HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI
MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

LEMPEL-ZIV FACTORIZATION IN EXTERNAL MEMORY

Juha Kärkkäinen
Dominik Kempa
Simon J. Puglisi

For over three decades, the Lempel-Ziv factorization (or LZ77 parsing) has been a fundamental tool for data compression (e.g. in 7-zip). More recently it has become the basis for several compressed text indexes which are particularly effective for massive, highly repetitive data sets. When computing the parsing for such large data sets, the space requirements of algorithms can become a problem. We escape the limitations of RAM by describing the first external memory LZ77 parsing algorithms and present their experimental comparison.

LEMPEL–ZIV FACTORIZATION

Lempel-Ziv factorization $LZ(T)$ of string T is a greedy partition of T into *longest previous factors* (LPFs). LPF at position i is the longest factor $T[i..i+\ell)$ that also occurs at some position $j < i$.

Example:

i	0	1	2	3	4	5	6	7	8	9
T[i]	A	B	A	B	B	A	B	B	A	B

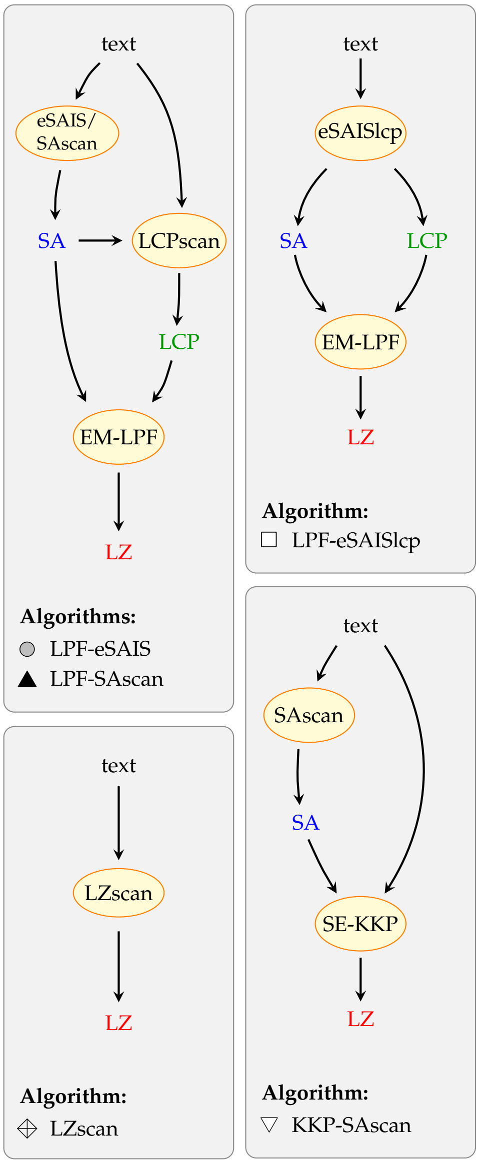
LPF[2] = AB (j = 0)
LPF[5] = ABBAB (j = 2)

LZ(T): | A | B | A B | B A B B A B |

BASIC ALGORITHMS

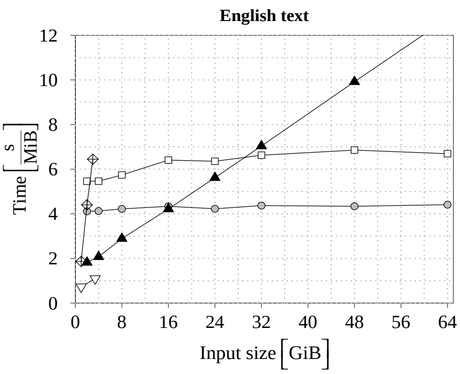
Name	I/O complexity	Space
SAscan [4]	$\mathcal{O}\left(\frac{n}{B}\left(1+\frac{n\log\sigma}{M\log n}\right)\right)$	$6.5n$
eSAIS [1]	$\mathcal{O}\left(\frac{n}{B}\log_{\frac{M}{B}}\frac{n}{B}\right)$	$28n$
eSAISlcp [1]	$\mathcal{O}\left(\frac{n}{B}\log_{\frac{M}{B}}\frac{n}{B}\right)$	$54n$
LCPscan [3]	$\mathcal{O}\left(\frac{n}{B}\left(1+\frac{n\log^2\sigma}{M\log^2 n}\right)\right)$	$16n$
EM-LPF [5, 2]	$\mathcal{O}\left(\frac{n}{B}\log_{\frac{M}{B}}\frac{n}{B}\right)$	$26n$
LZscan [5]	$\mathcal{O}\left(\frac{n}{B}\cdot\frac{n\log\sigma}{M\log n}\right)$	$1.5n$
SE-KKP [5]	$\mathcal{O}\left(\frac{n}{B}\right)$	$21n$

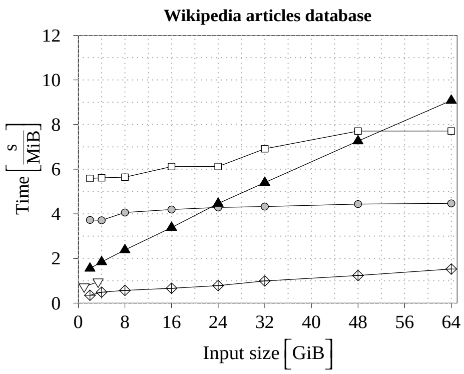
LZ FACTORIZATION ALGORITHMS



EXPERIMENTAL COMPARISON

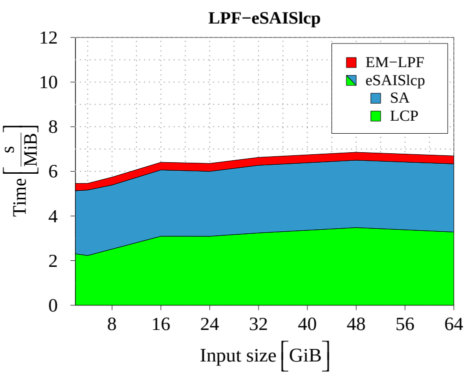
We implemented and compared all LZ factorization algorithms depicted on the right. The algorithms were executed on varying size prefixes of two testfiles: a large data set containing English text (left) and a database of Wikipedia articles containing many versions of the same articles (right). All algorithms were allowed to use 3.5GiB of internal memory. The results depend on the amount of repetitions in the input text. LZscan dominates all other algorithms for highly repetitive input but performs poorly when the data is less repetitive, such as the English test file. The fastest algorithm for such data is determined by the ratio of input size to available RAM.

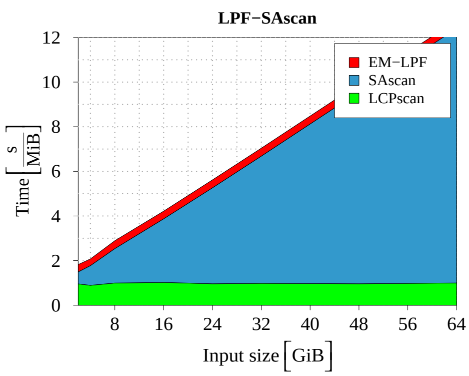




DETAILED RUNTIME BREAKDOWN

Below we present a detailed runtime breakdown of LPF-eSAISlcp and LPF-SAscan executed on English text. The graphs reveal that most of the time is spent during the computation of supporting data structures (SA and LCP). The LCP array construction is significantly accelerated with the use of our new algorithm (LCPscan) which makes SA construction the slowest phase of the factorization. The main challenge in efficient and scalable LZ factorization is therefore developing new methods for suffix sorting, possibly using parallel or distributed computation.





REFERENCES

[1] T. Bingmann, J. Fischer, and V. Osipov. Inducing suffix and LCP arrays in external memory. In *Proc. ALENEX*, pages 88–102, 2013.

[2] M. Crochemore, L. Ilie, and W. F. Smyth. A simple algorithm for computing the Lempel-Ziv factorization. In *Proc. DCC*, pages 482–488, 2008.

[3] J. Kärkkäinen and D. Kempa. LCP array construction in external memory. Accepted to SEA 2014.

[4] J. Kärkkäinen and D. Kempa. Engineering a lightweight external memory suffix array construction algorithm. In *Proc. ICABD*, 2014. To appear.

[5] J. Kärkkäinen, D. Kempa, and S. J. Puglisi. Lempel-Ziv parsing in external memory. In *Proc. DCC*, 2014. To appear.