Representing and Integrating Linguistic Knowledge*

Daniel Jurafsky
573 Evans Hall
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720
jurafsky@teak.Berkeley.EDU

Introduction

This paper describes a theory of the representation and use of linguistic knowledge in a natural language understanding system. The representation system draws much of its insight from the linguistic theory of Fillmore *et al.* (1988). This models knowledge of language as a large collection of *grammatical constructions*, each a description of a linguistic regularity. I describe a representation language for constructions, and principles for encoding linguistic knowledge in this representation. The second part of the theory is a conceptual analyzer which is designed to model the on-line nature of the human language understanding mechanism. I discuss the core of this analyzer, an information-combining operation called *integration* which combines constructions to produce complete interpretations of an utterance.

Representation

A natural way to model knowledge of language is as a large collection of facts or regularities about the language. In this theory I use a single representational device, the *grammatical construction*, to represent these regularities. Our entire model of linguistic knowledge consists of a database of these constructions, uniformly representing lexical knowledge, syntactic knowledge, and semantic knowledge.

Although the notion of grammatical construction I use is based on that of Fillmore *et al.* (1988) it differs in a few respects. Fillmore *et al.* define a construction as a structure which represents a "*pairing of a syntactic pattern with a meaning structure*". Such a construction is a sign in the sense of Saussure, or a rule-pairing in the sense of Montague. I use a somewhat extended notion: a construction is a relation between one information structure and one or more others. These information structures can be semantic, syntactic or both. Thus, whereas the "sign" expressed a relation between a set of ordered phonemes and a meaning structure, the construction abstracts over this by replacing 'ordered sets of phonemes' with abstractions over them. These abstractions can be syntactic or semantic ways of expressing more abstract categories to which these phoneme sequences belong. The construction is then a part-whole structuring relating these categories.

An Example Construction

To make this idea more concrete, consider some specific examples from the grammar, which currently includes about 30 constructions. In the examples I will be focusing on the knowledge that is necessary to handle the sentence:

> *"How can I find out how much disk space I am using?"*

The top-level construction for the sentence is the **WhNonSubjectQuestion** construction. This construction is a sub-type of the **WhQuestion** construction. The wh-questions are those which begin with a wh-element — an element which is semantically questioned. In the **WhNonSubjectQuestion** this questioned element begins the sentence but does not function as the grammatical subject of the sentence. Following are two other examples of **WhNonSubjectQuestions:**

> *Why did you run the wrong way?*
>
> *What did you pick up at the store?*

The construction is represented in figure 1 below:

```
(constr WhNonSubjectQuestion (Freq p)
  [(a $q
       ($q AIO Question)
       ($var -Queried $q)
       ([$\v $\presup] -Presupposed $q))]
   ->
  [(a $t
      ($t AIO Identify)
      ($var -Specified $t)
      ($presup -Presupposed $t))]
  [(a $v
      ($v AIO SubjectSecondClause))])
```

Figure 1

Figure 1 can be summarized as follows: There is a construction in English called **WhNonSubjectQuestion.** It consists of two constituents, **$t** and **$v**, and a "whole", **$q**. (In the example sentence, the **$t** constituent consists only of the word "*How*",

*I want to express my thanks to Peter Norvig, Nigel Ward, and Robert Wilensky for many helpful comments and discussions on these ideas. This research was sponsored by the Defense Advanced Research Projects Agency (DoD), monitored by the Space and Naval Warfare Systems Command under N00039-88-C-0292, and the Office of Naval Research, under contract N00014-89-J-3205.

while the **$v** constituent consists of the phrase "*can I find out how much disk space I am using.*") These three elements consist of statements in a knowledge representation language based on Wilensky (1986). The representation language is relatively perspicuous, not differing greatly from other popular representation languages (such as KL-ONE (Brachman & Schmolze 1985)). The operator "**a**" creates an existentially quantified variable, and is followed by the variable name and a set of statements in its scope. The infix operator "**AIO**"(*An Instance Of*) establishes one element as an instance of another, and the infix operator "**AKO**" (*A Kind Of*) establishes one element as a sub-type of another.

The first constituent, **$t**, is an instance of the **Identify** concept, which will be described later. This concept contains two relations, **Identify–Specified** and **Identify–Presupposed.** (Note that they are referred to simply as **–Specified** and **–Presupposed.**) The second constituent, **$v**, must be an instance of the construction **SubjectSecondClause**. This is a clause where the subject follows an initial verbal auxiliary. The name **SubjectSecondClause** was chosen to replace the traditional term *Aux-Inversion* in order to avoid employing terminology that uses the "movement" metaphor.

The construction builds the **$q** element, which is an instance of the **Question** concept. Note that the variable which fills the **–Queried** relation of the **Question** is the same element that fills the **–Specified** relation in **$t**. The **–Presupposed** relation is filled with the integration of the semantics of **$t** and **$v**. The integration process is discussed below.

Each constituent (enclosed in square brackets) is composed of a set of semantic relations. These constituents correspond to what are called *informational elements* in the unification literature. Each consists of a group of semantic relations expressing some information about some linguistic entity.

The relation among these informational elements is represented by the right-arrow "$\rightarrow$" symbol. In the current version of the representational system, the right-arrow indicates the conflation of part-whole and ordering relations. That is, the default is for constructions to be ordered. In order to indicate a part-whole relation without the ordering relations, the right-arrow is followed by the keyword "UNORDERED". Figure 2 below provides an example of an unordered construction.

Features of the Representation

The representation language and the grammar offer a number of distinguishing features. First is the ability to define constituents of constructions semantically as well as syntactically. For example, note that the first constituent of the **WhNonSubjectQuestion, $t**, was defined as any informational element which is an instance of a certain **Identify** concept. The **Identify** concept indicates some question about the identity of the individual filling the **Identify-Specified** relation. The information that is known about the individual fills the **Identify-Presupposed** relation. The **Identify** concept is the main characteristic of the lexical semantics of the wh-words. Thus this constituent is specified *semantically*, simply by requiring the presence of the **Identify** semantics. In more traditional grammars, the constituent would be defined syntactically, as some sort of **WhPhrase**. Capturing the syntax of the **WhPhrase** would entail duplicating huge parts of the grammar or introducing significant syntactic apparatus. Since the semantics of **Identify** must be in the grammar anyhow, using it to specify the construction simplifies the grammar at no cost [1].

A construction's constituents can be constrained to be instances of other constructions. For example, the second constituent in figure 1 is constrained to be an instance of **SubjectSecondClause**. But these constraints are the only form of syntactic knowledge that this representational system allows. That is, constituency and ordering are the only syntactic relations in the system. All others are semantic. In keeping with these last two ideas, this representation dispenses with any enrichment of surface form. Thus phenomena which might traditionally be handled by gaps, traces, or syntactic coindexing are handled with semantic relations. Thus, as we saw above, the first constituent of the **WhNonSubjectQuestion** construction is specified semantically, and no gaps or traces are involved [2].

Relations Among Constructions

As with other kinds of knowledge, linguistic knowledge includes an abstraction hierarchy. An example of two constructions related by abstraction is given in figure 2. Constructions are also augmented by information concerning their frequency. The use of frequency information in comprehension (suggested in Bresnan 1982) is discussed in Wu (1989), and is not discussed further here.

I noted above that a construction may constrain one of its constituents to be an instance of some other construction. In a similar manner, a construction may require that *several* of its constituents participate together in a second construction. This is represented by including specifications for additional constructions after the keyword **WITH.** The relevant constituents of these extra constructions are then marked with the proper constituent variables from the main construction.

To illustrate the last few points, consider the **VerbParticle1** construction in figure 3. It is one of two constructions in which discontinuous phrasal verbs like "find out" or "look up" occur. These two constructions specify the ordering of the verb and particle, differing in that **VerbParticle2**, which covers phrases like "find it out", includes an extra

[1] This makes it more difficult for the parser to index constructions to consider. Because a construction's constituents may be defined by any set of semantic relations rather than by a small set of syntactic categories, it is no longer possible to simply look for a syntactic handle in the input. Indeed, the "construction access problem" becomes much more like the general memory access problem. In this sense this analyzer resembles the Direct Memory Access Parser of Riesbeck (1986).

[2] Jurafsky (1988) discusses how other phenomena classically handled by transformations and redundancy rules can be represented as constructions.

constituent. But **VerbParticle1** and **VerbParticle2** both require that their constituents be filled by phrasal verbs like "find out". This fact is represented by requiring that these constituents be instances of the unordered **LexicalVerbParticle** construction, which is the ancestor of all these phrasal verbs. In the representation of the **VerbParticle** construction below, constituents **$v** and **$p** are constrained to be the **$verb** and **$particle** respectively of a **LexicalVerbParticle** construction. Since **FindOut** is a subtype of **LexicalVerbParticle**, it meets the necessary constraints, and can integrate with **VerbParticle** [3].

```
(constr VerbParticle1 (Freq $z)
  [ (a $a
       ($a AIO Verb))]
  ->
  [(a $v)]
  [(a $p)]

  WITH
  (constr LexicalVerbParticle
         [ (a $a )]
          -> UNORDERED
         [ (a $v)]
         [ (a $p)]))
```

Figure 2

```
  (constr LexicalVerbParticle (Freq $z)
         []
         -> UNORDERED
         [(a $v)]
         [(a $p)])

(constr FindOut AIO LexicalVerbParticle
     (Freq $z)
  [(a $f
      ($f AIO FindingOut))]
  -> UNORDERED
  [$verb (word find)]
  [$particle (word out)])
```

Figure 3

The Basic Integration Operation

Associated with the representational system is an information-combining operation called *integration*, used here in two ways. The more complex one is discussed in the next section. The simpler version of integration is used to match informational elements to constituents of a constructions. The set of relations which constitute a constituent act as a set of constraints on any candidate constituents. For example, consider what semantics must be present in an informational element for it to be a constituent of the **HowScale** construction. Examples of this construction include:

How wide/strong/accurate ... ?
How much/often/quickly/far ...?
How many ...?

The construction has two constituents, the lexical item "how" and a second constituent which is semantically specified to be some sort of *semantic scale*. This constituent may be an adjective, an adverb, or a quantifier so long as it has the proper semantics. In the cases above, the scales range over such things as width, strength, speed, and amount. The construction takes a constituent with these semantics, and builds an instance of the **Identify** concept. The **–Presupposed** relation of this **Identify** concept is filled by the semantics of the constituent. The **–Specified** relation is bound to the **–Location** of the presupposed object on this scale. That is, the meaning of the construction is something like "*The location of object $z on scale $s is in question*".

```
(constr HowScale
  [(a $i
      ($i AIO Identify)
      ($x -Specified $i)
      ($s -Presupposed $i)
      ($x -Location $z $s)]
   ->
  [ $h (word how) ]
  [ (a $s
       ($s AIO Scale)
       ($z -On $s))]
```

Figure 4

In order for an informational element to integrate with the $s constituent of the **HowScale** construction, its semantics must *already* include an instance of a scale, with some object on the scale. As in standard unification, the constraints are matched with the elements in a recursive fashion, binding variables in the process. Unlike unification, a match cannot succeed if a candidate constituent is merely *compatible* with the required information. For example, the element **[(a $x ($x AIO Scale))]** would *unify* with the $s constituent. However, it would *not integrate*, because it lacks the *-On* relation. We can summarize the simple integration algorithm as follows:

> **Simple Integration Algorithm:** Unify the set of constituent relations with the relations present in the candidate, subject to the constraint that every relation in the constraint must already be instantiated in the candidate.

There are times when we want the semantics of unification rather than integration. That is, we may want to ensure that a certain relation is in the semantics of a construction regardless of which constituents it may also occur in. We can accomplish this by putting information in the "whole" of the construction's part-whole structure. For example, in figure 4, the information "**($x -Location $z $s)**" is added to the final semantics whether or not it was present in the $s constituent.

Thus there is an asymmetry in the application of the integration operation, caused by the part-whole

[3] Note that a parse which uses **WITH** constraints does not have a parse tree, but a parse *graph*. This is because both VerbParticle and LexicalVerbParticle would occur in the parse tree above the phrase "find out". However, this grammar obviates the need to keep a parse tree at all, as the grammar itself specifies how semantic integration is to be done.

structure of the construction. The relations in the constituent slots of the construction definition are viewed as *constraints* on candidate constituents. The relations in the "whole" element, on the other hand, are used as *instructions* for creating a whole semantic structure.

Note that this interpretation of the construction is limited to its use in comprehension. When a construction is used in generation the exact opposite situation holds — the "whole" element imposes constraints, while the constituents give instructions.

More Complex Integration

The algorithm above is sufficient to match candidate constituents to the constraints of a construction. But this sort of simplistic combination is not common in natural language when combining constructions. Integrating constituents usually involves modifying some of the structure of one of the constituents. A particularly common type of modification involves binding the value of one constituent to some open variable in another constituent [4]. That is, we must decide whether two relations are at the same level, or whether one should fill some semantic gap in another.

In some cases, deciding the level and finding an appropriate semantic gap can be quite simple. For example, part of the verb phrase construction is shown in figure 5 below. Note that the variable **$/v** is marked with a slash. This indicates that the verb **$v** is the matrix structure, while the complement **$c** is to be integrated to some gap inside of **$v**. This slash can be used for any variable to indicate that it is the matrix for some integration.

```
(constr VerbPhrase (Freq $p)
  [ (a [$/v $c])
  ->
  [ (a $v
          ($v AIO Verb))]
  [ (a $c)]
  )
```

Figure 5

For a more complex example, consider figure 1 (duplicated below), examining how the two constituents of the **WhNonSubjectQuestion** are integrated for the sentence "*How can I find out how much disk space I am using?*". I will not discuss the details of the **$v** constituent except to note that its semantics involve an **AbilityState** predicated of some person asking a question, and concerning some *finding-out* action.

The first constituent (**$t**) of the **WhNonSubjectQuestion** is filled here by the lexical construction **how6**, one of the various "how" constructions. **How6** is concerned with the *means* of some action, asking for a specification of the means or plan by which some goal is accomplished. We can ignore for

```
(constr WhNonSubjectQuestion (Freq p)
  [(a $q
       ($q AIO Question)
       ($var -Queried $q)
       ([$\v $\presup] -Presupposed $q))]
   ->
  [(a $t
      ($t AIO Identify)
      ($var -Specified $t)
      ($presup -Presupposed $t))]
  [(a $v
      ($v AIO SubjectSecondClause))])
```

Figure 1

now exactly how this sense is related to the other senses of "how".

```
(word  how6 (Freq x)
  [(a $h
      ($h AIO Identify)
      ($p -Specified $h)
      ($x -Presupposed $h)
      ($x AIO Planfor)
      ($p -Plan $x)
      ($g -Goal $x))])
```

Figure 6

Given the semantics of the two constituents of the **WhNonSubjectQuestion** construction, the final result of the integration will look something like figure 7.

```
  [(a $q
       ($q AIO Question)
       ($p -Queried $q)
       ($pr -Presupposed $q)
       ($pr AIO Planfor)
       ($p -Plan $pr)
       ($g -Goal $pr)
       ($g AIO AbilityState))]
```

Figure 7

Here in integrating **$/v** and **$/presup**, the algorithm finds a gap inside the second structure. That is, the integration algorithm integrated the Planfor of the first constituent with the AbilityState of the second, in effect unifying the variable **$g** in the PlanFor structure with **$v**, the AbilityState structure. Thus the complete algorithm might be expressed as follows:

> **Full Integration Algorithm**: Integrate the set of constituent relations with the relations present in the candidate by finding an appropriate gap (variable) in one of the two structures to integrate with the other.

Integration is an augmentation of unification. In order to handle more complex constructions, the operation would need to be augmented further, adding more inferential power. For example, a certain class of inference is required to integrate constructions like DoubleNoun (Wu 1990), where the relation between the constituents can be quite indirect and contextually-influenced. Such an integration algorithm might also need to make the kind

[4] This occurs in such common phenomena as WH-movement, Y-Movement, Topicalization, and other phenomenon classically analyzed as movement rules, where there is some long-distance link between some element and the valence-bearing predicate into which it integrates.

of metaphoric inferences studied by Martin (1988), or the abductive inferences of Charniak & Goldman (1988) or Hobbs *et al.* (1988). But rather than making these inferences in the pipelined fashion that these other mechanisms use, augmenting the integration algorithm allows these inferences to be made in an on-line manner.

Previous Research

The theory I have presented draws many elements from other theories of grammar, especially Fillmore *et al.* (1988). The use of ordered and unordered constructions is based on the ID/LP notation of Gazdar *et al.* (1985) The theory differs from these and most other grammars (such as Bresnan (1982), Marcus (1980), Pereira (1985)) in allowing semantic constraints on constituents (indeed in emphasizing them), and in disallowing syntactic relations other than ordering and constituency. It also differs by constraining the grammar to be semantically informative enough to allow the analyzer to produce interpretations in an on-line fashion.

As for the tradition which has concentrated on semantic analysis, this theory owes much to such systems as ELI (Riesbeck & Schank 1978) and the Word Expert Parser (Small & Rieger 1982). It differs from these in its commitment to representing high-level linguistic constructions, to the use of declarative representations, and to capturing linguistic generalizations.

A number of earlier systems have integrated linguistic knowledge with other knowledge, including PSI-KLONE (Bobrow & Webber 1980) and Jacobs's (1985) ACE/KING system, on which this theory draws heavily. The use of constructions here draws also on the *pattern-concept pair* of Wilensky & Arens (1980) and Becker's 1975 work on the phrasal lexicon.

The integration operation is based on unification in ways discussed above. Unification was first proposed by Kay (1979), and has since been used and extended by many other theories of grammar.

Conclusion

I draw three conclusions for the representation and use of linguistic knowledge:

1. Allowing constructions to include semantic as well as syntactic constraints removes a great deal of complexity from the syntactic component of a grammar. This is useful because no corresponding complexity is added to the semantic component. Semantic knowledge which must be in the system anyway is employed.

2. Committing to exploring semantically rich constructions like **HowScale** from a semantic perspective produces a grammar of sufficient richness to allow lexical semantics to influence the integration process, and thus the interpretation, in an on-line way.

3. Using a single representational device, the grammatical construction, avoids the proliferation of syntactic devices and simplifies the representational theory. This may also simplify the corresponding learning theory.

4. Extending unification-style approaches from syntax to semantics requires augmenting the feature-based unification operation to richer, relational knowledge representation languages. I have shown how one such extension has been implemented, allowing some gap-seeking intelligence to guide the integration process.

References

Becker, J. (1975). The phrasal lexicon. In R. Schank & B. L. Nash-Webber, editors, *Proceedings of the First Interdisciplinary Workshop on Theoretical Issues in Natural Language Processing*, Cambridge, MA.

Bobrow, R. J. & B. Webber (1980). Knowledge representation for syntactic/semantic processing. In *Proceedings of the First National Conference on Artificial Intelligence*, pp. 316–323. Morgan Kaufmann.

Brachman, R. J. & J. G. Schmolze (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216.

Bresnan, J., editor (1982). *The Mental Representation of Grammatical Relations.* MIT Press, Cambridge.

Charniak, E. & R. Goldman (1988). A logic for semantic interpretation. In *Proceedings of the 26th Annual Conference of the Association for Computational Linguistics.*

Fillmore, C., P. Kay, & M. C. O'Connor (1988). Regularity and idiomaticity in grammatical constructions: The case of let alone. *Language*, 64(3):501–538.

Gazdar, G., E. Klein, G. Pullum, & I. Sag (1985). *Generalized Phrase Structure Grammar.* Basil Blackwell, Oxford.

Hobbs, J. R., M. Stickel, P. Martin, & D. Edwards (1988). Interpretation as abduction. In *Proceedings of the 26th Annual Conference of the Association for Computational Linguistics*, pp. 95–103, Buffalo, NY.

Jacobs, P. (1985). A knowledge-based approach to language generation. Technical Report 86/254, University of California at Berkeley Computer Science Division, Berkeley, CA.

Jurafsky, D. (1988). Issues in relating syntax and semantics. In *Proceedings of the Twelfth International Conference on Computational Linguistics*, pp. 278–284, Budapest.

Kay, M. (1979). Functional grammar. In *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistics Society*, pp. 142–158, Berkeley, CA.

Marcus, M. P. (1980). *A Theory of Syntactic Recognition for Natural Language.* MIT Press, Cambridge.

Martin, J. (1988). A computational theory of metaphor. Technical Report 88/465, University of California at Berkeley, Computer Science Division, Berkeley, CA.

Pereira, F. C. N. (1985). Characterization of attachment preferences. In D. R. Dowty, L. Kartunnen, & A. Zwicky, editors, *Natural Language Parsing*, pp. 307–319. Cambridge University Press, New York.

Riesbeck, C. K. (1986). From conceptual analyzer to direct memory access parsing: An overview. In *Advances in Cognitive Science 1*, pp. 236–258. Ellis Horwood, Chichester.

Riesbeck, C. K. & R. C. Schank (1978). Comprehension by computer: Expectation-based analysis of sentences in context. In W. J. M. Levelt & G. B. F. d'Arcais, editors, *Studies in the perception of language*, pp. 247–293. Wiley, London.

Small, S. L. & C. Rieger (1982). Parsing and comprehending with word experts. In W. G. Lehnert & M. H. Ringle, editors, *Strategies for Natural Language Processing*, pp. 89–147. Lawrence Erlbaum, New Jersey.

Wilensky, R. (1986). Some problems and proposals for knowledge representation. In J. L. Kolodner & C. K. Riebeck, editors, *Experience, Memory, and Reasoning*, pp. 15–28. Lawrence Erlbaum, New Jersey.

Wilensky, R. & Y. Arens (1980). Phran – a knowledge-based approach to natural language analysis. Technical Report UCB/ERL M80/34, University of California at Berkeley, Electronics Research Laboratory, Berkeley, CA.

Wu, D. (1989). A probabalistic approach to marker propagation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 574–580, Detroit, MI. Morgan Kaufmann.

Wu, D. (1990). Probabalistic unification-based integration of syntactic and semantic preferences for nominal compounds. In *Proceedings of the Thirteenth International Conference on Computational Linguistics*, Helsinki.