Masters's Thesis

Scalable Distributed String Sorting

Overview

Sorting is one of the most fundamental problems in algorithms. Here, we want to find a globally ordered permutation of the input with respect to a given comparison function (usually "less than"). However, most of the literature considers the input to be atomic, i.e., elements can be compared in constant time. When sorting strings, it is not possible to compare two arbitrary strings in constant time. Instead, at least the distinguishing prefix of both strings has to be considered, which describes the minimum number of characters that have to be compared to distinguish both strings.

This greatly differentiates string sorting from sorting atomic elements. There exists work on string sorting in shared [2, 1] and distributed memory [4, 5]. However, there are no distributed memory algorithms that scale beyond thousands of cores.

Furthermore, string sorting is closely related to suffix sorting. Here, we want to sort all suffixes of the input lexicographically. While there exist linear time distributed memory suffix array construction algorithm [**?**], they have not been implemented in a scaling fashion and the only practical implementations are quasilinear algorithms [3, 5, 6].

Objective

The main objective of this Master's thesis is to develop a scaling string sorting algorithms. To this end, features like longest common prefix compression and other tools used in previous work [**?**] should be used. Another goal is to develop a space-efficient distributed memory string sorting algorithm that does not rely on all string that should be sorted are available all the time, i.e., they only exist in a compressed form and have to be decompressed to be compared.

The space-efficient distributed memory string sorting algorithm can be used to implement a linear time distributed memory suffix array construction algorithm. This is an optional goal of this thesis.

Requirements

- Excellent C++ and MPI programming skills
- Interest in string algorithms and compact data structures

Contact

Dr. Florian Kurpicz (kurpicz@kit.edu)

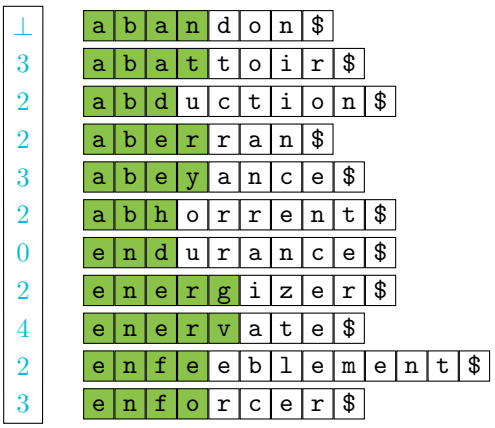

Figure 1: Longest common prefix (blue) and distinguishing prefixes (green) of a set of strings.

References

[1] Jon Louis Bentley and Robert Sedgewick. Fast algorithms for sorting and searching strings. In *SODA*, pages 360–369. ACM/SIAM, 1997.

[2] Timo Bingmann. *Scalable String and Suffix Sorting: Algorithms, Techniques, and Tools*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2018.

[3] Timo Bingmann, Simon Gog, and Florian Kurpicz. Scalable construction of text indexes with thrill. In *IEEE BigData*, pages 634–643. IEEE, 2018.

[4] Timo Bingmann, Peter Sanders, and Matthias Schimek. Communication-efficient string sorting. In *IPDPS*, pages 137–147. IEEE, 2020.

[5] Johannes Fischer and Florian Kurpicz. Lightweight distributed suffix array construction. In *ALENEX*, pages 27–38. SIAM, 2019.

[6] Patrick Flick and Srinivas Aluru. Parallel distributed memory construction of suffix and longest common prefix arrays. In *SC*, pages 16:1–16:10. ACM, 2015.