FROM CASES TO RULES AND VICE VERSA: ROBUST PRACTICAL PARSING WITH ANALOGY

Alex Chengyu Fang
Department of Phonetics and Linguistics
University College London
Wolfson House, 4 Stephenson Way
London NW1 2HE, England
alex@phon.ucl.ac.uk

This article describes the architecture of the Survey Parser and discusses two major components related to the analogy-based parsing of unrestricted English. Firstly, it discusses the automatic generation of a large declarative formal grammar from a corpus that has been syntactically analysed. Secondly, it describes analogy-based parsing that employs both the automatically learned rules and the database of cases to determine the syntactic structure of the input string. Statistics are presented to characterise the performance of the parsing system.

1 Introduction

As the title indicates, this article describes two components related to the parsing of unrestricted English. Firstly, it discusses the automatic generation of a large declarative formal grammar from a collection of pre-analysed sentences of English. Secondly, it describes a parsing methodology that employs both the automatically learned rules and the database of cases to determine the syntactic structure of the input string. The discussions will be based on the Survey Parser that has been implemented by the author (Fang 1996a), in the course of which some of the statistics will be presented to characterise the parsing approach to be reported here.

1.1 Background

In 1993-1996, the Survey of English Usage of University College London was engaged in the machine-aided syntactic analysis of the mega-word British Component of the International Corpus of English (ICE-GB; Greenbaum 1988 and 1996). The corpus comprises 600,000 words of transcribed speech and 400,000 words of writing. The analysis of the corpus included wordclass tagging and syntactic parsing. Each word in the corpus is assigned a contextually appropriate tag from a set of 270 grammatically possible tag-feature combinations. The parsing scheme specifies the analysis for the category names (covering the clause and the canonical phrases) and their syntactic functions such as subject, verb, and direct object. Both the tagging and parsing schemes were based on work by the TOSCA group of Nijmegen University, the Netherlands (Oostdijk 1991) but substantially modified for the project. ICE-GB initially used a parser that required certain amount of manual pre-processing of the input text. For instance, all cases of coordination had to be manually marked and indicated. The parser then produced all the possible analyses for each sentence, one of which was to be selected and modified if necessary as the correct representation of the constituent structure. About 70% of the corpus was parsed before it became clear in 1995 that the residue represented syntactic constructions beyond the capability of the parser. A parser that I had been developing since 1994 was used instead. Together with a Windows-based graphic tree editor, the parser completed the analysis of the corpus in 1996. It is now known as the Survey Parser, whose further development was supported in 1995 for two years by the Engineering and Physical Sciences Research Council, UK.

1.2 Design Requirements

Because of the specific needs arising from the analyses of the ICE corpus, the design of the Survey Parser was conditioned by the following requirements:

- *Speed* – The project required that input strings be batch parsed overnight so that researchers could supervise and modify the analyses the following day. Since each researcher was assigned several texts a time and since there were about six such people, the parser was typically required to batch parse about 20 texts at a time, which represented 40,000 words. In addition, because of the correction of wrong wordclass tags, researchers needed to frequently reparse individual sentences.
- *Robustness* – Because of the unsupervised batch parsing overnight of unrestricted English, the parser was required to be capable of handling situations where the input string represents linguistic constructions beyond the descriptive power of the formal grammar. In practice, the parser should be robust enough to produce a partial analysis when a complete parse could not be achieved.
- *One analysis per input string* – The experience was that it took shorter for the researcher to modify one incorrect analysis than to select from a number of possible analyses. This required the parser to produce one analysis that entails minimal manual intervention. To achieve this, the parser should either ensure that the first solution is a good one or be able to compute the best analysis from the competing ones.
- *Ability to adapt to new grammatical constructions* – The parser should be able to 'learn' and generalise about new grammatical constructions not described by the current grammar. In practice, this meant that the parser should be able to analyse a similar construction once the construction was manually analysed.

1.3 Analogy-Based Parsing

The parsing approach adopted in the Survey Parser seems to meet the above requirements. Briefly speaking, the parsing methodology may be roughly described as analogy-based, variously discussed under case-, explanation-, and example-based learning (Mitchell et al 1986; Minton 1988; van Harmelen and Bundy 1988; Knodner 1993). In the light of analogy-based approach to problem solving, solutions to old problems can be used for new but similar problems. In terms of parsing, this approach is conceptually very simple: given a database of input strings that have already been syntactically analysed, the parsing of a new string is seen as identifying a same or similar case in the database. Once the similarity is established, the analysis stored in the database is then transferred onto the new input string. This approach to parsing has been explored by Samuelsson and Rayner (1991), Neumann (1994), Samuelsson (1994), and Srinivas and Joshi (1995).

Such a parsing methodology requires a collection of syntactically analysed sentences as a case base and a mechanism to establish the similarity between the input string and one of the cases. Since a string may be represented as a sequence of words, a sequence of wordclass tags, or a sequence of grammatical phrases, there are three obvious options to establish such similarity: Two sentences are judged as structurally identical or similar if there is an exact match in terms of lexical items, wordclass tags, or phrases. Intuitively, these three matching criteria have different levels of generalisability or coverage. The use of wordclass tags as a measurement of similarity, for instance, should have a higher chance of finding a match than the use of lexical items because of the data sparsity problem that is typically related to word sequences, a problem that has been extensively addressed within the speech recognition community. The use of phrase sequence will, in turn, represent a more generalised model to measure sentence similarities. A related problem, however, is that a more general model tends to produce less reliable similarity indications. A match at the phrase level is less a guarantee than a match at the wordclass level that the two sentences are structurally similar or identical. The simple phrase sequence NP VP NP, for example, has at least three different syntactic structures according to the ICE-GB scheme. Thus a practical issue in analogy-based parsing is to increase the coverage of the case base while maintaining an acceptable degree of confidence that the retrieved syntactic structure is a good one.

1.4 An Overview of the Survey Parser

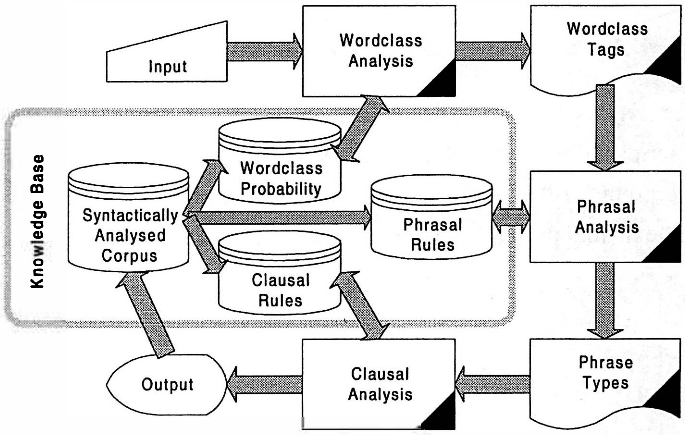


Figure 1: The major components in the Survey Parser

Generally put, the Survey Parser establishes analogy or similarity between the input string and a case in the knowledge base through a match at the phrase level. To ensure an acceptable degree of confidence, the phrase sequence is constrained by features inherited from the lexical properties of the head. The parser has two major components. The first is the syntactic knowledge base constructed on the basis of the syntactically analysed ICE-GB, from which we may automatically extract bi-gram wordclass transitional probability, phrasal rules anchored to wordclass tags or terminal symbols, and clausal rules anchored to phrase types. Each phrasal or clausal rewrite rule is associated to a tree structure. The parsing component comprises wordclass analysis (tagging), phrasal analysis, and clausal analysis. The stochastically selected tags serve as indexes that allow for the retrieval of a sub-tree for any phrase identified by the phrasal rules in a left-to-right longest match manner. At the stage of clausal analysis, phrase types are used as indexes to identify a similar clause and retrieve the tree structure as the proposal analysis for the input string. When the process fails to find an identical clause from the database, the sequence of phrases with their internal structures analysed is treated as an intermediate or partial analysis. The architecture described above is illustrated by Figure 1, where double arrows indicate system queries to components of the knowledge base.

The following discussions will be divided into three parts. I shall first of all describe the construction of the database, which is a process of automatic extraction of syntactic rules. I shall then describe the various analyses performed on the input string, including wordclass tagging and phrasal and clausal analyses. Finally, statistics will be presented to characterise the performance.

2 The Construction of the Syntactic Knowledge Base[1]

The syntactic knowledge base consists of two components: phrase structure (*PS*) rules and phrase structure cluster (*PSC*) rules. The purpose of *PS* rules is to analyse sequences of wordclass tags into grammatical phrases, while *PSC* rules mainly handles sequences of grammatical phrases and assigns the final hierarchical structure.

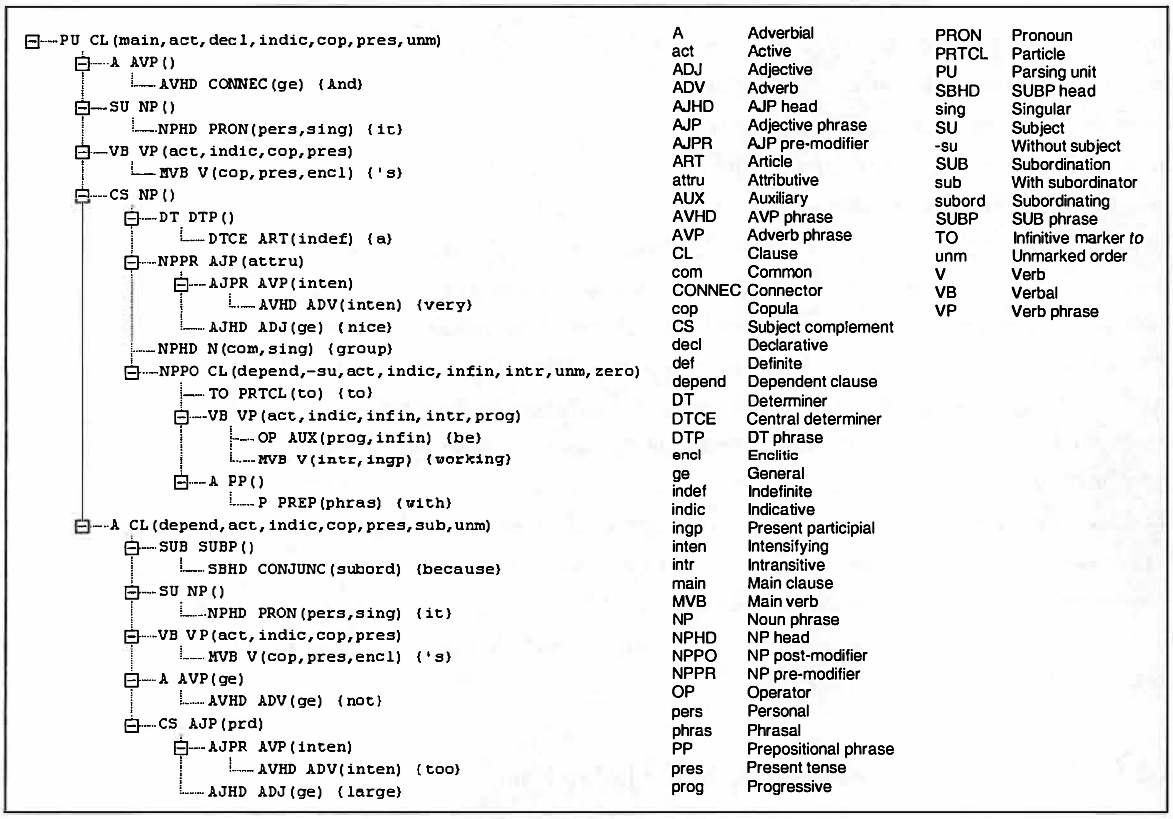
[1] See also Fang (1996c) for additional information.

2.1 Phrase Structure Rules

PS rules determine the analysis of wordclass sequences into phrases including noun phrases (NP), verb phrases (VP), adjective phrases (AJP), adverb phrase (AVP), and prepositional phrase (PP). The automatic generation of such rules is achieved by collecting all the tags as terminal symbols attributed to a particular phrase. Since the syntactic analyses of ICE-GB explicitly specify the boundaries of constituent structures as well as their syntactic functions, the extraction is a fairly straightforward matter. As a general rule, complementation and post-modification are not included. Thus, for instance, PS rules describing NPs all terminate at the head of the phrase; similarly those describing VPs all terminate at the main verb. Differently, however, PS rules for PPs cover the complete span of the phrase. Here is an example illustrating the extracting process.

[1] *And it's a very nice group to be working with because it's not too large*

The syntactic tree for [1] is graphically represented in Figure 2, where each constituent has two elements of description, the first being the name of syntactic function and the second that of category type. Thus, **SU NP()** is interpreted as "noun phrase functioning as clausal subject". As another example, **AVHD CONNEC(ge) {And}** is read as "lexical item *And* is a general connective and functions as the head of the adverb phrase".



```
⊟---PU CL(main,act,decl,indic,cop,pres,unm)
    ⊟---A AVP()
    |   └---AVHD CONNEC(ge) {And}
    ⊟---SU NP()
    |   └---NPHD PRON(pers,sing) {it}
    ⊟---VB VP(act,indic,cop,pres)
    |   └---MVB V(cop,pres,encl) {'s}
    ⊟---CS NP()
    |   ⊟---DT DTP()
    |   |   └---DTCE ART(indef) {a}
    |   ⊟---NPPR AJP(attru)
    |   |   ⊟---AJPR AVP(inten)
    |   |   |   └---AVHD ADV(inten) {very}
    |   |   └---AJHD ADJ(ge) {nice}
    |   ├---NPHD N(com,sing) {group}
    |   ⊟---NPPO CL(depend,-su,act,indic,infin,intr,unm,zero)
    |       ├---TO PRTCL(to) {to}
    |       ⊟---VB VP(act,indic,infin,intr,prog)
    |       |   ├---OP AUX(prog,infin) {be}
    |       |   └---MVB V(intr,ingp) {working}
    |       ⊟---A PP()
    |           └---P PREP(phras) {with}
    ⊟---A CL(depend,act,indic,cop,pres,sub,unm)
        ⊟---SUB SUBP()
        |   └---SBHD CONJUNC(subord) {because}
        ⊟---SU NP()
        |   └---NPHD PRON(pers,sing) {it}
        ⊟---VB VP(act,indic,cop,pres)
        |   └---MVB V(cop,pres,encl) {'s}
        ⊟---A AVP(ge)
        |   └---AVHD ADV(ge) {not}
        ⊟---CS AJP(prd)
            ⊟---AJPR AVP(inten)
            |   └---AVHD ADV(inten) {too}
            └---AJHD ADJ(ge) {large}
```

A	Adverbial	PRON	Pronoun
act	Active	PRTCL	Particle
ADJ	Adjective	PU	Parsing unit
ADV	Adverb	SBHD	SUBP head
AJHD	AJP head	sing	Singular
AJP	Adjective phrase	SU	Subject
AJPR	AJP pre-modifier	-su	Without subject
ART	Article	SUB	Subordination
attru	Attributive	sub	With subordinator
AUX	Auxiliary	subord	Subordinating
AVHD	AVP phrase	SUBP	SUB phrase
AVP	Adverb phrase	TO	Infinitive marker *to*
CL	Clause	unm	Unmarked order
com	Common	V	Verb
CONNEC	Connector	VB	Verbal
cop	Copula	VP	Verb phrase
CS	Subject complement		
decl	Declarative		
def	Definite		
depend	Dependent clause		
DT	Determiner		
DTCE	Central determiner		
DTP	DT phrase		
encl	Enclitic		
ge	General		
indef	Indefinite		
indic	Indicative		
ingp	Present participial		
inten	Intensifying		
intr	Intransitive		
main	Main clause		
MVB	Main verb		
NP	Noun phrase		
NPHD	NP head		
NPPO	NP post-modifier		
NPPR	NP pre-modifier		
OP	Operator		
pers	Personal		
phras	Phrasal		
PP	Prepositional phrase		
pres	Present tense		
prog	Progressive		

Figure 2: The tree structure for Example [1]

From the analysis of [1], PS rules for the five major phrases can be extracted and stored in the syntactic knowledge base. Each rule comprises a sequence of wordclass tags and is associated to a constituent structure that specifies the analysis of the sequence. Table 1 illustrates such rules from [1].

Phrase	*Rule*	*Associated constituent structure*	*Example*

AJP	ADV(inten)
	ADJ(ge)
		⊟— AJP(prd)
		⊟—AJPR AVP(inten)
		└—AVHD ADV(inten) {-}
		└—AJHD ADJ(ge) {-}
			too
			large

AVP	CONNEC(ge)	⊟— AVP() └—AVHD CONNEC(ge) {-}	furthermore
	ADV(ge)	⊟— AVP(ge) └—AVHD ADV(ge) {-}	briefly
NP	PRON(pers,sing)	⊟— NP() └—NPHD PRON(pers,sing) {-}	it

	ART(indef)
	ADV(inten)
	ADJ(ge)
	N(com,sing)
		⊟— NP()
		⊟—DT DTP()
		└—DTCE ART(indef) {-}
		⊟—NPPR AJP(attru)
		⊟—AJPR AVP(inten)
		└—AVHD ADV(inten) {-}
		└—AJHD ADJ(ge) {-}
		└—NPHD N(com,sing) {-}
			a
			very
			nice
			group

PP	PREP(phras)	⊟— PP() └—P PREP(phras) {-}	with
VP	V(cop,pres,encl)	⊟— VP(act,indic,cop,pres) └—MVB V(cop,pres,encl) {-}	's

	AUX(prog,infin)
	V(intr,ingp)
		⊟— VP(act,indic,infin,intr,prog)
		├—OP AUX(prog,infin) {-}
		└—MVB V(intr,ingp) {-}
			be
			working

Table 1: PS rules automatically extracted from [1]

2.2 Phrase Structure Cluster Rules

The second component of the syntactic knowledge base deals with phrase structure clusters and superimposes the hierarchical structure of this cluster. In most of the cases, these clusters correspond to the conventional clause, but occasionally they represent co-ordinated or juxtaposed phrases. Again, such rules may be automatically extracted from a set of pre-analysed sentences. For example, the syntactic analysis of [1] yields one PSC rule as represented in Table 2:

Type	*PS Cluster*	*Associated Tree*
Clause	AVP NP VP:cop NP TO VP:intr:infin PP:ps CONJUNC:subord NP VP:cop AVP AJP	⊟—PU CL(main,act,decl,indic,cop,pres,unm) ⊞—A AVP() ⊞—SU NP() ⊞—VB VP(act,indic,cop,pres) ⊟—CS NP() ⊟—NPPO CL(depend,-su,act,indic,infin,intr,unm,zero) ├—TO PRTCL(to) {to} ⊞—VB VP(act,indic,infin,intr,prog) ⊞—A PP() ⊟—A CL(depend,act,indic,cop,pres,sub,unm) ⊞—SUB SUBP() ⊞—SU NP() ⊞—VB VP(act,indic,cop,pres) ⊞—A AVP(ge) ⊞—CS AJP(prd)

Table 2: A phrase cluster automatically extracted from [1]

As mentioned at the beginning, analogy-based parsing has two practical issues: confidence and coverage. Confidence is the system assurance that the retrieved tree structure for the input string is a good one while coverage is the adaptation of the indexed cases so that they are useful to as many structural variations as possible. In the Survey Parser, confidence is maintained through the use of feature constraints and the increase of coverage is achieved through the identification and removal of non-obligatory syntactic elements.

Feature Constraints

To ensure the correct association between the PS cluster and the corresponding tree, some of the phrase types are normally described or restricted with features. This typically applies to VPs, whose sub-categorisation determines the analysis of their complements. For example, the phrase cluster NP VP PP may have at least two different analyses for the complementing PP: as subject complement if the VP is copula and as adverbial if the VP is intransitive. In the case of a non-finite VP, the very same phrase cluster needs to be analysed as a noun phrase post-modified by a non-finite clause, e.g., *countries pressurised by the decision* and *countries voting against the decision.* Feature constraints inherited from the main verb help to dissolve such ambiguities. In Table 2, the first and second VPs are described by **`cop`**, a feature name meaning *copula.* This feature ensures that the complementing NPs are correctly analysed as subject complement (**`cs`**). The other two constraint features for VPs are **`intr`** (intransitive) and **`tr`** (transitive). Non-finite VPs are described with additional features to indicate their forms, e.g., **`infin`** for infinitive, **`edp`** for past participle, and **`ingp`** for present participle.

Non-obligatory Elements

Non-obligatory elements include AVPs and PPs that do not complement any particular VPs. They are called non-obligatory in the sense that their removal does not affect the overall syntactic structure of the sentence. In order to maximise the coverage of phrase cluster rules, such elements are removed. The example in Table 2, for instance, is in fact written as `NP VP:cop NP TO VP:intr:infin PP:ps CONJUNC:subord NP VP:cop AJP.`

Adverbial clauses may also be treated as non-obligatory and, indeed, they are probably the most active constructions that contribute to the complexity of the clause. There is very good reason to expect a greatly increased coverage if such clauses could be treated separately and removed from the host clause. For the time being, however, this has not been implemented in the Survey Parser.

3 Parsing with PS and PSC Rules

The Survey Parser has three major modules that handle (1) assigning a wordclass tag to each item in the input string, (2) chunking the tags into a *PSC*, and (3) querying in the knowledge base for possible analyses for this *PSC*.

3.1 The Analysis of Wordclasses

The Survey Parser currently uses AUTASYS (Fang and Nelson 1994; Fang 1996b) as a pre-processor that tokenises the input string into lexical items and then assigns one wordclass tag to each of the tokens. This tagger has a probabilistic backbone supported by a list of rules in order to achieve the informationally rich tagset designed for the ICE-GB project. The tagset features 22 general wordclasses with around 70 descriptive features, totalling about 270 grammatically possible tags (Fang 1994; Greenbaum and Ni 1994). The descriptive features represent a detailed system of lexical sub-categorisation critical for parsing with wide-coverage lexicalised grammars (Briscoe and Carroll 1997). Consider

[2] *The search menu in the Circulation module may make additional search methods available to library staff.*

AUTASYS assigns one ICE tag to each of the lexical items in [2] and the result is illustrated in Table 3. It is worth noting that the tagging process provides the maximum grammatical information at this stage. For example, all compound nouns have already been marked up with 'ditto tags' that carry sequential numbers to indicate the boundary of the compound. The grammatical features related to the compound are selected according to the head.

As a result, *search* in the compound noun *search methods* is tagged as a plural common noun, the first in the two-item sequence. Lexical verbs are also analysed for detailed sub-categorisations and, as Figure 3 shows, there are 7 different types of verbs in the ICE tagging scheme. The verb *make* in [2], for instance, is tagged as infinitive ditransitive though it should be correctly analysed as complex transitive complemented by both a NP and an AJP.

The	**ART(def)**
search	**N(com,sing):1/2**
menu	**N(com,sing):2/2**
in	**PREP(ge)**
the	**ART(def)**
Circulation	**N(com,sing):1/2**
module	**N(com,sing):2/2**
may	**AUX(modal,pres)**
make	**V(ditr,infin)**
additional	**NUM(ord)**
search	**N(com,plu):1/2**
methods	**N(com,plu):2/2**
available	**ADJ(ge)**
to	**PREP(ge)**
library	**N(com,sing):1/2**
staff	**N(com,sing):2/2**
.	**PUNC(per)**

Table 3: Grammatical tagging of [2]

As mentioned in Section 2.2 where feature constraints are discussed, VPs are described as copula, intransitive, and transitive only. Detailed transitivity information for the verb is mainly for passivised VPs, which undergo the following valency shifts: mono-transitive→intransitive, complex-transitive→copula, ditransitive→mono-transitive. The transitivity of a passivised complex-transitive VP, for example, is shifted to that of copula in order to cater for the analysis of the complementiser as subject complement, e.g., *The home front was kept ignorant of the reality.* Detailed verb transitivity sub-categorisation is of great use when the system fails to find a global analysis for the input string and has to label syntactic functions from limited context.

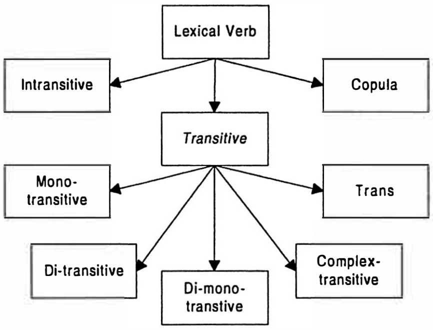
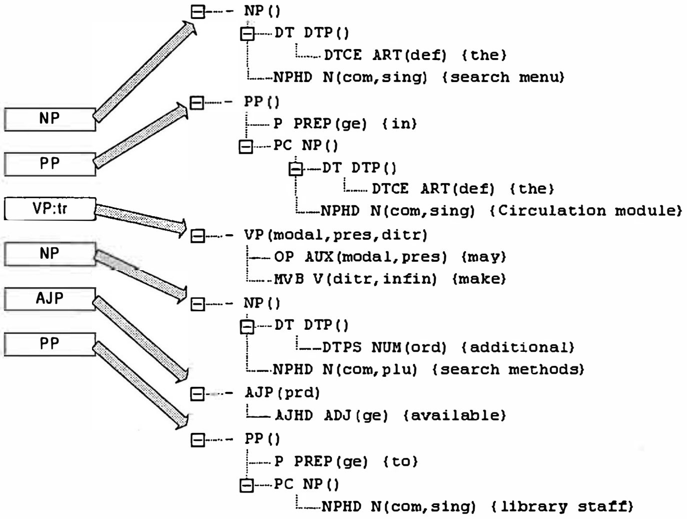


Figure 3: Subcategorisation of verbs

3.2 The Analysis of Phrases

The input string, as a sequence of wordclass tags, is then processed at the phrase level according to the *PS* rules, which are applied deterministically on a left-to-right longest match heuristic. When applied, these *PS* rules chunk the input string into a cluster of *PS*s, with feature information about the head. They also assign phrase types, boundaries, and internal structures to tag sequences that have a direct match in the PS rule base. Analysed at this level, the input string is represented as a cluster of syntactic phrases, each of which is now associated to a sub-tree. As demonstrated by Figure 4, the sub-trees already present a neat representation of the constituent structure of the phrases. Note that the VP is described by a feature (**tr**, meaning transitive) inherited from the corresponding wordclass tag except that there is no further distinction of sub-categorisation for transitive verbs (Figure 3). What still remains uncertain at this stage is the syntactic function to be determined by the PSC rules.



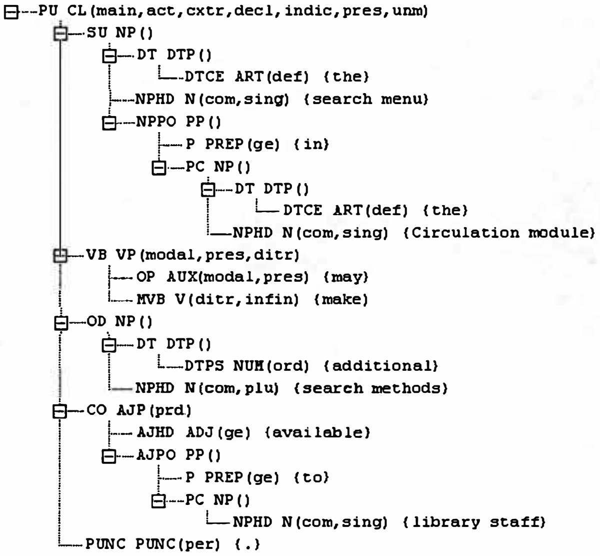
```
NP()
  DT DTP()
    DTCE ART(def) {the}
  NPHD N(com,sing) {search menu}
PP()
  P PREP(ge) {in}
  PC NP()
    DT DTP()
      DTCE ART(def) {the}
    NPHD N(com,sing) {Circulation module}
VP(modal,pres,ditr)
  OP AUX(modal,pres) {may}
  MVB V(ditr,infin) {make}
NP()
  DT DTP()
    DTPS NUM(ord) {additional}
  NPHD N(com,plu) {search methods}
AJP(prd)
  AJHD ADJ(ge) {available}
PP()
  P PREP(ge) {to}
  PC NP()
    NPHD N(com,sing) {library staff}
```

Figure 4: Input string as a PS cluster with associated sub-trees

3.3 The Analysis of Clauses

As a final step, the input string as a cluster of phrase structures is then queried in the database of PSC rules to see if an identical sequence can be located. With a positive feedback from the database, the associated tree structure for that sequence in the database is retrieved and used to specify the labelling and the attachment of the phrases of the input string. In our example, the parser determined that the PSC of the input string was the same as that of [3], already analysed and stored in the database:

[3]   *The cellular anatomy of the peripheral nervous system renders it vulnerable to injury.*

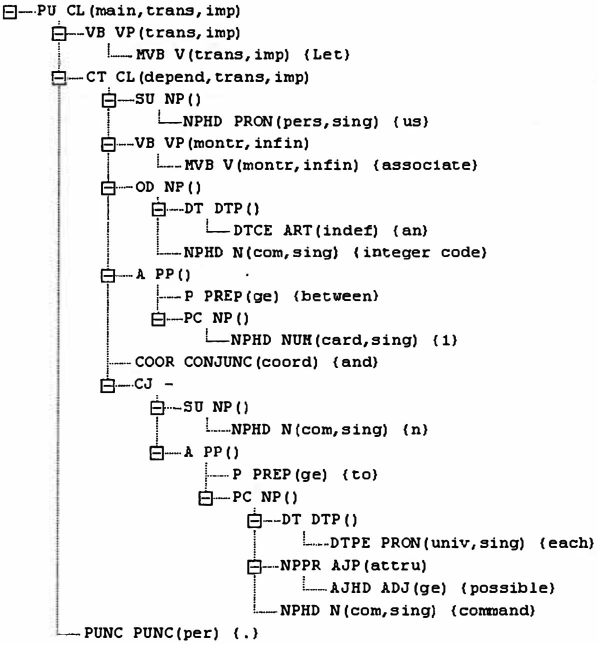


```
⊟--PU CL(main,act,cxtr,decl,indic,pres,unm)
    ⊟--SU NP()
    |   ⊟--DT DTP()
    |   |   └--DTCE ART(def) {the}
    |   ├--NPHD N(com,sing) {search menu}
    |   ⊟--NPPO PP()
    |       ├--P PREP(ge) {in}
    |       ⊟--PC NP()
    |           ⊟--DT DTP()
    |           |   └--DTCE ART(def) {the}
    |           └--NPHD N(com,sing) {Circulation module}
    ⊟--VB VP(modal,pres,ditr)
    |   ├--OP AUX(modal,pres) {may}
    |   └--MVB V(ditr,infin) {make}
    ⊟--OD NP()
    |   ⊟--DT DTP()
    |   |   └--DTPS NUM(ord) {additional}
    |   └--NPHD N(com,plu) {search methods}
    ⊟--CO AJP(prd)
    |   ├--AJHD ADJ(ge) {available}
    |   ⊟--AJPO PP()
    |       ├--P PREP(ge) {to}
    |       ⊟--PC NP()
    |           └--NPHD N(com,sing) {library staff}
    └--PUNC PUNC(per) {.}
```

Figure 5: The final analysis of [2]

Accordingly, the parser retrieved the analysis for [3] and superimposed it on [2]. The final analysis is shown in Figure 5. In this particular example, the parser successfully retrieved the correct tree structure for the input string. The incorrect sub-categorisation of the verb *make* as ditransitive at the stage of tagging did not prevent the parser from correctly labelling the sentence-final AJP as *object complement* (**CO**), suggesting the possibility of a post-parsing correction of wordclass tags.

Partial Analysis



```
⊟--PU CL(main,trans,imp)
    ⊟--VB VP(trans,imp)
    |   └--MVB V(trans,imp) {Let}
    ⊟--CT CL(depend,trans,imp)
    |   ⊟--SU NP()
    |   |   └--NPHD PRON(pers,sing) {us}
    |   ⊟--VB VP(montr,infin)
    |   |   └--MVB V(montr,infin) {associate}
    |   ⊟--OD NP()
    |   |   ⊟--DT DTP()
    |   |   |   └--DTCE ART(indef) {an}
    |   |   └--NPHD N(com,sing) {integer code}
    |   ⊟--A PP()
    |   |   ├--P PREP(ge) {between}
    |   |   ⊟--PC NP()
    |   |       └--NPHD NUM(card,sing) {1}
    |   ├--COOR CONJUNC(coord) {and}
    |   ⊟--CJ -
    |       ⊟--SU NP()
    |       |   └--NPHD N(com,sing) {n}
    |       ⊟--A PP()
    |           ├--P PREP(ge) {to}
    |           ⊟--PC NP()
    |               ⊟--DT DTP()
    |               |   └--DTPE PRON(univ,sing) {each}
    |               ⊟--NPPR AJP(attru)
    |               |   └--AJHD ADJ(ge) {possible}
    |               └--NPHD N(com,sing) {command}
    └--PUNC PUNC(per) {.}
```

Figure 6: A partial analysis

If a global analysis cannot be achieved, the parser will enter a *fallback mode*, where all the non-obligatory PPs are removed from the phrase cluster and a second attempt is made to find a match. When successful, the parser will paste the removed PPs back into the host clause. Failure in the fallback mode will then put the parser in the *partial mode*, where all the associated sub-trees in the phrase cluster are written out as a partial analysis. The boundaries and the internal structures of the component phrases have already been analysed and labelled. The parser also naively assigns missing names guess-estimated from neighbouring phrases.[2] Figure 6 is an example of a partial analysis. The major cause for the failure to find a global analysis was the exclusion of the coordinating conjunction *and* from the PP *between 1 and n*. The parser automatically inserted a **CT CL** node after the initial VP, indicating that what follows is a non-finite clause with an overt subject. This was correctly achieved

[2] As Fang (forthcoming) argues, it is feasible to determine the syntactic functions of, for instance, prepositional phrases according to lexical descriptions. It is also possible to label the syntactic functions of non-finite clauses in a similar fashion.

through the sub-categorisation of the antecedent VP as **`trans`**. Limited context and verb sub-categorisation information also enabled the correct labelling of the subject, the verb, and the direct object of the dependent clause.

4 Evaluation

In this section, I report empirical tests carried out to evaluate the performance of the Survey Parser. In particular, these tests were designed to indicate the coverage of the extracted grammar in terms of PS and PSC rules, labelling precision, the accuracy of analysis according to human judgements, and finally the processing speed.

4.1 Evaluating the Coverage of Phrase Structure Rules

The syntactic knowledge base currently makes use of about 50,000 syntactically analysed utterances from ICE-GB. Table 4 presents statistics about the rule extractions from the set.

Type	*AJP*	*AVP*	*NP*	*PP*	*VP*	*PSC*
No	103	49	3,695	8,974	987	178,045

Table 4: A summary of PS and PSC rules extracted from ICE-GB

To estimate the coverage of PS rules automatically extracted from ICE-GB, the analysed corpus was divided into 10 equal sets (1-10) of about 70,000 words each, with each individual set used as test data and the rest as training data in a rotating manner. Precautions were taken to make sure that the test set did not make up the training data. Figure 7 displays the coverage of VP rules when tested by the training data of varying sizes. The *Y*-axis represents the coverage (0.0-1.0), though the graph only displays a region of .95 to 1.0 as all the figures were higher than .95. The *X*-axis represents the size of the training data, which is an increment of one sample up to nine samples. Training data of one sample in size yields a coverage of over .95 for all the four testing samples. The coverage constantly rises with the increase of the training data size. The nine-sample training data produced a coverage of over .98 for the four testing samples used in the experiment. The start of plateau visibly rests on eight-sample training data. For discussions on the forms and sub-categorisations of these verbs, see Fang (1997).

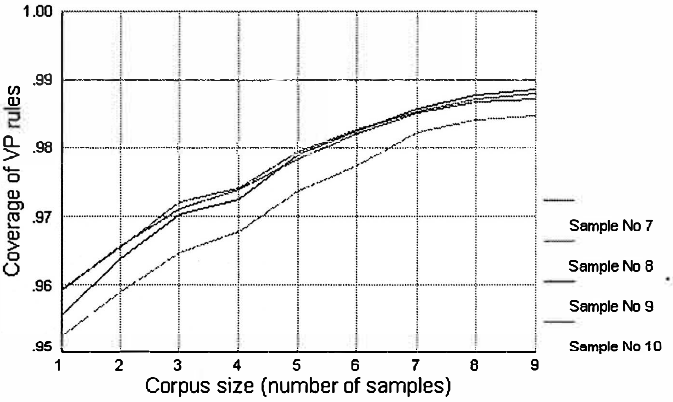


Figure 7: The coverage of VP rules extracted from ICE-GB

4.2 Evaluating the Coverage of Phrase Structure Cluster Rules

The corpora used for the evaluation included Air Travel Information System (ATIS), the Wall Street Journal (WSJ), and the Survey of English Usage Corpus (SEU; Fang and Nelson 1994), all different in terms of subject matter, style, and national variety. A total of 3,654 sentences were randomly selected to test the coverage of the PSC rules in order to estimate the probability of the input string as a phrase structure cluster to have a direct match in the database. According

Corpus	*Sent No.*	*Coverage*
ATIS	654	64.9%
WSJ	2,000	60.0%
SEU	1,000	60.3%

Table 5: The recall rates

to Table 5, such probability is 60% for test sets from WSJ and SEU. The ATIS set had a higher percentage mainly because of the relatively shorter sentence length in this corpus than the other two. Input strings that cannot be described by the PSC rules were given partial analyses.

4.3 Evaluating the Labelling Precision

A set of 60 AMALGAM sentences[3] from a computer manual were used to measure the precision of labelling by the Survey parser. A scoring program for evaluating the performance of speech recognition systems was used that measures not only the correct and wrong labels but also insertion and deletion rates. As Table 6 indicates, 90.1% of the constituents for the 60 sentences were correctly labelled by the Survey Parser. With wrong labels and deletion and insertion rates considered, the overall precision rate was 86.9%.

Sent No.	60	
Constituent No.	4,597	
Correct	4,142	90.1%
Wrong	272	5.9%
Deletion	183	4.0%
Insertion	147	
Overall	86.9%	

Table 6: Labelling precision

4.4 Evaluating the Accuracy of Analysis

Finally, the performance of the parser was subjected to the strictest human evaluation where an input string was judged to be wrong with a single parsing error in terms of labelling, attachment, or tokenisation. For this purpose, the test used a set of 117 dictionary definitions extracted from the Longman Dictionary of Contemporary English (see Briscoe and Carroll, 1991). Table 7 summarises the results. Of these input strings, 84 were fully parsed, a coverage of 71.8%. Of the fully parsed strings, 77 were correctly labelled and attached, a precision rate of 65.8% of the total number of input strings. It is significant that the majority of the fully analysed strings (91.7%) were correct even according to the strictest requirements, indicating a high level of system confidence that the proposed analysis is a good one.

Definition No.	117	
Full analysis	84	71.8%
Correct analysis	77	65.8%

Table 7: Accuracy of analysis

4.5 Evaluating the Processing Speed

Two sub-language corpora were used to measure the processing speech of the Survey Parser. One is a corpus of the English for science and technology collected at the Shanghai Jiao Tong University (JDEST; Huang 1991) and the other a corpus of the English of computer science collected at the Hong Kong University of Science and Technology (HKUST; Fang 1992). The statistics summarised in Table 8 were obtained with Dell OptiPlex Gxa. The tagging module, according to the test, ran at a speed of 6,012 words per second. The parsing module was able to process 177 words per second.

Corpus Source	*No. of words*	*No. of sentences*	*Tagging* *No. of seconds*	*Parsing* *No. of seconds*
JDEST	104,014	4,692	18	540
HKUST	70,331	4,297	11	443
Total	174,345	8,989	29	983

Table 8: The processing speed of the Survey Parser

[3] Available at http://www.scs.leeds.ac.uk/amalgam/amalgam/corpus/tagged/raw/ipsm_raw.html. The output from the Survey Parser for this set of sentences, manually checked and corrected by me, are available at http://www.scs.leeds.ac.uk/amalgam/amalgam/corpus/parsed/ice.html

5 Concluding Remarks

In this article, I have described the architecture of the Survey Parser as well as the construction of the syntactic knowledge base that comprises PS and PSC rules automatically extracted from a syntactically analysed corpus, ICE-GB. I then reported evaluation statistics that characterise the performance of analogy-based parsing. They indicate that while the clause structure rules have a coverage of 60%, the grammar has a high coverage in terms of phrase structures as the statistics for the verb phrase indicated. One conclusion we can draw here is that it is indeed feasible to automatically generalise a comprehensive formal grammar from a syntactically analysed corpus the size of ICE-GB and to apply it to large-scale practical parsing. A second conclusion is that analogy-based parsing enjoys a high degree of analysis precision, with over 90% of the constituents correctly labelled. When subjected to human inspection, 91.7% of the complete parses were found to be correct. Other observed advantages include high parsing speed and the ability to produce an analysis for every input string. A true strength of analogy-based parsing is its intrinsic ability to learn over the acquisition of new phrase structure clusters. Since the formal grammar can be automatically learned, the parser can easily adapt itself to new constructions. Unlike probabilistic grammars, the automatically constructed grammar can be visually supervised and manipulated because of its declarative nature. The empirical tests suggested that the analogy-based parsing reported here is capable of the design requirements outlined at the beginning of the article.

As can be envisaged at this stage, the future work on the analogy-based parser will focus on methods to increase the coverage of the clause cluster rules. I have already mentioned the removal of non-obligatory elements such as AVP and PP in order to increase the coverage. Another effective method, yet to be tested, is the segmentation of the input string into component finite clauses, which are then parsed individually and glued back into the host clause. The syntactic functions of the clauses can be reliably assessed independent of the host clause because of the conjunction markers. The key process to achieve this is an algorithm that automatically, and reliably, identifies the boundaries of component clauses.

Acknowledgement

The author would like to thank the three anonymous reviewers for their valuable comments on an earlier version. This article was supported in part by Engineering and Physical Sciences Research Council, UK, Grant No GR/L81406.

References

Briscoe, E., and J. Carroll. 1991. *Generalised Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars*. Technical Report No. 224. University of Cambridge.

Briscoe, E., and J. Carroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing, Washington, DC.* pp 356-363.

Fang, A.C. 1992. Building a Corpus of the English of Computer Science. In *English Language Corpora: Design, Analysis and Exploitation*, ed. by Aarts, de Haan and Oostdijk. Amsterdam: Rodopi.

Fang, A.C. 1994. ICE: Applications and Possibilities in NLP. In *Proceedings of International Workshop on Directions of Lexical Research, 15-17 August 1994, Beijing.* pp 23-44.

Fang, A.C. 1996a. The Survey Parser: Design and Development. In S. Greenbaum. pp 142-160.

Fang, A.C. 1996b. Grammatical Tagging and Cross-Tagset Mapping. In S. Greenbaum. pp 110-124.

Fang, A.C. 1996c. Automatically Generalising a Wide-Coverage Formal Grammar. In *Synchronic Corpus Linguistics*, ed. by C. Percy, C. Meyer, and I. Lancashire, Amsterdam: Rodopi. pp 207-222.

Fang, A.C. 1997. Verb Forms and Subcategorisations. In *Oxford Literary and Linguistic Computing, 12:4.*

Fang, A.C. forthcoming. A Lexicalist Approach towards the Automatic Determination for the Syntactic Functions of Prepositional Phrases. To appear in *the Journal of Natural Language Engineering.*

Fang, A.C., and G. Nelson. 1994. Tagging the Survey Corpus: a LOB to ICE experiment using AUTASYS. *ALLC Literary & Linguistic Computing,* 9:2. pp 189-194.

Greenbaum, S. 1988. A Proposal for an International Computerized Corpus of English. In *World Englishes 7, 315.*

Greenbaum, S. 1996. *The International Corpus of English.* Oxford: Oxford University Press.

Greenbaum, S., and Ni, Y. 1994. Tagging the British ICE Corpus: English Word Classes. In *Corpus-Based Research into Language*, ed. by N. Oostdijk and P. de Haan. Amsterdam: Rodopi. pp 33-45.

Huang, RJ. 1991. *A Technical Report of the JDEST Corpus Tagging Project.* Shanghai: Shanghai Jiao Tong University.

Kolodner, J. 1993. *Case-Based Reasoning.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Minton, S. 1988. Quantitative results concerning the utility of explanation-based learning. In *Proceedings of 7th AAAI Conference, Saint Paul, Minnesota,* pp 564-569.

Mitchell, T., R. Keller, and S. Kedar-Carbelli. 1986. Explanation-based generalization: A unifying view. In *Machine Learning 1:1,* pp 47-86.

Neuman, G. 1994. Application of explanation-based learning for efficient processing of constraint-based grammars. In *10th IEEE Conference on Artificial Intelligence for Applications, San Antonio, Texas.*

Oostdijk, N. 1991. *Corpus Linguistics and the Automatic Analysis of English.* Amsterdam: Rodopi.

Samuelsson, C. 1994. Grammar specialization through entropy thresholds. In *32nd Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico.*

Samuelsson, C. and M. Rayner. 1991. Quantitative evaluation of explanation-based learning as an optimization tool for large-scale natural language system. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence, Sydney, Australia.*

Srinivas, B. and A. Joshi. 1995. Some novel applications of explanation-based learning to parsing lexicalized tree-adjoining grammars. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95), Morgan Kaufmann, San Francisco, 1995.*

van Harmelen, F. and A. Bundy. 1988. Explanation-based generalisation = Partial evaluation. In *Artificial Intelligence, 36.* pp 401-412.