On Lexical Aggregation and Ordering

Hercules Dalianis‡ and Eduard Hovy¥

‡ Dep. of Computer and Systems Sciences
The Royal Institute of Technology and
Stockholm University
Electrum 230, S-164 40 Kista
SWEDEN
ph (+46) 8 16 49 16
mob. ph. (+46) 70 568 13 59
fax. (+46) 8 703 90 25
email: hercules@dsv.su.se

¥USC/Information Sciences Institute
4676 Admiralty Way
Marina Del Rey , CA 90292-6695
USA
ph. (+1) 310-822-1511
fax (+1) 310-822-0751
email: hovy@isi.edu

Abstract

Aggregation is the process of removing redundant information during language generation while preserving the information to be conveyed. Various types of aggregation (syntactic, lexical, referential) have been identified in other work. This paper investigates lexical aggregation, the process by which a set of items is replaced with a single new lexeme that encompasses the same meaning. It can be divided into two major types, Bounded and Unbounded. With Bounded lexical aggregation (where the aggregator lexeme covers a closed set of concepts) the redundancy is obvious and the aggregation process must be carried out, while Unbounded lexical aggregation has to be licensed by other factors, such as the hearer's goals. Furthermore, although lexical aggregation operates over lexis, interactions between syntactic and lexical aggregation necessitate the careful ordering of their respective rules. We performed an experiment to determine the optimal ordering(s) by applying several aggregation rules, in all permutations, to the clauses of a text plan, also permuted. The paper describes the somewhat surprising results.

1. Introduction: Lexical Aggregation

Aggregation in language generation is the process of removing redundant information while preserving the essential meaning to be conveyed, in order to produce more fluent, more readable, and less boring text. Various types of aggregation (syntactic, lexical, referential) have been identified in [Hovy88, Cook84, Reinhart91, Horacek92, Dalianis&Hovy96, Dalianis95a, 95b, Wilkinson95, Xiarong&Fiedler96]. In this paper we discuss important aspects of lexical aggregation.

Lexical aggregation is the process by which a set of items is replaced with a single new lexeme that encompasses the same meaning (see Figure 1) . We call the elements that will be aggregated the *aggregands* and the element (the lexeme) which is the result of the aggregation the *aggregator*.

Lexical aggregation can be divided into two major types, *bounded* and *unbounded*. Bounded Lexical (BL) aggregation is carried out over a closed set; the aggregator is the lexical name for the set of aggregands. In contrast, Unbounded Lexical (UL) aggregation is carried out over an open set of aggregands. The following example contains both types, where *fight* and *week* are the unbounded and bounded aggregators respectively.

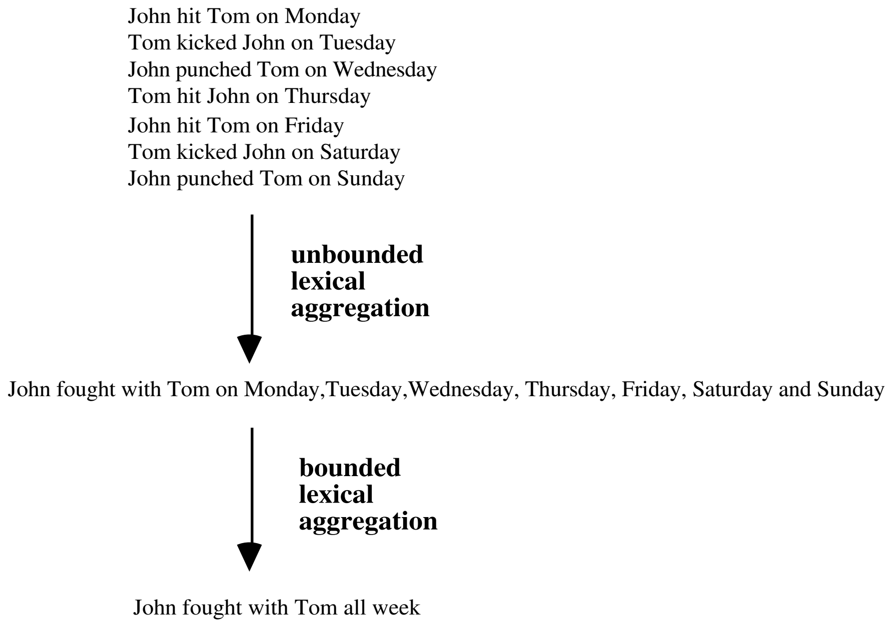

Figure 1. Examples on Bounded and Unbounded Lexical Aggregation

2. Corpora Studies

Different subsets of an information collection may give rise to many and varied opportunities for aggregation. In fact, human-authored text contains aggregations throughout, as our corpus studies shows.

We manually investigated in total 11 texts (see Appendix). The total amount of words in the first nine texts were 6.452 words and the ratio (syntactic aggregation cases)/(total words) was 1.8%. Including the two last texts, the ratio (syntactic aggregation cases)/(total sentences) was approximately 33%; i.e., one third of the sentences included syntactic aggregation.

If each aggregation saves approximately six words, this will make the text 1.8% aggregations x 8 words = 14% shorter, in some cases up to 20% shorter, than it would have been without aggregation. In addition the text becomes easier to read.

Adding to this the improved readability of aggregated text, one gets a sense of how important a phenomenon aggregation is in language generation.

Aggregated texts sometimes need cue words e.g., *each, together, separately, both*, to clarify the aggregation (see Example 1, next section). In the study we calculated the ratio cue words/sentences to be 2.0%, and the ratio (cue words)/(syntactic aggregation) to be 15% i.e., every seventh syntactic aggregation contains a cue word.

Some types of aggregation, such as Bounded lexical aggregation, refer to bounded sets, and are sometimes signalled by certain cue words, e.g., *except, all..except, exception(s) is/are, besides, excluding, exclusion, most...but, all...not, all...but*. An example of Bounded lexical aggregation with a cue word is:

Retail sales excluding auto dealers have remained practically unchanged since last June, Statistics Canada said.

Example taken from *Wall Street Journal 1992, March 24,* 60.862 words, which together with *Asiatisk Dagbok 1984,* 23.860 words contains 84.722 words and 5.807 sentences in both English and Swedish. The texts was scanned automatically for cue words and we found the ratio (Bounded Lexical aggregation cue words) / (total sentences) to be 0.5%, i.e., we have at least 0.5% BL-aggregations, because the ones with no BL-aggregation cue word are not visible or easy to find when scanning a text automatically.

3. Previous Research

Various types of aggregation have been discussed in the literature, although nobody has made a systematic study of all the different types. Much research has been done in Linguistics, in which the phenomenon of aggregation is called ellipsis, coordination, gapping, etc. For example, [Reinhart91] provides linguistic arguments to show that lexical aggregation (she calls it exception ellipsis) must take place at the Logical Form (i.e., semantic) level. Within Computational Linguistics, an early example of Unbounded lexical aggregation to support stylistic variation appears in [Hovy88], while [Horacek92] describes the integration of aggregation (which he calls grouping) with quantification under guidance of principles of conversational implicature. Syntactic aggregation is discussed in [Dalianis&Hovy96], and [Xiarong&Fiedler96] uses syntactic aggregation to present mathematical proofs, while some ideas concerning syntactic aggregation are described in [Shaw95]. In a recent paper, [Wilkinson95] argues that different kinds of aggregation are mainly side-effects of other sentence planning operations such as sentence scoping and semantic grouping. The various forms of aggregation and its nature and operation are discussed in much more detail in [Dalianis96a,96b].

4. Lexical Aggregation

4.1. Bounded Lexical Aggregation

We first discuss Bounded Lexical (BL) aggregation, in which the aggregands together constitute a known set with a fixed number of elements. As aggregator we simply choose the common name of the set. In a text plan, the aggregands can be replaced by the aggregator without loss of information. In the case that the aggregands do not completely cover the full set, one has to enumerate the residue explicitly in the text. For this task, the most common cue words in English are *except* and *all....but*.. These cue words are excellent indicators for automatically finding all the BL-aggregation cases with exception, and are valuable also for differentiating between Bounded and Unbounded lexical aggregation.

Definition: Bounded Lexical aggregation (BL-aggregation)

IF

The text plan contains a list of elements (the aggregands) that constitute the elements of a known finite (closed) set,

AND the number of aggregands is greater than half the finite set,

THEN

replace the aggregands with the name of the set (the aggregator), listing any exceptions explicitly using the cue word *except*.

Two examples:

Monday, Tuesday, Wednesday, Thursday, and Friday $\Rightarrow$ *weekdays*

Tuesday, Wednesday, Thursday, and Friday $\Rightarrow$ All weekdays *except Monday*

Bounded lexical aggregation should always be carried out since the resulting text is shorter and no information is lost during the aggregation. One exception occurs in Bounded lexical aggregation when the Hearer's goal is specifically to obtain certain information, e.g., the Hearer asks if the Speaker knows if *Mary did business this particular Sunday*, to which the Speaker should answer:

Yes, Mary did business this Sunday

and not

Mary did business all week.

We outline a simple algorithm for BL aggregation: Within the text plan,

1) first perform subject and predicate (SP) grouping or predicate and direct object (PDO) grouping (these syntactic aggregations rules are described in [Dalianis & Hovy96]),

2) for each list of elements in a single syntactic unit (such as VP or NP) (the aggregands), find every finite set all the elements together belong to,

3) for each set, if the number of aggregands is greater than half the size of the set, record the set name (the aggregator), and list all residual members of the set not included among the aggregands,

4) in the text plan, replace the aggregands with the aggregator, and generate the residual elements using the cue word *except* .

The only complex step in this algorithm is step 2, since it potentially involves considerable search or inference. Here a sophisticated knowledge representation has to be build up.

4.2. Unbounded Lexical Aggregation

Unbounded Lexical (UL) aggregation is performed over an open, unbounded set of elements, which are replaced by the aggregator. UL-aggregation does not employ cue words.

Definition: Unbounded Lexical aggregation (UL-aggregation)

IF

> the text plan contains a list of elements (the aggregands) that constitute the elements of a known enumerable (but not finite) set (named the aggregator),
>
> AND additional subconditions regarding events, states, and subtypes are met (see discussion below),

THEN

> replace the aggregands with the name of the set (the aggregator).

In order to find aggregator sets for a given list of potential aggregands it is necessary to perform some inference. This inference may be as simple as set intersection (if the aggregator sets are stored in list form) or as sophisticated as complex conceptual deduction. A common type of aggregation inference is *subconcept-superconcept access*, as in:

Sue saw a Mazda Miata. She saw a Chevrolet. She saw a Subaru. She saw a VW.

Unbounded Lexical aggregation $\Rightarrow$

Sue saw some cars

This type of aggregation however loses some information, and is hence licenced only when the information lost is not critical to the hearer's information-seeking goals. Thus UL-aggregation of this type can only be performed when an adequate model of the hearer is present. E.g., when the Hearer wants to know if the Speaker knows if *Mary did business*, then the Speaker should answer:

Yes, Mary did business all week.

and not

Mary sold tomatoes, purchased cars, vended CD's.....and had a garage sale all week.

A second common type of inference is *script-event matching*, in which the aggregands constitute all or part of a well-known sequence of events (such as a so-called *script* [Schank&Abelson77]), for example:

John hit Tom. Tom kicked John. John punched Tom. Tom slapped John.

Unbounded Lexical aggregation $\Rightarrow$

John and Tom fought

Temporal management can be achieved by sequenced time-stamps on the representation, as illustrated in the next section. Sometimes time stamps are also required for aggregating over state changes, as in:

The wall is red. Tom paints the wall. The wall is yellow.

which may not be aggregated to

The wall is red and yellow. Tom paints the wall.

It will often be the case, unfortunately, that numerous different aggregating sets will be found for a single given list of aggregands. To choose an appropriate aggregator, one needs to consider the hearer's goals and interests (This is not studied here). Taking an example from the telephone domain [Dalianis95a]:

Mary makes offhook. Mary gets dialtone. Mary dials John. John gets a ringsignal. Mary gets a ringtone. John makes offhook. A speech connection is established between Mary and John.

one may get

Mary calls John

Mary moves, and John makes offhook

(using an UL-aggregation rule that making offhook, dialing, etc., are movements), where the latter sentence does not really speak to the hearer goal *What is Mary doing?*.

Example Unbounded Aggregation Algorithm:

Consider the set of actions *John hit Tom. Tom kicked John. John punched Tom.*

which can be represented as

[f(past,hit_action,john,tom,t1)
& f(past,kick_action,tom,john,t2)
& f(past,punch_action,john,tom,t3)].

Assume the hearer-related reason to carry out UL aggregation is: *Who fought?*.

1) Use the rule
f(T,fight_rel,X,Y,T1) with
[f(T,bad_action,X,Y,T1) & f(T,bad_action,Y,X,T2)] necessary conditions
[f(T,bad_action,X,Y,Tn) & f(T,bad_action,Y,X,T(n+1))]* optional conditions

2) In the type hierarchy find that
hit_action ISA bad_action, kick_action ISA bad_action,
punch_action ISA bad_action, etc.

3) Reformulate the input as
[f(past,bad_action,john,tom,t1)
& f(past,bad_action,tom,john,t2)
& f(past,bad_action,john,tom,t3)].

4) Match the first two clauses with the necessary conditions portion of the rule and the remainder with the optional portion. The first match licenses application of the rule.

5) Return f(past,fight_rel,john,tom,t1)

5. The Problem of Ordering

5.1 Problems of Clause and Rule Ordering

The following problem is described in [Dalianis&Hovy96]: Since aggregation rules operate only over adjacent clauses, a reordering of the input clauses is essential for effective aggregation to occur. Certain combinations of input clauses give rise to less redundant text (and hence more readable text, by the basic assumption underlying aggregation) than others. But what are the optimal ordering(s)? And do other criteria apply when measuring optimality? We call issues relating to the ordering of input clauses *the clause ordering problem* of aggregation.

A second ordering problem rears its head. We call this the *rule ordering problem* of aggregation. Given various kinds of aggregation rules — lexical (bounded and unbounded), syntactic (various rules), referential, etc. — does it matter in which order the rules are applied? Depending on how the lexical aggregation rules are written, it might indeed:

a. *Mariette bought the Christmas tree*

b. *Mariette carried it inside*

c. *Mariette mounted it*

d. *Ann fetched the decorations*

e. *Ann hung the decorations on the tree*

a . b . c : Syntactic-SP (Subject and Predicate) aggregation⇒ f

f. *Mariette bought, carried inside, and mounted the Christmas tree*

d . e : Syntactic-SP (Subject and Predicate) aggregation⇒ g

g. *Ann fetched and hung the decorations on the tree*

f . g : Syntactic-PDO (Predicate and Direct Object) aggregation⇒ h

h. *Mariette and Ann bought, carried inside, and mounted, and fetched and hung the decorations on the Christmas tree respectively*

h : UL-aggregation⇒ i

i. *Mariette and Ann put up the Christmas tree*

or alternative rule ordering:

a . b . c : UL-aggregation⇒ j

j. *Mariette installed the Christmas tree*

d . e : UL-aggregation⇒ k

k. *Ann decorated the Christmas tree*

j . k : Syntactic-SP-aggregation⇒ l

l. *Mariette and Ann installed and decorated the Christmas tree respectively*

l : *no more aggregation possible*: new BL-aggregation inference required

(Note: the cue word *respectively* is introduced by aggregation to clarify the aggregated text; for more about cue words see [Dalianis97]).

In the first case, assuming the existence of a BL-aggregation inference rule that defines *put up a Christmas tree* as the sequence of events (a) to (e), this rule would produce (i). This rule would however not be able to produce (i) from (l), since (l) contains different actions altogether; here a new rule that decomposed *put up a Christmas tree* into the actions (j) *installed,* and (k) *decorated* would be required. Thus, unless the set of BL-aggregation rules were so crafted as to include all subdecompositions, different orderings of the aggregation rules will produce different results.

Furthermore, although lexical aggregation operates over lexis, interactions between syntactic and lexical aggregation necessitate the careful ordering of their respective rules. We performed an experiment to determine the optimal ordering(s) by applying several aggregation rules, in all permutations, to the clauses of a text plan. We implemented three aggregation rules (the Subject-Predicate and Predicate-Direct-Object (Syntactic) aggregation rules and the Bounded Lexical aggregation rule); also to control the order of input clauses, we created three ordering rules. An ordering rule orders the clauses in a text plan according to the weights of the ordering rule. The weights correspond to the predicate, subject, and object of the clause.

In order to determine the best order of applying aggregation rules and the ordering rules we performed the following experiment. We had a computer program cycle through all permutations of rules, and generate all possible texts for a given set of input clauses. We then analyzed these texts manually, trying to find a definition of (or failing that at least heuristics for) optimality.

Three aggregation rules and three ordering rules give 6! = 720 possible permutations (the 720 possible texts were generated automatically and came to 166 pages of A4 size). Some example permutation outputs are listed in [Dalianis96b]. To analyse the results (quite a job!), we had to make qualitative judgements. Our findings are as follows.

1. Somewhat surprisingly, text length (i.e., redundancy of words) is not the best measure of the readability of aggregated texts. Instead, a better measure is internal (structural) coherence, such as is the focus of, for example, Rhetorical Structure Theory RST, Mann&Thompson88].

2. One method to obtain good aggregation results is to perform pairwise application of one ordering and one aggregation rule at a time. A known good ordering rule should be applied on the input clauses and immediately followed by its corresponding aggregation rule, which can then be followed by another pair, etc. For example, the ordering 213 is best associated with the SP aggregation rule; the ordering 132 is best associated with the PDO aggregation rule; and the ordering 132 with the Bounded Lexical aggregation rule.

3. With respect to the rule ordering problem, the best order of aggregation rules is:
- first: Unbounded Lexical aggregation

(this is the most powerful aggregation rule);
- next, the syntactic aggregations (preferably PDO followed by SP);
- next, Bounded Lexical aggregation;
- finally, other sentence planning tasks such as pronominalization.

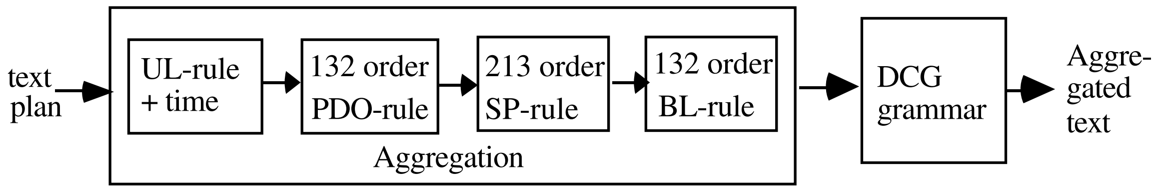

Figure 4. Overview of the input clauses ordering

6. Conclusions

While Unbounded and Bounded lexical aggregation are related to one another, UL-aggregation operates over an open set and loses some aggregated information, while BL-aggregation operates over a closed set and the aggregated information is retrievable. To select an appropriate UL aggregation one may employ a hearer model. In both types of lexical aggregation one must check that the aggregands follow each other consecutively in time. We consider lexical aggregation to take place at semantic level but it becomes visible at lexical level.

From the permutation experiment for obtaining the optimally aggregated text we conclude that one should not always select the shortest text, but the one with the best discourse organization (which we model by the best RST structure). We found certain optimal orderings of text plan clauses before aggregation, each ordering associated with aggregation rule. Regarding the order of applying the aggregation rules, we propose first to use the most powerful rule (namely UL aggregation), then the predicate and direct object (PDO) grouping rule, then the subject and predicate (SP) grouping rule, and finally BL aggregation.

A great deal more work is required on the various aspects of lexical aggregation. Thoroughly studies of text corpora are necessary as well as, more fine grained definitions of the various phenomena of lexical aggregation. The implementation of the finding will also be conditional upon the specific choices of knowledge representation system and inference support. This study is just a beginning of lot of exciting research !

7. References

[Cook84] Cook, M.E. et al., 1984. Conveying implicit content in narrative summaries. *Proceedings of 10th International Conference on Computational Linguistics*, (COLING-84), pp 5-7, Stanford University.

[Dalianis95a] Dalianis, H. 1995. Aggregation in the NL-generator of the VIsual and Natural Language Specification Tool. In *Proceedings of The Seventh International Conference of the European Chapter of the Association for Computational Linguistics* (EACL-95), Student Session. Dublin, Ireland (286-290).

[Dalianis95b] Dalianis, H. 1995. Aggregation, Formal Specification and Natural Language Generation. In *Proceedings of the NLDB'95, First International Workshop on the Applications of Natural Language to Data Bases,* (135-149), Versailles, France, June 28-29, 1995.

[Dalianis96a] Dalianis, H. 1996. Aggregation as a Subtask of Text and Sentence Planning, In the *Proceedings of Florida AI Research Symposium, FLAIRS-96*, Key West, Florida, May 20-22, pp 1-5, 1996..

[Dalianis96b] Dalianis, H. 1996. Concise Natural Language Generation from Formal Specifications, Ph.D. dissertation, (Teknologie Doktorsavhandling), Department of Computer and Systems Sciences, Royal Institute of Technology/Stockholm University, June 1996, *Report Series No. 96-008, ISSN 1101-8526, ISRN SU-KTH/DSV/R--96/8--SE.*

[Dalianis97] Dalianis, H. 1996. Natural Language Aggregation and Clarification Using Cue Words, In the *Proceedings of the 6th European Workshop on Natural Language Generation*, Gerhard Mercator University, Duisburg, Germany, March 24-26, 1997.

[Dalianis&Hovy96] Dalianis, H. and E.H. Hovy. 1996. Aggregation in Natural Language Generation. *Trends in Natural Language Generation: an Artificial Intelligence Perspective*, EWNLG'93, Fourth European Workshop, Adorni, G. & Zock, M. (Eds), Springer Verlag Lecture Notes in Artificial Intelligence no 1036, pp 88-105.

[Horacek92] Horacek, H. 1992. An Integrated View of Text Planning. *In Aspects of Automated Natural Language Generation*. Dale, R. et al. (eds.), Springer Verlag Lecture Notes in Artifical Intelligence no 587 (193-227).

[Hovy88] Hovy, E.H. 1988. *Generating Natural Language under Pragmatic Constraints*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

[Mann&Thompson88] Mann, W.C. and S.A. Thompson. 1988. Rhetorical Structure Theory: Towards a Functional Theory of Text Organization. *TEXT* 8(3) (243-281).

[Reinhart91] Reinhart, T. 1991. Elliptic Conjunctions-Non-Quantificational LF. In *The Chomskyan Turn*. A.Kasher (ed.), Basil Blackwell (360-384).

[Schank&Abelson77] Schank, R.C. and R.P. Abelson. 1977. *Scripts, Plans, Goals, and Undersanding.* Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

[Wilkinson95] Wilkinson, J. 1995. Aggregation in Natural Language Generation: Another Look. Unpublished M.Sc. thesis, Computer Science Department, University of Waterloo, Canada.

[Xiarong&Fiedler96] Xiarong, H. & Fiedler, A. (1996). Paraphrasing and Aggregating Argumentative texts Using Text Structure, In the *Proceeedings of the 8th International Workshop on Natural Language Generation*, INLG'96, pp 21-30, Herstmonceux, Sussex, UK, June 13-15, 1996.

Appendix: Some Empirical Statistics on Aggregation

In order to determine the occurrence and distribution of aggregation in naturally occurring texts in various genres we studied excerpts from the following: a road map, travel book, newspapers, a scientific book, and a diary. Some results are mentioned in Section 1 above.

1) *California Road Atlas (Thomas Bros. Maps)* pp.213-220, in total 1000 words.
2) *Fielding's Mexico 1985* by L. & L. Foster, Introduction pp. 1-2, in total 400 words.
3) *Macintosh Programmer's Workbench*, by Joel West, pp. 335-336, in total 636 words.
4) *Los Angeles Times Calendar,* Monday Oct 31 1994, 907 words.
5) *MacWeek, Vol 8 No 41, 1994,* p. 3, in total 518 words.
6) *Time Vol 145, No. 8, February 1995,* pp 36-38, in total 672 words.
7) *The Art of Probability, for Scientist and Engineers,* by R.W. Hamming: Addison Wesley Publishing Company 1991, pp 1-2, in total 870 words.
8) *Wall Street Journal 1992, March 24*, sample of 716 words and 52 sentences.
9) *Asiatisk Dagbok 1984, (Asian Diary 1984)* (in Swedish) by Hercules Dalianis, sample of 733 words and 45 sentences.
10) *Wall Street Journal 1992, Ma*rch 24. Text contained 60.862 words and 4.548 sentences. In this text the ratio Words/Sentences = 13.4; assuming aggregation saves on average 6 words then in the sentences where it did occur, aggregation saved about half the sentence on average. In this text, 24 cue words for Bounded Lexical aggregation were found: except (4), exceptions are (2), exceptions is (1), besides (2), excluding (2), exclusion (1), most ... but (4), all.. not (2), all... but (2). The following Bounded Lexical cue words were not found: apart of, aside from, exclusive of, exception of.
11) *Asiatisk Dagbok 1984, (Asian Diary 1984)* (in Swedish) by Hercules Dalianis. Text contained 23.860 words and 1.259 sentences. The ratio Words/Sentences = 20.0. The following 5 Bounded Lexical aggregation cue words were found: förutom(2) (besides), alla .... utom (3) (all...but). Bounded Lexical aggregation cue words not found: flesta ....utom, (most ....but).

In summary, the first nine texts contained 6.452 words, with the ratio (syntactic aggregation cases) / (total words) of 1.8% and the ratio (syntactic aggregation cases) / (total sentences) of approximately 33%. Assuming each aggregation saves on average six words, aggregation shortens a text by approx. 1.8% aggregations x 6 words = 11% shorter. In our study, eight of ten syntactic aggregations are subject and predicate grouping, while the remainder the rest are predicate direct object grouping.

Text	Words	Sent.	Synt.Agg.	S/SP	PDO	cue word.	BL cue
1	1000	65	29	26	3	2	
2	400		11	10	1	3	
3	636		9	9	0	0	
4	907		9	7	2	2	
5	518		5	4	0	0	
6	672		15	9	6	3	
7	870		9	5	4	4	
8[1]	716	52	10	8	2	2	0
9[1]	733	45	19	16	3	1	0
10	60862	4548				86	24
11	23860	1259				32	5
Total	91174	5969	116	94	21	135	29

Table 1. Table describing the results from the empirical study on texts.

Synt. Agg. = Syntactic Aggregation

S/SP = Subject/Subject Predicate Grouping

PDO = Predicate and Direct Object Grouping

BL = Bounded Lexical Aggregation Cue word

[1] Texts 8 and 9 are extracts from texts 10 and 11. Texts 10 and 11 have been computer scanned, while texts 8 and 9 have been manually read and elaborated on, to investigate if they seem to follow the pattern of the other texts 1..7. , which they did.