Building and Checking Suffix Array in External Memory

Yi Wu, Ge Nong, Wai Hong Chan, and Bin Lao

Abstract—Suffix array (SA) can be built within linear time and space by means of the induced sorting (IS) method. The recently proposed IS suffix sorter fSAIS has a performance competitive to that of the state-of-the-art parallel construction algorithms in external memory, which indicates a great potential for improving the other disk-based IS alternatives by engineering them carefully. In view of this, we redesign the reduction phase of DSA-IS by using a new substring sorting and naming method and implement the variant, called DSA-IS+, by using several space-saving techinques previously applied to fSAIS.
A constructed SA should be checked to ensure its correctness. We present in this paper a method that enables any IS suffix sorter to perform building and checking simultaneously, where the consumption for checking is negligible in comparison with that for building in our experiments.

Index Terms—Suffix array, construction and verification, external memory.

1 Introduction

The suffix array (SA) [**?**] is an essential data structure for string processing and information retrieval. Among the existing internal-memory algorithms for SA construction, SA-IS [1] achieves the optimal time and space complexities using the induced sorting principle, where the key operation is to retrieve the preceding characters of sorted suffixes in order for inducing the lexical order of unsorted ones. To handle massive datasets , several external memory algorithms have been proposed for building massive suffix arrays in recent years, e.g., DC3 [**?**], bwt-disk [**?**], SAscan [2], pSAscan [3], eSAIS [**?**], EM-SA-DS [**?**] and DSA-IS [**?**]. Among them, the latter three algorithms are based on the induced sorting (IS) method described in SA-IS [**?**].

This algorithm employs the IS method to induce the lexical order of all the suffixes from a selected subset.

This algorithm uses the induced sorting method to

In the past decades, great effort has been taken to study efficient algorithms for SA construction. th

extensive works have been put on designing time and space efficient suffix sorting algorithms

data structure that has been widely used in many string processing applications, e.g., biological sequence alignment, time series analysis and text clustering. Given an input string, traversing its suffix tree can be emulated by using the corresponding enhanced suffix array [**?**], which mainly consists of the suffix and the longest common prefix arrays. It has been realized that the application scope of an index mainly depends on the construction speed and the space requirement. This leads to intensive works on designing time and space efficient suffix sorting algorithms over the past decade, assuming different computation models such as internal memory, external memory, parallel and distributed models. Particularly,

The basic idea behind the induced sorting method is to induce the lexicographical order of all the substrings/suffixes from a sorted subset of substrings/suffixes. Following the idea, an IS-based suffix sorting algorithm is typically comprised of a reduction phase for sorting and naming substrings to reduce a string $x[0, n)$ to a short string $x1[0, n1)$ with $n1 \leq \frac{1}{2}n$ and an induction phase for sorting suffixes to induce $\mathsf{SA}(x)$ from $\mathsf{SA}(x1)$. During the two phases, the key operation is to retrieve the preceding character of a sorted substring/suffix. This can be done very quickly when x is fully accommodated in the internal memory, but will become slow when x resides in the external memory, as each operation takes a random disk access. For a high I/O efficiency, eSAIS, EM-SA-DS and DSA-IS use different auxiliary data structures to retrieve the preceding characters in a disk-friendly way. Particularly, both eSAIS and EM-SA-DS split a long substring into pieces and represent each piece by a fixed-size tuple, while DSA-IS does not. With an elaborate arrangement of the I/O operations, the programs for these three algorithms are competitive with those for others in terms of both time and space efficiencies.

Among the existing internal-memory suffix sorters, SA-IS [1] achieves the optimal time and space complexities using the induced sorting method, the key operation of which is to retrieve the preceding characters of sorted suffixes in order for inducing the lexical order of unsorted ones. In the past five years, several works adapted SA-IS to design suffix sorters specific for EM models [2], [4], [5], [6]. These variants use different approaches to retrieve the preceding characters by sequential I/O operations. However, they all suffer from a space bottleneck for taking at least twice disk space as SA-IS on real-world datasets. Recently, it was presented in [**?**] a new engineering version of SA-IS that achieves nearly optimal space efficiency, indicating a great

- *Y. Wu, G. Nong (corresponding author) and B. Lao are with the Department of Computer Science, Sun Yat-sen University, Guangzhou 510275, China. E-mails: wu.yi.christian@gmail.com, issng@mail.sysu.edu.cn, Laobin@mail3.sysu.edu.cn.*
- *Wai Hong Chan (corresponding author) is with the Department of Mathematics and Information Technology, The Education University of Hong Kong, Hong Kong. E-mail: waihchan@ied.edu.hk.*

potential for improving the other disk-based IS alternatives by engineering them carefully.

Spae

2 Conclusion

xxx

References

[1] G. Nong, S. Zhang, and W. H. Chan, "Two Efficient Algorithms for Linear Time Suffix Array Construction," *IEEE Transactions on Computers*, vol. 60, no. 10, pp. 1471–1484, October 2011.

[2] J. Kärkkäinen and D. Kempa, "Engineering a Lightweight External Memory Suffix Array Construction Algorithm," in *Proceedings of the 2nd International Conference on Algorithms for Big Data*, Palermo, Italy, April 2014, pp. 53–60.

[3] J. Kärkkäinen, D. Kempa, and S. J. Puglisi, "Parallel External Memory Suffix Sorting," in *In proceedings of the 26th Annual Symposium on Combinatorial Pattern Matching*, Ischia Island, Italy, July 2015, pp. 329–342.

[4] G. Nong, W. H. Chan, S. Zhang, and X. F. Guan, "Suffix Array Construction in External Memory Using D-Critical Substrings," *ACM Transactions on Information Systems*, vol. 32, no. 1, pp. 1:1–1:15, January 2014.

[5] G. Nong, W. H. Chan, S. Q. Hu, and Y. Wu, "Induced Sorting Suffixes in External Memory," *ACM Transactions on Information Systems*, vol. 33, no. 3, pp. 12:1–12:15, March 2015.

[6] W. J. Liu, G. Nong, W. H. Chan, and Y. Wu, "Induced Sorting Suffixes in External Memory with Better Design and Less Space," in *Proceedings of the 22nd International Symposium on String Processing and Information Retrieval*, London, UK, September 2015, pp. 83–94.