Logical Formalization of Concurrent Actions for Multi-Agent Systems

Mikhail Soutchanski and Eugenia Ternovskaia

Dep. of Computer Science, 10 King's College Road, SF3303
Toronto University, Toronto (Ontario), M5S 1A4, Canada
e-mail: (mes,eugenia)@ai.toronto.edu

Abstract. This paper presents a logical approach to formalization of some aspects of multi-agent systems in an incompletely known world. The contribution on this regard is as follows.
We formalized reasoning about concurrent actions inside the situation calculus framework which has the sufficiently clear and well-understood semantics to express different aspects of agent theory.
Conclusions can be drawn by default about results of a group of concurrent actions which may include unknown or unspecified actions.
The approach allows to examine all models corresponding both to different assumptions about possible failures in performing of one or another action, and to assumptions whether a particular action was executed really. This provides an opportunity to predict the consequence of joint activity of agents operating in complex environment and to explain apparently unexpected changes.
These results are achieved by choosing a suitable circumscription policy. The approach provides a general and formal framework for design and verification of wide class of agent-oriented systems such as telecommunication systems, network management, and evolving distributed data bases.

1 Introduction

The study of multi-agent systems composed from reasoning agents and embedded in a changing environment has been receiving considerable attention. An example of such a system is a team of mobile robots either jointly committing to a common goal or having competitive interests. Each member of a team to achieve both a shared goal and its own aims should be able to take into accounts its own and others states of knowledge, incorporates the plans of other agents into its own, and to reason about a result of joint actions.

Among the nowadays agent-based products are agent software that schedules meetings, responds automatically to incoming e-mail, delivers the message to the addressee's computer thus protecting programmers from many of the complexities of network protocols. It appears that agent technology will present a powerful example for remote computing, especially in a client/server environment. The message has to be smarter there whenever there is nobody to guide it.

Several mathematical formalisms are used for representing and analyzing different issues of agent systems. We adhere to logical approach because it will permit to combine the strategic analysis of agent interests, which is a part of the traditional game theory, with the analysis of argumentation, used by agents to overcome a conflict situation. Moreover, the logical approach allows naturally take into account constraints on cognitive abilities of agents. It is well known that 'complete rationality' is an extreme assumption and can lead to implausible predictions about an outcome in a repeated game. We suggested here as a way of dealing with 'bounded rationality' to restrict explicitly the logical language used for representation of agent beliefs and communications. Interesting reader can find in other papers from this volume aternative approaches to formalisations of multy-agent systems ([3, 23],etc).

We have focused our attention only on the development of the simple logical formalization of reasoning about actions executed simultaneously. As the starting point we used the situation calculus [15] which is now one of the most popular approaches for representing and reasoning about actions [2, 8, 14, 18, 22]. Moreover, recently it was conceived that some of its apparent limitations can be overcame easily [4]. The attractiveness of the situation calculus stems from its clear ontology and its simplicity. From the formal point of view, we use a small fragment of a MSFOL (many-sorted first order logic) with appropriate circumscription policies. We stick with a sorted logic given its general suitability in reducing the search space for many problems. However, the question how this choice can help us to develop a more effective circumscription prover is beyond the scope of our paper.

We solved the following problems. The first is: How should causal rules refer to simultaneous actions so that if we knew that some agents perform the group of actions which includes unspecified one, the appropriate causal rule nevertheless would be fired to derive default conclusions ? Such conclusions will agree with the intuition that the engagement in an activity (simultaneously with other actions) whose influence on the world is unknown, by default cannot prevent inferences about the results of other actions from deriving. These inferences will be defeasible and can be withdrawn in the presence of new causal rules with contrary specifications. For instance, in accordance with some ideological, theological or social doctrines all of us are unconsciously involved in a lot of activities. Nevertheless, it does not preclude us from reasoning about everyday actions. The approach we proposed put no restrictions on how many such actions unspecified by causal rules of the theory are performed, and moreover, its flexibility is achieved without complications of the causal rules.

While dealing with the first problem we concern ourselves with predictions, the second problem aroused from the need to gain explanations for obtained observations. Thus, the second problem is: how to recognize the actions performed by the multi-agent system under incomplete knowledge about its evolution ? The incompleteness is conditioned both by the intention to allow for the possibility that actions may fail to have their expected effects and by the lack of knowledge about what particular actions determined the observed phenomena. We proposed an elegant solution to this problem by applying the ideas of [2, 12, 13].

Our solutions allow also to derive easily new values of all properties of the world which can change after an action have been executed, in particular, values of those properties which were not influenced by the action. Thus, we cope with the *frame problem* sticking to the circumscription. Our method solves also the *ramification problem*. It arises when a theory includes a set of static constraints on the properties of the world. The problem is how all implicit consequences of actions can be found without explicit applications of causal rules (i.e. only from static constraints). We indicate also a way of dealing with *qualification problem*, which is the problem how all sufficient preconditions for each action could be enumerated concisely.

Despite we have not consider very important issues of intricate interaction of knowledge, intentions, collaboration, and actions, we will demonstrate briefly how agent beliefs and communication can be represented. We think this justifies our adherence to the situation calculus. In this respect, we follow a long standing tradition of syntactic approach to denoting agent's beliefs (first order language expressions) by terms [7, 16, 17].

An alternative to syntactic approach is representing propositional attitudes by means of a multi-modal language. However, first order formalizations have more attractive features in comparison with possible-worlds approach, so called due to Kripke-style semantics of possible worlds for a modal logic of knowledge and belief. This approach suffers from the *logical omniscience* problem: agents are so ingenious that they know all valid formulas and they know all the logical consequences of their knowledge. It is certainly not appropriate because real agents have resource limitations on their reasoning process. The second objection is that modal logics do not explicitly represent relationships among beliefs, particularly the process how one belief is caused by other beliefs and do not distinguish derivational capabilities of different agents [7]. A good example of multi-agent systems composed from reasoning agents with varying degree of intelligence is a client/server environment: client should not be so clever as a server. Our preferences are thus close to framework proposed by [24].

We follow a certain tradition, which is briefly described in section 2. After an introduction of sorts for object variables, function and predicate symbols in the section 3 we demonstrate in section 4 our method of axiomatizing concurrent actions. In particular, this section contains hints how axioms should be formulate in general case. In the section 5 we outline the circumscription policies. For expository purposes we illustrate our considerations by three examples in the section 6. We will focus our attention on reasoning forward and backward in time. Section 7 compares our approach with other works. The last Section 8 notes some possible extensions and summarizes our proposal.

2 Preliminaries

In this section we briefly describe approaches from which our work originates.

Our approach to concurrent actions representation stem from the idea of J.Weber [22]. He proposed to represent the dynamic component of the situational calculus ontological scheme by two distinct ontological entities. The first of them is a single action, second one is an operator. Roughly speaking, an operator is simultaneously performing actions, however it is an independent entity. Taken this step we can drastically simplify an axiomatization, because under this point of view on concurrent changes there is no need to add axioms about the inheritance of the properties of single actions by the operator which is composed from them. The operator is related with its compound actions by the binary predicate $Type$. Thus, J.Weber proposed to detach the type of an action from its name and make it into a predicate.

The predicate $Type$ will occur in causal rules and in the so-called scenario. To prevent causal rules from undesirable applying, $Type$ should be circumscribed. Without such circumscription it would be possible for some of the models to permit superfluously wide extents of the predicate.

We will use the *pointwise circumscription* [10] to deal with the frame problem.

We take J.Crawford and D.Etherington's [2] suggestion into consideration. [1] They introduced two abnormality predicates to capture the following two ideas.

First, when an action typically has certain effects, but the list of all combinations of circumstances in which the action may fail to have these effects is too long to be generated explicitly. The failures to perform actions are described in our proposal by the predicate $Wrong$, keeping in mind that usually such failures occur rarely.

Second, whenever actions have indirect effects, the task to capture by an axiomatization the exact effects of successful actions can be infeasible. To address the last problem, persistence axiom is usually added that state that if a particular value is not known to change as the result of an action, then one may assume that it persists. The abnormality predicate Ab is used to express this idea.

The extents of both these predicates are supposed to be as minimal as possible, i.e. logical models with fewer violations of default assumptions are preferred over those with more violations. Although one or another violation must take place when we have to explain certain observations, a straightforward axiomatization gives no hints for resolving a conflict to the benefit of one over another. The conflict has be avoided by choosing an appropriate circumscription policy.

The method is also applicable if a specification of a system is incomplete. For example, let one observes that a truth value of the particular property remains the same but s/he doesn't know all those actions which agents have performed concurrently. First, in order to explain the observed phenomena, s/he can form a hypothesis that some action which is known definitely as occurring didn't has its expected effect on the property. Second, s/he can assume that an unforeseen action took place concurrently with the actions which definitely occurred and that action was the reason for the observed phenomena.

[1] The idea of "filtering" appropriate models via special axioms encoding observations has been proposed by E.Sandewall in [12].

3 Language

We describe in this section the language of situation calculus and briefly review the syntactic approach to representing beliefs.

3.1 Situation Calculus

We will consider a subset of MSFOL [2]. In the sequel, we would like to express that a set of facts holds or does not in a situation. We would like also to describe agent's beliefs (explicitly represented in a subset of our language), deductions and communications. In order to allow this, the syntax of the underlying language must includes terms for formulas so that they can appear as arguments in predicates [7, 16, 17]. A first-order language so endowed with terms is called usually the *reified* language [3].

In accordance with a long standing tradition, in our notation we distinguish two sorts of reified formulas. Terms (object variables, constants) for truth-valued fluents (f) representing external fluently varying properties of the world are used without a quotation marks in a formulas of our language. Terms for beliefs, content of communicative acts, and other information-related entities (i) are obtained by applying a quotation construct to a formula. We use also object variables of other sorts: for situation (s), for operators (o), for actions (a), for agents (c). For each sort there is a finite set of constants denoting a corresponding set of objects in a domain.

We use the ordinary situation-valued function symbol Res having operator term and situation term as arguments. For example, $Res(O, S_0)$ denotes a situation resulting from the execution of the operator O in the initial situation S_0.

The binary predicate constant symbol $Holds(f, s)$ asserts that a property f holds in a situation s: $Holds(Communication(C_1, C_2), Res(O_2, Res(O_1, S_0)))$ means that agents C_1 and C_2 communicate in the situation achieved as the result of performing the operator O_2 in the earlier situation $Res(O_1, S_0)$.

Even if fluents have equivalent truth values everywhere, they may denote different properties of the world. Note that otherwise, if we were to include the axiom $Holds(f_1, s) \equiv Holds(f_2, s) \supset (f_1 = f_2)$, our theory would be monadic second order, because fluents are universally quantified.

We use the ordinary ternary predicate constant symbol $Ab(f, o, s)$ to assert that a fluent f is abnormal (i.e., its value does not persist) after the performance of an operator o in a situation s.

The binary predicate constant symbol $Type(o, a)$ means that an operator o contains, in particular, an action a, e.g. $Type(O_1, Send(C_1, "Message", C_2)$ asserts that the action $Send(C_1, "Message", C_2)$ belongs to the operator O_1, where C_1 is the sender, C_2 is the recipient, and Message is a formula.

[2] Variables begin with lower-case letters. Constants, function symbols, predicate symbols begin with upper-case letters. Unbound variables are universally quantified.

[3] alternative name is the *thingified* language, due to J.McCarthy

The predicate $Wrong(o, a, s, f)$ is used further to cope with several problems. Please note that whenever we apply the adjective 'wrong' to characterize an effect of an action a (which belongs to an operator o) on a fluent f at a situation s, we describe a state of affairs when the action was canceled out by other actions (which belong to the same operator) or failed due to some reasons.

The auxiliary binary predicate constant symbol $Earlier(s_1, s_2)$ taking situation terms as its argument says that a situation s_1 preceds a situation s_2.

3.2 Syntactic Approach to Encode Agent Beliefs.

To justify our choice of the situation calculus as a framework for formalizing concurrent actions in agent systems, we demonstrate in brief how the syntactic approach works. [4]

We may enrich our language by terms:

by appropriately sorted term constants. They are designated by applying quotation marks to object constants ("C", "S");

by the standard naming function;

for each predicate letter we assume corresponding function symbol designated by applying quotation marks to the predicate constant ("$Holds$");

by Boolean constructors (with obvious axiomatization): and, or, neg, imp. These function symbols allow to put together more complicated formulas starting from atomic ones.

To avoid cumbersome expressions, it is convenient to use sense quotes " " enclosing a $Formula$ as a syntactic abbreviation for the term that is intended. For example, we write "$Holds(F_1, S_0) \supset Holds(F_2, S_0)$" instead of $imp(\text{"}Holds\text{"}(\text{"}F_1\text{"}, \text{"}S_0\text{"}), \text{"}Holds\text{"}(\text{"}F_2\text{"}, \text{"}S_0\text{"}))$.

Note to simplify our definitions we consider as beliefs only formulas being boolean expressions obtained from ground atomic formulas.

The fluent-valued function symbol Bel having agent term and information term as arguments denotes a specific fluent expressing a fact that the agent believes in the information. For example, $Holds(Bel(C_1, \text{"}Inf\text{"}), S_0)$ means that the agent C_1 possesses a belief "Inf" at the initial situation (where Inf is a formula).

We can address the 'bounded rationality' imposing explicit restrictions on terms of information sort. For example, let beliefs be only Horn formulas without function symbols. Then, we may put *a priory* restrictions on the amount of disjuncts in a Horn formula and on the depth of beliefs nesting (beliefs about beliefs). For example, let belief be Horn formulas composed from no more than, say, 21 disjuncts, such that each belief may contain inside no more than one level of belief nesting. [5] We can ascribe to the agents certain (even varying from one

[4] Interested reader can find further details in [7, 16, 17, 24].

[5] Thus, an agent is able to believe that someone believes that "$Inform$", however this agent is not able to believe that someone believes that the agent believes that "$Inform$".

agent to another) deductive power, if we add to a logical theory suitable axioms, for example,

$$
\begin{gathered}
Holds(Bel(c, \text{"}Holds(F_1, S_0)\text{"}),\ s)\ \& \\
Holds(Bel(c, \text{"}Holds(F_1, S_0) \supset Holds(F_2, S_0)\text{"}),\ s) \\
\supset Holds(Bel(C_1, \text{"}Holds(F_2, S_0)\text{"}),\ s)
\end{gathered}
$$

Due to the restrictions, even if agents have a set of beliefs and some limited deductive abilities, they cannot derive facts which are expressed by too sophisticated Horn formulas. One of the most important problems here is *the knowledge precondition problem* [17]. Because of space limitations, we will not discuss these questions deeper.

4 Axioms

We divide axioms on several groups in accordance with their intention. We will discuss here only general case. Illustrating examples will be considered further.

The first group of axioms contains effect axioms (causal rules) with predicate $Type$, that imposes constraints on an operator variable.

$$
\begin{gathered}
[\neg]Holds(F_1, s)\&...\&[\neg]Holds(F_k, s) \\
\&Type(o, A_1)\&...\&Type(o, A_n) \\
\&\neg Wrong(o, A_1, s, F_{k+1})\&...\&\neg Wrong(o, A_n, s, F_{k+1}) \\
\supset [\neg]Holds(F_{k+1}, Res(o, s))
\end{gathered} \tag{1}
$$

The second group involves axioms formulating reasons why an action may fail to produce its intended effect:

- Whenever an operator includes an action or a group of actions, the effect of another action on a fluent F is canceled

  $$Type(o, A_1)\&...\&Type(o, A_n) \supset Wrong(o, A_{n+1}, s, F)$$

  Example, the actions Cl (close) and Op (open) cancel each other's effect on the fluent F (a closed file) whenever they occur in the same situation:

  $$Type(o, Op) \supset Wrong(o, Cl, s, F), \quad Type(o, Cl) \supset Wrong(o, Op, s, F)$$

- Whenever some fluents hold in a situation and some actions belong to the operator which is executed by a group of agents in a situation, a certain action must fail in this situation:

  $$
  \begin{gathered}
  [\neg]Holds(F_1, s)\&...\&[\neg]Holds(F_k, s) \\
  \&Type(o, A_1)\&...\&Type(o, A_n) \\
  \supset Wrong(o, A_{n+1}, s, F)
  \end{gathered} \tag{2}
  $$

  For example, a duplex line L_5 cannot be assigned to the computer C_4 for communication with the server C_1, if C_1 and client C_7 occupy currently this line ($Occ(Computer, Line)$) for their communication:

  $$
  \begin{gathered}
  Holds(Comm(C_1, C_7, L_5), s)\&Type(o, Occ(C_1, L_5))\&Type(o, Occ(C_7, L_5)) \\
  \supset Wrong(o, Assign(C_4, L_5), s, Comm(C_1, C_4, L_5))
  \end{gathered}
  $$

– Whenever some particular fluents hold in a situation, a certain action must fail in this situation:

$$Holds(F_1, s)\&...\&Holds(F_j, s)\&\neg Holds(F_{j+1}, s)\&...\&\neg Holds(F_k, s) \\ \supset Wrong(o, A, s, F) \tag{3}$$

For example, if a message "M" has been sent but not received, then the action of delivering the message M has been cancelled out:

$$Holds(Sent("M"),\ s)\ \&\ \neg Holds(Received("M"),\ s) \\ \supset Wrong(o, Deliver, s, Received("M"))$$

Each theory of actions under consideration includes the inertion axiom with the abnormality predicate:

$$\neg Ab(f, o, s) \supset ((Holds(f, s) \equiv Holds(f, Res(o, s))) \tag{4}$$

This axiom says informally, that values of those properties f are abnormal in the current situation s which don't persist in the next situation occurring as a result of performing an operator o. Roughly speaking, after the circumscription of the predicate Ab as much fluents will be persistent as it is possible.

The third group is comprised by a set of unique names axioms; for fluents and operators:

$$F_1 \neq F_2 \neq ... \neq F_n \qquad O_1 \neq O_2 \neq ... \neq O_r$$

and for situations: [6]

$$\begin{gathered} Earlier(s, Res(o, s)) \\ Earlier(s, s_1)\&Earlier(s_1, s_2) \supset Earlier(s, s_2) \\ Earlier(s, s_1) \supset s \neq s_1 \end{gathered} \tag{5}$$

And finally, the sixth group axioms is formed by scenario's assertions and, consequently, is specific only for an example under consideration:

$$[\neg]\ Holds(F_i, S_0), \quad i = 1, ..., n. \tag{6}$$

$$Type(O_j, A_l), \quad l = 1, ..., r. \tag{7}$$

$$[\neg]\ Holds(F_i, Res(O_k, Res(O_{k-1}, ..., S_0)))...)), \quad i = 1, ..., l\ , \tag{8}$$

for a certain fixed sequences $O_1, O_2, ..., O_k$.

Because axioms (8) refer to properties of the system after a set of operators have been performed, these axioms will be named as *observations*.

Let us denote by A_1 the conjunction of all axioms from this section besides those axioms (6) - (8) which belong to scenario.

[6] This axioms are used here in the form proposed by F.Lin&Y.Shoham

5 Circumscription Policies

As usually, whenever some circumscription policy is formulated we assume that sort and arity of each predicate variable exactly correspond to sort and arity of the initial predicate constant.

5.1 Preliminary Definitions

We will denote (see [9, 10]) by $Circum(A(P,Z);P;Z)$ the *global circumscription of the predicate P in the formula $A(P,Z)$ with Z allowed to vary* to represent the formula:

$$A(P,Z) \ \& \ \neg\exists p,z\ (A(p,z)\ \&\ p<P) \tag{9}$$

where z is a list of predicate and/or function variables whose arities equal to arities of corresponding letters from the tuple Z, and $p<P$ denotes

$$\forall x(p(x)\supset P(x))\ \&\ \neg\exists x(P(x)\supset p(x))$$

.

We remind that the corresponding form of *pointwise circumscription*

$$A(P)\ \&\ \neg\exists x,z\ [P(x)\ \&\ A(\lambda y(P(y)\&\ x\neq y),z)]$$

is denoted by $C_P\ (A(P,Z);Z)$.

We remind also that [10] defined *the pointwise circumscription of P in A(P,Z) with the predicate Z allowed to vary only on the part V of its domain*, where V is a λ-expression $\lambda uV(u)$ of the same arity as $Z(u)$, which has no parameters and contains neither P nor Z:

$$A(P)\ \&\ \neg\exists x,z\ [P(x)\ \&\ \forall u(\neg V(u)\supset z(u)\equiv Z(u))\ \&\ A(\lambda y(P(y)\&\ x\neq y),z)].$$

In the sequel we will need more flexible circumscription policies that also were defined by [10]. Let V be λ-expression $\lambda xuV(x,u)$ whose arity equals the sum of the arities of P and Z, and V_x is the function $\lambda uV(x,u)$ which maps every value of x into the set of all values of u satisfying $V(x,u)$. Then, whenever Z may vary only on that part V_x of its domain which depends of the point x where P is minimized, this circumscription is denoted by $C_P\ (A(P,Z);Z/V)$):

$$A(P)\ \&\ \neg\exists x,z\ [P(x)\ \&\ \forall u(\neg V_x\supset z(u)\equiv Z(u))\ \&\ A(\lambda y(P(y)\&\ x\neq y),z)]. \tag{10}$$

If instead of Z we may vary some values of the predicate P itself while minimizing its value at some point, this *pointwise circumscription of P in A(P) with P itself allowed to vary on the domain V_x* is $C_P(A(P);P/V)$:

$$A(P)\ \&\ \neg\exists x,p\ [P(x)\ \&\ \neg p(x)\ \&\ \forall u(\neg V_x\supset p(u)\equiv P(u))\ \&\ A(p)]. \tag{11}$$

Here p is a predicate variable similar to P, V is a λ-expression $\lambda xuV(x,u)$ which has no parameters and does not contain P. The second term of (11) says that if $P(x)$ is *true* it is impossible to change its value to *false* without losing the property $A(P)$, even if the values of P will change arbitrary on the domain V_x.

Note, that although circumscription gives rise to a second order formulas, there is a wide class of circumscribed theories of actions which are either equivalent to a first order formula [5] or to a formula in a more expressive *fixpoint logic* obtained by adding the *least fixed point formation rule* to first order logic [6].

5.2 Description of Our Circumscription Policy

Given a complete description of some multi-agent system, as a set of statement in first-order logic, we can predict possible behaviors of the system by listing facts which are true in each situation achievable from the current one.

That task is more difficult if we don't have the complete description of the system. This incompleteness can be caused (•) either if one allows possibility that actions may fail to have their expected effects; (•) or if one wants to take into account those actions which are not mentioned in the scenario to explain apparently uncaused changes.

Before we will proceed to the formal definition of the circumscription policy, we present its intuitive description.

1. While reasoning about an observed behavior of the system, one choose a set of hypotheses about unknown facts to fill a gap in his/her knowledge. Then he/she simulates the work of the system bearing these hypotheses in mind. Looking over all possible set of hypotheses in turn, he/she constructs gradually complete graph of transitions between states of the systems. These transition can be labeled by corresponding operators.
2. If under a particular set of assumptions the simulation of a system doesn't coincide with the observations (see axioms 8), then there is no sense to consider this set further.
3. When all inappropriate sets of hypotheses are removed one can compare all remained simulations and choose the most plausible ones. Among them will be those simulations which will not contradict to the intuition that an action, as a rule, produces its intended effect, and unrecognized actions, as a rule, occur very rarely.

The circumscription policy, used in this paper allows to carry out the process of recognizing how a system operates. Making assumptions about failures to perform one or another action corresponds in our framework to holding predicate $Wrong$ fixed. Making assumptions about what (in particular, uncanny and/or unconscious) actions have been executed corresponds to fixing the predicate $Type$.

Our minimization policy is based on [2]. We will divide the minimization process into three stages in accordance with the above informal description.

At the first stage, the predicate Ab is circumscribed pointwisely [10] with $Holds$ varying on situations other then current and having $Type$ and $Wrong$

fixed [7]. The minimization of Ab with Holds varied allows to construct full graph of transitions between states of a system, because for each situation and for each particular operator the circumscription determines values of all fluents minimizing the number of those fluents which change their values. Holding the predicates $Type$ fixed corresponds to making assumptions about what particular actions took place, and fixing the predicate $Wrong$ corresponds to assumptions about what actions failed to have their intended effects.

$$C_{Ab}(A_1;\ Holds/V_1) \equiv A_2, \quad V_1 = \lambda f, s'\ (s' \neq s);$$

Note, that the axioms of a scenario are excluded from the circumscribed theory at this stage (see remark at the end of the previous section).

At the second stage the resulting theory A_2 is augmented with the scenario axioms. This will rule out the models which do not correspond to observations.

At the third stage, the most preferable models from the remaining are chosen by minimizing $Wrong$ and then $Type$:

$$Circum(A_2\&(6)\&(7)\&(8);\ Wrong;\ Ab, Holds/V_1) \equiv A_3;$$

$$Circum(A_3;\ Type;\ Ab, Holds/V_1) \equiv A_4;$$

This last stage allows to choose the most plausible simulations which correspond to the intuition that actions, as a rule, have their intended influences and unforeseen actions (miracles) occur rarely.

6 Explanation and Prediction

In this section we consider two examples to clarify the effects of our circumscription policy. Everywhere below, we assume that a set of appropriate unique name axioms (in particular, for situations (5)) and the persistence axiom (4) are included in a theory.

Example 1. Prediction Problem

Let us consider a simple example of the theory which doesn't include statements about what fluents hold in the situations other than initial one. Such theories are concerned with prediction problems.

This example describes a toy telecommunication system where multiple actions are performed concurrently. Let us consider two connected computers which exchange messages via one 2-directional channel. If both computers C_1 and C_2 are ready to communicate and their modems send signals about readiness to each other, then the process of communication can start [8]:

[7] This policy doesn't need axioms to force the existence of all possible situations, so it can be used for reasoning about complex domains such as reasoning about space.

[8] The process *Communication* is represented here by the fluent. The general approach to logical representation of reasoning about processes extended in time is described in [21]

$$
\begin{aligned}
&Holds(Ready(c_1), s)\&Holds(Ready(c_2), s)\\
&\&Type(o, SendSignal(c_1))\&Type(o, SendSignal(c_2))\\
&\&\neg Wrong(o, SendSignal(c_1), s, Communication(c_1, c_2))\\
&\&\neg Wrong(o, SendSignal(c_2), s, Communication(c_1, c_2))\\
&\supset Holds(Communication(c_1, c_2), Res(o, s))
\end{aligned} \tag{12}
$$

Let the scenario involves the following axioms:

$$Holds(Ready(C_1), S_0) \qquad Holds(Ready(C_2), S_0) \tag{13}$$

$$Type(O_1, SendSignal(C_1)) \tag{14}$$

$$Type(O_2, SendSignal(C_1)) \qquad Type(O_2, SendSignal(C_2)) \tag{15}$$

The first two axioms from the scenario (13) describe the initial situation S_0 in which both computers are ready to start communication. The third axiom from the scenario states that the operator O_1 includes the action $SendSignal(C_1)$. The last two axioms of the scenario (15) describe the operator O_2. It is easy to see that the operator O_1 doesn't lead to any changes, because the effect axiom (12) states that the communication starts only if both computers send signals about their readiness. At the same time, after the operator O_2 have been performed by the computers, they will start communicate because all sufficient requirements are satisfied: $Holds(Communication(C_1, C_2), Res(O_2(Res(O_1, S_0)))$.

After the first stage of the circumscription we have all possible simulations of the system for each assumption about failures to perform the actions and about the types of all operators. This corresponds to determining the extent of the predicate Ab with the predicates $Wrong$ and $Type$ fixed. Our theory doesn't involve observations about situations other than initial one. So, the next step of minimization is circumscription of the predicate $Wrong$. To minimize $Wrong$ we compare all simulations under different assumptions and choose those models in which the extent of this predicate is smaller in the usual set-theoretic sense. The circumscription of the predicate $Type$ on the last stage allows to eliminate those simulations of the system which were gained from hypothesis that operators include unforeseen actions.

Let the action *Dial*(C_3,*425-3534*) unrelevant to the telecommunication system belongs to the second operator, i.e. the third computer dial a phone number. In this case our theory should include the axiom: *Type*(O_2,*Dial*(C_3,*425-3534*)).

The action *Dial*(C_3,*425-3534*) is not specified by any effect axiom of our theory. Nevertheless, the same conclusions as before about the effects of the operator O_2 can be inferred.

Example 2. Explanation Problem

This example illustrates how our approach allows to cope with explanation problem (a theory with at least one axiom which asserts what is true in a situation other than initial one). We want to recognize what particular actions have been performed.

Consider several processors of a distributed network. The processors apart from running programs can communicate by sending messages. Let a processor C_1 can perform two kinds of actions: run a program ($RunProgram(C_1)$) and send a message "inf" to another processor ($Send(C_1, "inf", C_2)$). We will consider in this example the only property describing the states of the processors: $Bel(c, "inf")$ - the message "inf" is avaiable for the processor c. The communication is described by the effect axiom (16); the axioms of scenario are (17)-(19).

$$\begin{array}{c} Holds(Bel(c_1, "inf"), s) \& Type(o, Send(c_1, "inf", c_2)) \\ \& \neg Wrong(o, Send(c_1, "inf", c_2), s, Bel(c_2, "inf")) \\ \supset Holds(Bel(c_2, "inf"), Res(o, s)) \end{array} \tag{16}$$

$$Holds(Bel(C_1, "Inf"), S_0) \qquad \neg Holds(Bel(C_1, "Inf"), S_0) \tag{17}$$

$$\neg Holds(Bel(C_2, "Inf"), Res(O_2, Res(O_1, S_0))) \tag{18}$$

$$Type(O_1, RunProgram(C_1)) \qquad Type(O_2, Send(C_1, "Inf", C_2)) \tag{19}$$

The axioms (17) describes the initial situation. The processor C_1 'believes' in "Inf" at the situation S_0 and the processor C_2 doesn't. The first operator O_1 (described by the axiom in (19)) involves the action $RunProgram(C_1)$ which is not specified by any effect axiom. The second one states that the action $Send((C_1, "Inf", C_2)$ belongs to the operator O_2. Note that the scenario includes the statement (18) representing the fact that after O_1 and O_2 have been performed in a sequence, the second processor does not receive the message "Inf". How could we explain this unexpected fact? Let us consider our circumscription policy.

At the first step of minimization we separately simulate the telecommunication system behavior under each fixed set of assumptions about what particular actions could belong to the operators and about what actions could fail to have their expected effects. Let us suppose that all actions which belong to the operators O_1 and O_2 are enumerated in the axioms (19). It is meaningless to generate new assumptions what other actions perhaps have been performed, because these assumptions will be definitely ruled out at the last step of minimization.

If we hypothesize that the action $Send(c_1, "inf", c_2)$ doesn't fail, then we can predict that C_2 receives the information "Inf". Similarly, if the sending action fails then the information "Inf" will not be available for C_2. Then we can check the resultant predictions against observations and rule out the case in which the sending was successful. The remaining model includes one violation of the assumption about normal performing of the action $Send(C_1, "Inf", C_2)$. Then, $Wrong$ is minimized to prefer those models which violate as few assumptions about normal performing of the actions as possible. As we have only one remaining model so the extent of the predicate $Wrong$ is defined. Finally, we circumscribe $Type$ and find that its extent is actually covered by the axioms (19).

Thus, the circumscribed theory implies that the action $Send(C_1, "Inf", C_2)$ fails and this fact quite well explains the observation. This proves that our approach allows to reconstruct the behavior of the telecommunication system (we cannot know why message was lost, because we have not axiomatized properties of communication channel).

7 Discussion

In this section we briefly review different approaches to reasoning about concurrent actions representation. More detailed comparison can be found in our paper [20].

The papers [1, 14] also dealt with the representation of concurrent actions inside the framework of the situation calculus has appeared practically simultaneously with [20]. The paper [1] proposed an approach based upon a specific formal language $\mathcal{A}$. In that paper, the authors provided a translation into the language of extended logic programming and prove its soundness. It is difficult to compare our approaches because our languages differs considerably. However, the paper [5] seems the most promising in this regard.

The authors of [14] also proposed the method for reasoning about concurrent actions. Their approach is similar to our one to some extent, because it is based on the situation calculus and circumscription too. Authors used the binary predicate $In(a, \{A_1, ...A_n\})$ to express that primitive action a belongs to global action $\{A_1, ...A_n\}$. This predicate plays the same role as our predicate $Type$. Because global actions which will be executed are described by explicit enumeration of what primitive actions they contain as the components, there is no need to circumscribe In.

After the closer comparison of two approaches it turns out that for [14] the issue of concurrent actions which cancel out each other's effects is more complicated then for us. They have to ensure that compound actions inherit the effects of their components [19]. Moreover, they concerned themselves with the problem how to take into account the fact that for an action A_{10} to override the effect of some global action $\{A_1, A_2\}$ itself must not be overridden by another action A_{23}. We have achieved the same result without compound actions, because we divide the dynamic component of the situational ontology on two distinct primitives.

In our previous paper [20] we also dealt with concurrent actions representation, but we proposed another approach. Instead of using the predicate $Wrong$, in order to represent actions which cancel each other effects, we used another method. Suppose that we know about a simple action that it can be executed only if some other actions $\{A_1, ..., A_n\}$ are not executed concurrently. In [20] we simply added conjunctively to the left side of the causal rule formulas $\neg Type(o, A_i)$ $(i = 1, .., n)$, to say explicitly what actions the performing operator should not contain. So, the method was very natural and it provided a monotonic solution to the *action-oriented frame-problem* [14]. The approach we present in this paper not only provides the alternative solution, but, most importantly allows also to recognize the behavior of multi-agent system even if an action had not its intended effect.

8 Conclusion

We proposed the logical theory of concurrent actions that allow to draw conclusions about the execution of actions not all of which are specified in the causal rules. We coped with the *fluent oriented frame problem* and the complexity of our solution is less than the complexity of monotonic solution. We proposed the nonmonotonic way to tackle the *action-oriented frame-problem* [14]. The problem of complete characterization of the preconditions (*qualification problem*) can be overcame by the circumscription of the predicate $Wrong$. Our approach also dealt with *ramification problem*, because implicit effects of actions can be found from constraints on the values of fluents. In this regard, the important distinctions proposed in [11] should be taken into account.

In our opinion, the concurrent actions representation must be used for reasoning about changes extended in time, because it may happen that continuous changes are overlapped in time [21]. It is important to note that correspondence between the situation calculus with action types and the explicit time-line temporal calculus established by Weber [22] facilitates future progress.

A matter of prime concern is how to compute circumscription algorithmically [6]. This point will be addressed in our future work.

Acknowledgements: We are indebted to A.Bondarenko and M.Shanahan for discussions, to R.Miller for detailed list of questions and comments, to Slava Ivanov and company Brokinvest for support by computing facilities, to Y.Gouz for help in telecommunications. We are grateful to E.Sandewall and P.Doherty for useful discussion during 11th ECAI'94 in Amsterdam.

References

1. C.Baral, M.Gelfond Representing concurrent actions in extended logic programming. Proc. of the 13th IJCAI, 1993, 866–871
2. J.M.Crawford, D.W. Etherington Formalizing Reasoning About Change: A Qualitative Reasoning Approach (Preliminary Report) In: Amer. Assoc. of Artif. Intell., 10th National Conference on Artif. Intell., 1992, 577-583.
3. B.Dunin-Keplicz, J.Treur Compositional formal specification of multi-agent systems, (this volume)
4. M.Gelfond, V.Lifschitz, A.Rabinov What are the limitations of the situation calculus ? In: Essays for Bledsoe, (Ed. by R.Boyer), Kluwer Academic, 1991, 167–177
5. G.N.Kartha Soundness and completeness theorems for three formalisations of action. Proc. of the 13th IJCAI, 1993, 724–729
6. Ph.G.Kolaitis, Ch.H.Papadimitriou Some computational aspects of circumscription. J. of the ACM, 37, 1990, 1–14
7. K.Konolidge A first-order formalisation of knowledge and action for a multi-agent planning system. In: Machine Intelligence, v. 10, (Eds. J.E.Hayes, D.Michie, Y.-H.Pao), Chichester, Ellis Horwood Ltd., 1982, 41–71
8. V.Lifschitz Toward a metatheory of action. Proc. of the 2nd Int'l Conf. on Principles of Knowledge Representation and Reasoning, (Eds. J.Allen, R.Fikes, E.Sandewall), 1991, p.376-387
9. V.Lifschitz Computing circumscription. Proc. 9th IJCAI, 1985, 121–127
10. V.Lifschitz Pointwise circumscription: preliminary report. In: Amer. Assoc. of Artif. Intell., 5th National Conference on Artif. Intell., 1986, v.1, p. 406-410
11. F.Lin, R.Reiter State Constraints Revisited. J. of Logic and Computation, 4 (1994), (to appear)

12. E.Sandewall Filter preferential entailment for the logic of action in almost continuous worlds. Proc. of 11th IJCAI, 1989, 894–899
13. E.Sandewall Features and Fluents: A Systematic Approach to the Representation of Knowledge about Dynamical Systems. Technical Report LiTH-IDA-R-94-15, 1994 (to be published by Oxford Un. Press)
14. F.Lin, Y.Shoham. Concurrent actions in the situation calculus. In: Amer. Assoc. of Artif. Intell., 10th National Conference on Artif. Intell., 1992, v.1, p. 590-595
15. J.McCarthy, P.Hayes Some philosophical problems from the standpoint of artificial intelligence. In: Machine Intelligence, v. 4, (Eds. B.Meltzer and D.Michie), Edinburgh University Press, 1969, p. 463–502
16. J.McCarthy First order theories of individual concepts and propositions. In: Machine Intelligence, v. 9, (Eds. J.E.Hayes, D.Michie, L.I.Mikulich), Chichester, Ellis Horwood Ltd., 1979, p.120–147
17. L.Morgenstern A first order theory of planning, knowledge, and action. In: Theoretial Aspects of Reasoning about Knowledge (Ed. by J.Halpern), Proc. of the conference held March 19–22, 1986, Morgan Kaufmann Publ.
18. R.Reiter The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. In: Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy, Academic Press, San Diego (CA),1991, 359–380.
19. M.Shanahan E-mail message N frigate.do.395 Date: Wednesday, 20th January 1993, 13:12:01.
20. M.E.Soutchanski, E.A.Ternovskaia Logical theory of concurrent actions. In: Soviet J. of Computer and System Sciences, 1993, vol.31.
21. E.A.Ternovskaia Interval situation calculus. Proc. of the W/Sh 'Logic and Change', held 8th August 1994, during 11th ECAI, 1994, 153–165
22. J.C.Weber On the representation of concurrent actions in the situational calculus. Proc. of the 8th Biennual Confer. of Canadian Society of Computat. Study of Intelligence, Ottava, 22-23 May, 1990, 28-32.
23. M.Wooldridge The logic of an agent-oriented testbed for DAI, (this volume)
24. M.Wooldridge, M.Fisher A first-order branching time logic of multi-agent systems. Proc. of the 10th ECAI, 1992, 234–238.

This article was processed using the LaTeX macro package with LLNCS style