Finite State Segmentation of Discourse into Clauses

Eva Ejerhed
Department of Linguistics
University of Umea
S-90187 Umea Sweden
ejerhed@ling.umu.se

Abstract. The paper presents background and motivation for a processing model that segments discourse into units that are simple, non-nested clauses, prior to the recognition of clause internal phrasal constituents, and experimental results in support of this model.

One set of results are derived from a statistical reanalysis of the Swedish empirical data in [18] concerning the linguistic structure of major prosodic units.

The other set of results is derived from experiments in segmenting part-of-speech annotated Swedish text corpora into clauses, using a new clause segmentation algorithm. The clause segmented corpus data is taken from the Stockholm Umea Corpus (SUC), 1 M words of Swedish texts from different genres, part-of-speech annotated by hand, and from the Umea corpus DAGENS INDUSTRI 1993 (DI93), 5 M words of Swedish financial newspaper text, processed by fully automatic means consisting of tokenizing, lexical analysis, and probabilistic POS tagging.

The results of these two experiments show that the proposed clause segmentation algorithm is 96% correct when applied to manually tagged text, and 91% correct when applied to probabilistically tagged text.

1 Introduction

The problem of recognizing simple clauses and segmenting discourse into such units is beginning to emerge as a problem area in its own right as evidenced by [1] and [2]. This is a problem area in which I have been interested since the early eighties ([6], [7], [9], [8], [10], [11]). Simple clauses share with simple noun phrases the property of being non-recursive ([4],[14]), but the problems of defining, and recognizing, simple clauses are harder than those of defining and recognizing simple noun phrases.

However, the problem of clause segmentation of written and spoken discourse is worth addressing because of the many applications that successful clause segmentation algorithms could be used for. The following are some important application areas:

- speech synthesis: improved prosody in text-to-speech systems
- speech recognition: automatic segmentation of input to speech recognizers
- text analysis: preprocessing input to parsers for orthographic sentences
- machine translation: clauses as translation units
- knowledge acquisition: databases of lexical preferences, SVO triplets, etc. databases of facts, events, etc.

2 Clause segmentation of read speech

2.1 Background

The study presented in [18] is based on read speech. Four adult Swedish speakers, two women and two men, read the same text, which had a length of 878 words, excluding punctuations. The study combines the following analytic data contributed by the three authors:

1) algorithmically derived information about the location of clause boundaries (C) in the text, based on [9];

2) acoustically derived information about the location of intonation unit boundaries (I), and silent intervals (S), based on [12];

3) perceptually derived information about perceived pauses (P) using two judges, based on [16], [17].

The exact definitions of clause, intonation unit, silent interval and perceived pause that were used in analyzing the data of [18], and the details of the procedures whereby these units were identified are described in the respective references provided above.

Table 1 taken from [18] is a sample that shows the distribution of C, I, P, S for the four speakers:

© 1996 E. Ejerhed
Proceedings of the ECAI 96 Workshop
Extended Finite State Models of Language
Edited by A. Kornai.

Table 1. Excerpt of data for all four speakers.

Text	F1	F2	M1	M2
I	∅	∅	∅	∅
stadens	∅	∅	∅	∅
fästning,	CI∅∅	CI∅∅	CI∅∅	C∅∅∅
vars	∅	∅	∅	∅∅∅S
normandiska	∅	∅	∅	∅
torn	∅	∅	∅	∅
ännu	∅	∅	∅	∅
reser	∅	∅	∅	∅
sig	∅	∅	∅	∅
oanfrätt	∅	∅	∅	∅
över	∅	∅	∅	∅
den	∅	∅	∅	∅
låga	∅	∅	∅	∅
bebyggelsen	∅	∅∅P∅	∅	∅
i	∅	∅	∅	∅
dalen,	CIPS	CIPS	CIPS	CIPS
befann	∅	∅	∅	∅
sig	∅	∅	∅	∅
prins	∅	∅	∅	∅
Edvard,	CIPS	CIPS	CIPS	CIP∅
kungens	∅	∅	∅	∅
son	∅IPS	∅IPS	∅IPS	∅IPS
och	∅	∅	∅	∅
känd	∅	∅	∅	∅
som	∅	∅	∅	∅
Långskank.	CIPS	CIPS	CIPS	CIPS
Edvard	∅	∅	∅∅PS	∅∅P∅
kommenderade	∅I∅∅	∅I∅∅	∅∅P∅	∅
3.000	∅	∅	∅	∅
beridna	∅	∅	∅	∅
riddare	∅	∅	∅∅P∅	∅
och	∅	∅	∅	∅
soldater.	CIPS	CIPS	CIPS	CIPS
Baronernas	∅	∅	∅	∅
här,	CIPS	C∅∅∅	C∅PS	C∅∅∅
som	∅	∅	∅	∅
leddes	∅	∅	∅	∅
av	∅	∅	∅	∅
Simon	∅	∅	∅	∅
de	∅	∅	∅	∅
Montfort,	CIPS	CIPS	CIPS	C∅P∅
earl	∅	∅	∅	∅
av	∅	∅	∅	∅
Leicester,	CIPS	CIPS	CIPS	CIP∅
var	∅	∅	∅	∅
lägrad	∅	∅	∅	∅
i	∅	∅	∅	∅
Fletching,	CIPS	CIPS	CIPS	C∅PS

A look at the data in table 1 invites a number of questions. In this presentation, I will concentrate on formulating a few questions that I find particularly interesting, and attempt to provide precise answers to them, using basic statistical concepts. First, I will deal with how the three acoustic/perceptual events I, P and S are related to each other, and show that they are not independent events. Second, I compare the relation of either of these three events to clause boundaries as opposed to phrase boundaries, showing that clause boundaries are better predictors of $I \vee P \vee S$ events than phrase boundaries are. Third, the agreement among the fours speakers in the location of $I \vee P \vee S$ events is investigated, and this agreement is found to be much greater than the agreement predicted by chance, despite the great differences between the four speakers in their individual speech styles with respect to prosodic segmentation.

2.2 Distribution of I, P and S

Question: How surprising is the occurrence of the three events I, P, and S in the same position?

A precise answer to this question can be formulated by consulting the observed distributional data in table 2, and the probabilities estimated from this data in table 3.

Table 2. IPS data.

Cases	Variables			Speakers				Sums	Means
				F1	F2	M1	M2		
i	I	P	S	136	125	102	91	454	113.50
ii	I	P	$\overline{S}$	0	7	10	14	31	7.75
iii	I	$\overline{P}$	S	5	0	2	1	8	2
iv	I	$\overline{P}$	$\overline{S}$	28	13	32	17	90	22.50
v	$\overline{I}$	P	S	33	9	12	5	59	14.75
vi	$\overline{I}$	P	$\overline{S}$	22	24	17	10	73	18.25
vii	$\overline{I}$	$\overline{P}$	S	22	7	9	2	40	10
viii	$\overline{I}$	$\overline{P}$	$\overline{S}$	632	693	694	738	2757	689.25
				878	878	878	878	3512	878

Table 3. IPS probabilities

$$p(I) = I/N = 583/3512 = 0.1660$$

$$p(P = P/N = 617/3512 = 0.1756$$

$$p(S) = S/N = 561/3512 = 0.1597$$

$$p(I \wedge P \wedge S) = 454/3512 = 0.1292$$

$$p(I \vee P \vee S) = 755/3512 = 0.2149$$

$$p(I) + p(P) + p(S) = 1761/3512 = 0.5014$$

$$p(I) \cdot p(P) \cdot p(S) = 0.0046$$

Based on the information in tables 2 and 3, and on the standard definition of independence provided below, we are entitled to the following conclusion as an answer to the first question that was posed.

Claim: I, P and S are not independent events.

Definition of independence:

Two events A and B are independent if their joint probability, $p(A \wedge B)$, equals the product of their probabilities, $p(A) \cdot p(B)$.

As can be seen from table 3, the joint probability of I, P and S, $p(I \wedge P \wedge S) = 0.1292$, does not equal but is greater than the product of the probabilities of I, P and S, $p(I) \cdot p(P) \cdot p(S) = 0.0046$, which supports the claim above.

2.3 Distribution of $I \vee P \vee S$ in relation to clause boundaries

Question: How surprising is the occurrence of either of the three events I, P, or S in the same position as a clause boundary?

C = Clause boundary (Def. clause in [9])

C DATA

The text of length 878 words has 141 clause boundary locations. $141 \cdot 4$ speakers = 564. Thus, 564 events out of a total of 3512 events are C events.

C PROBABILITY

$p(C) = 564/3512 = 0.1605$

Ph = Phrase boundary (Def. Phrase = $fn^{*}content^{+}$)

Ph DATA

The text of length 878 words has 314 phrase boundary locations. $314 \cdot 4$ speakers = 1256. Thus, 1256 events out of a total of 3512 events are Ph events.

Ph PROBABILITY

$p(Ph) = 1256/3512 = 0.3576$

Claim: C events and $I \vee P \vee S$ events are not independent.

$p(C \wedge (I \vee P \vee S)) = 0.1449 > p(C) \cdot p(I \vee P \vee S) = 0.0345$

$0.1449/0.0345 = 4.1984$

cf

$p(Ph \wedge (I \vee P \vee S)) = 0.1842 > p(Ph) \cdot p(I \vee P \vee S) = 0.0768$

$0.1842/0.0768 = 2.3965$

Claim: Clause boundaries predict $I \vee P \vee S$ events better than Phrase boundaries do, as evidenced from the data in table 4 and comparisons of precision and recall figures in table 5.

Table 4. Clause and phrase data.

Cases	Speakers				Sums	Means
	F1	F2	M1	M2		
$(I \vee P \vee S)$	246	185	184	140	755	189
$(C \wedge (I \vee P \vee S)$	135	127	130	117	509	127
$(Ph \wedge (I \vee P \vee S)$	193	165	160	129	647	162

Table 5. Comparison of precision and recall, clauses vs phrases.

	Speakers				Means
	F1	F2	M1	M2	
Clause boundary					
Precision	96%	90%	92%	83%	90.25%
Recall	55%	69%	71%	84%	69.75%
Phrase boundary					
Precision	61%	53%	51%	41%	51.50%
Recall	78%	89%	87%	42%	86.50%

2.4 Agreement among speakers

Question: How surprising is the occurrence of an $I \vee P \vee S$ event in the same position for all four speakers?

Recall that the probability of an $I \vee P \vee S$ event is $p(I \vee P \vee S) = 755/3512 = 0.2149$ which is roughly 0.2, so 1 event in 5 is an $I \vee P \vee S$ event.

If we assume that speakers produce $I \vee P \vee S$ events independently of each other, then the probability that all speakers produce an $I \vee P \vee S$ event at one and the same location is $(1/5)^4 = 0.0016 = 16/10000$.

However, the observed agreement between the 4 speakers in the location of $I \vee P \vee S$ events, when examined, was found to be much greater than that predicted by chance, as seen from the table 6:

Table 6. Speaker agreement in $I \vee P \vee S$ locations

100% agreement in 452/3512 events = 0.1287 =1287/10000

75% agreement in 140/3512 events = 0.0398 = 398/10000

75-100% agreement in 592/3512 events = 0.1685 =1685/10000

3 Clause segmentation of unrestricted text

3.1 Previous work

A recent clause parser that has attracted interest is Abney's parser, described in [1] as follows:

"1. The Parser CASS [=Cascaded Analysis of Syntactic

Structure] takes as its input the output of Church's POS (Part of Speech) program [Church 1988]. POS tags words with their part of speech, and also marks non-recursive NP's (i.e. the segments of an NP from the first word to the head noun). CASS consists of three main filters:

1 The Chunk filter builds chunks. It corrects some common errors made by POS's NP-recognition component.

2 The Clause filter recognizes clauses. It identifies the beginning and end of simplex clauses, and marks subject and predicate. If it does not find a unique subject and predicate, it attempts error correction.

3 The Parse filter assembles chunks into complete parse trees. Its primary tasks are dealing with conjunction and attachment."

Constraint based grammar, [13], is also a framework in which the recognition of simple clauses plays an important role.

In Scandinavia, the work of Paul Diderichsen [5], and particularly his definitions of main and subordinate clauses in Danish are well known. His definitions can easily be expressed as regular expressions over parts of speech, but the essentially finite state character of his definitions have not, to my knowledge, been much noted in the literature.

3.2 A new clause segmentation algorithm

The new clause segmentation algorithm that I want to propose here is described below in a way to facilitate comparison with Abney's parser.

The clause segmenter takes as its input the output of the probabilistic tagger for Swedish by Åström, using the SUC tagset, which consists of 160 morphosyntactic tags. This tagger only tags single word tokens. It does not have a component that recognizes and marks non-recursive NPs.

1 No identification of phrasal constituents, or correction of incorrectly marked non-recursive NPs precedes clause segmentation.

2 The clause segmenter identifies the beginning of each simplex clause. This supports an end-to-end segmentation of a text into clause units, where a clause unit is defined as the sequence of words from the beginning of one simplex clause to the beginning of the next simplex clause. The identification of the end of each simplex clause is postponed to a later stage of processing.

3 A clause parser takes as input the output of the clause segmenter, and clause internal parsing and identification of the ends of simple clauses is followed by the assembly of clauses into complete parse trees for orthographic sentences.

The inspiration for the clause segmentation component of this model came from reading the following passage by Salomaa on local and regular languages in [15], pp 96-97:

"6.1 Local and regular languages

Let Σ be an alphabet, let A and B be subsets of Σ, and let C be a subset of Σ^2. Then, by Theorem 2.7 [= The family of representable languages is closed under Boolean operations], the language

$$L = A\Sigma^* \cap \Sigma^* B \backslash \Sigma * C\Sigma^* \tag{6.2}$$

is regular. Languages L of the form (6.2), for some Σ, A, B, C are referred to as *local.*

The term "local" originates from the fact that if L is a language of the form (6.2), then it suffices to scan an arbitrary word w locally to find out whether or not w is in L. ...

Local languages are also referred to as "2-testable", reflecting the length of the local scans, i.e. the size of the hole. An analogous definition can be given for k-testable languages. Essentially, we are then considering a hole from which k adjacent squares can be seen.

Local languages are closely related with finite deterministic automata, ..."

The new clause segmentation algorithm is based on the hypothesis that the language of simple clauses is k-testable in the sense defined by Salomaa.

A more general and radical version of this idea is that what is k-testable makes a good processing unit in NLP.

Experiments were conducted in Umeå during the summer of 1996 with a clause recognition algorithm for unrestricted Swedish text by the author, with k=4, and with the assistance of Åström and Backman in the implementation (Åström: tokenizing, lexing, tagging [3], Backman: clause boundary insertion).

The basis for the clause segmentation rules formulated by the author was studies of bigrams and trigrams of parts of speech in the SUC corpus, combined with excerpts of tagged four-word sequences from the SUC corpus, in order to test the clause segmentation rules, before implementing them. The rules that were implemented are presented below. We are aware of the fact that there are important cases of simple clauses that are not covered by the rules that were implemented. Those are cases where a finite verb is the only indicator of the beginning of a new clause, and where there are three or more words between this finite verb and the previous finite verb. Before stating rules for these cases, we wanted to collect more information about them. When evaluating the performance of the current implementation of the clause segmentation algorithm, the failure to recognize clause boundaries in all such cases was considered as an error.

Samples of the output from the clause segmentation algorithm are presented in the appendix.

3.3 Rules

```
CLAUSE SEGMENTATION RULES

1 PUNCTUATION
1a
<h> XX => <h> <c> XX
<p> XX => <p> <c> XX


1b
```

```
DL_MAD XX => DL_MAD <c> XX,
where XX is not end tag

1c
DL_MID FIN => DL_MID <c> FIN

1d
DL_MID XX FIN => DL_MID <c> XX FIN,
where XX= PN, NN, PM or AB

2 COMPLEMENTIZERS
2a
special:
XX KN SN  => XX <c> KN SN
general:
XX SN  => XX <c> SN

2b
special:
XX KN HX  => XX <c> KN HX
XX HX HX  => XX <c> HX HX
general:
XX HX  => XX <c> HX

3 KN+FINITE VERB

special:
XX KN FIN => <c> XX KN FIN,
where XX is a closed class of finite forms of the
verbs "vara" ´be´, "g\}" ´go´, "st\}" ´stand´,
"sitta" ´sit´,
general:
$XX KN FIN => XX <c> KN FIN$

4 KN+XX+FINITE VERB, where XX=PN, NN, PM or AB

special:
$YY KN XX FIN => <c> YY KN XX FIN, if YY=XX$
general:
$YY KN XX FIN => YY <c> KN XX FIN, if YY!=XX$

5 SEQUENCES OF FINITE VERBS

5a CASE: 0 WORDS BETWEEN FINITE VERBS

$FIN FIN => FIN <c> FIN$

5b CASE: 1 WORD BETWEEN FINITE VERBS

$FIN XX FIN => FIN XX <c> FIN$

5c CASE: 2 WORDS BETWEEN FINITE VERBS
special:
$FIN YY XX FIN => FIN YY <c> XX FIN,$
where XX=PN, NN, or PM
general:
$FIN YY XX FIN => FIN YY XX <c> FIN$


ABBREVIATIONS IN CLAUSE SEGMENTATION RULES
```

```
<h> head
<p> paragraph
<<<<kk09>>>> block

</h>    end head
</p>    end paragraph
<<<</kk09>>>> end block

DL_MAD major delimiter ( . ? ! )

DL_MID  minor delimiter ( , - : )

FIN VB_PRS_AKT, VB_PRS_SFO,
VB_PRT_AKT, VB_PRT_SFO,
        VB_SUP_AKT, VB_SUP_SFO,
VB_IMP_AKT

PN PN_...._SUB, PN_...._SUB/OBJ
(subject forms of pronouns)

NN NN_...._NOM

PM PM_NOM

AB      AB, AB_POS, AB_KOM, AB_SUV
(adverbs)

KN      conjunction

SN      subjunction

HX      HA, HD_..., HP_..., HS_...
(Wh: adverbs, determiners, pronouns, possessives)
```

4 Evaluation and results

4.1 Evaluation

In order to evaluate the performance of the clause segmentation algorithm and the implementation of it, two tagged texts were clause segmented, and portions of the output of comparable length were manually scored for correcness. The two texts were taken from the SUC corpus and from the DI93 corpus respectively, and one relevant difference between the two texts is that the first text was manually tagged, whereas the second was automatically tagged. In a separate evaluation based on a test set of 19608 tokens, the fully automatic tagger for DI93 was found to be 95.45% correct, where correctness was defined as agreement with manual disambiguation (between given alternative analyses) of the same text.

Another relevant difference is that the two texts belong to very different genres. The SUC text is an excerpt from a novel (Stig Claesson, Rosine, Stockholm, Bonniers, 1991) and the DI93 text is financial newspaper text.

Descriptive data about the two texts are provided below, and those data are based only on the portions of the two texts that were scored. The length of the two texts, measured by their numbers of tokens, is not identical. The reason for this is that in order to have a roughly comparable number of clause units in the two texts that were scored, we needed to use more of text 2, because of its greater sentence length.

4.1.1 Text 1 – SUC kk09

#tokens, incl. punctuations, block and paragraph: 1251
#clause units: 219 (210 retrieved)
#orthographic sentences: 120
#finite verbs: 192
average number of tokens per sentence: 1251/120 = 10
average number of clauses per sentence: 219/120 = 1.8
average number of clauses per sentence: 1251/219 = 5.7
average number of finite verbs per sentence: 192/120 = 1.6

4.1.2 Text 2 – DI93 DI930320

#tokens, incl. punctuations, block and paragraph: 1579
#clause units: 194 (178 retrieved)
#orthographic sentences: 87
#finite verbs: 176
average number of tokens per sentence: 1579/87 = 18.15
average number of tokens per clause: 1579/194 = 8.1
average number of clauses per sentence: 194/87 = 2.2
average number of finite verbs per sentence: 176/87 2.0

4.2 Results

Table 7. Errors

	Text 1	Text 2
Error rate	9/219 = 4.1%	18/194 = 9.2%
Clauses underrecognized	9	16
Clauses overrecognized	0	2
Wrong place clause boundary	0	0

Table 8. Precision and recall

	Text 1	Text 2
Precision	210/210 = 100%	176/178 = 98.8%
Recall	210/219 = 95.8%	176/194 = 90.7%

A closer analysis of the nature of the errors revealed that in the case of text 1, all 9 errors were due to cases not covered by the current set of rules, and none were due to errors in the manual tagging. Future work will be directed to covering such cases. In the case of text 2, only 6 of the 16 errors of underrecognition were due to cases not covered by the current rules, 8 were due to tagging errors in the input to the clause segmenter, and 2 were due to a bug in the current implementation of the algorithm. Of the 2 errors of overrecognition in text 2, one was due to a tagging error, and the other was due to an error in the automatic, typographically driven paragraph segmentation that was used. This shows that improving the performance of the tagger would reduce errors in clause segmentation.

All in all, what is most striking in these results is that the number of cases that we knew would not be covered by the current set of clause rules represent such a small portion of the total number of clause units in actual empirical data, and that the majority of occurring cases are covered by these rules.

5 Conclusions

- Clause boundaries are better predictors than phrase boundaries for the occurrence of the acoustic and perceptual events I, S, P.
- The agreement between the four speakers in the location of $I \vee P \vee S$ events is much greater than chance, despite their widely different individual speech styles.
- Clause boundaries in unrestricted, tagged text can be recognized with great precision prior to any chunking of the text into constituents (non-recursive NPs, PPs, etc.).
- Automatic clause segmentation at an early stage of processing can provide the basis for an incremental parser that parses a clause at a time and assembles the result into parse trees for complete orthographic sentences.
- Relations between words and phrases within the same clause are qualitatively different from relations between words and phrases in different clauses, and clause segmentation makes it possible to exploit these differences in improving the performance of natural language processors.
- There are many open questions, even for a single language, concerning the definition of the clause units to have as targets for clause segmentation. The following are three important choices that come to mind. The approach to clause segmentation presented in this paper selects in each case the first option in the definition of the targeted clause units.

  –Only finite clauses or both finite and non-finite (infinitival and participial) clauses?
  –At most one finite verb per clause, or exactly one finite verb per clause?
  –Selective or indiscriminate use of punctuation?

- Clause definitions and clause segmentation rules, such as the ones presented in this paper, are highly language specific and new rules need to be written for each language and tested on large amounts of empirical corpus data.

REFERENCES

[1] S.P. Abney, 'Rapid incremental parsing with repair', in *Proceedings of the 6th New OED Conference*, pp. 1–9, Waterloo, Ontario, (1990). University of Waterloo.

[2] S.P. Abney, 'Parsing by chunks', in *Principle-Based Parsing*, pp. 257–278, Dordrecht, (1991). Kluwer Academic Publishers.

[3] M. Åström, 'A probabilistic tagger for Swedish using the SUC tagset', in *Proceedings of the Conference on Lexicon & Text*, Lexicographica Series Maior, Tuebingen, (to appear). Niemeyer.

[4] K. Church, 'A stochastic parts program and noun phrase parser for unrestricted text', in *Proceedings of the 2nd Conference on Applied Natural Language Processing*, pp. 136–143, Austin, Texas, (1988). ACL.

[5] P. Diderichsen, *Elementaer Dansk Grammatik*, Gyldendal, Copenhagen, 1946.

[6] E. Ejerhed, 'The processing of unbounded dependencies in Swedish', in *Readings on Unbounded Dependencies in Scandinavian Languages*, pp. 99–149, Stockholm, (1982). Almqvist & Wiksell Intl.

[7] E. Ejerhed, 'Finding clauses in unrestricted text by finitary and stochastic methods', in *Proceedings of the 2nd Conference on Applied Natural Language Processing*, pp. 219–227, Austin, Texas, (1988). ACL.

[8] E. Ejerhed, 'On corpora and lexica', in *SKY 1990 (The Linguistic Association of Finland, 1990 Yearbook)*, pp. 77–96, Helsinki, (1990).

[9] E. Ejerhed, 'A Swedish clause grammar and its implementation', in *Papers from the Seventh Scandinavian Conference of Computational Linguistics*, pp. 14–29, Reykjavik, (1990). Linguistic Institute, University of Iceland.

[10] E. Ejerhed, 'Nouveaux courants en analyse syntaxique', in *t.a.l.(Traitement Automatique des Langues) 1993*, volume 1, pp. 61–82, Paris, (1993). ATALA.

[11] E. Ejerhed and K. Church, 'Finite State Parsing', in *Papers from the Seventh Scandinavian Conference of Linguistics*, pp. 410–432, Helsinki, (1983). Department of General Linguistics.

[12] D. Huber, *Aspects of the Communicative Function of Voice in Text Intonation*, Department of Linguistics and Phonetics, University of Lund, Lund, 1988.

[13] F. Karlsson et al eds, *Constraint Grammar*, Mouton de Gruyter, Berlin and New York, 1995.

[14] L. Ramshaw and M. Marcus, 'Text chunking using transformation-based learning', in *Proceedings of the Third Workshop on Very Large Corpora*, pp. 82–94, Cambridge, Mass., (1995). SIGDAT/ACL.

[15] A. Salomaa, *Jewels of formal language theory*, Pitman Publ Ltd, London, 1981.

[16] E. Strangert, 'Perceived pauses, silent intervals and syntactic boundaries', in *PHONUM*, volume 1, pp. 35–38, Umeå, (1990). Department of Phonetics, University of Umeå.

[17] E. Strangert, 'Where do pauses occur in texts read aloud?', in *Papers from the Twelfth Scandinavian Conference of Linguistics*, pp. 403–414, Reykavik, (1990). Linguistic Institute, University of Iceland.

[18] E. Strangert, E. Ejerhed, and D. Huber, 'Clause structure and prosodic segmentation', in *Papers from the Seventh Swedish Phonetics Conference*, pp. 81–84, Uppsala, (1993). Uppsala University.

APPENDIX

An English literal translation of each word token has been added to the clause segmented output.

Text 1

```
("<<<<kk09>>>>" <1>)
("<h>"  <2>)
########## 1a  <h>  <c>  XX ##########
<c>
("<I>"  <3>              I
        (RG NOM "1"))
("<p>"  <4>)
########## 1a  <p>  <c>  XX ##########
<c>
("<Han>"        <5>      He
        (PN UTR SIN DEF SUB "han"))
("<vaknade>"    <6>      woke_up
        (VB PRT AKT "vakna"))
("<.>"  <7>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX ##########
<c>
("<Han>"        <8>      He
        (PN UTR SIN DEF SUB "han"))
("<var>"        <9>      was
        (VB PRT AKT "vara"))
("<vaken>"      <10>     awake
        (JJ POS UTR SIN IND NOM "vaken"))
########## 4  YY  <c>  KN  XX  FIN ##########
<c>
("<men>"        <11>     but
        (KN "men"))
("<han>"        <12>     he
        (PN UTR SIN DEF SUB "han"))
("<|ppnade>"    <13>     opened
        (VB PRT AKT "|ppna"))
("<inte>"       <14>     not
        (AB "inte"))
("<|gonen>"     <15>     his_eyes
        (NN NEU PLU DEF NOM "|ga"))
("<.>"  <16>
        (DL MAD "."))
########## 2a  XX  <c>  SN ##########
<c>
("<Han>"        <17>    He
        (PN UTR SIN DEF SUB "han"))
("<visste>"     <18>    knew
        (VB PRT AKT "veta"))
########## 2a  XX  <c>  SN ##########
<c>
("<att>"        <19>    that
        (SN "att"))
("<det>"        <20>    it
        (PN NEU SIN DEF SUB/OBJ "det"))
("<var>"        <21>    was
        (VB PRT AKT "vara"))
("<dager>"      <22>    day
        (NN UTR SIN IND NOM "dager"))
########## 4  YY  <c>  KN  XX  FIN ##########
<c>
("<men>"        <23>    but
        (KN "men"))
("<han>"        <24>    he
        (PN UTR SIN DEF SUB "han"))
("<ville>"      <25>    wanted
        (VB PRT AKT "vilja"))
("<beh}lla>"    <26>    to_keep
        (VB INF AKT "beh}lla"))
("<m|rkret>"    <27>    the_dark
        (NN NEU SIN DEF NOM "m|rker"))
("<.>"  <28>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX ##########
<c>
("<Han>"        <29>    He
        (PN UTR SIN DEF SUB "han"))
("<t{nkte>"     <30>    thought
        (VB PRT AKT "t{nka"))
("<.>"  <31>
        (DL MAD "."))
```

```
########## 1b  DL_MAD  <c>  XX ##########
<c>
("<Han>"        <32>    He
        (PN UTR SIN DEF SUB "han"))
("<lyssnade>"   <33>    listened
        (VB PRT AKT "lyssna"))
("<.>"  <34>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX ##########
<c>
("<Det>"        <35>    There
        (PN NEU SIN DEF SUB/OBJ "det"))
("<var>"        <36>    was
        (VB PRT AKT "vara"))
("<n}nting>"    <37>    something
        (PN NEU SIN IND SUB/OBJ "n}nting"))
########## 2b  XX  <c>  HX ##########
<c>
("<som>"        <38>    that
        (HP - - - "som"))
("<var>"        <39>    was
        (VB PRT AKT "vara"))
("<fel>"        <40>    wrong
        (PL "fel"))
########## 4  YY  <c>  KN  XX  FIN ##########
<c>
("<men>"        <41>    but
        (KN "men"))
("<{nd}>"       <42>    even_so
        (AB "{nd}"))
("<fanns>"      <43>    was
        (VB PRT SFO "finnas"))
("<det>"        <44>    there
        (PN NEU SIN DEF SUB/OBJ "det"))
("<n}got>"      <45>    something
        (PN NEU SIN IND SUB/OBJ "n}got"))
########## 2b  XX  <c>  HX ##########
<c>
("<som>"        <46>    that
        (HP - - - "som"))
("<var>"        <47>    was
        (VB PRT AKT "vara"))
("<alldeles>"   <48>    quite
        (AB "alldeles"))
("<v{lbekant>"  <49>    familiar
        (JJ POS NEU SIN IND NOM "v{lbekant"))
("<i>"  <50>            in
        (PP "i"))
("<det>"        <51>    that
        (PN NEU SIN DEF SUB/OBJ "det"))
########## 2b  XX  <c>  HX ##########
<c>
("<som>"        <52>    which
        (HP - - - "som"))
("<var>"        <53>    was
        (VB PRT AKT "vara"))
("<fel>"        <54>    wrong
        (PL "fel"))
("<.>"  <55>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX ##########
```

```
<c>
("<Jag>"        <56>    I
        (PN UTR SIN DEF SUB "jag"))
("<ligger>"     <57>    lie
        (VB PRS AKT "ligga"))
("<naken>"      <58>    naked
        (JJ POS UTR SIN IND NOM "naken"))
("<inte>"       <59>    not
        (AB "inte"))
("<i>"  <60>            in
        (PP "i"))
("<utan>"       <61>    but
        (KN "utan"))
("<ovanp}>"     <62>    on_top_of
        (PP "ovanp}"))
("<en>" <63>            a
        (DT UTR SIN IND "en"))
("<s{ng>"       <64>    bed
        (NN UTR SIN IND NOM "s{ng"))
("<,>"  <65>
        (DL MID ","))
########## 1c  DL_MID  <c> FIN ##########
<c>
("<t{nkte>"     <66>    thought
        (VB PRT AKT "t{nka"))
("<han>"        <67>    he
        (PN UTR SIN DEF SUB "han"))
("<.>"  <68>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX  ##########
<c>
("<S{ngen>"     <69>    The_bed
        (NN UTR SIN DEF NOM "s{ng"))
("<st}r>"       <70>    stands
        (VB PRS AKT "st}"))
("<i>"  <71>            in
v         (PP "i"))
("<ett>"        <72>    a
        (DT NEU SIN IND "en"))
("<rum>"        <73>    room
        (NN NEU SIN IND NOM "rum"))
                                        [Error]
("<jag>"        <74>    I
        (PN UTR SIN DEF SUB "jag"))
("<aldrig>"     <75>    never
        (AB "aldrig"))
("<vaknat>"     <76>    have_woken_up
        (VB SUP AKT "vakna"))
("<i>"  <77>            in
        (PP "i"))
("<f|rr>"       <78>    before
        (AB "f|rr"))
("<.>"  <79>
        (DL MAD "."))
("<p>"  <80>)
########## 1a  <p>  <c>  XX ##########
```

Text 2

```
("<<<<<930320-119768>>>>>" <18031>)
```

```
("<<h>>" <18032>)
########## 1a  <h>  <c>  XX ##########
<c>
("<Debatt>" <18033>  Debate
        (NN UTR SIN IND NOM "debatt"))
("<:>" <18034>
        (DL MID ":"))
########## 1d  DL_MID  <c>  XX  FIN ##########
<c>
("<Det>" <18035>  It
        (PN NEU SIN DEF SUB/OBJ "det"))
("<r{cker>" <18036>  suffices
        (VB PRS AKT "r{cka"))
("<inte>" <18037>  not
        (AB "inte"))
("<med>" <18038>  with
        (PP "med"))
("<ett>" <18039>  a
        (DT NEU SIN IND "en"))
("<Kommunrevisionsverk>" <18040> Municipal_Auditing_Agency
        (NN NEU SIN IND NOM "Kommunrevisionsverk"))
("<!>" <18041>
        (DL MAD "!"))
("<</h>>" <18042>)
("<<p>>" <18043>)
########## 1a  <p>  <c>  XX ##########
<c>
("<Sovjets>" <18044>  Soviet´s
        (PM GEN "Sovjets"))
("<fall>" <18045>  fall
        (NN NEU PLU IND NOM "fall"))
("<har>" <18046>  has
        (VB PRS AKT "ha"))
("<redan>" <18047>  already
        (AB "redan"))
("<bevisat>" <18048>  proved
        (VB SUP AKT "bevisa"))
########## 2a  XX  <c>  SN ##########
<c>
("<att>" <18049>  that
        (SN "att"))
("<myndigheter>" <18050> authorities
        (NN UTR PLU IND NOM "myndighet"))
("<inte>" <18051>  not
        (AB "inte"))
("<kan>" <18052>  can
        (VB PRS AKT "kunna"))
("<bedriva>" <18053>  manage
        (VB INF AKT "bedriva"))
("<produktion>" <18054>  production
        (NN UTR SIN IND NOM "produktion"))
("<.>" <18055>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX ##########
<c>
("<[nd}>" <18056>  Still
        (AB "{nd}"))
("<klamrar>" <18057>  cling
        (VB PRS AKT "klamra"))
("<kommunerna>" <18058>  the_municipalities
        (NN UTR PLU DEF NOM "kommun"))
```

```
("<sig>" <18059>  themselves
        (PN UTR/NEU SIN/PLU DEF OBJ "sig"))
("<fast>" <18060>  -
        (AB "fast"))
("<vid>" <18061>  to
        (PP "vid"))
("<sin>" <18062>  their
        (PS UTR SIN DEF "sin"))
("<planstyrda>" <18063>  plan_directed
        (PC PRF UTR/NEU SIN DEF NOM "planstyrd"))
("<produktion>" <18064>  production
        (NN UTR SIN IND NOM "produktion"))
("<->" <18065>
        (DL MID "-"))
########## 3  XX  <c>  KN  FIN ##########s
<c>
("<och>" <18066>  and
        (KN "och"))
("<f|rs|ker>" <18067>  try
        (VB PRS AKT "f|rs|ka"))
("<h|ja>" <18068>  to_raise
        (VB INF AKT "h|ja"))
("<kommunalskatten>" <18069> the_municipal_tax
        (NN UTR SIN DEF NOM "kommunalskatt"))
("<.>" <18070>
        (DL MAD "."))
########## 1b  DL_MAD  <c> XX  ##########
<c>
("<Det>" <18071>  The
        (DT NEU SIN DEF "den"))
("<Kommunrevisionsverk>" <18072> Municipal_Auditing_Agency
        (NN NEU SIN IND NOM "Kommunrevisionsverk"))
                                          [Error]
("<Lindbeck-kommissionen>" <18073> the_Lindbeck_Commission
        (NN UTR SIN DEF NOM "Lindbeck-kommission"))
("<f|resl}r>" <18074>  proposes
        (VB PRS AKT "f|resl}"))
########## 5a  FIN  <c>  FIN  ##########
<c>
("<{r>" <18075>         is
        (VB PRS AKT "vara"))
("<ett>" <18076>  a
        (DT NEU SIN IND "en"))
("<steg>" <18077>    step
        (NN NEU SIN IND NOM "steg"))
("<i>" <18078>           in
        (PP "i"))
("<r{tt>" <18079>    the_right
        (JJ POS UTR SIN IND NOM "r{tt"))
("<riktning>" <18080>    direction
        (NN UTR SIN IND NOM "riktning"))
("<.>" <18081>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX  ##########
<c>
("<Ett>" <18082>    A
        (DT NEU SIN IND "en"))
("<litet>" <18083>    small
        (JJ POS NEU SIN IND NOM "liten"))
("<steg>" <18084>    step
        (NN NEU SIN IND NOM "steg"))
```

```
("<!>" <18085>
        (DL MAD "!"))
("<</p>>" <18086>)
("<<p>>" <18087>)
########## 2a  XX  <c>  SN ##########
<c>
("<Lindbeck-kommissionen>" <18088> The_Lindbeck_Commission
        (NN UTR SIN DEF NOM "Lindbeck-kommission"))
("<f|resl}r>" <18089>    proposes
        (VB PRS AKT "f|resl}"))
########## 2a  XX  <c>  SN ##########
<c>
("<att>" <18090>    that
        (SN "att"))
("<det>" <18091>    there
        (PN NEU SIN DEF SUB/OBJ "det"))
("<inr{ttas>" <18092>    be_founded
        (VB PRS SFO "inr{tta"))
("<ett>" <18093>    a
        (DT NEU SIN IND "en"))
("<Kommunrevisionsverk>" <18094> Municipal_Auditing_Agency
        (NN NEU SIN IND NOM "Kommunrevisionsverk"))
("<som>" <18095>    that
        (KN "som"))
("<motsvarighet>" <18096>  corresponds
        (NN UTR SIN IND NOM "motsvarighet"))
("<till>" <18097>    to
        (PP "till"))
("<Riksrevisionsverket>" <18098> the_National_Auditing_Agency
        (NN NEU SIN DEF NOM "riksrevisionsverk"))
("<.>" <18099>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX  ##########
<c>
("<Det>" <18100>    It
        (PN NEU SIN DEF SUB/OBJ "det"))
("<{r>" <18101>            is
        (VB PRS AKT "vara"))
("<en>" <18102>            a
        (DT UTR SIN IND "en"))
("<bra>" <18103>    good
        (JJ POS UTR/NEU SIN/PLU IND/DEF NOM "bra"))
("<->" <18104>
        (DL MID "-"))
("<men>" <18105>    but
        (KN "men"))
("<otillr{cklig>" <18106>  insufficient
        (JJ POS UTR SIN IND NOM "otillr{cklig"))
("<->" <18107>
        (DL MID "-"))
("<ide>" <18108>    idea
        (NN UTR SIN IND NOM "ide"))
("<med>" <18109>    with
        (PP "med"))
("<tanke>" <18110>    consideration
        (NN UTR SIN IND NOM "tanke"))
("<p}>" <18111>            of
        (PP "p}"))
########## 2a  XX  <c>  SN ##########
<c>
("<att>" <18112>    that
```

```
        (SN "att"))
("<nationen>" <18113>    the_nation
        (NN UTR SIN DEF NOM "nation"))
("<{r>" <18114>            is
        (VB PRS AKT "vara"))
("<p}>" <18115>            on
        (PP "p}"))
("<v{g>" <18116>    its_way
        (NN UTR SIN IND NOM "v{g"))
("<att>" <18117>    to
        (IE "att"))
("<g}>" <18118>            go
        (VB INF AKT "g}"))
("<i>" <18119>            to
        (PP "i"))
("<putten>" <18120>    bankruptcy
        (NN UTR SIN DEF NOM "putt"))
("<,>" <18121>
        (DL MID ","))
("<till>" <18122>    to
        (PP "till"))
("<viss>" <18123>    a_certain
        (JJ POS UTR SIN IND NOM "viss"))
("<del>" <18124>    extent
        (NN UTR SIN IND NOM "del"))
("<beroende>" <18125>    depending
        (PC PRS UTR/NEU SIN/PLU IND/DEF NOM "beroende"))
("<p}>" <18126>            on
        (PP "p}"))
########## 2a  XX  <c>  SN ##########
<c>
("<att>" <18127>    that
        (SN "att"))
("<summor>" <18128>    sums
        (NN UTR PLU IND NOM "summa"))
("<motsvarande>" <18129>  corresponding_to
        (AB "motsvarande"))
("<25-30>" <18130>    20-30
        (RG NOM "25-30"))
("<procent>" <18131>    percent
        (NN UTR PLU IND NOM "procent"))
("<av>" <18132>            of
        (PP "av"))
("<BNP>" <18133>    BNP
        (NN AN "BNP"))
("<redan>" <18134>    already
        (AB "redan"))
("<disponeras>" <18135>    are_being_held
        (VB PRS SFO "disponera"))
("<av>" <18136>            by
        (PP "av"))
("<politikerstyrda>" <18137> politician-directed
        (PC PRF UTR/NEU PLU IND/DEF NOM "politikerstyrd"))
("<kommunala>" <18138>      municipal
        (JJ POS UTR/NEU PLU IND/DEF NOM "kommunal"))
("<myndigheter>" <18139>     authorities
        (NN UTR PLU IND NOM "myndighet"))
("<.>" <18140>
        (DL MAD "."))
########## 1b  DL_MAD  <c>  XX  ##########
```