Massively Parallel Expressed Sequence Tag Clustering

Scott J. Emrich
Electrical and
Computer Engineering
Iowa State University
semrich@iastate.edu

Ananth Kalyanaraman
Electrical Engineering
and Computer Science
Washington State University
ananth@eecs.wsu.edu

Srinivas Aluru
Electrical and
Computer Engineering
Iowa State University
aluru@iastate.edu

1 Corresponding author

Dr. Srinivas Aluru
Department of Electrical and Computer Engineering
Iowa State University
3227 Coover Hall
Ames, IA 50011
Ph: 515-294-3539
Fax: 515-294-8432
Email: aluru@iastate.edu

2 Topic area

Applications: Computational Biology

Massively Parallel Expressed Sequence Tag Clustering

Scott J. Emrich
Electrical and Computer Engineering
Iowa State University
semrich@iastate.edu

Ananth Kalyanaraman
Electrical Engineering and Computer Science
Washington State University
ananth@eecs.wsu.edu

Srinivas Aluru
Electrical and Computer Engineering
Iowa State University
aluru@iastate.edu

Abstract

Expressed Sequence Tag (EST) sequencing is a highly efficient technique that samples expressed genes required for most cellular functions. While this is a well-studied problem and many software tools have been developed, large-scale EST clustering has previously been pursued through incremental approaches, a pipeline of programs and manual efforts to achieve a modest degree of parallelism. Here, we present the first method that can directly cluster millions of ESTs on thousands of processors. This approach requires only linear space and uses rigorous alignment-based techniques to ensure biological accuracy. Further, we minimize computationally intensive alignments with single linkage clustering and develop a method to limit the formation of large spurious clusters. The computational scalability and biological validity of this approach is demonstrated by clustering mouse EST data, one of the two largest EST collections, on a 1,024 node BlueGene/L supercomputer.

1 Introduction

Unlike bacterial genome sequences that are highly compact archives of genetic information, the genomes of plants and animals are typically large and cluttered. In addition to *genes* that encode the information required to produce *proteins*, which perform most cellular functions, complex genomes often contain remnants of ancient evolutionary experiments that have generated families of related genes and now defunct versions called *psuedogenes*. Self-replicating sequences called *transposons* have further complicated matters by generating thousands of copies of themselves. Combined, these processes have led to a preponderance of so-called "junk" DNA in many genomes including our own; nearly 98% of the human genome does not encode proteins [1].

A fundamental goal of computational biology is to characterize the genomic contents of as many organisms as possible. Such information will help build the "Tree of Life" and provide fundamental insight into processes such as evolution. Current approaches either sample an entire genome or perform various filters that preferentially sample its gene-rich fraction. Obtaining a complete genome sequence is always the most comprehensive (but expensive) solution as it captures all protein-coding genes and additional functional sequences. The degree by which gene enrichment is important is directly correlated with the clutter of a large genome. For example, it is now commonplace to characterize a compact bacterial genome composed of a few million bases in a single afternoon, while billions of nucleotides of mammalian and many plant genomes still require significant resources. Given that only 2% of a mammalian genome is the most interesting, biologists have developed a compromise based on the observation that the genome acts as a template for messenger molecules that are *expressed* at specific times or under certain conditions. By capturing these intermediates from different tissues and developmental stages the protein-coding minority was preferentially sampled, greatly accelerating biological discovery [2].

In this paper, we address a grand challenge in computational genomics: analyzing millions of expressed sequences captured by a technique called Expressed Sequence Tag (EST) sequencing. There are currently almost 43 million EST sequences in the dbEST division of GenBank [3] led by the mammalian species of human (*Homo sapiens*; 7.9 million) and mouse (*Mus musculus*; 4.8 million). The primary *de novo* mechanism to detect ESTs derived from the same gene is the overlap between them. The quality of such an overlap is best detected using an alignment algorithm [4]. Unlike traditional genome sequencing, EST sequence data are nonuniformly sampled because genes are expressed at different levels at different times. This creates a quadratic number of genuinely overlapping pairs of ESTs, making large-scale EST analysis difficult.

Although many methodologies and programs have been developed over the past decade for analyzing ESTs [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15], most are only applicable for a few hundred thousand and do not scale well to larger problem sizes. At present, two approaches are often used: One technique is to take every pair of ESTs and detect if there is an overlap between them. Though this is computationally wasteful, this approach is trivially parallelizable and inherently incremental. This was especially useful several years ago when sequences used to arrive in small batches, allowing clusters to be built and refined over a period of time. A second method is to first break the problem into multiple independent subproblems by clustering using metrics such as approximate matches [5, 6] or single linkage clustering [9, 11]. Each cluster is then independently re-clustered well using any serial algorithm, easily providing parallelism across the initial clusters. The problem with this approach, however, is the size of the largest initial cluster tends be a significant fraction of the original problem size. This limits the overall utility of the preprocessing even if the initial steps are parallelized.

Previously, we developed a parallel EST clustering framework called PaCE [9]. This algorithm effectively handled a few hundred thousand ESTs, but when applied to larger collections it too suffered from the "one large cluster" problem. Here, we present a technique to significantly reduce the largest initial cluster size such that post-processing of initial clusters is both feasible and does not overwhelm the overall computational time. In addition, we have successfully applied the PaCE framework on an IBM BlueGene/L supercomputer. To demonstrate scalability to the largest-scale EST collections and test its biological validity, we clustered 3.7 million mouse ESTs obtained from Genbank.

Our paper is structured as follows. First, we define ESTs and draw comparisons to related work in Section 2. In Section 3 we summarize the key features of the PaCE framework. A novel approach to limit the size of the largest initial cluster is presented in Section 4. Section 5 contains results on mouse EST clustering that validate this method from a biological standpoint. In Section 6 we provide performance results and demonstrate scaling to over a thousand processors. In section 7, we present additional biological insight that can be drawn from massively parallel EST clustering and discuss the applicability of our framework for pyrosequencing-based EST sampling approaches [16]. Because these techniques are generating about 200,000 sequences in a 4 hr. experiment, they are creating a situation where the speed of analysis tools is falling behind the rate at which data can be generated. The contribution made in this paper resolves this problem.

2 Problem Description

EST sequencing is the most common technique used to selectively sample genes from large genomes. In a cell, information stored in a gene is copied into a messenger RNA molecules (mRNA). Biologists have taken advantage of this process to selectively isolate active genes from specific tissues or developmental stages. These are then placed into a bacterium called a *vector* that collectively comprise a *library*, in which each member contains a single messenger. After library construction, known sequences in the vector are used to sequence terminal ends of a DNA version of the original mRNA. These ends are the *Expressed Sequence Tags*.

Given that ESTs comprise a majority of the sequence data for many organisms, multiple research groups have developed software for EST analysis using clustering-based solutions. Some of the earliest clustering approaches were derived by adapting software developed for the related genome assembly problem [7, 8, 12], as both approaches rely on finding overlaps between sequences. Their chief disadvantage is that, given n ESTs the non-uniform nature of EST sampling can generate $\Theta(n^2)$ overlaps while random shotgun assembly is expected to generate only $O(n)$ overlaps. The $\Theta(n^2)$ time and memory requirements limit these approaches to a few hundred thousands ESTs.

To enable clustering of larger-scale EST data, several groups developed a two-stage clustering approach: First, the sequences are partitioned into an initial set of clusters such that ESTs from the same gene are never split across clusters but each cluster may contain ESTs from several genes. This breaks the problem into several independent subproblems, each of which is small enough to be refined through assembly. The initial clustering stage can be performed using a less rigorous metric. For example, the TGICL pipeline performs an all-vs.-all search using a customized version of megablast [11]. Each pairwise overlap can be computed by trivially partitioning the input file into p pieces, although because each overlap would be stored on disk and sorted prior to clustering implies a $\Theta(n^2)$ space complexity and limited scalability. Subsequent assembly is carried out using CAP3 [8]. The STACK algorithm [13] performs all-vs.-all pairwise comparisons using d^2-measure, a distance measure to assess sequence dissimilarities. Subsequence assembly is carried out by the Phrap assembler [7]. The PaCE software [9] performs single-linkage clustering using a rigorous alignment metric followed by CAP3 assembly. None of these programs, however, can be directly used if the largest EST cluster is on the order of hundreds of thousands of ESTs. Such large clusters are too large to post-process using assembly.

An alternative approach to EST clustering is to directly align the ESTs to the organism's genome. This method is currently utilized for human and mouse ESTs at Unigene [14], and other programs have been developed [15]. Obviously, these techniques are only applicable for organisms with known genomes.

3 The PaCE Framework

PaCE [9], which stands for "Parallel Clustering of ESTs", was developed to cluster large-scale EST data on distributed memory machines in the absence of a complete genome. Unlike other methods that require running multiple programs, e.g., manual partitioning of input files and running multiple instances of a serial program if parallelism is desired, PaCE is a direct parallel method. The framework uses a single linkage clustering strategy: when a pair of ESTs from different clusters exhibits a strong alignment, the clusters are combined into one. This allows sequences that do not directly overlap to be part of the same cluster if they are connected via a chain of overlaps. It has been applied to hundreds of thousands of ESTs [17].

Given an input of n ESTs, the algorithm initially places each sequence in a cluster of its own. Generated pairs are aligned only if they currently belong to different clusters. Successful alignments are used to merge clusters while failed ones are ignored. Note that a linear number of pairwise alignments would be enough to compute the clustering result because there can be no more than $n-1$ merges irrespective of the data. As these pairs cannot be predicted in advance, we increase the odds by generating pairs with longer shared exact matches first. This heuristic reduces the total number of alignment computations without affecting the clustering result. An on-demand pair generation algorithm completely eliminates the need for storing pairs making PaCE the only framework with worst-case linear space complexity; all other programs have worst-case quadratic space complexity. Thus, PaCE is inherently scalable to very large problem sizes and does not need slow intermediate disk storage like other methods. PaCE is in use by over 50 organizations.

4 Large-Scale EST Clustering

The primary contribution in this paper is a method for EST clustering that scales up to the largest currently available data. We demonstrate the effectiveness of this approach on 3.78 million mouse ESTs. To our knowledge, this is the first successful attempt to directly cluster such large scale EST data.

4.1 The Largest Cluster Problem

Because a single successful alignment between a pair of sequences is sufficient to combine them under single linkage clustering, any spurious alignments will a compounding effect and lead to the formation a large cluster. Our experimental studies show that no such clusters are formed for small-scale data but this becomes a pressing issue for large-scale data. For example, clustering 168,200 *Arabidopsis* ESTs using PaCE yielded a largest cluster size of only 1008 sequences (0.6% of the total). On the other hand, the largest after clustering 3.78 million mouse ESTs was composed of 807,671 ESTs (21.37% of the total)!

In principle, post-processing the initial clusters with an assembly program should break the largest (and other) clusters into appropriate subclusters. In practice, there are two potential drawbacks to this approach: As assembly programs require $\Theta(m^2)$ memory for processing a cluster of size m, it may not be feasible to post-process the largest cluster if $m = \Theta(n)$. Even if it were, the run-time taken by the serial assembler on the largest cluster would dominate the total time to solution. For instance, when TGICL was used to cluster a 1.7 million EST collection, the clustering took an hour on 20 processors [11]. Subsequent CAP3 assembly, however, took 24 hours. This completely negated any advantages of using even this modest degree of parallelism.

In general, if post-processing the largest EST cluster takes one tenth of the parallel work required for the entire data, then using thousands of processors yields no more benefit than using just 10 processors, if we consider the overall time to solution. This creates an interesting conundrum – While hundreds of processors can be effectively utilized for medium-scale EST data, only few processors can be effectively utilized for very large data no matter how well the initial clustering approach scales on parallel machines.

4.2 Our Solution

Single linkage clustering is required to achieve linear space and to significantly reduce the number of alignments performed. Once single linkage clustering is chosen, the data determine the size of the largest cluster. To overcome large clusters, let us examine their nature. Large clusters are often formed by merging relatively big clusters because of spurious alignments. Such spurious alignments may be caused by a chimera, or a repeat, or a region common to members of a gene family that has not sufficiently diverged in sequence similarity. In many of these cases, there will be one bridging sequence that brings clusters together and no

additional alignment evidence. This is illustrated in Figure 1(a). It is clear that finding such weak links and breaking them into smaller clusters, or not allowing cluster formation in the first place should fix this problem. However, any attempt to carry out and store alignments will substantially increase the memory requirement to $\Theta(n^2)$ in the worst case.

To overcome this problem, we devised the following algorithm:

1. Choose a small constant $c \geq 2$. Break each input EST into c consecutive and non-overlapping pieces of equal size and tag each piece to recognize its source EST. In practice, an EST is broken only it satisfies a minimum length, such as 200bp.

2. Run the PaCE algorithm on the resulting input of at most cn sequences. The total memory required by PaCE is linear with respect to the total input measured in nucleotides. Because breaking ESTs does not increase the number of nucleotides, the additional memory required is negligible. The clustering result, however, is no longer a partition of the original EST collection because each EST may belong to at most c clusters.

3. "Link" every pair of clusters that contain fragments containing the same EST identifier. As the number of fragments per original EST is at most c, the total number of links is bounded by $\Theta(c^2 n)$. Thus, this can be computed simultaneously in $\Theta(c^2 n)$ time.

4. In the last step, we merge a pair of clusters if the number of links between them is at least k, for some constant $k \geq 2$. One could also choose more sophisticated criteria, such as making k a function of the size of the clusters. By preserving the source EST information, we are essentially achieving multi-linkage clustering by requiring at least k links between disjoint clusters in the new result.

This method is illustrated for the case of $c = 2$ and $k = 2$ in Figure 1. The parameters c and k should be chosen appropriately. In general, increasing the value of c or k will reduce the size of the largest cluster. However, arbitrarily large values cannot be chosen as these would result in rejecting genuine alignments. It is also important to reevaluate the alignment parameters. For instance, when $c = 2$, an originally acceptable alignment may now be split across the two fragments. Thus, the parameters must be modified accordingly to guarantee all information in the original collection of ESTs will be preserved.

This algorithm works for the following reason. By reducing the number of "bridges" available to a single linkage clustering algorithm we are effectively reducing the probability that a spurious link would lead to the formation of a large cluster because at least k sequences must now be involved. In addition, this strategy removes many biological artifacts, which are sequences that are incorrect or have poor quality. Although it is possible to miss valid alignments because of fragmentation, remember that ESTs are generated from ends of messenger molecules (Section 2). As such, we expect many of these will align as shown in Figure 1 and this problem will be minimal.

We tested this algorithm on the largest mouse EST cluster composed of 807,000 ESTs. By constructing a new dataset where $c = 2$ (1.6 million total), we were able to break the large cluster into 3,420 clusters, which represented 780,718 of the original sequences (97%), in less than 2 hours on 512 processors. Significantly, the largest cluster size was 50% of the original without losing any substantial information. Using a value of $c = 3$ further reduced the largest cluster to 216,026 sequences. For the mouse dataset the value of k did not significantly impact the size of the largest cluster; however, our experimental data support increasing c incrementally. For example, using $c = 3$ on the largest cluster formed when $c = 2$ was able to reduce the largest cluster by an additional factor of two. Because of the quadratic behavior of assembly, each reduction in the largest cluster size by a factor of r reduces the time required by a factor of r^2.

5 Validation

Current approaches to validate EST clusters rely heavily on biological experts. The PaCE software tool has been successfully used to process primarily plant EST datasets including Arabidopsis (N=168,200; [17]) and cotton (N=185,000; [18]). Both studies reported the ability to accurately partition plant ESTs at the cost of putative false negative overlaps that lead to more than one cluster per gene. Alternative genome-based methods, on the other hand, use a known sequence to anchor ESTs to fixed locations. This eliminates noise and ambiguous ESTs that confound single-linkage clustering because these typically do not align to a single location in the genome. As such, genome-based solutions can serve as the "gold standard" for *de novo* clustering result validation. Therefore, we reclustered sequences present in the mouse UniGene collection provided by the National Center for Biotechnology (NCBI) using the complete mouse genome.

Our run was able to obtain homogeneous clusters, where all members correspond to a single gene, for 50,661 out of 60,862 (83%) clusters with more than one

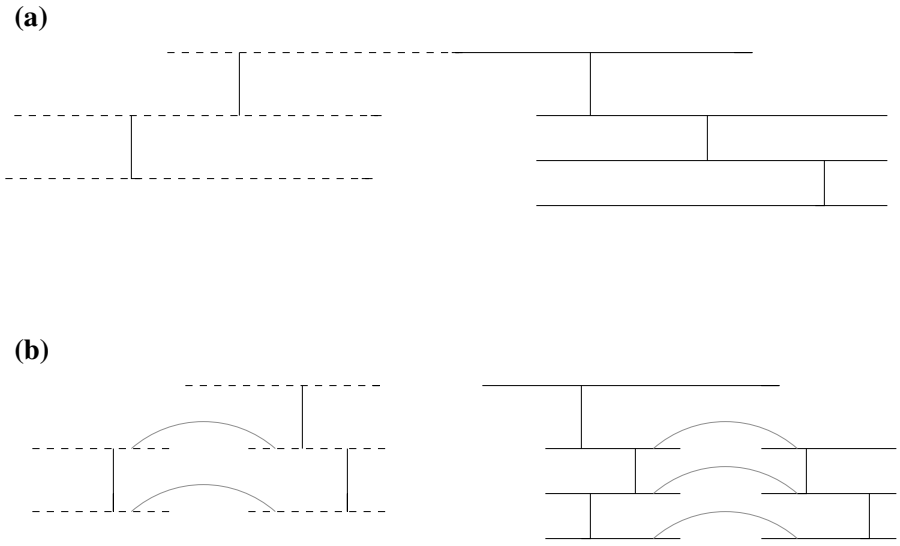

Figure 1: Illustration of our methodology to avoid formation of large clusters. In (a), two clusters are brought together by a bridging sequence which could be a chimera or contain a repeat or from parts of a gene family without significant divergence. Figure (b) shows the effect of breaking each EST into two pieces prior to clustering. The bridging sequence belongs to both the clusters. However, it is the only link between two clusters and is not sufficient to merge the clusters.

member. In other words, many PaCE clusters achieve high accuracy relative to the benchmark without any post-processing. Most importantly, all of these clusters could be refined by assembly. This overcomes the primary limitation encountered by previous attempts to perform large-scale EST clustering.

The above analysis suggests false positive merges are manageable, but does not address the extent of false negatives, where ESTs belonging to the same gene are placed in two or more clusters. In total, 422,598 clusters comprise the UniGene build downloaded in late March 2006. As previously alluded, any correct partition can be achieved by locating as many correct links as clusters. Based on this observation, we fixed the UniGene partition of the 3.78 million ESTs as truth and define the distance between results as the number of additional merges and splits required to transform our resulting PaCE partition to the benchmark partition. Conceptually, each merge in our transformation corresponds to a false negative that arises for many reasons such as poor-quality ESTs. Similarly, a split corresponds to a false positive that arises for reasons such as two unrelated ESTs being members of a large gene family. Doing this comparison indicated that most false negatives correspond to singletons that were separated likely due to poor quality in overlapping regions. Excluding these singletons, we determined that 3,213,878 of the links determined by PaCE were correct with 45,05826,125 false negatives; the number of false positives were only 26,125. Taken together, these results suggest that the vast majority of the links found by PaCE were valid with minimal false positives and negatives.

6 Performance Results

There are two main phases in PaCE: (i) "Tree construction phase", in which a distributed representation of a generalized suffix tree is constructed in parallel, and (ii) "Clustering phase", in which clustering is performed from overlaps detected and computed in parallel. The tree construction phase is expected to scale linearly with input size. For the clustering phase, the work is proportional to the number of pairs aligned.

We evaluated PaCE on mouse EST data composed of 3,783,854 ESTs using 1,024 BlueGene/L nodes. Table 1 shows the phasewise run-times as a function of both the input size and processors used. These tables show the range of processors on each input until which the run-time scales linearly, and beyond which the problem size becomes too small for the processor size. Significantly, the results show that as many as 512 processors can be used efficiently for even an input containing as few as 100,000 ESTs.

The observed increase of clustering phase run-time with input size in Table 1 conforms with expected asymptotic quadratic behavior. In addition, the re-

Number of ESTs	Tree construction phase run-time: Number of processors						Clustering phase run-time: Number of processors					
	32	64	128	256	512	1024	32	64	128	256	512	1024
100,000	44	25	11	6	3	2	36	18	9	5	3	3
250,000	110	56	29	15	8	6	115	58	22	11	6	5
500,000		123	63	32	16	12		152	83	42	18	13
1,000,000			135	69	38	22			146	69	40	24
2,000,000				174	91	51				247	181	102
3,783,854					248	158					950	416

Table 1: Phasewise run-times (in minutes) of PaCE as a function of input and processor sizes.

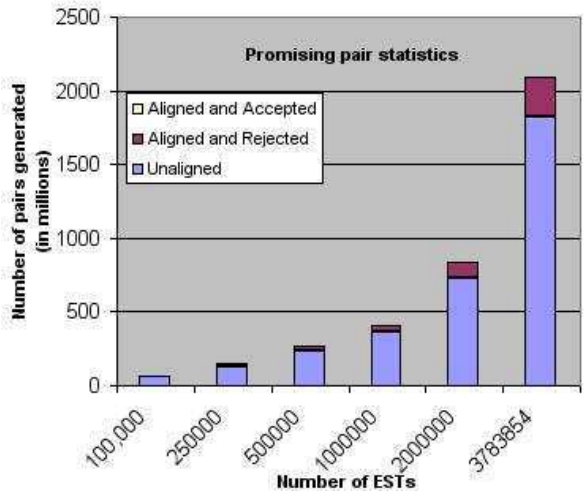

Figure 2: Pairs generated on mouse EST data using a maximal common substring cutoff of 30 *bp*. The substantial fraction of unaligned pairs are a result of the greedy heuristic used by the PaCE algorithm.

sults indicate that the run-time for the clustering phase dominates the total run-time for larger inputs. Figure 2 shows the number of promising pairs generated by PaCE based using a minimum cutoff maximal match length of 30 *bp*, as a function of the input number of ESTs. While the growth is input dependent, the figure shows an almost quadratic increase as expected in the worst case. Figure 2 also shows that alignments are computed for only ∼10-12% of the generated pairs. Computing alignments over the remainder is deemed unnecessary, demonstrating the significant run-time savings achieved without loss of quality.

We evaluated the effectiveness of the master-worker implementation of the PaCE algorithm as follows: Figure 3a shows the average run-time (as a percentage of the total run-time) spent by a worker processor waiting for the master processor without performing any computation. As expected, this idle time decreases with increases in input size for a fixed number of processors; Figure 3a shows that this idle time ranges from 7% to 22% on the data tested. More importantly, the plot also shows that the processor size can be quadrupled upon doubling the input size, without increasing the idle time of workers. We also evaluated idle time on the master processor. Figure 3b shows that the master processor is available for at least 80% of the clustering phase's run-time even for as few as 100,000 ESTs on 1,024 processors. We conclude that the master processor is not a bottleneck during the conditions tested.

7 Biological Insights

During the initial stages of the human genome project, EST sequences were used to estimate the number of genes in mammalian genomes [2]. Surprisingly, many of these predictions were off by tens of thousands of genes because single genes produce more than one message (and therefore EST) by selectively mixing and matching expressed portions in a biological process called *alternative splcing*. The frequency of these events has been estimated by Kan *et al.* [19] and others by mapping individual ESTs to genomic sequences. Alternative *de novo* approaches rely on serial assembly programs to produce related transcripts. We believe that massively parallel clustering, as presented in this paper, can facilitate both types of prediction by reducing a large problem into many disjoint subproblems.

Many cells contain machinery that degrades double-stranded mRNA formed when the message (sense) is bound to its reverse complement (antisense). It has been shown that at least 20% of mammalian transcripts form sense-antisense pairs [20]. Because antisense-based gene regulation appears to be evolutionarily conserved, this biological process should have important functions in many organisms. Although EST clustering can not resolve the different classes of antisense interactions, candidates can be detected using post-processing of single EST clusters.

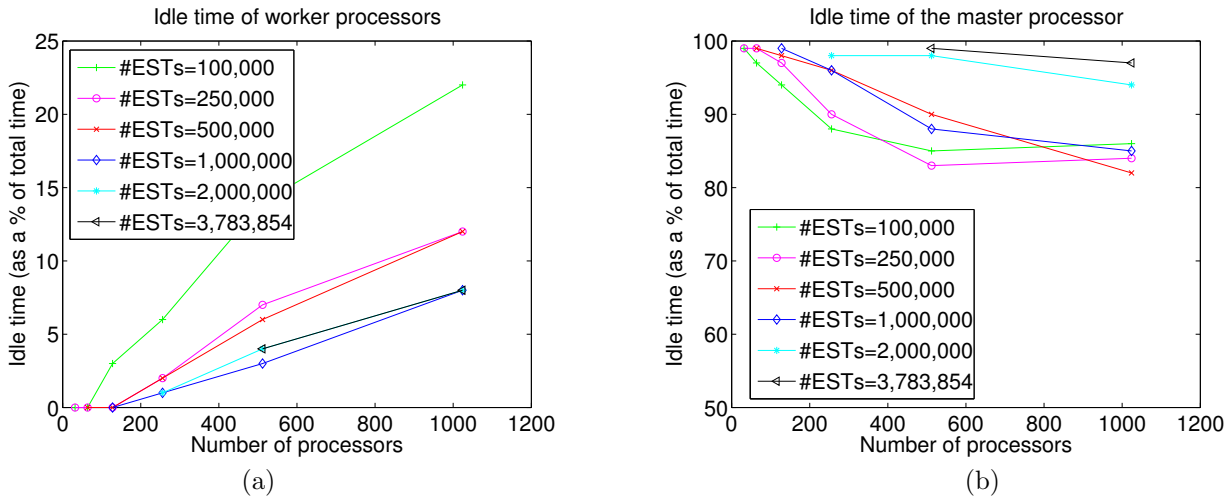

Figure 3: (a) Average idle time for each worker processor as a percentage of the total run-time. (b) Availability of the master processor, measured as the percentage of the clustering phase run-time that the master is idle.

8 Conclusions

In this paper, we present the first method that can cluster the largest known EST collections on massively parallel distributed memory machines. We demonstrate the applicability of our framework on the 3.78 million mouse EST collection represented by Unigene [14]. Validation confirms that our method produces biologically meaningful results that can facilitate important analyses such as detecting alternative splicing and antisense transcripts. Developing EST clustering programs that could handle the largest data available has been a problem frustrating many researchers including us. The contribution made in this paper is a solution to this long standing problem. While we expect that traditional EST sequencing will soon be replaced by the use of 454 sequencing technology, the underlying problem remains the same and our method is applicable. More importantly, the high throughput nature of these sequencers make the development of large-scale clustering methods even more important in the future.

Acknowledgments

The research is funded in part by NSF under CNS-0521568. A. Kalyanaraman was supported by an IBM Ph.D. Fellowship.

References

[1] I. H. G. S. Consortium, "Finishing the euchromatic sequence of the human genome," *Nature*, vol. 431, pp. 931–945, 2004.

[2] G. Schuler, "Pieces of the puzzle: expressed sequence tags and the catalog of human genes," *Journal of Molecular Medicine*, vol. 75, pp. 694–698, 1997.

[3] M. Boguski, T. Lowe, and C. Tolstoshev, "dbEST – database for "expressed sequence tags"," *Nature Genetics*, vol. 4, pp. 332–333, 1993.

[4] D. Gusfield, *Algorithms on strings, trees and sequences Computer Science and Computational Biology*. Cambridge, London: Cambridge University Press, 1997.

[5] J. Carpenter, A. Christoffels, Y. Weinbach, and W. Hide, "Assessment of the parallelization approach of d2_cluster for high performance sequence clustering," *Journal of Computational Chemistry*, vol. 23, pp. 755–757, 2002.

[6] A. Christoffels, A. Gelder, G. Greyling, R. Miller, T. Hide, and W. Hide, "STACK Sequence Tag Alignment and Consensus Knowledgebase," *Nucleic Acids Research*, vol. 29, pp. 234–238, 2001.

[7] P. Green, "Phrap - the assembler," *http//www.phrap.org*, 1994, (Date accessed 29 Apr 2007).

[8] X. Huang and A. Madan, "CAP3: A DNA sequence assembly program," *Genome Research*, vol. 9, pp. 868–877, 1999.

[9] A. Kalyanaraman, S. Aluru, V. Brendel, and S. Kothari, "Space and time efficient parallel algorithms and software for EST clustering," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 12, pp. 1209–1221, 2003.

[10] K. Malde, E. Coward, and I. Joassen, "Fast sequence clustering using a suffix array algorithm," *Bioinformatics*, vol. 19, pp. 1221–1226, 2003.

[11] G. Pertea, X. Huang, F. Liang, V. Antonescu, R. Sultana, S. Karamycheva, Y. Lee, J. White, F. Cheung, B. Parvizi, J. Tsai, and J. Quackenbush, "TIGR Gene Indices clustering tools (TGICL:) a software system for fast clustering of large EST datasets," *Bioinformatics*, vol. 19, pp. 651–652, 2003.

[12] G. Sutton, O. White, M. Adams, and A. Kerlavage, "TIGR assembler: A new tool for assembling large shotgun sequencing projects," *Genome Science and Technology*, vol. 1, pp. 9–19, 1995.

[13] N. Trivedi, J. Bischof, S. Davis, K. Pedretti, T. Scheetz, T. Braun, C. Roberts, N. Robertson, V. Sheffield, M. Soares, and T. Casavant, "Parallel creation of non-redundant gene indices from partial mRNA transcripts," *Future Generation Computer Systems*, vol. 18, pp. 863–870, 2002.

[14] D. Wheeler, D. Church, R. Edgar, S. Federhen, W. Helmberg, T. Madden, J. Pontius, G. Schuler, L. Schrimi, E. Sequeira, T. Suzek, T. Tatusova, and L. Wagner, "Database resources of the National Center for Biotechnology Information: update," *Nucleic Acids Research*, vol. 32, pp. D35–D40, 2004.

[15] X. Wu, W. Lee, and C. Tseng, "ESTmapper: Efficently clustering EST sequences using genome maps," in *Proceedings of the Parallel and Distributed Processing Symposium:HICOMB*, 2005.

[16] S. Emrich, W. Barbazuk, L. Li, and P. Schnable, "Gene discovery and annotation using LCM-454 transcriptome sequencing," *Genome Research*, vol. 17, pp. 69–73, 2007.

[17] A. Kalyanaraman, S. Aluru, S. Kothari, and V. Brendel, "Efficient clustering of large EST data sets on parallel computers," *Nucleic Acids Research*, vol. 31, pp. 2963–2974, 2003.

[18] J. Udall, J. Swanson, K. Haller, and *et al.*, "A global assembly of cotton ESTs," *Genome Research*, vol. 16, pp. 441–660, 2007.

[19] Z. Kan, E. Rouchka, W. Gish, and D. States, "Gene structure prediction and alternative splicing analysis using genomically aligned ESTs," *Genome Research*, vol. 11, pp. 889–900, 2001.

[20] R. G. E. R. G. *et al.*, "Antisense transcription in the mammalian transcriptome," *Proc. National Academy of Sciences*, vol. 309, pp. 1564–1566, 2005.