Available online at www.sciencedirect.com
SCIENCE DIRECT®
Future Generation Computer Systems 21 (2005) 731–735
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

Gene transcript clustering: a comparison of parallel approaches

Todd E. Scheetz[a,*], Nishank Trivedi[a], Kevin T. Pedretti[b], Terry A. Braun[a], Thomas L. Casavant[a]

[a] *Departments of Electrical and Computer Engineering, Biomedical Engineering, and Ophthalmology and Visual Sciences, Center for Bioinformatics and Computational Biology, The University of Iowa, Iowa City, IW 52242, USA*
[b] *Sandia National Laboratories, Albuquerque, New Mexico 87123, USA*

Available online 5 November 2004

Abstract

One of the fundamental components of large-scale gene discovery projects is that of clustering of expressed sequence tags (ESTs) from complementary DNA (cDNA) clone libraries. Clustering is used to create non-redundant catalogs and indices of these sequences. In particular, clustering of ESTs is frequently used to estimate the number of genes derived from cDNA-based gene discovery efforts. This paper presents a novel parallel extension to an EST clustering program, `UIcluster4`, that incorporates alternative splicing information and a new parallelization strategy. The results are compared to other parallelized EST clustering systems in terms of overall processing time and in accuracy of the resulting clustering.
© 2004 Published by Elsevier B.V.

Keywords: Parallel Algorithms; Performance measurement; Genome Analysis; mRNA clustering; Bioinformatics

1. Introduction

The sequencing of cDNA libraries is the most common format for gene discovery in higher eukaryotes. The goal of such a project is to utilize the sequences derived from the cDNAs (ESTs; expressed sequence tags) to derive a non-redundant set. This set ideally represents an organism's entire complement of genes. The benefits of EST-based gene discovery include the ability to rapidly identify transcribed genes, the ability to identify exon-intron structure (when coupled with genomic sequence), and information on gene expression.

However, different genes are multiply expressed at different levels in cells. The presence of this "random" redundancy within the EST databases requires a programmatic method to calculate the complement of genes they represent. These methods (termed clustering) utilize sequence-based comparisons to determine sets of strongly similar sequences (clusters). The primary difficulty associated with EST-based gene-discovery projects is that ESTs are single-pass sequences, and as such they are relatively error prone (approximately 3% on average [1]).

The NCBI provides an EST sequence repository (dbEST) [2] as well as a curated and annotated gene index (UniGene) [3] for several species, utilizing the available mRNA and EST sequences to estimate

* Corresponding author.
E-mail address: tscheetz@eng.uiowa.edu (T.E. Scheetz).

0167-739X/$ – see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.future.2004.05.014

the gene complement. This paper describes and compares several programs that may be used to create non-redundant "UniGene" sets from EST data, and analyzes three different approaches for parallelization of this task.

2. Background

Clustering plays an important role in large scale gene-discovery projects. It not only saves time by identifying redundant sequences but also provides useful information regarding gene-discovery rate [4].

There are several varying clustering methods and tools in use today. However, the objective of all such methods is to effectively assess the similarity between sequences and place them into equivalence classes. Ideally, these classes correspond, one to one, onto distinct genes. A number of other criteria, not apparent in the primary RNA sequence data, are necessary for such a classification. However, a sequence-based classification is highly useful. One of the most widely used clustering tools is NCBI's Unigene clustering [5]. It uses global pairwise sequence comparison, and a stringent protocol for assigning closely related sequences to a common cluster. However, it does not support incremental clustering. Hence, each clustering "build" must begin from the same initial starting point. As the number of known ESTs in *Homo sapiens* currently (2004) stands at oevr 5 million, and requires upto one month of computation time, the benefit of performing incremental clustering becomes obvious. Also, an EST relating to two different clusters is often discarded, overlooking any possible alternative splice sites. A number of other institutes and labs have developed other serial methods for EST clustering [6–8]. The `UIcluster` family of solutions (both serial and parallel) has been evolving in a production environment at the University of Iowa since 1997. The key characteristics of `UIcluster` are incremental clustering, the maintenance of a representative element (primary) for each cluster, and a hashing scheme to quickly identify potentially meaningful cluster matches for each newly considered input sequence. The stringency of the clustering is a user-definable parameter, although performance is sensitive to this aspect. The parallelization of `UIcluster` based upon both cluster space [9], and input space, is the main focus of this paper.

The following is a brief description of the underlying approach of `UIcluster`. As each sequence is read from an input file, it is compared against all existing clusters. This comparison is performed only with the primary element of each cluster, where the primary is a single representative sequence of the entire cluster (usually the longest, and therefore most informative member). If the incoming sequence matches, or "hits" any cluster primary, and further satisfies specified similarity criteria, it is added to that cluster. Otherwise, the incoming sequence itself becomes the primary element of a new cluster.

3. Approach and implementation

The performance of EST clustering is measured both by time as well as by memory resource utilization characteristics. Although the use of a primary sequence for each cluster significantly reduces both these requirements, space remains a limiting factor for large data sets. Given the nature of the problem and the size of the data set, parallelization is an obvious choice for the implementation of this process. Computational and memory requirements can be distributed across several computers. This allows the software to scale to larger problem sizes. `UIcluster` currently implements two different approaches of parallelization, distributing across the cluster space and the input space. The message passing interface (MPI) [10] standard is used for inter-process communications, and distribution is done among multiple UNIX processes.

3.1. Parallelization on cluster space

In this scheme of parallelization, implemented in `UIcluster3`, each cluster is stored on exactly one compute node. The clusters are evenly distributed among all nodes. When a sequence is brought in, it is copied to all available nodes and is processed in parallel. Since each node has a different set of clusters, the incoming sequence is compared with the divided cluster space in parallel. Each node then communicates the best local match to all other nodes. The node with the best match adds the sequence to its cluster space. When no match is found, the sequence is designated as a new cluster and assigned to one compute node.

3.2. Parallelization on input space

The input space is parallelized by dividing the sequences into N non-overlapping groups, where N is the number of compute nodes available. Instead of distributing clusters over different compute nodes and processing each sequence at all nodes, in this scheme each node gets a fraction of the sequences. This is similar to running the sequential version of `UIcluster` in parallel on all nodes with an abridged dataset. Each node computes its own set of primaries. In the second stage, these primaries are compared amongst themselves and related clusters are merged. The efficiency of this scheme is heavily dependent on the redundancy within the dataset. If the data is highly redundant, the clusters on each node are more likely to be merged, involving more communication and added processing. Alternatively, lower redundancy yields more clusters.

4. Results

4.1. Description of experiment

To evaluate the performance of different clustering methods, several data sets from *Arabidopsis thaliana* and *Homo sapiens* were used. The methods compared include parallelizing on the cluster space (`UIcluster3`), parallelizing on the input space (`UIcluster4`), and the suffix-tree based method of `PaCE`. The `PaCE` clustering program [8] was included to analyze both parallel speedup and memory requirements. The publically available UniGene clusters from NCBI were used to asses the accuracy of the results. The system used in this comparison was a 16 node, dual processor cluster of 500 MHz Pentium III's, each equipped with one Gigabyte of memory. The human EST data set consisted of 41,197 sequences with an average length of 403 bp. The *Arabidopsis thaliana* EST data set contained 81,414 sequences with an average length of 411 bp. The latter data set was used for comparing the accuracy between the programs. The use of *A. thaliana* rather than human ESTs was important in reducing the effect of known genes on the purely sequence-based clustering. Although performance is critical in making the clustering results available, and the main focus of this paper, they must also provide an accurate reflection of the underlying mRNA transcripts from which the cDNAs were derived. Details of this analysis may be found in [11].

Table 1
Execution time performance comparison

Number of sequences	`PaCE`	`UIcluster3`	`UIcluster4`
5,000	10 min	37 s	1 min
10,000	28 min	2 min and 7 s	3 min and 28 s
20,000	1 h and 44 min	8 min and 42 s	12 min
30,000	Out of memory	20 min	25 min 12 s

4.2. Performance assessment

Both memory utilization and computation time were measured across these data sets. Table 1 presents the execution time for the same analyses. In this comparison, `UIcluster3` requires approximately one-tenth of the time of `PaCE`. As the number of input sequences increased, the relative difference in computation time between `UIcluster4` and `UIcluster3` decreased. With only 5000 sequences, `UIcluster4` required approximately 60% longer than `UIcluster3` on the same set of sequences. However, on the set of 30,000 sequences, that difference was only 26%. A similar reduction in computation time is observed between `PaCE` and `UIcluster3` with `PaCE` requiring approximately 16-fold more computation for 5000 sequences, but only 12-fold more in the data set of 20,000 sequences.

Peak memory utilization was assessed on a single node with 1 GB of memory, using a subset of the human EST data set. Fig. 1 shows the peak memory usage by the three clustering programs. Although the `PaCE` program has to use at least two nodes (one master and one slave node) only the memory utilization for the slave node was measured, because it performs sequence comparisons. Values were unavailable for `PaCE` with the 30,000 EST data set, as it exhausted the available memory. Note from this figure that the memory requirements of `UIcluster4` increase faster than `UIcluster3` as the number of input sequences grows. When the same computation is run in parallel with `UIcluster4`, the memory requirement per node is significantly reduced, as fewer clusters must be stored.

A final performance analysis was performed using the complete set of human EST and mRNA sequences from human UniGene build (#159). The paralleliza-

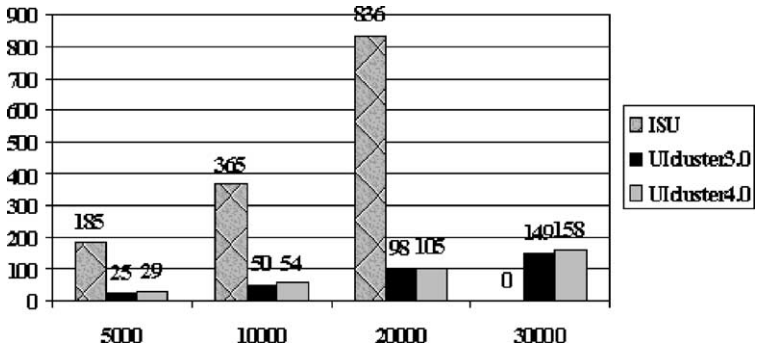


Fig. 1. Memory utilization.

tion on the input space method was used to predict the final number of clusters and the computation time required. This data set contained nearly 4.2 million sequences. The clustering, utilizing 12 nodes, requires an estimated computation time of 100 h. For this experiment, the data set was divided into 12 files each containing one twelfth of the ESTs. Thus each file contained roughly 400,000 EST sequences. All of the sequences longer than 1100 bp were put into a separate file. These thirteen sequence files were first clustered individually. The resulting cluster files were then clustered together to compute the complete set of clusters for the 4.2 million EST sequences. Unfortunately, the final clustering step required more memory than was available. Therefore, the computation time of that component was estimated.

5. Conclusions

An alternative scheme for parallel clustering using `UIcluster` has been described in this paper. The program is comparable in accuracy to other clustering programs, but requires less computation time. Depending upon the nature of the data set, either of the parallelization schemes may be used to optimize the memory or computation requirements.

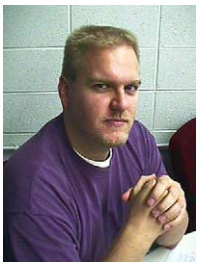
Acknowledgements

The authors would like to thank Dr. Volker Brendel from Iowa State University for providing us with the test set of 81,141 *A. thaliana* ESTs, Dr. Srinivas Aluru and Anantharaman Kalyanaraman from Iowa State University for their assistance in obtaining and using the `PaCE` clustering program, and Thomas Bair, Dylan Tack, Jason Grundstad, Jared Bischof, Brian O'Leary and Jesse Walters for their help and suggestions.

References

[1] L. Hillier, N. Clark, T. Dubuque, K. Elliston, M. Hawkins, M. Holman, M. Hultman, T. Kucaba, M. Le, G. Lennon, M. Marra, J. Parsons, L. Rifkin, T. Rohlfing, M. Soares, F. Tan, E. Trevaskis, R. Waterston, A. Williamson, P. Wohldmann, R. Wilson, Generation and analysis of 280,000 human expressed sequence tags, Genome Res. 6 (1996) 807–828.
[2] M.S. Boguski, T.M. Lowe, C.M. Tolstoshev, dbEST — database for 'expressed sequence tags', Nature Genet. 4 (1993) 332–333.
[3] G.D. Schuler, Pieces of the puzzle: expressed sequence tags and the catalog of human genes, J. Molec. Med. 75 (1997) 694–698.
[4] M.F. Bonaldo, G. Lennon, M.B. Soares, Normalization and subtraction: two approaches to facilitate gene discovery, Genome Res. 6 (1996) 791–806.
[5] http://www.ncbi.nlm.nih.gov/UniGene/build.shtml.
[6] M.D. Adams, A.R. Kerlavage, R.D. Flieshmann, R.A. Fuldner, C.J. Bult, N.H. Lee, E.F. Kirkness, K.G. Weinstock, J.D. Gocayne, O. White, Initial assessment of human gene diversity and expression patterns based upon 83 million nucleotides of cDNA sequence, Nature 377 (1995) 3–17.
[7] R.T. Miller, A.G. Christoffels, C. Gopalakrishnan, J.A. Burke, A.A. Ptitsyn, T.R. Broveak, W.A. Hide, A comprehensive approach to clustering of expressed human gene sequence: the sequence tag alignment and consensus knowledgebase, Genome Res. 9 (1999) 1143–1155.
[8] A. Kalyanaraman, S. Aluru, S. Kothari, Space and time efficient parallel algorithms and software for EST clustering, Int. Conf. Parallel Process. (2002) 331.
[9] N. Trivedi, J. Bischof, S. Davis, K. Pedretti, T.E. Scheetz, T.A. Braun, C.A. Roberts, N.L. Robinson, V.C. Sheffield, M.B. Soares, T.L. Casavant, Parallel creation of non-redundant gene

indices from partial mRNA transcipt, Future Generation Computer Syst. 18 (2002) 863–870.
[10] Message Passing Interface Form: MPI: a message-passing interface standard, University of Tennessee Technical Report, 1994, CS–94230.
[11] N. Trivedi, K.T. Pedretti, T.A. Braun, T.E. Scheetz, T.L. Casavant, Alternative parallelization strategies in EST clustering, in: V. Malyshkin (Ed.), Lecture Notes in Computer Science, vol. 237, Springer-Verlag, Heidelberg, 2003, 384–393.



Todd Edward Scheetz received the BS degree (1993) in Electrical Engineering, the MS degree (1995) in Electrical and Computer Engineering, and the PhD (2001) in Genetics from The University of Iowa. In 2003, he joined the Faculty of Ophthalmology and Visual Sciences at the University of Iowa, Iowa City, Iowa. He has co-authored several papers in the areas of high performance computer architecuture and parallel systems, and bioinformatics. His research interests include bioinformatics, disease gene identification, mapping, data-mining, parallel and distributed processing, and operating systems.

Kevin Thomas Pedretti received a BS (1999) in Electrical Engineering, and an MS (2001) in Electrical and Computer Engineering from The University of Iowa. He is currently a system software developer at Sandia National Laboratories in Albuquerque, NM. Current research interests include scalable operating systems, scalable computer architectures, and bioinformatics.

Terry Allen Braun received a PhD in Genetics (2001), and MS (1995) and BS (1993) degrees in Electrical and Computer Enigeneering from the University of Iowa. In 2002, he joined the Faculty of Biomedical Engineering and Ophthalmology and Visual Sciences at the University of Iowa. There, he is director of the Coordinated Laboratory for Computational Genomics. He has co-authored several papers in the area of high performance computer architecuture before research interests focussed on genetics, bioinformatics and computational biology. Current research interests include disease gene identification and prioritization using automated knowledge discovery and sequence analysis.

Thomas Lee Casavant is currently Professor of Electrical and Computer Engineering at the University of Iowa. He received the BS degree in Computer Science in 1982, the MS degree in Electrical and Computer Engineering in 1983, and the PhD degree in Electrical and Computer Engineering from the University of Iowa in 1986. In 1986, Dr. Casavant joined the the Faculty of the School of Electrical Engineering at Purdue University, West Lafayette, Indiana specializing in the design and analysis of parallel/distributed computing systems, environments, algorithms, and programs. From 1987 to 1989, he was Director of the PASM Parallel Processing Project, and the Purdue EE School's Parallel Processing Laboratory. He has developed graduate courses in advanced computer architecture, distributed computing, parallel processing, and computational biology. In 1989, he joined the faculty of the Iowa ECE Department and was promoted to Professor in 1999. There, he is director of both the Center for Bioinformatics and Computational Biology as well as the Parallel Processing Laboratory. Since 1996, he has led Computational Molecular Biology efforts in Gene Discovery, Mapping, and Disease Gene Identification/Isolation. From 1993 to 1994, he was a guest professor with the Department of Informatik at the ETH (Eidgenossisch Technische Hochschule — Swiss Federal Institute of Technology) in Zurich, Switzerland. In 2000, he was a guest researcher in the Biochimie et Biophysique des Systemes Integres Laboratory and the CEA/CNRS (Centre National de la Recherche Scientifique/Commissariat a l'Energie Atomique) in Grenoble, France. Dr. Casavant has authored or co-authored over 100 technical articles in Computer Science/Engineering and Computational Biology/Bioinformatics, edited two books on Parallel and Distributed Computing, served as editor for *IEEE Transactions on Parallel and Distributed Processing* and the *Journal of Parallel and Distributed Computing*, and has presented numerous tutorials worldwide.