Computing the optimal BWT using SAIS

Davide Cenzato and Zsuzsanna Lipták

University of Verona, Department of Computer Science, Verona, Italy,
{davide.cenzato,zsuzsanna.liptak}@univr.it

In the last few decades, the advance in sequencing technologies has dramatically reduced the cost for DNA sequencing, leading to never-before-seen amounts of genomic data. As a consequence, the focus has shifted from individual sequences to large collections of (often very similar) sequences, such as in the 1000 Genomes Project [1], the 10,000 Genomes Project [2], or the 100,000 Human Genomes Project [3]. One of the most effective ways to address this challenge consists in exploiting the repetitiveness in biological data. In this context, the Burrows-Wheeler-Transform plays a central role, since it allows querying the data while keeping the input compressed, if possible, in space proportional to the number r of runs of the BWT.

The BWT was originally defined for individual strings, and it is not immediately clear how to extend it to a string collection. In our recent work [4], we studied the different methods currently in use and showed that there were extensive differences in the resulting transforms. These differences extend to the parameter r, which is fundamental in data structures built on the BWT, such as the r-index [5]. We also showed that two of the most commonly used methods for defining the BWT of string collections depend on the input order; in other words, if the order of the input sequences is permuted, then the resulting transform will be different.

Bentley, Gibney, and Thankachan [6] gave a linear-time algorithm for computing the permutation of the input strings that minimizes r, without providing a practical implementation. Here we present our implementation, combining their algorithm with our results of [4] and an adaptation of the well-known Suffix Array Induced Sorting (SAIS) algorithm of Nong et al. [7]. We evaluated our algorithm on 32 million SARS-Cov-2 short reads of length 50 using 7 sets containing 2^i million strings for $i = -1, 0, 1, \dots, 5$. We compared it with `gsufsort`, a well-known tool that uses a variant of SAIS as a subroutine for computing the BWT of string collections. Our algorithm is time and space competitive with `gsufsort`, and always produces a BWT with fewer runs. In particular, on the largest string collection the optimal BWT has 14.2 times fewer runs than the one output by `gsufsort`.

Ours is the first tool for computing the BWT of a string collection that guarantees the fewest possible runs, and is thus optimal as a basis of data structures built on the BWT.

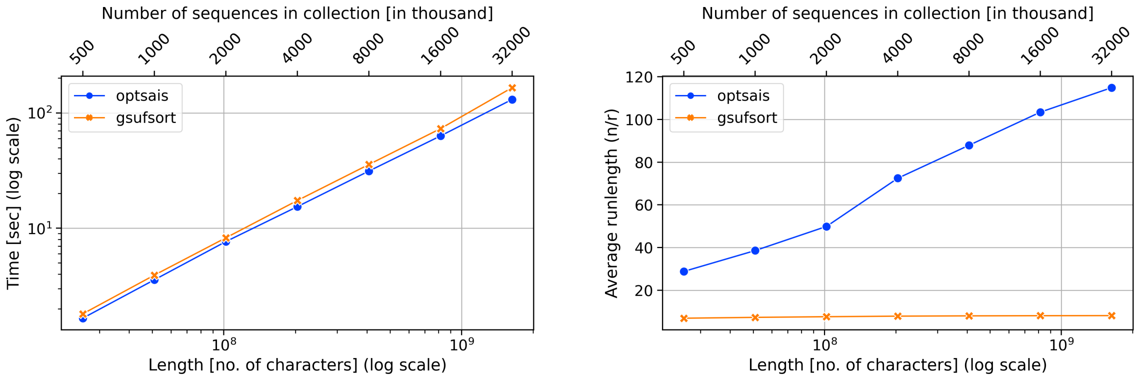

Figure 1: Construction CPU time (left) and average runlength of the BWT (right) on 32 million SARS-CoV2 short reads. We compare our implementation `optsais` with the `gsufsort` tool.

References

[1] The 1000 Genomes Project Consortium, "A global reference for human genetic variation," *Nature*, vol. 526, pp. 68–74, 2015.

[2] Genome 10K Community of Scientists, "A proposal to obtain whole-genome sequence for 10,000 vertebrate species," *J Hered.*, vol. 100:659-674, 2009.

[3] C. Turnbull et al., "The 100,000 genomes project: bringing whole genome sequencing to the NHS," *Br Med J*, vol. 361, 2018.

[4] Davide Cenzato and Zsuzsanna Lipták, "A theoretical and experimental analysis of BWT variants for string collections," in *Proc. of 33rd Annual Symposium on Combinatorial Pattern Matching, (CPM 2022)*, 2022, vol. 223 of *LIPIcs*, pp. 25:1–25:18.

[5] Travis Gagie, Gonzalo Navarro, and Nicola Prezza, "Optimal-time text indexing in BWT-runs bounded space," in *Proc. of SODA 2018*, 2018, pp. 1459–1477.

[6] Jason W. Bentley, Daniel Gibney, and Sharma V. Thankachan, "On the complexity of BWT-runs minimization via alphabet reordering," in *Proc. of 28th Annual European Symposium on Algorithms (ESA 2020)*, 2020, vol. 173 of *LIPIcs*, pp. 15:1–15:13.

[7] Ge Nong, Sen Zhang, and Wai Hong Chan, "Two efficient algorithms for linear time suffix array construction," *IEEE Trans. Computers*, vol. 60, no. 10, pp. 1471–1484, 2011.