Parsing with constructions and ontology

Wlodek Zadrozny
IBM Research, T. J. Watson Research Center
Yorktown Heights, NY 10598
wlodz@watson.ibm.com

Abstract

We address two themes of the workshop:
(A) We argue against the standard design of morphology-syntax- semantics-pragmatics.
(R) We argue for a close relationship between the knowledge representation language (used for reasoning) and the language for natural language processing.

Regarding the first issue, we present a new model of natural language based on the concept of *construction*, consisting of a set of features of form, a set of semantic and pragmatic conditions describing its application context, and a description of its meaning. A parser based on this model allowed us to build a prototype of a natural language system: a front end to an on-line calendar application.

With respect to the second issue, we show the advantages of embedding ontology into the grammar. Namely, our hierarchy of *np* constructions is based on the corresponding ontology. We also show how (A) and (R) are related.

1 Introduction

We address two themes of the workshop (which are, as we will see, in many ways linked):
(A) Defense or attacks of the standard design of morphology-syntax- semantics-pragmatics.
(R) The relationship between the knowledge representation language (used for reasoning) and the language for natural language processing.

Regarding the first issue, we present a new model of natural language based on the concept of *construction*, consisting of a set of features of form, a set of semantic and pragmatic conditions describing its application context, and a description of its meaning. We argue that the resulting model gives us a better handle on phenomena of real language than the standard approaches, such as syntax + semantics a la Montague. It is an alternative to the standard design based on the pipeline syntax-semantics-pragmatics (we have little to say about morphology at this point). Since this work has been implemented, there is also a computational argument in favor of this approach.

With respect to the second issue, we discuss in what ways ontologies are important for NLP. To see our position in perspective, notice that semantic grammars take an extreme position of embedding the ontology of the domain directly into the language, while "syntactic" grammars, on the other extreme almost completely ignore it. Our position is that the NL ontology and domain ontologies overlap, but are distinct. Hence, although the domain ontology might drive the building of a grammar, the distinction between the NL and domain ontologies should be preserved, for otherwise, the grammar will not be reusable. However, ontologies have a direct role to play in the grammar. Namely, some ontological hierarchies should map into grammatical/construction hierarchies.

The paper is organized as follows: We begin by presenting some examples that illustrate some problems of the interaction of a NLU and KR components in dialog systems. Then we present arguments against the separation of syntax, semantics and pragmatics (Section 3). In Section 4 we present our alternative (a formalism of constructions). Section 5 is about the role of ontologies in the grammar. Section 6 contains two examples of parsing with a construction grammar. In Section 7, we begin by showing how our way of integrating syntax, semantics and pragmatics fits with the postulates for a construction grammar presented in [2], and end relating (A) and (R) in a discussion of compositionality.

2 Interactions of Grammar and Representation in a Dialog System

We have built a dialog understanding system, MINCAL, for a small domain and a specific task, namely, for scheduling (as well as canceling, and moving) meetings in an on-line calendar [21]. This work has later been extended by Zadrozny and Harelick (to be reported) to other small discourse domains (fast food, banking, insurance etc.).

In MINCAL, the grammar consists of about a hundred constructions and a few hundred lexical entries, and includes both sentential constructions and discourse constructions. The system also contains an encoding of

elementary facts about time and parameters of meetings; this background knowledge is encoded in about a hundred Prolog clauses. To see some of the issues in the interaction between the grammar and the reasoning module, consider the problem of parsing of the following simple dialog of a user with the machine:

— *Schedule a meeting with Bob.*
— *At what time and date?*
— *On August 30th.*
— *At what time?*
— *8.*
— *Morning or afternoon?*
— *In the evening.*

The system we developed contains three main components with the following functions:

1. *The parser* takes an input utterance from the user and a context information from the discourse model. It outputs the set of messages corresponding to the possible meanings of the uttered phrase. The components of the parser are: the parser itself, a lexicon, a set of constructions and a set of filters.
2. *The interpretation module (semantic engine)* translates the messages it receives from the parser to a set of slots with values, corresponding to the entities recognized in the phrase, like `[action_name reschedule], [new_event_place "my manager's office"]` etc.
3. *The discourse module* gathers the computed slots, the current state of its internal database and the current context. From this it computes the new context, changes the database if necessary, informs the user about the action taken, and carries on the dialog.

MINCAL parser recognizes *Schedule a meeting with Bob* as an instance of *sent(imp)*, the imperative construction consisting of a verb and an *np*, in this case *np(event)*. [1] Then, in the context of the task and the question asked, it parses, the subsequent sentences. At each step of the dialog, the semantic engine extracts information from the results of parsing to yield (at the end of the dialog):

```
***Slots:
[  [  action_name schedule]
   [  event_name  meeting]
   [  event_time
      [  [  minute 0] [ hour 20]
         [  day 30]   [ month 8]
   [  event_partner
      [  bob]
```

Finally, another program changes the format of this data, and schedules the meeting in an on-line calendar, *xdiary*.

In regard to the two themes (A) and (R), we can make the following observations. (a) From the point of view of the operation of the system, we do not care about the structure of the sentences (provided the parser can handle them); we care only about their meanings, and these meanings depend on context. (b) We have a bit more than (R) as an issue in representation. Namely, we have to distinguish not between two but three languages (representations) and corresponding ontologies: the language/ontology for parsing, the language/ontology for representing information about the domain, i.e. calendar functions, and the language/ontology for representing information about the particular application, i.e. *xdiary*. A natural question arising is the role of all three representations in parsing.

We will talk about the interaction of form and meaning in the next section. Now we want to make a few observations about context. In the first sentence of the above dialog, the context is used to prevent another reading in which *with Bob* modifies *schedule*, as in *Dance a tango with Bob!*. That is, we use a contextual rule saying that for the calendar application people do not modify actions, nor places. By setting up a set of domain- and application-specific filters, we can remove ambiguities of PP attachment in most cases, e.g. in

Set up a lunch in the cafeteria with my boss
Postpone the interview at 10 to Monday.

In general, taking the context into account during parsing allows the parser to focus on certain constructions rather than others, as well as to compute certain meanings more accurately. Using context to restrict the constructions that are triggered during parsing greatly reduces the number of edges our parser has to consider, resulting in the increase of speed. The dependence of meaning on context is well known, e.g. deixis or deciding the reference of pronouns. But there are very natural examples not connected to deixis: in our dialog, the expression *8* is interpreted only as a time expression (and not a place or something else), because of the context of the preceding question (asking about the time of the meeting). Similarly, for computing the meaning of other time expressions: *4 to 6* is more likely to mean "beginning at 4 and ending at 6" than "5:56" in the context of a question about the time of a meeting. But by using our knowledge of the application, we can completely eliminate the second reading, since all meetings are scheduled in 5 minute increments.

While context is certainly a very important issue, it is not the only one. The most general one would be the problem of deciding what should be done by the parser and what by the reasoning module. E.g. if *8* should be interpreted as the time of the meeting, it can be done in many ways — by means of a discourse construction, in which an answer is interpreted as the value of the parameter the question was about; or by a procedure in the interpretation module (our current choice); or by a dialog rule guiding the building of the discourse model.

Other questions include the role of defaults — for instance, should the logics of linguistic, domain, and application defaults be the same; availability of information for the dialog component – e.g. whether only value of the slots or also the parses should be available.

[1] We will use the convention that *np, np(x), vp, vp(_)* etc. refer to elements of our construction grammar, while traditional linguistic/syntactic categories will be denoted by NP, VP, S etc.

What is common to all these questions is the lack of commonly accepted and precise criteria upon which they could be decided. In [18] we attempt to use variants of Kolmogorov complexity as the basis for such criteria. Roughly speaking, we should be able to justify on that basis our choice of the construction grammar as minimizing certain kinds of semantic complexity. The next section provides linguistic fundamentals for such arguments.

3 Arguments for a pairing of form and meaning

In this section we first present arguments for a pairing of form and meaning. Namely, that such a pairing is needed to accurately describe NL constructions, both "standard" and "non-standard". However, we are not going to argue for the unsuitability of syntax for describing a "core" of language or a universal grammar. We believe, that such a language core is small, and that some other methods are needed to handle phenomena of real languages. Our aim is not to come with linguistic generalizations about structural properties of sentences (although, of course, we have nothing against them), but to come with an effective method for natural language understanding, which in addition to computational effectiveness would also have some linguistic and psychological plausibility.

We begin by noting that it has been the experience of of many people working in the field of computational linguistics that combining syntactic and semantic information increases the efficiency and effectiveness of processing. This experience has been confirmed by the experiments of Lytinen [8], who has shown that a semantically driven approach is superior to syntax-first approach in processing text in narrow domains. What follows provides an explanation why this is the case, and why pragmatics will also have to be taken into account.

3.1 The syntax and semantics of some standard constructions

There are many linguistic phenomena that can be naturally described by the pairing of their syntactic form with their semantic features. Such patterns of interdependence of syntax and semantics are common both for standard constructions, like NPs, VPs, or Ss, and for more exotic ones, such as sentences with "let alone". Attempts to account for those phenomena in systems without such interdependence of syntax and semantics would lead to drastic overgeneralizations.

As an example involving familiar constructions, let us begin with conjunctions. It is easily seen that particular conjunctions select clauses with a specific logical relation between their respective meanings:

**This is a regular chair, but you can sit on it.*

This is an old chair, but you can sit on it.

In this case, the two phrases conjoined by "but" agree if the second clause contradicts what is known about entities mentioned in the first clause. This kind of agreement cannot be described without access to semantic information and world knowledge.

As another example, consider the role of the determiner "a" in NPs. An NP can only very crudely be described as (Det) (ADJ)* N , for we would like to permit *It's time*, but not *It's book*. The sentences *Give me freedom/love/ time/attention!* are fine, but not *Give me book/bicycle/chair/eagle!*. This is so, because bare NP's are acceptable if the noun is mass/uncountable, which is semantically significant and context dependent.

Virtually every construction of English displays a similar interplay of form and meaning. We can refer the reader to [10] for a discussion e.g. of the semantic significance of the order of modifiers of NPs, on the semantic conditions on the progressive, etc.; and to [19] for the meaning of the conjunction "but".

3.2 The pragmatics of standard constructions

Context, especially dialog context, changes the range of possible constructions. Everybody knows that *Kissinger thinks bananas* makes sense as an answer to the question *What is (was) Nixon's favorite fruit*. But, in spoken discourse context, some constructions that look natural in writing might appear strange, e.g. *Schedule a 30 minute meeting with Pat, Bob and Martin about the new office system at 4:30 pm on Tuesday December the 6th 1994 in room 202* is too long, although "grammatical". On the other hand, in spoken dialogs, we find an interesting problem of processing fragments of discourse; there, "afternoon" might mean "in the afternoon", "two" the second of the choices (e.g. for a meeting time), and "checking" can stand for "from (the) checking (account)", i.e. semantically be a "source". In general, computing the meaning of an utterance (truth-conditional or otherwise) is impossible without pragmatics. Consider e.g. the difference in the meaning of "now" in the following two sentences:

Send it now.

She's now 27 years old.

Computational pragmatics is possible, because most common constructions have standard pragmatic interpretations associated with them. Thus, isolated *VPs* are typically interpreted as imperative sentences, and in normal dialogs are understood as *directives* (orders, offers, etc.). Declarative sentences are interpreted as *statements/assertions*, and interrogative sentences as *questions*. And vice versa, there are standard constructions for expressing assertions, questions, and commands (e.g. imperative sentences or *I want you to do X*). Moreover, not only these standard constructions have explicit pragmatic effects, but they also have explicit pragmatic preconditions, e.g. speaker and hearer must share the situation, the agent who asks the question should believe the other party knows the answer, etc. Finally, in restricted domains, implicature, presupposition and deixis is tractable. That is, the relevant aspects of the domain of discourse and typical inferences can be encoded, and used in the process of understanding. To see it, consider temporal expressions such as *X hours later*; since our model of time and event structure are simple, their meanings are relatively easily computable.

3.3 Open idioms

Standard grammars do not handle *open idioms* ([2]) such as

1. *The more you work, the easier it will get.*
2. *Him be a doctor?*
3. *One more and I'll leave!*
4. *Rain or no rain, I'll go for a walk.*
5. *He doesn't eat fish, let alone shrimp.*

Each of these expressions "exhibits properties that are not fully predictable from independently known properties of its lexical make-up and its grammatical structure" [2] , p. 511. All of them have a certain form, but could not appear in standard printed dictionaries, because they do not involve specific words (sentence (2) and *Them go to France?* do not share any words). This makes it doubtful that lexicalist approaches could deal with the problem of representing open idioms and parsing sentences that include them. Similarly, standard grammars would have trouble handling them: it is not clear what the phrase structure of (1) is; in (2) the subject is not in the nominative; is (3) a strange example of coordination? Neither is it clear how to ascribe a natural compositional semantics to (4) — is "X or no X" a tautology, and if not, what is it and how should it be connected with the sentence following it? Fillmore et al. give an explanation of how (5) works, but they appeal not only to semantics, but also to pragmatics, showing that the constructions expresses a conflict between the Gricean maxims of relevance and quantity (informativeness). Our next example is a verbless sentence:

6. *H.W.Fowler his name.*

The structure of this sentence is $S \rightarrow NP\ NP$. But, clearly, we should distinguish this particular kind of sentences from simple concatenations of arbitrary noun phrases, such as *a dog a rose*. This, however, is impossible without help from semantics, as this construction only can be used with proper names and names of professions.

We will use (4) to show how constructions can combine form, meaning and context. Let us now discuss briefly its meaning. Roughly, the expression "X or no X, S" says something like this: the subject of S does not care about X, and S is true; also, X and S should be somehow related, since *c-command or no c-command, I'll go for a walk* is odd. Actually, we can be more specific about the meaning of the expression: it talks about a conflict between two goals, e.g. the necessity of walking a dog and avoiding the rain, and about the resolution of this conflict, i.e. choosing to go for a walk.

Obviously, this is not the only expression of English dealing with a conflict. Actually there is a common pattern linking this construction with others, such as sentences with "but", "despite", "although", "however", or less standard expressions such as *Much as I like Ronnie, I don't approve of anything he does.* In all these cases there is some relation between two parts of an expression (this is a relation of coherence, cf. [3], [9]), there is a conflict, e.g. between goals or knowledge about a discourse entity, and a resolution of this conflict. Furthermore, the type of the conflict cannot be determined without access to a body of background (non-linguistic) knowledge, something we called the *referential level* (in [19] and [17]). Finally, it is possible to express the meaning of all these expressions, but we need a richer ontology than entities and truth values. Namely, we have to introduce expectations, actions, states, and theories (background knowledge). (With that kind of theoretical apparatus, one can easily make linguistic generalizations, also across various languages).

4 Constructions: integrating forms, meanings and contexts

In this section we address theme (A): we show an alternative to the standard design in which syntax, semantics and pragmatics are separated. The formalism for encoding constructions will be introduced by an example, which will be followed by a discussion of the language of features (or *taxemes*) and methods for describing meanings and contexts.

A *construction* is given by the matrix:

$$\begin{bmatrix} \mathbf{N} & : \textit{name\_of\_construction} \\ \begin{bmatrix} \mathbf{C} & : & \textit{context} \\ \mathbf{V} & : & \textit{structure} \\ \mathbf{M} & : & \textit{message} \end{bmatrix} \end{bmatrix}$$

The *vehicle* **V** consists of formulas describing presence (or perhaps absence) of certain *taxemes*, or features of form, within the structure of the construction. Such a structure is given by a list of subconstructions and the way they have been put together (in all our examples this is concatenation, but there are other possibilities, e.g. wrapping). The *context* **C** consists of a set of semantic and pragmatic constraints limiting the application of the construction. It can be viewed as a set of *preconditions* that must be satisfied in order for a construction to be used in parsing. The *message* **M** describes the meaning of the construction, via a set of syntactic, semantic and pragmatic constraints. A grammar is a collection of constructions.

Standard syntactic grammars can be embedded into our formalism by assuming that all constructions have empty contexts and meanings. Under such a mapping, some constructions sharing the same structure will be made identical, e.g. some questions and requests, or the progressive referring to future events and to present ones. Montague-type grammars correspond to sets of constructions with empty preconditions and very restricted kinds of meanings.

4.1 Example: the "X or no X"-construction

Our first example of construction will be non-standard. The reason for this is to show that difficult cases can be handled by the formalism. We will see how the system handles standard cases in Section 6.

$$
\begin{array}{l}
\mathrm{N} : X \mathit{ornoX} \\
\left[\begin{array}{lcl}
\mathbf{C} & : & \left[\begin{array}{l}(\exists d_1, d_2)[goal(d_1)\&goal(d_2)\& \\ curr\_state\&next\_action \Rightarrow d_1\& \\ curr\_state\&next\_action \Rightarrow \neg d_2]\end{array}\right] \\
\mathbf{V} & : & \left[\begin{array}{l}struc((np(X)."or"."no".np(X).sent(Y))) \\ cons\_name(Y, sent(decl)) \\ cons\_name(X, np)\end{array}\right] \\
\mathbf{M} & : & \left[\begin{array}{l}curr\_state := msg(X) \\ next\_action := msg(Y)\end{array}\right]
\end{array}\right]
\end{array}
$$

The construction applies in the context where the current state is described by the message of X (e.g. "rain"), and the agent has two goals (e.g. "to walk a dog" and "to stay dry"), of which only one would be accomplished if the action described by the message of Y is taken (e.g. "going for a walk"). The symbol $\Rightarrow$ reads "accomplishes" — we assume that it has a semantics defined e.g. by a set of axioms (meaning postulates); $msg(Y)$ reads "the message of (the construction) Y", and is a way of referring to the meaning of the subconstruction Y . The structure of the present construction is described by using concatenation "." and the function $cons\_name(X, Y)$ which gives the name of the construction X; (this is enough to express recursive definitions).

4.2 The language of constructions

4.2.1 Expressing forms

Constructions are recognizable (in semantic contexts) by their structural properties. As we have seen, these are specified by the function $cons\_name(X, Y)$ where Y is the name of the construction associated with X, the concatenation, and a collection of features which we call *taxemes*, for instance, "accusative" or "infinitival" (cf. also [20]).

Since in practice generative grammars use only taxemes that are "meaningful", i.e. one can associate some meanings with their presence, we can include in our description of constructions the features used by GB, GPSG or PEG (cf. e.g. [14] and [4]). Thus we can use properties like *agreement, dominates, c-commands* etc., as well as their lexical markings.

4.2.2 Expressing meanings

We can hope that a substantial fragment of linguistic meanings can be classified in the same fashion as we classify sets of syntactic features. Talmy [15] presents a catalogue of semantic relations which are lexically expressible (in different languages). We believe that an extension of this list can be used to classify meanings of all constructions.

From the point of view of formal languages used to describe natural languages, syntactic and semantic markers behave similarly — they are simply sets of constants and relations. To see this parallel, let us consider the following list:

Forms	Meanings
agree(x,y,f)	*conflict(x,y,f)*
_	*affects(x,y,f)*
dominates(x,y)	*implies(x,y)*
is_present(feature, phrase)	*is_true(formula, model)*
_	*believes(a,p)*
case	*theme/topic*
NP, PP , ...	*action, state, agent,...*
The-X-er, the-Y-er	*scale*
(...)	(...)

In general, there is no one-to-one correspondence between syntactic features and semantic elements. For example, the fact that an agent believes a proposition can be expressed directly by a construction such as *John thinks that...*, or it can follow from assumptions about discourse (for example, by the Gricean cooperative principle you are allowed to assume that speakers believe what they say).

4.3 Expressing contexts

There is no agreement on what is context, or, even more important, on what is not. But, again, from the point of view of an abstract computing device, a context is a collection of relations that are interpreted as constraints. A partial list of such relations can be found in the books on pragmatics (e.g. [7]), and it includes such relations as *speaker, hearer, topic, presupposed, assertion, question, command, declaration, ...* . To this list we can add *previous_utterance, current_domain, current_application, current_question, default_value,* etc.

For example, we can analyze expressions *No, but I'll do it now/send it tomorrow/...*, typically given as an answer to a question whether something has been done, as a discourse construction. Its description uses such relations as *previous_utterance/p_utter, previous_sentence/p_sent*, the message of S, $msg(S)$.

$$
\begin{array}{l}
\mathrm{N} : sent(assrt, no.but.S) \\
\left[\begin{array}{lcl}
\mathbf{C} & : & [p\_utter(X)\&cons\_name(X, sent(ques, \_))] \\
\mathbf{V} & : & \left[\begin{array}{l}struc("no"."but".S)\& \\ cons\_name(S, sent(assrt, \_))\end{array}\right] \\
\mathbf{M} & : & [p\_sent(Y)\&truth\_value(Y, 0)\&msg(S)]
\end{array}\right]
\end{array}
$$

As we can see, the construction applies only in the context of a previously asked question, and its message says that the answer to the question is negative, after which it elaborates the answer with a sentence S.

5 The relationship between grammar and knowledge representation

We now address theme (R). Having argued for a close coupling of syntax, semantics and pragmatics, we now can discuss the relationship between the construction grammar and the knowledge representation language.

5.1 Embedding ontology into grammar

In our construction grammar we use both a richer set categories and a larger set of taxemes than provided by any of the generative grammars. In particular our representation of constructions is closely coupled with a semantic taxonomy. Thus, for instance, not only do we have an *np* construction, but also such constructions as *np(place), np(duration), np(time), np(time(hour))* etc. In other words, the semantic hierarchy is reflected in the set of linguistic categories. (Notice that categories are not the same as features). Hence we do not have a list, but a tree of grammatical categories.

Example. *Time* is divided into the following categories *minute, hour, weekday, month, relative, day_part, week_part, phase*, and more categories could be added if needed, e.g. *century*. Some of these categories are further subdivided: *day_part* into *afternoon, morning, ...*, or *relative(_)* (e.g. *in two hours*) into *hour, minute, day_part, week_part, weekday.*

Note that (a) different constructions can describe the same object of a given category, for example *hour* can be given by a numeral or a numeral followed by the word "am" or "pm"; (b) there is a continuum of both categories and constructions describing objects of a given category, for instance, we have the following sequence of types of constructions

$$np > np(time) > np(time(month)) >$$

$$> np(time(month(january))) = january$$

corresponding to the sequence of categories

$$entity > time > month = time(month) >$$

$$> january = time(month(january))$$

In the first case *january* is a word (and words are constructions, too), in the second case it is a concept.

5.2 Some questions

1. What about verbs and their hierarchies? We have no hierarchy of actions, and no need for a hierarchy of verbs at present. But perhaps when related actions can appear as parameters of a plan such a need would arise. However, it does not seem likely that we would have a direct correspondence between the meaning of verbs and types of constructions.

2. What is the relationship between ontologies and case systems? Copeck et al. [1] list 28 cases grouped in 5 classes (eg. *space (direction, orientation, location_to, ...), or causality (cause, contradiction, effect, ...)*). Clearly, there is a relationship between cases and concepts in knowledge representation. Can it be made explicit?

3. Can anything interesting be said about e.g. NL expressions in a domain or application area and its ontology? E.g. can we say that the mapping from ontology to a hierarchy of *nps* should extend to the ontology of an application? It seems that although we can say something interesting about the relationship between *domain* concepts and the NL grammar, such a relationship between *application* concepts and the grammar remains an open problem. Notice that in semantic grammars such a relationship is very close, but, in general, it cannot be too close if we want to reuse a grammar for another application. What is the optimal solution then?

6 Parsing with constructions – examples

Parsing with constructions differs a bit from syntactic parsing. First, the collection of features that are used to drive parsing is richer, because it contains terms with semantic and pragmatic interpretation. Second, semantic and pragmatic information is used during parsing. (This is not unlike Lytinen's [8] semantic-first algorithm, where grammar rules apply on the basis of "desirable semantic attachments between adjacent constituents", after such constituents have already been identified). Third, structures assigned to strings are different, namely, the structural information is lost; instead the meanings/messages are produced.

We now present two examples. First, let us consider the path from a complete sentence to its interpretation produced by the semantic engine. Next we discuss processing of a fragment.

```
| ?- sem.
|: I want to set up an appointment
   on November 11.
[i,want,to,set,up,an,appointment,on,november,11]
***Slots:
[  [  action_name schedule]
   [  event_name meeting]
   [  event_time
      [  [  month 11]
         [  day 11]
```

The path to an interpretation is encoded in the edges of the chart produced by the parser. Some of the information from a sample of edges is given below. We do not show syntactic information, because syntactic features are standard (e.g. `plural, 3rd, accusative, ...`).

```
| ?- li.
  Chart results: INACTIVE EDGES
* 0,1,[i] :
  pron(person) -> [i]
  [[type,person],[den,speaker]]
* 0,1,[i] :
  np(person) -> [pron(person)]
  [[type,person],[den,speaker]]
* 3,7,[set,up,an,appointment] :
  vp([]) -> [v(phrasal),np(event)]
  [[action,[[den,set_up]]],
   [object,[[type,event],[den,appointment],
            [mods,[[det,an]]]]
* 0,10,
  [i,want,to,set,up,an,appointment,
   on,november,11]
   s(assert(svc([]))) ->
    [np(person),v([]),prep(to),s(imp([]))]
  [[den,want(action)],
```

```
    [mental_agent,[[type,person],
                           [den,speaker]]]
    [action,[[den,set_up]]],
    [object,[[type,event],
             [den,appointment],
         [mods,[[det,an],
                [pp_msg,[[type,event_time],
                    [den,[[month,11],[day,11]
* 0,10,
  illoc(s(imp(_))) -> [s(assert(_))]
```

In (0, 1), i is first identified as a personal pronoun which denotes the speaker. This information comes from the lexicon. Since pronouns are treated as *nps*, the *np(person)* category is assigned to the same string.

In (3, 7), we see the message of the phrasal *vp*, which then in (0, 10) becomes a the main part of the complement of the verb *to want*. The meaning of the verb *to want* depends its complement structure: *want(action)* requires a *to s(imp(_))* complement and an *np(person)* subject, while *want(object)* requires an *np* complement, and *want(sex_object)* requires a complement of type *np(person)*. Finally, (0, 10) is interpreted as an having the illocutionary force of an imperative statement (because that is the meaning of the verb, and because in this application all statements are first attempted to be interpreted as commands).

The next example shows the processing of fragments. Getting the slots from a fragment is possible only in a very restricted context, when the meaning of such a fragment can be unambiguously computed.

```
| ?- sem.
|: on November 11th with Martin.
[on,november,11,th,with,martin,.]
***Slots:
[  [  event_time
      [  [  month 11]
         [  day 11]
   [  event_partner
      [  martin]

| ?- li.
  Chart results: INACTIVE EDGES
* 0,1,[on] : prep(on) -> [on]
* 1,2,[november] : n(time(month)) -> [november]
  [[type,time(month)],[den,11]]
* 1,2,[november] :
  np(time(month)) -> [n(time(month))]
* 2,4,[11,th] :
  ordinal -> [numeral,st_nd_rd_th]
  [[den,11]]
```

The preposition *on* by itself does not carry any meaning, we treat it as a lexicalized feature. In (2, 4), to prevent overgeneralizations such as *3 th*, the vehicle of the rule should contain the list of cases *if 1 then st, if 2 then nd, ...*

```
* 1,4,[november,11,th] :
  np(time(day)) -> [np(time(month)),ordinal]
  [[type,time],[den,[[month,11],[day,11]]]
* 0,4,[on,november,11,th] :
  pp(on,time) -> [prep(on),np(time(day))]
  [[type,event_time],[den,[[month,11],[day,11]]]
* 0,4,[on,november,11,th] :
  pp_list(on,time) -> [pp(on,time)]
* 0,6,[on,november,11,th,with,martin] :
  pp_list(with,person) ->
          [pp(on,time),pp_list(with,person)]
  [[pp_msg,[[type,partner],[den,martin]]],
   [pp_msg,[[type,event_time],
            [den,[[month,11],[day,11]]]]
* 0,6,[on,november,11,th,with,martin] :
  pp_list(on,time) ->
            [pp_list(on,time),pp(with,person)]
```

The meaning of the *pp* (0, 4) is a simple transformation of the meaning of the *np* in (1, 4). But notice that this happens because we have a specific construction which says that a combination of *on* with an *np(time(_))* produces *event_time*. Altogether, we have encoded a few dozens various *pp* constructions, but in a given application only a fraction of them are used.

In the same context, please note that, because of the close relationship between the domain ontology and the hierarchy of constructions, we can also postulate a close relationship between the type of meaning a construction expresses and its category (i.e. name), for example, the type of meaning of *np(time(hour))* is `[type, event_type]`.

At the end, we obtain two parses of (0, 6), but they have the same messages. The difference lies in the category assigned to *pp_list*; and in this particular case the choice of that category is not important.

We have experimented with different representations for constructions. All of them share the basic scheme of *Name, Context, Vehicle, Message*, but differ in the implementation details. For instance, we have built two parsers of constructions, of which one was a postprocessor to a standard grammar — the English Slot Grammar (ESG) [11], and its working was described in [12]. We used ESG to compute the vehicles of constructions, and built the construction processor separately. The advantages of building the postprocessor were twofold: we could assign very fast basic syntactic structures to strings, and this allowed us to concentrate on how constructions could improve parsing; secondly, we got much better coverage than if we tried to build a parser of constructions from scratch. But the disadvantages outweighed the advantages. It turned out that to account for the meanings of *pps* we have to redo the treatment of *pps* from scratch; and having to translate between different representations proved very tedious.

In another application, we used our collection of temporal constructions as a basis for a text skimmer. M. Harelick has built a transducer for finding date expressions and computing their meanings. Given that building a wide coverage construction grammar for English is not likely to happen soon (because semantics and pragmatics seems to be more difficult than syntax), we can imagine that construction grammars can be used for text processing and message understanding, provided the types of meanings are simple (e.g. dates, money, pro-

motions).

7 Conclusions, some unexpected

We have already seen that the two issues (A) and (R) are related. That is, we could not postulate a close coupling of ontology and a *syntactic* grammar. In this section we will first discuss postulates for a construction grammar of Fillmore et al. [2], and how our formalism satisfies them. Then we will show how the two issues relate to the problem of compositionality (and this connection might be somewhat unexpected).

7.1 Satisfying postulates for a construction grammar

We have shown it is possible to precisely specify the notion of construction and parsing with constructions. The approach to language understanding we are advocating in this paper has several advantages: (1) it agrees with the facts of language; (2) it is not restricted to a "core" of language, but applies both to standard and more exotic constructions; and (3) it is computationally feasible, as it has been implemented in a complete working system. Our formalism agrees with the proposal of [2], and, by being formal, it exemplifies and clarifies it; in particular this applies to the following claims:

I. "The proper units of a grammar are more similar to the notion of construction in traditional and pedagogical grammars than to that of rule in most versions of generative grammar. This is not to say that the generative ideal of explicitness is foregone; nor is the necessity of providing for recursive production of large structures from smaller ones set aside."

— The parser of constructions takes care of the explicitness and recursion. By proposing concrete data structures and a set of procedures for parsing with constructions, we have actually exceeded the usual standards of explicitness in generative grammars.

II. "Constructions are much like the nuclear family (mother plus daughters) subtrees admitted by phrase structure rules, **except** that

1. constructions need not be limited to a mother and her daughters, but may span wider ranges of the sentential tree;
2. construction may specify, not only syntactic, but also lexical, semantic, and pragmatic information;
3. lexical items, being mentionable in syntactic constructions, may be viewed, in many cases at least, as constructions themselves."

   — Notice, that words have both form and meaning; the preconditions consist e.g. of an assumption about which language we are dealing with (this is not that obvious if a text mixes two or more languages).
4. "Constructions are idiomatic in the sense that a large construction may specify a semantics (and/or pragmatics) that is distinct from what might be calculated from the associated semantics of the set of smaller constructions the could be used to build the same morphosyntactic object."

   — The reader should have noted that constructions are idiomatic by definition – if the meaning of a combination of taxemes is completely predictable there is no need for a construction with such a structure.

The innovations we have proposed — the close coupling of semantic hierarchies with linguistic categories, the use of context in representing linguistic knowledge, and the representation of the grammar as a dictionary of constructions — not only facilitate the development of the grammar, but also its interaction with the inference engine.

Obviously, the usefulness of this approach is not limited to natural language interfaces. We have used parts of the grammar of constructions for some information retrieval tasks; and we can easily imagine it being applied to text skimming and machine translation in limited domains.

7.2 Representation and compositionality

Since the issues of reasoning and semantics are closely related, we should discuss also the issue of *compositionality*. The idea of compositionality is used to account for the ability of the language user to understand the meaning of sentences not encountered before. The new sentence can be understood, because it is composed of parts (words) that the user knows, and the meaning of the new sentence is computable from the meanings of the words that compose it.

Compositionality is defined as the property that the meaning of a whole is a function of the meaning of its parts (cf. e.g. [6], pp.24-25). Very often it is expressed as the postulate of the existence of a homomorphism from syntax to semantics (e.g. [13]). Logical formalisms, and many of the formalisms used in knowledge representation, satisfy this postulate. For example, the meaning of a (logical) conjunction of two statements is its truth value, which depends only on the truth values of the conjuncts.

But for natural language, although very intuitive, this idea is problematic. First, we can prove a theorem stating that any semantics can be encoded as a compositional semantics, which means that, essentially, the standard definition of compositionality is formally vacuous [16]. Secondly, if we believe in a construction based approach, without autonomous syntax there is no homomorphism from syntax to semantics.

As it turns out, a grammar of constructions can account for the ability of the language user to understand the meaning of novel sentences without separating the language into syntax, semantics and pragmatics. Namely, the meaning of a larger construction is a combination of the meanings of its parts (and the way they are put together). The message of the larger construction specifies how its meaning depends on the meanings of the parts.

Furthermore, construction-based approach to grammar can overcome the problem of vacuity of compositional semantics. In [16] we argued that one should put restrictions on the concept of compositionality, and

showed that such restrictions can make the concept non-vacuous. Although we have not said it explicitly before, we have assumed that the meaning functions used in the message part of constructions are cognitively justified. This point has been made e.g. by Jurafsky [5], who proposed another construction formalism. (The differences between our formalism and the formalism of Jurafsky's are discussed in [20]. The most significant is no account of the role of context in the latter; on the other hand Jurafsky's thesis contains treatment of such syntactic phenomena as gap filling, wh-questions, etc., which we have not covered so far. Furthermore he pays attention to the psychological adequacy of his model).

These issues of constraints on compositionality and psychological adequacy bring us to the idea of embedding an ontology into the grammar. If we postulate that the *np(_)* hierarchy corresponds to a natural ontological hierarchy, we have both satisfied the postulate of psychological adequacy and imposed constraints on the types of meanings. This is perhaps an unexpected benefit of our approach.

Summarizing, the most important innovations implemented in the system are a close coupling of semantic hierarchies with the set of linguistic categories; the use of context in representing linguistic knowledge, esp. for discourse constructions; and a non-lexicalist encoding of the grammar in a dictionary of constructions. We have obtained a new language model in which forms cannot be separated from meanings. We can talk about meaning in a systematic way, but we do not have compositionality described as a homomorphism from syntax to semantics (This is theoretically interesting, also because it is closer to intuitions than current models of semantics). We have validated this model by building prototypes of a natural language interfaces.

References

[1] T. Copeck, S. Delisle, and S. Szpakowicz. Parsing and case analysis in Tanka. 1992.

[2] C.J. Fillmore, P. Kay, and M.C. O'Connor. Regularity and idiomaticity in grammatical constructions. *Language*, 64(3):501–538, 1988.

[3] J. R. Hobbs. Towards an understanding of coherence in discourse. In W.G. Lehnert and M.H. Ringle, editors, *Strategies for Natural Language Processing*, pages 223–244. Lawrence Erlbaum Associations, Hillsdale, New Jersey, 1982.

[4] K. Jensen. Parsing strategies in a broad-coverage grammar of english. Technical Report RC 12147, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1986.

[5] D. Jurafsky. *An On-line Computational Model of Sentence Interpretation.* PhD thesis, University of California, Berkeley, 1992. Report No. UCB/CSD 92/676.

[6] E. L. Keenan and L. M. Faltz. *Boolean Semantics for Natural Language.* D Reidel, Dordrecht, Holland, 1985.

[7] S. G. Levinson. *Pragmatics.* Cambridge Univercity Press, Cambridge, MA, 1985.

[8] S.L. Lytinen. Semantics-first natural language processing. In *Proceedings AAAI-91*, pages 111–116, Anaheim, CA, 1991.

[9] W.C. Mann and S.A. Thompson. Relational propositions in discourse. Technical report, Information Sciences Institute Research Report, Marina del Rey, CA, 1983.

[10] J. D. McCawley. *The Syntactic Phenomena of English.* University of Chicago Press, Chicago, IL, 1988.

[11] M. McCord. A new version of slot grammar. Technical Report RC 14710, IBM Research, 1989.

[12] M. McCord, A. Bernth, S. Lappin, and W. Zadrozny. Natural language processing technology based on slot grammar. *International Journal on Artificial Intelligence Tools*, 1(2):229–297, 1992.

[13] B.H. Partee, A. ter Meulen, and R. E. Wall. *Mathematical Methods in Lingusitics.* Kluwer, Dordrecht, The Netherlands, 1990.

[14] P. Sells. *Lectures on Contemporary Syntactic Theories.* CSLI Lecture Notes (3), Stanford, CA, 1985.

[15] L. Talmy. Lexicalization patterns: semantic structure in lexical forms. In T. Shopen, editor, *Language Typology and syntactic description Vol.III*, pages 57–149. 1985.

[16] W. Zadrozny. From compositional to systematic semantics. *Linguistic and Philosophy*, (4):329–342, 1994.

[17] W. Zadrozny. Reasoning with background knowledge – a three-level theory. *Computational Intelligence*, 10(2):150–184, (1994).

[18] W. Zadrozny. Measuring semantic complexity. *submitted to Workshop on Computational Semantics*, 1994.

[19] W. Zadrozny and K. Jensen. Semantics of paragraphs. *Computational Linguistics*, 17(2):171–210, 1991.

[20] W. Zadrozny and A. Manaster-Ramer. The significance of constructions. *submitted to Computational Linguistics*, 1994.

[21] W. Zadrozny, M. Szummer, S. Jarecki, D. E. Johnson, and L. Morgenstern. NL understanding with a grammar of constructions. *Proc. Coling'94*, 1994.