Location-Time Relation Extraction using a Link Grammar

Abhiram Ravi and Karthik Abinav

Computer Science and Engineering Department, IIT Madras, India
{abhiram@cse.iitm.ac.in,kabinav@cse.iitm.ac.in}

Abstract. Link grammars present a new approach to easily representing Natural Language, retaining the expressive power of Context Free Grammars. In this assignment, we look at an approach to Location-Time relation extraction from english sentences using the links generated by the link parser. We then go ahead to implement the idea using the Link Parser API provided by the Link Grammar Parser developed at CMU, and analyze the functioning of the algorithm on various different types of sentences.

Keywords: Link Parser, Relation Extraction

1 Introduction

The usage of formal grammars to model Natural Language syntax is an approach that has been actively explored for several years. Context Free Grammars, Dependency grammars, and Link Grammars fall under the category of syntactic theories where the language is viewed as a set of sentences, and sentences as an ordered collection of one or more words in a vocabulary; with the Dependency and Link Grammars being the modern approaches. Context Free Grammars work using the concept of derivation, and define a set of rules as to how one particular symbol can be rewritten as a sequence of others. They fall under the category of Constituent grammars, and establish phrase-structure rules. On the other hand, dependency and link grammars establish syntactic relations between pairs of words, and define constraints on these relations.

Relation extraction involves the detection and classification of semantic relationships between words. Our goal is to use the syntactic relations established by Link Parsers in order to make semantic deductions among the words/entities in the sentence.

The rest of this report is organized as follows. We first present a literature survey of existing papers in Dependency and Link grammars along with existing approaches to relationship extraction using them. We then go ahead to define our problem statement which is to search and establish a specific kind of relationship, the *Location-Time* relation, along with the motivation behind the problem. We then explain our methodology for performing this task and describe the details of our implementation over the Link Parser API. We then analyze the functioning of our algorithm on different types of sentences. We conclude by presenting the advantages and disadvantages of our approach, and our plans for the future on extending this work.

2 Literature Survey

Dependency Grammars define a syntactic structure which consists of binary asymmetric relations called *dependencies*. They lack phrasal nodes as compared to the constituency representation. The dependency relation holds between a head(governor) and a dependent(subordinate), where the head determines the syntactic and semantic *category* of the construction, and can usually replace the construction. The optional dependent gives semantic specification, whose position and form is defined by the head. As listed

by [3], dependency systems have three rules 1. For each category γ , assuming γ is a head, two lists of categories that could appear as dependents of γ on the corresponding sides of γ . 2. For each category, a list of words belonging to it. 3. List of categories that could possibly be governors in a sentence. Data driven approaches to Dependency parsing rely on a formal dependency grammar and use a corpus to induce probabilistic models for disambiguation.

Link Grammars [2] on the other hand specify a set of words with linking requirements, which are specified in a dictionary. Link grammars build undirected relations between pairs of words. A sentence is said to be part of a Link Grammar if there exists a way to draw arcs among the words that satisfy three criterion 1. Planarity - Crossing links are disallowed 2. Connectivity - The links connect all the words of the sequence together and 3. Satisfaction - All the linking requirements of each word must be satisfied. Link Parsers compute a set of links that satisfy these requirements, and a satisfying assignment is termed a *Linkage*. Each connector(left or right) in the linking requirement set begins with an upper case letter, followed by a sequence of lower-case letters or *'s. Two connectors match if, after adding an infinite sequence of *'s to each connector, all the non *'s are matched and the *'s correspond to *'s or lower-case letters in the other connector. [1]. Link grammars have a problem handling conjunctions, as they would inevitably result in crossing links. In order to handle them and some other constructs, the link parser also defines a post processing system that checks for certain conditions on the structure of the linkage.

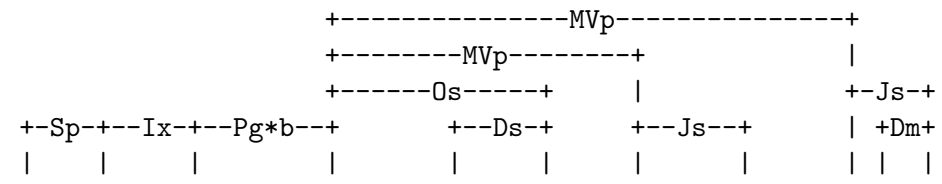

Figure 1 : The above diagram shows a linkage from link parsed sentence

RelEx [1] is an approach for extracting relations between proteins by generating dependency parse trees and extracting pathways between the words. The approach looks at three specific rules that reflect the most common relation constructs in english 1. A {action on} B, 2. {action by} A {on} B, 3. {Interaction between} A {and} B. and establishes conditions that must be satisfied in the dependency parse tree in the event of occurence of such constructs. Unlike dependency parsers like the Stanford parser which only looks at syntactic dependencies, RelEx extracts semantic information by paying special attention to whether the sentence is hypothetical or speculative by applying a small number of simple rules.

3 Problem Statement and Methodology

Keeping track of important dates and events is an essential aspect of everyday life. Information about these events and dates is most often received in the form of e-mail

[1] For example, Sp and Su don't match. S matches both Sp and Su. D*u matches Dmu but not D*m

or messaging services. Managing one's calendar by manually adding these events and dates to it is one way to approach the problem. We instead present an automatic way to extract these location-time relations from *sentences* and associate them with the corresponding event, all the information being obtained from the sentence itself. More formally, we extract the relation *location_time(event, place, time)* by working with link parsed sentences. Our approach, given a sentence, is as follows.

Step 1 : We first perform Named Entity Recognition (using Stanford's NER parser) on the given sentence and extract the named entities before using the Link Parser API.

Step 2 : We then feed the sentence into the Link Parser and obtain the Linkages corresponding to the sentence.

Step 3 : For each Linkage, we do the following : If there is a location entity, we start at the corresponding word, and trace backward the links generated by the link parser, to the subject of the sentence. This ensures that the location entity is associated with this subject(This is the advantage of using the link parser). From the subject, we then trace the links forward, with a specific set of rules on the connectors, and determine if there is path to a time entity that follows these rules. We then perform some additional analysis to check if an object(other than the location/time entities) is associated with the subject, and correspondingly identify the right event. If any of the above checks fail, we terminate and move on to the next linkage.

Step 4 : We output the location-time relations extracted from every Linkage as the output.

Step 3 can be more formally described as a flow diagram, which checks for conditions at each state and then makes a transition. Any condition violation i.e conditions other than the transition conditions mentioned, result in the reject state for that linkage. When all conditions are satisfied, the relation is established(accept state) between the corresponding elements.

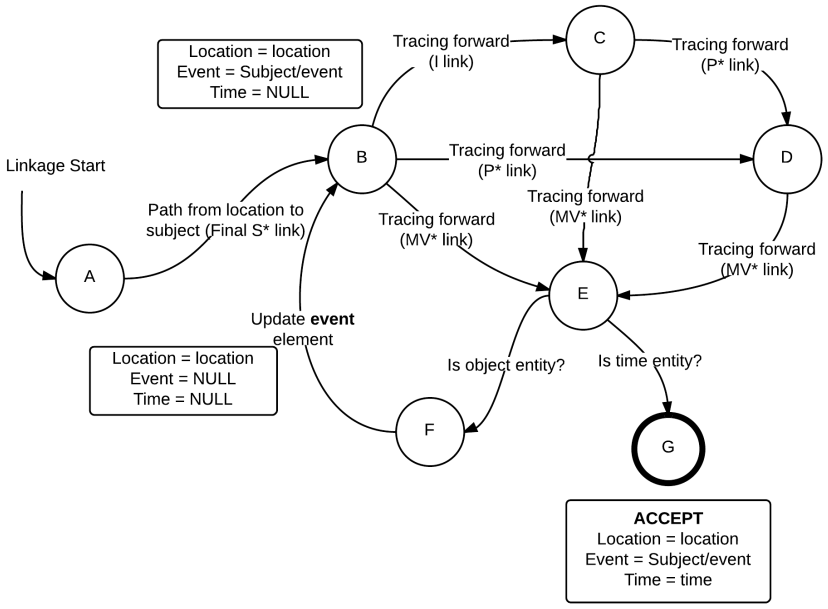

Figure 2 : A rough flow diagram for the analysis of each linkage.

4 Experiments and Results

Here is a sample list of sentence types that our algorithm handles. The Notation A/B for a word is used where A is the actual word, and B is the corresponding Named Entity Tag.

- The party begins in California/location at 5/time AM/time. - *location_time(party.n, California, 5 AM)*
- The party will happen in California/location at 5/time AM/time. - *location_time(party.n, California, 5 AM)*
- The party will be happening in California/location at 5/time AM/time. - *location_time(party.n, California, 5 AM)*
- We conduct the party/OBJECT in California/location on Fridays/time. - *location_time(party.n, California, Fridays)*
- We will be conducting the party in California/location on Fridays/time. - *location_time(party.n, California, Fridays)*
- The meeting with the Professor is scheduled at 9/time AM/time in BSB/location. - *location_time(meeting.n, BSB, 9 AM)*
- The Professor will meet you in BSB/location at 9/time AM/time. - *location_time(you, BSB, 9 AM)*
- An earthquake has been reported in Ohio/location at 4/time AM/time. - *location_time(earthquake.n, Ohio, 4 AM)*

5 Conclusion and Future Work

We have shown the capabilities of a link grammar in relation extraction by demonstrating a simple location-time relation extraction mechanism with the link parser. Link grammars are easy to describe, provide the right level of abstraction for semantic analysis in terms of the links, and hence demonstrate substantial power for in-depth analysis of sentences.

With respect to future work, we have the following ideas in mind. Currently, our evaluation measure is purely subjective. As an objective evaluation measure, we would like to work on a dataset with sentences labeled with the corresponding location-time relation(if existing), and then evaluate precision and recall measures of our algorithm on the dataset. We would also like to extend our algorithm to support more complex location-time relations, for example *The party happens in California/location and Michigan/location at 10/time AM/time and 11/time AM/time respectively.* We would then also like to build a google app that integrates itself into gmail, to automatically establish these relations from email sentences, and automatically adds them to the user's Google Calendar.

References

1. Fundel, K., Kffner, R., Zimmer, R., and Miyano, S. Relexrelation extraction using dependency parse trees. *Bioinformatics 23* (2007).
2. Nivre, J. Dependency grammar and dependency parsing. Tech. rep., Vxj University, 2005.
3. Sleator, D. D., and Temperley, D. Parsing english with a link grammar. *CoRR abs/cmp-lg/9508004* (1995).