# Efficiently Merging $r$-indexes

Marco Oliva[1], Massimiliano Rossi[1], Jouni Sirén[2], Giovanni Manzini[3], Travis Gagie[4], and Christina Boucher[1]

[1] CISE Department, University of Florida, US
[2] Genomics Institute, University of California Santa Cruz, US
[3] Department of Science and Technological Innovation, University of Eastern Piedmont, Italy
[4] Faculty of Computer Science, Dalhousie University, Canada

Advances in DNA sequencing technologies have led us to generate genetic data at an unprecedented pace, moving the bottleneck towards the analysis. Currently, widely used read alignment methods — such as Bowtie and BWA [3, 4] — cannot index thousands of genomes because they require linear space for construction. Gagie, Navarro, and Prezza's $r$-index [2] on the other hand, by exploiting the repetitiveness of genetic databases, allows us to index huge datasets with a small memory footprint, making them easier to handle. In fact, the index of a genetic database of ∼80GB can be shrunk to less than 2GB when properly compressed. Moreover, large sequencing projects — such as GenomeTrakr and MetaSub — get updated frequently (sometimes daily, in the case of GenomeTrakr) with new data, requiring any index over the data to be re-built.

Bannai, Gagie, and I proposed the dynamic $r$-index [1], a data structure that supports the incremental construction of the index. However, this tool is not powerful enough to support substantial updates. Our approach, called RIMERGE, is the result of the combination of the theoretical findings of the dynamic $r$-index with a known algorithm for merging Burrows-Wheeler transforms (BWTs) [5].

The main challenge in merging the $r$-index succinctly and efficiently is identifying all SA samples in the merged index. While merging two BWTs it may happen that some samples in the original BWTs are not samples in the resulting BWT or, vice versa, SA samples in the resulting BWT are not samples in the original ones. In this latter case such samples have to be efficiently retrieved. We show how to preserve the $r$-index structure computing the candidate Suffix Array (SA) samples while computing the Rank Array (RA) and storing them into a self-balanced tree. RIMERGE performs batch updates that allow us to exploit parallelism while keeping the memory overhead small. Our experiments aim to evaluate the scenario where an index has already been built for a large dataset and multiple updates are performed at a later time. We show that RIMERGE requires less time and less memory for updating than rebuilding the index from scratch, e.g., it requires approximately 4 times less time when inserting 64 new sequences of Chromosome 19 in a pre-built index of 1000 sequences.

Lastly, we show that the theoretical results that allow us to merge two $r$-indices can be used to support deletion as well. The information contained in the RA can be used to mark the sequences that we want to remove. In this way, during the interleave step, instead of inserting the characters from the second index we will remove them.

## References

1. Bannai, H., Gagie, T., I, T.: Refining the r-index. Theoretical Computer Science **812**, 96–108 (2020)
2. Gagie, T., Navarro, G., Prezza, N.: Fully functional suffix trees and optimal text searching in bwt-runs bounded space. Journal of the ACM (JACM) **67**(1), 1–54 (2020)
3. Langmead, B., Salzberg, S.: Fast gapped-read alignment with Bowtie 2. Nature Methods **9**(4), 357–359 (2012)
4. Li, H., Durbin, R.: Fast and accurate short read alignment with Burrows–Wheeler Transform. Bioinformatics **25**(14), 1754–1760 (2009)
5. Sirén, J.: Compressed suffix arrays for massive data. In: International Symposium on String Processing and Information Retrieval. pp. 63–74. Springer (2009)