Converting the Penn Treebank to Systemic Functional Grammar

Matthew Honnibal
Department of Linguistics, Macquarie University
Macquarie University
2109 Sydney
Australia
mhonn@it.usyd.edu.au

Abstract

Systemic functional linguistics offers a grammar that is semantically organised, so that salient grammatical choices are made explicit. This paper describes the explication of these choices through the conversion of the Penn Treebank into a systemic functional grammar corpus. Developing such a resource can help connect work in natural language processing to a significant body of research dealing explicitly with the issue of how lexical and grammatical selections create meaning.

1 Introduction

The Penn Treebank was designed to maximise consistency and annotator efficiency, rather than conformity with any particular linguistic theory (Marcus et al., 1994). This results in trees that strongly suggest the use of synthetic features to explicate semantically significant grammatical choices like *mood*, *tense*, *voice* or *negation*. These distinctions lie latent in the configuration of the tree in the Treebank II annotation scheme, making it difficult for a machine learner to make use of them.

Rather than the ad hoc addition of this information at the feature extraction stage, the corpus can be re-presented in a way that makes feature extraction more principled. This involves increasing the size and complexity of the representation of a sentence by organising the tree semantically. Organising a grammar semantically is by no means a trivial task, and has been an active area of linguistic research for the last forty years. This paper describes the conversion of the Penn Treebank into a prominent output of such research, systemic functional grammar (SFG).

Systemic functional grammar does not confine its description to syntactic structure, but includes a representation of the choices grammatical configurations represent — or 'realise', to use the term preferred in the linguistics literature (Halliday, 1976).

There is growing evidence that systemic functional grammar can be usefully applied to natural language processing (Munro, 2003; Couchman and Whitelaw, 2003), and there is a strong history of interaction between systemic functional linguistics and natural language generation (Matthiessen and Bateman, 1991). However, there is currently a lack of computational SFG resources. There is no standard format for machine readable annotation, no annotated corpora, and no useable parsers. Converting the Penn Treebank will make a large body of SFG annotated data available to computational linguists for the first time, an important step towards addressing this situation.

We first discuss some preliminaries relating to the nature of systemic functional grammar, and the scope of the converted corpus's annotation. We then discuss the conversion of the treebank's phrase-structure representation to SFG constituency structure, and finally we discuss the addition of interpersonal and textual function structures.

2 Some preliminaries

2.1 Structure of the SFG analysis

Systemic functional grammar divides the task of grammatical analysis — the process of stating the grammatical properties of a text — into two parts: analysis of syntactic structures, and analysis of function structures.

SFG syntactic analysis is constituency based, and is predicated on Halliday's notion of the rank scale (Halliday, 1966): clauses are composed of groups/phrases, which are composed of words, which are composed of morphemes. The main concerns of SFG syntactic analysis are the chunking of words into groups/phrases, and the chunking of groups/phrases into clauses. Levels of constituency between groups/phrases and their words are recognised in the literature (Matthiessen, 1995), but rarely brought into focus in research unless the group/phrase contains, or is, an embedded constituent from another rank (e.g., a nominal group like 'the man' with an embedded relative clause like 'who knew too much').

Function structures can refer to any rank of the constituency, but clause rank functional analysis is generally regarded as the most important. The grammar defines a set of systems, which can be defined recursively using conjunction and disjunction. They are usually represented graphically in system networks (Matthiessen, 1995), as in Figure 1.

In this figure, the nested disjunction 'indicative or interrogative' represents a more delicate, or finer grained, distinction than that between indicative and imperative. After selecting from the initial choice, one proceeds from left to right into increasingly delicate distinctions. These systems are categorised into three metafunctions, which represent different types of meaning language enacts simultaneously (ideational, interpersonal and textual) (Halliday, 1969).

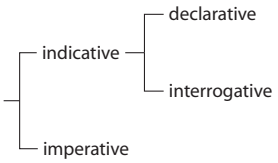

Figure 1: A simple *mood* system, '(indicative or interrogative) or imperative'

2.2 Scope of target annotation

There is no clearly defined limit to systemic functional grammar, in the sense that one could say that a text has been 'fully' analysed. The grammar is constantly being extended, with new kinds of analysis and levels of delicacy suggested. The ultimate aim of the approach is to distinguish every semantically distinct different wording choice (Hasan, 1987).

When working with systemic functional grammar, then, practitioners generally define the scope of their analysis. We must do the same, although the reasons are different. Analysis, so far, has always been performed manually, with only finite time available. Projects have therefore had to decide between the size of a sample and the detail of its analysis. In our case, we are limited to the kinds of analysis which can be directly inferred from the Penn Treebank. Future research will doubtless leverage other resources to extend the analysis of the corpus we present, but attempts to do so are beyond the scope of this paper.

The Penn Treebank presents accurate constituency and part-of-speech information. This is enough information to annotate the corpus automatically with roughly two thirds of the most important clause rank systems: *mood* and *theme*, but not *transitivity*.

The distinction between systems which can be automatically annotated and systems which cannot lies in the way the systems are realised. *Mood* and *theme* are realised primarily through the order of constituents (the order of Subject and Finite in the case of *mood*, and the first Adjunct, Subject, Complement or Predicator in the case of *theme*). They are realised structurally, as opposed to lexically. Other systems are realised through the selection of grammatical items (also called 'function words' — a term we prefer not to use because of the special sense of 'function' in the context of SFG).

Systems that are realised with grammatical items, such as *voice*, *polarity* and *tense*, can also be automatically annotated. Lexically realised systems, on the other hand, require a lexicon or equivalent resource, since the choice of words within identical syntactic structures changes the selection from the system. Trees which are identical at every level except their leaves have different *process type* selections. The central system of transitivity, process type, cannot be analysed for this reason.

The annotation of the corpus we present therefore attempts to include selections from the following systems at clause rank:

- interpersonal
  - mood (i.e. mood type and role tags for Subject, Finite, Predicator, Adjunct, Complement, Vocative)
  - clause class
  - status
  - tense
  - polarity
- textual
  - theme (i.e. role tags for Textual Theme, Interpersonal Theme, Topical Theme, Rheme)
  - voice

Ideational analysis is omitted entirely, because transitivity analysis requires a more complicated approach, as discussed above. Although arguably some aspects of taxis and expansion type could be annotated automatically, because the central information cannot be annotated, we have left it out entirely.

3 Constituency Conversion

We have not found it necessary to use a method of automatic rule induction to generate a CFG. The

lack of a suitable training set made that approach impractical for the time and resources we have had available; and good results have been obtained by simply using a set of hard-coded transformation functions, implemented as a Python script. This approach does have a significant drawback, however: because the script does not output a conversion grammar, correcting systematic errors and other maintenance or extension tasks are much more difficult.

The first process in the conversion of a sentence is to parse the Lisp-style string representation into a tree of generic node objects. Each node contains a function tag (which may be null), a node label and a set of children (which may be empty). The root node is then used to initialise a sentence object, which sorts its immediate children into clause, group and verbal group objects. As each class is initialised, it initialises a clause, verbal group, other group or lexis object with each of its children. The tree is thus recursively re-represented by more specific constituent objects, rather than generic node objects. Subtyping the nodes facilitates the changes to the structure that must be performed, since the structural changes are mostly specific to either verbal groups or clauses.

These changes are divided into a series of steps, each coded as a function. Each function contains a series of conditionals which identify the structure being targeted and how it should be altered. The most significant functions are described in more detail below. This is not an exhaustive list, however, as several trivial changes have been omitted. These include things like node relabelling and the addition of group nodes for conjunctions. There are many changes of this sort, some introduced by the specific mechanics of altering the tree. They are not generally interesting differences between the constituency representations of the Treebank's phrase-structure representation and systemic functional grammar.

3.1 Raising verb phrase predicates

The most obvious difference between SFG constituency and the Treebank II annotation scheme is the flatter, 'minimal bracketing' style SFG uses. To convert a tree to SFG clause constituency, all complements and adjuncts must be raised by attaching them to the clause node; in the Treebank annotation they attach to the verb. Figure 2 illustrates the raising of clause constituents from the verb phrase.

3.2 Raising hypotactic clauses

SFG represents the distinction between hypotaxis and parataxis with features, rather than tree structure. All non-nominalised, non-embedded clauses are therefore siblings dominated by the root clause complex.

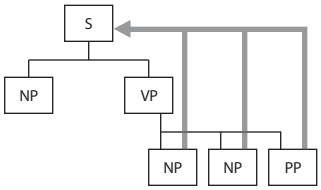

Figure 2: Raising of NP and PP nodes dominated by a VP

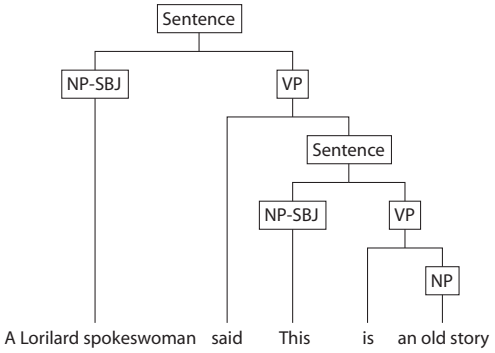

Figure 3: A clause dominating another

Figure 3 shows the Treebank representation, with a hypotactic clause as a child of a VP. Hypotactic clauses are raised to be siblings of the nearest clause node above them. Figure 4 shows the tree after this has been performed.

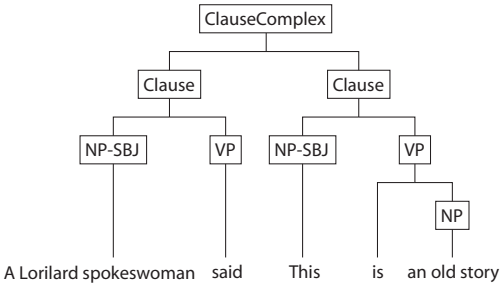

Figure 4: Equally ranked clauses

3.3 Flattening auxiliaries

In the Treebank II annotation scheme, each auxiliary — and the main verb — is given its own node, dominated by the auxiliary before it. This structure needs to be flattened to match the SFG representation. If all of a verb phrase's lexical items have POS tags in the following list: VB, VBD, VBG, VBN, VBP, VBZ; and it only has one verb phrase child, then its lexis attaches to the verb phrase below it. The empty internal node will later be removed in

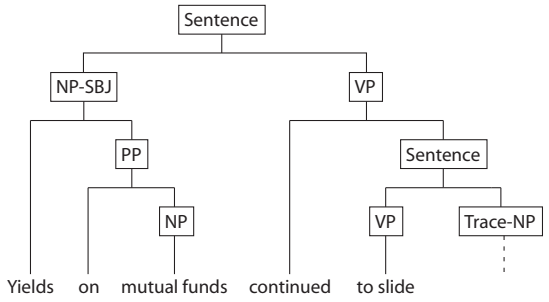

Figure 5: Treebank representation of a sentence that contains a verbal group complex

the generic 'flattening' stage.

3.4 Verbal group complexing

SFG distinguishes between clause complexes and verbal group complexes. The rules for parsing a tree as one or the other type of construction are quite simple.

If a verb phrase has one verb phrase child, and dominates a lexis node that is not a finite, then it is treated as a verbal group complex. Additionally, if a verb phrase has a sentence child that is not a direct quotation, does not have the function tag PRN (parenthetical), and is not labelled SBAR (used for relative and subordinate clauses), it is treated as a verbal group complex. For example, SFG renders the tree in Figure 5 as a single clause, with the verbal group "(continued) (to slide)".

Group and phrase complexing is actually represented a little inaccurately in the script. Ideally, a structural Complex node should be created, and all groups attached to it. This representation would mirror the way clause complexing is handled. Instead, group or phrase complexing is treated like rank-shifting, with the first group dominating the others. This concern is not crucial, however, since it does not affect the clause division or the annotation of function structures.

3.5 Ellipsis

Ellipsis was the most difficult case to deal with, since it involves more than just relocating nodes in the tree. A new clause is created when a verb phrase is identified as part of a clause with an ellipsed subject. The verb phrase is moved to the new clause, along with all of its children, and any items identified as ellipsed are copied and attached. Lexis that is copied in this way must be renumbered, so that the clause sorts properly.

When a verb phrase has two or more verb phrase children, each verb phrase child after the first is moved to a new clause. Figure 6 shows the structure of a sentence containing an ellipsed clause. The siblings of the dominant verb phrase (such as the subject), all lexis of the dominant verb phrase (such as the finite), and all children of the ellipsed verb phrase (such as the complement) are copied to the new clauses. In effect, the only items in the 'original' clause that are not in the 'ellipsis' clauses are children of the first verb phrase (such as the adverbial phrase).

It is not entirely clear that copying the words is the best solution. A trace — an empty group that simply references the original version — is possibly more convenient. The trace solution is more convenient when using the corpus as training data for a computational linguistics task, while copying the elements makes the corpus easier to use for linguistic research. The SFG literature is unhelpful for these kinds of decisions: it is concerned with content descriptions, not representation descriptions.

3.6 Pruning and truncating

Lexical nodes that contain only punctuation or traces are pruned from the tree. Group nodes that contain no lexis are also pruned. This operation is performed recursively, from the bottom up, clearing away any branches that have no lexical leaves. Internal nodes that contain only one child are replaced by that child, truncating non-branching arcs of the tree.

The clearance of punctuation is a problem with the script as it currently stands, since clearly this information should not be lost.

4 Adding Metafunctional Analysis

Function structures must be added after the constituency conversion. The structures attach to clauses in the constituency tree, making separation into clauses essential before systems can be annotated.

Function structures fall into two categories: metafunctional roles, and systems. Metafunctional roles describe the interpersonal, textual or ideational function of a particular constituent, which is considered the role's *realisation*. Systems are instead disjunctions from which a term is selected if the entry condition is met. The names of metafunctional roles are generally capitalised in the literature, while system names are given in italics. We follow this convention to help make the distinction clearer.

As with the constituency conversion, function structures were added by hard-coded functions, implemented as a Python script. Four kinds of information are used for metafunctional analysis:

1. The Penn Treebank's function tags
2. The Penn Treebank's POS tags

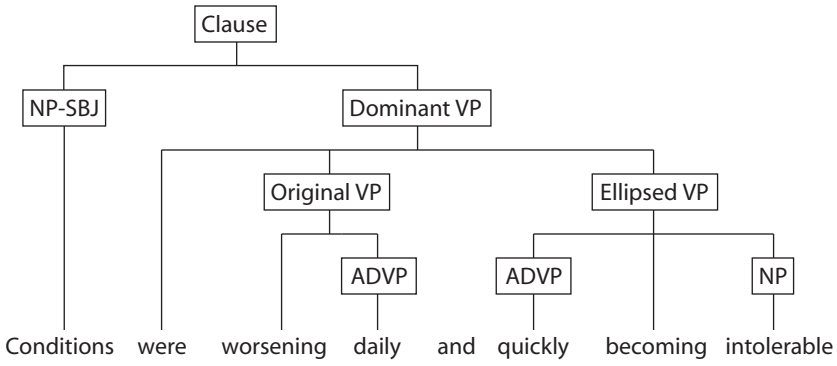

Figure 6: Treebank representation of an ellipsed clause, with verb phrases named

3. The value of other systems
4. The order of constituents in the SFG representation

The use of values from other systems makes the annotation procedure order dependent. They are usually used to determine whether a system's entry condition has been met. For instance, *tense* is not selected by non-finite clauses — so the function that discerns *tense* first checks the that requirement, and assigns null *tense* if the clause has no Finite.

The subsections below give a brief linguistic description of the system being annotated, and then describe the way its selection is calculated. If the entry condition is not met, the selection is considered 'none'.

4.1 Class

Class is an interpersonal system with the possible values 'major' and 'minor'. Major clauses are those with a verbal group. Minor clauses are equivalent to sentence fragments in other grammatical theories. An example from the Penn Treebank is the fragment "Not this year."

If a clause contains a verbal group, it is marked 'major clause'. If it has no verbal group, it is marked 'minor clause'.

4.2 Finite

Finite is an interpersonal role. The Finite is the *tense* marker of a verbal group. It is either the first auxiliary, or it is included with the lexical verb as a morphological suffix. The Finite is a significant unit of the grammar, because the placement of it in relation to the Subject realises *mood type*, and its morphology realises *tense selection* and number agreement with the Subject.

If a clause is minor *class*, or the first word of its verbal group has one of the following POS tags: TO, VBG, VBN; then it does not contain a Finite. Otherwise the first word of the verbal group receives the interpersonal role Finite.

4.3 Predicator

Predicator is an interpersonal role. The Predicator is the lexical verb of a verbal group.

If a clause is minor *class*, it does not contain a Predicator. Otherwise, the last word of the verbal group receives the interpersonal role Predicator. If a verbal group has only one word, that word will therefore receive two interpersonal roles (Finite and Predicator). This is the analysis recommended in the literature (Halliday, 1994).

4.4 Status

Status is an interpersonal system with the possible values 'free' and 'bound'. *Status* refers to whether a clause is 'independent' or 'dependant', to use the terms from traditional grammar.

Minor clauses do not select from the *status* system, so receive the value 'none'. Major clauses that have no Finite, or were originally attached to another clause and were tagged SBAR, or are rank-shifted, are considered bound. All other clauses are considered free.

4.5 Subject

Subject is an interpersonal role. The Subject of a verbal group is the nominal group whose number the verbal group must agree with.

Nominal groups realising Subject are generally tagged explicitly in Treebank II annotation. The exception to this is wh- subjects like 'who', 'what' or 'which'. If no nominal group has the function tag SBJ, and there is a wh- nominal group that was not attached to the verbal group, that nominal group is considered the Subject.

In clauses with an Initiator ('I made him paint the fence'), two nominal groups will usually have been marked subject ('I', 'him'). In these cases, the first

occurring nominal group is considered the subject ('I').

4.6 Mood type

Mood type is an interpersonal system with the possible values 'declarative', 'interrogative' and 'imperative'. *Mood type* refers to whether a clause is congruently a question (interrogative), command (imperative) or statement (declarative).

Minor and bound clauses do not select from this system, and therefore receive the value 'none'. Free clauses with no subject are marked 'imperative'. Clauses with the node labels SQ or SBARQ are marked 'interrogative'. Other free clauses are marked 'declarative'.

4.7 Tense

Tense is an interpersonal system whose value is some sequence of 'present', 'past', 'future', 'modal'. *Tense* refers to the temporal positioning of the process of a clause, with respect to the time of speaking. In English, it is a serial value, because sequences of tenses can be built ('have (present) been (past) going (present)').

Finite declarative and interrogative clauses receive one or more *tense* values. The function iterates through the words of the verbal group (or the first verbal group in a verbal group complex), and assigns these values based on the words' POS tags, and in special cases their text.

If a tag is either VBD or VBN, the value 'past' is appended to the *tense* list. If the tag is either VB, VBG, VBZ or VBP, the value 'present' is appended to the *tense* list. If the tag is MD, then the text is checked. If the word is "'ll", 'will' or 'shall', the value 'future' is appended to the *tense* list. The value 'modal' is appended to the *tense* list for lexical items tagged MD. When an MD tag is seen, the next word in the list is skipped, since it will be a bare infinitive that does not represent a *tense* selection. If the lexical items 'going' or 'about' are seen, the value 'future' is appended to the *tense* list, and the next two words are skipped, as they will be 'to' and an infinitive verb. This does not occur if 'going' is the last word of the verbal group, since in that case it is the process, not a *tense* marker.

Passive clauses will have received an extra 'past' *tense* value, so when a clause is labelled passive, its last *tense* selection is removed.

4.8 Polarity

Polarity is an interpersonal system with the possible values 'positive' and 'negative'. Polarity refers to whether the verbal group is directly negated.

Polarity is the simplest system to determine, since it only involves checking the verbal group for the word "not" (or "n't"). Looking at negation more generally would be far more difficult, since it is more of a semantic motif than specific grammatical system.

4.9 Adjuncts, Complements, Vocatives

Adjunct, Complement and Vocative are interpersonal roles. Nominal groups can be either Vocatives, Adjuncts or Complements. Adjuncts represent circumstances of a clause — the where, why and when of its happening. Complements represent its non-Subject participants — the whom, to whom and for whom of its happening. Vocatives are nominal groups that name the person the clause is addressed to.

Adverbial groups, prepositional phrases and particles are always given the interpersonal function 'Adjunct'. Vocatives are explicitly marked in the Treebank, with the VOC tag. Nominal groups that realise an adverbial function are also explicitly tagged, with either TMP, DIR, LOC, MNR or PNR. Nominal groups with one of these tags receive the interpersonal role 'Adjunct'. All other non-Subject nominal groups receive the interpersonal role 'Complement'.

4.10 Voice

Voice is a textual system with the possible values 'active', 'passive' and 'middle'. *Voice* refers to whether the Subject is also the 'doer' of the clause, or whether the participants have been switched so that the Subject is the 'done to'. Compare the active clause "the dog bit the boy" with the passive version "the boy was bitten by the dog". If clauses do not have a 'done to' constituent which might have been made Subject (i.e. a Complement), they are considered 'middle' ('the boy slept').

Minor clauses do not select for *voice*, and therefore receive the value 'none'. Non-finite clauses are typed according to the POS tag of their Predicator. If the tag is VBG, *voice* is determined to be active; if the tag is VBN, *voice* is determined to be passive. Infinitive non-finite clauses receive the value 'none'.

Finite clauses with a final *tense* other than 'past' are labelled active. If the final *tense* is 'past', and the penultimate word of the verbal group is a form of the verb 'be', the clause is labelled passive, and the *tense* sequence is corrected accordingly.

Active clauses are then subtyped into true active and middle voices. Middle clauses are active clauses which have at least one complement.

4.11 Theme/Rheme

Theme and Rheme are textual roles. Theme refers to the order of information in a clause. The Theme/Rheme structure of a clause is often called Topic/Comment in other theories of grammar. The Theme is the departure point of information in a clause. The Rheme is the information not encompassed by the Theme.

The first Adjunct, Complement, Subject or Predicator that occurs is marked 'Topical Theme'. Any conjunctions that occur before it are marked 'Textual Theme', while any vocatives or finites that occur before it are marked 'Interpersonal Theme'. All other clause constituents are marked 'Rheme'.

5 Accuracy

Accuracy was checked using 100 clauses that had not been sampled while the script was being developed or debugged. Each clause was checked for constituency accuracy to the group and phrase rank — i.e., clause division and clause constituency were checked. Each of the eleven function structures were also checked: *clause class*, *status*, *mood*, *tense*, *polarity*, Subject, Finite, *voice*, Topical Theme, Textual Themes, Interpersonal Themes.

Two errors were found, both on the same clause. The *status* selection of an indirect projected speech clause was marked 'free' instead of 'bound'. This occurred because the projected clause was topicalised (i.e., it occurred before the projecting clause), which is rare for indirect speech. To correct this, the script must consider the presence or absence of quotation marks, which may be complicated by the slightly inconsistent attachment of punctuation in the Penn Treebank (Bies, 1995). Because the *status* of this clause was given as free, the clause incorrectly met the entry condition for the *mood type* system, causing the second error — a *mood type* selection of 'declarative' instead of 'none'.

In this somewhat small sample, 1198/1200 (99.83%) properties were correct, and 99% of clauses were annotated without any errors. The lack of plausible Adjunct subtyping may present problems for the accurate determination of Topical Theme in a more register varied sample, such as the Brown corpus.

Adjuncts should be subtyped into Modal Adjuncts (such as 'possibly'), Comment Adjuncts (such as 'unfortunately'), Conjunctive Adjuncts (such as 'however') and Experiential Adjuncts (such as 'quickly'). Only Experiential Adjuncts can be Topical Theme; if another kind of Adjunct occurs first it should be marked Interpersonal Theme (in the case of Modal and Comment Adjuncts), or Textual Theme (in the case of Conjunctive Adjuncts).

The Wall Street Journal corpus, which was the only section of the Penn Treebank available for this research, contains very few Mood, Comment or Conjunctive Adjuncts, so the extent of this problem could not be properly measured.

6 Conclusion

This work is approximately ten years overdue, in the sense that that is how long the resources required to perform it have existed. The motivations for it are even older: corpus linguistics has been a pillar of systemic functional linguistic research since it began, and raw text corpora are inadequate for many of the questions systemic functional linguistics asks (Honnibal, 2004). The first effort to convert the Penn Treebank to another representation was presented within months of the corpus's completion (Wang et al., 1994). Since then, treebanks have been converted to several grammatical theories (cf. (Lin, 1998; Frank et al., 2003; Watkinson and Manandhar, 2001)). It is unclear why SFG has been left behind for so long.

A corpus of over two million words of SFG constituency analysed text, annotated with the most important clause rank interpersonal and textual systems and functions, is now available. This is an important resource for linguistic research, the development of SFG parsers, and research into applying systemic linguistics to language technology problems.

Acknowledgements

I would like to thank Jon Patrick for his useful feedback on this paper. I also owe thanks to the many people who have helped me on my honours thesis, from which this paper is mostly drawn. Christian Matthiessen and Canzhong Wu have both been wonderful supervisors. Paul Nugent helped with the graphics used in this paper and my thesis, and proof-read with invaluable diligence. James Salter has shown remarkable patience over the last year and half while teaching me to program. Finally, the language technology research group at Sydney Uni have all contributed sound advice and interesting discussions on my work.

References

A. Bies. 1995. Bracketing guidelines for Treebank II style. Penn Treebank Project.

Maria Herke Couchman and Casey Whitelaw. 2003. Identifying interpersonal distance using systemic features. In *Proceedings of the first*

Australasian Language Technology Workshop (ALTW2003).

Anette Frank, Louisa Sadler, Josef van Genabith, and Andy Way, 2003. *From Treebank Resources To LFG F-Structures - Automatic F-Structure Annotation of Treebank Trees and CFGs extracted from Treebanks*. Kluwer, Dordrecht.

Michael A. K. Halliday. 1966. The concept of rank: a reply. *Journal of Linguistics*, 2(1):110–118.

Michael Halliday. 1969. Options and functions in the english clause. Brno Studies in English.

Michael Halliday. 1976. *System and Function in Language*. Oxford University Press, Oxford.

Michael Halliday. 1994. *Introduction to Functional Grammar 2nd ed.* Arnold, London.

Ruqaiya Hasan. 1987. The grammarian's dream: lexis as most delicate grammar. In Halliday and Fawcett, editors, *New developments in systemic linguistics: theory and description*. Pinter, London.

Matthew Honnibal. 2004. Design, creation and use of a systemic functional grammar annotated corpus. Macquarie University.

Dekang Lin. 1998. A dependency-based method for evaluating broad-coverage parsers. *Natural Language Engineering*, 4(2):97–114.

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the 1994 Human Language Technology Workshop*.

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

Christian Matthiessen. 1995. *Lexicogrammatical Cartography*. International Language Sciences Publishers, Tokyo, Taipei and Dallas.

Robert Munro. 2003. Towards the computational inference and application of a functional grammar. Sydney University.

Jong-Nae Wang, Jing-Shin Chang, and Keh-Yih Su. 1994. An automatic treebank conversion algorithm for corpus sharing. In *Meeting of the Association for Computational Linguistics*, pages 248–254.

S. Watkinson and S. Manandhar. 2001. In *Proceedings of the workshop on evaluation methodologies for language and dialogue systems*.