Parsimonious Vole

A Systemic Functional Parser for English

Universität Bremen

Eugeniu Costetchi

Supervisor: Prof. John Bateman

Advisor: Dr. Eric Ras

Faculty 10: Linguistics and Literary Studies
University of Bremen

This dissertation is submitted for the degree of
Doctor of Philosophy

May 2018

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Eugeniu Costetchi
May 2018

Acknowledgements

And I would like to acknowledge ...

Abstract

This is where you write your abstract ...

Table of contents

List of figures

List of tables

List of definitions

Chapter 1

Introduction

1.1 On Artificial Intelligence (AI) and Computational Linguistics

In 1950 Alan Turing in a seminal paper (Turing 1950) published in Mind was asking if "machines can do what we (as thinking entities) can do?" He questioned what intelligence was and whether it could be manifested in machine actions indistinguishable from human actions.

He proposed the famous *Imitation Game* also known as the *Turing test* in which a machine would have to exhibit intelligent behaviour equivalent or indistinguishable from that of a human. The test was set up by stating the following rules. The machine (player A) and a human (player B) are engaged in a written *natural language* conversation with a human judge (player C) who has to decide whether each conversation partner is human or a machine. The goal of players A and B is to convince the judge (player C) that they are human.

This game underpins the question whether "a computer, communicating over a teleprinter, (can) fool a person into believing it is human?", moreover, whether it can exhibit (or even appear to exhibit) human(-like) cognitive capacities (Stevan Harnad 1992). Essential parts of such cognitive capacities and intelligent behaviour that the machine needs to exhibit are of course the linguistic competences of comprehension (or "understanding") and generation of "appropriate" responses (for a given input from the judge C).

The *Artificial Intelligence* (AI) field was born from dwelling on Turing's questions. The term was coined by McCarthy for the first time in 1955 referring to the "science and engineering of making intelligent machines" (McCarthy et al. 2006).

The general target is to program machines to do with language what humans do. Various fields of research contribute to this goal. Linguistics, amongst others, contributes with theoretical frameworks systematizing and accounting for language in terms of morphology, phonology, syntax, semantics, discourse or grammar in general. In computer science increasingly more efficient algorithms and machine learning techniques are developed. Computational linguistics provides methods of encoding linguistically motivated tasks in terms of formal data structures and computational goals. In addition, specific algorithms and heuristics operating within reasonable amounts of time with satisfiable levels of accuracy are tailored to accomplish those linguistically motivated tasks.

Computational Linguistics (CL) was mentioned in the 1950 in the context of automatic translation (Hutchins 1999) of Russian text into English and developing before the field of Artificial Intelligence proper. Only a few years later CL became a sub-domain of AI as an interdisciplinary field dedicated to developing algorithms and computer software for intelligent processing of text (leaving the very hard questions of intelligence and human cognition aside). Besides *machine translation* CL incorporates a broader range of tasks such as *speech synthesis and recognition, text tagging, syntactic and semantic parsing, text generation, document summarisation, information extraction* and others.

This thesis contributes to the field of CL and more specifically it is an advancement in *Natural Language Parsing* (NLP), one of the central CL tasks informally defined as the process of transforming a sentence into (rich) machine readable syntactic and semantic structure(s). Developing a program to automatically analyse text in terms of such structures by involving computer science and artificial intelligence techniques is a task that has been pursued for several decades and still continues to be a major challenge today. This is especially so when the target is *broad language coverage* and even more when the desired analysis goes beyond simple syntactic structures and towards richer functional and/or semantic descriptions useful in the latter stages of *Natural Language Understanding* (NLU). The current contribution aims at a reliable modular method for parsing unrestricted English text into a feature rich constituency structure using Systemic Functional Grammars (SFGs).

In computational linguistics, broad coverage natural language components now exist for several levels of linguistic abstraction, ranging from tagging and stemming, through syntactic analyses to semantic specifications. In general, the higher the degree of abstraction, the less accurate the coverage becomes and, the richer the linguistic description, the slower the parsing process is performed.

Such working components are already widely used to enable humans to explore and exploit large quantities of textual data for purposes that vary from the most theoretical, such as understanding how language works or the relation between form and meaning, to very pragmatic purposes such as developing systems with natural language interfaces, machine translation, document summarising, information extraction and question answering systems to name just a few.

1.2 Living in technologically ubiquitous world

Developed over thousands of years, the human language has become a versatile highly nuanced form of communication that carries a wealth of meaning which by far transcends the words alone. When it comes to *human-machine* interaction this highly articulated communication form is deemed impractical. So far humans had to learn to interact with computers and do it in formal, strict and rigorous manner via graphical user interfaces, command line terminals and programming languages. Advancements in *Natural Language Processing* (NLP) which is a branch of *Artificial Intelligence* (AI) are a game changer in this domain. NLP starts to unlocks the information treasure locked in the human speech and make it available for processing to computers. NLP becomes an important technology in bridging the gap between natural data and digital structured data.

In a world such ours, where technology is ubiquitous and pervasive in almost all aspects of our life, NLP becomes of great value and importance regardless whether it materializes as a spell-checker, intuitive recommender system, spam filter, (not so) clever machine translator, voice controlled car, intelligent assistants such as Siri, Alexa or Google Now.

Every time you ask Siri or Alexa for directions to the nearest Peruvian restaurant, how to cook Romanian beef stew or what is the dictionary definition for the word *germane*, a complex chain of operations is activated that allows 'her' to understand the question, search for the information you are looking for and respond in a human understandable language. Such tasks are possible only in the past few years thanks to advances in NLP. Until now we have been interacting with computers in a language they understand rather than us. Now they are learning our language.

1.3 NLP for businesses

NLP opens new and quite dramatic horizons for businesses. Navigating with limited resources stormy markets of competitors, customers and regulators and finding an optimal answer/action to a business question is not a trivial task. Markets are influenced by the information exchange and being able to process massive amounts of text and extract meaning can help asses the status of an industry and play an essential role in crafting a strategy or a tactical action. Relevant NLP tasks for gathering market intelligence are *named entity recognition* (NER), *event extraction* and *sentence classification*. With these tasks alone one can build a database about companies, people, governments, places, events together with positive or negative statements about them and run versatile analytics to audit the state of affairs.

Compliance with governmental, European or international regulations is a big issue for large corporations. One question for addressing this problem is whether a product is a liability or not and if yes then in which way. Pharma companies for example, once a drug has been released for clinical trials, need to process the unstructured clinical narratives or patient's reports about their health and gather information on the side effects. The NLP tasks needed for this applications are primarily *NER* to extract names of drugs, patients and pharma companies and *relation detection* used to identify the context in which the side effect is mentioned. NER task help transforming a sentence such as "Valium makes me sleepy" to "(drug) makes me (symptom)" and relation detection will apply patterns such as "I felt (symptom) after taking (drug)" to detect the presence of side effects.

Many customers, before buying a product, check online reviews about the company and the product whether it is pizza or a smartphone. Popular sources for such inquiry are the blogs, forums, reviews, social media, reports, news, company websites, etc. All of them contain a plethora of precious information that stays trapped in unstructured human generated text. This information if unlocked can play a great deal in company's reputation management and decisions for necessary actions to improve it. The NLP tasks sufficient to address this business required are *sentiment analysis* to identify attitude, judgement, emotions and intent of the speaker, and *co-reference resolution* which connects mentions of things to their pronominal reference in the following or preceding text. These tasks alone can extract the positive and negative attitudes from sentence "The pizza was amazing but the waiter was awful!" and connect it to the following sentence "I adore when it is topped with my favourite artichoke" about pizza and not the waiter and discover a topping preference.

NLP is heavily used in customer service in order to figure out what customer means not just what she says. Interaction of companies with their customers contain many hints pointing towards their dissatisfaction and interaction itself is often one of the causes. Companies record, transcribe and analyse large numbers of call recordings for extended insights. They deploy chat bots fo increased responsiveness by providing immediate answers to simple needs and also decrease the load of the help desk staff. NLP tasks that are essential in addressing some of the customer service needs are *speech recognition* that converts speech audio signal into text and *question answering* which is a complex task of recognising the human language question, extract the meaning, searching relevant information in a knowledge base and generate an ineligible answer. Advances in deep learning allow nowadays to skip the need for searching in a knowledge base by learning from large corpora of question-answer pairs complex interrelations.

The above cases underline the increased need in NLP whereas the variation and ever increasing complexity of tasks reveal the need in deeper and richer semantic and pragmatic analysis across a broad range of domains and applications. Any analysis of text beyond the formal aspects such as morphology, lexis and syntax inevitably lead to a functional paradigm of some sort which can be applied not only at the clause level but at the discourse as a whole. This makes the text also an artefact with relation socio-cultural context where it occurs.

1.4 The linguistic framework

Any description or analysis involving language implies some theory of about its essential nature and how it works. A linguistic theory includes also goals of linguistics, assumptions about which methods are appropriate to approach those goals and assumptions about the relation between theory, description and applications (Fawcett 2000).

In this thesis I adopt the Systemic Functional Linguistic (SFL) framework because of its versatility to account for the complexity and phenomenological diversity of human language providing descriptions along *multiple semiotic dimensions* i.e. paradigmatic, syntagmatic, meta-functional, stratification and instantiation dimensions (Halliday 2003) and at different *delicacy levels* of the *lexico-grammatical cline* (Halliday 2002; Hasan 2014). Just like the resolution of a digital photo defines the clarity and the amount of detail in the picture, the same way delicacy refers to the how fine or coarse grained distinctions are made in the description of the language. And intuitively the lexico-grammar is the combination of the lexis and grammar into a single description.

These notions and other elements of the SFL theory are addressed below in Chapter **??**.

In his seminal paper "Categories of the theory of grammar" (Halliday 1961), Halliday lays the foundations of SFL following the works of his British teacher J. R. Firth, inspired by Louis Hjelmslev (Hjelmslev 1953) from Copehagen School of linguistics and by a group of European linguists from Prague Linguistic Circle. This paper constitutes a response to the need for a *general theory of language* that would be holistic enough to guide empirical research in the broad discipline of linguistic science:

> ... the need for a *general* theory of description, as opposed to a *universal* scheme of descriptive categories, has long been apparent, if often unformulated, in the description of all languages (Halliday 1957: p.54; emphasis in original)

> If we consider general linguistics to be the body of theory, which guides and controls the procedures of the various branches of linguistic science, then any linguistic study, historical or descriptive, particular or comparative, draws on and contributes to the principles of general linguistics (Halliday 1957: p.55)

With this perspective, paradigmatic organization of language received priority as the primary focus of linguistic description and subsequently the structure is analysed as a realisation of features.

SFL regards language as a social semiotic system where any act of communication is regarded as a conflation of *linguistic choices* available in a particular language. Choices are organised on a paradigmatic rather than structural axis and represented as *system networks*. Moreover, in the SFL perspective language has evolved to serve particular *functions* influencing their the structure and organisation of the language. However, their organisation around the paradigmatic dimension leads to a significantly different functional organisation than those found in several other frameworks which Butler (2003a,b) extensively addresses.

Embracing the *oragnon model* formulated by Bühler (1934), Halliday refers to the language functions as metafunctions or lines of meaning offering a trinocular perspective on language through *ideational*, *interpersonal* and *textual* metafunctions. In SFL, language is first of all an interactive action serving to enact social relations under the umbrella of the *interpersonal metafunction*. Then it is a medium to express the embodied human experience of inner (mental) and outer (perceived material) worlds

via *ideational metafunction*. Finally the two weave together into a coherent discourse flow whose mechanisms are characterised through the *textual metafunction*.

Until today, two major Systemic Functional Grammars (SFG) have been developed: the *Sydney Grammar* (Halliday & Matthiessen 2013) and the *Cardiff Grammar* (Fawcett 2008). The latter, as Fawcett himself regards it, is an extension and a simplification of Sydney Grammar (Fawcett 2008: xviii). Each of the two grammars has advantages and shortcomings (presented in Chapter **??**) which I discuss from the perspective of theoretical soundness and suitability to the goals of the current project.

Both Cardiff and Sydney grammars had been used as language models in natural language generation projects within the broader contexts of social interaction. Some researchers (Kasper 1988; O'Donoghue 1991; O'Donnell 1993; Souter 1996; Day 2007) attempted to reuse the grammars for the purpose of syntactic parsing within the borders of NL generation coverage. I come back to these works in more detail in Section 2.1.

As we part away from the surface form of text and aim for rich semantics or aim at analyses higher than the clause level i.e. discourse, the functional is increasingly useful and revealing of meanings in text. Such analyses have been done manually by linguists, semioticians and educators in an informal manner as there have not been any tools to automate such processes. Besides Linguistics, there is a plethora of linguistic analysis using SFL framework in other fields of research. SFL has been used extensively as a descriptive framework in Critical Discourse Analysis and in Education studies. Automatising the language analysis with SFL framework will unlock the potential of these fields. Next I provide a glimpse of what opportunities they offer.

1.5 An example of Systemic Functional analysis

This section provides a taste of what does a systemic functional analysis looks like. I provide a parallel analysis between SFL and a traditional grammar in order to highlight the richness and high descriptive potential of SFGs. A source of the descriptive abundance in SFL is achieved through a practice of feature systematisation as mutually exclusive choices which is exemplified for three features of traditional grammar below. The SFL feature analysis provided here is partial and restricted to only two constituents as this suffices to provide the reader with an intuition of what to expect from an SFL analysis.

Traditional linguistics teaches us how to carry on a syntactic analysis of a sentence. So let's consider Example 1 in order to perform one. First, we focus on clustering

words together into constituents guided by the intuitive rule *which word stands goes together with* within the sentence resulting in a grouping such as in Example 2.

(1) He gave the cake away.

(2) ((He) (gave) ((the) (cake)) (away) (.))

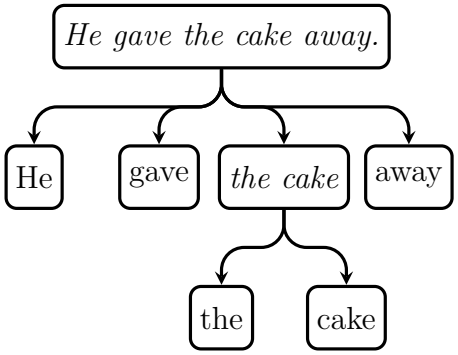


Fig. 1.1 Representation of the Example 2 as constituency tree

Figure 1.1 depicts the constituency division of the clause which is identical to the bracket notation in Example 2. The nodes represent grammatical constituents and the edges stand for the structure-substructure composition.

Next we can move on to assign constituent class and a grammatical function. Table 1.1 provides a constituency analysis in SFL tradition. Here the sentence is formed of a single clause which has four constituting functional parts: a Subject, a Main Verb (also known as Predicate), a Complement and an Adjunct. Each of these functional parts is filled correspondingly by a pronoun, a verb, a nominal group and an adverb as assigned in the table below.

He	*gave*	*the*	*cake*	*away.*
clause				
Subject	Main Verb	Complement		Adjunct
pronoun	verb	nominal group		adverb
		Deictic	Thing	
		determiner	noun	

Table 1.1 Constituency analysis with unit classes and grammatical functions

Next each constituent can be assigned a set of relevant features. For example The subject “He” is a pronoun that has features known in traditional grammar: *singular*, *masculine*, and 3^{rd} *person*. These features are well differentiated in traditional grammar. For example *singular* means *non-plural*, *masculine* means *non-feminine* and 3^{rd} *person*

means *non-1st* and *non-2nd*. These are closed classes meaning that there is no *4th person* or that there is no *neutral* grammatical gender in English as other languages have. These features can be systematised (see Figure 1.2) as three systems of mutually exclusive choices that can be assigned to pronominal units. Note that the gender is enabled for *3rd* person singular pronouns which can be expressed as is the figure below representing a *system network* (which is properly introduced in Chapter **??**).

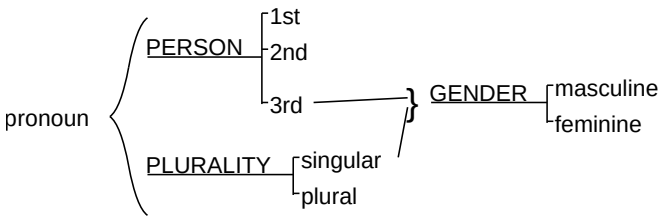


Fig. 1.2 The systematisation of three pronominal features in traditional grammar

In SFG the pronouns are systematised in the system network of Person from *Introduction to Functional Grammar* (Halliday & Matthiessen 2013: 366) that is depicted in Figure 1.3. The red rectangles from the figure represent the selections that are applicable to the Subject constituent "He" in example above.

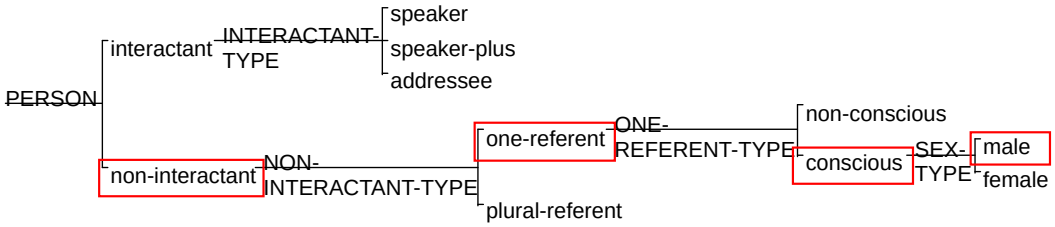


Fig. 1.3 The selections in Person system network from Halliday & Matthiessen (2013: 366)

Lets take now the clause constituent that is the root of the constituency tree and see how SFL features can be applied to it. If in in terms of traditional grammar the clause can be ascribed relatively few features i.e. as having *passive voice*, *positive polarity* and *simple past tense* then in terms of SFL grammar the features are many more i.e. *major, positive, active, effective, receptive, agentive, free, finite, temporal, past, non-progressive, non-perfect, declarative, indicative, mood-non-assessed, comment-non-assessed*. Figure 1.4 depicts the selections applicable to clause constituent in Example 1 from Mood system network that is an adaptation of Mood network proposed in Halliday & Matthiessen (2013: 162).

So far you have seen constituents assigned syntactic functions such as Subject, Complement, Adjunct etc. SFL covers a wider range of functions depending on the kind of meaning it aims at describing. For example what in other grammars

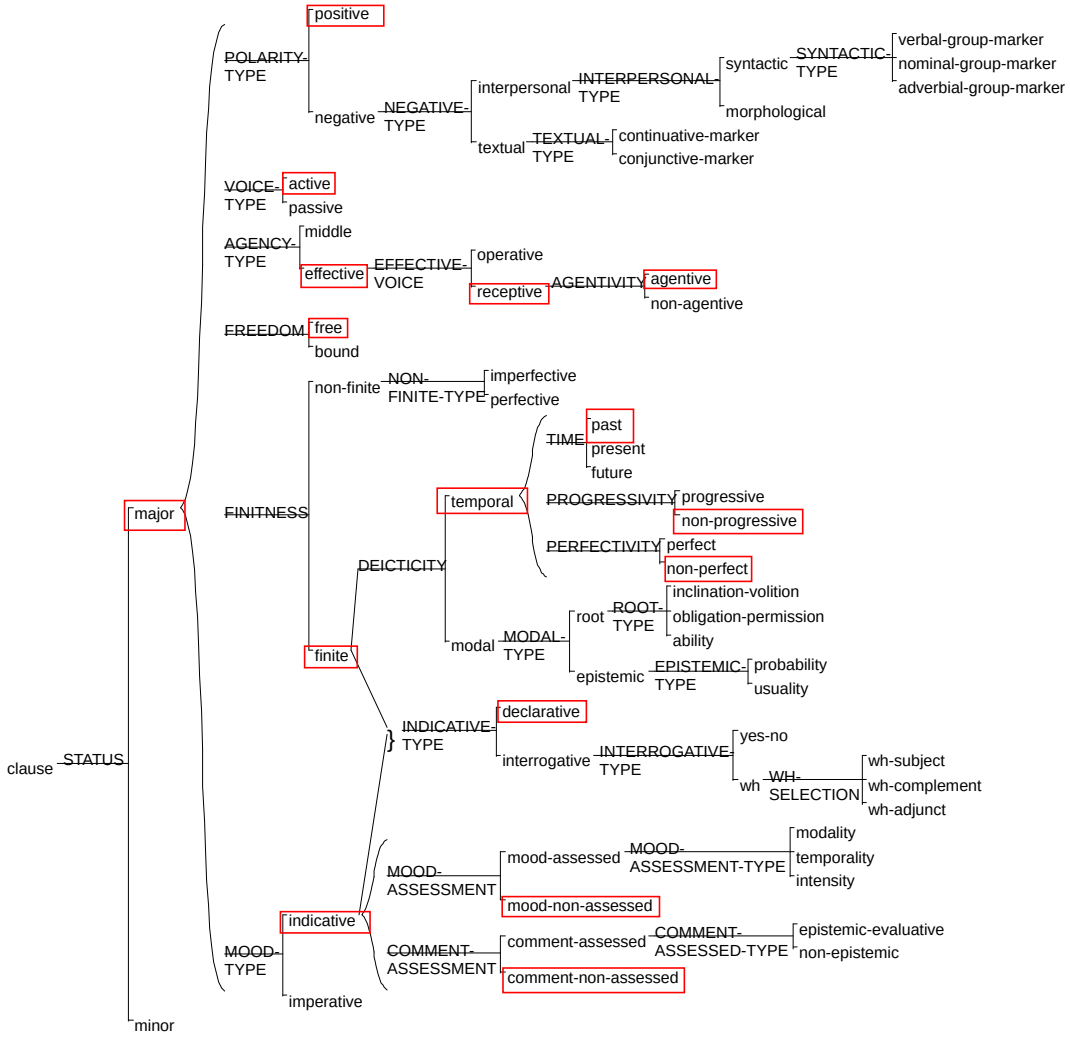


Fig. 1.4 The systematisation of three pronominal features in traditional grammar

is known as *semantic labels*, *thematic* or *θ roles* SFL systematises as Transitivity system network (which will be introduced in Chapter **??** below). Transitivity aims at providing domain independent *semantic frames* called in SFL *process configurations* which describe semantic actions and relationships, along with *semantic roles* ascribed to their *participants*. These semantic frames generally are governed by verbs and more specifically each verb meaning has a dedicated semantic frame.

For example 1 corresponds to a Possessive semantic frame where "He" is the Agent and Carrier whereas "the cake" is the Possessed thing as marked in Example 3. These configurations and participant roles correspond to Transitivity system network proposed in (Neale 2002).

(3) [$_{Agent-Carrier}$ He] gave [$_{Possessed}$ the cake] away.

There are more functions and features that can be assigned to the constituents in the Example 1 but I stop here. The analysis provided so far highlights that SFG grammar has a variety of functions serving to express different meanings. The traditional grammar distinguishes them as syntactic and semantic functions but, as we will see in Chapter **??** below, SFL does not make such a distinction. Another aim of the current section was to provide a glance of the feature rich grammar and I hope the example with Mood feature selection in Figure 1.4 fulfils this goal.

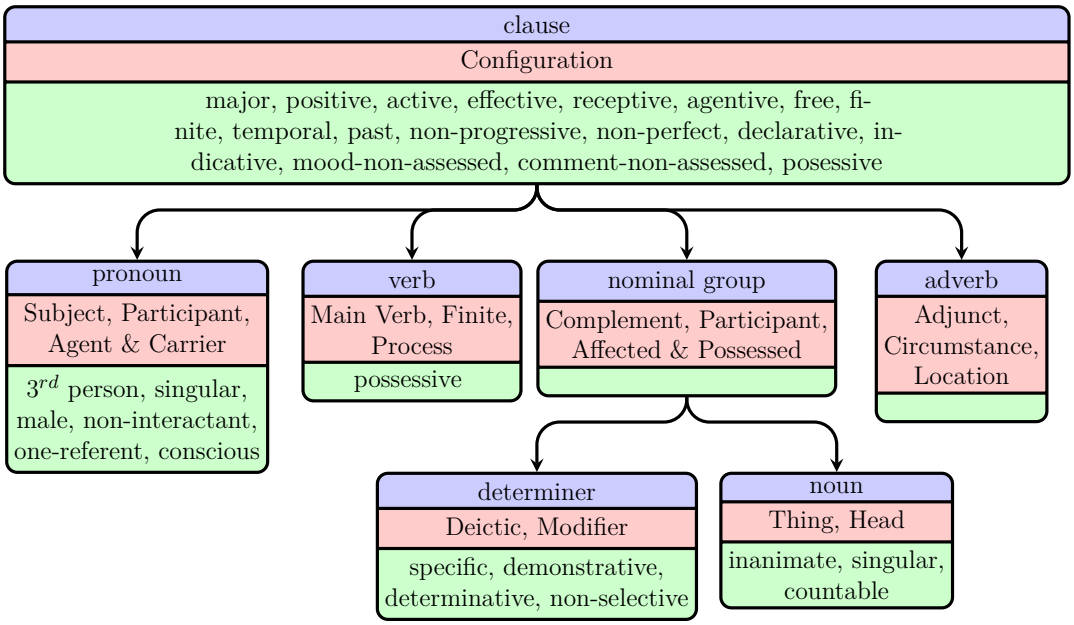


Fig. 1.5 Representation of Example 1 as feature rich constituency graph

Next, Figure 1.5 summarises everything discussed above into a partially filled constituency tree. The constituents that were not discussed are assigned only few high level functions and features. As you can see, every node is richly decorated with syntactic and semantic features. The blue part of each node denotes grammatical class, the red part carries functions some of which are important to establishing a valid constituency structure (that are Mood functions) and the the Transitivity functions; and the green part some grammatical features selected from system networks. In practice, the feature set is much richer than what nodes in Figure 1.5 carry here the restriction aims simply to avoid an over-crowded example.

Next I describe what opportunities and limitations exist in automatically generating rich SFL analyses as until now it has not been possible to use these detailed analysis in computational contexts. This makes them unavailable for corpus work, for training data

in machine learning and other end-user application scenarios provided as motivation in the Sections 1.2 and 1.3 above.

1.6 The problem of parsing with SFGs

SFL since it has been established has been primarily concerned with the paradigmatic axis of language. Accounts of the syntagmatic axis of language, for example syntactic structure, have been put in the background. The structure has been placed on the theoretical map and defined in terms of rank, unit, class and function, as we will see in detail in Chapter **??** but afterwards it received minimal attention as most of the focus was on the paradigmatic organisation in language (in fact this is is the feature that sets SFL apart from other approaches to study language). This has led to progress in accounting how language works at all strata but little was said about the language constituency. And this can be considered "unsolved" within SFL accounts leaving a "gap in what must be one the central areas of any characterisation of language" (Bateman 2008: 25).

This may be surprising as the syntagmatic dimension is implicit and present everywhere in the SFL literature. For instance all example analyses in the *Introduction to Functional Grammar* (Halliday & Matthiessen 2013) are predominantly syntagmatic. Moreover, Robin Fawcett for decades promotes the motto *no system network without realisation statements* (Fawcett 1988: 9). Bateman (2008) presents in detail why there is a severe imbalance between syntagmatic and paradigmatic axes in SFL, how it came to be this way and how it is especially damaging to the task of automatic text analysis. Next I describe the main problems and hint at how potential solutions may look like (for a full account of the solution persuaded in this thesis see Chapter 3).

O'Donnell & Bateman (2005) offer a detailed description to the long history of SFL being applied in computational contexts yielding with productive outcomes on language theorising, description and processing. The transfer between SFL and computation typically involved a delay between the theoretical formulation and the computational instantiation of that formulation (Bateman & Matthiessen 1988: 139) (Matthiessen & Bateman 1991: 19). The theoretically formulated ideas contain hidden pitfalls that are revealed only upon explicit formulations required by in computation (Bateman 2008: 27).

The active exchange between the SFL theory and computation has been almost entirely oriented towards automatic *natural language generation*. Such systems would use abstract semantic specifications and communicative goals to produce grammatically

correct and well connected texts achieving those goals. This area has been shown to be successful in part due to decomposition of language along the paradigmatic axis using functionally motivated sets of choices between functionally motivated alternatives (McDonald 1980).

The computational processes driving the natural language generation relied heavily on the notion of *search*. A well defined search problem is defined in terms of a precise description of the search space which then helps navigation process effectively to find solutions. The paradigmatic organization of the *lexicogrammar* as system networks (that is in SFL the combination of grammar and lexis explained in Section **??** below) assumed within SFL turns out to organise the search space for possible grammatical units appropriate for communicative goals in almost ideal manner (Bateman 2008: 28).

The *automatic analysis* or *parsing* can be seen as a reverse problem of finding appropriate analysis within a search space of possible solutions. That is identify the exact meaning, systematised in the grammar, of a given natural language sentence. As seen in Section 1.5 above, an account of the sentence meaning would have to provide both, in terms of a formal structure of the sentence revealing the constituents plus their syntactic relations to each other, and in terms of a complete set of features (detailed to the extent that grammar permits) applicable to each constituent of structure. If, in the generation process, the abstract semantic specifications are increasingly materialised through choice making by traversing the system network towards finally generated text, then, in the parsing process, the reverse is the case. The process starts from a given sentence aiming to derive/search the feature choices in the system network afferent to each of the constituents. But if the paradigmatically organised lexicogrammatical resource is effective for generation it turns out, as we will see next, to be by far unsuitable for the analysis task because of the *size problem*. Halliday himself mentions this problem when he asks *how big is a grammar?*.

> Given any system network it should in principle be possible to count the number of alternatives shown to be available. In practice, it is quite difficult to calculate the number of different selection expressions that are generated by a network of any considerable complexity (Halliday 1996: 10).

The issue emerges from the way connections and (cross-)classifications are organised in a system network. In addition to that, the orientation of systemic grammars towards choice means that the grammar full of disjunctions leading to the problem of search complexity. Also the abstract nature of systemic features leads to a structural richness that adds logical complexity to the task (O'Donnell 1993). So estimating the size of the

grammar would in fact mean estimating the potential number of feature combinations. For example, a hypothetical network of 40 systems the "size of the grammar it generates lies somewhere between 41 and 2^{40} (which is somewhere around 10^{12})" (Bateman 2008: 28). However it is not easy to predict where would the upper limit of the grammar would fall even when the configuration of relations of a particular system network is known.

For the generation task, that size, is not a problem at all as the number of choice points is actually rather small. Such a paradigmatic organisation is, in fact, an incredibly concise and efficient way to express the linguistic choices where the possible feature selections are relevant only when they are enabled by prior paradigmatic choices and it is only those alternatives that need to be considered (Halliday 1996: 12–13) .

In the analysis task, the paradigmatic context of choice, that helps navigation during the generation process, is no longer available. It is not known any longer which features of a systemic network are relevant and which are not. This leads to a radical asymmetry between the two tasks. That is: in generation, the simple traversal of the network finds only the compatible choices because that is what the network leads to; whereas in analysis it is not evident in advance which path to follow therefore the task is virtually to explore entire search space in order to discover which features apply to the text. This means that any path is potentially relevant and shall be passed and checked leading to evaluation of the system network as a whole; and that there is no way to restricting the search space, as in the case of generation, to a a set of familiar paradigmatic lines (Bateman 2008: 29).

One of the grammars successfully used in generation tasks is Nigel grammar developed within Penman generation project (Mann 1983). It contains 767 grammatical systems defined over 1381 grammatical features which Bateman evaluates as "a very large computational grammar by current standards, although nowadays by no means the broadest when considered in terms of raw grammatical coverage" (Bateman 2008: 29). To parse with such a grammar would means exploring an incredibly vast search space 3×10^{18} to be more precise. A more detailed break down the complexity by rank or primary class as provided in Table 1.2 below.

Another difficulty in parsing with SFGs lays in the fact that, as the analysis moves away from directly observable grammatical variations towards more abstract semantic variations, the difficulty of generating an accurate account increases drastically. Transitivity system network for example is made up of such semantic features and it is comparable to what is called in computational linguistics (shallow) *semantic parsing*.

rank or primary class	*size*
adverbial-group	18
words	253
quantity-group	356
prepositional-phrase	744
adjectival-group	1045
nominal-group	$>2 \times 10^9$
clause	$>3 \times 10^{18}$

Table 1.2 Size of major components of the Nigel grammar expressed in terms of the number of selection expressions generated (Bateman 2008: 35)

The main challenges (well explained in (Gildea & Jurafsky 2002: 245–250)) in here remains the same since Winograd (1972) that is moving away from the domain specific, hand-crafted semantic specifications towards domain independent and robust set of semantic specifications. This goal was undertaken in several projects to build large broad-scope lexico-semantic databases such as WordNet (Fellbaum & Miller 1998), FrameNet (Baker et al. 1998; Johnson & Fillmore 2000; Fillmore et al. 2003) and VerbNet (Schuler 2005; Kipper et al. 2008). A similar database exists for Transitivity system network as described in Fawcett (2009) called Process Type Database (Neale 2002).

Such databases describe domain independent *semantic frames* (know in SFL as *configurations* or *figures*) which describe semantic actions and relationships, along with *semantic roles* ascribed to their *participants*. The semantic frames generally are governed by verbs and more specifically each verb meaning has a dedicated semantic frame. For instance the perception frame contains *Perceiver* and *Phenomenon* roles as can be seen in Example 4.

(4) [$_{Agent-Perceiver}$ Jaqueline] glanced [$_{Phenomenon}$ at her new watch].

The tendency is to identify frames that are generic enough to cover classes of verb meanings (for example Action, Cognition, Perception, Possession frames) and the same applies to participant roles where the tendency is to reuse roles across semantic frames (for example agent role from Action frame is reused in Perception or Possession frames, or Phenomenon is reused in Cognition and Perception frames).

Besides being difficult to pin them down in text, the problem with semantic features goes even step further. Sometimes the semantic roles correspond to constituents that are displaced or not even realised in the text which makes them even more difficult to identify and correctly assign.

Now the question would be how could the vast search space of grammars such as Nigel be restricted to a reasonable size and how can be compensated the lack of proper syntagmatic description in SFGs? The first part of the question has already been addressed in O'Donnell (1993) but the lack for an answer to the second part and probably for other hidden reasons the results are not usable in real world applications. In fact the have been more attempts, Kasper (1988), Kay (1985), O'Donoghue (1991), O'Donnell (1993) and Day (2007), to mention just a few, none of which managed to parse broad coverage English with full SFG without aid of some sort. Each had to accept limitations either in grammar or language size and eventually used simpler syntactic trees as a starting point of the parsing process. A detailed account of the current state of the art in parsing with SFGs is provided in Chapter 2.

1.7 The problem of semantic parsing

Semantic role labelling, as motived in (Gildea & Jurafsky 2002: 246), can play an important role in word sense disambiguation, information extraction, question answering, semantic dialogue systems, machine translation, automatic text summarisation.

1.8 On theoretical compatibility and reuse

In the past decades many significant progresses have been made in natural language parsing framed in one or another linguistic theory each adopting a distinct perspective and set of assumptions about language. The theoretical layout and the available resources influence directly what is implemented into the parser and each implementation approach encounters challenges that may or may not be common to other approaches in the same or other theories.

Parsers implementing one theoretical framework may face common or different problems to those implementing other theories. The same can be said of the solutions. When a solution is achieved using one theory it is potentially reusable in others. The successes and achievements in any school of thought can be regarded as valuable cross theoretical results to the degree links and correspondences can be established. Therefore reusing components that have been shown to work and yield "good enough results" is a strong pragmatic motivation in the present work.

This thesis employs three linguistic frameworks namely the *Systemic Functional Linguistics*, *Dependency Grammar* and *Governance & Binding Theory*. SFL has already been motivated in Section 1.4 and is introduced in detail in Chapter **??**. The other two

frameworks are employed because some of the accomplishments using them are directly reused in this thesis and I explain below how. Chapter ?? and 4 besides introducing the Dependency grammar and correspondingly Government and Binding Theory show how these frameworks relate to each other and to which degree they are compatible to undergo a conversion process for the purpose of reuse.

In the last years *Dependency Grammar* (Tesniere 2015) became quite popular in natural language processing world favoured in many projects and systems. The grammatical lightness and the modern algorithms implemented into dependency parsers such as Stanford Dependency Parser (Marneffe et al. 2006), MaltParser (Nivre 2006), MSTParser (McDonald et al. 2006) and Enju (Miyao & Tsujii 2005) are increasingly efficient and highly accurate. Among the variety of dependency parsing algorithms, a special contribution bring the *machine learning* methods such as those described in McDonald et al. (2005); McDonald & Pereira (2006); Carreras (2007); Zhang & Nivre (2011); Pei et al. (2015) to name just a few.

As the dependency parse structures provide information about functional dependencies between words and grants direct access to the predicate-argument relations and can be used off the shelf for real world applications. This information alone, would makes the dependency grammar a suitable candidate to supplement the syntagmatic account missing in SFGs and provide some functional hooks for reducing complexity in parsing with SFGs. Hence once of the goals in this work is investigating to which degree the dependency grammar is structurally and functionally compatible with SFGs to undergo a cross theoretic transformation. This hypothesis is theoretically investigated in Chapter ?? and then evaluate empirically in Chapter ??. The investigation is based specifically on Stanford Dependencies parser version 3.5 (Marneffe & Manning 2008b,a; Marneffe et al. 2014).

to be continued for GBT

1.9 Thesis Goal and Proposed Solution

The first difficulty that needs to be addressed, in the analysis process, is discovering from a sequence of words what possible groups are combinable into grammatical groups, phrases or clauses. This is a task of bridging a sequence of words as input and the grammatical description of how they can combine to form a (syntactic) structure (known in SFL as *syntagmatic organizations* and described in Section ??) can already partition the search space into relevant network sub-parts cutting down the complexity by a large factor. Moreover if the syntagmatic account would involve *configurations*

of grammatical functions (see Section **??**) then these grammatical functions can serve as paradigmatic context (similar to the one available during generation task) for traversing the system network and extend to the full set of systemic features. This sort of description can restrict the search space to applicable network parts and provide to an extent traversal context during analysis task.

Addressing the gap of the syntagmatic account within the SFG framework, can be done by, first, providing information about which grammatical functions operate at each rank, second which grammatical functions can be filled by which classes of units and third by providing relative and absolute account of ordering within each unit structure. This sort of information can guide the building of a constituency backbone structure. As a second stage, as mentioned above, the unit classes and grammatical functions can operate as "hooks" on system network to guide the traversal in the same way the paradigmatic context available in the generation process.

Alternatively the problem of structure construction can be outsourced for parsing with other grammars. Then the problem changes into creating a transformation mechanism to obtain the SFL constituency structure rather than build it from scratch. Starting the SFG parsing process from a syntactic tree produced with other grammars reduces the computational complexity of the task and reduce the search space.

The second stage of constituent enrichment by network traversal can be further aided by checking an arbitrary set of patterns for preselecting even more features recoverable via lexico-syntactic patterns. The pattern recognition plays an essential role in current parsing method for fleshing out the constituent backbone with systemic selections.

I Reuse the structural account from other theories

Your proposed solution to this problem, and the goal of the thesis, is therefore to add some more structural information to a complete augmented SFG account by drawing on frameworks which have demonstrated coverage of structural detail and which also have supported computational instantiation. This will be shown and evaluated in the thesis.

Reuse structural accounts from other theories that have shown to work well computationally. Could use famous Chomskyan approach but will use it latter for a more specific problem, that of finding covert constituents (give example of null subject in embedded clause). Instead I turn to Dependency Grammar that increasingly gained popularity in the last 15 years. It has been shown to be computationally fast and it is compatible in many ways with SFG which I will show latter how (mapping DG to SFG).

how: (1) by aligning the primitives and drawing cross-theoretical bridges (see section X), (2) then by mapping grammars (see section Y), (3) implemented using graph transformations (see section Z)

II Enrich the constituency structure with features (since the original goal is to have rich language analysis), that in traditional sense are spanning from purely syntactic (such as tense, voice, case, gender, number etc.) or more semantic or even pragmatic (such as semantic role labels, speech acts, sentiment and appraisal and others)

how: (1) by mapping structure to features (SFL Mood) (see section X), (2) by mapping lexical-semantic resources to fragments of syntactic structure and lexis (lexis contextualised by syntactic structure) (SFL Transitivity) (see section Y), (3) implemented by graph pattern matching (see section Z)

So, the thesis goal and outline will be to (and you list them explicitly like this too):
- characterise SFL in its two major variants
- characterise the previous attempts to parse with SFL and their problems
- set out two further linguistic frameworks which (a) have strong accounts of structural relationships, (b) have shown themselves supportive of computational instantiation, and (c) can be shown to exhibit suggestive theoretical/descriptive links with SFG: in particular, DG and GB. Chapter X does this for DG Chapter Y does this for GB.

1.10 Provisional Thesis Structure

1: introduction
2: SFG parsing problem (simple explanation) and SOTA
3: Architecture presentation, introducing challenges of every step and make future references, also the structure of the thesis.
4: SFL grammar
5: Dep Grammar
5: GBT (has to stay here somehow)
7: the structure creation (introducing basic computer scientific definitions of data structures and other needs) (introducing the exact SFG grammar constituents)
8: the syntactic feature enrichment (introducing basic computer scientific definitions of data structures and other needs) (introducing exact Mood features)
9: the semantic feature enrichment (introducing basic computer scientific definitions of data structures and other needs) (introducing exact Cardiff Transitivity features)
10: Empirical Evaluation
11: Conclusions (what has been achieved and outlook)

1.11 References

[Butler2003] Structure and Function [Hjelmslev1953] - Prolegomena-to-a-Theory-of-Language-by-Luis-Hjmeslev [Elke Teich 1999] - Systemic Functional Grammar & Natural Language Generation - Ch5

% feedback Chapter 1 + 2
Dear Eugene,

thanks for file; attached are the detailed comments and corrections and suggestions for Chapters 1 + 2. I suggest some reorganisation of the introduction and how the materials in the current chapter 2 are described, you will sense of this. But, in short, I think an organisation along the lines:

Chapter 1: introduction, reasons and goals
Chapter 2: SFG
Chapter 3: State of the Art in approaches to Parsing with SFG and complexity
Chapter 4: DepGrammar

Chapter 5: GBT (perhaps, haven't read these yet)

would get the thesis off to a better start. Also you need to think about whether all the detail of the SFG variants is important enough for your task. You will need to provide some more detail of the organisation of the actual grammars as well in any case, as otherwise you can't talk about Mood and Transitivity and the like. This is all clarified in the comments. Alternatively you say very little about these and introduce them when you get to the later chapters: that might make sense; I'll see when I get that far. If one went that road, it would mean not including comments about Mood and Transitivity in the current chapter 2 though, which might be awkward.

I'll proceed with the other chapters, but as you will see, you have a fair bit to get going with in any case.

I will not be able to work on the thesis after the end of March.

theses don't really work like that; so we'll see how far you get. You (and I) don't want a repeat of the Daniel situation.

Let me know if anything is unclear.

Best,
John.

%feedback Chapter 1 + 2

Dear Eugene,

Comments/corrections for chapters 3 + 4 attached.

Now I'm getting more of a view of the thesis, I'd say that at present, systemicists will get confused because they'd wonder why alien things like GB and dependency grammar appear, and formal/computational linguists would get

confused because they wouldn't be clear why one would
want to take something like SFL. This can be managed
fairly straightforwardly I suspect by setting up the
argument in the Introduction in a clear way, so that
everyone knows just why these things are coming together.
I'd suggest the following kind of outline for the
introduction to make that work, let me know if you
have any problems or questions about this as it
would seem (to me) to be a good way of making all
the bits fits together in a reasonably convincing
fashion. This would also help avoid a reoccurring problem
in your text at the moment, where you frequently want
to talk about things that you have not yet introduced - this
just makes the text confused and impossible to follow (many
examples of this are picked out explicitly in the comments).

So...

Structure the Intro to the thesis more like this:

First, point to the increasing and increasingly recognized
need for deeper, richer semantic/pragmatic analyses across
a broad range of applications: corpora, human-machine
interaction, intelligent interfaces and assistance robotics,
whatever you can find with references supporting the
claim.

Second, a large amount of description of this kind has
traditionally been done, and done a lot, in SFL. Here again
you need to find a good collection of example 'applications'
in SFL (not computational) where the deeper analysis
has been found useful and give references: education,
text/discourse analysis (critical), whatever plus references.

But, until now it has not been possible to use these
detailed analysis in computational contexts: this makes

them unavailable for corpus work, for training data in
machine learning, etc. etc. (add as many points as occur
to you).

There have been attempts to make this work (which you
will come back to and describe in Chapter X in detail), however,
but these have not worked. As you you say you will describe in detail
in Chapter X, there is however a strong diagnostic as to
just why these attempts have not been successful: i.e., the
lack of structural detail that SFG descriptions typically
provide. This is argued in general in Bateman (2008)
and Teich (1999) [and any other references you can find].

You then give EXAMPLES of some difficult cases, where
you illustrate what an SFG analysis would like look and
you point out the lack of structural detail, informally
so that it can be understood directly without further
technical detail. Preferably bringing out some
cases where it is evident that there is no information,
e.g., about raising and control (Teich) and anything
else which would make interpretation difficult.

Your proposed solution to this problem, and the goal of the thesis,
is therefore to add some more structural information to a
complete augmented SFG account by drawing on frameworks which
have demonstrated coverage of structural detail and which
also have supported computational instantiation. This will
be shown and evaluated in the thesis.

So, the thesis goal and outline will be to (and you list them
explicitly like this too):
- characterise SFL in its two major variants
- characterise the previous attempts to parse with SFL and their problems
- set out two further linguistic frameworks which (a) have
strong accounts of structural relationships, (b) have shown
themselves supportive of computational instantiation, and (c) can

be shown to exhibit suggestive theoretical/descriptive
links with SFG: in particular, DG and GB.

Chapter X does this for DG
Chapter Y does this for GB.

- Following this, Chapter Y+1 brings these altogether in a single
architecture (can be short: material from the current introduction
about the system architecture goes here, or can be longer, if
you take the material about merging GB and DG and then with SFG
here too: this might be best).

- rest of chapters go into details.

- Chapter $-1 Evaluation
- Chapter $ What has been achieved and outlook.

I think this kind of explicit form in the Introduction of
the thesis would tell a convincing story that
would make the most of what you currently have and simply wrap
this in a structure that readers can follow and accept. Then
you strengthen the existing bits of text to explicitly draw
attention to these goals as you go so that the reader
remembers where they are and what you are trying to do (and why).
I think this is a fair bit of work still, but relatively
straightforward as it is more about imposing structure and
getting things in the right order. Definitely a thesis in
there struggling to get out! :-)

Best,
John.

%feedback Chapter 5

Hi Eugen,

here is chapter 5 commented. In this one, there are many more
comments about content that will need fixing up, so not just
style of presentation. Many of the problems though come, I suspect,
because you have not yet introduced the algorithm and pipeline
and its datastructures sufficiently that the reader has any
idea what your formalisations here are attempting to do. I think
many of them can just disappear, since you certainly won't
be able to use them anywhere. To define a data structure, you
don't need a full first-order theory, that is overkill. You
do not get any points for formalisation; you'd only get points
for appropriate, necessary and well motivated formalisation,
and many of the definitions in this chapter do not meet
this requirement. You only need as much formalism as necessary
to get the job done. And the job is the task that you need
to have described as the pipeline of the system: probably best
immediately after the discussion of GB. There are many
interesting decisions made in this chapter, but they are
just lost in the mass of probably hardly relevant detail.
So introducing the pipeline and its data structures first,
would give you a better way of picking out just that which
is a crucial contribution of your thesis, i.e., the stuff
that makes parsing work. Providing definitions of
morphisms between graphs does *not* do that; and it is
hardly your job and has been done more or less completely
before in appropriate formal texts in any case.

In short, you need to provide the new architecture and pipeline
chapter and rewrite this one accordingly.

Let me know when that has happened, as that will be the next
major version that it would be sensible for me to comment on
I think. The actual details of the parsing algorithm that
occurs in subsequent chapters will I hope be more straightforward,
once the groundwork is out of the way.

Best,
John.

Am 08.03.18 um 22:00 schrieb Eugen Costezki:
> I wanted to say that this chapter 5 represented a special kind of struggle as I

yes, I noticed! :-) Fortunately, you do not need to do this...
so simplifications are ahead!

Best,
John

Chapter 2

State of the art review

2.1 Previous works on parsing with Systemic Functional Grammars

There have been various attempts to parsing with SFGs. This section covers the most significant attempts to parse with a Systemic Functional Grammar. The firs attempt was made by Winograd (Winograd 1972) which was more than a parser, it was an interactive a natural language understanding system for manipulating geometric objects in a virtual world.

Starting from early 1980s onwards, Kay, Kasper, O'Donell and Bateman tried to parse with Nigel Grammar (Matthiessen 1985), a large and complex natural language generation (NLG) grammar for English used in Penman generation project. Other attempts by O'Donoghue (1991), Weerasinghe (1994), Souter (1996), Day (2007) aim for corpus based probability driven parsing within the framework of COMMUNAL project starting from late 1980s.

In a very different style, Honnibal (2004); Honnibal & Curran (2007) constructed a system to convert Penn Treebank into a corresponding SFGBank. This managed to provide a good conversion from phrase structure trees into systemic functional representation covering sentence mood and Thematic constituency (a kind of analysis in SFL which is not considered in current work). Transitivity has not been covered there because of its inherently semantic nature but it is in the current work.

2.1.1 Winograd's SHRDLU

SHRDLU is an interactive program for understanding (if limited) natural language written by Terry Winograd at MIT between 1968-1970. It carried a simple dialogue about a world of geometric objects in a virtual world. The human could as the system to manipulate objects of different colours and shapes and the ask questions about what has been done or the new state of the world.

It is recognised as a landmark in natural language understanding demonstrating that a connection with artificial intelligence is possible if not solved. However, his success was not due to the use of SFG syntax but rather due to small sizes of every system component to achieve a fully functional dialogue system. Not only it was parsing the input but it was developing an interpretation of it, reason about it and generate appropriate natural language response.

Winograd combined the parsing and interpretation processes such that the semantic interpreter was actually guiding the parsing process. The knowledge of syntax was encoded in the procedures of interpretation program. He also implemented an ingenious backtracking mechanism where the the program does not simply go back, like other parsers, to try the next possible combination choice but actually takes a decision on what shall be tried next.

Having data embedded into the program procedures, as Winograd did, makes it non-scalable for example in accommodation of larger grammars and knowledge bodies and unmaintainable on the long term as it becomes increasingly difficult to make changes (Weerasinghe 1994).

2.1.2 Kasper

Bob Kasper in 1985 being involved in Penman generation project embarked on the mission of testing if the Nigel grammar, then the largest available generation grammar, was suitable for natural language parsing. Being familiar with Functional Unification Grammar (FUG), a formalism developed by Kay and tested in parsing (Kay 1985) which caught on popularity in computational linguistics regardless of Kay's dissatisfaction with results, Kasper decided to re-represent Nigel grammar into FUG.

Faced with tremendous computational complexity, Kasper (1988) decided to manually create the phrase-structure of the sentences with hand-written rules which were mapped onto a parallel systemic tree structure. Kasper in 1988 was the first one to parse with a context-free backbone. He first parsed each sentence with a Phrase Structure Grammar (PSG), typical to Chomsky's Generative Transformational Linguistics

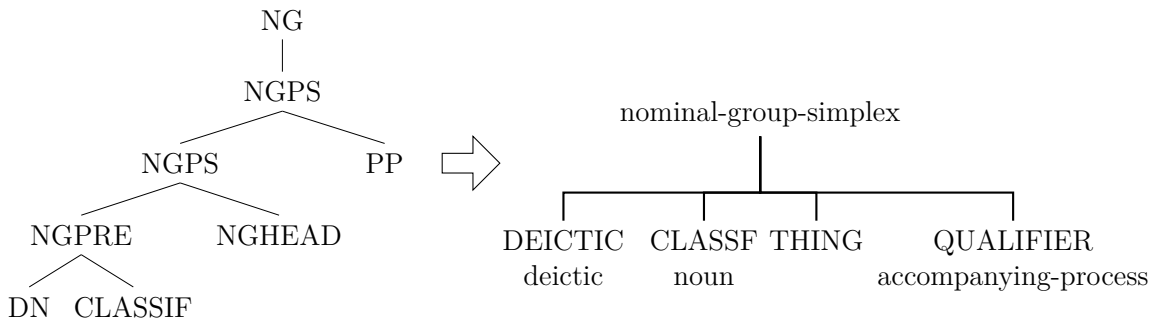


Fig. 2.1 Transformation from phrase structure into systemic constituency structure. Rule example from O'Donnell & Bateman (2005).

Chomsky (1957). He created a set of rules for mapping the phrase structure (PS) into a parallel systemic tree like the one depicted in Figure 2.1. When all possible systemic tree were created they were further enriched using information from Nigel Grammar (Matthiessen 1985).

Once the context-free phrase-structure was created using bottom-up chart parser it was further enriched from the FUG representation of Nigel grammar. This approach to parsing is called *parsing with a context-free backbone* as phrase-structure is conveyed as simplistic skeletal analysis, fleshed out by the detail rich systemic functional grammar.

Even though Kasper's system is represents the first attempt to parse with full Hallidayan grammar, it's importance is lowered, as O'Donnell & Bateman (2005) point out, by the reliance on phrase structure grammar.

2.1.3 O'Donnell

Since 1990, Mick O'Donnell experimented with several parsers for small Systemic grammars, but found difficulty when scaling up to larger grammars. While working in EAD project, funded by Fujitsu, he recompiled a subset of Nigel grammar into two resources: the set of possible function bundles allowed by the grammar (along with the bundles preselections) and a resource detailing which functions can follow a particular function (O'Donnell 1993, 1994).

This parser was operating without a syntactic backbone directly from a reasonable scale SFG. However when scaled to the whole Nigel grammar the system became very slow because of the sheer size of the grammar and its inherent complexity introduced by multiple parallel classifications and functional combinations - a problem well described by Bateman (2008). Then O'Donnell wrote his own grammar of Mood that was more suitable for the parsing process and less complex than the recompiled Nigel.

In 2001, while working in a Belgian company O'Donnell came to conclusion that dependency grammars are very efficient for parsing. Together with two colleagues, he developed a simplified systemic grammar where elements were connected through a single function hence avoiding (functional) conflation. Also the ordering of elements was specified relative to the head rather than relative to each other.

More recently, O'Donnell in UAM Corpus Tool embedded a systemic chart parser (O'Donnell 2005) with a reduced systemic formalism. He classifies his parser as a left to right and bottom up with a custom lexicon where verbs are attributed features similar to Hallidayan process types and nouns a unique semantic category like thing-noun, event-noun, location-noun etc.

Because of previously reported complexity problems (O'Donnell 1993) with systemic grammars, the grammatical formalism is reduced to a singular functional layer of Mood-based syntactic structure (Subject, Predicate, Object etc.) ignoring the Transitivity (Actor/Goal, Sensor/Phenomenon etc.) and Textual (Theme/Rheme) analyses. O'Donnell deals away with the conflation except for the verbal group system network. He also employs a slot based ordering where elements do not relate to each other but rather to the group head only simplifying the number of rules and calculation complexity.

In his paper (O'Donnell 2005) does not provide a parser evaluation so its accuracy is still unknown today. The lexicon that was created is claimed to deal with word semantic classes but it is strongly syntactically based assigning a single sense to nouns and verbs ignoring the peculiar aspect of language polysemy. Moreover it is not very clear the framework within which the semantic classes have been generated.

2.1.4 O'Donoghue

O'Donoghue proposes a corpus based approach to parsing using *Vertical Strips* (O'Donoghue 1991). They are defined as a vertical path of nodes in a parse tree starting from the root down to the lexical items but not including those. He extracted the set of vertical strips from a corpus called Prototype Grammar Corpus together with their frequencies and probability of occurrence. This approach differ from the traditional one with respect to the kind of generalization it is concerned and specifically, the traditional approached are oriented towards horizontal order while the vertical strip approach is concerned with vertical order in the parse tree.

To solve the order problem O'Donoghue uses a set of probabilistic collocation rules extracted from the same corpus indicating which strips can follow a particular strip. He

also created a lexical resource indicating for each word which elements can expanded it.

The parsing procedure is a simple lookup of words in the lexical resource selecting all possible elements it can expound and then selecting possible strips starting with the elements expounded by the word. Advancing from left to right for each sentence word more strips compatible with the previously selected ones are selected within the collocation network constrains. The parser finds all possible combinations of strips composing parse trees representing possible output parses.

The corpus from which the vertical strips were extracted is 100,000 sentences large and was generated with Fawcett's natural language generation system and was tested on the same corpus leaving unclear how would the parser behave on a real corpus. In 98% of cases the parser returns a set of trees (between 0 and 56) that included the correct one with an average of 6.6 trees per parse.

Actually, using a larger corpus could potentially lead to a combinatorial explosion in the step that looks for vertical strips. It would decrease the accuracy of the parse because of the higher number of possible trees per parse.

2.1.5 Honnibal

Honnibal (2004; 2007) describes how Penn Treebank can be converted into a SFG Treebank. Before assigning to parse tree nodes synthetic features such as mood, tense, voice and negation he first transforms the parse trees into a form that facilitates the feature extraction.

The scope of SFG corpus was limited to a few Mood and Textual systems leaving aside Transitivity because of its inherently lexico-semantic nature. He briefly describes how he structurally deals with verb groups, complexes and ellipses as functional structures are much flatter than those exhibited in the original Treebank. Then he describes how are identified metafunctional features of unit class, mood function, clause status, mood type, polarity, tense, voice and textual functions.

The drawback of his approach is that the Python script performing the transformation does not derive any grammar but rather implements directly these transformations as functions falling into the same class of problems like Winograd's SHRDLU. By doing so the program is non-scalable for example in accommodation of larger grammars and knowledge bodies and unmaintainable on the long term as it becomes increasingly difficult to make changes.

Chapter 3

Parsing Architecture

Chapter 4

Government and binding theory

Transitivity analysis in SFL is similar to what *semantic role labelling*, *thematic* or θ *role analysis* means in other theories. This thesis provides, in Chapter **??**, an account of how to perform SF Transitivity parsing resulting in a configuration of a process, participants and circumstances. For an illustration take Example 5 whose Transitivity analysis is available in Table 4.1. Here the entire clause is analysed as a Possessive configuration governed by the verb "receive" where "Albert" plays the *role* of the Affected-Carrier and "a phone call" is the thing being Possessed.

(5) Albert received a phone call.

(6) He asked to go home immediately.

Albert	*received*	*a*	*phone*	*call*
Possessive configuration				
Affected Carrier	Process	Posessed		

Table 4.1 Transitivity analysis with Cardiff grammar of Example 5

Example 6 is slightly more complex and illustrates the main motivation behind the current chapter. It is analysed in Table 4.2, according to Cardiff grammar, as a Three Role Cognition configuration with "ask" being the process, "he" the Agent and "to go home immediately" the cognised Phenomenon. The Phenomenon is filled by a non-finite clause "to go home immediately" which is, in Transitivity account, a Directional configuration governed by the verb "go" and has as participants the Destination "home" and the Agent Carrier in an empty Subject position that is said to be *non-realised*, *empty*, or *covert*. This is a case when the empty constituent is recoverable from the clause above and corresponds to the Subject "He". This way,

the constituent “He” plays two roles: first as Agent in the Cognition process of the top clause and second as Agent Carrier in the Directional process of the embedded clause. In this work, the way to assign a second role coming from the lower clause, is by detecting and making explicit the empty constituents and resolving them locally with a link to the corresponding constituent.

<table>
<tr><td>He</td><td>asked</td><td>[empty subject]</td><td>to</td><td>go</td><td>home</td><td>immediately</td></tr>
<tr><td colspan="7">Three Role Cognition configuration</td></tr>
<tr><td>Agent</td><td>Process</td><td colspan="5">Phenomena</td></tr>
<tr><td colspan="2"></td><td colspan="5">Directional Configuration</td></tr>
<tr><td colspan="2"></td><td>Agent-Carrier</td><td></td><td>Process</td><td>Destination</td><td></td></tr>
</table>

Table 4.2 Transitivity analysis with Cardiff grammar of Example 6

In language there are many cases where constituents are empty but recoverable from the immediate vicinity relying in most cases on syntactic means and in a few cases additional lexical-semantic resources are required. The mechanisms of detecting and resolving the empty constituents are captured in the Government and Binding Theory (GBT) developed in (Chomsky 1981, 1982, 1986) and based on the phrase structure grammar. GBT explains how some constituents can *move* from one place to another, where are the places of *non-overt constituents* and what constituents do they refer to i.e. what are their *antecedents*.

The GBT approach explains grammatical phenomena using *phrase structures* (PS). This is more distant from SFG than the approach taken by the dependency grammar. Section 4.2 briefly introduces the theoretical context of GBT and then formulates the principles and generalisations relevant for current work. Then Section 4.3 translates the introduced principles and generalisations into Dependency Grammar rules and patterns. To lay the ground for the two sections, I first place GBT into the context of transformational grammar and introduce the basic concepts.

4.1 Introduction to GBT

This section is set as introduction to the fundamental concepts from Government and Binding Theory. It belongs to the family of Transformational grammars (TG) or transformational-generative grammars (TGG). It is part of the theory of generative grammar that considers grammar to be a system of rules that generate exactly those combinations of words which form grammatical sentences in a given language (Chomsky

1965). TG involves the use of defined operations called transformations to produce new sentences from existing ones.

Chomsky developed a formal theory of grammar (Chomsky 1956) where transformations manipulated not just the surface strings, but the parse tree associated with them, making transformational grammar a system of tree automata (Stockwell et al. 1973).

A transformational-generative (or simply transformational) grammar thus involved two types of productive rules: *phrase structure rules*, such as "S -> NP VP" (meaning that a sentence may consist of a noun phrase followed by a verb phrase) etc., which could be used to generate grammatical sentences with associated parse trees (phrase markers, or P markers); and *transformational rules*, such as rules for converting statements to questions or active to passive voice, which acted on the phrase markers to produce further grammatically correct sentences (Bach 1966: 59-66). This notion of transformation proved adequate for subsequent versions including the "extended", "revised extended" and Government-Binding (GB) versions of generative grammar, but may no longer be sufficient for the latest "minimalist" grammar **??**. It require a formal definition that goes beyond the tree manipulation. For the purpose of the current work, however, the GBT employing the idea of transformations is perfectly suitable. I selected it because of clear and extensive descriptions of the mechanisms for identification of *null elements* (also known as *empty categories*) and how to provide them with an interpretation.

4.1.1 Phrase structure

The notion of structure in a generative grammar refers to the way words are combined together to form phrases and sentences. *Merging* is the technical term used in GBT for the operation of bringing two words together into a phrase. In this operation one word will always be more prominent and is therefore called the *head* of the phrase. The resulting combination is a new constituent and is called a *projection* of the head. This is known as *X-bar theory* (often denoted as X' or $\bar{X}$) and embodies two primary claims: (a) that the phrases may contain intermediary constituents projected from a head X and (b) that this system of projected constituency may be common to more than one category (such as N, V, A, P etc.).

These combinations of words and projections can be represented using the *labelled bracketing notation* where the labels denote constituent categories. The bracketed notation is a representation equivalent to a hierarchical tree of constituent parts or *parse tree* (also known as *syntactic tree*, *phrase structure*, *derivation tree*). The parse

tree represents the syntactic structure of a string according to some grammar. The equivalence between a bracketed notation and parse tree is exemplified in the following two representations of 7 from (Haegeman 1991: 83).

(7) Poirot will abandon the investigation.

(8) $\Big[_{S}\Big[_{NP}\big[_{N}Poirot\big]\Big]\big[_{AUX}will\big]\Big[_{VP}\big[_{V}abandon\big]\Big[_{NP}\big[_{Det}the\big]\big[_{N}investigation\big]\Big]\Big]\Big]$

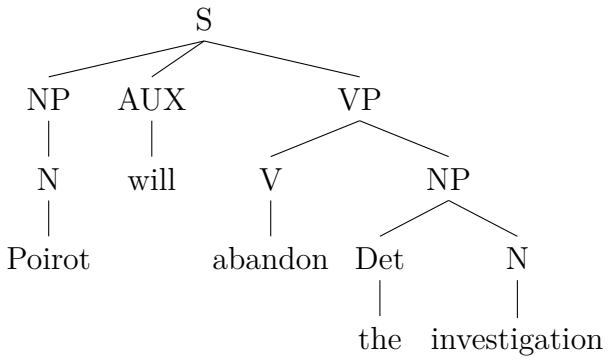


Fig. 4.1 The parse tree of Example 7 from (Haegeman 1991: 83)

A node is said to be *non-branching* if there is a single line starting below and it is called *branching* if there are more than one line going downwards. The children of a branching node are said to be bound by a *sisterhood* relation and in relation to a *parent* or *mother* node. In a phrase structure the vertical relations are referred as *dominance* relations defined below.

Definition 4.1.1 (Dominance)**.** Node A dominates node B if and only if A is higher up in the tree than B and if you can trace a line from A to B going only downwards (Haegeman 1991: 85).

Looking at the tree diagram along the horizontal axis, GBT describes left-to-right ordering of constituents using the *precedence* relation.

Definition 4.1.2 (Precedence)**.** Node A precedes node B if and only if A is to the left of B and neither A dominates B nor B dominates A (Haegeman 1991: 85).

In Figure 4.1 NP, AUX and VP nodes are sisters, they precede each other and are dominated by S parent node. A more specific type of dominance, that will be employed latter in this chapter, is the *immediate dominance* which is when there is no intermediary node between A and B. In this case, the node "Poirot" is also dominated

by S but only the grandparent NP is immediately dominated by S. The same holds for precedence: the *immediate precedence* is when a node A precedes a node B and there is no intervening node in between. Node NP precedes VP but only AUX is immediately preceded.

Generalization 4.1.1 (Projection principle)**.** Lexical information is syntactically represented.

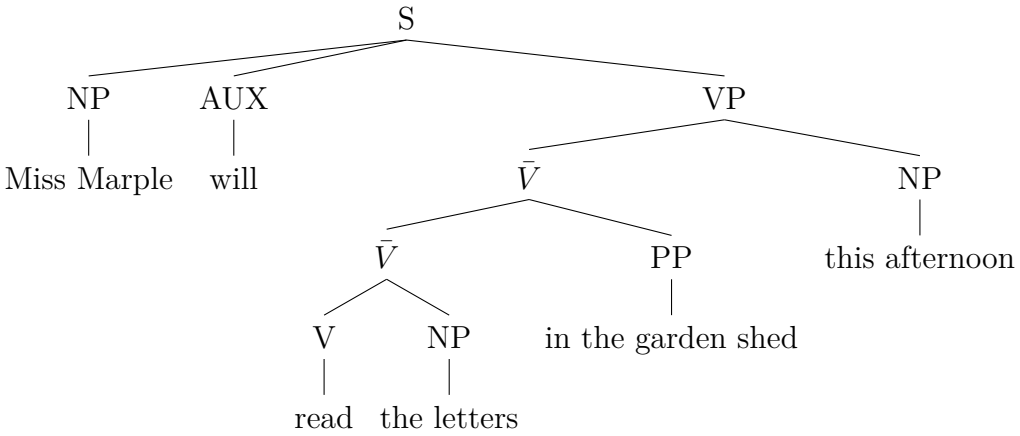


Fig. 4.2 Example of projections from (Haegeman 1991: 90)

An important principle in GBT is that of *projection* formulated in Generalisation 4.1.1. For example in Figure 4.2, projections of V that are dominated by more comprehensive projections of V are called *intermediate projections* while the node labelled VP is the *maximal projection* of V. Maximal projections are also barrier to government (see Definition **??** below). The role of lexicon in syntax from to GBT perspective is discussed at large in Stowell & Wehrli (1992).

4.1.2 Theta theory

This section introduces which constituents are minimally required to form a sentence and why. Traditionally three types of verbs are recognised: *transitive*, *di-transitive* and *intransitive*. This distinction is based on how many complements a verb requires to form a minimal complete sentence. If a verb is transitive then one NP direct object is required. If the verb is di-transitive then two NP or one NP and one PP direct and indirect objects are required. Finally, if it is intransitive then no NP complement is allowed.

Logicians for a long time have been concerned with formulating representations corresponding to semantic structure of sentences or *propositions*. Like Tesniere (Tes-

niere 2015: 97) discussed in Section **??**, Haegeman employs the metaphor of a theatre play when discussing the argument structure of predicates. A play not only describes the number of participants but also what corresponding roles they play. The specific semantic relations between the verb and its arguments is comparable with the identification of characters in a play script (Haegeman 1991: 49).

In logical notation such as in Example 9 a proposition comprises of a predicate (P) that takes a certain number of *arguments* (here a and b). By analogy to the logical tradition, in GBT, the verb is said to be like the predicate while the subject together with complements are like the arguments that the predicate requires.

In Example 10 Maigret, taking subject position, is the Agent in the process of killing while Poirot in the complement position is the Patient that receives the effects of the process of killing. The generic argument structure for the verb "to kill" can be expressed as in Example 11. The first argument is of NP category and takes the role of an Agent while the second argument is also an NP but it takes the Patient role. The transitivity of a verb dictates how many arguments there should be.

(9) P(a, b)

(10) Maigret killed Poirot.

(11) V kill: 1 (NP:Agent), 2 (NP:Patient)

In literature these relations between the verb and the arguments are called *thematic roles* or theta-roles (θ-roles). It is said that the verb *theta marks* its arguments. The component of the grammar that regulates the assignment of thematic roles is called *theta theory*.

In GBT the theory of thematic roles is very sketchy and does not go beyond distinction of several thematic roles (Agent/Actor, Patient, Theme, Experiencer, Beneficiary, Goal, Source, Location and the controversial Theme) (Haegeman 1991: 50). The theta theory has a central criterion that is stipulated in Generalisation **??**.

Generalization 4.1.2 (Theta criterion)**.** Theta criterion requires that:

- each argument is associated one and only one theta role
- each theta role is assigned to one and only one argument (Haegeman 1991: 54)

(12) *It* surprised Jeeves that the pig had been stolen.

In English, however, there is a special case, that of *expletives*, when the subject argument is filled by the pronoun *it* that receives no thematic role and acts rather as a

dummy slot filler without any semantic contribution to the meaning of the sentence (Haegeman 1991: 62). Worth noticing is also the fact that *auxiliary verbs* and *copula verbs* do not assign thematic roles (Pollock 1989).

The verb that assigns a theta role does not need to specify which syntactic category it shall be realised by. In more technical it means that the categorial selection (*c-selection*) follows from semantic relation (*s-selection*). When a theta role can be assigned to an argument it is said that it is saturated. In order to identify the assignment of respective role the arguments are identified by the means of an index provided as subscript in the sentence.

(13) Maigret$_i$ killed the burglar$_j$.

(14) Maigret$_i$ said that he$_i$ was ill.

In the Example 13 Maigret has the index i and the burglar j meaning they are distinct referents. On contrary, in Example 14, "he" receives the same index as Maigret because they are interpreted as referring to the same entity. We say that the two are *coindexed*.

4.1.3 Government and Binding

Using the terminology from the traditional grammar it is said that a verb governs its object. This is generalised in GBT as a rule that the head of a phrase, called *governor*, *governs* its complement, called the *governee*. This relation is loosely defined in Definition 4.1.3 below and formally in Definition 4.1.5.

Definition 4.1.3 (government i)**.** A governs B if

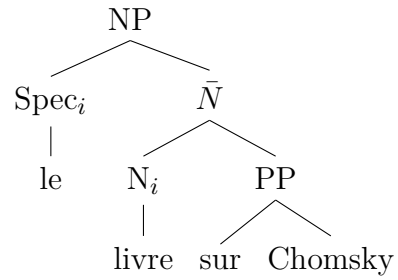
- A is a governor;
- A and B are sisters

Governors are heads (Haegeman 1991: 86).

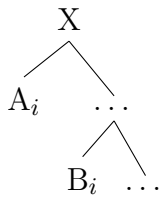
In Figure 4.1 the verb "abandon" is the head of the verb phrase (VP) and governs the direct object - nominal phrase (NP) "the investigation". V does not govern the subject NP "Poirot". All the constituents governed by a node constitute the *governing domain* of that node. In this case VP is the governing domain of V.

Before providing the next definition of government, I first introduce the notion of C-command which provides a general pattern of how the agreeing elements relate to each other in the parse tree. C-command is formally defined in Definition 4.1.4.

When considering the geometrical relation between the agreeing elements, one always is higher in the tree than the other one in a manner depicted in Figure 4.3b. In Figure 4.3a the co-subscripted nodes indicate agreement. Here the [Spec, NP] c-commands all the nodes dominated by the NP. These nodes constitute the *c-command domain* of Spec element (Haegeman 1991: 134).



(a) Example of agreement in French (Haegeman 1991: 132)



(b) Schematic representation of the c-command (Haegeman 1991: 133)

Fig. 4.3 Agreement example and schematic representation

Definition 4.1.4 (c-command)**.** A node A c-commands a node B if and only if

- A does not dominate B;
- B does not dominate A;
- the first branching node dominating A also dominates B (Haegeman 1991: 212).

Definition 4.1.5 (Government)**.** X governs Y if and only if

- X is either of the category A, N, V, P, I;
  or
  X and Y are coindexed
- X c-commands Y;
- no barrier intervenes between X and Y;
- there is no Z such that Z satisfies the points above and X c-commands Z (Haegeman 1991: 557).

IN GBT three types of NP are distinguished: *full noun phrases* (e.g. Maigret, the doctor, etc.), *pronouns* (e.g. he, me, us, etc.), and *anaphors* comprised of reflexives (e.g. myself, herself, etc.) plus reciprocals (e.g. each other, one another,). Pronouns

and anaphors (reflexives and referential) lack inherent reference. Anaphors need an antecedent for their interpretations whereas pronouns do not. Pronoun indicate some inherent features of the referent so that they can be identified from the contextual information. The full noun phrases called Referential expressions or *R-expression*, for short, are inherently referential and do not need an antecedent, moreover they do not tolerate an antecedent (Haegeman 1991: 226). The NP types can be defined in terms of features Anaphor and Pronominal (systematised together with the empty categories in the Table 4.3 below). This way the Pronouns have features [+Pronominal,-Anaphor], the Anaphors [+Pronominal,-Anaphor] and the R-expressions [-Pronominal,-Anaphor]. The last combination [+Pronominal,+Anaphor] corresponds to *PRO empty category* which will be discussed in the Section 4.2.1 coming up next.

The module of the grammar regulating interpretation of the noun phrase (NP) interpretation is referred, in GBT, as the *binding theory*. It is formally defined in terms of c-command in Definition 4.1.6. And because the BT is essentially concerned with binding of NPs in argument positions (*A-position*) then it is rather the **A-binding** (see Definition 4.1.7) of interest here. An A-position is a position in the tree to which a theta role can (but not necessarily) be assigned (Haegeman 1991: 115).

Definition 4.1.6 (Binding)**.** A binds B if and only if

- A c-commands B;
- A and B are coindexed (Haegeman 1991: 212).

Definition 4.1.7 (A-Binding)**.** A A-binds B if and only if

- A is in A-position;
- A c-commands B;
- A and B are coindexed (Haegeman 1991: 240).

Each of these NP types have an associated binding principle (ways in which to interpret, if needed, the reference of the NP) provided in Generalisations 4.1.3, 4.1.4 and 4.1.5 below. These principle use the idea of *governing category* which for a node A is the minimal domain containing it, its governor and an accessible subject. A subject A is said to be accessible for B if the co-indexation of A and B does not violate any grammatical principle (Haegeman 1991: 241).

Generalization 4.1.3 (Principle A of binding theory)**.** An anaphor (i.e. a NP with the feature [+Anaphor] covering reflexives and reciprocals) must be bound in its governing category (Haegeman 1991: 224).

Generalization 4.1.4 (Principle B of binding theory)**.** The pronoun (i.e. a NP with feature [+Pronominal]) must be free in its governing category (Haegeman 1991: 225).

Generalization 4.1.5 (Principle C of binding theory)**.** An R-expression (i.e. a NP with independent reference) must be free everywhere (Haegeman 1991: 227).

4.2 On Null Elements

In certain schools of linguistics, in the study of syntax, an *empty category* is a nominal element that does not have any phonological content and is therefore unpronounced. Empty categories may also be referred to as *covert nouns*, in contrast to overt nouns which are pronounced (Chomsky 1993). Some empty categories are governed by the *empty category principle* (see Definition 4.2.1). When representing empty categories in trees, linguists use a null symbol to depict the idea that there is a mental category at the level being represented, even if the word(s) are being left out of overt speech.

GBT recognises four main types of empty categories: *NP-trace*, *Wh-trace*, *PRO*, and *pro*. They are subject to Principles A, B and C of the binding theory provided above and differentiated, like the over NPs, by two binding features: the anaphoric feature [a] and the pronominal feature [p]. The four possible combinations of plus (+) or minus (-) values for these features yield four types of empty categories.

[a]	[p]	Symbol	Name of the empty category	Corresponding overt NP type
-	-	t	Wh-trace	R-expression
-	+	pro	little Pro	pronoun
+	-	t	NP-trace	anaphor
+	+	PRO	big Pro	none

Table 4.3 Four types of empty categories (adaptation from (Haegeman 1991: 436))

In Table 4.3, [+a] refers to the anaphoric feature, meaning that the particular element must be bound within its governing category whereas [+p] refers to the pronominal feature which shows that the empty category is taking the place of an overt pronoun.

Definition 4.2.1 (Empty Category Principle (ECP))**.**

- Traces must eb properly governed.
- A properly governs B if and only if A theta-governs B or A antecedent-governs B (Chomsky 1986: 17).

- A theta-governs B if and only if A governs B and A theta-marks B.
- A antecedent-governs B if and only if A governs B and A is coindexed with B (Haegeman 1991: 442).

Next I describe in detail each empty category and the properties of corresponding overt noun type.

4.2.1 PRO Subjects and control theory

PRO stands for the non-overt NP that is the subject in non-finite (complement, adjunct or subject) clause and is accounted by the *control theory* (CT).

Definition 4.2.2 (Control)**.** Control is a term used to refer to a relation of referential dependency between an unexpressed subject (the control element) and an expressed or unexpressed constituent (controller). The referential properties of the controlled element are determined by those of the controller (Bresnan 1982).

Control can be *optional* or *obligatory*. While *Obligatory control* has a single interpretation, that of PRO being bound to its controller, the *optional control* allows for two interpretations: *bound* or *free*. In Example 15 the PRO is controlled, thus bound, by the subject "John" of the matrix clause (i.e. higher clause) whereas in 16 it is an arbitrary interpretation where PRO refers to "oneself" or "himself". In 17 and 18 PRO must be controlled by the subject of the higher clause and does not allow for the arbitrary interpretation.

(15) John asked how [PRO to behave himself/oneself].

(16) John and Bill discussed [PRO behaving oneself/themselves in public].

(17) John tried [PRO to behave himself/*oneself].

(18) John told Mary [PRO to behave herself/*himself/*oneself].

Sometimes the controller is the subject (as in Examples 15, 16, 17) and sometimes it is the object (Example 18) of the higher clause. Haegeman (1991: 278) proposes that there are two types of verbs, verbs of *subject* and of *object control*. The following set of generalizations from Haegeman (1991) are instrumental in identifying places where a PRO constituent can be said to occur and identifies its corresponding binding element.

Generalization 4.2.1. Each clause has a subject. If a clause doesn't have an overt subject then it is covertly (non-overtly) represented as PRO (Haegeman 1991: 263).

Generalization 4.2.2. A PRO subject can be bound, i.e. it takes a specific referent or can be arbitrary (equivalent to pronoun "one") (Haegeman 1991: 263). In case of obligatory control, a PRO subject is bound to a NP and must be c-commanded by its controller (Haegeman 1991: 278).

Generalization 4.2.3. PRO must be in ungoverned position. This means that (a) PRO does not occur in object position (b) PRO cannot be subject of a finite clause (Haegeman 1991: 279).

Generalization 4.2.4. PRO does not occur in the non-finite clauses introduced by *if* and *for* complementizers, but it can occur in those introduced by *whether* (Haegeman 1991: 279).

Examples 19 and 20 illustrate Generalisation 4.2.4

(19) John doesn't know [whether [PRO to leave].

(20) * John doesn't know [if PRO to leave].

Generalization 4.2.5. PRO can be subject of complement, subject and adjunct clauses (Haegeman 1991: 278).

Generalization 4.2.6. When PRO is the subject of a declarative complement clause it must be controlled by an NP, i.e. arbitrary interpretation is excluded (Haegeman 1991: 280).

Generalization 4.2.7. The object of active clause becomes subject when it is passivized and also controls the PRO element in complement clause (Haegeman 1991: 281).

Generalization 4.2.8. PRO is obligatorily controlled in adjunct clauses that are not introduced by a marker (Haegeman 1991: 283).

Adjuncts (clauses or phrases) often are introduced via prepositions. Nonetheless there are rare cases of adjunct clauses free of preposition. Examples 21 and 22 illustrate such marker-free adjunct clauses.

(21) John hired Mary [PRO to fire Bill].

(22) John abandoned the investigation [PRO to save money].

Generalization 4.2.9. PRO in a subject clause is optionally controlled; thus by default it takes arbitrary interpretation (Haegeman 1991: 283).

(23) PRO_i smoking is bad for the health_j.

(24) PRO_i smoking is bad for your_i health_j.

(25) PRO_i smoking is bad for you_i.

(26) PRO_i lying to your_i friends decreases your_i trustworthiness_j.

A default assumption is to assign arbitrary "one" interpretation to each PRO subject in subject clauses. However, there are cases when it may be bound (resolved) to a pronominal NP in the complement of the higher clause. The binding element can be either the entire complement or a *pronominal* part of it like the qualifier or the possessor. Example 23 illustrates that PRO has only arbitrary interpretation since it cannot be bound to the complement "health". Moreover PRO can also be bound to (a) the possessive element of a higher clause - example 24, (b) the complement of the higher clause - example 25 and (c) either the possessives in lower or higher clause, Example 26.

4.2.2 NP-traces

In GBT, *movement* is a kind of transformation used to explain discontinuity or displacement phenomena in language. It is based on the idea that some constituents appear to have been displaced from the position where they receive important features of interpretation.

GBT distinguishes three types of movement: (a) *head-movement* - the movement of auxiliaries from I to C , *Wh-movement* - when the wh-constituent lands in Spec position of a CP (i.e. [Spec, CP]) and (c) NP-movement when a NP is moved into an empty subject position. NP-movement in GB theory is used to explain *passivization*, *subject movement* (in interrogatives) and *raising*. The raising phenomenon (Definition 4.2.4) is the one that is of interest for us here as it is the one involving an empty constituent.

When an NP moves it is said to leave *traces* (Definition 4.2.3). The moved constituent is called *antecedent* of a trace. Both the trace(s) and the antecedent are coindexed and form what is called a *chain* (Haegeman 1991: 309).

Definition 4.2.3 (Trace)**.** A trace is an empty category which encodes the base position of a moved constituent and is indicated as t (Haegeman 1991: 309).

Definition 4.2.4 (NP-raising)**.** NP-raising is the NP-movement of a subject of a lower clause into subject position of a higher clause (Haegeman 1991: 306).

Consider Examples 27 to 30 where I used square brackets to indicate boundaries of an embedded clause. There are two cases of expletives (27 and 29) and their non-expletive counterparts (28 and 30) where the subject of the lower clause is moved to the subject position of the matrix clause by replacing the expletive. The movement of NPs are described in GB as leaving traces which here are marked as *t*. This phenomena is called *raising* (Definition 4.2.4) or as Postal (1974) calls it the *subject-to-subject raising*.

(27) It was believed [Poirot to have destroyed the evidence].

(28) Poirot$_i$ was believed [t$_i$ to have destroyed the evidence].

(29) It seems [that Poirot has destroyed the evidence].

(30) Poirot$_i$ seems [t$_i$ to have destroyed the evidence].

The subjects "It" and "Poirot" in none of the examples 27–30 receive a semantic role from the main clause. "It" is an expletive and never receives a thematic role while "Poirot" in 28 and 30 takes an Agent role from "destroy" and is not the Experiencer neither of "believe" nor of "seem". So verbs "believe" and "seem" do not theta mark their subjects in these examples.

Raising is very similar to obligatory subject control with a difference in thematic role distribution. In the case of subject control, both the PRO element and it's binder (the subject of higher clause) receive thematic roles in both clauses. However in the case of raising, the NP is moved and it leaves a trace which is theta marked but not to the antecedent (Haegeman 1991: 314). This is expressed in generalization 4.2.10.

Generalization 4.2.10. The landing site for a moved NP is an empty A-position. The chain formed by a NP-movement is assigned only one theta role and it is assigned on the foot of the chain, i.e. the lowest trace (Haegeman 1991: 314).

So, in case of raising, the landing sites for a moved NP are empty subject positions or the ones for expletives. As a result of movement, these positions are filled or expletives are replaced with the moved NP. The last type of movement is the one of NPs that have a *wh-element* described in the next section.

4.2.3 Wh-traces

Wh-movement is involved in formation of Wh-interrogatives and in formulation of relative clauses. We are interested in both cases as both of them leave traces of empty elements that are relevant for transitivity analysis.

(31) [What] will Poirot eat?

(32) [Which detective] will Lord Emsworth invite?

(33) [Whose pigs] must Wooster feed?

(34) [When] will detective arrive at the castle?

(35) [In which folder] does Margaret keep the letter?

(36) [How] will Jeeves feed the pigs?

(37) [How big] will the reward be?

Haegeman (1991: 375) offers the Examples 31 – 37 of *Wh-constituents* which are any NPs or PPs that contain a *Wh-word* in their componence. Thus the wh-constituent can be a single word or a *Wh-phrase*. The Wh-phrase is then the NP or PP which is the maximal projection from a Wh-word. She treats each Wh-word as the head of the Wh-phrase and as we will see in Section 4.3.3 below, I provide a different definition of Wh-group such that it is congruent with the Systemic Functional Grammars.

In GBT the *case* occupies an important place in the grammar. The rules governing the case are known as *case theory*. Verbs dictate the case of their arguments, a property called *case marking*. The Subjects are marked with Nominative case while the Complements of the verb are marked with Accusative case.

In English, however, case system is very rudimentary as compared to other languages. Hence, *who*, *whom* and their derivatives *whoever* and *whomever* are the only Wh-elements with overt case differentiation. The other Wh-elements *what, when, where* and *how* do not change their form based on the case.

Example 38 shows that the Accusative (*whom*) is disallowed when its trace is in Subject position since this requires Nominative (*who*). In example 39 the reverse holds and the Nominative form is disallowed as the Wh-element moved from Complement position requires Accusative case.

(38) Who$_i$/*Whom$_i$ do you think [t$_i$ will arrive first?]

(39) Whom$_i$/*Who$_i$ do you think [Lord Emsworth will invite t$_i$?]

Another important distinction to be made among English Wh-elements is the *theta-marking*, i.e. the argument and non-argument distinction. Some wh-constituents will be in A-positions (i.e. functioning as subject or complement) such as in Example 31 – 33 or in non A-position (i.e. functioning as adjunct) such as in Example **??** and 36.

When the Wh-constituent moves, There are two places where it can land: (a) either in the subject position of the matrix clause changing its mood to interrogative (example

40) or (b) subject position of the embedded clause creating embedded questions (example 41). However regardless of the landing site, the movement principle is subject to what Haegeman describes as *that-trace filter* expressed in Generalisation 4.2.11 and the *Subjacency condition* (Generalisation 4.2.12). Note that the matrix or embedded clauses correspond to the category of inflectional phrase (IP).

(40) Whom$_i$ do [you believe [that Lord Emsworth will invite t$_i$]]?

(41) I wonder [whom$_i$ you believe [that Lord Emsworth will invite t$_i$]].

Generalization 4.2.11 (That-trace filter)**.** The sequence of an overt complementizer "that" followed by a trace is ungrammatical (Haegeman 1991: 399).

The examples in 42 to 45 provided by Haegeman (1991: 398) illustrate how the above generalization applies.

(42) * Whom do you think that Lord Emsworth will invite?

(43) Whom do you think Lord Emsworth will invite?

(44) * Who do you think that will arrive first?

(45) Who do you think will arrive first?

Generalization 4.2.12 (Subjacency condition)**.** Movement cannot cross more than one bounding node, where bounding nodes are IP and NP (Haegeman 1991: 402).

The Subjacency condition captures the grammaticality of NP-movement and exposes two properties of the movement, namely as being *successive* and *cyclic*. Consider the chain creation resulting from Wh-movement in Examples 46 – 48 provided by Haegeman (1991: 403–406). The Wh-movement leaves (intermediate) traces successively jumping each bounding node.

(46) Who$_i$ did [he see t$_i$ last week?]

(47) Who$_i$ did [Poirot claim [t$_i$ that he saw t$_i$ last week?]]

(48) Who$_i$ did [Poirot say [t$_i$ that he thinks [t$_i$ he saw t$_i$ last week?]]]

What Generalisation 4.2.12 states is that a Wh-constituent cannot move further than subject position of the clause forming an interrogative form. Or also it can move outside into the subject position of the clause higher above leaving a Wh-trace as can be seen in the Example 47 and 48.

Now that the kinds of null elements have been concisely described laying out the main rules governing their behaviour, I turn next to discuss what how these elements can be identified in terms of Dependency grammar.

4.3 Placing Null Elements into the Stanford dependency grammar

This section provides a selective translation of principles, rules and generalisations captured in GB theory into the context of dependency grammar. The selections mainly address the identification of places where (and by which relations) the null elements should be injected into dependency structure that will later help the semantic parsing process described in Chapter **??**.

4.3.1 PRO subject

Coming back to definition of PRO element in Section 4.2.1, it is strictly framed by the non-finite subordinate clauses. In dependency grammar the non-finite complement clauses are typically linked to their parent via *xcomp* relation which is defined in Marneffe & Manning (2008a) as introducing an open clausal complement of a VP or ADJP without its own subject, whose reference is determined by an external reference as can be seen in Figure 4.4.

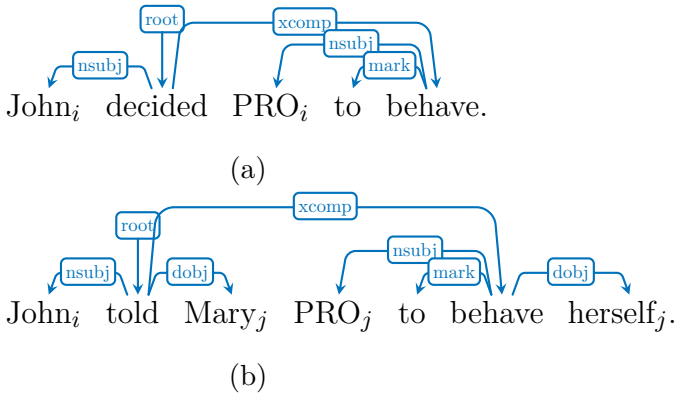


Fig. 4.4 Dependency structure with a PRO subject

These complements are always non-finite. Following the principles stated in Generalisation 4.2.1 and 4.2.3 the non-finite complement clause introduced by *xcomp* relation would receive by default a PRO subject (controlled or arbitrary).

The markers (conjunctions, prepositions or Wh-elements) at the beginning of the embedded clause are no longer connected via *xcomp* relation but instead via either *prepc*, *rcmod*, *partmod* and *infmod* and a slight variation in clause features and constituency. Those cases are no longer treated under the PRO null element considerations and will be discussed later in this chapter since they correspond to other types of empty

elements. The only exception, however, is the *prepc* relation only with the preposition "whether" which also introduces a complement clause with PRO element.

I have explained earlier how to identify the place where a PRO element should be created. Before creating it we need to find out (1) whether PRO is arbitrary (equivalent to pronoun "one") or it is bound to another constituent. And if it is bound then decide (2) whether it is bound to (and coindexed with) subject or object (case in which we say that PRO is subject or object controlled) as can be seen in Figure 4.4.

Generalisation 4.2.6 above introduced test whether the complement clause is interrogative or declarative (which resembles mood determination in SFG). Many grammars, including SFG, do not consider that the non-finite clauses can have interrogative/declarative variation (called by Halliday & Matthiessen (2004: 107-167) *mood* feature). Nonetheless, in GBT, even if a clause is non-finite such a distinction is useful. The complement clauses can have structural variation resembling a declarative or interrogative mood for the reason that a complement clause can start with a Wh-constituent which turns it into an interrogative one. Thus the test is whether there is a Wh-marker (who, whom, why, when, how) or the preposition "whether". The presence of any marker like in example 49 at the beginning of the complement clause will change the dependency relation from *xcomp* to another one and sometimes the structure of the dependent clause as well. The only case when the complement clause remains subjectless non-finite is the case it is introduced via *prepc* relation and the preposition "whether". However if any such marker is missing then the clause is declarative and thus it must be controlled by a NP, so the arbitrary PRO is excluded. The cases of Wh-marked non-finite clauses will be treated in the Section 4.3.4 about the Wh-element movement.

(49) Albert asked [whether/how/when/ PRO to go].

Based on the above I propose Generalisation 4.3.1 enforcing obligatory control for *xcomp* clauses.

Generalization 4.3.1. If a clause is introduced by *xcomp* relation then it must have a PRO element which is bound to either subject or object of parent clause.

(50) Albert$_i$ asked [PRO$_i$ to go alone].

(51) Albert$_i$ was asked [PRO$_i$ to go alone].

(52) Albert$_i$ asked Wendy$_j$ [PRO$_j$ to go alone].

(53) Albert$_i$ was asked by Wendy$_j$ [PRO$_i$ to go alone].

The generalization 4.2.7 required a test for passivization (also know in dependency grammar and SFL as *voice*). Knowing the voice of the parent clause is necessary

in order to determine what NP is controlling the PRO element in the complement clause. Consider Example 50 and it's passive form 51. In both cases there is only one NP that can command PRO and it is the subject of the parent clause "Albert". So we can generalise that the voice does not play any role in controller selection in one argument clauses (i.e. clauses without a nominal complement). In Examples 52 and 53 the parent clause takes two semantic arguments. Second part of principle 4.2.2 states that in case of obligatory control PRO must be *c-commanded* by an NP. In 52 both NPs("Albert" and 'Wendy") c-command PRO element, however according to Minimality Condition(Haegeman 1991: 479) "Albert" is excluded as the commander of PRO because there is a closer NP that c-commands PRO. In case of 53 the only NP that c-commands PRO is the subject "Albert" because "by Mary" is a PP (prepositional phrase) and also only NPs can control a PRO as stated in principle 4.2.6. In the process of passivization the complement becomes subject and the subject becomes a prepositional (PP-by) complement then the latter is automatically excluded from control candidates, thus conforming to Generalisation 4.2.7 (Haegeman 1991: 281).

The above can be synthesised into Generalisation 4.3.2 appealing to linear proximity of the words. But the linear order dimension is beyond the borders of dependency grammar in the sense that the word order is not being accounted explicitly as a relation. Rather, the solution is technical: each word receives an index for the position it occupies within a sentence which suffices to implement Generalisation 4.3.2 presented in Chapter **??**.

Generalization 4.3.2. The controller of PRO element in a lower clause is the closest nominal constituent of the higher clause.

The adjunct non-finite clauses such as the ones in Example 21 ("John hired Mery [PRO to fire Bill]") and 22 ("John abandoned the investigation [PRO to save money]") shall be treated exactly as the non-finite complement clauses are. Generalisation 4.2.8 emphasises obligatory control for them. The only difference between the adjunct and complement clauses is dictated by the verb of the higher clause and whether it theta marks or not the lower clause. In dependency grammar the adjunct clauses are also introduced via *xcomp* and *prepc* relations, so syntactically there is no distinction between the two patterns.

The *prepc* relation in dependency grammar introduces a prepositional clausal modifier for a verb (VN), noun (NN) or adjective (JJ). Adjective and noun modification are cases of copulative clauses. Such configuration are not relevant to the context of this work because, as we will see in Section **??**, the dependency graphs are normalised.

This process involves, among others, transforming the copulas into verb predicated clauses instead of adjective or noun predicated clauses.

The last subordinate type concerned with the PRO element is the subject clause such as the one in Example 23 ("PRO$_i$ smoking is bad for the health$_j$"). In dependency structure, the subject non-finite clauses are introduced via *csubj* relation. They are quite different from complement and adjunct clauses because, according to generalization 4.2.9 the PRO is optionally controlled. Since in this case it is not possible to bind PRO solely on syntactic grounds, the generalization 4.2.9 proposes arbitrary interpretation discussed in Section 4.2.1. Next I turn to identifying the second type of null elements (NP-traces) in the dependency structure of a sentence.

4.3.2 NP-traces and Process Type Database

Syntactically, NP raising can occur only when there is a complement clause by moving the subject of a lower clause into a position of a higher clause. The subject of the higher clause c-commands the subject of the lower clause. This is exactly the same syntactic configuration as in the case of PRO subjects. In dependency grammar the lower clause is introduced via *xcomp* (as explained in Section 4.2.1).

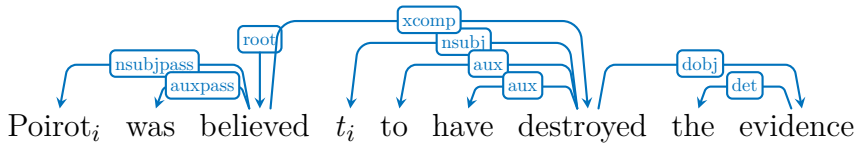


Fig. 4.5 Dependency parse for Example 28

Figures 4.5 and 4.6 represent a dependency parses for Examples 28 and 30. In such cases the subject position in the embedded clause is a NP-trace coindexed with the subject position in the higher clause (whether it is also a trace that is a part of a chain or an overt element). This is the case only if the embedded clause, of course, does not have already an over subject, if it is not introduced with the conditional marker "if" or with preposition "for" and if the higher clause has no nominal complement between the subject and the embedded complement clause. However, just by doing so it is not possible to distinguish whether the empty subject is a PRO or a NP trace *t*.

Table 4.4 represents the semantic role distribution for verb senses in examples above. Example 28 is a passive clause with an embedded complement clause. The subjects and complements switch places in passive clauses and so do the semantic roles. That would mean that Cognizant role goes is to be assigned to embedded/complement

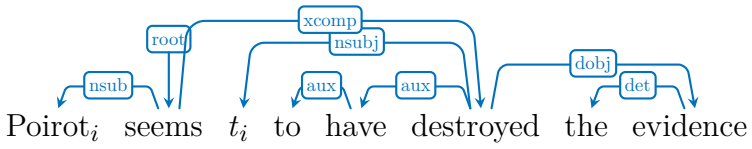


Fig. 4.6 Dependency parse for Example 30

clause. However Phenomena is the only semantic role that can be filled by a clause, all other roles take nominal, prepositional or adjectival groups.

Verb	*Process type*	*canonical distribution of semantic roles*
Believe	Cognition	Cognizant + V + Phenomenon
Seem	Cognition	It + V + Cognizant + Phenomenon
Destroy	Action	Agent + V + Affected

Table 4.4 Semantic role distribution for verbs "believe" and "seem"

In example 30 the verb "seem" assigns roles only to complements and as the embedded clauses can take only the phenomenon role, the cognizant is left unassigned and so does not assign any thematic role to the subject which then can be filled either by an expletive or a moved NP.

When the empty subject in the embedded clause is detected and instantiated through the *obligatory subject control* pattern it is important to distinguish among the cases.

First of all this distinction is crucial for assignment of thematic roles to the constituents. So the problem is deciding on the type of relationship between the empty constituent and it's antecedent (subject of matrix clause) to which it is bound. In the case of *PRO* constituent, the thematic roles are assigned to both the empty constituent and to its antecedent locally in the clause they are located. So the PRO constituent receives a thematic label dictated by the verb of the embedded clause and the antecedent by the verb of the matrix clause. In the *t*-trace case the thematic role is assigned only to the empty constituent by the verb of the embedded clause and this role is propagated to its antecedent.

Making such distinction directly involves checking role distribution of upper and lower clause verb and deciding the type of relationship between the empty category and its antecedent. If such a distinction shall be made at this stage of parsing or postponed to Transitivity analysis (i.e. semantic role labeling) is under discussion because each approach introduces different problem.

Before discussing each approach I would like to state a technical detail. When the empty constituent is being created it requires two important details: (a) the antecedent constituent it is bound to and (b) the type of relationship to it's antecedent constituent or if none is available the type of empty element: t-trace or PRO. Now identifying the antecedent is quite easy and can be provided at the creation time but since the empty element type may not always be available then it may have to be marked as partially defined.

The first solution is to create the empty subject constituents based only on syntactic criteria, ignoring for now element type (either PRO or *t*-trace) hence postponing it to semantic parsing phase. The advantage of doing so is a clear separation of syntactic and semantic analysis. The empty subject constituents are created in the places where they should be and it leaves aside the semantic concern of how the thematic roles are distributed. The disadvantage is leaving the created constituents incomplete or under-defined. Moreover the thematic role distribution must be done within the clause limits but because of raising, this process must be broadened to a larger scope beyond clause boundaries. Since transitivity analysis is done based on pattern matching, the patterns rise in complexity as the scope is extended to two or more clauses thus excluding excludes iteration over one clause at a time (which is desirable).

Otherwise, a second approach is to decide the element type before Transitivity analysis (semantic role labeling) and remove the burdened of complex patterns that go beyond the clause borders. Also, all syntactic decisions would be made before semantic analysis and the empty constituents would be created fully defined with the binder and their type but that means delegating semantically related decision to syntactic level (in a way peeking ahead in the process pipeline).

The solution adopted here is mix of the two avoiding two issues: (a) increasing the complexity of patterns for transitivity analysis, (b) leaving undecided which constituents accept thematic role in the clause and which don't.

The process to distinguish the empty constituent type starts by (a)identifying the antecedent and the empty element (through matching the subject control pattern in Figure **??**), (b) identifying the main verbs of higher and lower clauses and correspondingly the set of possible configurations for each clause(by inquiry to process type database(PTDB) described in the transitivity analysis Section **??**).

Generalization 4.3.3. To distinguish the *t*-traces check the following conditions:

- the subject control pattern matches the case AND

- the process type of the higher clause is two or three role cognition, perception or emotion process (considering constraints on Cognizant and Phenomenon roles). AND
- among the configurations of higher clause there is one with:
  - an expletive subject OR
  - the Phenomenon role in subject position OR
  - Cognizant in subject position AND the clause has passive voice or interrogative mood (cases of movement).

If conditions from generalization 4.3.3 are met then the empty constituent is a subject controlled *t*-trace. Now we need a set of simple rules to mark which constituents shall receive a thematic role. These rules are presented in the generalization 4.3.4 below.

Generalization 4.3.4. Constituents receiving thematic roles shall be marked with "thematic-role" label, those that do not receive a thematic role shall be marked with "non-thematic-role" and those that might receive thematic role with "unknown-thematic-role". So in each clause:

- the subject constituent is marked with "thematic-role" label unless (a) it is an expletive or (b) it is the antecedent of a *t*-trace then marked "non-thematic-role"
- the complement constituent that is an nominal group (NP) or an embedded complement clause is marked with "thematic-role" label.
- the complement that is a prepositional group (PP) is marked with "unknown-thematic-role".
- the complement that is a prepositional clause is marked with "unknown-thematic-role" label unless they are introduced via "that" and "whether" markers then it is marked with "thematic-role" label.
- the adjunct constituents are marked with "non-thematic-role"

4.3.3 Wh-trances

Let's turn now to how Wh-movement and relative clauses are represented and behave in dependency grammar and SF grammars. Figures 4.7, 4.8 present dependency parses

for Wh-movement from subject and object positions of lower clause while 4.9 from adjunct position.

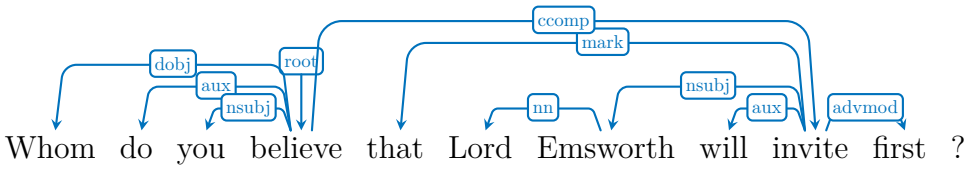


Fig. 4.7 Dependency parse for Example 39

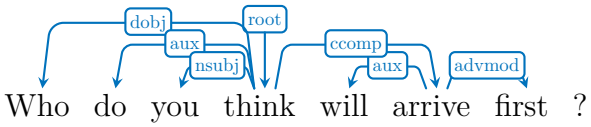


Fig. 4.8 Dependency parse for Example 38

As mentioned in Haegeman (1991: 375) treats each Wh-element as the head of the Wh-phrase. I do not share this perspective. In some cases they function as heads but there are other cases when they act as determiners, possessors or adjectival modifiers. This issue is extensively discussed by Abney (1987); Quirk et al. (1985); Halliday & Matthiessen (2013).

For clarity purposes I define the Wh-group and Wh-element as given in Definition 4.3.1. Note that Wh-group is not a new unit class in the grammar but a constituent feature that can span across three unit classes.

Definition 4.3.1 (Wh-group, Wh-element)**.** *Wh-group* is a nominal, prepositional or adverbial group that contains Wh-element either as head or as modifier. The *Wh-element* is a unit element filled by any of the following words or their morphological derivations (by adding suffixes *-ever*, *-soever*): *who, whom, what, why, where, when, how.*

The Functional distribution for Wh-elements and Wh-groups is presented in the table ?? below.

Features	**Clause functions of the Wh-group**		**Group functions of Wh-element**
	Subject	**Complement**	
person	who, whoever	whom, whomever, whomsoever	head/thing
person, possessive	whose		possessor
person/non-person	which		determiner
non-person	what, whatever		head/thing
	Adjunct		
various circumstantial features	when, where, why, how (whether, whence, whereby, wherein) (and their *-ever* derivations)		head/modifier

Table 4.5 Functions and features of Wh-elements and groups

Just like in cases of NP-movement, the Wh-groups move only into two and three role cognition, perception and emotion figures. In contrast, if the NP-antecedents land in expletive or passive subject position then the Wh-antecedents land in subject or subject preceding position functioning as subject, complement or adjunct functions depending on the Wh-Element.

The essential features for capturing the Wh-movement in dependency graphs are (a) the finite complement clause identified by *ccomp* relation between the matrix and embedded clause (b) the Wh-element/group plays a complement function in higher clause which is identifiable by *dobj*, *prep* or *advmod* relations to the main verb (c) the function of the Wh-trace in lower clause is either Subject(e.g. 39), Object(e.g. 38) or Adjunct.

ccomp relation is defined in Marneffe & Manning (2008a) to introduce a complement clauses with an internal subject which are usually finite. I must emphasize the fact that the lower clause must be embedded into the higher one and receive a thematic role in higher clause.

Regardless whether the syntactic function of the traces in the lower clause is Subject or Object, in the higher clause the Wh-group takes Object function and is bound to the main verb via *dobj* relation but is positioned before the main verb and the Subject (a structure corresponding to Wh-interrogatives). Wh-group can take also Subject

function in the higher clause, but then it is not a case of Wh-movement and is irrelevant for us at this point because there is no empty element, i.e. Wh-trace. The attribution of clause function to the Wh-trace is based on either the case of the Wh-group or the missing functional constituent in the lower clause.

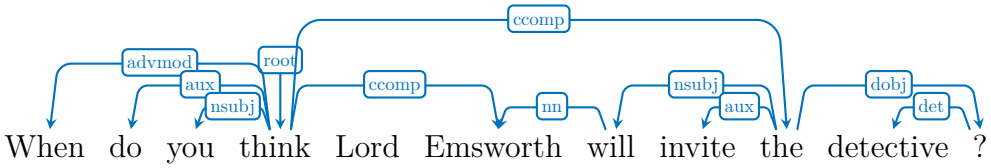


Fig. 4.9 Example dependecy parse with Adjunct Wh-element

In case of Wh-traces with Adjunct function in the lower clause, like in figure 4.9, their antecedents also receive adjunct function in the higher clause. Adjunct Wh-group cannot bind to the clause it resides in if the clause has (generic) present simple tense and thus it has to be antecedent for a trace in lower clause which has other tense or modality than present simple. The reader can experiemnt with changing tense in the example 4.9.

(54) Who$_i$ believes that Lord Emsworth will invite a detective?

(55) To whom$_i$ did Poirot say t$_i$ that Lord Emsworth will invite a detective?

Not always the Wh-groups are movements from lower clause. It is possible that the trace of the moved element to reside in the higher clause (complement) or even have a case of no movement when Wh-element takes the subject function in the higher clause. 54 and 55 are good examples of a *short* movement (in clause movement). However the short movement in dependency grammar has no relevance because the dependency grammar is order free and the functions are already assigned accordingly so short movement is not a subject of interest for the current chapter.

4.3.4 Chaining of Wh-trances

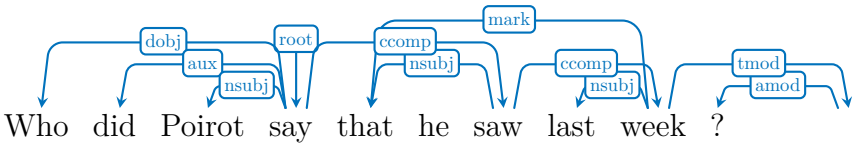


Fig. 4.10 Dependency parse for Example 48

Recall the *cyclic* and *successive* properties of Wh-movement from previous section underlined by example 48 and its dependecy parse in figure 4.10. GBT suggests that the Wh-movement leaves traces in all the intermediary clauses. In dependency grammar these properties shall be treated instrumentally for determining intermediary hops in search for the foot of the chain and none of the intermediary traces shall be created. There simply is no further purpose for them as they do not receive a thematic role in the intermediary clauses.

4.3.5 Wh-trances in relative clauses

In GB theory the Wh-elements that form relative clauses (*who, whom, which, whose*) are considered moved. In dependency grammar such movement is redundant since the Wh-element and its trace are collapsed and take the same place and function. The Wh-elements function either as subject or complement. When the relative clause is introduced by a Wh-group there is no empty element to be detected, rather there is anaphoric indexing relation to a noun it refers to. Focusing now on the relative clauses, there are three more possible constructions that introduce them: (a) a prepositional group that contains a Wh-element, (b) "that" complementizer which behaves like a relative pronoun (c) the Zero Wh-element which is an empty element and which functions the same way as a overt Wh-element. The table 4.6 lists possible elements that introduce a relative clause, their features and the functions they can take.

Next I discuss how to identify, create and bind the traces of Wh-elements to their antecedents.

Relativizing element	**Feature**	**(Clause) Function**	**Examples**
who	person	subject	... the woman who lives next door.
whom	person	complement (non defining clause)	... the doctor whom I have seen today.
which	non-person	subject/ complement	... the apple which is lying on the table. ... the apple which George put on the table.
whose	possessive, person	possessor in subject	... the boy whose mother is in a nurse.
(*any of the above in prepositional group*)	person/ non-person	thing/possessor in subject	... the boy to whom I gave an advice. ... the cause for which we fight.
that	person/ non-person	subject	... the apple that lies on the table.
Zero Wh-element	person/ non-person	subject	... the sword sent by gods.

Table 4.6 The Wh-elements introducing a relative clause.

Compare the dependency parse with an overt Wh-element in Figure 4.11 and the covert one in 4.12.

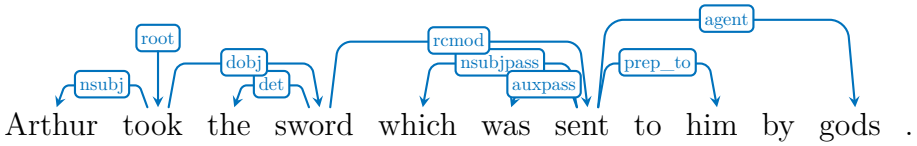


Fig. 4.11 Dependency parse for “Arthur took the sword which was sent to him by gods.”

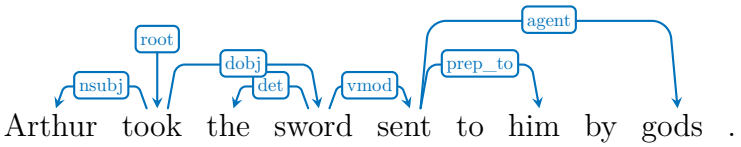


Fig. 4.12 Dependency parse for "Arthur took the sword sent to him by gods."

Relative clauses in dependency graphs are introduced by *rcmod*, *partmod*, *infmod* and *vmod* relations. The *rcmod* introduces relative clauses containing a Wh-group while the *partmod* and *infmod* introduce finite and non-finite relative clauses with Zero Wh-element. After the Version 3.3 of Stanford parser the *partmod* and *infmod* relations have been merged into *vmod*. So the dependency relation is the main signaller if empty Wh-elements even though they are subject to corrections in preprocessing (Section **??**) phase to ensure a uniform treatment of each relation type.

The Zero Wh-element behaves exactly like the *PRO element* in the case of non finite complement clauses discussed in Section 4.2.1. It receives thematic roles in both the higher clause and in lower clause and is not a part of a chain like the cases of NP/Wh-movement.

4.4 Discussion

This chapter treats the identification of the *null elements* in syntactic structures. Section 4.2 presents how GBT theory handles null elements and then Section 4.3 shows how the same principles translate into Stanford dependency graphs.

Identification of null elements is important for the semantic role labeling process described in Chapter **??** because usually the missing elements are participant roles (theta roles) shaping the semantic configuration. The semantic configurations (gathered in a database) are matched against the syntactic structure and missing elements lead to failing (false negatives) or erroneous matches (false positives). Therefore to increase the accuracy of semantic role labeling spotting null elements is a prerequisite.

This Chapter also contributes to establishing cross theoretical connections that is among current thesis objectives. Specifically it provides translations of necessary principles and generalizations from GB theory into the context of dependency grammar. These results are directly used in Section **??** for generating graph patterns.

References

Abney, S. 1987. *The English noun phrase in its sentential aspects*. MIT Press.

Bach, Emmon. 1966. *An introduction to transformational grammars*. Holt, Rinehart and Winston. Inc.

Baker, Collin F, Charles J Fillmore & John B Lowe. 1998. The Berkeley FrameNet Project. In Christian Boitet & Pete Whitelock (eds.), *Proceedings of the 36th annual meeting on association for computational linguistics*, vol. 1 ACL '98, 86–90. University of Montreal Association for Computational Linguistics. doi:10.3115/980845.980860. <http://portal.acm.org/citation.cfm?doid=980845.980860>.

Bateman, John A. 2008. Systemic-Functional Linguistics and the Notion of Linguistic Structure: Unanswered Questions, New Possibilities. In Jonathan J. Webster (ed.), *Meaning in context: Implementing intelligent applications of language studies*, 24–58. Continuum.

Bateman, John A. & Christian M. I. M. Matthiessen. 1988. Using a functional grammar as a tool for developing planning algorithms — an illustration drawn from nominal group planning. Tech. rep. Information Sciences Institute Marina del Rey, California. (Penman Development Note).

Bresnan, Joan. 1982. Control and complementation. *Linguistic Inquiry* 13(3). 343–434.

Bühler, Karl. 1934. *Sprachtheorie: die Darstellungsfunktion der Sprache*. Jena: Fischer.

Butler, Christopher S. 2003a. *Structure and function: A guide to three major structural-functional theories; Part 1: Approaches to the simplex clause*. Amsterdam and Philadelphia: John Benjamins.

Butler, Christopher S. 2003b. *Structure and function: A guide to three major structural-functional theories; Part 2: From clause to discourse and beyond*. Amsterdam and Philadelphia: John Benjamins.

Carreras, Xavier. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (emnlp-conll)*, .

Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2. 113–124.

Chomsky, Noam. 1957. *Syntactic Structures*. Mouton & Co.

Chomsky, Noam. 1965. *Aspects of the theory of syntax*. Cambridge, Massachusetts: M.I.T. Press.

Chomsky, Noam. 1981. *Lectures on Government and Binding*. Dordrecht: Foris.

Chomsky, Noam. 1982. *Some concepts and consequences of the theory of government and binding*, vol. 6. MIT press.

Chomsky, Noam. 1986. *Barriers*, vol. 13. MIT press.

Chomsky, Noam. 1993. *Lectures on government and binding: The pisa lectures* 9. Walter de Gruyter.

Day, Michael David. 2007. *A Corpus-Consulting Probabilistic Approach to Parsing : the CCPX Parser and its Complementary Components*: Cardiff University dissertation.

Fawcett, Robin. 2000. *A Theory of Syntax for Systemic Functional Linguistics*. John Benjamins Publishing Company paperback edn.

Fawcett, Robin P. 1988. What makes a 'good' system network good? In James D. Benson & William S. Greaves (eds.), *Systemic functional approaches to discourse*, 1–28. Norwood, NJ: Ablex.

Fawcett, Robin P. 2008. *Invitation to Systemic Functional Linguistics through the Cardiff Grammar*. Equinox Publishing Ltd.

Fawcett, Robin P. 2009. How to Analyze Process and Participant Roles. In *The functional semantics handbook: Analyzing english at the level of meaning*, Continuum.

Fellbaum, Christiane & George Miller (eds.). 1998. *WordNet: An electronic lexical database*. The MIT Press.

Fillmore, Charles J, Christopher R Johnson & Miriam RL Petruck. 2003. Background to framenet. *International journal of lexicography* 16(3). 235–250.

Gildea, Daniel & Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics* 28(3). 245–288.

Haegeman, Liliane. 1991. *Introduction to Government and Binding Theory*, vol. 2. Blackwell.

Halliday, Michael A. K. 1957. Some aspects of systematic description and comparison in grammatical analysis. In *Studies in Linguistic Analysis*, 54–67. Oxford: Blackwell.

Halliday, Michael A. K. 1961. Categories of the theory of grammar. *Word* 17(3). 241–292.

Halliday, Michael A. K. 1996. On grammar and grammatics. In Ruqaiya Hasan, Carmel Cloran & David Butt (eds.), *Functional descriptions – theory in practice* Current Issues in Linguistic Theory, 1–38. Amsterdam: Benjamins.

Halliday, Michael A.K. 2002. Categories of the theory of grammar. In Jonathan Webster (ed.), *On grammar (volume 1)*, 442. Continuum.

Halliday, Michael A.K. 2003. On the "architecture" of human language. In Jonathan Webster (ed.), *On language and linguistics*, vol. 3 Collected Works of M. A. K. Halliday, 1–32. Continuum.

Halliday, Michael A.K. & Christian M.I.M. Matthiessen. 2013. *An Introduction to Functional Grammar (4th Edition)*. Routledge 4th edn.

Halliday, Michael A.K. & M.I.M. Matthiessen, Christian. 2004. *An introduction to functional grammar (3rd Edition)*. Hodder Education.

Hasan, Ruqaiya. 2014. The grammarian's dream: lexis as most delicate grammar. In Jonathan Webster (ed.), *Describing language form and function*, vol. 5 Collected Works of Ruqaiya Hasan, chap. 6. Equinox Publishing Ltd.

Hjelmslev, Louis. 1953. *Prolegomena to a theory of language*. Bloomington, Indiana: Indiana University Publications in Anthropology and Linguistics. Translated by Francis J. Whitfield.

Honnibal, Matthew. 2004. Converting the Penn Treebank to Systemic Functional Grammar. *Technology* 147–154.

Honnibal, Matthew & Jr James R Curran. 2007. Creating a systemic functional grammar corpus from the Penn treebank. *Proceedings of the Workshop on Deep ...* 89–96. doi:10.3115/1608912.1608927. <http://dl.acm.org/citation.cfm?id=1608927>.

Hutchins, W John. 1999. Retrospect and prospect in computer-based translation. In *Proceedings of mt summit vii "mt in the great translation era"* September, 30–44. AAMT.

Johnson, Christopher & Charles J. Fillmore. 2000. The framenet tagset for frame-semantic and syntactic coding of predicate-argument structure. In *Proceedings of the 1st north american chapter of the association for computational linguistics conference* NAACL 2000, 56–62. Stroudsburg, PA, USA: Association for Computational Linguistics. <http://dl.acm.org/citation.cfm?id=974305.974313>.

Kasper, Robert. 1988. An Experimental Parser for Systemic Grammars. In *Proceedings of the 12th International Conference on Computational Linguistics*, .

Kay, Martin. 1985. Parsing In Functional Unification Grammar. In D.Dowty, L. Karttunen & A. Zwicky (eds.), *Natural language parsing*, Cambridge University Press.

Kipper, Karin, Anna Korhonen, Neville Ryant & Martha Palmer. 2008. A large-scale classification of English verbs. *Language Resources And Evaluation* 42(1). 21–40. doi:10.1007/s10579-007-9048-2.

Mann, William C. 1983. An Overview of the PENMAN Text Generation System. Tech. Rep. ISI/RR-83-114 USC/Information Sciences Institute Marina del Rey, CA.

Marneffe, Marie-Catherine, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre & Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the ninth international conference on language resources and evaluation (lrec-2014)(vol. 14)*, European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2014/pdf/1062{_}Paper.pdf>.

Marneffe, Marie-Catherine, Bill MacCartney & Christopher D Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Lrec 2006*, vol. 6 3, 449–454. Stanford University. <http://nlp.stanford.edu/manning/papers/LREC{_}2.pdf>.

Marneffe, Marie-Catherine & Christopher D. Manning. 2008a. Stanford typed dependencies manual. Tech. Rep. September Stanford University. <http://nlp.stanford.edu/downloads/dependencies{_}manual.pdf>.

Marneffe, Marie-Catherine & Christopher D. Manning. 2008b. The Stanford typed dependencies representation. *Coling 2008 Proceedings of the workshop on CrossFramework and CrossDomain Parser Evaluation CrossParser 08* 1(ii). 1–8. doi:10.3115/1608858.1608859. <http://portal.acm.org/citation.cfm?doid=1608858.1608859>.

Matthiessen, Christian M. I. M. & John A. Bateman. 1991. *Text generation and systemic-functional linguistics: experiences from English and Japanese*. London and New York: Frances Pinter Publishers and St. Martin's Press.

Matthiessen, M.I.M., Christian. 1985. The systemic framework in text generation: Nigel. In James Benson & Willian Greaves (eds.), *Systemic perspective on Discourse, Vol I*, 96–118. Ablex.

McCarthy, John, Marvin L. Minsky, Nathaniel Rochester & Claude E. Shannon. 2006. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955. *AI Magazine* 27(4). 12. doi:10.1609/aimag.v27i4.1904. <http://www.aaai.org/ojs/index.php/aimagazine/article/view/1904{%}5Cnhttp://www.mendeley.com/catalog/proposal-dartmouth-summer-research-project-artificial-intelligence-august-31-1955/{%}5Cnhttp://www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.htmlhttp://>.

McDonald, David D. 1980. *Natural Language Production as a Process of Decision Making under Constraint*: MIT, Cambridge, Mass dissertation.

McDonald, Ryan, Koby Crammer & Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, 91–98. Association for Computational Linguistics.

McDonald, Ryan, Kevin Lerman & Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the tenth conference on computational natural language learning* CoNLL-X '06, 216–220. Stroudsburg, PA, USA: Association for Computational Linguistics. <http://dl.acm.org/citation.cfm?id=1596276.1596317>.

McDonald, Ryan & Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th conference of the european chapter of the association for computational linguistics*, .

Miyao, Yusuke & Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage hpsg parsing. In *Proceedings of the 43rd annual meeting on association for computational linguistics* ACL '05, 83–90. Stroudsburg, PA, USA: Association for Computational Linguistics. doi:10.3115/1219840.1219851. <https://doi.org/10.3115/1219840.1219851>.

Neale, Amy C. 2002. More Delicate TRANSITIVITY: Extending the PROCESS TYPE for English to include full semantic classifications. Tech. rep. Cardiff University.

Nivre, Joakim. 2006. *Inductive dependency parsing (text, speech and language technology)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

O'Donnell, Michael. 1993. Reducing Complexity in Systemic Parser. In *Proceeedings of the third international workshop on parsing technologies*, .

O'Donnell, Michael. 1994. Sentence Analysis and Generation: a systemic perspective. Tech. rep. Department of Linguistics, University of Sydney.

O'Donnell, Michael. 2005. The UAM Systemic Parser. *Proceedings of the 1st Computational Systemic Functional Grammar Conference* <http://www.wagsoft.com/Papers/ODonnellUamParser.pdf>.

O'Donnell, Michael J. & John A. Bateman. 2005. SFL in computational contexts: a contemporary history. In Ruqiaya Hasan, M.I.M. Matthiessen, Christian & Jonathan Webster (eds.), *Continuing discourse on language: A functional perspective*, vol. 1 Booth 1956, 343–382. Equinox Publishing Ltd.

O'Donoghue, Tim. 1991. The Vertical Strip Parser: A lazy approach to parsing. Tech. rep. School of Computer Studies, University of Leeds.

Pei, Wenzhe, Tao Ge & Baobao Chang. 2015. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, vol. 1, 313–322.

Pollock, Jean-Yves. 1989. Verb movement, universal grammar, and the structure of ip. *Linguistic inquiry* 20(3). 365–424.

Postal, P. M. 1974. *On Raising*. MIT Press.

Quirk, Randolph, Sidney Greenbaum, Geoffrey Leech, Jan Svartvik & David Crystal. 1985. *A comprehensive grammar of the English language*, vol. 1 2. Longman. <http://www.amazon.com/dp/0582517346http://journals.cambridge.org/production/action/cjoGetFulltext?fulltextid=2545152>.

Schuler, Karin Kipper. 2005. Verbnet: A broad-coverage, comprehensive verb lexicon .

Souter, David Clive. 1996. *A Corpus-Trained Parser for Systemic-Functional Syntax*: University of Leeds Phd. <http://etheses.whiterose.ac.uk/1268/>.

Stevan Harnad. 1992. The Turing Test Is Not A Trick: Turing Indistinguishability Is A Scientific Criterion. *SIGART Bulletin* 3(4). 9–10. <http://users.ecs.soton.ac.uk/harnad/Papers/Harnad/harnad92.turing.html>.

Stockwell, Robert P., Barbara Hall Partee & Paul Schacter. 1973. *The major syntactic structures of english*. New York: Holt, Rinehart and Winston. <http://hdl.handle.net/2027/mdp.39015002216417>. Bibliography: p. 811-840.

Stowell, T.A. & E. Wehrli. 1992. *Syntax and the lexicon* Syntax and semantics. Academic Press. <https://books.google.lu/books?id=yiEcAQAAIAAJ>.

Tesniere, Lucien. 2015. *Elements of Structural Syntax*. John Benjamins Publishing Company translation by timothy osborne and sylvain kahane edn.

Turing, Allan. 1950. Computing machinery and intelligence. *Mind* 59. 433–460.

Weerasinghe, Ruvan. 1994. *Probabilistic Parsing in Systemic Functional Grammar*: University of Wales College of Cardiff dissertation.

Winograd, Terry. 1972. *Understanding natural language*. Orlando, FL, USA: Academic Press, Inc. <http://linkinghub.elsevier.com/retrieve/pii/0010028572900023>.

Zhang, Yue & Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies: short papers-volume 2*, 188–193. Association for Computational Linguistics.