Generating Sentences from Different Perspectives

Lee Fedder,

The Computer Laboratory,
University of Cambridge,
Pembroke Street,
Cambridge CB2 3QG, England.
lf@uk.ac.cam.cl
Keywords : Generation, Natural Language interfaces.

Abstract

Certain pairs or groups of sentences appear to be semantically distinct, yet specify the same underlying state of affairs, from different perspectives. This leads to questions about what that underlying state of affairs might be, and, for generation, how and why the alternative expressions might be produced. This paper looks at how such sentences may be generated in a Natural Language interface to a database system.

Introduction

The following sentences would have a different semantics if parsed, yet they seem to specify the same state of affairs at some level of representation.

1a. I can stay until 5.

1b. I must leave by 5.

For generation, we ought to be able to produce either. McDonald comments on these sentences :-
"What mutually known cognitive structure do we recognise from them that would show them to be two sides of the same coin?"
(McDonald 1988)

This paper describes a language generation system which is designed as the output component of a database interface, and is capable of producing similar synonymous sentences. The architecture relies on a two level semantic representation: one describes data in the system's application database, and plays the role of McDonald's "mutually known cognitive structure"; the other describes the semantics of sentences of Natural Language, and the primitives correspond to specific entries in the lexicon. Information to be communicated is initially expressed in the application level semantics, and is be mapped to the language level semantics as part of the generation process. Alternatives similar to 1a and 1b arise during this mapping, and represent a complexity inherent in language which did not exist in the original data:- they are a property of the description.

Application level information is described by linking it with an event or state (from now on the term "event" will cover both these), for which it provides some parameter. Thus, the origin of a flight could be described by saying that the plane "flies from" the origin. The mapping process exploits a "domain model" which has two parts. The first lays out how non-temporal information is related to domain events. The second describes the temporal characteristics these events using an ontology which is rich enough to capture the temporal semantics of English expressions. Temporal information

from the application is described by first expressing it in a way that relates it to times in the model, and by then attempting to add it to the description of the event which is currently active. The alternatives arise when more than one event can be used.

The temporal ontology is based on a recent theory of temporal semantics developed by Moens and Steedman (1988). This allows a modular representation of the semantics of temporal adverbials like "until" and "by", and also aids in the generation of tense and aspect.

This system looks at the mechanics of how the alternatives can be generated from the initial data, but we will have less to say about choosing between them. Some simple choice criteria are presented, although these do not properly address the issue of what perspective is and how it can be quantified and used. We point to proposals from McDonald (1991) which seem more promising on this front.

In more general terms, this work addresses just one of the many issues involved in mapping between Natural Language descriptions of data and the more restricted representation an application database affords.

Overview

The generation system has been designed as the output stage of an airline information system. The application database holds timetabling data such as plane origins and destinations, departure and arrival times and so on. Input to the generator is a semantic form compiled from database relations. For example :-

DEST(BA123,ROME)∧ARR-TIME(BA123,2PM)

This is an expression of the application level semantics, and states that the destination of flight BA123 is Rome, and that the arrival time is 2 p.m. One of the possible surface level semantic descriptions of this would be is :-

arrive(BA123,E)∧in(E,ROME)∧at(E,2PM)

Once the information is in this form, it can be handed to a grammatical encoder for production of the surface form. The final sentence for this example would be :-

BA123 arrived in Rome at 2 p.m.

In this example, the input data has been described as a point event occurring at a given time. As we will see, other descriptions could view it in other ways, such as a state ending at that time, or as a state beginning at that time.

The Domain Model

So, database relations may be described by finding events in a model of the domain to which they correspond. This assumes, of course, that the hearer has a similar model of the domain. Figure 1 (overleaf) shows the model for an airplane flight, giving the various events and states. It shows an agent, A, flying from an origin O, to a destination at D. The state which can be described as "A be at O" or "A not leave O" leads on to an event of "A leave O" which initiates a state described as "A not arrive at D", and so on. The causal relations between the events are included in the model, and used in the generation of tense and aspect, but their use is not described in this paper.

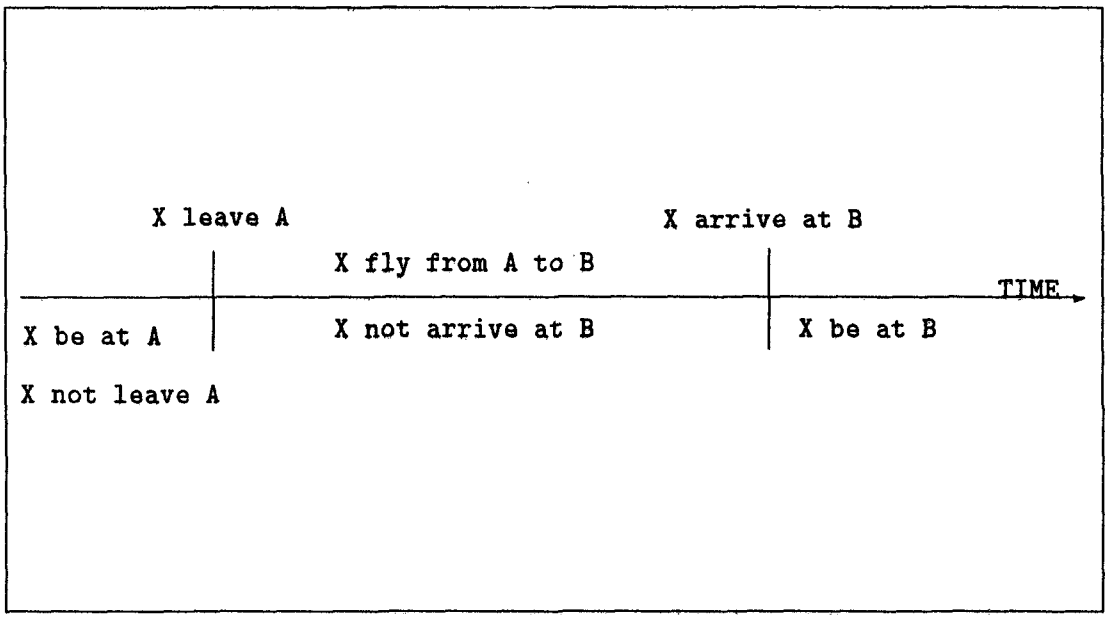
The model is represented declaratively in a Prolog style database. For each event there are two sorts of entry. The first sort record how non-temporal input-data can be translated to event based logical forms. These entries link up the data parameters with the case roles of the event. For example :-

trans(@E,@Input-sem,@Ling-sem)

The "@" is used here to denote a variable. The first argument is the event index, the second is the semantic form of the input data, and the third is the language level semantics describing the event. An example is :-

trans(e5,DEST(@A,@D),arrive(e5,@A)∧at(e5,@O))

Figure 1 - Domain Model for a Flight



The language level event here is that of "arriving", and is recorded using a Davidsonian style semantics (Davidson 1967).

The second sort of entry records the temporal characteristics of the event, using a temporal calculus developed by Moens (1987), and based on Kowalski's event logic (1986). Each event is classified according to its temporal characteristics, and entries in the calculus are made accordingly. The "arrive" event is classified as a *culmination* type of event, for which, the entry is :-

occur(cul(e5),T6)

This characterises the event e5 as a punctual event represented by the single marker "cul(e5)" which occurs at the time T6. The model is a prototypical one for the events of the domain, and actual times are unknown. Instead, temporal information is recorded using temporal indices, of which "T6" is an example. A process such as "fly" is represented by two entries, one for the start point, and one for the end.

The model includes a record of the relative times of the indices, and actual times may be included if they become known. The model also includes causal relations between events, which can be used in the generation of tense and aspect. This model has been identified by Moens as capable of capturing the semantics of English temporal expressions more fully than other formalisms, such McCarthy and Hayes (1969), or Allen (1984).

Semantics of Temporal Adverbials

With this sort of model, the semantics of adverbials may be defined in modular fashion. For instance, "until" is defined as describing the time at the end of a process type of event. So, if a process such as "Jim ran" ends at the time "2 p.m.", this would be described as "Jim ran until 2 p.m.". Similar interpretations may be defined for "for", "in", "since", "by", "later" and so on.

An Example

An example will show how several different descriptions of the same initial data may be pro-

duced using this machinery. Beginning with the input data structure shown previously in the overview, the first step is to split it into temporal and non-temporal data, which is done with a simple set of rewriting rules :-

Temp Data - ARR-TIME(BA123,2PM)

Other Data - DEST(BA123,ROME)

This is mapped onto the model by attaching the temporal data to one (or more if necessary) of the temporal indices, and by inserting the non-temporal data into a "trans" predicate :-

Temp Data - =(T6,2PM)

Other Data - trans(@E,DEST(BA123,ROME), @Ling-sem)

A duration, such as the flight time could be attached to two indices using "span(T5,T6,Flight-time)".

Instantiating the "trans" predicate in the model picks out an event that describes the data. Backtracking allows all possibilities to be produced. In the current model, this picks out four events, giving the linguistic semantics :-

fly(e3,BA123) ∧ to(e3,ROME)

not(arrive(e4,BA123) ∧ at(e4,ROME))

arrive(e5,BA123) ∧ at(e5,ROME)

be(e6,BA123) ∧ at(e6,ROME)

Of these, e3 is characterised as a culminating process (like a process, but with a definite end point) ending at T6, e4 is a state ending at T6, e5 is a culmination occurring at T6, and e6 is a state beginning at T6.

Next, we must describe the temporal data "=(T6,2PM)". A set of rules looks at the event characteristics, and the data to be expressed, to see which adverb is appropriate. For e4, the "until" adverb is chosen, and added to the semantic form to give :-

not(arrive(e4,BA123) ∧ at(e4,ROME))
∧ until(e4,2PM)

Similarly, for e5, the adverbs "at" or "by" can be used, and for e6 "by" or "since". That "since" is only used if conditions for the perfect also hold. Insufficient space prevents discussion of the details here. No adverb is available to describe the end time of a culminating process, and so no phrase can be built using e3.

The successful cases could eventually be realised as :-

2a. BA123 didn't arrive at Rome until 2 p.m.
2b. BA123 arrived at Rome at 2 p.m.
2c. BA123 arrived at Rome by 2 p.m.
2d. BA123 was at Rome by 2 p.m.

If conditions for using the perfect held, the last of these could be replaced by :-

2e. BA123 has been at Rome since 2 p.m.

Choosing Between The Alternatives

For the question answering system, several criteria are being investigated for choosing between the alternatives. The first is a simple mirroring of the phrasing of the question, the syntactic and semantic analysis of the question being retained in the discourse model. For example :-

3a. User: When will BA123 be at Rome?
3b. System: It will be at Rome by 2 p.m.

The main verb of the question is "be" with a subject of "BA123". One of the possible descriptions uses the same verb and subject (albeit pronominalised), and would be the chosen alternative. This criteria is used when the generated sentence is simply supplying new information which the user has requested.

A second criteria seems to be useful when the answer violates a presupposition detected in the query. For example, take the question :-

4a. User: Will BA123 be at Rome by noon?

This includes the presupposition that BA123 arrives at noon. If it doesn't, the best form for the answer seems to depend on the actual time of arrival.

4b. System: No, it doesn't arrive here until 2 p.m.
4c. System: No, it will be here by 11 a.m.

Construction 4b would be chosen if the pre-supposed time lay before the arrival time, and thus within the timespan covered by the state "not arrive". On the other hand, construction 4c would be chosen if the pre-supposed time lay after the actual time, placing it within the timespan covered by the state "be at Rome".

Finally, the alternatives could be useful to promote cohesion in multi sentence explanations of the following sort :-

5a. BA123 won't be here until noon. It was delayed at Paris.
5b. BA123 arrives at noon. It will taxi to Terminal 3.

The second sentence is an explanation or elaboration of the first. In the first example, the explanation refers to an event located in the time period before the arrival, and in the second, it is more closely associated with the arrival time. The description of the arrival time is chosen to reflect this.

Related work and Discussion

In a description of the process of language given by Levelt (1989), a module called "micro-planning" is included. This module comes after the content of the output has been decided on, and before grammatical encoding. Micro-planning consists of choosing the language related semantic primitives used for describing a data structure which is not linguistically based. Levelt notes that, because of the nature of language, this process will be forced to make choices of perspective. Much work on generation has assumed that the input semantic form is already in some sort of "languagese" (see, for example McDonald 1983, McKeown 1985), but the processing described in this paper would be part of the micro-planner.

There are several precedents for the use of two level semantic descriptions for generation. The first, perhaps, was HAM-ANS (Wahlster 1983),in which the generator translated from the language DEEP to the language SURF. More recently there has been the TENDUM system (Bunt 1987), using the model theoretic logical languages EL/F and EL/R, and others (Kempen 1987, De Roeck 1986). These systems translated between the levels, but did not address the issues of alternative mappings.

However, this question has been investigated by McDonald (1991). He has proposed a solution in which the data structures of the application program (a diary manager) are based on primitives such as "transition-at-4PM". These primitives are then linked to sets of lexemes such as [stay,until] and [leave,at]. One of these sets is selected and included in evolving text structure. This doesn't seem to take account of the nature of the the events described by "leave" and "stay", or the temporal semantics involved in using adverbials like "at" and "until".

McDonald does, however, address the important issue of the criteria for choosing between alternatives. The choice of perspective is intimately bound up with the reasoning of the manager, which can use knowledge about intentions and surrounding events to decide which version of the description is the most appropriate. This sort of approach seems to be necessary for the development of more comprehensive choice criteria.

Conclusion

This paper describes a generation system which is capable of generating A range of Natural Language descriptions of the output of a database enquiry program. The system uses a two level model of semantics. The possibility of alternative descriptions arises from the mapping be-

tween the two levels. Some simple criteria are used to choose the alternative which fits best into the dialogue context.

Acknowledgements

The author is supported by the Science and Engineering Research Council, and by Logica UK. I would like to thanks the many colleagues who have provided support and encouragement, especially Steve Pulman, Julia Galliers, Richard Crouch, Ann Copestake, Nick Youd, Victor Poznanski, Arnold Smith and Derek Bridge.

References

[1] Allen, J. 1984. Towards a general theory of action and time. Artificial Intelligence , 23, 123-154.

[2] Bunt, H. 1987. Utterance generation from semantic representations augmented with pragmatic information. In Natural Language Generation, by G. Kempen (Ed.), Martinus Nijhoff.

[3] Davidson, D. 1967. The logical form of action sentences. In Rescher, N. (ed.). The logic of Decision and Action. University of Pittsburgh press.

[4] De Roeck, A., and B. Lowden. 1986. Generating English paraphrases from formal relational calculus expressions. Coling proceedings.

[5] Levelt, W. 1989. Speaking. MIT press.

[6] McCarthy, J. and Hayes, P. J. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D. (eds.) Machine Intelligence, Volume 4, pp463-502. Edinburgh University Press.

[7] McDonald, David D. 1983. Natural Language Generation as a Computational Problem: an Introduction. In Computational Models of Discourse. Brady and Berwick (Ed).

[8] McDonald, D. 1988. On the place of words in the generation process. Abstract presented to the Catalina workshop on generation.

[9] McDonald, D. 1991. On the place of words in the generation process. In Natural Language Generation in Artificial Intelligence and Computational Linguistics, by C. Paris, W. Swartout, and W. Mann (Eds.) Kluwer, Dordrecht.

[10] McKeown, K. 1985. Text generation. Cambridge University Press.

[11] Moens, M. Tense, Aspect and Temporal Reference. PhD thesis, Centre for Cognitive Science, Edinburgh University.

[12] Moens, M. and Steedman M. 1988. Temporal ontology and temporal reference. Computational Linguistics, Vol. 14 No. 2.

[13] Wahlster, Jameson, Buseman and Marburger. 1983. Over-Answering yes-no questions. IJCAI, Karlsruhe.