Using dynamic logic programming to model cooperative dialogues

Paulo Quaresma and Irene Pimenta Rodrigues
pq@dmat.uevora.pt ipr@dmat.uevora.pt

Departamento de Matemática
Universidade de Évora
7000 Évora
Portugal

CENTRIA/AI Center
Universidade Nova de Lisboa
2825 Monte da Caparica
Portugal

Abstract

In this paper we present a system that is able to perform cooperative information-seeking dialogues for the interrogation of a text knowledge base.

In our system each event (utterance) is represented by logic programming facts which are used to dynamically update the previous user model. Using this approach it is possible to represent new events as update logic programs and to obtain the new "mental" states. Moreover it is possible to reason about past events and to represent non-monotonic behavior rules.

The system has to represent four levels of knowledge using dynamic logic programming. The knowledge levels are: Interaction, Domain, Information Retrieval and Text.

The interaction level is responsible for the dialogue management. This includes the ability of the system to infer user intentions and attitudes and the ability to represent the dialogue sentences in a dialogue structure in order to obtain the semantic representation of the dialogue.

The domain level includes knowledge about the text domain and it has rules encoding that knowledge. For instance, in the law field it is necessary to represent under which conditions a pension for relevant services may be given to someone; those pensions are usually attributed to militaries or to civilians such as firemen, doctors, and nurses.

The Information Retrieval Level includes knowledge about what we should expect to find in texts about a subject, for instance that in texts about pensions for relevant services, the pension may be attributed or refused.

The Text Level has knowledge about the words and sequence of words that are in each text of the knowledge base.

A detailed example of a cooperative dialogue in the law field will be shown.

Introduction

In this paper we use dynamic logic programming to integrate four sources of knowledge in order to implement a cooperative dialogue system for a web based information retrieval text database with juridical texts.

Our dialogue system achieves cooperation with the user by using knowledge: about the database texts (Information Retrieval level); about the domain, in our case some juridical rules (Domain level); about the words and expressions that occur in each text of the text base (Text level).

There is another knowledge level that our system has to represent which is the Interaction. This level includes the representation of the user and system utterances and the intentions associated to each utterance, as well as knowledge that enables the system to infer the user intentions.

These four levels of knowledge are represented using dynamic logic programming.

Dynamic logic programming (Alferes et al. 1998) defines how a logic program can be updated by other logic programs. In our approach, each event is represented by a logic program (composed only by facts) which is used to update the previous program and to obtain a new one. In fact, events are represented by an update situation and there is no need to explicitly represent time points. Inertia rules are also guaranteed by the dynamic logic programming semantics.

The information retrieval system is based on SINO, a text search engine from the AustLII Institute (Greenleaf, Mowbray and King 1997).

The Interaction Level represents the speech acts using a dialogue structure that groups the utterances of the dialogue in a way that at any point of the dialogue it is possible to obtain its semantic content. This is obtained with a precise definition of the dialogue segments that include inheritance rules for the dialogue attributes.

The following dialogue is an example of a user session in our system (this example will be fully explained in the example section).

Dialogue 1:

User - Q1: Documents about pensions for relevant services?
System - Q2: Pensions that were given or refused?
User - Q3: Both.
System - Q4: Pensions for militaries or civilians?
User - Q5: Civilians
System - Q6: There are 129 documents, listed in the right frame.
User - Q7: where there was a drowning?
System - Q8: Since there are 45 documents of the 129 above that match the concept, I've listed them in the usual frame.
User - Q9: No, I mean to look for all documents that match the concept.
System - Q10: There are 347 documents listed in the right frame.
User - Q7': Documents about drugs?
System - Q8': I assume that you want a new query about drugs since there are no documents with the concept drugs and pensions for relevant services for civilians. Do you want to see documents about the use of drugs while driving?
User: - Q9' yes.
...

Given a user utterance, such as Q1, the system is able to cooperatively interact with user in order to refine its query.

The system reply to Q1 will be Q2, this reply is achieved by recognizing that this query can be refined since the texts that mention pensions can divided into two disjoint sets, one where pensions were given and another one were pension were refused. This kind of knowledge is encoded in what we have called the Information Retrieval level. This sort of knowledge can be obtained with a preprocessing of the texts in the text base and we have rules encoding it.

After the user answer (that could be: given, rejected or both), by using knowledge of the Domain level the system will generate question Q4. This is achieved by knowing that pensions by relevant service have different conditions when there is a military or a civilian. This is juridical knowledge independent of the texts present in the text base.

Our system is also able to decide if the user intends to continue its previous query (its utterance is to be interpreted in the context of the previous dialogue) or to open a new query (a new interrogation context).

If, after Q1 the user asks Q7, the system will be able to decide that the user intends to look for text where there are a pension and a drowning. But if the user utters Q11 instead of Q7 the system will conclude that the user intends to open a new interrogation context.

User - Q11: Documents where there are drugs?

This is achieved by using the Textual level that encodes knowledge about the texts words and expressions (concepts). Using our retrieval information system SINO, it is possible to see that are some texts where the concepts *pension and drowning* appears but no texts where the concepts *pension and drugs* appears. This is what the user expects the system behave in most cases. When this is not the case the user may clarify its query in order oblige the system to behave differently. For instance after Q7 the system will reply Q8 and the user may reply Q7'.

Q9 will be understood by the system as a user clarification and it will forget the semantic content of sentences Q1-Q8 by opening a new context with Q9. In order to interpret the sentence Q9 in particular to solve the nominal anaphora *the concept*, the dialogue structure of sentences Q1-Q8 will be used.

Dynamic logic Programming framework

Dynamic logic programming (Alferes et al. 1998) defines how a logic program can be updated by other logic programs. In fact it defines a semantic for a sequence of logic program updates *P1,…,Pn*. In the update process, each state (*P1,…,Pn*) may represent a different situation, or even a different time point. This feature allows us to model dialogue events by logic programs (composed only by facts) and to use them to update the previous programs. Inertia rules are also guaranteed by the dynamic logic programming semantics. Alferes et al. propose in their paper a declarative and a procedural semantics for dynamic logic programming.

In order to describe rules in DLP it is possible to use the keywords **before** and **now** to represent the previous and the actual state.

For instance, the speech act inform may be described by "bel(H,P)<-inform(S,H,P)/before", meaning that after an inform speech act, the hearer starts to believe in the informed proposition (we have assumed cooperative and sincere users).

Suppose we have:

P0 = {int(s,X) <- bel(s,int(u,X)), bel(H,P)<-inform(S,H,P)/before}, meaning that the system intends to do an action X if he believes the user intends to do that action;

P1 = {inform(u,s,int(u,search_documents))}

Then, in state P2 we'll have:

{int(s,search_documents}

Moreover, it is possible to use explicit negation and abduction over the DLP framework allowing the representation on non-monotonic characteristics.

Knowledge Representation Levels

In this section we describe the four knowledge representation levels and we describe how they are integrated in the dynamic logic programming framework.

Our four representation levels are:

1. Textual Level
2. Domain Level
3. Information Retrieval Level
4. Interaction (dialogue level)

The Text Level has knowledge about the words and sequence of words that are in each text of the knowledge base.

The Domain level includes knowledge about the text domain such as juridical knowledge (for instance, under which conditions a pension for relevant services may be given to someone). It has rules that encoded the domain knowledge, such as: normally those pensions are attributed to militaries or to civilians (firemen, doctors, nurses, etc.) and the conditions to be fulfilled by them are different.

The Information Retrieval Level includes knowledge about what we should expect to find in texts about a subject, for instance that in texts about pensions for relevant services, the pension may be attributed or refused.

The Interaction level is responsible for the dialogue management. This includes the ability of the system to infer user intentions and attitudes and to build the representation of the dialogue sentences in a dialogue structure in order to obtain the semantic representation of the dialogue whenever it is necessary.

Text level

As it was already pointed out the information retrieval system is based on SINO, a text search engine from the AustLII Institute (Greenleaf, Mowbray and King 1997). SINO is a word based text search engine which allows boolean and free text queries.

We have changed SINO in order to be adapted to the Portuguese Language. Namely, the new system uses the Portuguese lexicon (more than 900,000 words) in order to handle morphological errors and to obtain the base queried word.

Domain Level

This knowledge level is built using the Laws describing the requisites for some juridical condition. For instance the law describing the requisites to obtain a pension for relevant services can be encoded by the following rules:

pension(X) <- military(X), action(X,A), behind_duty(A).
pension(X) <- civilian(X), action(X,A), save_life(Y,A), life_at_risk(X,A), not X=Y.

These rules state that:

1. A military may have a pension for relevant services if he has been the agent of an action, and that action was behind is duty.
2. A civilian may have a pension for relevant services if he has been the agent of an action that saves someone life putting his live at risk.

Information Retrieval Level

This level of knowledge is built with rules that can be obtained by processing the text documents looking for keywords that give rise to disjoint sets of documents. By now we obtain these rules using a thesaurus with keywords for text juridical classification.

Example of rules:

pension(X) <- pension_attributed(X).
pension(X) <- pension_rejected(X).

false <- pension_attributed(X), pension_rejected(X)

These rules state that a document with the concept pension either mentions the concept attributed or rejected.

In order to allow the system to obtain the possible explanations of the user queries, we define attributes as abducible predicates. Using this approach it's possible to obtain the set of non-contradictory logic models that explain the user query.

Abducted = {pension_attributed, pension_rejected, ...}

Interaction Level

This knowledge level will represent rules for the interaction at the dialogue level. It includes: 1) the rules for inferring the user intentions necessary to generate the system question and answers; 2) the rules necessary to build the discourse structure (dialogue) in order to obtain the semantic representation of the user utterances and the context for solving discourse phenomena such as anaphora resolution.

These two sets of rules will be detailed in the next two sections.

Cooperation with the user is achieved due to the existence of the representation and the inference of user intentions. The system tries to infer the user intentions in order help him to find out the set of documents that the user is looking for.

The system helps the user by informing him about the domain knowledge (juridical) and particularities of the texts in the knowledge base. This way the user is guided by the system in the task of refining his queries.

The dialogue representation structure supplies the context for the user and system utterances. This representation structure takes into account that an utterance may: specify the information contained in a set of previous utterances; clarify the interpretation of a set of previous utterances; open a new context, the user does not intend to continue refining its query and desires to start a new one.

Inference of user Intentions

In order to be collaborative our system needs to model user attitudes (intentions and beliefs). This task is achieved through the use of logic programming framework rules and the dynamic LP semantics.

The system mental state is represented by an extended logic program that can be decomposed in several modules (see QL95 for a more complete description of these modules):

- Description of the effects and the pre-conditions of the speech acts in terms of beliefs and intentions;
- Definition of behavior rules that define how the attitudes are related and how they are transferred between the users and the system (cooperatively).

For instance, the rule which describes the effect of an inform and a request speech act from the point of view of the receptor (assuming cooperative agents) is:

bel(A,bel(B,P)) <- inform(B,A,P)/before.

bel(A,int(B,Action)) <- request(B,A,Action)/before.

In order to represent collaborative behavior it is necessary to model how information is transferred from the different agents:
bel(A,P) <- bel(A,bel(B,P))/now, (not bel(A,P))/before.
int(A,Action) <- bel(A,int(B,Action))/now, (not neg int(A,Action))/before.

These two rules allow beliefs and intentions to be transferred between agents if they are not inconsistent with the previous mental state (neg stands for the explicit negation and not stands for the negation by omission).

After each event (for instance a user question) the agents' model (logic program) needs to be updated with the description of the event that occurred. The dialogue system recognizes the speech act and it constructs the associated speech act (request or inform). The speech act will be used to update the logic program in order to obtain a new model. Using this new model it is possible to obtain the intentions of the system.

Interrogation Context

The Dialogue structure is made of segments that group sets of sentences (user and system sentences). The dialogue structure reflects the user intentions, it is built taking into account the user and system intentions. The dialogue segments have precise inheritance rules defining how segments heritage their attributes from the attributes of their sentences.

The dialogue structure is built by recognizing the user intentions and using them in order to enable the system to intervene in the dialogue using pertinent discourse phenomena such as anaphoric references.

In order to define our dialogue structure we first present the dialogue segments and their attribute inheritance rules, and finally we present the rules that enables the system to build the dialogue structure and use it to solve discourse phenomena such as anaphora in the user and system utterances.

Dialogue Structure

In this paper we shall present the following 5 segments that enable us to built the dialogue structure of our example dialogue 1 presented in the next section:

- Empty , []- an empty dialogue structure. It is what we have initially in a dialogue.

- Basic - has 2 arguments:
  Speaker; Sentence Semantic Representation

- New - has 2 arguments:
  Dialogue Structure; Dialogue Structure

  This Dialogue Structure inherits their attributes from their second argument
  ex: New([],basic(User,Q1))

- Specify - has 2 arguments
  Dialogue Structure; Dialogue Structure

  This Dialogue structure inherits their attributes from both dialogues structure
  ex: Specify(Basic(User,[],Q1),Basic(System, Q2)

- Clarify - has 2 arguments
  Dialogue Structure; Dialogue Structure

  This Dialogue structure inherits their attributes from the second dialogue structure
  ex: Clarify(specify(basic(User,Q7), basic(System,Q8)), basic(User, Q9))

By now we may consider that Dialogue Attributes are the semantic representation of their sentences and the discourse entities introduced by the sentences.

Rules to build the discourse structure

Given sentence(S1,Speaker) where S1 is the first sentence semantic representation, the update of the new sentence dialogue is:

sentence(basic(Speaker,S1)).

This fact gives rise to the update of the new DS according to the above rules

```
ds(specify(Old_ds,Ds3))/now <-   ds(Old_ds)/before,
                                 sentence(Ds3)/now,
                                 possible(specify(Old_ds,Ds3).

ds(new(Old_ds, Ds))/now <-       ds(Old_ds)/before,
                                 sentence(Ds)/now.

ds(Ds)/now <-    ds(Old_ds)/before,
                 sentence(Ds3)/now,
                 last(Old_ds,specify(Ds1,Ds2)),
                 clarify(specify(Ds1,Ds2),Ds3),
                 substitute(Old_ds,specify(Ds1,Ds2),clarify(specify(Ds1,Ds2), Ds3),Ds),
```

where:

```
possible(specify(Ds,basic(User,S1)))<- search_sino(semantic(specify(Ds,basic(User,S1))),Y),not Y=[].
possible(specify(Ds,basic(System,S1))).
```

These rules encode that it is possible for an user utterance to specify a discourse structure if the resulting structure gives rise to a SINO query that match one or more documents. A system utterance always specifies the previous dialogue.

```
clarify(Ds1,Ds2)<- incompatible(Ds1,Ds2).
```

This rule encodes that if two dialogue structures are incompatible then the second one clarifies the first one.

With these rules the system always intends to specify the user query, either by answering it or by asking the user to give more information.

The dialogue structure has preference rules over the segments to build when it is possible to built more than one. The preference are: clarify segment, specify segment and new.

Detailed Example of a Collaborative Dialogue

Dialogue 1:

User - Q1: Documents about pensions for relevant services?
System - Q2: Pensions that were given or refused?
User - Q3: Both.
System - Q4: Pensions for militaries or civilians?
User - Q5: Civilians
System - Q6: There are 129 documents, listed in the right frame.

User - Q7: where there was a drowning?
System - Q8: Since there are 45 documents of the 129 above that match the concept, I've listed them in the usual frame.
User - Q9: No, I mean to look for all documents that match the concept.
System - Q10: There are 347 documents listed in the right frame.

User - Q7': Documents about drugs?

System - Q8': I assume that you want a new query about drugs since there are no documents with the concept drugs and pensions for relevant services for civilians. Do you want to see documents about the use of drugs while driving?

User: - Q9' yes.

.....

System reasoning steps

Q1 causes an update of:

sentence(basic(user,[x,y:documents(x),concept(y), y=pension,about(x,y)]))

which gives rise to the update of

DS2=ds(new([],basic(user,[x,y:documents(x), concept(y), y=pension,about(x,y)]))

and

request(u,s,inform(s,u,[document(X),concept(Y),Y=pension, about(X,Y)])).

Which gives rise to the following update

int(s,inform(s,u, [document(X), concept(Y), Y=pensao, sobre(X,Y)])).

Due to the speech act rule for requests in the Interaction knowledge level: int(S,A) <- request(S,H,A)/before

In order to execute the inform action we must obtain values for the free variables in the semantic representation.

A non collaborative version will simple launch the query: "sino> search Y" at the Text knowledge level. Our collaborative version will use the Information Retrieval and the Domain level to predict the user goals.

The steps of the reasoning process are:

- Update at IR level of the concepts to be searched: Y (pension)
- Obtain models at IR level: {pension, pension_attributed, military},{pension, pension_refused, military},{pension, pension_attributed, civilian},{pension, pension_refused, civilian},{pension, military},{pension, pension_attributed}, {pension}
- Choose the most suitable model:
- Obtain the predicate list: {pension_attributed, military}
- Ask the user to supply the value of those predicates:
  request(s,u,inform(u,s,[document(Y),concept(Z),Z=military, about(Y,Z)])) e
  request(s,u,inform(u,s,[document(Y),concept(Z),Z=pension_attributed, about(Y,Z)])).

This will cause the system to generate questions Q2 and Q4.

Q2 and Q4 will be incorporated in the discourse structure in a specify segment as well the user answers Q3, Q5 and Q6.

When the user poses question Q7.

User - Q7: where there was a drowning?

The system will update:

sentence(basic(user,[x,y:documents(x),y=drown,about(x,y)]))

Which gives rise to the update of:

DS2= ds(specify(specify(basic(user,[x,y:documents(x),y=pension,about(x,y)], ...(sentences Q2 to Q6)),basic(user,[x,y:documents(x),y=drawn,about(x,y)])).

Since possible(DS2) is a logic consequence because there are some documents in our text base that are about pensions given to someone that saves another from drowning in the sea.

Which according to our segments inheritance rule gives rise to the following semantic representation:
[x,y,z: documents(x),concept(y) y=pension,about(x,y), ... semantic of Q2-Q6 ..., concept(z),z=drown,about(x,z)]

having as consequence the update of the following speech act:

request(u,s,inform(s,u,[document(X),concept(Y),Y=pension, about(X,Y), ...Q2-Q6...., concept(Z), Z=drown, about(X,Z)])).

Since in both knowledge levels this query will have only a model the system will generate the answer Q8:

System - Q8: Since there are 45 documents of the 129 above that match the concept, I've listed them in the usual frame.

But user may be want to start a new context with Q7, so it will utter Q9:

User - Q9: No, I mean to look for all documents that match the concept.

The system will recognize this utterance as a clarification of discourse segment DS2. And it will start a new context by creating the dialogue structure DS3:

DS3= new(DS2, [v,w: documents(v),concept(w) w=drawn,about(v,w)])

If instead of Q7 the user had uttered Q7':

User – Q7': Documents about drugs?

The discourse structure that will be obtained will be a

DS4= new(DS2, [v,w: documents(v),concept(w) w=drug,about(v,w)])

Since the evaluation of sino_search> drug and pension will give an empty set of documents.

And the system will answer Q8' that includes the prediction of the user goals with the query Q7'.

Conclusions

In this paper we have proposed a system which is able to cooperatively participate in dialogues, namely in information-seeking dialogues.

The system uses dynamic logic programming to represent and to reason about events. Four levels of knowledge are described using DLP extended with explicit negation: Interaction, Domain, Information Retrieval and Text.

The interaction level is responsible for the dialogue management; the Domain level includes knowledge about the text domain; the Information Retrieval Level includes knowledge about what we should expect to

find in texts about a specific subject; and the Text Level has knowledge about the words and sequence of words in each text of the knowledge base.

Cooperation is achieved through the inference of user attitudes using the knowledge representation.

References

[AKPT91] James Allen, Henry Kautz, Richard Pelavin, and Josh Tenenberg. Reasoning about Plans. Morgan Kaufman Publishers, Inc., 1991.

[AP96] José J. Alferes and Luís Moniz Pereira. Reasoning with Logic Programming, volume 1111 of Lecture Notes in Artificial Intelligence.Springer, 1996.

[ALPPP98] José J. Alferes, João Leite, Luís Moniz Pereira, H. Przymusinska and T. Przymuzinski. Dynamic Logic Programming, Proceedings of KR'98- Knowledge Representaion, 1998.

[Bra90] Michael Bratman. What is Intention?, in Intentions in Communication. MIT, 1990.

[CL90a] P. Cohen and H. Levesque. Intention is choice with commitment. Artificial Intelligence, 42(3), 1990.

[Car88] Sandra Carberry. Modelling the user's plans and goals. Computational Linguistics, 14(3):23--37, 1988.

[GS86] Barbara Grosz and Candice Sidner. Attention, intention, and the structure of discourse. Computational Linguistics, 12(3):175--204, 1986.

[HM87] S. Hanks and D. McDermott. Nonmonotonic logic and temporal projection. Artificial Intelligence, 33, 1987.

[KR93] Hans Kamp and Uwe Reyle. From Discourse to Logic: An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory. Dordrecht: D. Reidel., 1993.

[KS86] Robert Kowalski and Marek Sergot. A logic-based calculus of events. New Generation Computing, 4:67--95, 1986.

[LA87] D. Litman and J. Allen. A plan recognition model for subdialogues in conversations. Cognitive Science, (11):163--200, 1987.

[LA91] Alex Lascarides and Nicholas Asher. Discourse relations and defeasible knowledge. In Proceedings of the 29th Annual Meeting of ACL, pages 55--62, 1991.

[PR93] J.Pinto and R.Reiter. Temporal reasoning in logic programming: A case for the situation calculus. In D.S. Warren, editor, Proceedings of the 10th ICLP. MIT Press, 1993.

[PP91] F.Pereira and M.Pollack. Incremental interpretation. Artificial Intelligence, 50:40--82, 1991.

[PQ98] Luís Moniz Pereira and Paulo Quaresma. Modeling Agent Interaction in Logic Programming. In Proceedings of the 11th International Conference on Applications of Prolog. Tokyo, Japan, 1998.

[QL95] Paulo Quaresma and José Gabriel Lopes. Unified logic programming approach to the abduction of plans and intentions in information-seeking dialogues. Journal of Logic Programming, (54), 1995.

[QR98] Paulo Quaresma and Irene Pimenta Rodrigues. Keeping Context in Web Interfaces to Legal Text Databases. In Proceedings of the 2nd French-American Conference on AI&LAW, Nice, France, 1998.

[RL92] Irene Pimenta Rodrigues and José GabrielPereira Lopes. Discourse temporal structure. In Proceedings of the COLING'92, 1992.

[RL93] Irene Pimenta Rodrigues and José Gabriel Lopes. Building the text temporal structure. In Progress in Artificial Intelligence: 6th Portuguese Conference on AI. Springer-Verlag, 1993.

[RL97] Irene Pimenta Rodrigues and José Gabriel Lopes. AI5, An Interval Algebra for the temporal relations conveyed by a text. In Mathematical Linguistics II, Eds Carlos Martin-Vide, John Benjamins, 1997.

[Son91] F.Song. A Processing Model for Temporal Analysis and its Aplication to Plan Recognition. PhD thesis, University of Waterloo, Waterloo, Ontario, Canada, 1991.