Survey: Semantic Role Labeling and Dependency Parsing

by

Janardhan Singh
(Roll No. 10305067)

Under the guidance of
Prof. Pushpak Bhattacharyya

Abstract

Semantics is a field of Natural Language Processing concerned with extracting meaning from a sentence. *Semantic Role Labeling* takes the initial steps in extracting meaning from text by giving generic *labels* or *roles* to the words of the text. The meaning of this small set of labels can be assumed to be understood by the machine. To help semantic extraction the relationship between the words in a text needs to be understood at the syntactic level. Dependency Grammar and Parsing give binary relationships between the words of a text giving clues to their semantic relations. This document attempts to give a brief survey on these two important fields concerned with Semantic extraction from text.

Table of Contents

Chapter 1

Semantic Role Labeling

Semantic Role Labeling is the task of assigning semantic roles to the constituents of the sentence. Although, many rule-based techniques like Link Parser, MiniPar and Lexical Functional Grammar have been traditionally used for this task, success has also been achieved by statistical techniques. One of the foundational works on this ground was done by Jurafsky and Gildea(2002) [DD02]. This work uses lexical resources like *WordNet* and *Framenet* . The section starts by describing semantic roles. It then discusses about the different lexical resources that can be used for Semantic Role Labeling. It then describes the different *Semantic Role Labeling* techniques.

1.1 Semantic Roles

In linguistic theory, *semantic roles* are one of the oldest classes of constructs. The Paninian *karaka theory* is probably one of the oldest works in this field[DD02]. A lot of variety in semantic roles exist today. The semantic roles could be domain-specific or generic. Fillmore[BFL98] gives a hierarchical classification of semantic roles. The Framenet project was based on these Framenet roles given by Fillmore. We will look at Framenet later in the chapter. Let's look at some examples of semantic roles.

Consider an example from the *Cognition* domain in figure 1.1

Here, the semantic roles *Judge*, *Evaluee* and *Reason* are specific to the cognition domain when a judgment is made. In the Framenet hierarchy, these roles fall in the *Judgment* frame.

For an example with more generic roles, consider the sentence in figure 1.2.

1.2 Lexical Resources

This section looks at some of the useful lexical resources used for Semantic Role Labeling.

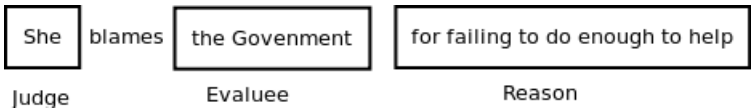

Figure 1.1: Semantic Role Labeling example

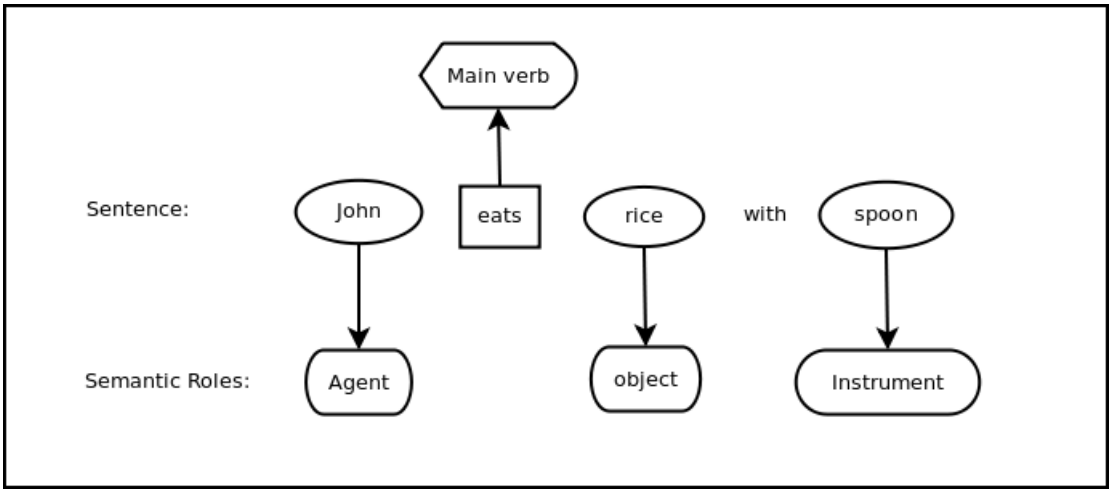

Figure 1.2: Semantic Role Labeling example

1.2.1 Framenet

Framenet currently has about 170,000 manually annotated sentences providing a unique training dataset for semantic role labeling [fra11]. In Framenet dataset, the sentences are arranged in a hierarchical order with each frame referring to a concept. Frames at the higher level refer to a more generic concept while frames at the lower level refer to more specific concepts. Figure 1.3 [DD02] gives the structure of frames in the Framenet.

As shown in figure 1.3, every frame has invoking *predicates* attached to it. These are the verbs and some nouns of English that invoke the concept, referred to, by the frame they are attached to. Sentences that have these *predicates* would have constructs that play the role given by the *frame elements* of the invoked *frame*. For example, in figure 1.1 the predicate *blame* invokes the *Judgment frame* and other constructs in the sentence play the invoked semantic roles. In that example:

(She) plays the role *(Judge)*,
(the Government) plays the role *(Evaluee)*,
(for failing to do enough to help) plays the role *(Reason)*,

1.2.2 Propbank

Propbank[KP03] is another important lexical resource for Semantic Role Labeling. Propbank is a *proposition bank* in which sentences are annotated with *verbal propositions* and their *arguments*. It was proposed by Martha Palmer et. al. It is similar to Framenet but differs in two major ways[Dut11]:

1. All the verbs in the corpus are annotated.
2. All arguments to a verb must be syntactic constituents. A standard set of argument labels have been defined for this purpose.

Verbs are annotated with coarse grained senses and with inflectional information. Inflection describes how does the form of the verb modify with change in tense, person, aspect, mood,

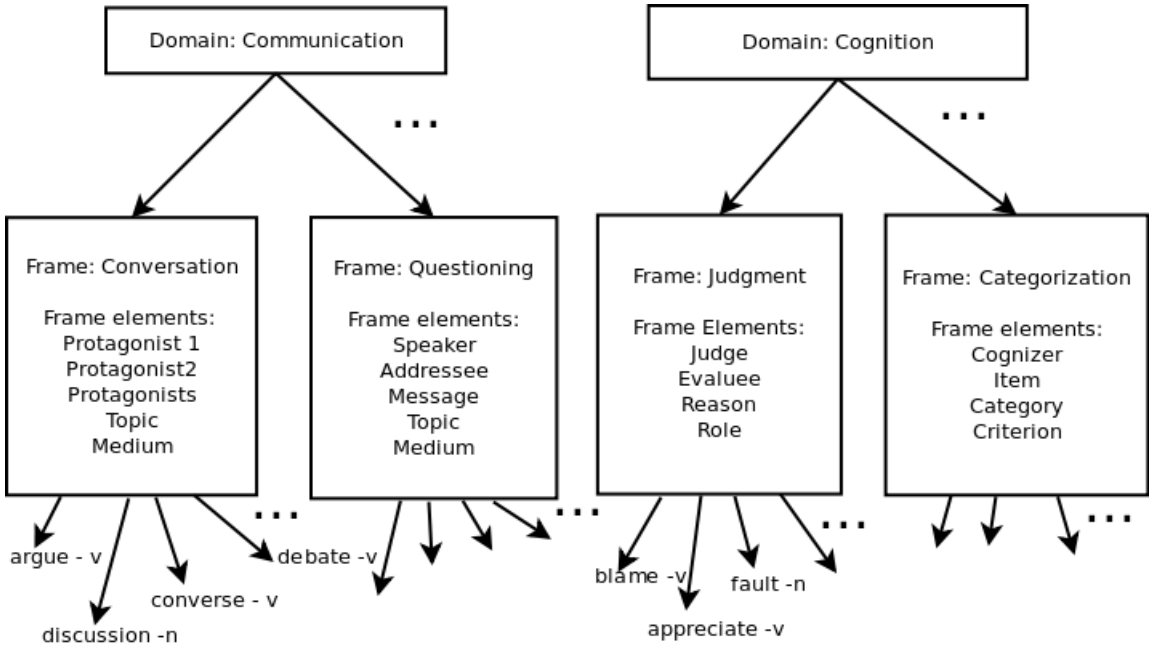

Figure 1.3: Sample domains and Frames in Framenet

number and other grammatical categories [inf11]. The arguments to a verb include:

1. Core argument labels from **ARG0** to **ARG5**. Each of them have a specific meaning like that of the *karakas* in Paninian karaka theory.
2. All arguments to a verb must be syntactic constituents. A standard set of argument labels have been defined for this purpose. eg **ARGM-ADV**: General purpose modifier label.

1.2.3 Verbnet

Verbnet[ver11] is a hierarchical domain-independent, broad-coverage verb lexicon with mappings to other lexical resources such as Wordnet, Xtag, Framenet and Propbank. Verbnet is organized into verb classes extending Levin (1993) classes through refinement and addition of subclasses to achieve syntactic and semantic coherence among members of a class.

1.2.4 Wordnet

WordNet[Fel98] is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The result is a network of meaningfully related words and concepts.

1.3 Link Parser based on Link Grammar

Link parser uses Link Grammar[ST95] to give semantic roles to words in the form of relation between pair of words. The pair forms a grammatical relation. This grammatical relation

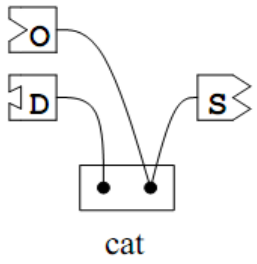

Figure 1.4: The roles ”cat” can play

defines the role of one word with respect to the other. The word whose role is defined modifies the other word, the governing word.

1.4 Link Grammar

Link grammar contains set of words which act as its terminal symbols, a set of relations which define the links between a pair of words and a set of linking requirement which are the properties of the words. The linking requirements of words are stored in a dictionary.

1.4.1 Links and Linking Requirements

Link is the connection or relation between two words. Linking requirements of a word define the roles the word can play. It also defines about the word it can be linked with. Example: *The word ”cat”*

- can be a Subject (S)
- can be an Object (O)
- will have a Determiner (D)

1.4.2 Connectors and Formulae

The symbols S, O and D represent connectors of the words. The connectors define what the word can be and what the word needs. Connectors end with ”+” or ”-” which indicates the direction of its connecting word. ”+” means the word it will be connected to is in the right of it and ”-” means the word is to the left of it. The connectors at the opposite ends must match (+ -). A *linking requirements* is represented by a *formula* of connectors combined by binary operators ”&” and ”or”. Let C and D be two connectors of a word. Then C & D would mean the linking requirement of the word is that both connectors C and D must be connected to their complement connectors. C or D means only one of C and D needs to get connected to a corresponding complementary connector.

Word	Formulae	Disjunctive Form
Sit	S- & (O+ or B+)	((S)(O)), ((S)(B))
A	D+	(()(D))
Dog	{@A-} & (D-) & {B+} & (S+ or O-)	((D) (S)), ((D,O) ()), ((D) (S,B)), ((D,O) (B)), ((A,D) (S)), ((A,D,O) ()), ((A,D) (S,B)), ((A,D,O) (B))
Black	A+	(()(A))

Table 1.1: Disjunctive Form

1.4.3 Disjunctive Form

When formula of each word is represented in disjunctive form it has a set of disjuncts associated with it. A disjunct has two ordered lists of connectors -

1. Left list - Connectors with ”-” mark with it.
2. Right list - Connectors with ”+” mark with it.

A disjunct looks like ((L1,L2,L3,L4,,Ln)(R1,R2,R3,R4,,Rm)). To convert a formula into its disjunctive form we need to find all the valid permutations of connectors. By ”valid” it means that connectors separated by “&“ must be present in the list and among connectors separated by “or“ only one should be present in the list. Some examples are shown in table1.1

1.4.4 Grammar Rules

Link Grammar contains rules which put constraints on the formation of relations among the words of the given sentence. These rules are:

1. **Planarity** - Links between the words, when all of the links are drawn above the words, will not cross.
2. **Connectivity** - All the words should have a connection.
3. **Satisfaction** - The linking requirement of each word is satisfied by the provided links.
   (a) **Ordering**: When the left connectors of a formula are traversed from left to right, the words to which they connect proceed from near to far. When the right connectors of a formula are traversed from left to right, the words to which they connect proceed from far to near.
   (b) **Exclusion**: No two links may connect the same pair of words.

1.5 MiniPar

Minipar is a principle-based English parser. It represents its grammar as a network where the nodes represent grammatical categories and the links represent types of syntactic (dependency) relationships. Minipar uses grammatical principle instead of grammatical rules which act as constraints associated with the nodes and links.[Roy11]

1.5.1 Principle-based Parser

Principle-based grammars[Lin94] use principles like government-binding (GB) theory. Let us consider a sentence in passive voice: The ice-cream was eaten. If a rule-based parser handles this passive voice while finding the object of the action "eat" it would use an IF-THEN rule:

```
If  (Subject be-Verb + ed No Object)
The     make the Subject the Object
```

This is a shallow approach. The basic idea of principle-based parser is to replace this shallow approach with a much deeper and explanatory set of principles. This set of principles is classified into Generators and Filters.

1.5.2 Generating Principles

Generating principles produce possible structures of a given sentence. This class includes:

- **X-bar Theory**: This theory describes the basic shapes of tree allowed in the language. In natural language there are roughly two forms of tree: function-argument form, like a verb begins in a verb phrase and argument-function form where X ends in a XP.
- **Movement Principle**: Movement principle says that any phrase can be moved anywhere. This would create new possibilities, though this might violate other principles. Example: *John likes ice-cream.* This can changed after moving to: Ice-cream, John likes.
- **Binding theory**: Binding theory specifies how pronouns can be bound to their antecedents. Multiple mappings of one pronoun to different antecedents would create new possible structures.
  Example: *John thinks he likes ice-cream.*
  "he" may refer to John or some other person.(two possibilities).
  He thinks John likes ice-cream.
  "He" refers surely some other person except John.(one possibility)

1.5.3 Filtering Principles

Filters remove possible structures which fail some given constraints.

- **Case Filter**: Case theory specifies that every noun phrase should be assigned a case. The subject is given a nominative case and direct objects are given accusative case. If a phrase tree fails to satisfy that the tree is an invalid one.
  Example: *It is likely that John will win.*
  John: nominative case. This is a valid sentence structure.
  It is likely John to win.
  No case for John. This is an invalid sentence structure.
- **Theta Criterion**: Theta theory determines the number of arguments a verb needs and assigns thematic roles to its arguments. This brings in the concepts of transitive and intransitive verbs. Transitive verbs should specify the agent and patient of the action. If any verb requires two objects it must specify the thematic roles of both. One such

example is the verb give which demands what to give and whom to give.
Example: *John put the book on the shelf.*
"put" has two objects book, thing to put, and shelf, place to put. This is valid.
John put the book.
"put" has only objects book, thing to put ,not the place to put. This is invalid.

- **Empty Category Principle and Locality Theory**:Empty category principle says that all traces must be properly governed. The trace should not be too far away from its antecedent.
  Example: *Who do you think likes Mary?*
  Who_i *do you think* $[t_i likesMary]$*?*
  Here the trace t_i is governed by its antecedent who_i. This is valid.
  Who_i *do you think that* $[t_i likesMary]$*?*
  Here the trace t_i far away from its antecedent who_i. This is invalid.

Given the generator-filter model, the simplest way build a parser is to cascade the principles in a sequence.

1.6 Automatic Semantic Role Labeling

[DD02] This is a statistical technique of semantic role labeling, in which, a statistical classifier is trained over a corpora of sentences for the Semantic Role Labeling(SRL) task. Two sets of experiments were described by Jurafsky and Gildea, which were conducted by them on the Framenet corpus. In the first set, inputs to the system were a sentence, a target word, a frame and the frame element boundaries labeled by hand. The outputs were the frame element labels. In the second set of experiments, inputs to the system included just a sentence, a target word and a frame. The system now performed the dual task of *frame element boundary identification* and *frame element labeling*. Again the outputs were the frame element labels.

1.6.1 Features for frame element labeling

The sentence is first parsed by a constituent parser to obtain a parse tree. Then following features are derived:

Phrase type For every constituent of the sentence its phrase type can be determined by the constituent parse.

Governing Category A Noun Phrase(NP) can be directly a constituent of a Sentence(S) or a Verb Phrase(VP). This is used as a feature for only NPs giving a strong indication if it is used as subject or object of the verb.

Parse Tree Path A parse tree path of a constituent of a sentence is the path of the constituent from the target word in the constituent parse tree which includes the intermediate nodes and the arrow directions. Figure 1.5 shows an example path between verb *eat* and noun phrase *He* as: $VB \uparrow VP \uparrow S \uparrow NP$

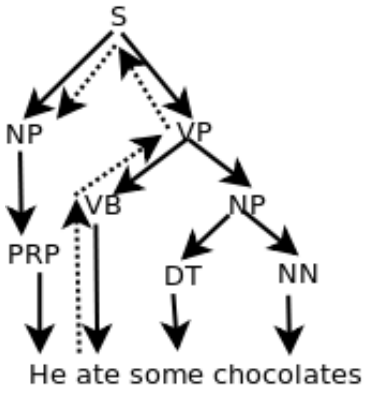

Figure 1.5: Parse tree path example.

Position The position feature indicates whether the constituent occurs before or after the predicate invoking the frame.

Voice The voice of the sentence is a feature. Active or passive voice is identified with the help of 10 passive identifying patterns.

Head Word The head words of each constituent acts as a very useful feature.

1.6.2 Features for frame element boundary identification

Similar features are also used for frame element boundary identification namely head word, parse tree path and target word.

1.6.3 Probability estimation of a single role

In order to automatically label the semantic role of a constituent, we wish to estimate a probability distribution indicating how likely the constituent is to fill each possible role, given the features described above and the predicate, or target word, t:

$$P(r|h, pt, gov, position, voice, t)$$

This could be done by direct counts as:

$$P(r|h, pt, gov, position, voice, t) = \frac{count(r,h,pt,gov,position,voice,t)}{count(h,pt,gov,position,voice,t)}$$

But there were on an average 34 sentences per target word in the Framenet dataset used. In general, the data is sparse for estimating the conditional probability of a label given all the above features together. This is because, a particular combination of the above six features along with the target word occurs rarely in the dataset. To overcome this, conditional probabilities of the frame element labels given subsets of above features like $p(role \mid targetword)$, $p(role|path, targetword)$ *etc.* are computed. These are combined using different strategies like

equal linear interpolation, EM linear interpolation, Geometric Mean, Back-off linear interpolation and Back-off geometric mean. This strategy gives a significant improvement in performance over the baseline approach of directly estimating the conditional probability of the labels given all the six features in the conditioning set.

1.6.4 Probability estimation of all the roles in the sentence

If we assume that the roles played by the different constituents of a sentence are independent of each other then the probability estimation of the previous section is enough to label the roles. But, it is trivial to note that this is just a simplifying assumption. If we relax this assumption, then we have to compute role assignment over the entire sentence r^*:

$$r^* = argmax_{r_{1...n}} P(r_{1...n}|t, f_{1...n})$$

Here, $f_{1...n}$ are the set of features as discussed above. Applying Bayes rule, removing elements not contributing to the argmax computation and assuming that features f_i are independent of each other given the target word t, we get the following equation:

$$r^* = argmax_{\{r_{1...n}\}} P(r_{1...n}|t) \prod_i P(f_i|r_i, t)$$

Applying Bayes rule again and removing constant part of numerator (not contributing to argmax) we get:

$$r^* = argmax_{\{r_{1...n}\}} P(r_{1...n}|t) \prod_i \frac{P(r_i|f_i,t)}{P(r_i|t)}$$

The frame element identification part can be incorporated in the above evaluation as follows:

$$r^* = argmax_{\{r_{1...n}\}} P(r_{1...n}|t) \prod_i \frac{P(r_i|f_i,fe_i,t).P(fe_i|f_i))}{P(r_i|t)}$$

1.6.5 Generalizing lexical semantics

The head word feature is observed to be the most useful. But due to large vocabulary of English, training on all possible head words is infeasible. Hence, to generalize our training from a small set of head words to other head words three techniques viz. automatic clustering, bootstrapping and making head word hierarchy using WordNet is described.

1.7 Extensions to Automatic SRL

[PWH$^+$04][PWH$^+$05] extend on this basic work. In the [PWH$^+$04], one vs rest Support Vector Machines(SVM) are trained for Automatic Semantic Role Labeling. The features used extend on the basic features combining features like:

- **Named Entities in Constituents** Named entities (PERSON, ORGANIZATION, LOCATION, PERCENT, MONEY, TIME, DATE) using Identi-Finder (Bikel et al,1999) added as 7 binary features
- **Head Word POS** Part of Speech tag of the headword

- **Verb Clustering** Predicate clustering to counter unknown predicates
- **Partial path** To avoid data sparsity parse tree paths in partial forms
- **Verb Sense Information** Word sense of the predicate
- **Head word of prepositional phrases** Replacing the preposition with the first noun as the head word
- **First and Last word/POS in constituent** Found to be discriminative
- **Ordinal constituent position** To avoid false positives of elements far away from predicate
- **Constituent Tree Distance** Finer way of depicting an existing feature
- **Constituent relative features** Features of parents and siblings
- **Temporal cue words** Temporal words not captured by NER
- **Dynamic class content** Hypotheses of at most two previous nodes belonging to the same tree

Some of these features were given by [SHWA03]

[PWH$^+$05] extend their own work by combining parses obtained from semantic parsers trained using different syntactic views like Charniak parser and Mini-par dependency parser.

[HY10] model word spans using an 80 state HMM model. Taking the hidden states as features in addition to the previous features an improvement is achieved in open domain SRL.

1.7.1 Other work

[ARR09] present an unsupervised argument identification algorithm for SRL. Using an unsupervised parser which generates unlabeled parse tree and POS tag annotation the algorithm is able to achieve 56% precision on the argument identification task.

1.8 Summary

Semantic Role Labeling is a fast developing area of research. It is a shallow level representation of semantics and has applications in large number of *Natural Language Processing* tasks. These include the Semantic Web, Information Extraction, Textual Entailment *etc.* With advances in both rule based and statistical techniques *Semantic Role Labeling* is a rapidly developing area in NLP.

Chapter 2

Dependency Grammar and Dependency Parsing

Dependency grammar is a theory that defines how the words in a sentence are connected to each other. The basic idea is that in a sentence all but one word is dependent on some other word. The word which is independent is usually the main verb of the sentence. Consider the following example :

Sentence: *A man sleeps.*

The main verb in the sentence is *sleeps* which is independent of all the other words and gives the central idea of the sentence. If the sentence is about a *sleep*, then there should be an *agent* or a *subject* of this action. Thus the word *man* is the *agent* or the *subject* who is sleeping and it depends on the word *sleeps*. We can also say that *man* fills the *verb-argument frame* for the verb *sleeps*. Now, in the sentence we are not talking about some specific *man* but about some indefinite *man*. This is denoted by the article *a*. Thus the word *a* is dependent on the word *man*, or in other words, *modifies* the word *man*. Thus, the dependency relations coming out of this simple relation is given in 2.1.

Figure 2.1: Dependency Relation

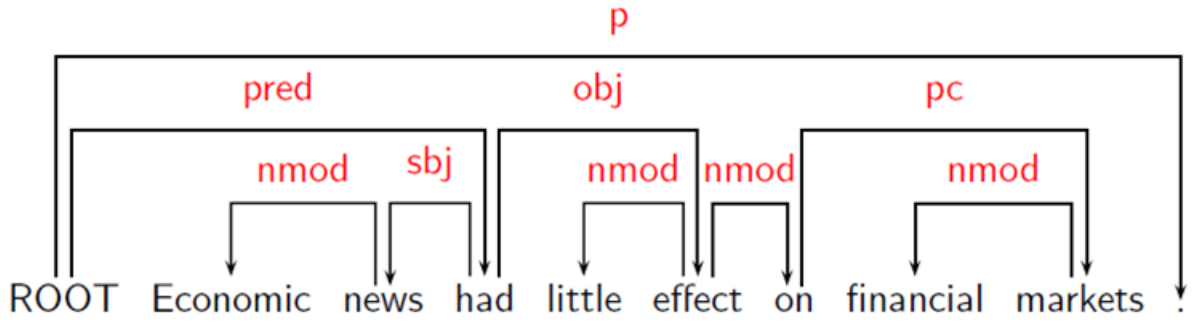

Figure 2.2: Projective example

2.1 Robinson's axioms

Robinson(1970) [Rob70] described four basic axioms for dependency grammar which govern the well-formedness of dependency structure. It says that in a dependency structure:

1. One and only one element is independent.
2. All others depend directly on some element.
3. No element directly depends on more than one element.
4. If A depends directly on B and some element C intervenes between them (in the linear order of the string), then C depends directly on A or B or some other intervening element.

The first axiom has the result that the dependency structure has a *root* node. The second axiom allows the dependency structure to be a single connected component. The third axiom is also called the *single-head* property says that every element can have at most one parent. Thus, the dependency structure can be a tree or a graph and different formalisms exist for both of them. The fourth axiom is the *projective* property, but many formalisms do not obey this axiom.

2.2 Projective and Non-projective dependency structures

A dependency structure is said to be *projective*, if it follows the fourth axiom of Robinson. A projective structure essentially means that no two edges of the graph cut-across each other. Let us look at an example 2.2.

As seen in the figure, no edges of the graph here cut-across each other. Algorithm designing for projective dependency structures is easier. But the problem with projective formalisms is, that languages with relatively free word order do not follow projectivity constraint. An example of a non-projective dependency structure is given in figure 2.3.

As can be seen, this sentence has a projective version, as seen in figure 2.4 if we modify the word order.

These three examples have been taken adapted from [Niv08].

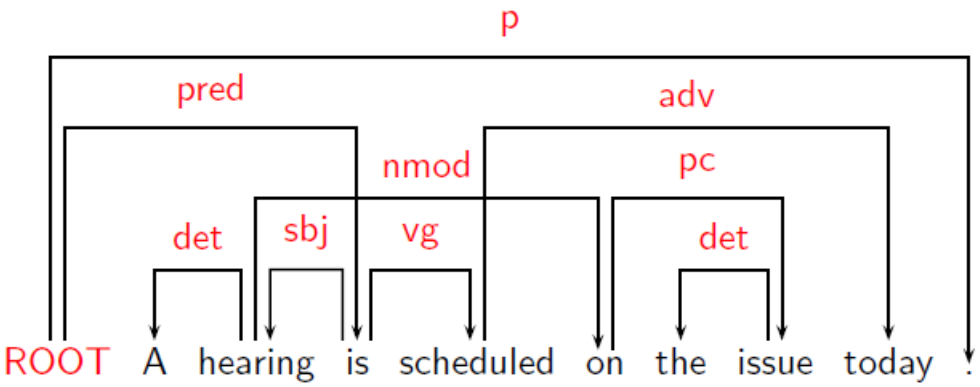

Figure 2.3: Non-projective example

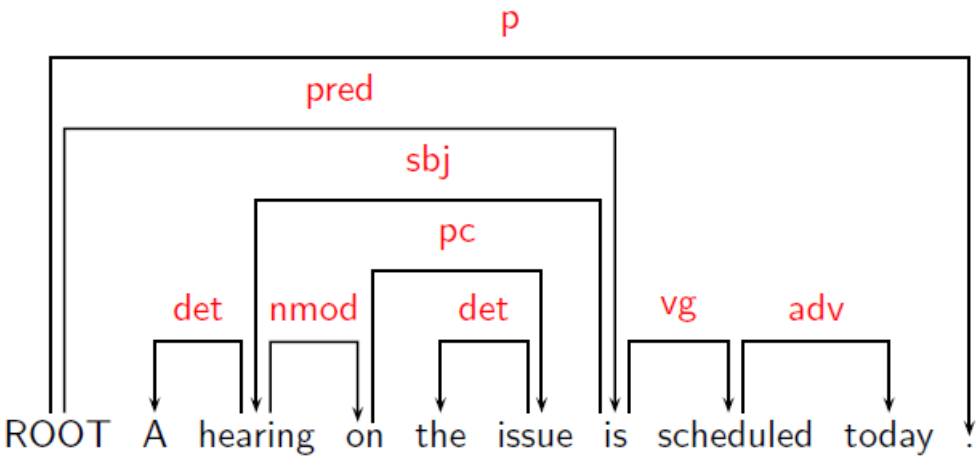

Figure 2.4: Projective version of the Non-projective example

2.3 Dependency Parsing Techniques

Dependency Parsing is the method of parsing sentences into *Dependency Structures*. In this section, we look at some of the prevalent *Dependency Parsing* techniques.

2.3.1 Data-based Dependency Parser

Data-driven dependency parsing uses machine learning from linguistic data to parse new sentences. In this report the supervised approaches will be discussed. The sentences used for machine learning are annotated with their correct dependency structures. The goal is to learn a good predictor of the dependency tree of a sentence given an input sentence. A model for this is M where M = (T, P, h), where T is the given set of constraints that helps in forming the structures for the sentence, p is a set of parameters to be learned from data and h is a fixed parsing algorithm.

2.3.2 Transition-based dependency parsing

Transition-based parsing system parameterizes a model to learn to predict the next transition given the input sentence and the parse history. The dependency trees are predicted using a greedy, deterministic algorithm.

A transition system generally contains a set of states, a set of rules to define transition of one state to another, an initial state and a set of final states. A simple stack-based transition system which uses a form of shift-reduce parsing will be explained. A configuration would be defined as a triple of stack, input buffer and a set of dependency arcs. A formal definition is:
Given a set of dependency types R, an input sentence S=$w_0 w_1 ... w_n$, the configuration c of the sentence S is defined as c = (σ, β, A), where

- σ is a stack of words w_i
- β is the input buffer
- A is a set of dependency arcs $(w_i, r, w_j) \in V_S R V_S$.

A configuration c contains partially processed words in the stack, remaining words to be processed in the buffer and the partially constructed dependency tree in the form of dependency arcs in the set A. σ and β are represented in the form of lists. The stack σ has its top at the right.
Initial configuration: $c_0(S) = ([w_0]_\sigma, [w_1, w_2, , w_n]_\beta, \phi)$.
Terminal configuration: $c_m(S) = (\sigma, []_\beta, A)$ for any σ and A.
Make w_0=ROOT and push it to stack and we reach configuration $c_0(S)$.

2.4 Summary

Dependency Grammar is one of the earliest language grammars which has regained it's importance with applications in Information Extraction and various Natural Language Processing tasks. A wide variety of dependency formalisms exist but most of them follow the basic axioms of Robinson. Dependency parsers like the Stanford Dependency Parser and the XLE parser

are now available and the accuracy of these parsers is improving rapidly with the use of both statistical and rule-based techniques.

Bibliography

[ARR09] O. Abend, R. Reichart, and A. Rappoport. Unsupervised argument identification for semantic role labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 28–36. Association for Computational Linguistics, 2009.

[BFL98] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *COLING-ACL '98: Proceedings of the Conference*, pages 86–90, Montreal, Canada, 1998.

[DD02] Gildea Daniel and Jurafsky Daniel. Automatic labeling of semantic roles. In *Computational Linguistics*, 2002.

[Dut11] Subhajit Dutta. Semantics extraction from text, stage 2 report. Master's thesis, IIT Bombay, 2011.

[Fel98] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[fra11] About framenet. `https://framenet.icsi.berkeley.edu/fndrupal/about`, October 2011.

[HY10] F. Huang and A. Yates. Open-domain semantic role labeling by modeling word spans. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 968–978. Association for Computational Linguistics, 2010.

[inf11] Inflection. `http://en.wikipedia.org/wiki/Inflection`, October 2011.

[KP03] P. Kingsbury and M. Palmer. Propbank: the next level of treebank. In *Proceedings of Treebanks and lexical Theories*, 2003.

[Lin94] D. Lin. Principar: an efficient, broad-coverage, principle-based parser. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 482–488. Association for Computational Linguistics, 1994.

[Niv08] J. Nivre. Sorting out dependency parsing. *Advances in Natural Language Processing*, pages 16–27, 2008.

[PWH+04] S. Pradhan, W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky. Shallow semantic parsing using support vector machines. In *Proceedings of HLT/NAACL*, pages 233–240, 2004.

[PWH+05] S. Pradhan, W. Ward, K. Hacioglu, J.H. Martin, and D. Jurafsky. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 581–588. Association for Computational Linguistics, 2005.

[Rob70] J.J. Robinson. Dependency structures and transformational rules. *Language*, pages 259–285, 1970.

[Roy11] Gourab Roy. Semantics extraction from text, stage 2 report. Master's thesis, IIT Bombay, 2011.

[SHWA03] M. Surdeanu, S. Harabagiu, J. Williams, and P. Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 8–15. Association for Computational Linguistics, 2003.

[ST95] D.D.K. Sleator and D. Temperley. Parsing english with a link grammar. *Arxiv preprint cmp-lg/9508004*, 1995.

[ver11] Verbnet. `http://verbs.colorado.edu/~mpalmer/projects/verbnet.html`, October 2011.