A Treebank Conversion Algorithm for Non- configurational Languages

Ahmad Pouramini[1], Naser Mozayani[2]
[1]Department of Computer Engineering
Iran University of Science and Technology, Tehran, Iran
puraminy@yahoo.com

[2]Department of Computer Engineering
Iran University of Science and Technology, Tehran, Iran
mozayani@iust.ac.ir

Abstract

In this paper we propose an algorithm for converting dependency structures to phrase structures. This algorithm mainly concerns the characteristics of non-configurational languages. We review current works in the field and on the basis of these works we try to adopt a more flexible approach to the problem.

1 Introduction

In this paper we present a method for converting dependency structures to phrase structures, which is adequate to be employed in the case of non-configurational languages. The algorithm is intended to bridge between an existing dependency-based representation for a non-configurational language and a well-studied constituency-based scheme. It can also be employed to automate the annotation process of a Treebank in a way in which the user provides the dependency structure of a sentence and the algorithm automatically builds the desired phrase structure.

In section 2 we investigate the notion of non-configurationality in natural languages, in Section 3 we review current methods for converting dependency structures to phrase structures and in section 4 we see how these methods might apply to non-configurational languages. In general, we try to adopt a more flexible approach to the conversion algorithm.

2 Non-configurationality in Natural Languages

An important distinction can be drawn among languages where the grammatical relations, and in particular terms, can be mostly identified from word order, and languages which employ various syntactic markers such as case and other inflectional features [Bresnan, 1982], for this same purpose. For instance, Persian makes an extensive use of verbal inflectional marking in realizing the relation between a subject and its finite Verb (agreement). By contrast, in English the position of a word in the linear order of the sentence usually is function of its role, e.g. the position of an NP determines if it is the subject or the object of the Verb. This difference in the encoding of grammatical relations in the syntactic structure by mainly using word order or other devices, in literature has been referred to *configurationality* and *non-configurationality*, which correspond to a distinction between fixed word order languages (such as English) and free word order languages (such as Czech).

In configurational languages grammatical relations could be denoted in terms of structural configurations of constituents. In fact, by using a notion of dominance (i.e. hierarchical top-bottom ordering) and precedence (i.e. linear left-to-right ordering) on nodes of a tree, we can provide a configurational definition of a structure, i.e. a definition based on the relative position occupied by different nodes in the tree [Radford, 1997]. For instance, the definition of the grammatical relation subject as external argument is a typical configuration-based definition in use by constituency-based formalisms. It follows that in configurational languages a reduction of relational structure to constituent structure (as in Chomsky's approach) is possible.

However a sharp distinction between these two kinds of languages cannot be drawn because non-configurationality (i.e. the fact that grammatical functions are not entirely recoverable from the phrasal order) is present in all languages to some extent. For instance, in the example of Figure 1 from the Penn Treebank [Marcus et al., 1993], the Adverbial Phrase "There" is dislocated to the left rather than to the right side of the Verb, as usual in English.

```
(S (ADVP-TCP-1 There)
    ,
    (NP-SBJ I)
    (VP put
        (NP the book)
        (ADVP *T*-1)))
```

Figure 1: The sentence "There, I put the book".

The sentence structure may depend on pragmatic factors too, and the speaker/writer's communicative purpose can determine the organization of information within a sentence regardless of the standard word order.

As another example consider the following German sentence:

(1) Der Lehrer gab gestern den Kindern die Bücher
The teacher (NOM) gave yesterday the children (DAT) the books (ACC)
"Yesterday, the teacher gave the children the books"

In this example, for the arguments of the verb all 4! = 24 possible orderings are correct sentences. See (2) for some examples:

(2) Gestern gab der Lehrer die Bücher den Kindern.
Den Kindern gab die Bücher der Lehrer gestern.
Die Bücher gab gestern den Kindern der Lehrer.
Der Lehrer gab den Kindern gestern die Bücher.

This unconstraint word order freedom of arguments of the verb is only the general rule, things change e.g. when pronouns are used. Consider the sentence:

(3) Ihr gab es er.
her (DAT) gave it (ACC) he (NOM)
"He gave it to her"

This sentence is only acceptable with a strong emphasis on "ihr" and on "er". The free ordering of arguments within one clause is also called clause-internal scrambling. Yet, in some languages (such as German, Korean Hindi and Persian) a constituent of an embedded clause may be moved from that clause into the matrix clause. This property is called long distance scrambling in the literature. In the matrix clause the scrambled embedded constituents may occupy any position. Furthermore, constituents may be moved out of more than one nested embedded clause. An example of multiple movements out of one embedded clause is given in sentences (4a) and (4b) below.

(4a) …daß der Detektivi dem Klienten [PRO$_i$ den Verd_achtigen des Verbrechens zu überführen] versprochen hat
…that the detective (nom) the client (dat) the suspect (acc) the crime (gen) to indict promised has

(4b) …daß [des Verbrechens]$_k$ der Detektiv$_i$ [den Verdächtigen]$_j$ dem Klienten [PRO$_i$ t$_j$ t$_k$ zu überführen] versprochen hat.
…that the crime gen the detective nom the suspect acc the client dat to indict promised has
"…that the detective has promised the client to indict the suspect of the crime

The relative independence of the dependency structures from the word order determines the suitability of dependency-based approaches in the representation of non-configurational languages.

On the other hand, in no case, the non-configurationality actually means that the words of one sentence can come in any order, some constraints always apply. Even in the language most commonly cited as an example of free word order namely the Australian Aboriginal language Warlpiri some non-syntactical constraints regarding focus apply see [Becker, 1994]. In other languages such as German only some of the words may appear in any order, other words have fixed positions and as in Warlpiri, topic/focus information plays a key role in determining and further constraining the word order. For example in Sentence 1, the order is free only between the noun-phrases "der Lehrer", "den Kindern", "die Bücher" and the adverbial phrase "gestern" while the finite verb "gab" always appears at the second position. The word order is also fixed within one noun-phrase e.g. "Lehrer der" is unacceptable. This is especially true in the case of languages in which the word order cannot be altered within a constituent while there is fairly free order among constituents at the surface e.g. Persian, Spanish, Italian. At this stage, constituent annotation is convenient as a previous step for the annotation of syntactic functions.

Note due to the frequency of discontinuous constituents in non-configurational languages, the filler-trace mechanism which is usually used in the pure phrase structure representation, would be used very often, yielding syntactic trees fairly different from the underlying predicate-argument structures. Furthermore, the structural handling of free word order means stating well-formedness constraints on structures involving many trace-filler dependencies, which has proved tedious. Since most methods of handling discontinuous constituents make the formalism more powerful, the efficiency of processing deteriorates, too [Wojciech et al, 1997].

Currently, there are some Treebanks, such as NEGRA corpus for German, which employ a hybrid framework to combine advantages of both dependency-based representation and phrase structure representation. In NEGRA corpus, the syntactic structure is represented by a tree. The branches of a tree may cross, allowing the encoding of local and non-local dependencies and eliminating the need for traces. In this structure the syntactic categories (S, VP, NP, etc.) are encoded as node labels and the grammatical functions (such as HD ("head"), SB ("subject"), etc.) as edge labels. This approach has considerable advantages for free-word order languages such

as German, which show a large variety of discontinuous constituency types [Wojciech et al, 1997].

On the basis of these considerations, in the next section after reviewing the current methods for converting dependency structures to phrase structures, we investigate how these methods might be modified in order to be applicable in the case of non-configurational languages.

3 Converting Dependency Structures to Phrase Structures

The main information that is present in a phrase structure but not in a dependency structure is the type of the multiple word syntactic units (e.g. NP, VP and S); In fact, in a dependency structures each node corresponds to a single word, and each arc corresponds to a grammatical relation while in a phrase structure each terminal node corresponds to a word, but several non-terminal nodes are included in order to represent groupings of words (i.e. phrases or constituents).

So far, several algorithms have been presented for converting dependency structures to phrase structures [Collins et al, 1999; Covington, 1994a; Covington, 1994b; Xia and Palmer, 2001]. They usually differ in the shape of the resulting phrase structure e.g. flat, binary (see [Collins et al]) or according to X-bar theory (see [Covington, 1994a; Covington, 1994b]) and in their flexibility to use language-specific information [Xia and Palmer, 2001].

[Xia and Palmer, 2001] explains two main approaches utilized in these algorithms and present a more general algorithm which subsumes these previous ones; this algorithm by using simple heuristic rules and taking as input certain kinds of language-specific information such as the types of arguments and modifiers that a head can take, produces phrase structures that are very close to the ones in an annotated phrase-structure treebanks. However, it originally has been designed to build constituency trees close to the trees of Penn treebank for English [Marcus et al., 1993]. Here, we intend to alter this algorithm in order to make it suitable for the case of non-configurational languages. With this aim in view, in the following we review it in more detail.

3.1 Algorithm 1

This algorithm distinguishes two types of dependents: arguments and modifiers. It also makes use of language-specific information in the form of three tables: the projection table, the argument table, and the modification table. The projection table specifies the projections for each category. The argument table (the modification table, resp.) lists the types of arguments (modifiers, resp.) that a head can take and their positions with respect to the head. For example, the entry V → VP → S in the projection table says that a verb can project to a verb phrase, which in turn projects to a sentence; the entry (P: 0 1 NP/S) in the argument table indicates that a preposition can take an argument that is either an NP or an S, and the argument is to the right of the preposition; the entry (NP: DT/JJ PP/S) in the modification table says that an NP can be modified by a determiner and/or an adjective from the left, and by a preposition phrase or a sentence from the right. Given these tables, it uses the following heuristic rules in order to carefully represent the Penn format:

One projection chain per category: Each category has a unique projection chain, as specified in the projection table.

Minimal projection for dependents: A category projects to a higher level only when necessary.

Lowest attachment position: The projection of a dependent attaches to a projection of its head as lowly as possible

3.2 Algorithm 2

Considering the characteristics of non-configurational languages several difficulties can be expected arising from applying Algorithm 1 to this type of languages. In the following we propose some modifications which can be made to this algorithm, in order to account for these difficulties:

- In the argument table, an argument of a head can be given without being restricted to a fixed position with respect to the head. For example an entry in the argument table can be like this: (V: 0 1 (2) <>, S, <NP, PP>); this entry shows that a verb can take three arguments namely S, NP and PP. None of them is obligatory to the left of the verb; Argument S is only to the right of the verb and two arguments NP and PP can be either to the right or to the left of the verb. We made this change because in non-configurational languages we can not suppose a fixed order for the positions of constituents in a sentence. For example in Persian, the subject and object can be permuted without changing the gross meaning of the sentence, therefore providing the information of Algorithm 1 which requires that the position of all the dependents with respect to the head be specified, faces problem.

- Similarly, an entry in the modification table may include the same modifiers on the right and on the left of a head. For example an entry in the modification table can be like this (V: PP/ADVP PP/ADVP). this indicates that the verb can be modified by a prepositional phrase or an adverbial from the both sides. The reason for this selection is the same as above.

- A phrasal category in the argument or modification table can be distinguished according to its grammatical function in the sentence. For example an entry in the argument table may be given like this: (S: 1 0 (1) <NP(subj)>,<>,< NP(obj)). This entry shows that S can take two arguments namely NP(subj) and NP(obj). NP(subj) is only to the left of the verb phrase and NP(obj) can be either to the left or to the right of the verb phrase. Here, the *subj* and *obj* point to the grammatical functions of the two NPs respectively. Note if we don't use such information, we can not distinguish the NP which can appear on the either side

of the verb phrase from the NP which can only appear on the right side of the verb phrase. This clearly has application to the word order variations which target just some specific constituents. For example Persian exhibits the following unmarked word order in a double object construction:

a) S $O_{specific}$ PP V

b) S PP $O_{nonspecific}$ V

The specific direct object appears in a higher position, preceding the indirect object. The nonspecific object is adjacent to the verb, following the indirect object. In addition to the different structural position of these two types of objects, the specific direct object can appear on the both sides of VP while the nonspecific object just can precede the verb (e.g. *man* (I) *xandam* (read) *ketab ra* (the book), but not *man xandam yek ketab* (a book)). This is a property seen in many other languages such as Hindi, Turkish, German, and Dutch [Karimi, 2003]. In general, non-configurational languages give lot of stress on morphology and case markers. So, the only information which can best distinguish different types of constituents from each other is their feature and morphology information. This extra information about constituents can help us to better determine their positions in the phrase structure.

Applying above modifications to the tables of Algorithm 1, some modifications to the procedure through which the algorithm builds the phrase structures, are expected:

- In the attachment of an argument to the head projection chain, in addition to the entry which is corresponded to the relative position of the argument with respect to the head (left or right), the list of the arguments which can appear on either side of the head must be checked.

- After building the constituency subtree for each dependent, its corresponding grammatical information in the dependency tree must be assigned to the root of the subtree. This information will be later used for matching the subtree with an argument or modifier in the argument or modification table.

- In the representation of the resulting phrase structure, the order of the phrases is determined by their actual positions in the sentence, in this way crossing edges may be employed, thereby covering the possible long distance scrambling.

As can be observed from the modifications mentioned above, although the two algorithms adopt different heuristic rules to build phrase structures, the first algorithm is a special case of the last algorithm; In fact, Algorithm 2 offers some extra options which can easily be omitted.

4 Conclusion

In this paper we have proposed some modifications which can be applied to the current algorithms for converting dependency structures to phrase structures, in order to make them suitable for the case of non-configurational languages. We first selected the most general one of these algorithms, and tried to adopt it for our purpose. With this aim in view, we have proposed some modifications in this algorithm, which concern the word order variations in such languages. In this way, we may use extra information to better distinguish constituents and offer more options for their positions in the sentence.

References

[Becker, 1994] Tilman Becker, *HyTag: A new type of TAGs for hybrid syntactic representation of free word order languages*. Phd thesis, University of Saarlandes. 1994.

[Bresnan, 1982] J. Bresnan, ed., *The mental representation of grammatical relations*. MIT Press, Cambridge. pages 282–390, 1982

[Collins et al, 1999] M. Collins, J. Hajiˇc, L. Ramshaw and C. Tillmann, A Statistical Parser for Czech. In *Proc. of ACL-1999*, pages505–512, 1999.

[Covington, 1994a] M. Covington, An Empirically Motivated Reinterpretation of Dependency Grammar, 1994. Research Report AI-1994-01.

[Covington, 1994b] M Covington, GB Theory as Dependency Grammar, 1994. Research Report AI-1992-03.

[Karimi, 2003] Simin Karimi, On object positions, specificity, and scrambling in Persian. In Simin Karimi (ed.) *Word Order and Scrambling*. Oxford: Blackwell Publishing, pp 91-124., 2003.

[Marcus et al., 1993] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19:313–330, 1993.

[Radford, 1997] A. Radford, *Syntactic theory and the structure of English. A minimalist approach.* Cambridge University Press, Cambridge, 1997.

[Wojciech et al, 1997] Skut Wojciech, Krenn Brigitte, Brants Thorsten and Uszkoreit Hans, An annotation scheme for free word order languages, In *Proc. of the Fifth Conference on Applied Natural Language Processing (ANLP 97)*, Washington, DC, 1997.

[Xia and Palmer, 2001] Fei Xia and Martha Palmer, Converting dependency structures to phrase structures. In *Proc. of the Human Language Technology Conference (HLT-2001)*, San Diego CA., 2001.