An Interaction Grammar for English Verbs

Shohreh Tabatabayi Seifi

INRIA / Université de Lorraine, BP 239
54506 Vandoeuvre-les-Nancy cedex

Abstract. This paper accounts for the construction of a grammar for English verbs using Interaction Grammars. Interaction Grammar is a grammatical formalism based on the two key notions: polarities and constraint system. A polarity expresses an available resource or a lack of resource and is used to discriminate between saturated and unsaturated syntactic structures. A grammar is viewed as a system of constraints of different kinds: structural, feature and polarity constraints, which should be satisfied by the parse trees of sentences. We have developed a grammar for English verbs in affirmative clauses and finally we evaluated our grammar on the portion of a test suite of sentences, the English TSNLP, with LEOPAR parser which is a parser devoted to Interaction Grammars.

Keywords: Grammatical formalism, interaction grammar, tree description, polarity, unification grammar

1 Introduction

The goal of this work is to construct an Interaction Grammar for English verbs. Interaction Grammar(IG) [1] is a very powerful formalism which has the advantages of both Lexical Functional Grammar (LFG) [5] and Categorial Grammar (CG) [3] and [4]. From LFG, it takes the flexibility with the mechanism of unification to perform syntactic composition. From CG, it takes the resource sensitivity to control syntactic composition with polarities. It is inspired by chemical reactions. In this formalism each lexicalized elementary tree acts as a potentially active element which can participate in a reaction with another tree. Two nodes of different trees can merge if they are able to neutralise each other's active polarities and make stable combination. Whenever all the polarities are neutralised correctly to build a unique tree, the obtained tree would be the parse tree of the sentence.

Guillaume and Perrier [9] investigated the principle underlying IG formalism from more theoretical point of view. In general we can see that in IG polarities are attached to the features where as in CG they are attached to the constituents themselves. The other more essential difference lies in the frameworks of these two formalism: CG are usually formalized in generative deductive framework while IG is formalized in a model–theoretic framework. Pullum and Scholz highlighted the advantages of changing the framework [8].

Currently a very detailed French Grammar with quite high coverage has been developed in IG formalism [9]. This work is an attempt to construct a portion of English grammar using the same formalism.

2 Interaction Grammars

In order to get a view of how Interaction Grammars work, the following notions are required.

2.1 Tree Description

A tree description is a tree like structure which uses the notion of underspecification relations to describe a family of trees instead of only one tree [7]. This allows to make an unlimited number of trees from a unique underspecified tree representation. A tree which does not have any underspecified relation is a model. For instance, one tree description with an underspecified dominance link, illustrated in the left side of Fig. 1, can produce three (and more) different models.

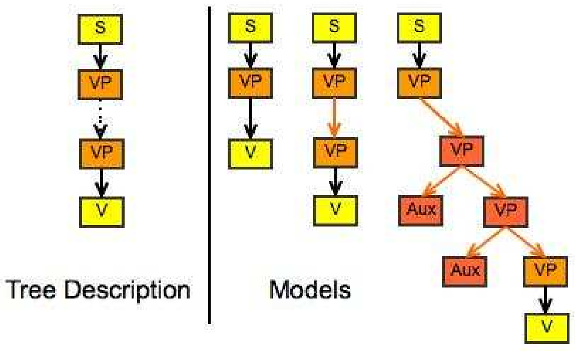

Fig. 1. A description tree with three of its models.

2.2 Feature Structure

IG associates features with the nodes of tree descriptions to put constraints and prevent building ungrammatical parse trees. Feature structure models the internal structure of the language where grammatical properties are represented. Not only do feature structures prevent building ungrammatical sentences, but also they carry valuable linguistics information. Using feature structure to construct a grammar is a salient characteristics of a formalism like LFG but unlike LFG there is no recursive feature in IG and feature structures have only one level.

2.3 Polarities and Saturated Models

Polarities are one of the basic notions in Interaction Grammars. A polarized feature is used to control the process of merging nodes of trees to be combined. Different kinds of polarities are used in IG. There are features with positive or negative polarities meaning an available resource to be consumed or an expected resource to be provided respectively. A neutral feature indicates a feature which is not to be consumed but just to participate in a simple unification process while combining with other nodes. A saturated feature is the result of combination of a positive and a negative feature and can be unified with no more positive or negative feature.

Finally there are virtual features which should be merged with real features during the parsing process including positive, negative, neutral and saturated features. Virtual polarities are used to express required contexts. Positive, negative and virtual features constitute the active polarities because they need to combine with other polarities.
In Fig. 2 features with positive (–>), negative (<–) and virtual (∼) polarities can be seen in the tree descriptions associated with the words ***been*** and ***arranged***. Saturated features indicated with symbol <==> and neutral features indicated with symbol = or == can be seen in the non-completed parse tree in the left side of Fig. 2. The horizontal arrows between nodes indicate linear precedence constraints and comes in two forms: large precedence (- - >)which means there can be several nodes between these two node and immediate precedence (–>) which obstacles locating any other node between these two nodes.

Tree descriptions along with polarized feature structure are called polarized tree descriptions or PTDs. The composition process is defined as a sequence of PTD superpositions controlled by polarities. When two nodes merge, their features will be unified according to the standard process of unification with the extra limitations coming from the rules of combination for polarities. Saturated trees are those completely specified trees which have no active polarities anymore. The goal of parsing is to generate all saturated trees from the set of input PTDs which come from a particular IG grammar.

2.4 Grammar

An IG grammar is defined by a finite set of PTDs which are called the elementary PTDs or EPTDs of the grammar. Each EPTD has one (or more) leaf which is linked with a word of a lexicon and is called an anchored node. This means that the grammar is strictly lexicalized and there is no EPTD without any anchor. In practice there is more than one EPTD for each lexical entry in the grammar with respect to its different syntactic properties.
The parser of an IG grammar has two main roles. First to select the EPTDs of the words in the input sentence from the set of EPTDs inside the grammar then to build all valid minimal models with the use of these EPTDs.

Fig. 2 and Fig. 3 show an example of building a model for the sentence *The interview* ***has been arranged***. The first figure is a snapshot of the intermediate results during parsing process; a non-completed parse tree with unsaturated nodes along with the EPTDs of the rest of the words of the sentence. The second figure shows the result yielded by combining the EPTDs of the remaining words with the non-completed tree: it is one valid model, the parse tree of the input sentence.

3 Building grammars with wide coverage

Tackling with real language problems needs a large scale lexicalized grammar which is hard to build and without using automatic tools it is an overwhelming task. The main reason for that is the huge degree of grammatical and lexical redundancy. For instance in IG, several PTDs may share same subtrees which is due to the grammatical redundancy of syntactic structures. Moreover different lexical entries share the same elementary PTDs owing to the fact that they are in the same syntactic category. These

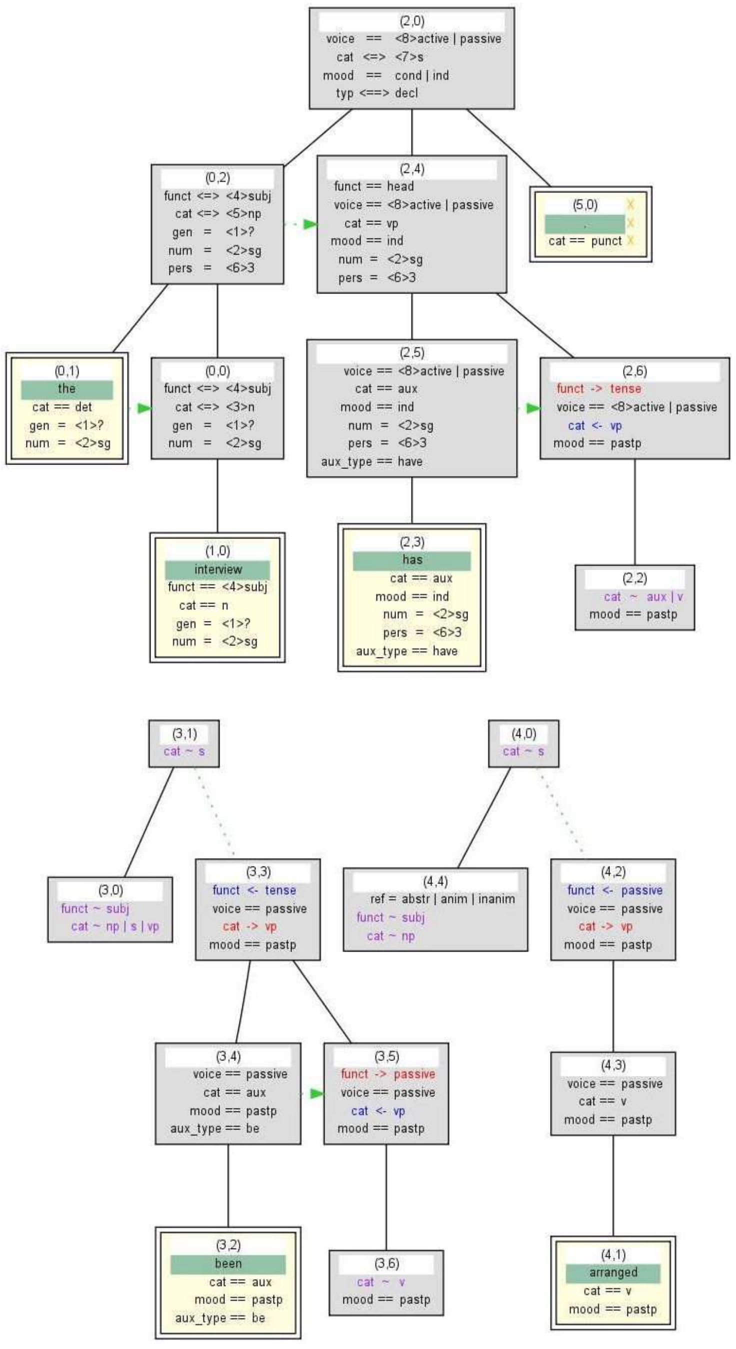

Fig. 2. A non-complete parse tree on the top with two of its EPTDs on the bottom waiting to be used to make a saturated model for the sentence *The interview* ***has been arranged***.

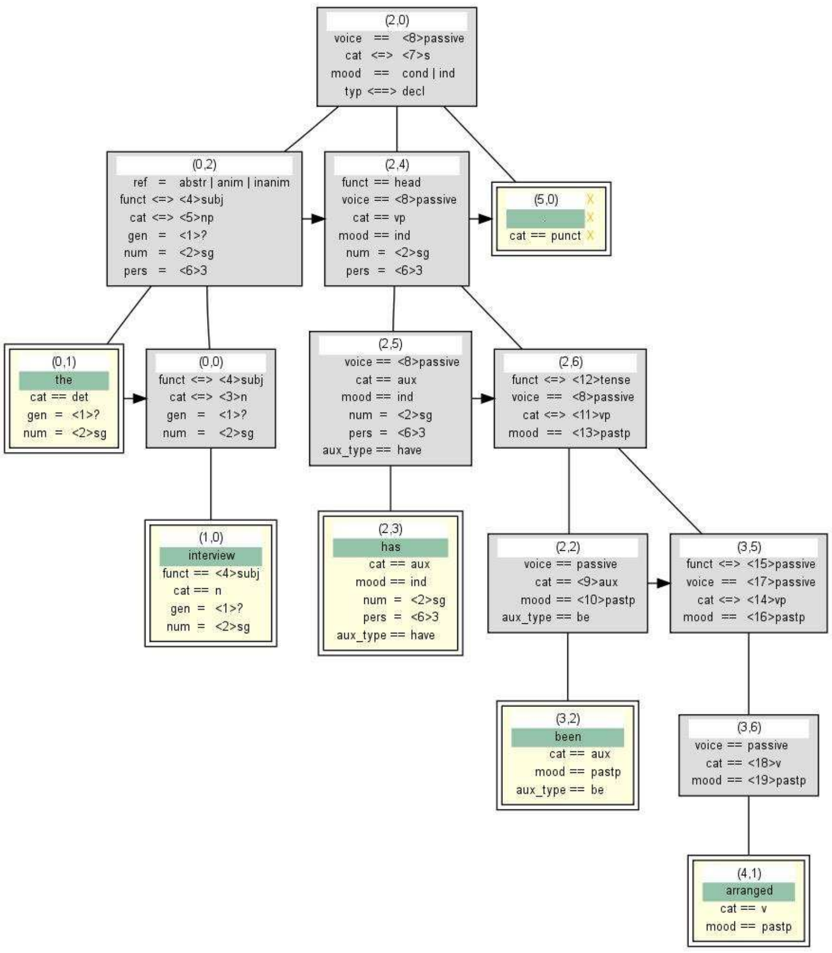

Fig. 3. A parse tree (model) for the sentence *The interview* ***has been arranged***.

redundancies turn even a small modification of the grammar into a big change in large amount of grammar trees. In order to conquer these obstacles eXtensible MetaGrammar (XMG) [6] is used as a facilitating tool to write the IG grammar. XMG is a tool for constructing large scale grammars. The main feature of XMG is to distinguish between source grammar and object grammar. A source grammar is a collection of statements written in a human readable language which produces object grammars which are usable by NLP systems.

3.1 Source grammar and object grammar

The terms source grammar and object grammar here are analogous with the source and object codes in programming languages. In the current task first we wrote the source grammar and then we compiled it with XMG into the object grammar which is encoded in XML. XMG is widely used in construction of grammars in two formalisms: IG and TAG [2].
In source grammar, each small fragment of tree structure is written in an individual class.

By use of class disjunction, conjunction and inheritance, more complex classes can be built. The compilation of terminal classes of the source grammar produces the EPTDs of the object grammar. Each EPTD does not contain any lexical entry yet, but has a detailed description of the word which is going to be nested in its anchor node.

3.2 Linking the grammar with the lexicon

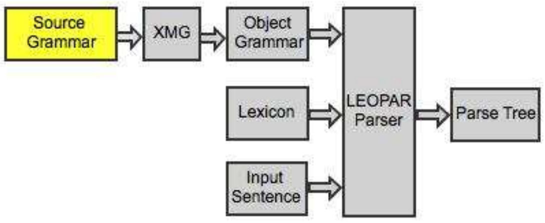

Fig. 4. The process tool chain

The characteristics of the appropriate word to be settled in an anchor node of an EPTD is described in the EPTD interface. An interface is a feature structure describing the words which are able to anchor the EPTD. Every entry in the lexicon is a pair of a word and a feature structure hence a word can anchor an EPTD if its feature structure is unifiable with the EPTD's interface in the exact same means of unification we have in node merging.

The whole process chain is illustrated in Fig. 4 and the highlighted box is the part that has been developed in our work.

4 The Grammar of English Verbs

To build a complete Interaction Grammar for a specific language which can be able to parse any possible sentence, we need a set of EPTDs for wide range of words from different categories like verb, auxiliary, noun, adjective, adverb, prepositions, determiner, pronoun etc. Our main effort was on the construction of a grammar for verbs in affirmative clauses (e.g. different tenses, moods and voices). We have used a small grammar for noun phrase, pronouns, prepositions and adjectives to provide appropriate EPTDs to parse a sentence [11].
The central focus in writing a grammar with IG formalism is to find a way to write classes in a manner that with the use of heredity, conjunction and disjunction all different grammar trees needed in a complete grammar can be obtainable.

4.1 An Sketch of The Major Modules

Five major modules contribute building the EPTDs for verbs. Module **VerbalKernel** is the kernel of all lexical and auxiliary verbs. It implements different possible voices and moods of the verbs and also it provides syntactic functions of a verb phrase (e.g. head, subject, modifier and etc). A subclass in a module is an intermediate tree description that we use along with operators such as heredity, conjunction and disjunction to build the final tree descriptions.

There are seventeen subclasses in module **VerbalKernel**. For instance we model the long distance dependency between a verb and its subject with the use of underspecification relation. All tree descriptions of the verbs which are coming out of this module have a subject node which is an empty node for imperative verbs and a non empty node for all other verbs.

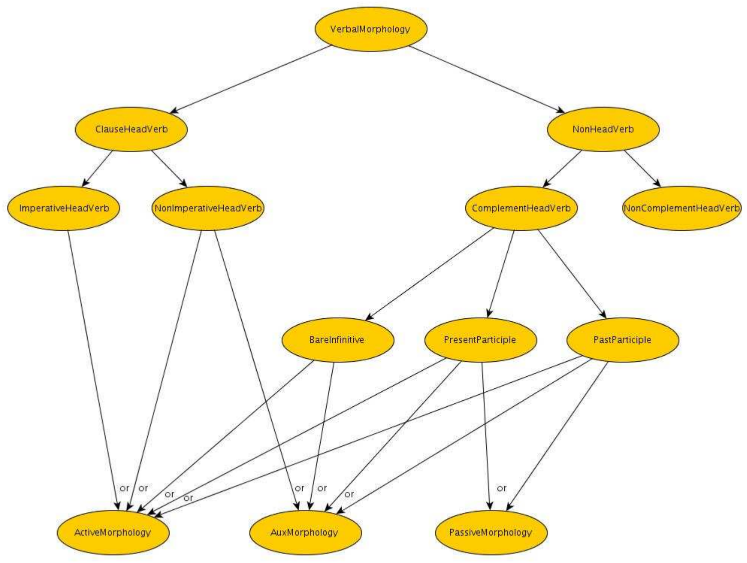

Fig. 5. A schema of relations between some of classes in the module **VerbalKernel**.

Fig. 5 aims to illustrate different relations between some of classes in the module **VerbalKernel**. Some classes are inherited from another one and some are built out of the disjunction of two or more other classes.

Whenever there is no complement nodes in the verb tree, auxiliaries are following the same pattern as non auxiliary verbs therefore class **Auxiliary** is inherited from **VerbalKernel**. The EPTD associated with the auxiliary ***been*** in Fig. 2 is an example of an auxiliary EPTD. Eight subclasses are in class auxiliary implementing all different kinds of auxiliaries. We treated **to** ,the infinite marker, like auxiliaries in the current grammar. The main reason for this approach is the complementary distribution between ***to*** and modal auxiliaries e.g. they never appear together but they appear in the exact same position in the sentence and both force the verb after them to be in the bare infinitive form.

Module **Complements** builds all different kinds of verb complements. It contains 20 different classes and all number, category and function of verb complements are defined in this module.

VerbalKernel and **Complements** will merge together in order to make a new module **Diathesis** with 20 classes which describes different diathesis along with different verb complements e.g. 13 classes for active verbs and 7 classes for passive verbs. For instance to build an EPTD for an active verb with a direct object a tree describing a predicate node with one complement of type noun phrase will merge to a tree associated with active verb and give back the appropriate PTD with active verbs with direct nominal object. Finally module **Verb** with 15 classes will implement trees for different verb families according to their subcategorization frame. For example the family of transitive verbs can anchor the tree associated with active verbs with direct object and the tree associated with passive verbs. So the class for transitive verbs will be the disjunction of two classes of module **Diathesis**. The PTDs which are coming out of module **Verb** and module **Auxiliary** are elementary PTDs and used to attach to the lexicon.

5 Evaluation Results

The English TSNLP (Test Suite for Natural Language Processing) [12] project has aimed to compose a tool for Natural Language Evaluation purposes. It is a database of phrases which carry different linguistic phenomena to be used as a reference for evaluation tasks. These phrases are divided into several categories according to different linguistic phenomena. Each category has both grammatical and ungrammatical examples to provide an infrastructure not only for determining the true positives (success in parsing the grammatical sentences), but also for counting the true negative (failure to parse ungrammatical sentences).

The English TSNLP has 15 different categories and 3 of those contain phenomena exclusively related to verbs which was the subject of this research. However those three categories also contain some other structure which was not included in the current grammar and sentences of those form were put out as well including phrasal verbs, sentences having *there* as their subject and sentences containing relative clauses. (e.g. *That she leave is requested by him.*) We have used LEOPAR parser [10] to construct the models for the input sentences and the result of evaluation of the current grammar can bee seen in table 1.

	Grammatical	Ungrammatical
Total Number	148	832
Parsing Success	115	50
Parsing Failure	33	782
Precision	86.7%	94%

Table 1. Evaluation results of the proposed IG grammar on portion of English TSNLP

5.1 Result Analysis

The major reason of failure in parsing grammatical sentences is that the construction of verbs up to now requires that every verb tree has a subject node which should be filled with a real subject while parsing the sentence. However there are situations in which a verb acts like a noun or an adjective and there is no such a subject node in its grammar tree. (e.g. *He finds the office **closed**.*) Owing to the fact that we were focusing on construction of the verbs of the main VP of the sentence, other grammatical phenomena which are related to verbs were not properly treated and this failure should not be regarded as a weakness of the framework. In appropriate time all this structures can be constructed in the grammar which is one of the goals in the future works.

The other failure is to mistakenly parse some ungrammatical sentences. One of the main reasons, among other defeats, is that there are some sentences that are not correct because of semantic issues which is not recognizable by our grammar.

6 Conclusions and Future Works

Interaction grammars is a powerful formalism that has advantages of both unification grammars and categorial grammars at the same time. Writing a wide coverage grammar for the English language needs a huge effort and this work can be regarded as the starting point of such a project. Using tools like XMG to accomplish this goal is quite helpful and makes the writing and then the tuning of the grammar a lot more easier than before. Moreover, a high degree of factorization is possible when we separate source grammar and object grammar which leads to more efficiency.

The potential future aims of this project are first to continue to construct a complete grammar for English incorporating all different phenomena in order to parse any grammatical English sentence and second try to cope with the still open problems like coordination in the sentences within the same framework.

On the other hand English TSNLP is a relatively simple set of sentences and is not quite similar to real corpora. Therefore some steps further would be to enrich the grammar some how we would be able to cope with real corpora like newspaper or spoken language corpora where the structure of sentences are more complicated and our grammar should be able to handle several non quite grammatical sentences too .

References

1. Perrier, G.: Interaction grammars. In 18th International Conference on Computational Linguistics, CoLing 2000, Sarrebrucken, pages 600–606. (2000)
2. Crabbé, B.: Grammatical development with XMG. LACL 05. (2005)

3. Retoré, C.: The Logic of Categorial Grammars. ESSLLI 2000, Birmingham.(2000)
4. Steedman, M.: Categorial Grammars. A short encyclopedia entry for MIT Encyclopedia of Cognitive Science. (1999)
5. Bresnan, J.: Lexical-Functional Syntax. ESSLLI 2000, Birmingham.(2000)
6. Duchier, D., Le Roux, J., Parmentier,Y.: The Metagrammar Compiler: an NLP Application with a Multi-paradigm Architecture. In Second International Mozart/Oz Conference, MOZ 2004, Charleroi, Belgium, pages 175–187 (2004)
7. Marcus, M.C., Hindle, D., Fleck. M.M. : D-Theory: Talking about Talking about Trees. In 21st Annual Meeting of the Association for Computational Linguistics, pages 129–136.(1983)
8. Pullum,G.K., Scholz, B.C. : On the Distinction between Model–Theoretic and Generative–Enumerative Syntactic Frameworks. In Logical Aspects of Computational Linguistics, LACL 2001, Le Croisic, France, volume 2099 of Lecture Notes in Computer Science, pages 17–43. Springer Verlag. (2001)
9. Guillaume,B., Perrier, G.: Interaction Grammars. INRIA Research Report 6621: `http://hal.inria.fr/inria-00288376/` (2008)
10. Guillaume,B., Le Roux,J., Marchand, J., Perrier,G., Fort,K., Planul,J.:A Tool–chain for Grammarians. CoLING 08, Manchester. (2008)
11. Planul, J.: Construction d'une Grammaire d'Interaction pour l'anglais. Mémoire, présenté et soutenu publiquement le 27 juin. (2008)
12. Lehmann,S., Oepen, S., Regnier–Prost,S., Netter,K., Lux,V., Klein,J., Falkedal,K., Fouvry,F., Estival, D., Dauphin, E., Compagnion, H., Baur, J., Balkan, L., Arnold,D.: TSNLP– Test Suite for Natural language Processing. In Proceedings of COLING 1996, Kopenhagen. (1996)