Events, Time and Argumentative Systems *

María Laura Cobo[†]   **Guillermo R. Simari**[‡]

Department of Computer Science and Engineering – Universidad Nacional del Sur
Artificial Intelligence Research and Development Laboratory (L.I.D.I.A)
Institute of Computer Science and Engineering
Av. Alem 1253 – B8000CPB Bahía Blanca, ARGENTINA
[†] Email: `mlc@cs.uns.edu.ar`
[‡] Email: `grs@cs.uns.edu.ar`

ABSTRACT

The accomplishment of systems with abilities to reason about actions and change and systems that can manage incomplete or not very reliable information with abilities to discuss or argument has been of great importance for artificial intelligence community. These two ways of reasoning were attacked independently, but they are complementary, since a lot of applications need of both, since all dynamic systems (dynamic on information) counts with uncomplete information and information that depends on events and time.

The line of investigation suggested on this present work tries to achieve a system that can reason about action and at the same time can elaborate a discussion, *i. e.* intends to conciliate *argumentative systems* with *reasoning about actions and change* or *temporal reasoning*.

Keywords: Argumentative Systems, Knowledge Representation, Defeasible Reasoning, Commonsense Reasoning, Temporal Reasoning, Reasoning about actions and change.

1 INTRODUCTION AND MOTIVATION

The accomplishment of systems with abilities to reason about actions and change has been of great importance for artificial intelligence community. Research community is interested on this issue because a wider variety of problems is could be solved, problems where actions and when they took place or happens set a difference [5].

Other important systems, *argumentative*, [2, 13] were or are being developed, in order to deal with not completely reliable information, or with incomplete one. In real scenarios this kind of information is quite common, specially when we treat with dynamic systems, *i.e.* systems where the knowledge we count on to reason changes with frequency (new information become available or information we used to count on with is no longer available or valid). Usually incomplete information appears in any way of reasoning because its very difficult to represent absolutely all the information related to the objects we count on. As a matter of fact there are systems such as *Situation Calculus* [16] where this problem is clear. Any time information about a new entity becomes available we must revise all the axioms on the representation. *Argumentative Systems*'s devolvement is based on previous research on Logic Programming, Nonmonotonic Reasoning. *Argumentation* has obtained important results, providing powerful tools for knowledge representation and some aspects of Commonsense reasoning. In this sense $DeLP$ [3] was developed. $DeLP$ is a formalism that combines results of Logic Programming and Defeasible reasoning.

In general research on Argumentative Systems and Temporal Reasoning is made independently. The systems developed considered only one of this ways of reasoning, although there are several scenarios where both are needed to grant a correct solution. These research areas both has ingerence on commonsense reasoning, but this particular form of reasoning has its basis as any kind other kind. According to Mueller [11], commonsense reasoning should count with these fundamental concepts:

- *Representation:* The language must be able to build a representation of scenarios in the world and also must represent commonsense knowledge about the word. The representation can be made through some data structure or sentence based language. The representation should facilitate automated reasoning, because is the main goal. Usually commonsense knowledge is represented through rules known as "commonsense law of inertia". This set of law tries among other things to solve the *frame problem i. e.* what remains equal after the occurrence of an action.

- *Commonsense entities:* The method must represent objects, agents, time-varying properties, events and time. This last two entities are crucial because they establish how the information changes.

- *Commonsense domains:* The method must represents and reason about time, space, and mental states. Must also deal with object identity, be able to determine when we are talking about an specific object.

- *Commonsense phenomena:* The method must address the common sense law of inertia, release form the commonsense law of inertia, concurrent events with cumulative and canceling effects, context-sensitive effects, continuous change, delayed effects, indirect effects, nondeterministic effects, preconditions and triggered events.

- *Reasoning:* The method must specify must specify a process for reasoning about the representation of scenarios and commonsense knowledge. The method must support default reasoning, temporal projection, abduction, and postdiction.

Taking these on consideration we can see that argumentative systems has left apart *events* and *time* from their representation. They do not take on consideration these special entities and their effect over reasoning. $DeLP$ is a particular argumentative system [3] which deals with arguments very well but has the problem stated before. We proposed as investigation line to extend the abilities of $DeLP$ in order to consider aspects of temporal reasoning.

Looking at the problem form other point of view, we can see that the main languages developed on Temporal Reasoning do not consider argument notion. Although they do consider aspects of default reasoning. The most popular and used language are *Event Calculus* and *Situation Calculus*. They deal with incomplete information trough the use of *circumscription*. This strategy grant only one answer an annulate any possibility for discussion. The beauty in argumentation is that makes possible more that one answer and establish some kind of debate, which is needed on multiagent environments or deliberation.

The paper is structured as follows, in section 2 we present the basics on argumentation systems particularly $DeLP$. In section 3 we introduce temporal reasoning along with *Event Calculus* and *Situation Calculus*. Finally in section 4 we present the state of advance in the investigation line presented.

2 ARGUMENTATIVE SYTEMS

In general, an argumentative system counts with five elements, at least in the abstract layer:

1. *Underlying logical language:* in this particular case we need a temporal-logic language, we choose *Event Calculus*.

2. *Argument definition*

3. *Conflict and rebuttal among arguments*

4. *Argument evaluation*

5. *Notion of defeasible logic consequence:* again in this case it must be *defeasible temporal-logic consequence*

In many cases, the five above mentioned elements are not explicitly defined because they are clearly not independent. In fact dependencies among them allow the identification of four fundamental layers [13] in argumentative systems.

- *Logical Layer:* It comprises language definition, inference rules and argument construction.

- *Dialectic Layer:* This layer both involves the definition of conflict between arguments and formalizes the way of solving those possible conflicts.

- *Procedural Layer:* Defines arguments interchange.

- *Strategic Layer:* Present heuristics for argument selection during a debate based on maximizing success possibilities.

Defeasible Logic Programming ($DeLP$) is an argumentative systems that considers the layer expressed above. As a matter of fact, $DeLP$ is a formalism that combines results of Logic Programming and Defeasible Argumentation. $DeLP$ provides the possibility of representing information in the form of weak rules in a declarative manner, and a defeasible argumentation inference mechanism for warranting the entailed conclusions. In $DeLP$ an argumentation formalism will be used for deciding between contradictory goals. Queries will be supported by arguments that could be defeated by other arguments. A query q will succeed when there is an argument $\mathcal{A}$ for q that is warranted, *i. e.* the argument $\mathcal{A}$ that supports q is found undefeated by a warrant procedure that implements a dialectical analysis. The defeasible argumentation basis of $DeLP$ allows to build applications that deal with incomplete and contradictory information in dynamic domains. Thus, the resulting

approach is suitable for representing agent's knowledge and for providing an argumentation based reasoning mechanism to agents.

$DeLP$ adds the possibility of representing information in the form of weak rules in a declarative manner and a defeasible argumentation inference mechanism for warranting the conclusions that are entailed. Weak rules represent a key element for introducing defeasibility and they are used to represent a defeasible relationship between pieces of knowledge. This connection could be defeated after all things are considered. General Common Sense reasoning should be defeasible in a way that is not explicitly programmed. Rejection should be the result of the global consideration of the available knowledge that the agent performing such reasoning has at his disposal. Defeasible Argumentation provides a way of doing that.

$DeLP$ language is defined in terms of three disjoint sets: a set of facts, a set of strict rules and a set of defeasible rules. In DeLP's language a literal "L" is a ground atom "A" or a negated ground atom "$\neg A$", where "$\neg$" represents the strong negation.

$DeLP$ [3] is a language developed in term of three disjoint sets: a set of *facts*, a set of *strict rules* and finally one of *defeasible rules*, where

- A *fact* is a literal, i.e. a ground atom, o a negated ground atom.

- A *strict rule* is an order pair, denoted as "$Head \leftarrow Body$", whose first member is a literal and the second one, $Body$, es finite set of literals. A strict rule can also be written as:

$$L_0 \leftarrow L_1, \ldots, L_n (n > 0$$

where L_0 is rule's $Head$ and each $L_i, i \geq 0$ is a literal.

- A *defeasible rule* is also an order pair, noted as $L_0 \prec L_1, \ldots, L_n$. Again L_i is a literal and $i \geq 0$

Notice that strict negation may affect any literal, in particular may affect L_0, i.e. any rules *Head*. At simple sight the only difference between strict and defeasible rules is the way they are denoted, although their meaning is clearly different. In the first kind there are no doubts about the conclusion expressed on the rule, while in the other ones we only assure that we have a "good feeling" about the conclusion but we can not be completely sure about it.

3 TEMPORAL REASONING

In the last decades logic programming, which is extremely related with temporal reasoning, was developed in a notable way [7]. But the classical model of logical programmes, based on Horn clauses [20], is not good enough in to represent certain model of change. This models require an extended language and as a consequence a new computational approach, suitable for the new language. To overcome this limitations of traditional logic some non-logic constructors, annotations or especial predicates, were introduced. These languages and their implementations are of mayor help to the area. In this category there very well known languages such as *Event Calculus* [6, 19, 10, 9, 18] and *Situation Calculus* [16, 12]. These languages are very efficient but they are based on a non-standard logic. Which means that a program could not be interpreted only by its specification. Another way to avoid the limitations of traditional logic programming is the use of temporal logics. In this sense modal and intentional logics are used. As a result many languages appears, some are purely declarative while other count with operational semantics.

In this kind of reasoning we can choose different language according to what conception of time we need. There are different ways to conceive time, conceptions that are borough from philosophy. We can think in linear or branching time, discrete or dense time, etc. Another aspect is on the spot when we combine time an actions. We can think time as an entity were events take place. Or you can think on events as a entity and time can be seen only as a collateral effect of events occurrence. You can get more presise information about different time conceptions at [15, 14, 17, 1, 8, 4].

We center our attention in the first group, particularly on *Event Calculus* and *Situation Calculus* because they are very representative in commonsense reasoning area. This is so, because they consider the fundamental concepts required by this kind of reasoning [11].

3.1 Event Calculus

Event Calculus was introduced in the eighties by Kowalski and Sergot as a logic programming formalism to represent event and their effects [6]. Many dialects have been developed since then, e.g. [18, 10, 9]. In the original language events initiates time periods during which properties hold. Since a property or "fluent" is initiated it holds, unless it is terminated by the occurrence of an event. In Kowalski and Sergot version, a discrete time ontology was chosen to indicate changes. A particular extension of the language is required in order to represent continuous characteristics. Most known extensions of this calculus were developed by Shanahan [18].

In general it is a logical mechanism capable of making inferences to determine *what is true when*
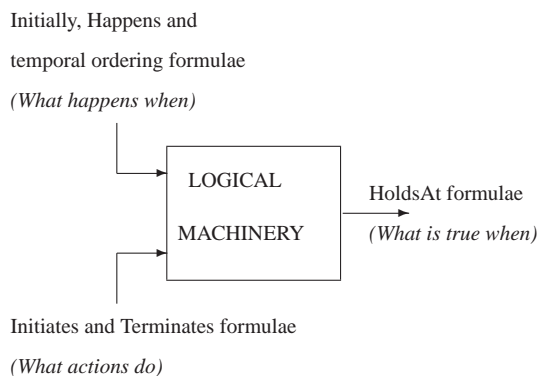
Initially, Happens and

temporal ordering formulae

(What happens when)

LOGICAL

MACHINERY

HoldsAt formulae

(What is true when)

Initiates and Terminates formulae

(What actions do)

Figure 1: How Event Calculus works out

from *what happens when*(knowledge about the state of the world) and *what actions do* (effect of an action on the world). The logical machinery includes arithmetic to set a relation between time references. The kind of arithmetic involved depends on the selected temporal ontology. The basic ontology of the calculus are *actions* or *events*, *fluent* and *time points*. A *fluent* is anything whose truth value is subject to change over time. It could be a quantity such as "temperature in a room" or "amount of liquid in a bottle" whose numerical value is subject to variation, or a proposition such as "it is sunny" whose truth value change from time to time. The predicate deals basically with propositional fluents although the other ones are allowed in some dialects. Another important issue in the choice of the ontology is the choice of the predicates. The main predicates used on a simple version of *Event Calculus*, SEC, are:

$$
\begin{aligned}
&happens(E,T): &&E \text{ takes place on } T. \\
&holdsAt(F,T): &&F \text{ holds at } T. \\
&initiates(E,F,T): &&F \text{ starts to hold after } E, \\
& &&\text{and is not freed on } T+1. \\
&terminates(E,F,T): &&F \text{ ceases to hold after} \\
& &&E \text{ at } T. \\
&releases(E,F,T): &&F \text{ is not subject to inertia} \\
& &&\text{after } E \text{ at } T \\
&initiallyP(F) &&: F \text{ holds form time zero.}
\end{aligned}
$$

where E represents events, T time moments and F fluents. Calculus complete axiomatization depends on time ontology. For example if we consider a discrete ontology, we can use ontology presented by Mueller [10] or more completely from Miller and Shanahan research [9].

Reasoning mechanism uses circumscription to deal with default information and to solve incompleteness.

The latest versions of this calculus used as reasoning technique a first-order logic automated theorem proving. Previous versions used propositional satisfiability or abductive logic programming.

3.2 Situation Calculus

Situation Calculus were developed by Ray Reiter and his research team [16]. This calculus is a second order language designed for representing changing worlds. All changes to the world are their result of an action, so a possible world history , is a sequence of actions represent through a first-order term, called situation. A situation is like a snapshot of a possible world where we set what holds there. The properties that may holds depends on the initial situation and the changes performed by event occurrence. This initial situation is represented by an empty sequence of actions. There is a distinguished function $do(\alpha, s)$ who denotes the situation that success situation s if the action performed is α. In Situation calculus actions are denoted as function symbols while situations as first-order terms. The values of relations and functions may vary from situation to situation. Relations with this behavior are called *fluents*, while functions are called *functional fluents*.

To determine the behavior of the actions on the system and reason with the specification, we need to formalize axioms. We will need axioms to determine if an action is possible on a situation, this action checks action precondition. We will use axioms, also, to indicate what fluents change after an action takes place and which remains equal (effect axiom, frame axiom). There is only one axiom of each kind for every event. The undesirable aspect of this is, that any change on the specification mean a revision of all the axioms in the specification.

The first version of this calculus do not consider time in an explicit way. The evolution of time is hidden on situations, and events occurrence. Unfortunately this is not enough for our current investigation line, so we need to enrich the calculus with *annotations*. The annotations introduced time in a explicit way to the previous scenario. Now we can get track of when an event takes place independently of how many situations we pass form the initial one.

4 PREVIOUS RESULTS AND CONCLUSION

Argumentative systems as $DeLP$ has been important for the grow of commonsense reasoning area. But they are short for today's definition of this kind of reasoning. The main fault is that its representation language although is based on sentences do not consider actions and time in an explicit way. If events and time behave as any other object we can use $DeLP$

as it is now. Unfortunately these two elements has a completely different semantic form other objects in the system. Events and when they took place creates a huge impact on when properties holds. The same fluent can be true and false now, but in different moments of time. Commonsense law of inertia plays a different role now. So we need to extend $DeLP$ in order to solve problems associated to the temporal aspects of the information. Timed information is in certain ways more complex. The complexity appears form the possible interactions between fluents that holds in different moments. This duality in temporal systems is common while in non temporal systems never takes place.

The first goal we what to achieve is to make a combination between $DeLP$ and *Event Calculus*. Research in these directions is currently being pursued. Later on we will try to change the temporal language selected for *Situation Calculus*. We can take all the profits we can from the first combination.

If we analyze the problem form the other perspective, we can think in this investigation line as a proposal to change the reasoning mechanism of *event and/or situation calculus*. Changing the current one based on circumscription from a more open mechanism as argumentation. Circumscription is appreciated for being closed to logic programming, but forces to considerate any abnormal situation. Any time we want to infer something we must ask for its normality, while default reasoning released us from that. Of course the final result is the same, but in the version we are working on we will give more freedom to the representation, and we will also offer the possibility to discuss about the possible results. This last is important when we deal with agents and negotiation.

We are currently analyzing the effects of using *Event Calculus* as the representation language for a temporal version of argumentation. Particularly the role of the commonsense law of inertia on argument construction is being pursued.

References

[1] J. F. A. K. Van Benthem. *The Logic of Time*. Reidel, 1983.

[2] Carlos I. Chesñevar, Ana G. Maguitman, and Ron Loui. Logical models of argument. *ACM Computing Surveys*, 4(32):337–383, 2000.

[3] Alejandro J. Garcia and Guillermo R. Simari. Defeasible logic programming: an argumentative approach. *TPL*, 4:95–138, 2004.

[4] Jean Louis Gardies. *Lógica del Tiempo*. Paraninfo, Madrid, 1979.

[5] Steve Hanks and Drew Mc Dermott. Non-monotonic logic and temporal projection. *Artificial Intelligence*, 33:379–412, 1987.

[6] Robert Kowalski and Marek Sergot. A logic-based calulus of events. *New Generation Computing*, 4(1):67–895, 1986.

[7] John W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, New York, third edition edition, 1995.

[8] Robert McArthur. *Tense Logic*. Reidel, 1976.

[9] R. Miller and Murray Shannahan. Some alternative formulations of the event calculus. *Computational Logic: Logic Programming and Beyond*, 14:703–730, 2004.

[10] Erik T. Mueller. Event calculus reasoning through satisfiability. *Journal of Logic and Computation*, pages 452–490, 2002.

[11] Erik T. Mueller. *Commonsense Reasoning*. Morgan Kaufman an imprint of ELsevier, 2006.

[12] J. Pinto and R. Reiter. Temporal reasoning in logic programming: A case for the situation calculus. In *Proceedings of the Tenth International Conference on Logic Programming*, pages 203–221, 1993.

[13] Henry Prakken and Gerard Vreeswijk. Logics for defeasible argumentation. In Dov Gabbay (ed.), editor, *Handbook of Philosophical Logic*. Kluwer Academic Publishers, 1998.

[14] Arthur Prior. *Past, Present and Future*. Clarendon Press, 1967.

[15] W. B. O. Quine. *Word and Object*. MIT Press, 1960.

[16] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

[17] Nicholas Rescher and Alasdair Urquhart. *Temporal Logic*. Springer-Verlag, 1971.

[18] Murray Shanahan. Representing continuous change in the event calculus. In *Proceedings ECAI 90*, pages 598–603, 1990.

[19] Murray Shanahan. *Solving the Frame Problem*. MIT Press, 1997.

[20] M. van Emdem and Robert Kowalski. The Semantics of Predicate Logic as a Programming Language. *Journal of the Association for Computing Machinery*, 23(4):733–742, 1976.