Browsing and Searching Compressed Documents

Raymond Wan

September 24, 2003

Motivation
Compression through Recursive Pairing
Word-based RE-PAIR
Phrase Browsing
Locating Phrase Contexts
Compression Results
Conclusion

Motivation

Two goals:

- Effectively compress text documents.
- Browse and search compressed documents while minimising the level of decompression.

Documents first compressed with an off-line dictionary-based algorithm called RE-PAIR.

Phrases derived from the compression process are browsed.

After a phrase of interest has been located, decode its context within the document.

Compression through REcursive PAIRing

RE-PAIR [Larsson and Moffat, 2000] compresses documents by recursively replacing the most frequently occurring pairs of adjacent symbols with a new symbol, until no pair of symbols occur in the text twice.

Example:

a b r a c a d a b r a	① → a b
① r a c a d ① r a	② → r a
① ② c a d ① ②	③ → ① ②
③ c a d ③	

The RE-PAIR process produces two outputs: a *phrase hierarchy* and a *reduced sequence*.

Phrase hierarchy gives details about the symbols that occur in the message and the relationships they form with other symbols. Sequence provides the contexts for the phrases.

Phrase hierarchy suitable for browsing; searching applied to sequence.

Word alignment of phrases

RE-PAIR phrases for the “Woodchuck” tongue twister:

14 → ␣|w
15 → ␣|m
16 → ␣|a
17 → c|o
18 → c|k
19 → c|h
20 → l|d
21 → o|d
22 → i|f
23 → ␣w|o
24 → ␣a|s
25 → u|ld
26 → u|ch
27 → ch|u
28 → uld|␣
29 → ␣m|uch
30 → chu|ck
31 → ␣wo|od
32 → chuck|␣
33 → ␣a|␣wood
34 → ␣much|␣wood
35 → chuck|␣wood
36 → chuck␣|if
37 → uld␣|chuck␣wood
38 → ␣a␣wood|chuck␣
39 → ␣a␣woodchuck␣|co
40 → ␣a␣woodchuck␣co|uld␣chuck␣wood
41 → chuck␣if|␣a␣woodchuck␣could␣chuck␣wood

Word alignment of phrases (continued)

One solution is to guide the pairing process by adding heuristics which label each character as a word character or a non-word character.

Rules ensure that:

- clusters of words and non-words must be “complete” before combining with characters of another type.
- complete words can only combine with complete non-words on their right

Word-based RE-PAIR

Better solution: Apply a word-based parsing mechanism prior to RE-PAIR. Parse each message into an alternating sequence of words and non-words.

Word tokens are also case folded to lowercase and then stemmed to its root form using Porter's stemming algorithm. Modifiers are produced at both steps for each word token. For example,

woodchuck	0	0	woodchuck
WOODchuck	0	15	woodchuck
probabilities	275,458	0	probabl
SeArChInG	131,072	341	search

Modifiers ensure that the phrase hierarchy can be browsed and that the compression process is lossless.

Word-based RE-PAIR (continued)

Divide the message into six streams:

- word lexicon and word sequence
- non-word lexicon and non-word sequence
- case folding modifiers
- stemming modifiers

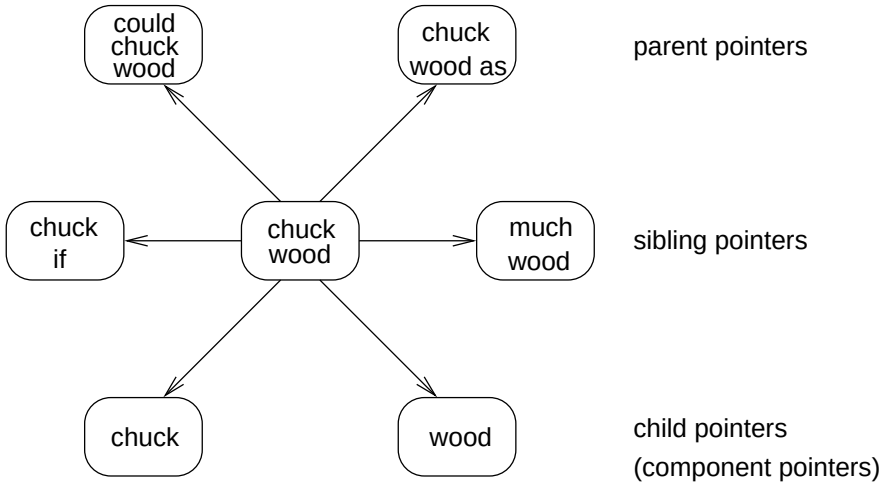
The lexicons includes every distinct word (or non-word), while the sequences consist of references to the lexicon.

The word sequence is compressed with RE-PAIR, so that it is replaced with a phrase hierarchy and a reduced word sequence. In total, there are seven streams available to the phrase browsing system.

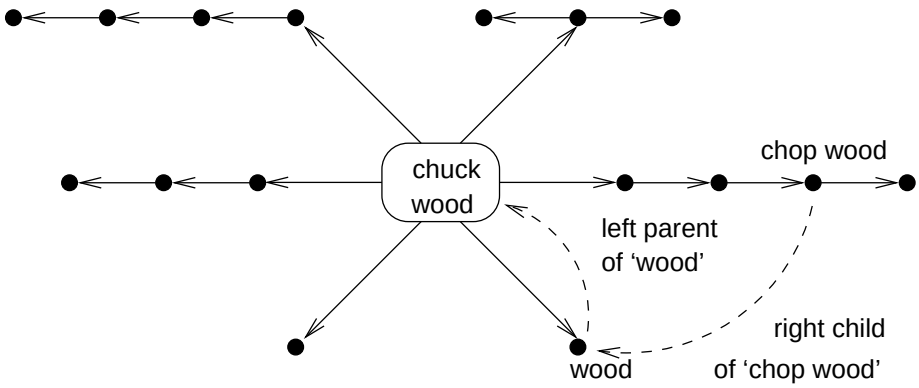
The first step in using the streams is phrase browsing...

Phrase browsing

Phrase browser builds a graph at runtime with each node equal to a symbol in the phrase hierarchy:

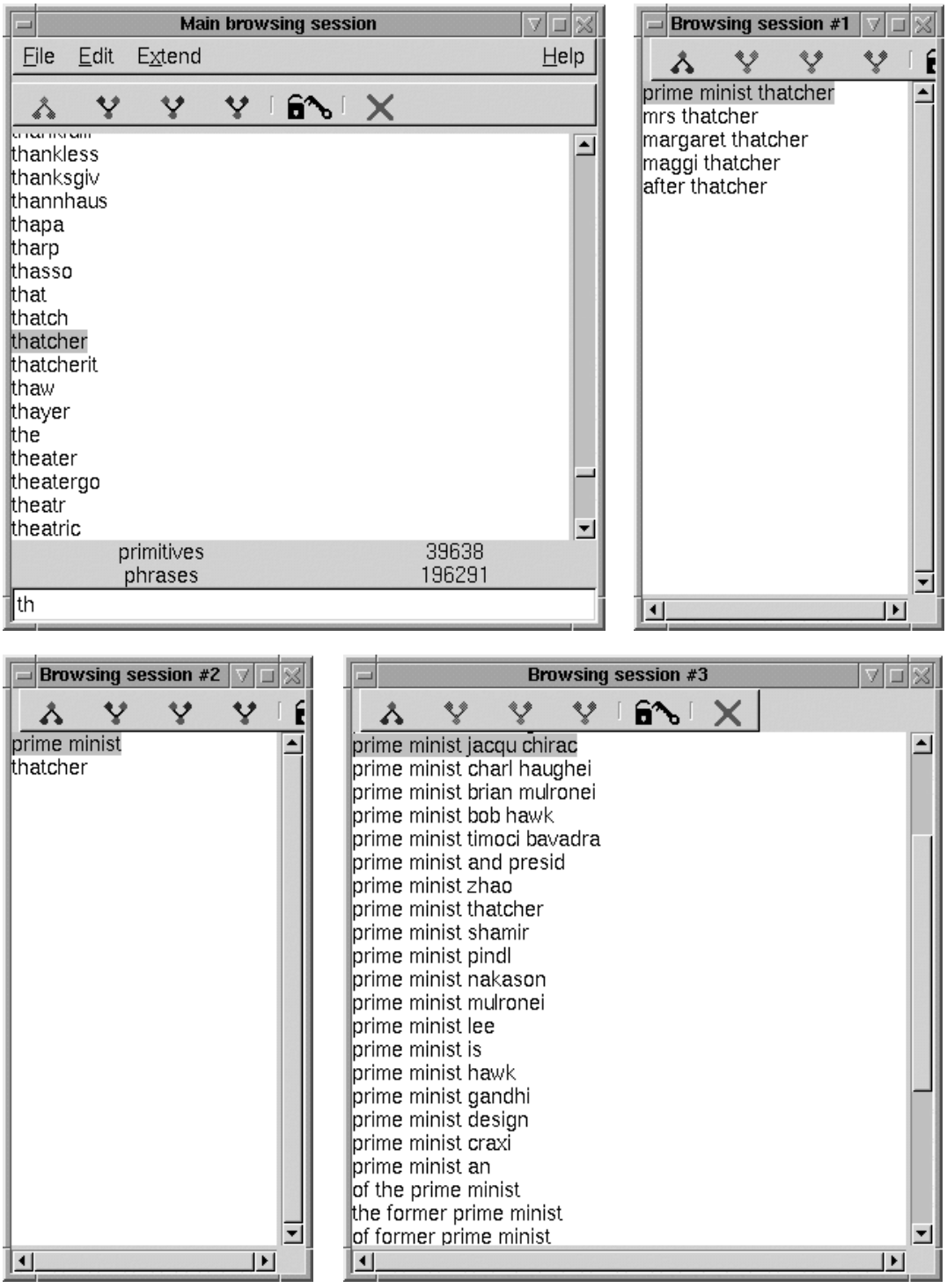


A less detailed look at the node “chuck wood”:



With this structure, phrase browsing is simply a user-directed walk through the graph.

Sample Browsing Session

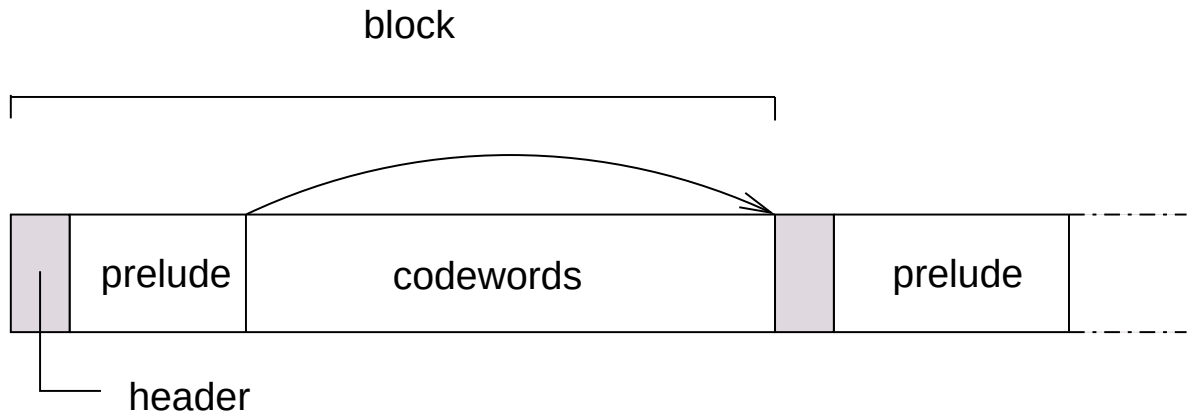


Searching for Phrase Contexts

Reduced word sequence needs to be compressed and searchable.

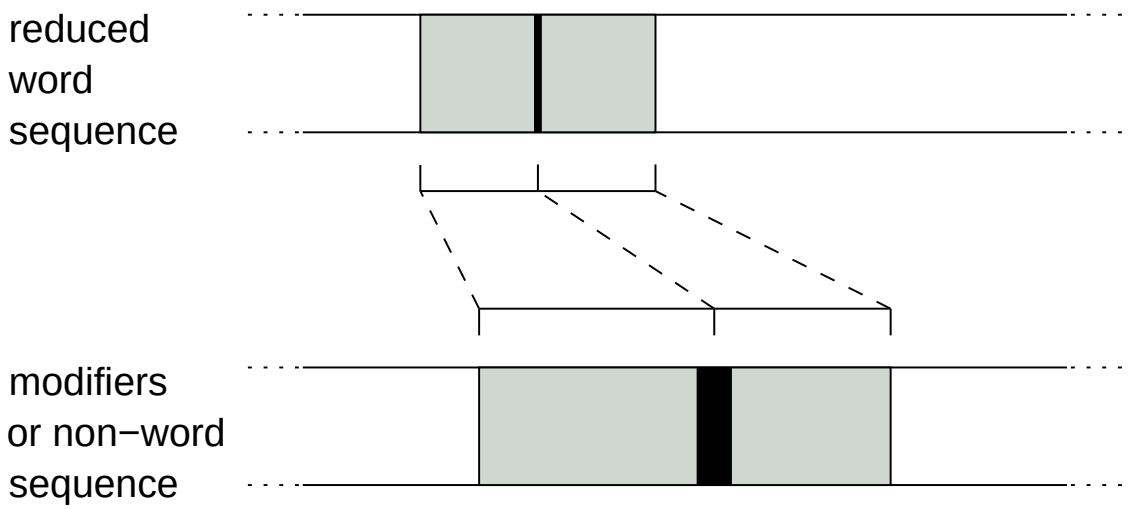
The RE-VIEW coder combines byte-aligned codes with self-contained blocks, each with its own prelude.

Create blocks of 2^{16} distinct symbols each, with a prelude to show the mapping. Prelude encoded as symbol differences using nibble-aligned codes.



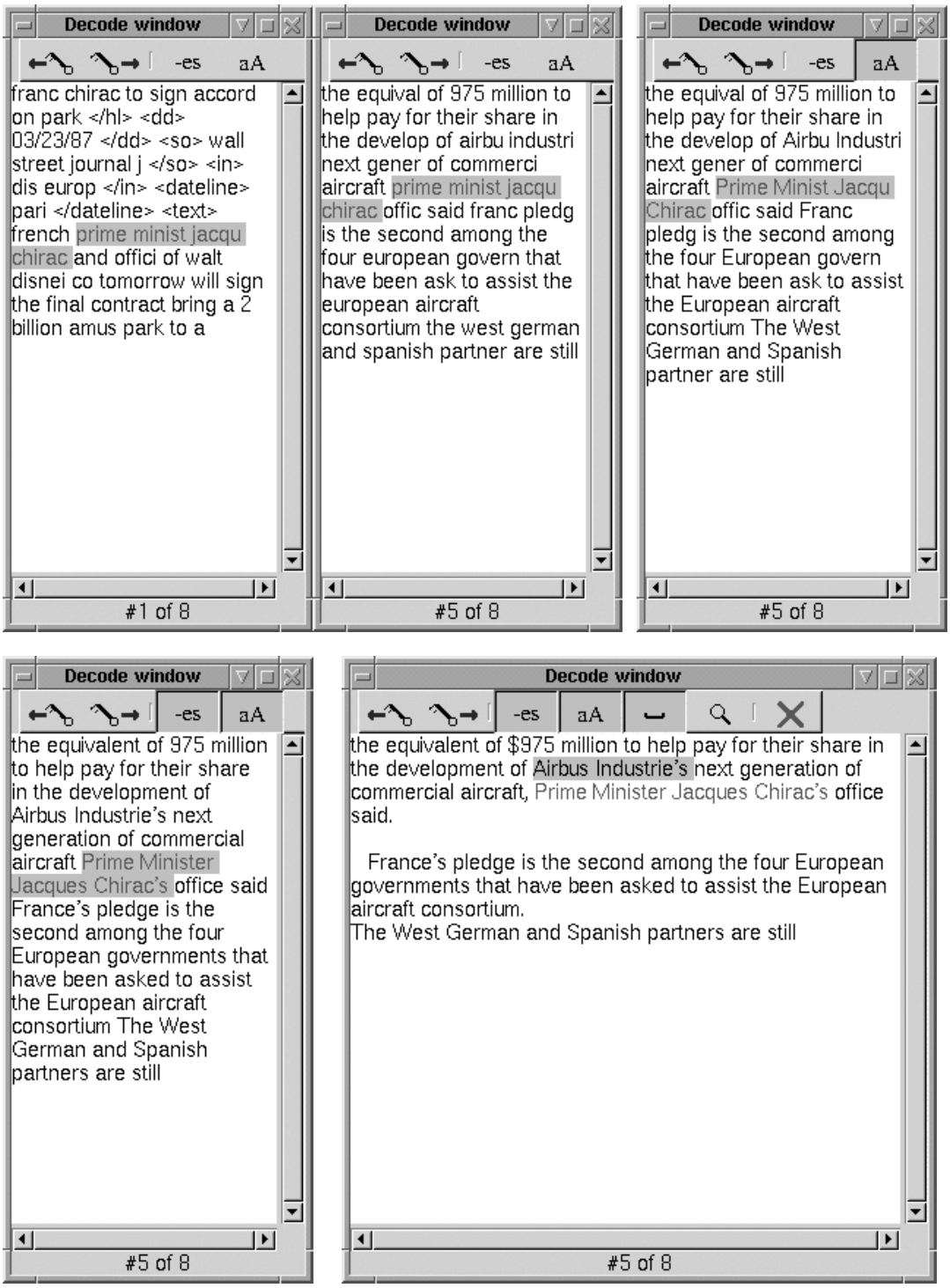
Reversing Stemming, Case Folding, and Non-word Information

For each phrase context found, the corresponding stemming, case folding, and non-word information are required.

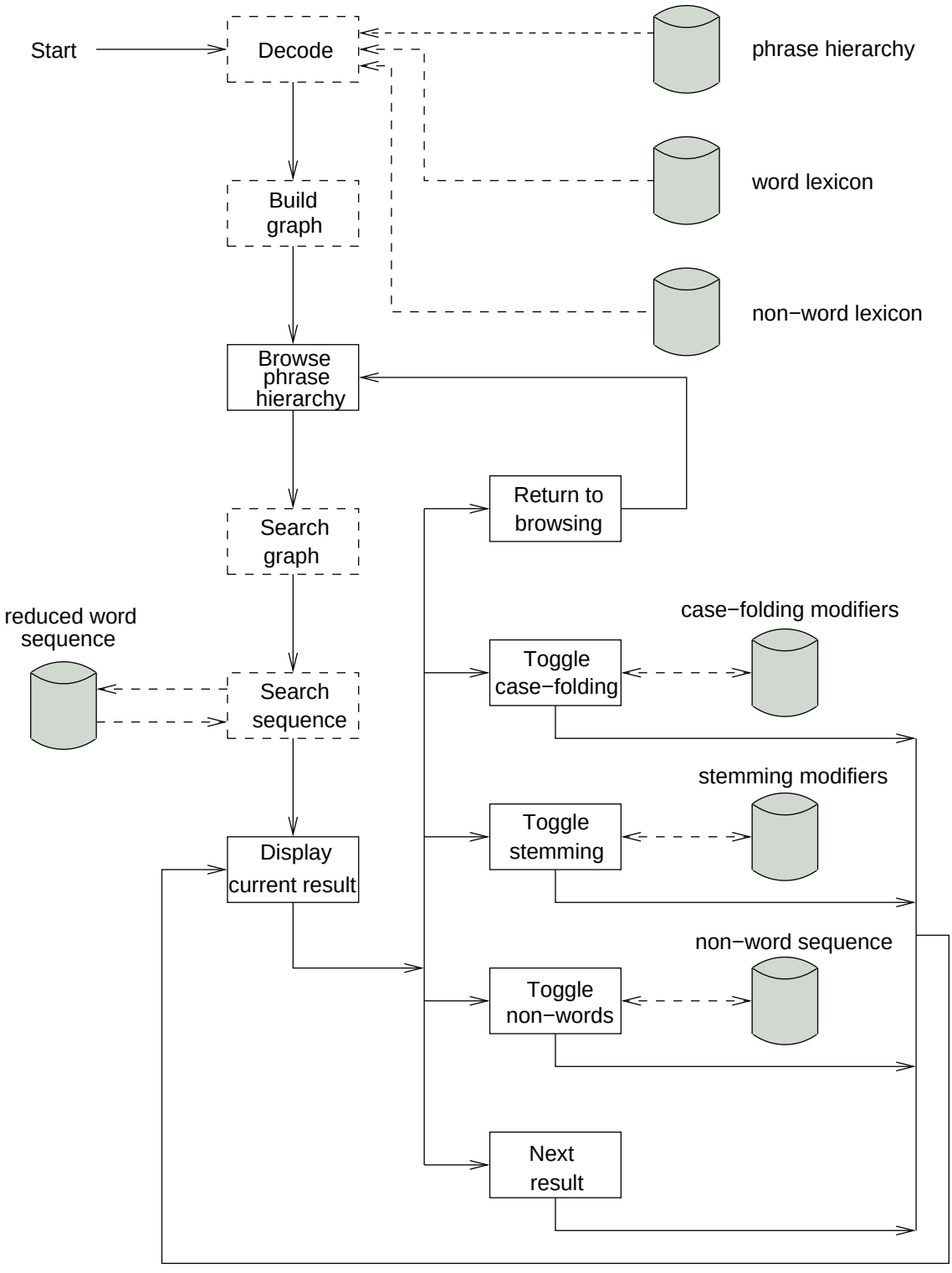


Currently a block-based Huffman coder with a top-level index is employed. Each block has a fixed number of symbols.

Sample Decoding Session



Summary of Browsing and Decoding



Compression Results

Test data is a 20 MB document collection in SGML mark up. Test machine is a 933 MHz Pentium III with 1 GB RAM and 256 kB on-die cache.

	Comp.	Time	
	ratio	Encoding	Decoding
GZIP	2.908	6.4	0.8
BZIP2	2.078	17.9	6.2
PPMD	1.656	31.0	31.5
RE-PAIR (c)	1.768	90.5	2.6

Compression Results (continued)

	Ratio	Encoding	Decoding
parsing	N/A	14.2	2.8
words			
lexicon	0.029	0.1	0.1
RE-PAIR	0.143 1.345	20.7	1.0
case folding	0.235	0.9	1.3
stemming	0.399	2.9	1.4
non-words			
lexicon	0.001	0.1	0.1
sequence	0.325	0.8	1.3
Total	**2.683**	39.7	8.0

Alternative compression mechanisms for each stream can result in a compression ratio of 2.140 bpc.

The phrase hierarchy and the two lexicons make up 2.2% of the original message, or 6.4% of the compressed document.

Related Work

Merging compressed RE-PAIR blocks with RE-MERGE, so that document collections of up to 1 GB can be browsed.

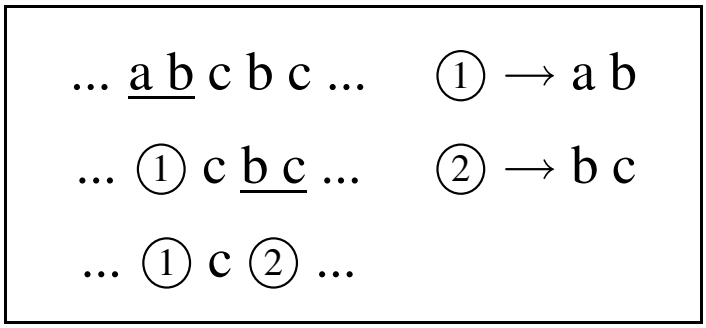
Phrase selection through semantic or statistical means.

- LINKIT [Wacholder et al., 2001] and KEA [Frank et al., 1999] use NLP or machine-learning techniques to identify phrases for browsing.
- SEQUITUR selects phrases based on frequency, and its phrase browser is called PHIND [Nevill-Manning et al., 1997]
- Nextword indexes is a data structure which supports phrase browsing [Bahle et al., 2002].
- Combining semantic and statistical methods are also possible [Paynter et al., 2000].

Areas for Further Improvement

Some areas for further work:

- Improving compression effectiveness for the individual components of word-based RE-PAIR.
- Developing heuristics for selecting phrases for display within the browser.
- Problem with recall during decoding:



Conclusion

Phrase browsing with RE-PAIR demonstrates the compromises required between compression and retrieval.

Phrase browsing of compressed documents with minimal decompression was demonstrated.
Word-based parsing with case folding and stemming, as well as coding mechanisms, biased towards efficient searching were used.

Also, no explicit indexing structures are created for browsing.

References

D. Bahle, H. E. Williams, and J. Zobel. Efficient phrase querying with an auxiliary index. In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. H. Myaeng, editors, *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–221, Tampere, Finland, August 2002.

E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-Specific keyphrase extraction. In *Proc. International Joint Conference on Artificial Intelligence*, pages 668–673, 1999.

N. J. Larsson and A. Moffat. Offline dictionary-based compression. *Proc. IEEE*, 88(11):1722–1732, November 2000.

C. G. Nevill-Manning, I. H. Witten, and G. W. Paynter. Browsing in digital libraries: A phrase-based approach. In *Proc. 2nd ACM Conference on Digital Libraries*, pages 230–236, 1997.

G. W. Paynter, I. H. Witten, S. J. Cunningham, and G. Buchanan. Scalable browsing for large collections: a case study. In *Proc. 5th ACM Conference on Digital Libraries*, pages 215–223, 2000.

M. F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980. Reprinted in Readings in Information Retrieval, pages 313–316, 1997.

N. Wacholder, D. K. Evans, and J. L. Klavans. Automatic identification and organization of index terms for interactive browsing. In E. A. Fox and C. L. Borgman, editors, *Proc. First ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 126–134. ACM Press, 2001.