A Syntactic Approach to Automatic Book Indexing

Gerard Salton*

TR 89-979
March 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This research was supported in part by NSF grant IRI 87-02735 and in part by a grant from OCLC Incorporated.

A Syntactic Approach to Automatic Book Indexing

Gerard Salton*

Abstract:

Automatic publishing systems are now widely used to produce books and documents of many types. As a result, large masses of text become available in machine readable form ready for automatic processing. This study describes automatic methods to generate back-of-the-book indexes, based on a syntactic analysis of the text of book chapters followed by a phrase generation system that identifies meaningful book indexing entries.

*Department of Computer Science, Cornell University, Ithaca, NY 14853.
This study was supported in part by the National Science Foundation under grant IRI 87-02735, and in part by a grant from OCLC Inc.

1. Automatic Text Analysis

In recent years significant advances have occurred in the area of natural-language text processing. Large masses of machine-readable text are now available, obtained largely through word processing and other automatic text publishing systems. Furthermore, the technology used to process natural-language text has evolved to the point where inexpensive, high-density storage devices, such as compact optical disks, are available to store texts and graphic images, and high-resolution graphic displays are usable permitting a human operator to interact with the text processing system. Finally, certain practical language processing tools, including automatic syntactic analysis systems, are now capable of handling arbitrary, unrestricted text items, and of performing many useful text transformations. Because of such advances in text processing software and hardware technologies, it is now possible to design sophisticated, and realistic text transformation systems to derive important by-products such as text concordances, text summaries, tables of content, and text indexes.

Consider, as an example, the document indexing task designed to assign content identifiers, or index terms, to individual texts, to be used collectively to identify text content. In the past, the best automatic indexing policy led to the assignment to each document of large numbers of properly weighted *single terms* for content representation purposes. [1-3] Although single terms are often ambiguous, automatically assigned term weights can be used to distinguish important from less important terms. Furthermore, the greater number of single terms automatically assigned to the items can provide many different access points to the stored items, leading to better retrieval results than appears possible in many manual indexing environments.

Over the years, many attempts have been made to enhance the single term indexing systems. Thus *thesauruses* were introduced in attempts to recognize synonyms [4,5] and *term phrases* derived from co-occurring words in document texts were used for content identification. [6,7] Unfortunately, these enhancements could be proven effective only in special circumstances--for example, when a hand-tailored thesaurus would be constructed for a particular document collection. An exhaustive study of the use of statistically derived term phrases shows that substantial improvements in search effectiveness are obtainable for some text collections, but little effect is noticed for other collections. [8,9] Such inconsistent results explain why the simple single-term indexing systems are normally preferred in automatic environments over more refined text analysis schemes with more questionable effectiveness.

When the user is part of the processing chain, or when an indexing product is destined for use by human beings, the simple text representation by sets of single terms cannot be maintained. In such a case, indexing units must be used that are more directly representative of text meaning, providing a better guide to the document content. In practice, this means that the index terms must be surrounded by context, as they are in a book index. The remainder of this note deals with the problem of automatically constructing book indexes consisting of phrase entries. An index generation method is proposed, and the effectiveness of a phrase index usable by human beings for text access is investigated.

2. Term Phrase Environment

In conventional text processing systems, phrases are widely used for the content representation of text items. Thus, phrases are used as subject indicators in conventional library card catalogs. Phrases also serve as principal entries in concordances and in context indexes, and they form the bulk of the entries in book indexes. In automatic text processing environments, phrases are usable for a multiplicity of purposes:

- In the formulation, or reformulation of search queries, where a display of terms or phrases from previously retrieved documents may aid the user in generating useful search queries. [10]
- In manual, or automatic text and book indexing, where phrase lists are used by authors, and indexers to construct appropriate text indexes. [11]
- In information search and retrieval where phrase entries can be added to other indexing products to help in retrieving useful information. [12,13]

Various methodologies are available for constructing phrase entries extracted from the texts of natural-language documents:

- The statistical approach consists in using pairs or triples of text words that occur jointly sufficiently often in certain contexts within a document or document collection. A phrase might, for example, be formed from two single words exhibiting specified frequency characteristics, and occurring sufficiently often in common in the document sentences of a collection. [8,9]

- Alternatively, phrases might be formed by considering particular text excerpts-- for example, chapter titles and headings--and using specified word patterns extracted from such titles or headings. A title such as *"Spreading of virus infection* among *wild bird* during *influenza epidemic"* could produce phrases such as "spreading of virus epidemic", "wild birds", and "influenza epidemic". [14]
- Phrases can also be generated by assigning to each text word one or more *syntactic tags* derived from stored dictionaries. A phrase is then defined as a sequence of words carrying specified syntactic tags. For example, the sequence "little European bird" might be characterized by the tag assignments adjective-adverb-noun (little); noun-adjective (European); noun-intransitive verb (bird). This word sequence might then be accepted as an indexing phrase because one choice of tag sequences consists of the triple adjective-adjective-noun. [15,16]
- Finally, the phrase formation process may be dependent on a prior full *syntactic analysis* of the document texts, producing one or more parse trees for each input sentence. A phrase is then definable as a syntactically dependent structure of a certain type occurring in a parse tree. [17-20]

It may be noted that when linguistic considerations are used for phrase formation, such as those dependent on the assignment of syntactic tags to the text words, or on full syntactic text analyses, substantial complications may arise in the phrase formation process because of ambiguities inherent in the text input. Table 1 shows the results of a syntactic tag assignment for the sample input "a fat little European bird". Since each word of the example carries a multiplicity of syntactic tags, a total of 108 different tag sequences are possible for the five-word input. Fortunately, disambiguation methods based on probabilistic considerations are available that find the correct tag assignment (article-adjective-adjective-adjective-noun for the example under consideration) with a high degree of accuracy. [21,22]

When a full syntactic analysis is used as a basis for a phrase generation method, syntactic parse trees reflecting the grammatical structure of each input sentence can be used for phrase formation. Fig. 1 shows syntactic output obtained for the sample sentence "Numeric coding extends this idea by assigning fixed-length codes to larger subsets of adjacent characters". The syntactic analysis system used in this study is the PLNLP English grammar (PEG) developed at the IBM Research Laboratory in Yorktown Heights*. [23,24]

*The writer is grateful to the IBM Corporation and to Dr. George Heidorn for making available the PEG parsing programs for use at Cornell.

The parse trees of Fig. 1 are shown rotated by ninety degrees from the usual format. The root of the tree appears at the left end of the display, and the leaves appear on the right. The output of Fig. 1 shows that the sentence is a declarative construction (DECL), which is broken down into an initial noun phrase (NP), a verb (VERB), and a terminal noun phrase (NP). The terminal noun phrase is itself broken down further to the right into a determiner (DET), a noun (NOUN) and a prepositional phrase (PP). The latter consists of a preposition (PREP), a verb (VERB), and a noun phrase (NP), and so on. Each subtree contains a phrase head identified by an asterisk (*), and of additional dependent structures that may be either premodifiers (preceding the head) or postmodifiers (following the head). The dependent structures are located at lower tree levels, that is, to the right of the head in the output of Fig. 1. Thus the head of the phrase "larger subsets of adjacent characters" is the noun "subsets" with premodifier "larger" and postmodifier "of adjacent characters", respectively.

The phrase identification system described in the next section uses syntactic parse trees as input to reduce each sentence to a set of phrases indicative of sentence content. This task is unfortunately substantially complicated by the fact that several different, apparently correct, parse trees are produced for many input sentences. The two parse trees of Fig. 1 differ by the assignment of verb and adjective tags, respectively, to the input word "assigning". (Most present participles in English carry this type of ambiguity and a correct resolution proves difficult in many cases.) If the second analysis is falsely assumed to be correct, the phrase production system will necessarily recognize the complete adjective-noun-noun sequence "assigning fixed-length codes" as a significant phrase, instead of the proper sequence "fixed-length codes". In these circumstances, the generation of representative term phrases remains problematical, even when accepted language analysis tools are incorporated in the processing chain.

3. Phrase Generation System

It was noted earlier that phrase generation systems that are based on statistical co-occurrences of text words in the original texts may produce uneven test results in information retrieval. Many good phrases are generated, but also many poor ones. For this reason the phrase identification system described in this note is based on a syntactic analysis of document texts, followed by phrase generation and phrase normalization methods appropriate for the application under consideration. The following main components may be distinguished in a phrase generation system using parse trees as principal input: [25]

- A system recognizing tree excerpts consisting of adjective and/or noun premodifiers, followed by head nouns, followed (optionally) by one or more postmodifiers, all located in the same subtree. In the output of Fig. 1(a), this component leads to the identification of entries such as "numeric coding", "idea", and "fixed-length codes".
- A component that identifies prepositional phrases using the preposition "of". Such prepositional phrases are inverted when the number of phrase components is limited to at most three. For example, the phrase "retrieval of information" is transformed into "information retrieval". In the output of Fig. 1 the phrase "larger subsets of adjacent characters" is not inverted because four adjective and/or noun components are present. In that case two separate phrases ("larger subsets" and "adjacent characters") are obtained.
- A system that distributes pre- and postmodifiers across certain conjunctive constructions identified by commas or the conjunction "and". For example, the input "automatic indexing, enciphering, and decryption", produces the output phrases "automatic indexing", "automatic enciphering", and "automatic decryption".
- A phrase elimination system that rejects constructed phrases when the head noun appears on a special list of excluded phrase heads ("ability", "activity", "amount", "application", "approach", "area", and so on). A list of excluded phrase modifiers is also used to reject unwanted modifiers such as "above", "additional", "apparent", "applicable", "appropriate", "atypical", and so on. When a modifier is rejected, the head noun and other modifiers contained in the phrase are maintained intact.
- A phrase recognition system that identifies phrases containing capitalized components (e.g. "German cipher clerks", "British cryptanalysts"), as well as phrases extracted from titles and section headings. Such phrases may receive special treatment during phrase normalization, as explained in the sequel.

The foregoing rule system produces raw phrase lists as shown in the example of Fig. 2.* The raw output contains one entry for each detected phrase, together with identifying information specifying the text location (paragraph number and sentence number) where the phrase was located. In the output of Fig. 2 the location information

*The phrase output used for illustration in this study is produced by a full text analysis of chapters 5 and 6 of reference [26].

is contained in the 5-digit code terminating each entry. In addition, information is provided about the phrase generating rule used to obtain a phrase (the R code), the type of article preceding the phrase (the A code), and the type of parse tree from which the phrase was extracted (the P code). As an example, the R:8 codes attached to the "ciphertext" entries in Fig. 2 identify the phrase as one lacking both premodifiers and postmodifiers; A:2 identifies the article preceding a phrase as "the"; and P-1 indicates that the normal grammar rules were used to produce the parse trees for the sentence containing a given phrase, as opposed to special grammar rules that maybe invoked in certain cases.

The raw phrase output can be used as basic input for a variety of text processing applications, including

- Automatic document indexing, where identifying phrases are assigned to individual documents; in that case, identical, or similar, phrases may be recognized and phrase weights may be assigned in decreasing order of presumed usefulness of the phrases for indexing purposes.
- Phrase display systems in interactive systems where users choose particular entries from a phrase display for query formulation or computer-aided indexing operations.
- Automatic book indexing operations where back-of-the-book indexes may be automatically constructed.

In all these applications, the raw phrase output must first be appropriately transformed and normalized, leading to the choice of certain phrases and the rejection of others. A typical phrase normalization process useful in book indexing is described in the next section.

4. Phrase Normalization for Book Indexing

It is well known that the manual construction of book indexes is onerous and time-consuming. Indeed, the construction of useful book indexes represents a major stumbling block in a book production system. Since book texts are normally provided in machine readable form, the hope has been expressed that useful automatic book indexing methods could be derived. The book indexing problem has thus been considered repeatedly in the past. [27-31] The methods described in this study differ from earlier attempts by basing the book index on a state-of-the art syntactic analysis of book texts, and combining the syntactic methodologies with phrase normalization methods designed to generate final indexing entries.

The following normalization methods are used to reduce the raw phrase output:

- The phrase lists are condensed by reducing all identical entries to a single entry, and computing frequency indicators giving the occurrence frequency of each entry. Hyphenated entries such as "high-frequency" are considered identical to the corresponding nonhyphenated entry ("high frequency"); entries differing only by a final 's' ending ("cats" and "cat") are considered identical; entries appearing in all capital letters ("ENIGMA") are considered identical to similar entries appearing with an initial capital letter only ("Enigma").

- Single word entries that are not contained in titles are eliminated together with all nontitle phrase entries exhibiting total occurrence frequencies equal to 1. Title phrases are preserved regardless of frequency.

- Certain entries among mutually overlapping sets of phrases are eliminated (a phrase "wxy" consisting of adjacent words w, x, and y contains the shorter phrase "xy"; similarly, phrases "xyz" and "xy" are mutually overlapping):

  i) For overlapping nontitle phrases exhibiting equal occurrence frequencies, the longer entry is eliminated and the shorter one used for indexing purposes; thus, given phrases "code assignment" and "Huffman code assignment" with occurrence frequencies equal to 2, the longer phrase ("Huffman code assignment") is dropped.

  ii) For overlapping nontitle phrases with unequal frequencies, the entry with the highest frequency is preserved and the other is eliminated; thus, for entries "code length" (frequency 2) and "single code length" (frequency 3), the "code length" entry is eliminated.

  iii) For overlapping phrase sets including title entries, any title entry exhibiting a frequency at least equal to 2 is preserved. For title entries of frequency 1, rules i) and ii) apply.

  iv) Phrases consisting of one hyphenated word pair are entered in the index without hyphen; thus "variable-length" is entered as "variable length".

The foregoing phrase normalization rules were used to construct automatic indexes for chapters 5 and 6 of reference [26], dealing with text compression and text encryption, respectively. The original texts of approximately 5,000 and 7,000 words for the two chapters were initially reduced to about 2,200 and 2,900 entries in the corresponding raw phrase lists. These lists were further condensed by elimination of identical and partly overlapping entries to a total of 180 and 220 entries in the final indexes. Excerpts from the respective automatic indexes are shown in Table 2. The phrase frequency is included in the table together with references to all paragraphs of the text in which the corresponding phrases occur. For comparison purposes, Table 3 contains excerpts of manually constructed indexes covering the same material.

A comparison of the entries of Tables 2 and 3 shows that some entries in the automatic index are relatively nonspecific (for example, "string character", or "text character"), and a few are marginal (for example, "appending code increments"). However, most entries in the automatic index consist of proper English phrases, and they are for the most part as specific as those appearing in manually built indexes.

A more detailed comparison of normal and automatic indexes appears in Table 4. As the table shows, the manual indexes contain many fewer entries than the automatic indexes--in general fewer than half. At the same time, a substantial number of manual entries are also generated automatically (between 50 and 75 percent). Moreover, among the automatic entries that are not also included in the manual index, many are useful indexing phrases that would be missing if a manual process were used. The phrase lists of Table 5 provide the following illustrations: whereas some specific phrases such as "ASCII code" and "Zipf law" are missing from the automatic index, presumably because the required formal frequency criteria are not met in these cases, other good entries such as "decoding dictionary" and "semantic redundancy" are not available in the manual index. This suggests that any lack in term specificity in an automatic index may be made up by the presence of useful extra entries that are not available in the manual case.

5. Retrieval Test

Ultimately, an evaluation of index performance must be based not on an examination of indexing entries, but on the use of the index in a retrieval environment. A number of search requests were therefore used, covering the areas of text compression and text encryption, respectively. The following typical queries were included in the text compression sample:

- What are the reasons for considering text compression? What are the advantages and disadvantages of text compression?
- Can one rationalize Zipf's principle of least effort?
- How are word-frequency statistics used in text compression?

In the text encryption case, the query set included:

- What are the main text encryption methods?
- What is the difference between computational and unconditional security?
- How are digital signatures used in data transmission?

Table 6 is a summary of retrieval results reflecting the performance of 33 queries in text compression and 30 queries covering text encryption. The table shows that the majority of the queries can be satisfactorily answered by both manual and automatically built indexes. For text encryption, no difference in performance could be found for 24 out of 30 queries used. Among the queries that were answerable only by using one or the other of the indexes, a somewhat larger number could be handled in the manual case than the automatic one. However, it is difficult to tell whether these differences are significant, because the relevance assessments and test results used here depended entirely on the writer's personal judgment, rather than on the opinions of real users in operational test environments.

In summary, it appears that the available language processing technology is sufficiently advanced to make possible an automatic construction of text indexes containing useful phrase entries, based on an analysis of the texts of corresponding documents. The automatically produced indexes will normally be larger in scope than comparable manual indexes. In general, the majority of the indexing phrases used in conventional manual indexes will also be provided in comparable automatic indexes. A test using typical queries in two subject areas shows that a majority of the available queries will be handled correctly in either index.

The procedures used for the construction of automatic indexes in this study were based on simple criteria of usefulness, including notably the occurrence frequency of the phrases in the document texts, and the location of the phrases in the texts. More refined phrase assignment methods might be used: in particular, the type of syntactic parse used to generate a phrase might help in distinguishing better phrases from more marginal ones; similarly, the type of phrase use--whether used as subject or object in a sentence--and the type of preposition appearing in preposition phrases could be considered. Finally, the introduction of phrase weighting can help in distinguishing potentially important, more specific phrases from more general indexing entries that may prove less useful. Based on the available evidence, it appears that useful book and text indexes can be constructed automatically that perform adequately in manual as well as automatic text-search and retrieval environments.

References

[1] G. Salton, A Blueprint for Automatic Indexing, *ACM SIGIR Forum*, 16:2, Fall 1981, 22-38.

[2] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw Hill Book Company, New York, 1983.

[3] G. Salton, C.S. Yang, and A. Wong, A Vector Space Model for Automatic Indexing, *Communications of the ACM*, 18:11, November 1975, 613-620.

[4] K. Sparck Jones, *Automatic Keyword Classification for Information Retrieval*, Butterworths, London, 1971.

[5] G. Salton, Experiments in Automatic Thesaurus Construction for Information Retrieval, *Information Processing 71*, North Holland Publishing Company, Amsterdam, 1972, 115-123.

[6] G. Salton, On the Role of Words and Phrases in Automatic Text Analysis, *Computers and the Humanities*, 10:2, March-April 1976, 69-87.

[7] M.E. Lesk, Word-word Associations in Document Retrieval Systems, *American Documentation*, 20:1, January 1969, 27-38.

[8] J. Fagan, Experiments in Automatic Phrase Indexing for Document Retrieval: An Examination of Syntactic and Nonsyntactic Methods, *Proceedings of the 10th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, C.T. Yu and C.J. van Rijsbergen, editors, Association for Computing Machinery, New York, 1987, 91-101.

[9] J. Fagan, The Effectiveness of a Nonsyntactic Approach to Automatic Phrase Indexing for Document Retrieval, *Journal of the ASIS*, 40:2, March 1989, 115-132.

[10] D. Harman, Towards Interactive Query Expansion, *Proceedings of the 11th International Conference on Research and Development in Information Retrieval*, Y. Chiaramella, editor, Grenoble, France, June 1988, 321-331.

[11] R.K. Jurgen, Desktop Publishing: What it Can Do and Cannot Do, *IEEE Spectrum*, 24:3, March 1987, 50-52.

[12] P. Atherton, Improved Subject Access to Books, in *On-Line Library Catalogs*, First International On-Line Information Meeting, London, December 1977, Learned Information, New York, 1977, 131-138.

[13] B. Settel and P.A. Cochrane, Augmenting Subject Descriptions for Books in Online Catalogs, *Database*, 5:4, December 1982, 29-37.

[14] G. Vladutz, Natural Language Text Segmentation Techniques Applied to the Automatic Compilation of Printed Subject Indexes and for On-Line Database Access, *Proceedings of the Conference in Applied Natural Language Processing*, Association for Computational Linguistics, Santa Monica, CA, February 1983, 136-142.

[15] A.H. Klingbiel, Machine Aided Indexing of Technical Literature, *Information Storage and Retrieval*, 9:2, 79-84, and 9:9, 477-494, 1973.

[16] M. Dillon and A.S. Gray, Fully Automatic Syntactically Based Indexing System, *Journal of the ASIS*, 34:2, March 1983, 99-108.

[17] F.J. Damerau, Automatic Parsing for Content Analysis, *Communications of the ACM*, 13:6, June 1970, 356-360.

[18] D.J. Hillman and A.J. Kasarda, The Leader Retrieval Systems, *AFIPS Proceedings*, AFIPS Press, Montvale, NJ, 34, 1969, 447-455.

[19] A.F. Smeaton and C.J. van Rijsbergen, Experiments on Incorporating Syntactic Processing of User Queries into a Document Retrieval Strategy, *Proceedings of the 11th International Conference on Research and Development in Information Retrieval*, Y. Chiaramella, editor, Grenoble, France, June 1988, 31-52.

[20] Y. Chiaramella, B. Defude, M.F. Bruandet and D. Kerkouba, IOTA: A Full Text Information Retrieval System, *Proceedings of the 9th International Conference on Research and Development in Information Retrieval*, F. Rabitti, editor, Pisa, Italy, September 1986, 207-213.

[21] S.J. DeRose, Grammatical Category Disambiguation by Statistical Optimization, *Computational Linguistics*, 14:1, Winter '88, 31-39.

[22] G. Leech, R. Garside and E. Atwell, The Automatic Grammatical Tagging of the LOB Corpus, *ICAME News*, 7, 1983, 13-33.

[23] G.E. Heidorn, K. Jensen, L.A. Miller, F.J. Byrd and M.S. Chodorow, The EPISTLE Text Critiquing System, *IBM Systems Journal*, 21:3, 1982, 305-326.

[24] K. Jensen, G.E. Heidorn, L.A. Miller and Y. Ravin, Parse Fitting and Prose-Fitting: Getting Hold of Ill Formedness, *American Journal of Computational Linguistics*, 9:3-4, July-December 1983, 147-160.

[25] J. Kaye, Back-of-the-Book Indexing Project, Term Report CS 490, Department of Computer Science, Cornell University, Ithaca, NY, Spring 1988.

[26] G. Salton, *Automatic Text Processing*, Addison-Wesley Publishing Company, Reading, MA, 1989.

[27] S. Artandi, Book Indexing by Computer, Doctoral Dissertation, Rutgers University, New Brunswick, NJ, 1963.

[28] M. Dillon, Thesaurus-Based Automatic Book Indexing, *Information Processing and Management*, 18:4, 1982, 167-178.

[29] L.L. Earl, Experiments in Automatic Extracting and Indexing, *Information Storage and Retrieval*, 6, 1970, 313-334.

[30] H. Borko, Experiments in Book Indexing by Computer, *Information Storage and Retrieval*, 6:1, May 1970, 5-16.

[31] C. Bell and K.P. Jones, Back-of-the-Book Indexing: A Case for the Application of Artificial Intelligence, in The Analysis of Meaning, Informatics 5, M. MacCafferty and K. Gray, editors, *Aslib*, London, 1979, 155-161.

Appendix

A. Automatic Index for Text Compression (with frequency indicators)

A	adjacent character	2
	alphabetic character	7
	appending code increments	2
	code assignment	4
	average length	2
B	binary digits	6
	bit level	3
	eight-bit bytes	6
	byte length	3
C	character combination	3
	character entropy	3
	adjacent character	2
	alphabetic character	7
	combining character	3
	dependent character	2
	frequent character	4
	frequency characters	5
	individual characters	2
	master characters	7
	single characters	7
	string characters	2
	terminal characters	2
	text characters	2
	frequency characteristics	3
	occurrence characteristics	5
	statistical language characteristics	1
	code assignment	4
	code length	18
	appending code increments	2
	corresponding code	6
	fixed-length code	11
	Huffman code	9
	unique code	2
	variable-length code	8
	character combination	3
	code combination	7
	letter combination	2
	combing character	3
	communications theory	4
	data compaction	2
	nonzero components	2
	compressed files	2
	compression ratio	14
	compression system	5
	data compression	4
	text compression	5
	frequency consideration	1
	information content	10
	corresponding code	6
	frequency count	2
D	data compaction	2
	data compression	4
	decoding dictionary	2
	dependent character	2
	dictionary file	2
	binary digits	6
	nonzero digits	4
	distinct words	10
E	eight-bit byte	6
	linguistic element	2
	encoded text	2
	English texts	5
	English words	5
	entropy measurements	1
	character entropy	3
F	compressed files	2
	dictionary file	2
	text files	2
	fixed-length codes	11
	fixed-length codes	11
	word fragments	4
	frequency characteristics	3
	frequency characters	5
	frequency considerations	1
	frequency counts	2
	frequency information	2
	frequency order	4
	occurrence frequency	5
	target frequency	3
	word frequency	2
	frequent characters	4
	frequent words	8
H	Huffman code	9
	variable-length Huffman	2

Automatic Index for Text Compression

B. Automatic Index for "Text Encryption" (Chapter 6)

A	frequency analysis	4
	THE CAT ATE THE RAT	2
B	ciphers based	1
	bit key length	2
	input bits	2
	message block	3
	character blocks	2
	data blocks	3
	code book	5
	P- boxes	2
	British cryptanalysts	2
C	THE CAT ATE THE RAT	2
	communication channel	2
	character blocks	2
	character combination	6
	character position	10
	character substitution	2
	ciphertext character	3
	corresponding character	2
	enciphered character	5
	key character	5
	output character	2
	plaintext character	7
	occurrence characteristics	3
	cipher clerks	6
	product cipher	4
	RSA cipher	2
	ciphering methods	2
	ciphers based	1
	polyalphabetic ciphers	3
	running-key ciphers	3
	substitution ciphers	4
	ciphertext character	3
	cipher clerks	6
	code books	5
	code WQSEUP	2
	combination lock	2
	character combination	6
	communication channel	2
	computational security	3
	computationally hard problems	1
	cryptographic concepts	1
	confusion factor	2
	conventional cryptography	2
	corresponding character	2
	British cryptanalysis	2
	cryptanalytic effort	2
	cryptographic concepts	1
	cryptographic systems	5
	cryptographic transformation	4
	conventional cryptography	2
	public key cryptosystem	5

D	data blocks	3
	Data Encryption System	2
	deciphering operation	2
	decoding machine	2
	decrypting effort	2
	decrypting key	2
	DES system	5
	key distribution	2
	reversing drum	2
E	cryptanalytic effort	2
	decrypting effort	2
	vector element	3
	enciphered character	5
	enciphered message	4
	enciphered text	6
	enciphering keys	3
	enciphering operation	2
	Data Encryption System	2
	text encryption	2
	German Enigma	2
F	confusion factor	2
	simplified form	2
	frequency analysis	4
	one-way functions	2
G	random number generator	3
	German Enigma	2
H	Hagelin machine	4
	computationally hard problems	1
	home position	2
I	indicator setting	2
	key information	14
	secret information	2
	secret trapdoor information	4
	input bits	2
	input character	2
	input signal	2
	input stream	2
	input wires	2
	positive integer	5
	inverse transformation	2
	multiplicative inverse	2
K	bit key length	2
	public key cryptosystem	5
	key character	5
	key distribution	2
	key information	14
	key sequence	8
	key stream	3

Input Sentence: Numeric coding extends this idea by assigning fixed-length codes to larger subsets of adjacent characters.

Syntactic Parse Tree:

DECL	NP	AJP	ADJ*	"numeric"				
		NOUN*	"coding"					
	VERB*	"extends"						
	NP	DET	ADJ*	"this"				
		NOUN*	"idea"					
		PP	PREP	"by"				
			VERB*	"assigning"				
			NP	NP	NOUN*	"fixed-length"		
				NOUN*	"codes"			
				PP	PREP	"to"		
					NP	NOUN*	"larger"	
					NOUN*	"subsets"		
					PP	PREP	"of"	
						AJP	ADJ*	"adjacent"
						NOUN*	"characters"	
	PUNC	"."						

a) Parse Tree Obtained for Sample Input Sentence

DECL	NP	AJP	ADJ*	"numeric"			
		NOUN*	"coding"				
	VERB*	"extends"					
		DET	ADJ*	"this"			
		NOUN*	"idea"				
		PP	PREP	"by"			
			AJP	ADJ*	"assigning"		
			NP	NOUN*	"fixed-length"		
			NOUN*	"codes"			
			PP	PREP	"to"		
				NP	NOUN*	"larger"	
				NOUN*	"subsets"		
				PP	PREP	"of"	
					AJP	ADJ	"adjacent"
					NOUN*	"characters"	
	PUNC	"."					

b) Alternative Parse Tree

Typical Syntactic Analysis Output

Fig. 1

available ciphertext	:	R:2:A:1	P-0/56/4/69/
ciphertext	:	R:8:A:2	P-0/44/2/14/
ciphertext	:	R:8:A:2	P-0/44/3/15/
ciphertext	:	R:8:A:2	P-1/30/1/41/
ciphertext	:	R:8:A:2	P-1/30/3/43/
ciphertext	:	R:8:A:2	P-1/30/6/46/
ciphertext	:	R:8:A:2	P-1/32/2/50/
ciphertext	:	R:8:A:2	P-1/34/2/62/
ciphertext	:	R:8:A:2	P-1/47/5/30/
ciphertext	:	R:8:A:2	P-1/75/1/92/
ciphertext	:	R:8:A:2	P-1/9/3/47/
ciphertext	:	R:8:P:1:A:1	P-0/30/4/44/
ciphertext	:	R:8:P:1:A:1	P-1/4/3/13/
ciphertext	:	R:8:P:1:A:2	P-0/28/4/32/
ciphertext	:	R:8:P:1:A:2	P-1/28/1/29/
ciphertext	:	R:8:P:1:A:2	P-1/29/7/40/
ciphertext	:	R:8:P:1:A:2	P-0/60/1/79/
ciphertext character	:	R:1:A:2	P-0/30/5/45/
ciphertext character	:	R:1:P:1:A:2	P-0/36/3/70/
ciphertext letter groups	:	R:4:A:1	P-0/47/3/28/
encrypted ciphertext	:	R:1:A:2	P-0/4/1/11/
single ciphertext characters	:	R:3:P:1:A:1	P-1/47/1/26/
circumstances	:	R:8:P:1:A:4	P-0/78/4/104/
class	:	R:8:A:3	P-0/59/1/75/
problem class	:	R:2:A:4	P-1/59/2/76/
clear text	:	R:2:P:1:A:2	P-0/44/3/15/
German cipher clerk	:	R:5:A:2	P-1/46/2/24/
receiving clerk	:	R:2:A:3	P-1/43/5/11/1
German cipher clerks	:	R:5:A:2	P-1/46/3/25/
cipher clerks	:	R:1:A:2	P-0/43/3/9/
cipher clerks	:	R:1:A:2	P-1/41/1/1/
cipher clerks	:	R:1:A:2	P-1/41/3/3/
cipher clerks	:	R:1:A:2	P-1/43/4/10/
clockwise	:	R:8:A:1	P-0/38/2/75/
code	:	R:8:A:2	P-1/22/4/10/
code	:	R:8:A:4	P-1/38/1/74/
code WQSEUP	:	R:5:A:2	P-0/43/3/9/
code WQSEUP	:	R:5:A:2	P-1/43/4/10/
code book	:	R:1:A:2	P-1/47/5/30/
code book transformation	:	R:1:A:3	P-1/47/5/30/
code books	:	R:1:A:2	P-0/47/3/28/
code books	:	R:1:A:2	P-1/47/6/31/
code breaking	:	R:1:A:1	P-1/14/2/69/
code groups	:	R:1:A:1	P-0/47/3/28/
code specification	:	R:1:A:1	P-0/26/3/22/
code use	:	R:4:A:2	P-1/47/7/32/

Excerpt from Raw Phrase List Derived from Syntactic Parse Trees

Fig. 2

Input	Syntactic tags	Number of tag sequences	
a	noun, article, preposition	3	
fat	adjective, noun, transitive verb	3	9
little	adjective, adverb, noun	3	27
European	noun, adjective	2	54
bird	noun, intransitive verb	2	108

Syntactic Tag Assignments for Sample Phrase
"a fat little European bird"

Table 1

Indexing Entry		Frequency	Paragraph References	Marginal * Poor Term * *
string	character	2	(58, 59)	*
terminal	characters	2	(47, 51)	*
text	characters	2	(36)	*
frequency	characteristics	3	(9, 34, 62)	
occurrence	characteristics	5	(21, 23, 28, 71)	
state language	characteristics	1 (title)	(4)	
	code assignment	4	(57, 59, 71, 76)	
	code combination	7	(94, 56, 57, 64)	*
	code length	18	(54, 62-66, 68, 69)	
appending	code increments	2	(64)	* *
corresponding	code	6	(64, 65, 68)	* *
fixed-length	code	11 (title)	(45, 52, 53, 60, 68, 72, 76)	
Huffman	code	9	(69, 70, 71-73, 76, 82, 84)	
unique	code	2	(36)	* *
variable-length	code	8 (title)	(52, 61, 62, 64, 66, 67, 69)	

a) Excerpt from Automatic Index (Chapter 5 [26])

Indexing Entry		Frequency	Paragraph Reference	Marginal * Poor Term **
frequency	analysis	4	(22, 25, 27, 28)	
THE CAT	ATE THE RAT	2	(42, 46)	* *
ciphers	based	1	(58)	* *
	bit key length	2	(55, 59)	* *
input	bits	2	(17, 54)	*
message	block	3	(66, 68, 78)	*
character	blocks	2	(15, 68)	*
data	blocks	3	(22, 49)	*
code	book	5	(47)	
P-	box	2	(15)	
	British cryptanalyst	2	(42, 46)	
communication	channel	2	(3, 7)	
	character combination	6	(32, 33, 34)	*
	character permutation	2	(35, 78)	
	character position	10	(15, 19, 21, 26, 33, 35, 39)	*
	character sequence	2	(30, 34)	*
	character substitution	2	(17,22)	*

b) Excerpt from Automatic Index (Chapter 6 [26])

Excerpts from Automatically Constructed Indexes

Table 2

Indexing		Entry
base	case	12
shift	case	12
master	character	13-14
	character frequency	1-7
combining	character	13-14
	code efficiency	22
minimum redundancy	code	17-19
digram	coding	13
fixed-length	coding	10, 12-14
numeric	coding	14
variable-length	coding	10, 14-19
	combining character	13-14
	communications theory	6
data	compaction	8-10
	compression ratio	8

a) Excerpt from Manual Index (Chapter 5 [26])

Indexing Entry		Page Reference
Euclidian	algorithm	23
Euler	algorithm	23
	authentication	1
code	book	17
P-	box (see permutation)	
permutation	box	6, 7, 18, 19
S-	box (see permutation)	
substitution	box	6, 7, 18, 19
code	breaking	15, 16, 17
knapsack	cipher	24, 25, 26, 27
permutation	cipher	9
polyalphabetic	cipher	9
product	cipher	7
RSA	cipher	21, 22, 23, 24
running key	cipher	10, 11
stream	cipher	7

b) Excerpt from Manual Index (Chapter 5 [26])

Excerpts from Manually Constructed Indexes

Table 3

Automatic index (phrases only)	total entries	180	220
	distinct entries	83	108
	good terms	29	48
	marginal terms	27	43
	poor terms	27	17
Manual index (single terms and phrases)	total entries	85	96
	distinct entries	58	56
	Manaul index entries also included in automatic index	31/58 (53%)	42/56 (75%)
	Good indexing entries in automatic index not in manual index	6/29 (21%)	13/48 (27%)

Comparison between Automatic and Manual Indexes

Table 4

Chapter 5 (text compression)	Chapter 6 (text encryption)
1. Phrases included in manual index that are missing from automatic index	
language ambiguity	Euclidian algorithm
ASCII code	authentication
base case	Euler totient function
code efficiency	code breaking
minimum redundancy code	key wheel
differential coding	digital signature
Zipf law	
2. Phrases included in automatic index that are missing from manual index	
code length	frequency analysis
decoding dictionary	British cryptanalyst
target frequency	decoding machine
low-frequency words	reversing drum
semantic redundancy	inverse transformation
	language redundancy
	pin setting

Comparison of Indexing Entries

Table 5

Query Statistics	Chapter 5 (text compression)		Chapter 6 (text encryption)	
Number of queries processed	33		30	
Number of queries answered correctly by manual as well as automatic index	18	(55%)	24	(80%)
Number not answered by either index	2	(6%)	0	(0%)
Number answered only in manual index	11	(33%)	5	(17%)
Number answered only in automatic index	2	(6%)	1	(3%)

Retrieval Test Results

Table 6