# Evaluation of Expressed Sequence Tag Clustering

Katryna Cisek
Department of Computer Science
University of Northern Iowa
Cedar Falls, IA 50614-0507
cisekk@uni.edu

## Abstract

Bioinformatics — the application of computer technology to the management of biological information — is essential to deciphering the genetic code of life. Novel approaches to genome sequencing, such as microarray technology, high-performance supercomputing and computational simulations in high-throughput DNA analysis have led to an explosion of genomic data available. Accurate genomic assembly of this data using computational methods has been one of the biggest challenges in bioinformatics. The reason for this is that different algorithms and different parameter settings in the software give inconsistent results for the same dataset. The objective of the following paper is an analysis of the performance of the PaCE (Parallel Clustering of ESTs) algorithm, implemented as genomic assembly software via Expressed Sequence Tag (EST) clustering. For comparison purposes, the PaCE software performance is compared with different algorithms implemented in Glimmer2.12 and GeneMark genomic assemblers.

# 1 Introduction

Experimental techniques used to identify gene positions in the genome are the most accurate methods of defining gene structures. These experimental techniques usually employ microarray technology and are used as validation tools for most computational predictions. However, due to the high cost, manpower, and duration of experimental genome annotation, emphasis is being placed on finding more accurate algorithms for computational analysis of genome data.

Expressed Sequence Tags, or ESTs, are complementary DNA (cDNA) sequences, usually 200 to 500 nucleotides in length that represent the expressed portions of genes. Therefore, ESTs can be used in gene identification, expression profiling and polymorphism analysis [7]. By computationally clustering sequenced ESTs, sets of unique genes, characterizing the transcription products of an organism, can be identified.

The EST clustering problem, partitioning the ESTs into clusters, such that ESTs from the same gene only are grouped together in a distinct cluster, is complicated by several factors. These include: poor average sequence quality, incomplete sampling, polymorphism, alternative transcript isoforms, cloning artifacts, as well as the fact that ESTs are sequenced from multiple cDNAs. [7]

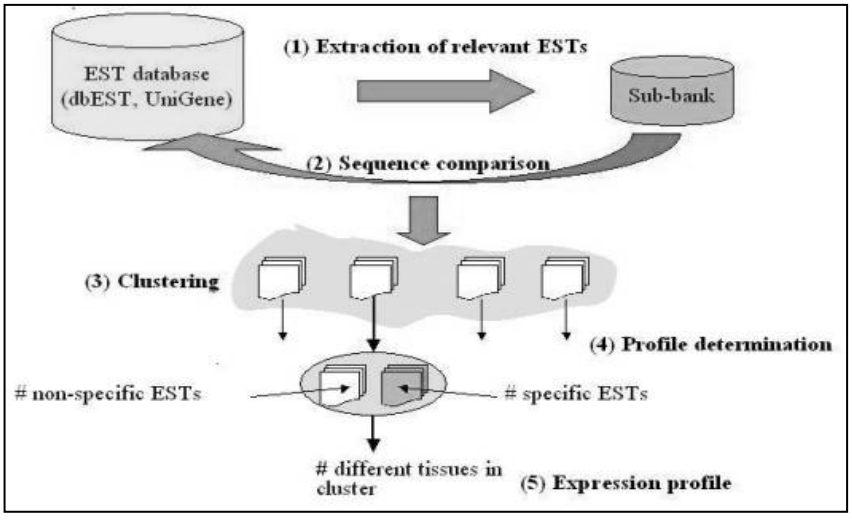


Figure 1: EST Evaluation and Profiling [6]

There are several clustering algorithms with three implemented in PaCE, Phrap, CAP3, and TIGR assembler software [7]. However, different clustering algorithms and different parameter settings in the software give inconsistent results for the same EST dataset. PlantGDB, an EST database, uses the PaCE software for clustering of EST from various plant species, such as *Triticum aestivum*, commonly known as wheat. There is interest in optimizing parameter settings for such genomic assemblers in order to get biologically meaningful clustering results.

The EST clustering experiment described in this paper is based on comparing the performance of the PaCE2.3 program with Glimmer2.12 and GeneMark genomic assemblers. While the algorithm behind PaCE uses a string data structure and exact alignment, both the Glimmer and GeneMark assemblers are based on mathematical predictions models or Markov models, to identify gene structures. Each assembler allows for a set of parameters and constraints to be manually set by the user in order to fine tune clustering results to the specific dataset. The input dataset for this experiment are bacterial EST sequences from the Genbank database. The genome for this organism has been assembled experimentally and thus allows for the clustering results from each program to be checked against the experimentally assembled genome.

# 2 Previous Research

Iowa State University is one of the nine universities in the country to offer an NIH-NSF funded Bioinformatics and Computational Biology Summer Internship. As an intern at Iowa State last summer, I participated in a formal workshop, met with leading bioinformatics scientists, such as the developer of Biology Workbench, Dr. Shankar Subramaniam, and I worked on a research project. I focused my research in the direction of genomic assembly methodology and nucleotide sequence analysis of plant genomes. In particular, I evaluated Expressed Sequence Tag (EST) clustering results using the PaCE (Parallel Clustering of ESTs) software developed at Iowa State University.

## 2.1 Internship Project Description

The goal of my summer research project was to evaluate the PaCE algorithm using the *Triticum aestivum,* or wheat, test dataset. The evaluation of the clustering results consisted of multiple test runs of the *T. aestivum* dataset and optimization of the PaCE clustering program parameters. The evaluation procedure involved the following steps:

- The *T. aestivum* dataset consisted of 554,859 sequences downloaded from Genbank.
- The dataset was matched against a vector database from NCBI and a repeat database from TIGR to remove any repeats and contaminants.
- The clean *T. aestivum* dataset of 529,320 sequences was run through the PaCE program preprocessing to remove poly-A tails from the sequences.
- A perl script removed any empty sequences (PaCE preprocessing side-effect) from the dataset.
- The *T. aestivum* dataset was run through the PaCE software to group ESTs that share similarity into distinct clusters.
- Test runs were performed on the ISU Rocks machine, a Linux parallel cluster consisting of 32 nodes, 2 processors per node, 2G of memory per node, and a Gigabit interconnect.
- Test runs were also performed on the ISU Gateway machine, a Linux parallel cluster consisting of 16 nodes, 2 processors per node, processor hyper-threading, 2G of memory per node, and a Gigabit interconnect.

## 2.2 Internship Project Results

Following several test runs, the PaCE 2.3 program did not successfully process the *T. aestivum* dataset on either the Gateway or Rocks parallel machines. The dataset was further analyzed by partitioning it into smaller sequential subsets. A 2,500 sequence subset, which failed clustering attempts, was identified. As characteristic of most plant genomes, the repetitive nature of the *T. aestivum* sequences is suspected to be the cause of observed program overload and preemptive termination.

These results have launched the next phase in the PaCE algorithm optimization, namely the isolation of highly repetitive sequences for independent assembly. This step will involve fine tuning the algorithm at a specific phase, which will be discussed later in the paper. Further evaluation of the algorithm will also focus on optimization of the program parameters and constraint settings.

# 3 Current Research Project

As a central theme of my undergraduate research, I have continued to investigate computational methods and algorithms in bioinformatics. The objective of my undergraduate research project was further research into EST clustering methodology by comparing the performance of the PaCE2.3 program with Glimmer2.12 and GeneMark genomic assemblers. The fundamental concept behind this project is the global application of the PaCE assembler. While GeneMark and Glimmer are based upon Markov Models to predict the genomes of bacteria and archaea, the PaCE program uses a string data structure, a Generalized Suffix Tree, and has been primarily used for the assembly of plant genomes. For that reason, I was interested in determining whether the PaCE assembler could successfully assemble a bacterial genome, or if the PaCE algorithm was limited to plant genomes.

## 3.1 Design of the PaCE Algorithm Evaluation Experiment

In order to evaluate the global application potential of the PaCE algorithm, I designed an experiment comparing the performance of the PaCE algorithm to other algorithms implemented in genomic assembly programs. Thus far, the PaCE program has been primarily used for the assembly of plant genomes, such as wheat, rice and corn. However, the underlying methodology of the PaCE algorithm is a string data structure, the nature of which is not dataset specific. More specifically, the algorithm makes no assumptions about the input dataset and is not dependent on training datasets or other genomic models. Therefore, the focus of this experiment was a new input dataset for the PaCE genomic assembler. Instead of eukaryote EST sequences, e.g. *T. aestivum* in the summer project, test runs of the PaCE program were performed using a much smaller prokaryote microbial genome. In addition, test runs on GeneMark and Glimmer, genomic assemblers that specifically predict the genomes of bacteria and archaea, were performed using the identical dataset. The GeneMark and Glimmer test run results were used as a comparison standard to ascertain the efficacy of the PaCE results.

### 3.1.1 Genomic Assembly Programs in Brief

Glimmer is the primary microbial gene finder at The Institute for Genomic Research (TIGR). Glimmer, short for Gene Locator and Interpolated Markov Modeler, is a system for finding genes in microbial DNA, especially the genomes of bacteria and archaea. It uses interpolated Markov models (IMMs) to identify the coding regions and distinguish them from non-coding DNA. [1]

PaCE, Parallel Clustering of ESTs, is a parallel clustering program developed at Iowa State University. The program is based on parallel construction of a generalized suffix tree (GST) and performance of dynamic pairwise alignments on EST pairs. EST pairs are produced on-demand, in decreasing order of quality, while the quality of overlap equals the maximal common substring length in the GST. Clusters corresponding to each EST are found through successful alignment, and clusters are merged if sufficient overlap exists. [2]

GeneMark for Prokaryotes and Low Eukaryotes (Version 2.1) is a program that predicts genes of microbes with a different methodology. The algorithm in GeneMark improves the gene prediction quality by finding exact gene boundaries. A Markov model framework is the underlying foundation of the GeneMark models, where transitions between hidden states in Markov chains model gene boundaries in GeneMark. [5]

### 3.1.2 Input Dataset

The input dataset are 2,079 *Neisseria meningitidis* strain MC58 (serogroup B) EST sequences from the Genbank database. The genome for this microbe has been assembled and as such allows for the clustering results from each program to be mapped back against the genome. General information, as well as a schematic representation of the Neisseria meningitidis MC58 chromosome, is available at http://www.tigr.org/tigr-scripts/CMR2/GenomePage3.spl?database=gnm.

## 3.2 The PaCE algorithm

The PaCE algorithm is implemented in two stages: the memory intensive parallel construction of the GST and the run-time intensive dynamic programming on the promising EST pairs [2]. The *promising EST pairs* term refers to member EST sequences that may result in the merging of two clusters upon their successful pairwise alignment.

The first phase of the algorithm involves the parallel construction of the GST. A GST is a string data structure built by finding the longest common substrings of the input sequences. A given string of length $l$ has $l$ leaves, each corresponding to a unique suffix, and the string contains at most $l$ internal nodes. The main idea is that shared paths in the GST represent common substrings. Two or more common prefixes will share a common path from the root of the GST, or any path representing more than one string ends in a node or leaf containing the common substring position of the strings that path covers. [4]

In PaCE, the edge-labels of nodes in the GST are the common substrings of input EST sequences. The GST generates promising EST pairs based on the maximal common substring length of two ESTs, which are found rapidly in linear time by traversing the nodes of the GST. Moreover, the promising pairs are created on demand, in decreasing order of quality, or the maximal order of substring length in this algorithm. [3]

**PaCE Algorithm**

- **Phase One**: Identification of pairs with significant overlap potential
  - Parallel construction of generalized suffix tree (GST)
  - Quality of overlap equals the maximal common substring length
  - Production of pairs on-demand, in decreasing order of quality
  - Memory intensive

- **Phase Two**: Perform dynamic programming on the promising pairs
  - Perform pairwise alignment on the promising pair
  - Find clusters corresponding to each EST
  - Merge clusters if sufficient overlap
  - Run-time intensive

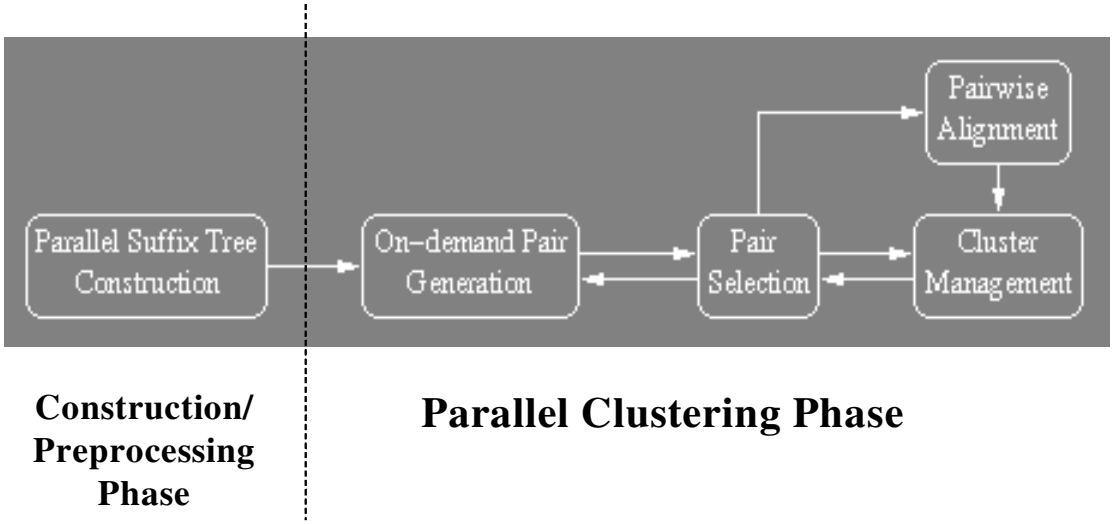


Figure 2: Organization of PaCE software [2]

Phase two of the PaCE algorithm is run-time intensive; it is the pairwise alignment phase. At the start of the second phase, each EST input sequence is a cluster by itself. If an EST sequence from one cluster is successfully aligned with an EST in another cluster, the two clusters are merged. The pairwise alignment continues until all possible clusters are merged. When a pair of ESTs is chosen for alignment and the resulting alignment does not yield sufficient overlap, the time and effort in generating the alignment is wasted. Unsuccessful alignment does not mean that the two clusters cannot be merged; there may exist another pair of member ESTs from these clusters that will produce sufficient overlap upon alignment. [2]

An optimal run time in the second phase can be achieved through early identification of pairs with significant overlap potential, that is, member ESTs resulting in the merging of clusters. The promising EST pairs are identified amongst those pairs with a common substring of at least twenty characters in length, as these EST pairs have the greatest chance of a successful alignment. Early generation of cluster-merging promising pairs would prevent unnecessary extra pairwise alignments. The clusters with sufficient EST overlap would be merged early on in this phase, reducing the number of unsuccessful alignments and wasted run time [3]. Figure 3 relates the increase in aligned and rejected EST promising pairs with the growth of the input dataset.

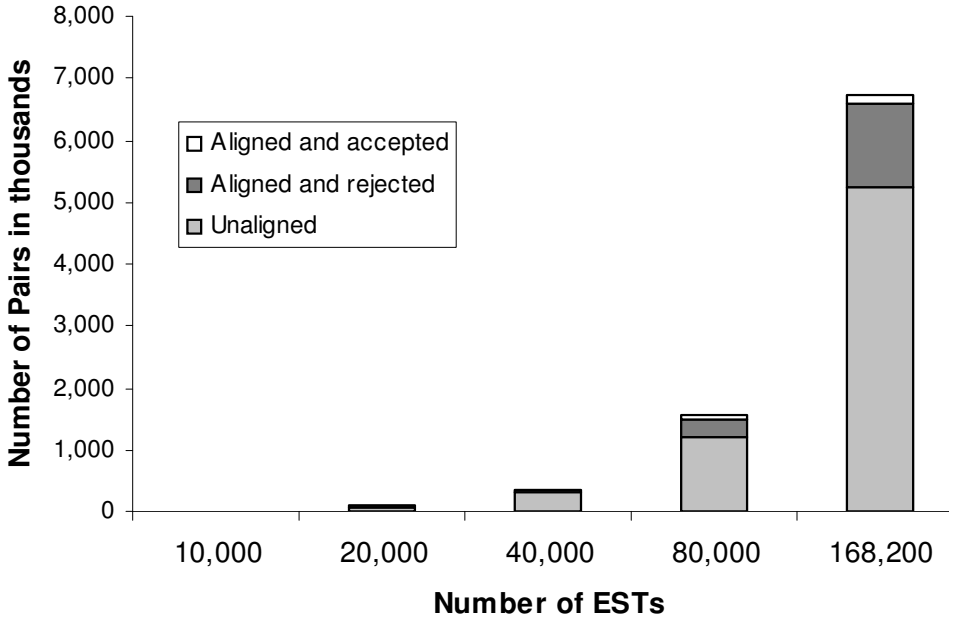


Figure 3: PaCE pair generation [3]

Clearly, as the number of EST sequences in the input dataset grows in size, the number of promising pairs generated for pairwise alignment also rises. However, just as the number of successful alignments increases, there is a quite noticeable rise in the number of unsuccessful alignments. This is caused by generation of EST promising pairs that meet the common substring criteria, yet do not produce a favorable alignment.

The pairwise alignment algorithm seeks to extend the EST pair match from the common substring region. Dynamic programming limits the effort in alignment computation through the use of mismatches and gaps to the left and right of the common substring region [3]. Frequently though, such an extension is not feasible and the promising pair fails the alignment test. In such cases the number of unsuccessful alignments and wasted run time reduce the efficiency of the algorithm.

The next phase of the PaCE algorithm optimization undertaken by PaCE developers focuses on this phenomena and aims to reduce the number of aligned and rejected promising pairs in the second phase of the PaCE program.

# 4 Experiment Results

The input dataset *Neisseria meningitidis* strain MC58 contains 2079 sequences. Experimentally, the genome has been found to contain ~2160 genes. Computational tests in this experiment included the *N. meningitidis* EST sequence dataset assembly by GeneMark.hmm, Glimmer2.12, and PaCE2.3 genomic assemblers.

In addition to performance comparison testing, additional test runs were performed to find the optimal parameter and constraint settings of the programs that yield the closest EST clustering as compared to the actual completed *N. meningitidis* genome.

- GeneMark.hmm PROKARYOTIC (Version 2.1), using the pseudonative model, predicted 1984 genes, or 91.9% of the genome.
- GeneMark.hmm PROKARYOTIC (Version 2.1), using the heuristic model based on *E. coli* model organism, predicted 1914 genes, or 88.6% of the genome.

- Glimmer2.12, with the following optimal parameters, predicted 2157 genes, or 99.7% of the genome.
  - Minimum gene length = 300
  - Minimum overlap length = 30
  - Minimum overlap percent = 60.0%
  - Threshold score = 90

- PaCE2.3, with the following optimal (default) parameters, predicted 1974 genes, or 91.4% of the genome.
  - Minimum length of exact match for cluster merge 30
  - EndToEndScoreRatioThreshold (global alignment match %) 15
  - EndToEndAlignLenThreshold (length of aligned region) 100
  - MaxScoreRatioThreshold (local alignment match %) 5

Detailed information and publications about each of the genomic assemblers described in this paper along with relevant links and actual output files generated by the programs can be found at http://neamh.cns.uni.edu/~cisekk01/cgi-bin/folders/.

# 5 Conclusions

From the raw results, the methodology behind Glimmer2.12 yields the best results, predicting 99.7% of the *Neisseria meningitidis* strain MC58 genome.

However, a few factors influencing these results have to be noted. GeneMark.hmm and Glimmer2.12 are programs developed for the assembly of bacterial (circular plasmid) genomes, and the algorithm takes into account the circular structure of the bacterial genome assembled. PaCE2.3 has been mostly used to assemble linear plant genomes, where the suffix tree data structure takes advantage of large datasets with lots of coverage and duplication.

The PaCE2.3 program seems to be much more data sensitive, because of the exact string matching. The program needs a larger dataset with distributed coverage, whereas the *N. meningitidis* dataset contains many sequences that are contained within other larger sequences, which decreases coverage.

There is no standard of comparison of parameters between PaCE and the other programs. GeneMark.hmm does not have adjustable specific parameters; instead it has the model option. Glimmer2.12 has a gene length and overlap options, while PaCE2.3 has alignment match scores. With the high percentage of gene overlap in this dataset, ~30% on average, it is possible that PaCE2.3 is missing mostly overlap genes.

Bacterial vectors are used as masking vectors in eukaryote genome assembly and frequently become part of the EST sequence dataset of the eukaryote. The bacterial EST sequences present in a eukaryote input dataset may skew genomic assembly results, but eukaryote input datasets are matched against vector databases to remove any repeats and contaminants. However, in this case, a bacterial genome is assembled. It's unclear what or if any vector sequence is present in the dataset.

# 6 Future Directions

The evaluation of EST clustering results in this experiment proved challenging because the experiment lacked a uniform standard of comparison with respect to assembler parameters and settings that could be quantitatively measured. Upon completion, several issues arose in the evaluation of the EST clustering results, as mentioned in the conclusions section of the paper. For this reason, the experiment could be expanded in the following areas:

- Build a program to compare output from the assemblers with the experimental genome data, e.g. gene lengths, gene positions.
- Develop a standard of comparison for the program parameters.
- Test a wider range of datasets, ranging from prokaryotic to eukaryotic.
- Research the mathematical models and training sets underlying GeneMark and Glimmer.

# Acknowledgements

The author would like to thank Qunfeng Dong, Dr. Volker Brendel and Ann Wu for their collaboration on the summer research project. She would like to acknowledge Anantharaman Kalyanaraman for his help with the PaCE program and Dr. Srinivas Aluru for his in-depth explanation of the PaCE algorithm. Special thanks to Lori Kroiss for her ideas and help in presenting the summer project. Thank you to Dr. Phillip East, Dr. Kevin O'Kane, and Dr. Mark Fienup for their contributions. This research project was supported in part by the NIH-NSF Summer Institute Grant.

# References


[1] Delcher, A., D. Harmon, S. Kasif, O. White and S. Salzberg. (1999) Improved Microbial Gene Identification with Glimmer. *Nucleic Acids*. Vol. 27, No. 23, 4636—4641.

[2] Kalyanaraman, A., Aluru, S., Brendel, V. & Kothari, S. (2003) Space and time efficient parallel algorithms and software for EST clustering. *IEEE Trans. Parall. Distrib. Syst.* **14**, 1209-1221.

[3] Kalyanaraman, A., Aluru, S., Kothari, S. & Brendel, V. (2003) Efficient clustering of large EST data sets on parallel computers. *Nucl. Acids Res.* **31**, 2963-2974.

[4] Kalyanaraman, Anantharaman, Srinivas Aluru, Volker Brendel, Suresh Kothari. (2003) Space and Time efficient parallel algorithms and software for EST clustering. *IEEE Transactions on Parallel and Distributed Systems*, 14 (12):1209-1221.

[5] Lukashin A. and Borodovsky M.. (1998) GeneMark.hmm: new solutions for gene finding. *Nucleic Acids*. Vol. 26, No. 4. 1107-1115.

[6] Seemann, Stefan, Do Hong Hai and Erhard Rahm. Genexpressionsanalyse: Verfahren & Datenbankanforderungen. Problemseminar Bio-Datenbanken. WS 2002/2003. Universität Leipzig. http://dbs.uni.leipzig.de/en/seminararbeiten/semWS0203/arbeit4/genexpression.html

[7] Wei Zhu, Shannon D. Schlueter, and Volker Brendel. (2003) Refined Annotation of the Arabidobsis Genome by Complete Expressed Sequence Tag Mapping. *Plant Physiology*. 132, 469-484.