A Logic Programming Framework for the Abductive Inference of Intentions in Cooperative Dialogues

Paulo Quaresma* and José Gabriel Lopes

No Institute Given

Abstract. In this paper we propose a general logic programming framework allowing the recognition of plans and intentions behind speech acts through abductive reasoning. These inferences enables each agent to have an active participation in dialogues, namely in cooperative information-seeking dialogues. In our framework the possible actions, events, states and world knowledge are represented by extended logic programs (LP with explicit negation) and the abductive inference process is modeled by a framework wich is based on the Well Founded Semantics augmented with explicit negation (WFSX) and contradiction removal semantics (CRSX) ([13]). It will be shown how this framework supports abductive reasoning with Event Calculus ([4]) and some classical examples in the domain of information-seeking dialogues will be shown ([9, 14]). Finally, some open problems will be pointed out.

1 Introduction

A natural language understanding system needs to infer the beliefs, the intentions and the plans of its interlocutors in order to intelligently participate in dialogues. One possible approach follows the classical planning scheme developed in the STRIPS ([5]) and NOAH ([16]) model. In this model each plan is defined as a sequence of actions and each action is composed by a head, preconditions, constraints, effects and sub-actions. The inference of plans is done through the use of a library of plans and actions, some heuristic rules and the user possible goals. This approach has been used by Litman and Allen ([9, 8]) in order to infer plans behind speech acts in dialogues. Another approach has been followed by [2] and uses weighted abductive reasoning taking interpretation as an abductive process ([7]).

We also take interpretation as abduction but, in contrast to the weighted abduction approach, we propose a general extended logic programming framework which supports the abduction of intentions behind speech acts delimiting the set of abducible predicates. We have used the event calculus to represent events, time and actions and a logic programming framework - Well Founded Semantics of eXtended Logic Programs (WFSX) augmented with Contradiction Removal

* Owns a scholarship from JNICT, reference n^o BD/1766/IA

Semantics (CRSX) - from the work of Pereira et al. ([13]). This framework, has a well defined semantics and extends logic programming allowing to model several kinds of non-monotonic reasoning, namely default, abductive and hypothetical reasoning.

In section 2 a description of the extended logic programming framework showing how non-monotonic reasoning is dealt with is given. In section 3 the process of abductive reasoning with event calculus is described. Section 4 describes the language that was used to represent actions (from Gelfond and Lifschitz [6]). Section 5 describes the epistemic operators used to model the different attitudes. In section 6 the speech acts necessary to handle dialogues are described. In section 7 some examples are presented. Namely, it is shown how this framework enables a system to handle a dialogue where its interlocutor has built an incorrect plan and how it handles dialogues where there are misunderstandings. Finally in section 8 some open problems and future work will be pointed out.

2 Extended Logic Programming Framework

Non-monotonic reasoning capability is a requirement of any natural language processing system. In our approach this kind of reasoning is modeled by an extended logic programming framework which models actions, events, states and the world knowledge by extended logic programs. These programs are sets of normal and integrity rules having the form:

$H \leftarrow B_1, \ldots, B_n, not\ C_1, \ldots, not\ C_m (m \geq 0, n \geq 0)$

where H, B_1, …, B_n, C_1, C_m are classical literals. A classical literal is either an atom A or its explicit negation $\neg A$. *not* stands for the negation by failure (NAF). In integrity rules H is the symbol $\perp$ (contradiction).

Programs with integrity constraints and explicit negation may be contradictory. The contradiction can be removed by adding to the original program the complements of some revisables ([13]).

The framework described allows to model different types of non-monotonic reasoning, namely default, hyphotetical, and abductive reasoning.

– Default reasoning can be modeled adding to the program rules of the form:

Normally birds fly.

which can be written as:

1. fly(X) $\leftarrow$ bird(X), not ab(X)

which states that if it is not possible to prove the abnormality of the bird X then it should fly.

– With a slight change it is possible to handle hypothetical reasoning:

Quakers might (or not) be pacifists

which can be written as:

1. pacifist(X) $\leftarrow$ quaker(X), hypqp(X)
2. hypqp(X) $\leftarrow$ not $\neg$ hypqp(X)
3. $\neg$ hypqp(X) $\leftarrow$ not hypqp(X)

which states that quakers are pacifists if it is not possible to prove (by NAF) explicitly that they are not (and vice-versa).

– Abductive reasoning can be modeled with the rules:

F might be true (or not)

which can be written as:

1. $F \leftarrow not\ \neg F$
2. $\neg F \leftarrow not\ F$

which state that if it is not possible to prove $\neg F$ then F should hold (and vice-versa). This is the way how Pereira et al. ([13]) relate explicit negation and negation by failure.

Using this approach it is possible to create an abductive program from an abductive theory (P, Ab) by adding to the program P for all literals L in the abducible list Ab two rules of the form:

1. $L \leftarrow not\ \neg L$
2. $\neg L \leftarrow not\ L$

In the domain of dialogues, we will allow the abduction of a limited set of predicates (see next section). This process is done through the use of this framework and the Event Calculus.

3 Abductive Planning with Event Calculus

In order to represent and reason about actions and events a modified version of the event calculus proposed by Eshgi ([4]) and Missiaen ([12]) was used.

In this formalism, events are action instances. They can be represented by the time at which they ocurred. Time is linear and time points are ordered by the precedence relationship $<$. Events are referents which are related to actions by the predicate $act(E, A)$. $happens(E)$ states that the event E occurs. $initiates(E, P)$ means that the event E initiates the property P, and $terminates(E, P)$ states that property P is terminated by the event E. $succeeds(E)$ means that the event E can occur, i.e., its preconditions are satisfied (this predicate is different from *happens* which means that the event did really happened). $holds\_at(P, T)$ means that property P holds at time T.

A plan consists of the set of facts defined using the predicates $<$, *happens*, and *act*. This set can represent a plan because it defines the events that may happen, the actions that are associated with those events, and the time ordering between the events.

The following logic program is proposed by Missiaen in order to describe what properties hold at a given time:

$$\begin{aligned}
holds\_at(P,T) &\leftarrow happens(E), initiates(E,P), && (1)\\
&\qquad succeeds(E), E < T, persists(E,P,T). &&\\
persists(E,P,T) &\leftarrow notclipped(E,P,T). && (2)\\
clipped(E,P,T) &\leftarrow happens(C), terminates(C,P), && (3)\\
&\qquad succeeds(C), notout(C,E,T). &&\\
out(C,E,T) &\leftarrow (T = C; T < C; C < E). && (4)
\end{aligned}$$

which states that the property P holds at time T if there was an event that happened before T, if that event initiates P and if P persists until T. A property P persists until T if it is not possible to prove (by NAF) the existence of an event that terminates the property P before T. However it is still an open problem to prove that this approach is correct, complete and/or sound wrt. the well founded semantics framework described in the previous section.

In this approach planning can be seen as an abductive process in which the set of abducible predicates is composed by:

Ab = {happens/1, act/2, </2}

Note that the predicate *succeeds/1* is not abducible. This feature implies that only the actions that are possible in a given time can be abduced.

4 Representing actions

In order to represent actions we have used a language which extends the proposal of Gelfond and Lifschitz ([6]). In this language actions can be represented by the following propositions:

1. A causes F if $P_1, \ldots, P_n$
2. F if $P_1, \ldots, P_n$
3. F after $A_1, \ldots, A_n$

The first rule means that action A causes the effect F if the pre-conditions P_1, ..., P_n hold; the second rule means that the fluent F holds if $P_1, \ldots, P_n$ hold; the last rule means that the fluent F holds after the sequence of actions $A_1, \ldots, A_n$.

We have defined a translation mechanism from this language into the event calculus language introduced earlier. For each action described by the first statement the following two rules are obtained:

$$succeeds(E) \leftarrow act(E, A), holds\_at(P_1, E), \ldots, holds\_at(P_n, E).$$
$$initiates(E, F) \leftarrow act(E, A), holds\_at(P_1, E), \ldots, holds\_at(P_n, E).$$

This scheme states that an event E associated with an action A at a time T succeeds if the pre-conditions hold at that time and as a consequence of this event the property F will hold in the future. As *act/2* is an abducible predicate, if it is necessary to prove *succeeds(E)* then *act(E,A)* might be abduced (if the pre-conditions hold at time E).

The second rule is translated into the following logical rule:

$$holds\_at(F, E) \leftarrow holds\_at(P_1, E), \ldots, holds\_at(P_n, E).$$

This rule states that the property F holds if the conditions $P_1, \ldots, P_n$ hold.

The last rule is translated into the following logical rule:

$$\begin{aligned} initiates(E_n, F) \leftarrow\ & happens(E_1), act(E_1, A_1), \\ & happens(E_2), act(E_2, A_2), E_1 < E_2, \\ & \ldots, \\ & happens(E_n), act(E_n, A_n), E_{n-1} < E_n. \end{aligned}$$

which states that the property F is initiated by the sequence of actions A_1, ..., A_n.

In order to apply the described framework to a specific domain we have to define the domain specific rules using these meta rules and then translate them into logical rules.

On the other hand, it is important to point out that we have not proved that the proposed transformations into the abductive event calculus are correct and complete.

5 Epistemic Operators

The epistemic operators needed for describing the agents' mental state are the following ([2, 3]):

- int(a, α): agent a intends to do α
- bel(a, p): agent a believes that p is true
- ach(a, p):agent a believes p will become true as a consequence of its actions
- exp(a, p) $\leftarrow$ bel(a, p) or ach(a, p): agent a expects that p is true
- know(a, p): agent a knows property p
- knowif(a, p): agent a knows if property p is true
- knowref(a, t, p): agent a knows the term t of property p

More complex actions can be constructed from the operators *to* e *by*:

- to(α, p): the plan of performing α in order to make p true
- by(α, β, p): the plan of making β by doing α, while some fluent p holds

It's also necessary to define some rules that connect these epistemic operators:

int(A, α) *if* int(A, to(α, P)).
ach(A, P) *if* int(A, to(α, P)).

This rule means that if, at a given time, an agent A intends to do α in order to make P true, then he intends to do α and he wants P to become true. There is also the corresponding rule for the relation *by*:

int(A, α) *if* int(A, by(α, β, P))).
int(A, β) *if* int(A, by(α, β, P)))
exp(A, P) *if* int(A, by(α, β, P)))

This rule means that if, at a given time, an agent A intends to do α in order to have β done while P is true then he intends to do α and β and he expects P to be true.

We have also defined a different rule that connect the epistemic operators *bel* and *int* with the other actions:

bel(s, ach(a, E)) *if* bel(s, int(a, Act)) & (Act *causes* E)

This rule means that if s believes that a intends to do an *Act* then s believes a wants to achieve the state where the effects of that *Act* hold.

6 Speech Acts

Using the language described in the previous section we have defined expressions for some speech acts (from [1, 3]). We will elaborate on the expressions for the speech acts needed in the examples of section 7.

1. inform(speaker, hearer, proposition)
2. informref(speaker, hearer, term, proposition)
3. informif(speaker, hearer, proposition)
4. request(speaker, hearer, action)

The first action is described by:

inform(S, H, P) *causes*
know(H, P) & know(H, know(S, P)) *if*
know(S, P) & int(S, inform(S, H, P)).

If a speaker S knows a specific proposition and intends to inform the hearer H about it and actually informs the hearer about it then the hearer will learn the proposition and he will know that the speaker knows it.

The second speech act is described by:

informref(S, H, T, P) *causes* knowref(H, T, P) *if*
knowref(S, T, P) & int(S, informref(S, H, T, P)).

If a speaker S knows about some aspect T of a proposition P and intends to inform the hearer H about it and actually acts in order to inform the hearer about it then the hearer will learn that fact.

The third speech act is defined by:

informif(S, H, P) *causes* knowif(H, P) *if*
knowif(S, P) & int(S, informif(S, H, P)).

This expression means that if the speaker S knows that a given property P is true and s/he informs the hearer H about that status then the hearer will learn about that fact.

The last speech act can be defined by:

request(S, H, A) *causes* int(H, A) & know(H, int(S, A)) *if*
int(S, A) & bel(S, cando(H, A)) & ¬ bel(S, cando(S, A)).

If a speaker S intends an action A to be done, and he believes he can't do action A, then he may request the hearer to do the action. This action causes the hearer H to want to perform the action (we are assuming a cooperative dialogue) and to know that the speaker wants the action to be done.

7 Examples

7.1 Incorrect User Plan

In the following example (adapted from [14]) the user plan is detected to be incorrect if the user wants to achieve the stated goal.

- Q: I want to talk to Kathy. Do you know her phone number at the hospital?
- A: She has already been discharged. Her home number is 555-8321.

It is necessary to define a basic rule that elaborates on common sense reasoning:

bel(s, by(call(s, p), talk(s, h), at(h, p))) *if*
int(s, talk(s, h)) & bel(s, at(h, p)) & knowref(s, n, phone_number(n, p)).

This rule means that if a speaker intends to talk to a hearer, he believes the hearer is at place p, and he knows the phone number of p, then he believes that by calling to p he will talk to the hearer.

Using this rule and the Event Calculus as described in the previous sections it is possible to reason about the example presented. In this example the user question might create the following facts:

$happens(e1).$
$act(e1, inform(user, sys, int(user, talk(user, kathy)))).$
$happens(e0), e1 < e0$
$act(e0, request(user, sys, informref(sys, user, N, phone\_number(hospital, N)))).$

With action $e1$, and the $inform$ rule we'll have:

$$holds\_at(bel(sys, int(user, talk(user, kathy))), e0)$$

Using the rule for the request action $e0$ it is possible to abduce that the user wants to be informed about the hospital number.

$$holds\_at(int(user, informref(sys, user, N, phone\_number(hospital, N))), e0).$$

Using the rule for *informref* the system may infer that the user wants to be informed about the hospital phone number in order to know that number:

$$holds\_at(int(user, knowref(user, N, phone\_number(hospital, N))), e0).$$

With the first rule described in this section and these facts it is possible to infer:

$$holds\_at(bel(user, by(call(user, hospital), talk(user, kathy), at(hospital, kathy))), e0)$$

Meaning that the user believes that by calling to the hospital he will talk to Kathy. In order to do this inference it was necessary to abduce an action that initiated the fluent *bel(user, at(hospital, kathy))*.

On the other hand, suppose that the system knows that Kathy is at home:

$$initiates(e, bel(system, at(home, kathy))), e < e0.$$

It is possible to generate by the system an answer similar to the one given in the example showing the incorrectness of the user plan (from the system's point of view) and pointing out a possible solution.

7.2 Misunderstanding Dialogue

In this example (adapted from [17, 11]) there is a misunderstanding of the real intentions of the first speaker. In fact, after the second utterance, the second agent needs to revise the beliefs about the other agent intentions:

1. A: Do you know who's going to that meeting?
2. B: Yes.
3. A: Who?
4. B: Oh. Probably Mrs. McOwen and probably Mrs. Cadry and some of the teachers.

Using the speech acts and the epistemic operators previously defined it is possible to handle this kind of dialogue. The first utterance creates the following facts:

$$\begin{array}{l} happens(e1), \\ act(e1, request(a, b, informif(b, a, knowref(b, T, going(T, meeting))))). \end{array}$$

From these facts, using the rule for *request*, and abducing the pre-condition of that act it is possible to infer that:

$$holds\_at(int(a, informif(b, a, knowref(b, T, going(T, meeting)))), e1).$$

This property means that A wants to be informed if B knows who is going to the meeting.

Using the rule for *informif*, and inferring the effects of that act it is possible to deduce:

$$holds\_at(int(a, knowif(a, knowref(b, T, going(T, meeting)))), e1).$$

This inference means that the agent A has the intention to know if B really knows who is going to the meeting.

Consider that there was a previous event $e0$ stating that B knows that property:

$$\begin{array}{l} e0 < e1, \\ initiates(knowref(b, people, going(people, meeting)), e0). \end{array}$$

Then it is possible to B to generate the second utterance (using the rule for *informif*).

With the third utterance, the agent A creates a new set of facts:

$$happens(e2), e1 < e2,$$
$$act(e2, request(a, b, informref(b, a, T, going(T, meeting)))).$$

Using the same inference process it is possible to infer the following facts (using the rule for *informref*):

$$holds\_at(int(a, informref(b, a, T, going(T, meeting)))), e2).$$

And inferring the effects of the speech act *informref*:

$$holds\_at(int(a, knowref(a, T, going(T, meeting)))), e1).$$

This inference process allows the agent B to generate the fourth utterance.

Note that our framework handles this dialogue but is also able to support a more intelligent dialogur where the second utterence is replaced by the fourth, i. e. agent B understands that the first utterence is an indirect question.

8 Conclusions and Future Work

We have presented in this paper a logic programming framework that supports the abductive inference of actions and events through the use of extended logic programs, the event calculus, speech acts, and epistemic operators. The integration of these features allows us to model and to handle a large class of co-operative information-seeking dialogues in a very general approach. In fact, the results obtained show that this framework is able to handle some of the classical problems that arise in dialogues (non-specified goals, clarification sub-dialogues, error situations, wrong user plans, misunderstandings).

As it was pointed out this work has some connections with the work of Appelt and Pollack ([2]) using weighted abduction for plan ascription. In our work we have the abductive reasoning process integrated in a general non-monotonic logic programming framework using Event Calculus and based on epistemic operators and a speech acts theory.

As future work we will try to show how this framework can also model other kinds of dialogues, namely missleading and non-cooperative ones. In order to handle this larger class of problems, the framework should be incorporated in a more general natural language processing system such as the one described by Lopes and Quaresma ([10, 15]) where a multi-headed architecture coordinates several independent modules with the shared objective of supporting a robust natural language interaction.

References

1. J. F. Allen and C. R. Perrault. Analyzing intention in utterances. *Artificial Intelligence*, (15):143–178, 1980.
2. Douglas E. Appelt and Martha E. Pollack. Weighted abduction for plan ascription. *User Modeling and User-Adapted Interaction*, 2(1), 1992.
3. P. Cohen and H. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(3), 1990.
4. Kave Eshghi. Abductive planning with event calculus. In *Proceedings of the International Conference on Logic Programming*, 1988.
5. R. E. Fikes and Nils J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intellligence*, (2):189–208, 1971.
6. M. Gelfond and V. Lifshitz. Representing actions in extended logic programs. In *Proceedings of the International Symposium on Logic Programming*, 1992.
7. J. Hobbs, M. Stickel, P. Martin, and D. Edwards. Interpretation as abduction. In *Proceedings of the 26th Annual Meeting of ACL*, 1988.
8. D. Litman and J. Allen. A plan recognition model for subdialogues in conversations. *Cognitive Science*, (11):163–200, 1987.
9. Diane J. Litman. *Plan Recognition and Discourse Analysis: An Integrated Approach for Understanding Dialogues.* PhD thesis, Dep. of Computer Science, University of Rochester, 1985.
10. J. G. Lopes. Architecture for intentional participation of natural language interfaces in conversations. In C. Brown and G. Koch, editors, *Natural Language Understanding and Logic Programming III.* North Holland, 1991.
11. Susan McRoy and Graeme Hirst. Abductive explanation of dialogue misunderstandings. In *EACL'93*, 1993.
12. Lode Missiaen. *Localized Abductive Planning with the Event Calculus.* PhD thesis, Univ. Leuven, 1991.
13. Luís Moniz Pereira, José Júlio Alferes, and Joaquim Nunes Aparício. Contradiction removal semantics with explicit negation. In M. Masuch and L. Pólos, editors, *Knowledge Representation and Reasoning Under Uncertainty, Volume 808 of LNAI*, pages 91–106. Springer-Verlag, 1994.
14. Martha E. Pollack. *Inferring Domain Plans in Question-Answering.* PhD thesis, Dep. of Computer and Information Science, University of Pennsylvania, 1986.
15. P. Quaresma and J. G. Lopes. A two-headed architecture for intelligent multimedia man-machine interaction. In *B. de Boulay and V. Sgurev (eds). Artificial Intelligence V - methodology, systems, applications.* North Holland, 1992.
16. Earl D. Sacerdoti. *A Structure for Plans and Behavior.* American Elsevier, New York, 1977.
17. Emanuel Schegloff. Repair after next turn: The last structurally provided defense of intersubjectivity in conversation. *American Journal of Sociology*, 5(97):1295–1345, 1992.