Important Additional Notes about Dependency Parsing

Michael A. Covington
Artificial Intelligence Center
The University of Georgia
Athens, Georgia 30602-7415
www.ai.uga.edu/mc

This copy formatted April 15, 2004

This note pertains to the parser described in my paper:

Covington, Michael A. (2003)
A free-word-order dependency parser in Prolog.
http://www.ai.uga.edu/mc/dparser/dparser.pdf

The important point is that **the parser described there is not adequate for parsing any human language,** and is not presented as such.

It accepts totally free word order (which does not exist in any human language) and allows multiple dependents of any type (e.g., 3 subjects on the same verb).

To parse human language adequately, it needs many capabilities added, among them:

1. Requirements that some dependents precede or follow the head (not both). For example, in Japanese, all the dependents of the verb precede it. Most languages have a strong tendency to either put most dependents before their heads (Japanese), or put most dependents after their heads (Arabic). Some languages have a mixture of the two preferences (English). And of course when free word order is present, some languages leave this requirement unspecified.

This could be handled by adding a feature to each dependency rule restricting the search for dependents or heads.

2. Requirements that some dependents of the same head occur in a specific order. For example, in English, the direct object follows the indirect object, if both are present. Both are dependents of the verb, and both come after the verb; further, the indirect object must come first.

   This is a rather uncommon situation and is not so easy to handle; it may be necessary for both of the objects to communicate with the verb and with each other through feature unification.

3. Requirements of projectivity ("no crossing branches"). For example, in English and most languages, prepositional phrases are projective; that means they are not broken up. The preposition's dependents, dependents' dependents, etc., form a continuous sequence of words.

   Parsing algorithms to enforce projectivity are discussed by Covington (2001). In a language where some phrases are projective and others are not, there could be a feature on each dependency rule indicating which way to parse the structures under it.

Although many people have implemented dependency parsers in different ways, I do not have "the correct solution" or a ready-to-use algorithm for handling these requirements. I consider them to be interesting research problems.

References

Covington, Michael A. (2001) A fundamental algorithm for dependency parsing. *Proceedings of the 39th Annual ACM Southeast Conference,* ed. John A. Miller and Jeffrey W. Smith, pp. 95-102.