HPC-CLUST: Distributed hierarchical clustering for very large sets of nucleotide sequences

João F. Matias Rodrigues¹,* and Christian von Mering¹,*

¹Institute of Molecular Life Sciences and Swiss Institute of Bioinformatics, University of Zurich, Zurich, Switzerland.

ABSTRACT

Motivation: Nucleotide sequence data is being produced at an ever increasing rate. Clustering such sequences by similarity is often an essential first step in their analysis—intended to reduce redundancy, define gene families, or suggest taxonomic units. Exact clustering algorithms, such as hierarchical clustering, scale relatively poorly in terms of run time and memory usage, yet they are desirable because heuristic shortcuts taken during clustering might have unintended consequences in later analysis steps.

Results: Here we present HPC-CLUST, a highly optimized software pipeline that can cluster large numbers of pre-aligned DNA sequences by running on distributed computing hardware. It allocates both memory and computing resources efficiently, and can process more than a million sequences in a few hours on a small cluster.

Availability: Source code and binaries are freely available at http://meringlab.org/software/hpc-clust/; the pipeline is implemented in C++ and uses the MPI standard for distributed computing.

Contact: joao.rodrigues@imls.uzh.ch, mering@imls.uzh.ch

Supplementary Information: Supplementary data is available at Bioinformatics online.

1 INTRODUCTION

The time complexity of Hierarchical Clustering Algorithms (HCA) is quadratic \( O(N^2) \) or even worse \( O(N^2 \log N) \), depending on the selected cluster linkage method (Day and Edelsbrunner, 1984). However, HCAs have a number of advantages that make them attractive for applications in biology: (i) they are well-defined and should be reproducible across implementations, (ii) they require nothing but a pairwise distance matrix as input, and (iii) they are agglomerative, meaning that sets of clusters at arbitrary similarity thresholds can be extracted very quickly by post-processing, once a complete clustering run has been executed. Consequently, HCAs have been widely adopted in biology, in areas ranging from data mining to sequence analysis to evolutionary biology.

Apart from generic implementations, a number of hierarchical clustering implementations exist that focus on biological sequence data; and in average-linkage, the average of all pairwise similarities. The latter method is also known as the Unweighted Pair Group Method with Arithmetic Mean (UPGMA) and is often used in the construction of phylogenetic guide trees.

In the type of approach used by CD-HIT and UCLUST, each input sequence is considered sequentially, and is either added to only the first cluster; this effectively results in a reduction of the clustering threshold locally; and iii) different sequence input orders will result in different sets of clusters, due to different choices of the seed sequences. Point (i) also affects hierarchical clustering algorithms using single-linkage, and to a lesser extent average-linkage, but does not occur with complete-linkage.

Here we present a distributed implementation of an HCA that can handle very large numbers of sequences. It can compute single-, complete-, and average-linkage clusters in a single run and produces a merge-log from which clusters can subsequently be parsed at any threshold. In contrast to CD-HIT, UCLUST, and ESPRIT which all take unaligned sequence data as their input, HPC-CLUST (like MOTHUR) takes as input a set of pre-aligned sequences. This allows for flexibility in the choice of alignment algorithm; a future version of HPC-CLUST may include the alignment step as well. For further details on implementation and algorithms, see the supplementary material.

*to whom correspondence should be addressed

© The Author(s) 2013. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.
2 RESULTS

3.1 Clustering performance on a single computer

HPC-CLUST has been highly optimized for computation speed and memory efficiency. It is by far the fastest of the exact clustering implementations tested here, even when running on a single computer. Compared to MOTHUR, it produces identical or nearly identical clustering results (see supplementary material). Since CD-HIT and UCLUST use a very different approach to clustering, they are not directly comparable, and are included for reference only.

In HPC-CLUST, the largest fraction of computation time is spent calculating the pairwise sequence distances, the second largest in sorting the distances, and the final clustering step is the fastest. HPC-CLUST can make use of multithreaded execution on multiple nodes and practically achieves optimal parallelization in the distance calculation step. Additional benchmarks are shown and discussed in the supplementary material.

3.2 Distributed clustering performance

Clustering the full dataset (833,013 unique sequences) to 97% identity threshold required a total of 2 hours and 42 minutes on a compute cluster of 24 nodes with 8 cores each (192 total cores). Due to parallelization, the distance and sorting computation took only 57 minutes (wall clock time), corresponding to more than 10,000 minutes CPU time. The remaining 1 hour and 45 minutes (wall clock time) were spent collecting and clustering the distances. The combined total memory used for the distance matrix was 59.8GB, or 2.6GB per node. The node on which the merging step was performed used a maximum of 4.9GB of memory when doing single-, complete- and average-linkage clusterings in the same run.

4 CONCLUSION

Clustering is often among the first steps when dealing with raw sequence data and therefore needs to be as fast and as memory efficient as possible. The implementation of a distributed version of hierarchical clustering in HPC-CLUST makes it now possible to fully cluster a much larger number of sequences, essentially limited only by the number of available computing nodes.

ACKNOWLEDGEMENT

We thank Thomas S. B. Schmidt for his feedback and help in testing HPC-CLUST.

Funding: Supported by an ERC grant to CvM (Starting Grant ‘UMICIS/242870’).

REFERENCES


