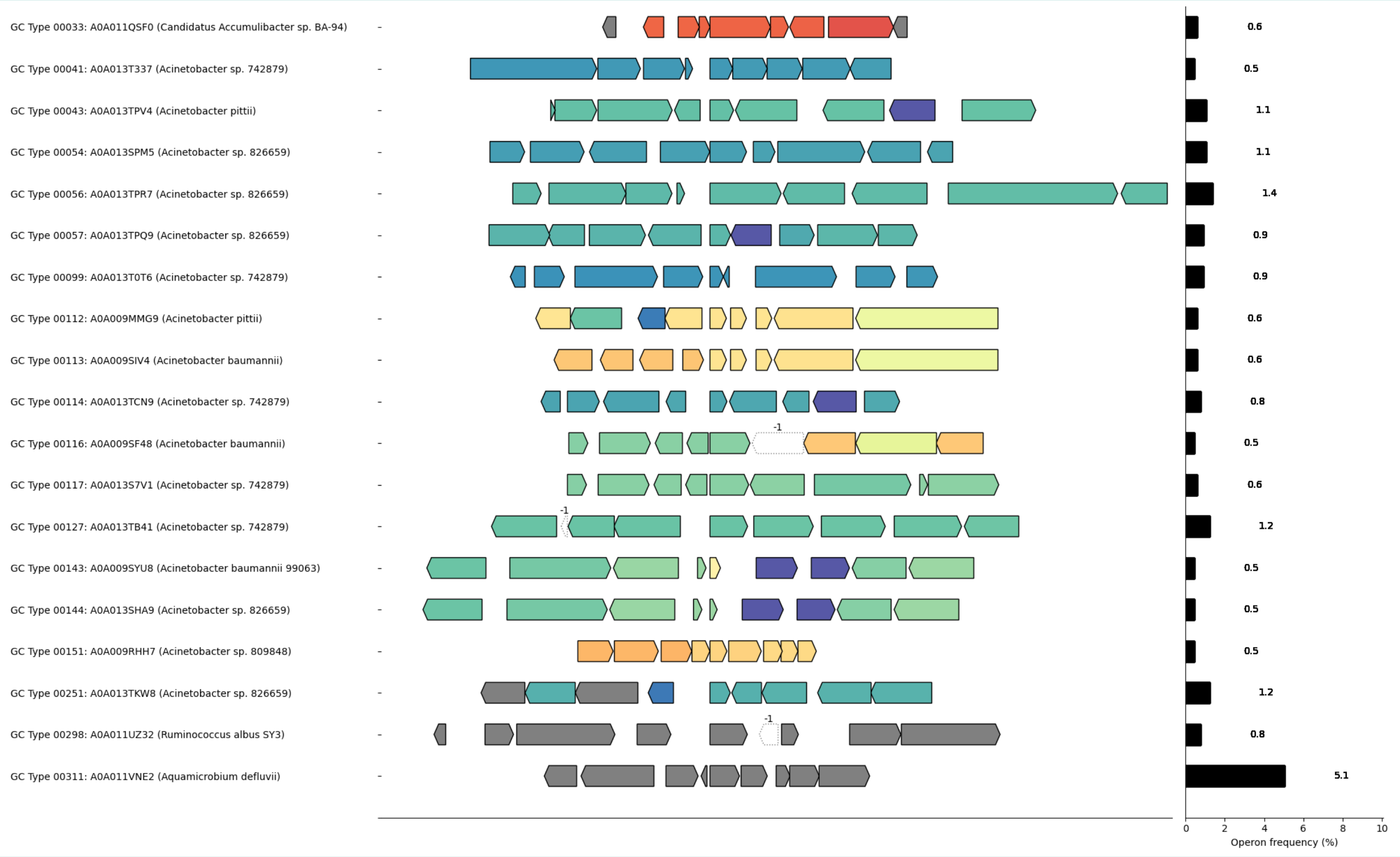
Scalable Genomic Context Analysis with GCsnap2 on HPC Clusters

Reto Krummenacher[1], Osman Seckin Simsek[1], Michèle Leemann[2,3], Leila T. Alexander[2,3], Torsten Schwede[2,3], Florina M. Ciorba[1], and Joana Pereira[2,3]

[1]Department of Mathematics and Computer Science, University of Basel, Switzerland [2]Biozentrum, University of Basel, Switzerland [3]SIB Swiss Institute of Bioinformatics, Basel, Switzerland

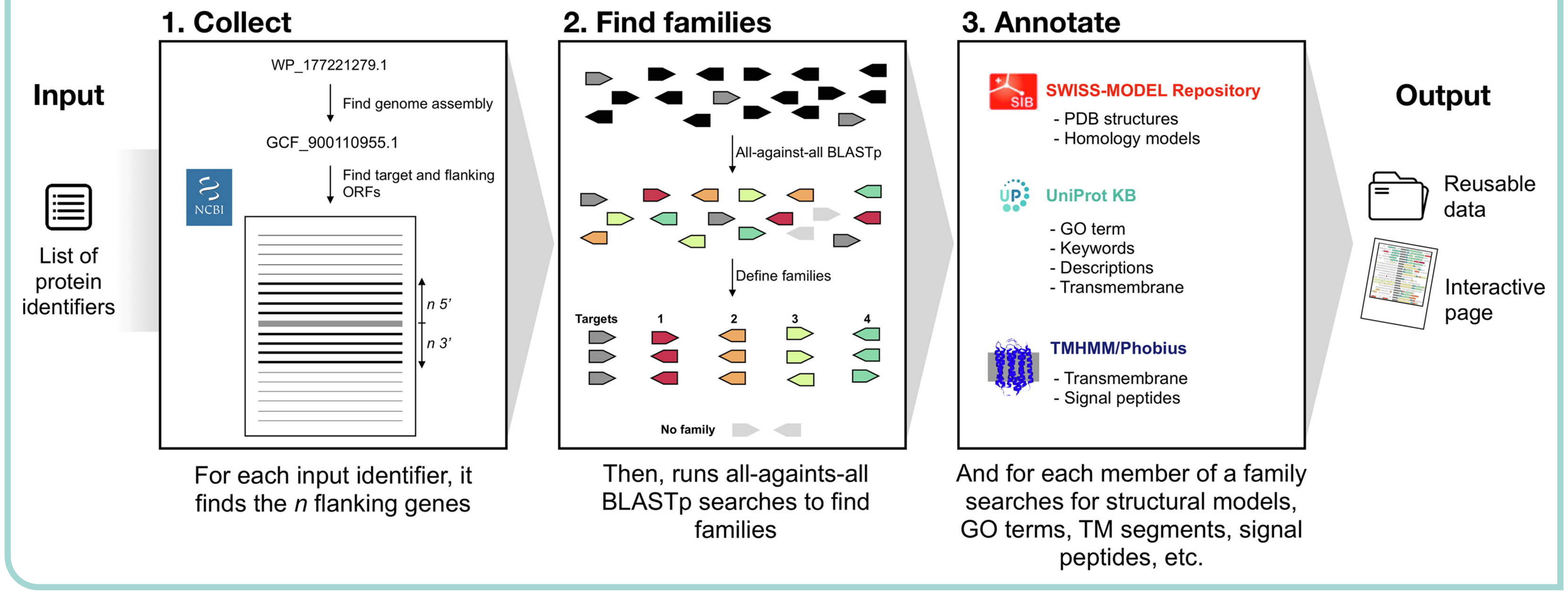
1. Genomic Context Analysis

- Genomic context analysis studies the genomic neighborhood of a specific protein-coding gene by analyzing which are nearby protein-coding genes that may be associated with the target protein.
- When applied across multiple species, comparing such genomic neighborhoods can assist in predicting a protein's biological function [1].
- By investigating patterns of gene presence and conservation in these neighborhoods across species, functional associations between proteins can be inferred.


GC Type 00033: A0A011QSF0 (Candidatus Accumulibacter sp. BA-94)
GC Type 00041: A0A013T337 (Acinetobacter sp. 742879)
GC Type 00043: A0A013TPV4 (Acinetobacter pittii)
GC Type 00054: A0A013SPM5 (Acinetobacter sp. 826659)
GC Type 00056: A0A013TPR7 (Acinetobacter sp. 826659)
GC Type 00057: A0A013TPQ9 (Acinetobacter sp. 826659)
GC Type 00099: A0A013T0T6 (Acinetobacter sp. 742879)
GC Type 00112: A0A009MMG9 (Acinetobacter pittii)
GC Type 00113: A0A009SIV4 (Acinetobacter baumannii)
GC Type 00114: A0A013TCN9 (Acinetobacter sp. 742879)
GC Type 00116: A0A009SF48 (Acinetobacter baumannii)
GC Type 00117: A0A013S7V1 (Acinetobacter sp. 742879)
GC Type 00127: A0A013TB41 (Acinetobacter sp. 742879)
GC Type 00143: A0A009SYU8 (Acinetobacter baumannii 99063)
GC Type 00144: A0A013SHA9 (Acinetobacter sp. 826659)
GC Type 00151: A0A009RHH7 (Acinetobacter sp. 809848)
GC Type 00251: A0A013TKW8 (Acinetobacter sp. 826659)
GC Type 00298: A0A011UZ32 (Ruminococcus albus SY3)
GC Type 00311: A0A011VNE2 (Aquamicrobium defluvii)
-1
0.6
0.5
1.1
1.1
1.4
0.9
0.9
0.6
0.6
0.8
0.5
0.6
1.2
0.5
0.5
0.5
1.2
0.8
5.1
0
2
4
6
8
10
Operon frequency (%)

2. Workflow of GCsnap1 Desktop

- GCsnap1 Desktop [2] is a Python-based tool that supports genomic context analyses.
- Starting from a user-provided list of target genes, GCsnap1 Desktop follows the workflow below to **collect** data from multiple public databases, **find** gene families, summarize and **annotate** the collected information, and generate interactive visual **outputs**.



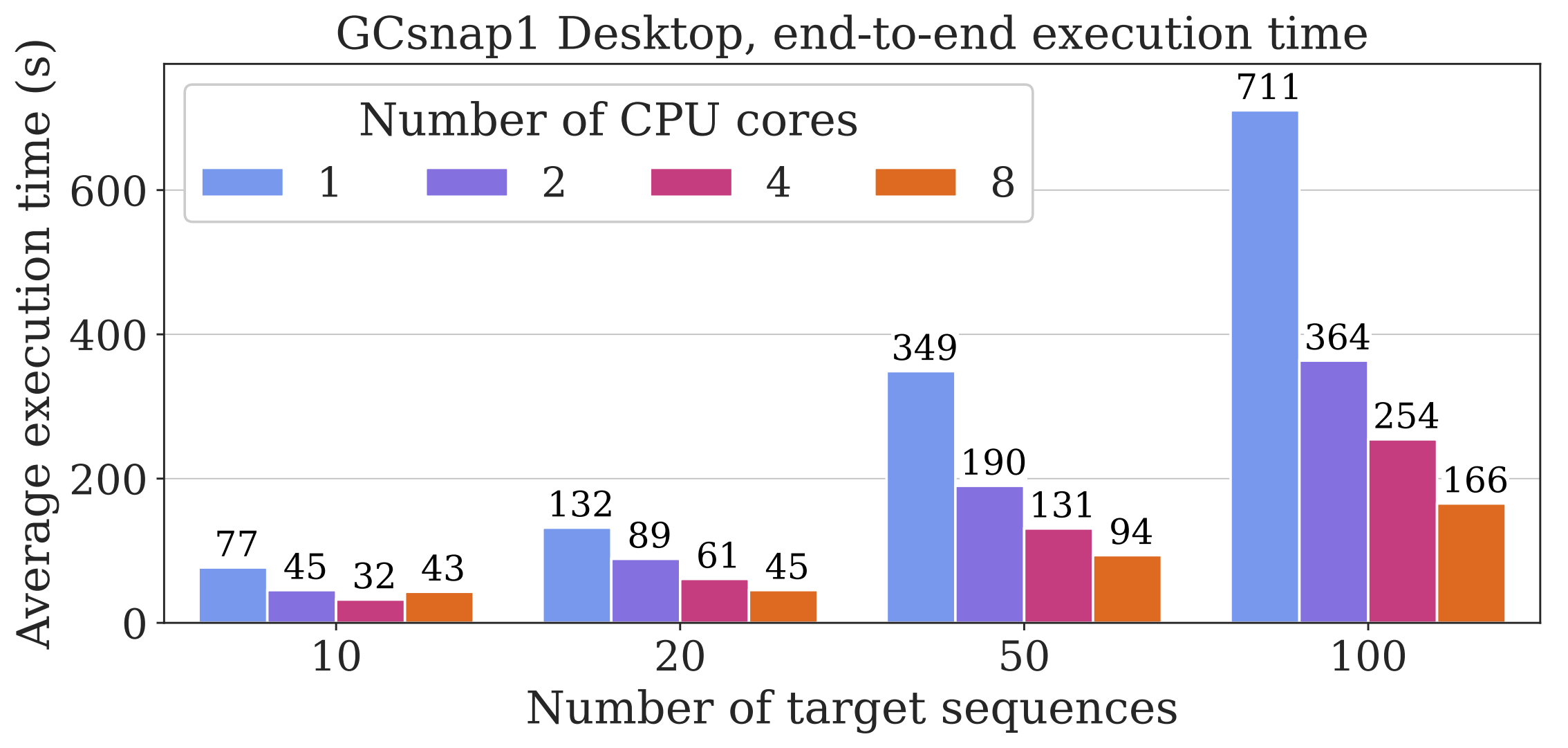
For each input identifier, it finds the *n* flanking genes

Then, runs all-againts-all BLASTp searches to find families

And for each member of a family searches for structural models, GO terms, TM segments, signal peptides, etc.

3. Limitations of GCsnap1

- While effective for small datasets, GCsnap1 Desktop does not scale well for more complex workloads.
- In a multicore setting, the average end-to-end genomic context analysis time for a single protein-coding gene is 1.66 seconds.
  ⇒ **Large-scale analysis is infeasible**

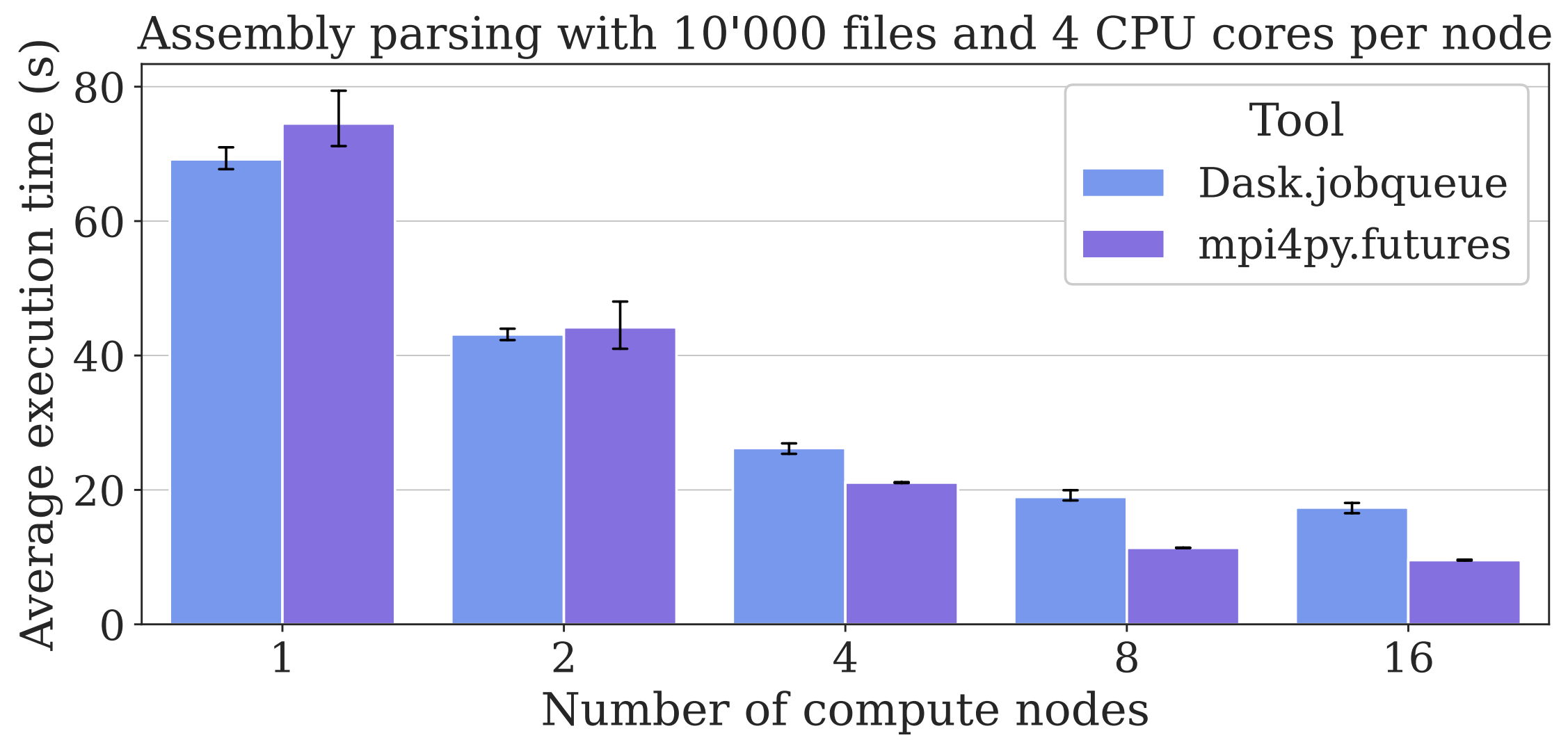


4. Approach

- We redesigned GCsnap1 Desktop to execute in distributed HPC environments.
- We considered Dask [3] and mpi4py [4] to enable distributed execution.
- We pre-downloaded the required data.

5. Distributed Execution

- We conducted preleminary experiments, to evaluate a suitable tool for distributed execution.
- Up to two computing nodes, *Dask.jobqueue* shows superior performance
- Beyond two computing nodes **mpi4py.futures** exhibits a lower average execution time.



6. Code Repository

- The modular code of GCsnap2 Cluster v1.0.0 is publicly available on GitHub.
- Scan the QR code to access the repository.

7. Evaluation of GCsnap2 Cluster

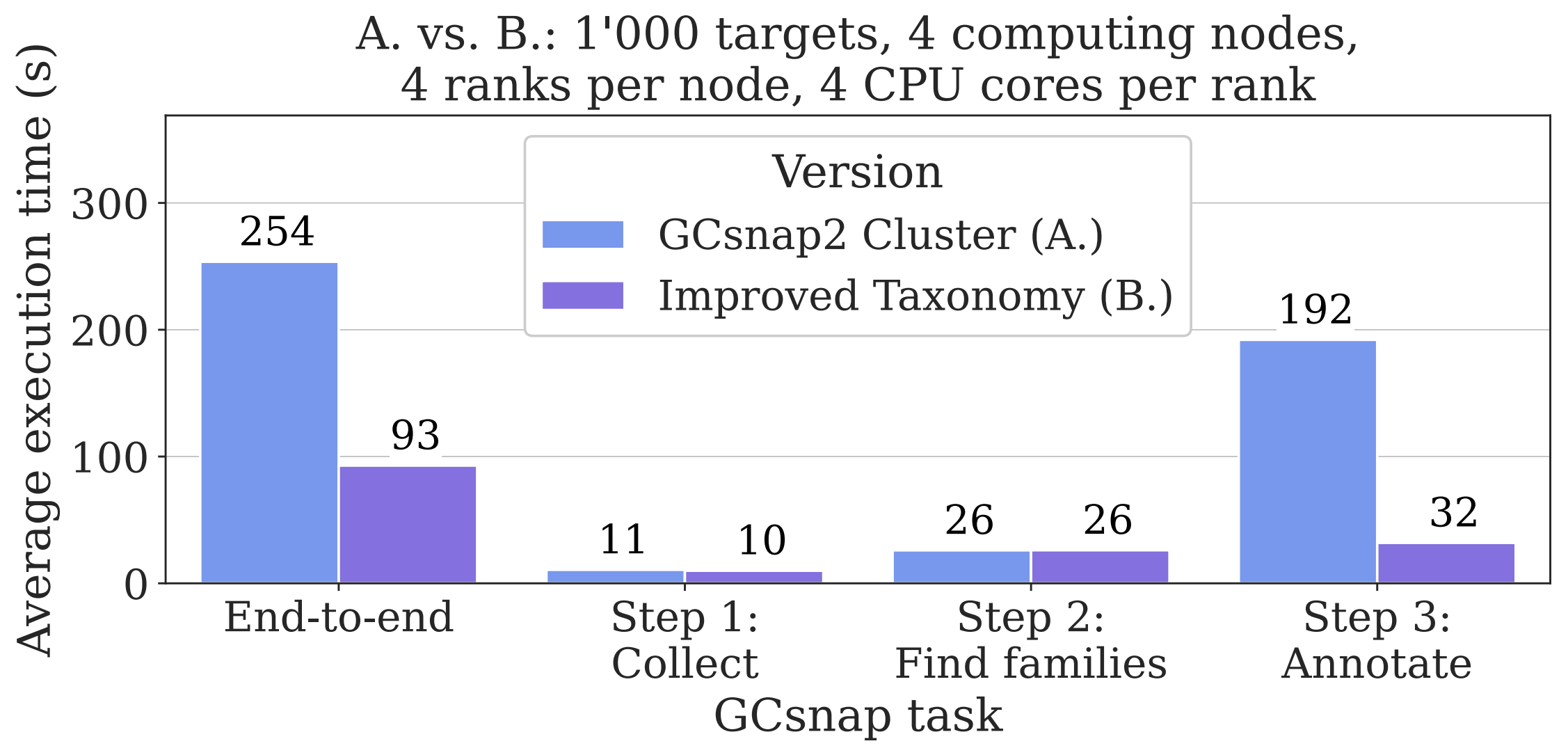
- We conducted three sets of experiments:
  A: GCsnap2 Cluster with mpi4py.futures
  B: Experiment A. + Improved taxonomy parsing
  C: Portability and Performance of Experiment B.

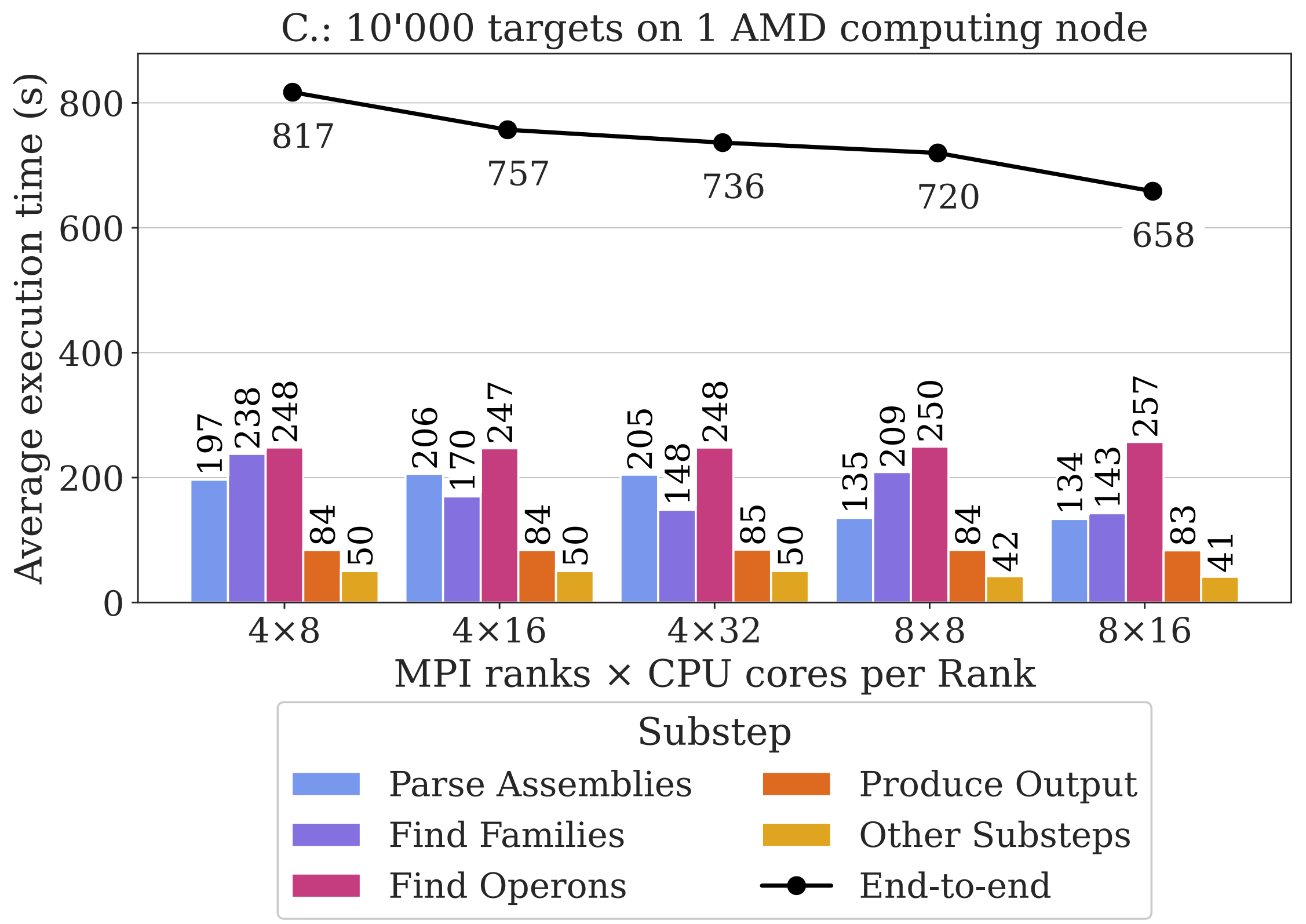
A Per-target execution time is 0.254 seconds.

B Code refinement of experiment A. reduced the execution time to 0.093 seconds per target.

C Average end-to-end execution time is ≃ 740 seconds for 10'000 sequences ⇒ 0.074 seconds per target, much smaller than 1.66 seconds per target with GCsnap1 Desktop

- Therefore, **GCsnap2 Cluster is 22× faster**.
- The sub-steps *Find Families* and *Find Operons* of the workflow remain unoptimized.





8. Conclusion and Future Work

- GCsnap2 Cluster is 22× faster then its predecessor.
- The design features a modular architecture supporting the development of custom workflows and the flexibility to execute in various computational environments.
- GCsnap2 Cluster enables bioinformatics analyses of hundreds of thousands of input genetic sequences in a matter of a few hours.
- Additional work is needed to optimize the less performing aspects of our implementation, notably the sub-steps *Find Families* and *Find Operons*.
- Future developments of GCsnap2 Cluster will focus on streamlining data update processes, maintaining accessibility, and its ease of use for life scientists.
- The full paper [5] includes a comprehensive description of the methodology, experimental setup, and extended results.

Acknowledgments

This work was partially supported by the University of Basel Research Fund for Excellent Junior Researchers (Grant number: U.570.0006 to J.P.) and Swiss National Science Foundation Weave project (Grant number: 310030L_220141 to T.S.).

References

[1] Konstantinos Mavromatis, Ken Chu, Natalia Ivanova, Sean D. Hooper, Victor M. Markowitz, and Nikos C. Kyrpides. Gene context analysis in the integrated microbial genomes (img) data management system. *PLoS ONE*, 4(11), November 2009. ISSN 1932-6203. doi: 10.1371/journal.pone.0007979.

[2] Joana Pereira. Gcsnap: Interactive snapshots for the comparison of protein-coding genomic contexts. *Journal of Molecular Biology*, 433(11), May 2021. ISSN 0022-2836. doi: 10.1016/j.jmb.2021.166943.

[3] Matthew Rocklin. Dask: Parallel computation with blocked algorithms and task scheduling. In *Proceedings of the 14th Python in Science Conference*, pages 126–132, June 2015. doi: 10.25080/Majora-7b98e3ed-013.

[4] Lisandro Dalcin and Yao-Lung L. Fang. mpi4py: Status update after 12 years of development. *Computing in Science & Engineering*, 23(4):47–54, 2021. doi: 10.1109/MCSE.2021.3083216.

[5] Reto Krummenacher, Michèle Leemann, Osman S. Simsek, Leila T. Alexander, Torsten Schwede, Florina M. Ciorba, and Joana Pereira. Scalable genomic context analysis with gcsnap2 on hpc clusters. *In Proceedings of the Platform for Advancing Scientific Computing (PASC 2025)*, 2025.