PSCAN: Parallel, density-based clustering of protein sequences

Joshua Brulé
Department of Computer Science
University of Maryland, College Park
jbrule@cs.umd.edu

August 11, 2015

Abstract

Sequence clustering algorithms generally use greedy and other heuristic approaches to cluster DNA or protein sequences. PSCAN is a parallel implementation of DBSCAN* that provides exact density-based clustering and significant speedups over serial implementations, while running in O(n) memory. PSCAN clustering matches existing UniProt clusters exactly for a wide range of minimum cluster sizes and suggests a general strategy of favoring parallelization over memoization to efficiently produce high-quality clusters.

Introduction

Traditional clustering algorithms typically scale poorly for protein clustering due to the expense of calculating precise similarity scores and the volume of data. Existing techniques generally use a greedy approach that identifies a representative sequence for each cluster, and assigns a new sequence to a cluster if it is sufficiently similar to the representative or assigns the new sequence as the representative sequence for a new cluster.

PSCAN is designed to explore the tractability of using a parallel implementation of a higher-quality (and more time/space expensive) clustering algorithm on a representative sample of the sequences to be clustered. After producing high-quality clusters, it is possible to calculate the most representative (medoid) sequence for each cluster, and perform a simple nearest-neighbor clustering to classify the rest of the sequences.

Past Related Work

Sequence (protein and DNA) clustering involves clustering large sets of long strings, with similarity defined as the sequence alignment score or, equivalently, the edit distance [Sellers 74]. Due to the size of the data sets, and the computational expense, existing techniques generally involve greedy, approximate approaches. A popular algorithm is CD-HIT [Li and Godzik, 2006], where each consecutive sequence is compared to previously discovered cluster representatives and is added to a cluster if it is within a certain distance from the cluster representative. Otherwise, the sequence becomes the representative for a new cluster. CD-HIT also uses a “short word filtering” heuristic to avoid computing many of the expensive pairwise alignments. If the k-mer counts for two sequences significantly differ, it is unlikely that the sequences align well, and the algorithm will skip the exact alignment. UCLUST [Edgar, 2010] and DNACLUST [Ghodsi, 2011] are other approaches that follow the same basic greedy algorithm, with different heuristics to minimize the number of alignments that have to be computed.

Many standard clustering algorithms often require the members of the data set to be representable as points in R^n. For example, the definition of “means” in the k-means algorithm or mixture model approaches both

require real-valued points [Xu, 2005]. For this reason, many standard clustering algorithms are not applicable to sequence clustering, although the natural extension of k-means, k-medoids, has seen some use [Sperisen and Pagni, 2005].

The one standard clustering algorithm that is very popular in bioinformatics is hierarchical clustering, especially in the context of trying to create phylogenetic trees or perform multiple-sequence alignment. CLUSTAL [Higgins and Sharp, 1988], one of the most cited multiple-sequence alignment tools, uses hierarchical clustering with UPGMA (average) linkage. There are also known algorithms for parallel, hierarchical clustering algorithms [Olson, 1995] however, hierarchical clustering still requires $O(n^2)$ space which becomes prohibitive with large data sets.

Methods

PSCAN is a parallel implementation of the density-based clustering algorithm DBSCAN*, using a global sequence alignment score as the distance measure.

DBSCAN [Campello, 2013] clusters points based on density reachability - a point is directly density reachable if it is within *epsilon* distance of at least *min-pts* points. A point p is density reachable to q if there is some sequence of points p_1, p_2, ... p_n=q where each point in the sequence is directly density reachable to the next. A cluster is the collection of all points that are all mutually density reachable.[1]

DBSCAN requires O(n) region queries - for each point, the algorithm has to determine the collection of points within *epsilon* distance, for a total runtime of $O(n^2)$, executed serially. However, region queries are very amenable to parallelization. PSCAN uses fork/join parallelism to execute region queries, partitioning the collection of sequences into approximately *partition-size* sized subcollections, computing and filtering the distance scores in each partition and recursively combining the partitions into the final result.[2] In addition, PSCAN optionally memoizes distance computations, CASing (compare-and-set) the distance value into the cache.

Note that the distance function does not have to satisfy all of the metric axioms - it only has to be symmetric. PSCAN uses the Needleman-Wunsch algorithm with affine gap penalty to calculate a similarity score for each pair of protein sequences, with a higher score corresponding to more closely related proteins. Under this definition of distance, "within epsilon distance" refers to two proteins having a similarity score of at least *epsilon*.

As described, PSCAN has a large number of free parameters. Defaults based on previous work were used wherever possible: *min-pts* = 4 (recommended by the original DBSCAN authors) [Ester, 1996]; substitution matrix = BLOSUM62, gap existence penalty = 11, gap extension penalty = 1 (NCBI BLAST defaults).

---

[1] In the original DBSCAN algorithm, density-reachability is not necessarily symmetric. It is possible for there to be "border" points, which are density reachable from some "core" point, but do not have *min-pts* neighbours within *epsilon* distance. This can result in the same border point being assigned to different clusters, depending on the order in which points are visited by the algorithm. PSCAN implements the DBSCAN* variation of DBSCAN, which guarantees a deterministic classification by treating border points as noise points.

[2] PSCAN uses clojure.core.reducers for parallelism, built on Java's Fork/Join framework. Idle threads can "work-steal" tasks from other threads, minimizing the number and time of idle threads [Lea 2000]. Under theoretically ideal conditions, with a sufficient number of processors available, a region query can be executed in O(log(n)) time.

Test Data and Cluster Validation

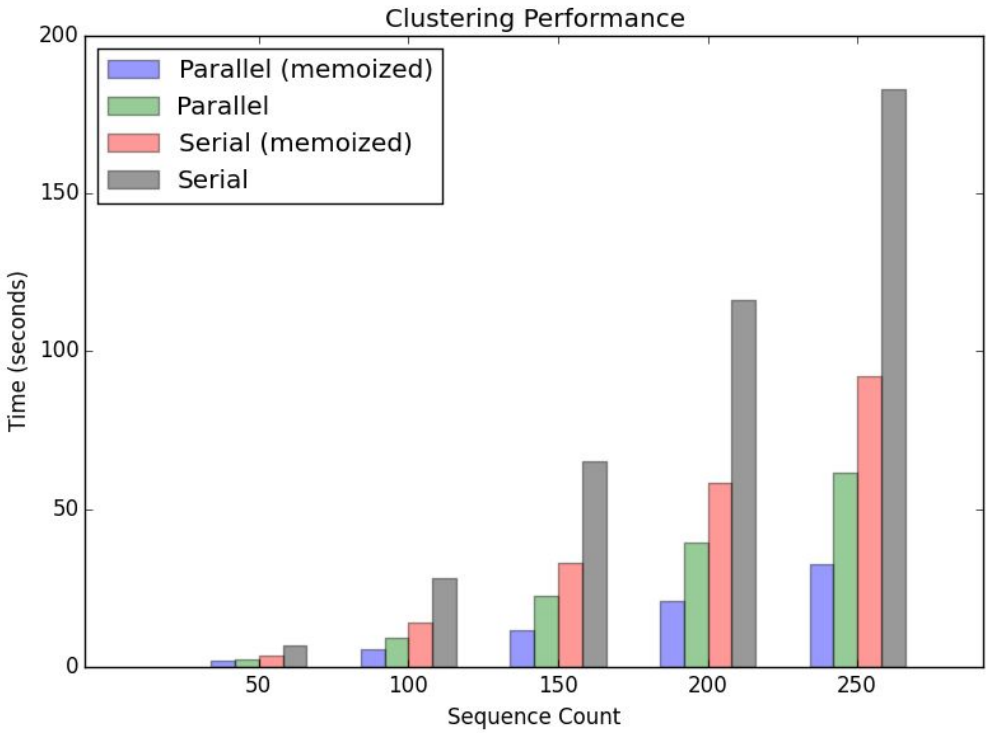
PSCAN was tested using data from the UniProt Reference Clusters [UniProt 2015], using 25 UniRef90 clusters of 10 (unique) sequences per cluster with sequence length between 247-257.[3]

An appropriate *epsilon* was determined empirically. PSCAN began incorrectly classifying UniRef90_I0C2H0 and UniRef90_P0A0Q0 as the same cluster at *epsilon* ≈ 200 or less. PSCAN began to fail to cluster some sequences at *epsilon* ≈ 1000 or greater. Between these values, PSCAN correctly assigned all 250 proteins to their original UniRef90 clusters. Lacking any additional insight as to the interpretation of epsilon, the rest of the clustering validation was run with *epsilon* = 250 (approximately the average sequence length).

Partition sizes of 4, 8, 16, 32 and 64 were all tested, with no significant effect on runtime.

Using the medoid of each cluster to perform nearest-neighbor clustering, the entirety of the UniRef90_Q0A457 cluster (3,175 proteins) was classified correctly.

Performance (4 core Intel i7-3610 @2.30 GHz; 8GB RAM)



Discussion

Memoization consistently provided an approximately 2x speedup over a non-memoized DBSCAN. However, this comes at the expense of $O(n^2)$ additional memory. Parallelization on a 4 core machine provided

[3] The median sequence lengths for Archaea and Bacterial proteins, respectively [Brocchieri and Karlin, 2005].

approximately 3x speedup over a serial DBSCAN. As written, PSCAN should generalize to an arbitrary number of process with *shared main memory*. However this has not been tested.

These results suggest a general strategy of favoring parallelism over memoization. Nearest-neighbor classifying runs in linear time and constant space and non-memoized DBSCAN runs in linear space. A scalable strategy to cluster a large number of proteins would be to run DBSCAN on a sample of the protein sequences that can fit into main memory, followed by nearest-neighbor clustering on the remaining dataset, using the medoids of the DBSCAN clusters as representatives.

However, exact global alignment may be far more precise (and disproportionately more expensive) than necessary, seeing as how DBSCAN produced the original UniProt clusters for a very wide range of *epsilon*. For this reason, the (theoretically) higher-quality clustering provided by DBSCAN seems unlikely to provide significant improvements in the cluster quality.

However, should a well-understood, biologically significant similarity score for proteins be available, PSCAN's higher precision may warrant the additional run time. Since, as the number of processors available increases, PSCAN gains a near-linear speedup, density-based protein clustering may be quite valuable in the future.

Source Code

The full source code for PSCAN is available (https://github.com/jtcb/pscan) under the Eclipse Public License.

References

[Edgar, 2010] Robert C Edgar. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, 2010.
http://bioinformatics.oxfordjournals.org/content/26/19/2460.full

[Li and Godzik, 2006] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 2006.
http://bioinformatics.oxfordjournals.org/content/22/13/1658.full

[Campello, 2013] Ricardo Campello et al. Density-Based Clustering Based on Hierarchical Density Estimates. *Advances in Knowledge Discovery and Data Mining,* 2013.
http://link.springer.com/chapter/10.1007/978-3-642-37456-2_14

[Ester, 1996] Martin Ester et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining,* 1996.
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.71.1980

[UniProt, 2015] The UniProt Consortium. UniProt: a hub for protein information. *Nucleic Acids Research*, 2015.
http://nar.oxfordjournals.org/content/43/D1/D204

[Lea, 2000] Doug Lea. A Java Fork/Join Framework. *Proceedings of the ACM 2000 conference on Java Grande,* 2000.
http://dl.acm.org/citation.cfm?id=337465

[Brocchieri and Karlin, 2005] Luciano Brocchieri and Samuel Karlin. Protein length in eukaryotic and prokaryotic proteomes. *Nucleic Acids Research*, 2005.
http://nar.oxfordjournals.org/content/33/10/3390.full

[Sellers, 1974] Peter Sellers. On the Theory and Computation of Evolutionary Distances. *SIAM Journal on Applied Mathematics,* 1974. http://www.jstor.org/stable/2099985

[Ghodsi, 2011] Mohammadreza Ghodsi et al. DNACLUST: accurate and efficient clustering of phylogenetic marker genes. *BMC Bioinformatics,* 2011. http://www.biomedcentral.com/1471-2105/12/271

[Xu and Wunsch, 2005] Rui Xu and Donald Wunsch II. Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, 2005. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1427769

[Sperisen and Pagni, 2005] Peter Sperisen and Marco Pagni. JACOP: A simple and robust method for the automated classification of protein sequences with modular architecture. *BMC Bioinformatics*, 2005.
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1208858/

[Higgins and Sharp, 1988] Desmond Higgins and Paul Sharp. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene*, 1988.
http://www.sciencedirect.com/science/article/pii/0378111988903307

[Olson, 1995] Clark Olson. Parallel algorithms for hierarchical clustering. *Parallel Computing,* 1995.
http://www.sciencedirect.com/science/article/pii/016781919500017I