$$

This formula is exactly the successor state axiom Reiter would generate from the effect axioms of the available actions [32]. The first three lines of the right hand side of the equivalence list the actions and conditions that make *premise* true in the new state. The last three lines list the conditions under which *premise* remains true, given it was true in r already.

Note that the update axiom entails the effect axioms in $\mathcal{D}_{action}$. In principle, we might just forget about writing effect axioms at all. However, we found it useful to write down effect axioms first and to generate the update axiom from these.

[2] As usual we assume that $\land$ has stronger binding than $\lor$.

We therefore leave our representation somewhat redundant, even if this requires the introduction of a notion of compatibility.

We next show the following lemma:

Lemma 11 The formula described above is complete wrt. an arbitrary *premise* complete description of the initial state.

Proof Let $premise(g, p, r)$ be a ground atom. We show the result by induction on the number n of appearences of do in r. For $n = 0$ the result follows from the *premise* completeness of the initial state description. For $n > 0$ we can reduce the query to an equivalent query where the number of appearences of do in the record terms is smaller than n. The only subformulas where this is not immediate are the quantified subformulas obtained from expanding the macro $prevPremise(g, q, r)$. However, since r is ground it follows from the unique names axioms that there is, in each case, exactly one ground term for each variable satisfying the condition $r = do(a', r')$. We can therefore eliminate the quantifiers and continue with those ground terms. The number of appearences of do in the ground term obtained for r' is obviously smaller than n.

We conclude this subsection with the following definition:

Definition 12 An argumentation context is a set of formulas

$$\mathcal{D} = \Sigma \cup \mathcal{D}_{group} \cup \mathcal{D}_{action} \cup \mathcal{D}_{up} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$$

where Σ are the foundational axioms of situation calculus, $\mathcal{D}_{group}$ is a group description, $\mathcal{D}_{action}$ an action description, $\mathcal{D}_{up}$ contains the update axiom and is compatible with $\mathcal{D}_{action}$ and complete wrt $\mathcal{D}_{S_0}$, $\mathcal{D}_{una}$ are the unique names axioms for actions, agents and propositions, and $\mathcal{D}_{S_0}$ is a *premise*-complete description of the initial record.

4.6 The argumentation protocol

The argumentation protocol describes whether a particular action can be performed legally by a participant of the discussion in a particular state. This is the purpose of the binary predicate *legal*. Formally, an argumentation protocol is simply a set of formulas of the meta-logic L':

Definition 13 Let L' be an L-meta logic. $\mathcal{P}$ is an L'-argumentation protocol iff it is a set of L'-formulas.

The argumentation protocol contains the rules of order, that is, it describes declaratively whether a certain action can be performed legally. We assume that, given an argumentation context $\mathcal{D}$, an action a can be legally performed in a state described by argumentation record r iff $\mathcal{D} \cup \mathcal{P} \vdash_{L'} legal(a, r)$.

Although argumentation protocols describe norms for argumentation we do not require that a deontic logic be used, although this would certainly be a possibility. In our example applications we found it more important to be able to use defeasible rules to describe what is legal. This is why we use default logic to represent the protocols in this paper. Examples of argumentation protocols will be given later.

4.7 Static argument systems

We now define the central notion of this section. All we have to do is put together the pieces we have identified in the preceding discussions.

Definition 14 A (static) argument system AS is a tuple $(L, L', \mathcal{D}, \mathcal{P})$ where

- L is the logic of disputation,
- L' is an L-meta logic,
- $\mathcal{D}$ is an argumentation context,
- $\mathcal{P}$ is an L'-argumentation protocol.

We assume that the starting point of the debate is the initial record S_0. If the discussion starts with some initial agreement this has to be represented through the description of S_0.

We next define the possible argumentation records:

Definition 15 Let $AS = (L, L', \mathcal{D}, \mathcal{P})$ be a static argument system. An argumentation record r is possible in AS iff

- $r = S_0$, or
- $r = do(a, r')$, $\mathcal{D} \cup \mathcal{P} \vdash_{L'} poss(a, r')$, and r' is possible in AS.

The possible states an argumentation may take are thus represented as argumentation records. As long as an action is possible in a state, it can be performed, be it legal according to the protocol or not. However, objections can be raised against illegal actions. The effect of an objection is that the effects of the illegal action are eliminated.

We say an L-formula q is decided in a state r iff $\mathcal{D} \cup \mathcal{P} \vdash_{L'} accepted(q, r)$ (in which case q is positively decided) or $\mathcal{D} \cup \mathcal{P} \vdash_{L'} accepted(\neg q, r)$ x (in which case q is negatively decided), otherwise q is open.

Let us conclude this section with a remark on consistency. Our argument systems do not provide any theory revision mechanisms for keeping the premises accepted by a group member consistent. This lies fully within the responsability of each agent. Of course, consistency should be of primary interest to each agent - at least if the logic of disputation has no built in inconsistency handling mechanisms. If inconsistent premises imply acceptance of every formula then the agent's opinion has no impact whatsoever on what is accepted by the group as a whole.

5 Case study I: Rescher's theory of formal disputation

5.1 Rescher's approach

In [33] Rescher views disputation as a three player game. The proponent claims a certain thesis, the opponent can question the thesis through counterarguments or simply by not accepting it. If proponent and opponent do not come to an agreement the third player, the determiner, decides whether the thesis was defended successfully by the proponent or not.

Rescher identifies the following three fundamental moves:

1. *Categorical assertions* of the form $!P$ where P is a logical formula. The intuitive meaning of $!P$ is "P is the case". This move can be made by the proponent only.

2. *Cautious assertions* of the form $\dagger P$ where P is a logical formula. The intuitive meaning of $\dagger P$ is "P is the case for all you have shown" or "P is compatible with everything you've said". This move can be made by the opponent only.

3. *Provisoed assertions* of the form P/Q for "P generally obtains provided that Q". Provisoed assertions can be made by opponent and proponent but have to be preceded or accompanied by one of the two forms of asserting Q.

From the two-page discussion of provisoed assertions in Rescher's book it is obvious that what Rescher had in mind is exactly what is called default in AI today. As we will see soon he even was aware of the specificity principle, i.e. the idea that more specific defaults have to be preferred over more general ones. He thus anticipated ideas that became research topics in AI years later, see for instance [38, 39, 40].

Each move in the disputation game consists of one or more fundamental moves. A disputation always starts with a categorical assertion of the form $!P$ of the proponent. The opponent may counter this assertion by a *challenge* of the form $\dagger\neg P$ or a *provisoed denial* of the form $\neg P/Q \wedge \dagger Q$ for some suitable Q.

In the first case the proponent can now come up with an argument for P. This argument may have the form of a proof, i.e. the proponent can categorically assert formulas $P_1, \ldots, P_n$ that logically imply P and then in turn are open to challenge by the opponent. The argument may also have the form of a provisoed assertion accompanied by a categorical assertion of its precondition.

In the second case, i.e. when the proponent has to defend her thesis against a provisoed denial, the proponent can attack either the prerequisite Q of the attacking provisoed assertion or come up with a *strong distinction*, also called *strong exception*, of the form $P/(Q \wedge R) \wedge !(Q \wedge R)$. In modern terminology this means that the proponent asserts a more specific default together with its prerequisite such that the more general default of the opponent is overridden. Now, of course, the opponent in turn may attack this more specific default by a *weak distinction* of the form $\neg P/(Q \wedge R \wedge S) \wedge \dagger(Q \wedge R \wedge S)$.

The game continues like this until one of the two adversaries accepts the arguments, respectively counterarguments, of the other one. If the players cannot make any further reasonable moves or if some fixed time limit is exceeded before agreement was reached, then the determiner makes a decision based on the plausibility of the proponent's claims that were not conceded by the opponent.

Rescher assumes that concessions are made tacitly, i.e., if the opponent does not explicitly challenge a claim made by the proponent it is assumed that she concedes the claim ("silence implies consent"). Thus not only actions, but also certain non-actions have effects in Rescher's account. This is the reason why the update condition which we will need cannot simply be the successor state axiom for the available actions.

Here is a typical debate taken from Rescher's book (for an interpretation of the propositional symbols see [33], p. 16:

	proponent	opponent
(1)	$!P$	$\neg P/Q \wedge \dagger Q$
(2)	$!\neg Q$	$Q/R \wedge \dagger R$
(3)	$!\neg R$	$R/S \wedge \dagger S$
(4)	$\neg R/(S \wedge T) \wedge !(S \wedge T)$	$Q/U \wedge \dagger U$
(5)	$\neg Q/(U \wedge V) \wedge !(U \wedge V)$	

Note that in Rescher's approach provisoed assertions are always assumed to be "correct" and cannot be challenged directly. This implies that after the provisoed assertion P/Q was made it is impossible for the adversary to make the contradictory provisoed assertion $\neg P/Q$. The only way to attack a provisoed assertion is to come up with a more specific conflicting provisoed assertion. We find this somewhat implausible since disagreement about what is typically the case is as common as any other form of disagreement. Our reconstruction will therefore differ from Rescher in this respect.

Note also that Rescher did not think of the possibility of undermining a provisoed assertion through a "consistent" conflicting provisoed assertion. The standard example is

$$Pacifist(Nixon)/Quaker(Nixon)$$

which can be invalidated by the conflicting assertion

$$\neg Pacifist(Nixon)/Republican(Nixon)$$

when both $Quaker(Nixon)$ and $Republican(Nixon)$ is known. Our formalization will admit this form of attacking provisoed assertions as well.

5.2 The reconstruction

We now present our reconstruction of Rescher's theory of disputation.[3] Since, as mentioned earlier, several of the main ingredients of Rescher's theory remain vague and informal, this task involves a lot of interpretation, and obviously there is no way of proving the reconstruction correct in any reasonable sense. Moreover, making informal ideas precise often uncovers aspects and options that remained undetected in the original informal presentation of the ideas. This also happened in our reconstruction. Finally, we deliberately deviate from Rescher in some respects which we found unnecessary or implausible. Here is a list:

1. We do not use Rescher's special symbols ! and †. Although Rescher discusses their intuitive meaning extensively it is obvious that the formulas preceded by ! are exactly those stated by the proponent and the formulas preceded by † those stated by the opponent. Since our argument systems keep track of who stated what anyway the symbols are not necessary.

2. We allow disagreement not only about classical formulas but also about default rules.

[3] In [2] the author gave a different reconstruction using a version of Reiter's default logic with specificity. The reconstruction presented here is closer to what Rescher had in mind. In particular, contrary to [2] we take Rescher's principle *silence implies consent* seriously.

3. Defaults can be "attacked" not only by more specific defaults but also by other conflicting defaults.

In spite of these differences, we believe to have captured the spirit of Rescher's theory in our reconstruction. It is almost obvious how Rescher's fundamental moves can be modeled. A categorical assertion $!P$ corresponds to a claim made by the proponent. A cautious assertion $\dagger P$ whose meaning is somewhat ambiguous in Rescher's account corresponds either to a claim by the opponent or to a denial of $\neg P$ by the opponent. A provisoed assertion P/Q corresponds to the claim of a default rule, by the proponent or the opponent.

We base our approach on default logic with preferences, as described in Sect. 2. Although specificity is not built in, this logic allows to represent preferences (in particular those based on specificity) explicitly. In many cases specificity is not the only source of preferences, and other preference criteria may be more important. For this reason, we think generalizing Rescher in this respect is justified. Without going into technical detail we assume that the preference ordering $<$ prefers more specific defaults, in particular we assume $a(x) : b(x)/b(x) < c(x) : d(x)/d(x)$ whenever $W \vdash \forall x.a(x) \rightarrow c(x)$ but $W \not\vdash \forall x.c(x) \rightarrow a(x)$.

A Rescher type disputation is defined through a static argument system

$$Arg_R = (L, L', \mathcal{D}, \mathcal{P})$$

We now specify the components of this argument system. L is preferential default logic, L' is standard default logic with the necessary sorts, function and predicate symbols. The components of $\mathcal{D}$ are as follows:

$$\mathcal{D}_{group} = \{member(x) \equiv x = pro \vee x = opp \vee x = det\}$$

We foresee the actions

$$claim(g, q), concede(g, q), deny(g, q), retract(g, q), object(g), declare\text{-}winner(g, g')$$

where q is of sort L-proposition and g, g' are of sort agent. Possibility and effect axioms for $claim, concede, retract$ and $object$ are as in Section 4.4. The axioms for the remaining actions are as follows:

- Denying a formula:
  $possible(deny(g, q), r) \equiv \exists g'.g' \neq g \wedge premise(g', q, r)$
  $\neg premise(g, q, do(deny(g, q), r))$
- Declaring a winner:
  $possible(declare\text{-}winner(g, g'), r)$
  $premise(g, q, do(declare\text{-}winner(g', g''), r)) \leftarrow g'' = pro \wedge topic = q$

The last formula says that if the proponent is declared winner of the dispute about q (by the determiner), then every group member has to accept q. The constant *topic* is used to describe the proposition the debate is about. This information (which is independent of the state of argumentation) is added to the description of the initial state $\mathcal{D}_{S_0}$.

The update condition $\mathcal{D}_{up}$ is as follows:

$$
\begin{aligned}
&premise(g,q,do(a,r)) \equiv \\
&\quad a = concede(g,q) \vee \\
&\quad a = claim(g,q) \vee \\
&\quad a = declare\text{-}winner(det, pro) \wedge q = topic \vee \\
&\quad \exists g'.\ [a = object(g') \wedge prevPremise(g,q,r)] \vee \\
&\quad premise(g,q,r) \wedge \forall g'.\ [(a = retract(g',q) \vee a = deny(g',q) \rightarrow g \neq g') \wedge \\
&\qquad\qquad\qquad\qquad\qquad (a = object(g') \rightarrow prevPremise(g,q,r))] \vee \\
&\quad agt(a) = g \wedge premise(ad(g),q,r) \wedge \neg in(deny(g,q), do(a,r)) \wedge \\
&\qquad\qquad\qquad\qquad\qquad \neg in(retract(g,q), do(a,r))
\end{aligned}
$$

where $ad(pro) = opp$, $ad(opp) = pro$ and $ad(det) = det$ and

$$
\begin{aligned}
agt(a) = g \equiv\ & a = claim(g,q) \vee a = concede(g,q) \vee a = deny(g,q) \vee a = retract(g,q) \vee \\
& a = object(g) \vee a = declare\text{-}winner(g,g')
\end{aligned}
$$

The update condition captures the silence implies consent principle. The opponent as well as the proponent can be committed to a premise claimed by the proponent, respectively opponent, also by not explicitly denying or retracting it. In other words, our update condition models the effect of certain non-actions. This is reflected in the last alternative which, intuitively, says: if one of the disputing agents performs an action then she is assumed to accept the premises held by the other agent, unless she has (now or earlier) denied or retracted that premise.

$\mathcal{D}_{S_0}$ contains the formula $\neg premise(g,q,S_0)$, that is, nothing is accepted in the beginning. Additionally, an atomic formula of the form $topic = p$ describes the topic of the debate.

Again we have to prove:

Lemma 16 $\mathcal{D}_{up}$ as defined above is complete wrt. $\mathcal{D}_{S_0}$.

Proof Similar to the proof of Lemma 11.

The protocol $\mathcal{P}$ is based on (standard) default logic and consists of the following formulas and defaults:

$$
\begin{aligned}
& true : legal(a,r)/legal(a,r) \\
& legal(declare\text{-}winner(g,g'),r) \rightarrow g = det \wedge det \neq g' \\
& legal(a,r) \wedge agt(a) = det \rightarrow [\exists g.\ a = declare\text{-}winner(det,g)] \\
& legal(declare\text{-}winner(det,g),r) \rightarrow \\
&\qquad\qquad g = pro \wedge accepts(g, topic, r) \vee \\
&\qquad\qquad g = opp \wedge \neg accepts(g, topic, r)
\end{aligned}
$$

These rules of order basically say that actions normally are legal. However, only the determiner can declare who the winner of the debate is (and only proponent or opponent can be the winner). Moreover, this is the only action the determiner can make. Finally, certain reasonable restrictions have to hold to make the decision of the determiner legal. For instance, the proponent cannot be declared winner if she does not believe in the topic of the debate any longer.

As usual, objections are possible after illegal actions only. This is guaranteed by the possibility condition for *object* and does not need to be specified in the protocol.

To illustrate the behaviour of this argument system we formalize a legal dispute based on US law. The example was first discussed by Gordon [11] who also gives additional legal background.

The proponent claims that her security interest in a certain ship is perfected. The opponent denies this. The proponent argues that according to the Uniform Commercial Code (UCC, §9-305) a security interest in goods may be perfected by taking possession of the collateral, and that she currently has possession of the ship. The opponent refers to a federal law called the Ship Mortgage Act (SMA) according to which a security interest in a ship may only be perfected by filing a financing statement. Such a statement has not been filed. Moreover, he claims that according to the principle of *Lex Posterior* which gives precedence to newer laws UCC should be preferred over SMA since the former is newer. The proponent now refers to the principle of *Lex Superior* which gives precedence to laws supported by the higher authority. Moreover, the opponent claims that Lex Superior is to be preferred over Lex Posterior. The opponent accepts this. The determiner (erroneously) declares the opponent to be the winner of the debate. Since the premises of both proponent and opponent now imply that the security interest is perfected, the dispute is settled and the determiner's decision is not legal according to the protocol. Therefore the proponent objects. The determiner corrects his mistake and declares the proponent to be the winner of the debate. Here is the formal representation of this debate:

$$
\begin{aligned}
S_1 &= do(claim(pro, perfected), S_0)\\
S_2 &= do(deny(opp, perfected), S_1)\\
S_3 &= do(claim(pro, UCC :: possession : perfected/perfected), S_2)\\
S_4 &= do(claim(pro, possession), S_3)\\
S_5 &= do(claim(opp, SMA :: ship \wedge \neg fin\text{-}statement : \neg perfected/\neg perfected), S_4)\\
S_6 &= do(claim(opp, LexPos :: true : UCC \prec SMA/UCC \prec SMA), S_5)\\
S_7 &= do(claim(pro, LexSup :: true : SMA \prec UCC/SMA \prec UCC), S_6)\\
S_8 &= do(claim(pro, LexSup \prec LexPos), S_7)\\
S_9 &= do(concede(opp, LexSup \prec LexPos), S_8)\\
S_{10} &= do(declare\text{-}winner(det, opp), S_9)\\
S_{11} &= do(object(pro), S_{10})\\
S_{12} &= do(declare\text{-}winner(det, pro), S_{11})
\end{aligned}
$$

6 Argumentation with dynamic protocol

6.1 Levels of Argumentation

The question we want to address now is the following: how can we model that, at any point in the argumentation, participants can start a meta level debate, that is, the rules of order can be made the current topic of discussion?

Note that rules of order speak about propositions expressed in the language of disputation. One possibility would be to amalgamate the languages of L and L', that is to use a self-referential language with quoting (see [23] for a study of such languages) and to express the rules of order in that language. Although this might lead to a very uniform and monolithic treatment of argumentation, we would have to use heavy logical machinery even for the simplest kinds of argumentation. We are not interested here in solving problems related to self-reference. Moreover, a number of approaches dealing with problems of self-reference explicitly use levels (e.g. Russell's typed system of logic which tried to avoid the inconsistency of Frege's original system). We therefore believe that it is more reasonable to keep different levels of discussion separate from the beginning.

We strongly believe that participants in a discussion are able to distinguish between the levels of discussion they are currently engaged in (object level, meta-level, meta-meta-level etc.). For instance, in a trial it is straightforward to distinguish between arguments about whether, say, Jack is a murderer, or whether a speech act is legal, or whether the rules used to determine legality of speech acts should be changed.

We therefore base our approach on argument systems with different argumentation levels. In static argument systems the rules of order can be represented as a fixed rule base. Since we now want to model potential disagreement about the rules of order as well as dynamic changes of these rules we have to use an argumentation record not only for the basic level, but also for the meta-level, the meta-meta-level and so on. In each case the argumentation record r^{i+1} of level $i+1$ will be used to keep track of the current state of debate at this (meta-) level and to determine whether performing an action a in state r^i of level i is legal or not. The action is considered legal if there is agreement about this in the group, in other words, if $legal(a, r^i)$ is provable from the argumentation context of level i together with the formulas accepted at level $i + 1$.

6.2 Dynamic Argument Systems

To represent the different levels of argumentation we have to extend the notion of a static argument system accordingly. Each level may have its own argumentation context. However, since we now assume that the current state of debate about the protocol of level i is represented on level $i + 1$, we do not have to include a protocol in the levels, except for the highest level. This leads to the following definition:

Definition 17 A dynamic n-level argument system is a tuple

$$(L_0, (L_1, \mathcal{D}_1), \ldots, (L_n, \mathcal{D}_n), \mathcal{P})$$

where L_0 is the logic of disputation, $\mathcal{P}$ is an L_n protocol representing the rules of order for level n, and for all $i \in \{1, \ldots n\}$ L_i is an L_{i-1}-meta logic and $\mathcal{D}_i$ is the argumentation context of level i.

Dynamic argument systems thus provide a fixed number of argumentation records.[4] The idea is that in each state the current record of level i describes the state of debate about how to infer whether an action of level i-1 is legal.

Dynamic argument systems are very general. They can model situations where the group involved in the debate varies from level to level. Also the available actions may differ from level to level. In many examples (e.g. in the next section) this full generality is not needed. Note, however, that even if actions like claiming, conceding or retracting often appear in all levels, their proposition arguments will differ from level to level.

[4] Extending our definition to the infinite case is straightforward: just provide an infinite number of pairs $(L_i, \mathcal{D}_i)$ and eliminate $\mathcal{P}$ which is not necessary any longer. We don't think inifinite systems are of more than theoretical relevance, though.

The following result is obvious

Lemma 18 A static argument system is a dynamic argument system with exactly one level.

The definition of possible states for dynamic argument systems is as follows:

Definition 19 Let $AS = (L_0, (L_1, \mathcal{D}_1), \ldots, (L_n, \mathcal{D}_n), \mathcal{P})$ be a dynamic argument system. The possible states of AS are defined inductively as follows:

1. $(S_0^1, \ldots, S_0^n)$, where S_0^i is the initial argumentation record of level i, is a possible state of AS,

2. let $(r_1, \ldots, r_n)$ be a possible state of AS, $i < n$, and a an action of level i whose agent argument is g. If
$$\mathcal{D}_i \cup \{q \mid \mathcal{D}_{i+1} \vdash \mathit{premise}(g, q, r_{i+1})\} \vdash_{L_i} \mathit{poss}(a, r_i)$$
then $(r_1, \ldots, r_{n-1}, \mathit{do}(a, r_i), r_{n+1}, \ldots, r_n)$ is a possible state of AS.

3. let $(r_1, \ldots, r_n)$ be a possible state of AS, a an action of level n whose agent argument is g. If
$$\mathcal{D}_n \cup \mathcal{P} \vdash_{L_n} \mathit{poss}(g, a, r_n)$$
then $(r_1, \ldots, r_{n-1}, \mathit{do}(a, r_n))$ is a possible state of AS.

Clause 2 in this definition probably needs further explanation. The possibility of almost all actions can be determined from $\mathcal{D}_i$ alone. Only for actions like *object* whose possibility condition mentions legality, the protocol, that is the higher level, is relevant. We want to make sure that objections are possible for an agent g whenever, according to g's current opinion, the action performed earlier was illegal. We therefore have to add those premises which represent g's current view of the protocol to $\mathcal{D}_i$ to determine the possibility of an objection. Adding only those formulas which are accepted by the whole group in the higher level is not desirable as it would make fewer objections possible.

Note that in dynamic argument systems the protocol for actions referring to the highest level is fixed. We conclude this section with the following definition:

Definition 20 Let AS be a dynamic argument system, $S = (r_1, \ldots, r_n)$ a possible state of AS. A proposition q (written in the language of L_{i-1}) is accepted in S in level i iff $\mathcal{D}_i \vdash_{L_i} \mathit{accepted}(q, r_i)$.

7 Case study II: A three level argument system

After defining the notion of a dynamic argument system in the last section we now present one particular such system. Again we will use an example taken from the legal domain to show how the system works. As before our main motivation is to illustrate our notions, not to give a realistic picture of legal argumentation. The argument system to be defined here, we call it Arg_3, has 3 levels, that is it has the form:

$$Arg_3 = (L_0, (L_1, \mathcal{D}_1), (L_2, \mathcal{D}_2), (L_3, \mathcal{D}_3), \mathcal{P})$$

We now define the constituents of Arg_3. We will need a judge, a defender (def), and a prosecutor (pro). We assume that the group is the same in all levels, that is we have

$$member(x) \equiv x = judge \vee x = def \vee x = pro$$

in the group description of all levels. We further assume that actions and update condition are the same in all levels, upto the type of propositions used as arguments of the actions. The actions we admit are

$$claim(g, q), concede(g, q), retract(g, q), decide(g, q), propose(g, q), object(g)$$

The effect and possibility axioms for those actions not already defined in Sect. 4.4 are as follows:

- Deciding that a proposition is to be accepted:
  $possible(decide(g, q), r)$
  $premise(g, q, do(decide(g', q), r))$
- Proposing a new rule of order:
  $possible(propose(g, q), r)$

Thus deciding and proposing a formula is always possible (although not always legal, as we will see). Note that there is no effect axiom for *propose*: proposing a formula has no effect on the accepted premises. As we will see the only effect here is that a certain other action becomes legal after a formula was proposed.

The specification of the axioms for the other actions are as in Sect. 4.4.

The specification of the update condition (used in all levels) is as follows:

$$\begin{aligned}
&premise(g, q, do(a, r)) \equiv \\
&\quad a = claim(g, q) \vee \\
&\quad a = concede(g, q) \vee \\
&\quad \exists g'.\ a = decide(g', q) \vee \\
&\quad \exists g'.\ [a = object(g') \wedge prevPremise(g, q, r)] \vee \\
&\quad premise(g, q, r) \wedge \forall g'.\ [(a = retract(g', q) \rightarrow g \neq g') \wedge \\
&\qquad\qquad\qquad\qquad (a = object(g') \rightarrow prevPremise(g, q, r))]
\end{aligned}$$

Note that Rescher's silence implies consent principle is not valid in our dynamic argument system: according to the specification of the update condition all concessions have to be explicit. In addition, the judge can decide that all participants of the discussion have to accept a certain proposition.

As the underlying logic of all levels we use (standard) default logic with the required sorts and predicates.

The (meta-meta-) protocol $\mathcal{P}$ simply says that on the highest level every action is illegal, that is $\mathcal{P} = \{\neg legal(a, s)\}$. This has the effect that the meta-protocol, represented in level 3, can only be changed if nobody objects.

Let us consider a dispute about whether Jack is a murderer. We assume that argumentation starts in a state $S_0 = (AR_0, MR_0, MMR_0)$ where several pieces of information are already accepted:
accepted in AR_0:

$motive \wedge \neg alibi \wedge on(blood, shoe) : murderer/murderer$
$motive$
$\neg alibi$

accepted in MR_0:

$true : legal(a, r)/legal(a, r)$
$\neg legal(decide(def, q), r)$
$\neg legal(decide(pro, q), r)$

accepted in MMR_0:

$legal(claim(g, illegal\text{-}info(q)), r)$
$legal(propose(g, q), r)$
$legal(decide(judge, q), do(propose(g, q), r))$
$legal(retract(judge, q), r)$

Intuitively, there is agreement in MR_0 that only the judge can make decisions regarding basic level propositions. Furthermore, there is agreement in MMR_0 that the judge can decide that a meta level proposition, that is a rule of order, is to be accepted, given the proposition was proposed. Additionally, the judge can retract meta level propositions, and each group member can claim that information was illegally obtained ($illegal\text{-}info(q)$).

The dispute now proceeds as follows: there is evidence (victim's blood on Jack's shoe) which convicts Jack as the murderer. However, the evidence was obtained illegally. The clever defender proposes to change the rules of order in such a way that illegal evidence cannot be used. The judge agrees and the prosecutor is forced to retract his claim since this is the only action he can legally perform in this situation.

We use R as abbreviation for the rule

$$illegal\text{-}info(q) \wedge premise(pro, q, r) \wedge a \neq retract(pro, q) \rightarrow \neg legal(a, r)$$

In addition, we use the following abbreviations:

$AR_1 = do(claim(pro, on(blood, shoe)), AR_0)$
$MR_1 = do(claim(def, illegal\text{-}info(on(blood, shoe))), MR_0)$
$MR_2 = do(decide(judge, illegal\text{-}info(on(blood, shoe))), MR_1)$
$MR_3 = do(propose(def, R), MR_2)$
$MR_4 = do(decide(judge, R), MR_3)$
$AR_2 = do(retract(pro, on(blood, shoe)), AR_1)$

Argumentation proceeds as follows:

$S_0 = (AR_0, MR_0, MMR_0)$	the initial argumentation state
$S_1 = (AR_1, MR_0, MMR_0)$	pro claims there was blood on Jack's shoe
$S_2 = (AR_1, MR_1, MMR_0)$	def claims this evidence was obtained illegally
$S_3 = (AR_1, MR_2, MMR_0)$	judge decides evidence is illegal
$S_4 = (AR_1, MR_3, MMR_0)$	def proposes new rule of order
$S_5 = (AR_1, MR_4, MMR_0)$	judge decides the new rule is to be accepted
$S_6 = (AR_2, MR_4, MMR_0)$	pro has to retract the evidence

This means that in S_6 none of the participants accepts that Jack is the murderer. In order to convict Jack, the prosecutor has to come up with a new strategy based on legal evidence only.

8 Discussion and related work

During recent years the field of AI has seen a continuously growing interest in problem solving techniques which are based on the cooperation of multiple intelligent agents. To model the cooperation of such agents one has to come up with models of how they communicate, negotiate, exchange arguments, and settle disputes.

There is a vast amount of literature on this topic. In particular, negotiation has been a subject of central interest. The term negotiation has been used in a variety of ways. Zlotkin and Rosenschein [46, 34] have focussed on task redistribution. The overall goal of the negotiation here is to minimize the costs associated with tasks to be performed by agents. Others have investigated negotiation in the context of plan generation [8], goal modification [36], argumentation-based proposal generation [13] or negotiation under time constraints [14]. In these papers the focus is more on the optimization of the associated utility. The logical structure of the argumentation plays a minor role, and the underlying logical language is extremely simple.

There is also an important difference between our notion of argumentation and the one used by Kraus, Sycara and Evenchik [35]. What they are studying is dialogues in which one party tries to persuade another party to adopt (or change) a certain intention for action. Kraus et al. model this by representing agents as BDI (belief - desire - intention) architectures. Arguments are pieces of information which attempt to change the other agent's intentions by changing the other agent's beliefs about how it can or cannot satisfy its goals or desires.

Our interest in this paper was on debate or critical discussion, i.e., a form of communication where one or more participants try to convince others that a certain proposition holds by communicating the beliefs on which the proposition rests. Intentions of the agents did not play a role in our formalization. It is a topic of future research whether our framework can be applied fruitfully to the type of dialogues studied by Kraus et al. and others. Of course, a unifying theory for different types of dialogues would be a nice thing to have, in particular since in practice different types of dialogues will often be intertwined.

Here is a list of the contributions of this paper:

1. We have shown that situation calculus can play a central role in the formalization of structured debates in settings where different group members may have different rights and obligations.

2. We have shown how protocols can be represented declaratively in this context.

3. We have proposed a solution to the regimentation problem: how to use norms to guide the behaviour of agents without strictly enforcing these norms? Our solution to the regimentation problem admits norm violations as long as nobody objects. If one of the group members insists in sticking to the norms, however, then the effects of illegal actions are taken back. Violating the norms thus is a possibility but bears some risk.

4. We have demonstrated how meta-debates about the protocol can be formalized.

5. We have proposed a novel reconstruction of Rescher's theory of formal disputation. In particular, this formalization fully captures Rescher's *silence implies consent* principle.

As mentioned in the introduction restricted forms of meta-argumentation were investigated by Vreeswijk [42]. He uses propositional constants to represent certain aspects of the argumentation protocol, so-called partial protocol specifications (PPSs). A PPS can be adopted or given up during argumentation which modifies the behaviour of the implementation of the core protocol in a predefined way. It should be obvious that, due to the declarative representation of the protocol, our approach is much more flexible. Changes to the protocol are not restricted to a fixed number of predefined properties which can be switched on or off. Moreover, in our approach meta-argumentation is not restricted to one single meta-level but can be extended to an arbitrary number of levels.

The argumentation protocols we used in our examples were extremely simplistic. Needless to say that real argumentation processes follow much more sophisticated rules of order. A widely used standard protocol for parliamentary debate was developed by General Robert in the last century. His rules are now known as Robert's Rules of Order (RRO). Prakken [26] has demonstrated how RRO can be formalized. It is straightforward to use this formalization as part of the specification of a dynamic argument system. This would lead to a system which is much more realistic than the ones we used here to illustrate our notions.

In future work we plan to implement systems based on our model which are based on such more realistic protocols. We will also try to classify types of protocols and investigate notions of fairness in the context of our approach.

The framework developed in this paper is highly general. We have demonstrated that it provides the necessary means to formalize interesting types of discussions. To the best of our knowledge a formal definition of argument systems which is general enough to capture dialogues of the form described in our murder case example has not been given before. We hope that our formalization will prove helpful, not only for the long term AI goal of developing artificial arguing agents but also for somewhat less ambitious (some would say more realistic) goals. For instance, our formalization could be the formal basis of a mediating system (see Zeno, http://nathan.gmd.de/projects/zeno/zenoSystem.html, for an interesting example) supporting, say, electronic discussions among natural agents. Such a system could keep track of the state of the discussion, help detecting inconsistencies in the statements of the participants, and give warnings if certain speech acts violate the accepted rules of order.

Acknowledgements

The work reported in this paper was funded by DFG, Forschergruppe Kommunikatives Verstehen. I would like to thank Tom Gordon, Henry Prakken and Gerard Vreeswijk for interesting discussions on the topic of this paper. In particular, Henry convinced me of the importance of admitting protocol violations. Special thanks to one of the anonymous reviewers whose suggestions helped to simplify the formalization considerably.

References

[1] A. Bondarenko, P.M. Dung, R. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence 93(1-2)*, pages 63–101, 1997.

[2] G. Brewka. A reconstruction of rescher's theory of formal disputation based on default logic. In *Proc. ECAI-94*, pages 366–370, 1994.

[3] G. Brewka and T. Eiter. Prioritizing default logic. In *Festschrift 60th Anniversary of W. Bibel*. Kluwer, 1998.

[4] G. Brewka and T. Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence 109*, pages 297–356, 1999.

[5] C. Cayrol. On the relation between argumentation and coherence based entailment. In *Proc. IJCAI-95*, pages 1443–1448, 1995.

[6] C.I. Chesñevar, A. Maguitman, and R.P. Loui. Logical models of arguments. *ACM Computing Surveys*, to appear 2000.

[7] P.M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence 76(2)*, 1995.

[8] E.H. Durfee. *Coordination of Distributed Problem Solvers*. Kluwer Academic Publishers, Boston, 1988.

[9] J. Fox, P. Krause, and S. Ambler. Arguments, contradictions and practical reasoning. In *Proc. ECAI-92*, pages 623–627, 1992.

[10] H. Geffner and J. Pearl. Conditional entailment: Bridging two approaches to default reasoning. *Artificial Intelligence 53*, 1992.

[11] T. Gordon. *The Pleadings Game: An Artificial Intelligence Model of Procedural Justice*. TU Darmstadt, dissertation, 1993.

[12] C.L. Hamblin. *Fallacies*. Methuen, London, 1970.

[13] N.R. Jennings, S. Parsons, P. Noriega, and C. Sierra. On argumentation-based negotiation. In *Proc. Intl. Workshop on Multi-Agent Systems, Boston*, 1998.

[14] S. Kraus, J. Wilkenfeld, and G. Zlotkin. Multiagent negotiation under time constraints. *Artificial Intelligence 75(2)*, pages 297–345, 1995.

[15] P. Krause, S. Ambler, M. Elvang-Goransson, and J. Fox. A logic of argumentation for uncertain reasoning. *Computational Intelligence 11(1)*, pages 113–131, 1995.

[16] C. Krogh. Obligations in multi-agent systems. In *Proc. 5th Scandinavian Conference on AI, Trondheim*, pages 19–30, 1995.

[17] H. Levesque, F. Pirri, and R. Reiter. Foundations for the situation calculus. *European Transactions on AI (reference paper)*, 1998.

[18] P. Lorenzen and K. Lorenz. *Dialogische Logik.* Wissenschaftliche Buchgesellschaft, Darmstadt, 1978.

[19] R.P. Loui. Process and policy: Resource-bounded non-demonstrative argument. Technical report, Dep. of Computer Science, Washington Univ., St. Louis, 1992.

[20] J. McCarthy. Situations, actions and causal laws. Technical report, Stanford Artificial Intelligence Project, Memo 2, 1963.

[21] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *B. Meltzer, D. Michie (eds.), Machine Intelligence 4*, pages 463–502, 1969.

[22] S. Parsons, C. Sierra, and N.R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation 8(3)*, pages 261–292, 1998.

[23] D. Perlis. Languages with self-reference i: Foundations. *Artificial Intelligence 25*, pages 301–322, 1985.

[24] J.L. Pollock. Defeasible reasoning. *Cognitive Science 11*, pages 481–518, 1987.

[25] J.L. Pollock. Oscar - a general purpose defeasible reasoner. *Journal of Applied Non-Classical Logics 6(1)*, pages 89–113, 1996.

[26] H. Prakken. Formalizing robert's rules of order. Technical report, GMD Sankt Augustin, Report 12, 1998.

[27] H. Prakken and G. Sartor. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-classical Logics 7*, pages 25–75, 1997.

[28] H. Prakken and G. Vreeswijk. Logics for defeasible argumentation. In *D. Gabbay (ed.), Handbook of Philosophical Logic, 2nd edition.* Kluwer Academic Publishers, to appear.

[29] C.A. Reed and D.P. Long. Content ordering in the generation of persuasive discourse. In *Proc. IJCAI-97, Nagoya, Japan*, 1997.

[30] R. Reiter. A logic for default reasoning. *Artificial Intelligence 13*, pages 81–132, 1980.

[31] R. Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In *V. Lifschitz (ed.), Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy.* Academic Press, 1991.

[32] R. Reiter. *Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems.* book draft, 1999.

[33] N. Rescher. *Dialectics: A Controversy-Oriented Approach to the Theory of Knowledge.* State University of New York Press, Albany, 1977.

[34] J.S. Rosenschein. Consenting agents: Negotiation mechanisms for multi-agent systems. In *Proc. IJCAI-93, Chambery*, 1993.

[35] S. S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation: A logical model and implementation. *Artificial Intelligence 104*, pages 1–69, 1998.

[36] K.P. Sycara. Argumentation: Planning other agent's plans. In *Proc. IJCAI-89, Detroit*, 1989.

[37] S. Toulmin. *The Uses of Argument.* Cambridge University Press, Cambridge, 1956.

[38] D.S. Touretzky. *The Mathematics of Inheritance.* Pitman Research Notes in Artificial Intelligence, London, 1986.

[39] D.S. Touretzky, J.F. Horty, and R.H. Thomason. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In *Proc. IJCAI-87, Milan*, 1987.

[40] D.S. Touretzky, R.H. Thomason, and J.F. Horty. A skeptic's menagerie: Conflictors, preemptors, reinstaters, and zombies in nonmonotonic inheritance. In *Proc. IJCAI-91, Sydney*, 1991.

[41] F.H. van Eemeren and R. Grootendorst. *Argumentation, Comunication and Fallacies: A Pragma-Dialectical Perspective.* Lawrence Erlbaum, London, 1992.

[42] G. Vreeswijk. Representation of formal dispute with a standing order. *Artificial Intelligence and Law, Special issue Dialectical Legal Argument: Formal and Informal Models*, to appear 2000.

[43] D. Walton. Applying labelled deductive systems and multi-agent systems to source-based argumentation. *Journal of Logic and Computation 9(1)*, pages 63–80, 1999.

[44] D. Walton and C.W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning.* State University of New York Press, Albany, 1995.

[45] M. Wooldridge and N.R. Jennings. Cooperative problem-solving. *Journal of Logic and Computation 9(4)*, pages 563–592, 1999.

[46] G. Zlotkin and J.S. Rosenschein. A domain theory for task oriented negotiation. In *Proc. IJCAI-93, Chambery*, 1993.