New Algorithms for EST clustering

by Andrey Ptitsyn

submitted in partial fulfilment of the requirements for the degree of Philosopfae Doctor in the Department of Microbiology,

University of the Western Cape

Supervisor Winston Hide, Ph.D.

Co-supervisor Sean Davidson, PhD.

15 April 2000

I declare that "New algorithms for EST clustering" is my own work and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.

List of figures and tables

example of similarity function for two sequences with local similarity region.

Figure 7. Examples of categorized distribution (DF_{ex}) of similarity function $F(W_i)$ values in case of unrelated sequences (**A**) and sequences with a similarity region (**B**).

Figure 8. Partly overlapping distributions X_g (generalized for non-similar sequences) and X_{gs} (generalized for sequences with introduced similarity region).

Figure 9. Box and whiskers plot of simulated Xg (left) and Xgs (right) distributions, 1000 samples each. Medians are plotted as dots, boxes contain 90% of the samples. Two sets overlap by less then 1% of each (1% margins are plotted as whiskers).

Figure 10. The principal data structures of the stringent clustering program.

Figure Basic algorithms of the stringent clustering program

Figure 12. Basic algorithm for loose clustering program

Figure 3 Effect of parameter variation on the clustering results. The only free parameter in both loose and stringent clustering applications is the stringency threshold. This picture shows distribution of cluster size (axis Y, logarithmic scale) after clustering of 885 human EST with a threshold value (parameter) ranging from 0.05 to 0.9 with a step of 0.05

Chapter 1. Literature and software review

1.1 General Aspects of EST data

1.1.1. What is an EST?

An Expressed Sequence Tag or EST is a single pass read from a randomly selected cDNA clone. A typical EST represents only a tiny portion of an entire gene. Initially ESTs were intended to merely identify the expressed gene. Adams et al. (1991 published a widely read article describing use of ESTs in 1991. The process by which ESTs are manufactured requires the construction of an mRNA library. Baldo et al. 1996) have provided a detailed description of how libraries are constructed and how normalization and library subtraction can be used to increase relative representation of less abundantly transcribed mRNAs. An average size of EST libraries ranges typically from 1000 to 10 000 entries (clones). It is estimated, that from 10 to 30 thousand different genes are expressed in a particular cell, depending of the cell type, with an average number of approximately 300 000 mRNA molecules per cell. Thus, an EST library cannot be regarded as a proper representation of the gene expression pattern of a tissue (Vingron and Hoheisel, 1999). An EST library represents only a coarse grained snapshot of the mRNA contents of a certain tissue at a certain moment. Especially, low-expression genes are underrepresented. In some cases EST libraries have been 'normalized', i.e. some additional procedures (usually self-hybridization) has been applied to reduce the representation of highly expressed genes. Normalization enhances the probability of finding clones deriving from rarely expressed genes, but, in turn, distorts the picture of expression pattern in particular tissue.

1.1.2 EST Manufacture

The reverse transcriptase used to manufacture each cDNA in the library will eventually fall off the template (Figure 1), and this will terminate the production of the cDNA. Thus a series of length-differentiated 3' delimited cDNA fragments may be produced for each mRNA that is a viable template in the library. The length of the cDNA will vary, and this is an important factor for development of coverage for each mRNA template of an available gene. Clones are sequenced a single time, from one or both ends of the DNA insert, using universal primers, which are complementary to the vector at the multiple cloning sites. The M13 forward primer may be located near the 5' or the 3' end of the cloned insert, depending on how the inserts were directionally cloned. Consequently, untranslated regions on the ends of the genes tend to be over-represented by sequence tags. Only a few hundred readable bases are produced from each sequencing read, and yet a full gene transcript may be several thousands of bases long. In publicly available databases, EST length varies from less then 20 to over 7000 base pairs, with an average length of 360 base pairs and standard deviation of 120 base pairs (data from dbEST, Genbank rel. 104). Obviously, not all of these sequences are true single-read tags, but they are submitted and accepted as such, bringing extra complications to the EST database analysis. There are also countless variations in the EST generation technique. One of the most significant is using random primers, which results in production of fragments without direction, originating from different non-overlapping parts of the same mRNA (Kapros et al., 1994). ESTs thus provide a "tag level" association with an expressed gene sequence, trading quality and total sequence length for the high quantity of genes, which can be tagged in a given amount of time.

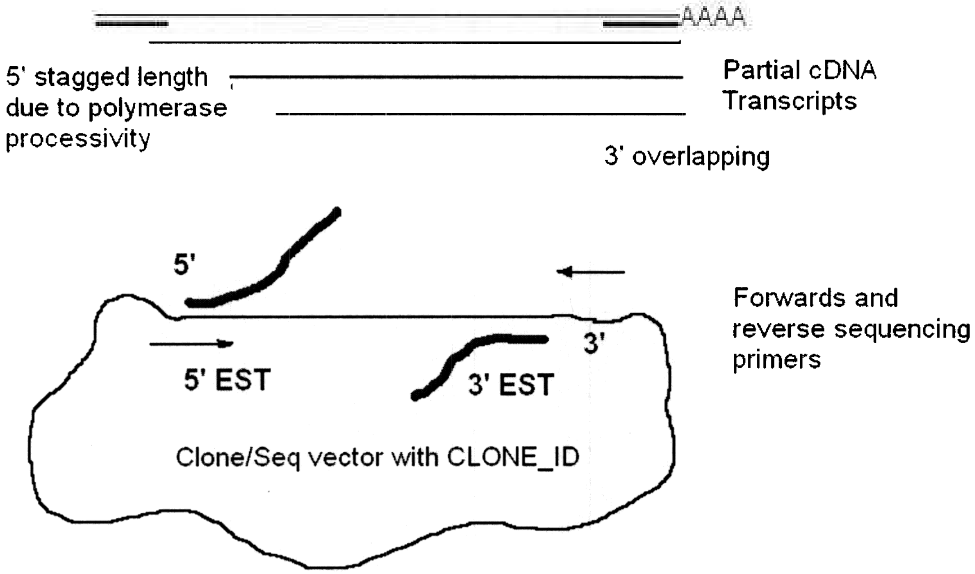
AAAA
5' stagged length due to polymerase processivity
Partial cDNA Transcripts
3' overlapping
5'
3'
5' EST
3' EST
Forwards and reverse sequencing primers
Clone/Seq vector with CLONE_ID

Figure Manufacture of a typical EST

1.1.3. EST data accumulation

A large scale and systematic public effort to isolate all human genes began in 1993 when the Integrated Molecular Analysis of Genomes and their Expression (IMAGE) consortium was formed to create, collect and characterize cDNA libraries from various tissues and different state of normalization (Lennon et al., 1996). This initiative gained significant momentum when Merck & Co. provided funding to the Washington University Genome Sequencing Center to partially sequence clones from the IMAGE cDNA libraries to generate expressed sequence tags. EST sequences are submitted to dbEST - a special division of Genbank (Boguski et al., 1993). Currently over 2 13 391 ESTs are present, including 1497807 human, 486434 mouse, 53636 Soybean and 52829 Zebrafish ESTs (data from 03.09.2000). This information is updated daily and available at http://genome.wustl.edu/est/EST_WEB_TOTALS.html.

1.1.4. EST data quality.

Generation of ES' data results in 'low quality' sequence information. A single read is generated for each EST, and as such will contain errors from its generation at each step. These can include clone orientation, associated clone ID chimeras and missing 3' and 5' reads. Because data are single-pass unedited sequences, they are also subject to errors caused by compressions and base-calling problems resulting in frame shifts. Reference to the Washington University website (http://genome.wustl.edu/est/est_general/disclaimer.html) details common aspects of ES' error. It also claims that clones are sequenced with a typical accuracy of 98%. Occasionally, a longer than average polyA tail at the 3' end of a clone prevents obtaining sequence from the 3' end. Due to the sheer number of clones processed (about 5,000 per week), no manual editing is done on the resulting data. Sizing of the inserts is performed by restriction digestion; agarose gels are imaged and insert sizes determined by computer analysis

Before submission to the public databases, like Genbank, analysis of the ESTs is performed using an automatic processing script developed by Ms. LaDeana Hillier (lhillier@watson.wustl.edu). Most of bacterial and mitochondrial sequences are removed, and the ESTs are compared to the NCBI non-redundant (nr) database using BLASTX (see "contaminating sequences"). The sequences are annotated with respect to tissue and library source, 5' or 3' read, database similarities, and range of high quality data, and submitted directly to the NCBI dbEST database. EST sequence has regions of high quality very close to regions of low quality, where quality can be defined as the number of correctly sequenced bases within a known window of reference. It is possible to utilize poor quality sequence as long as relevant strategies for maximizing their utility are taken.

An attempt of quantitative estimation of data quality has been undertaken in (Hillier et al., 1997). Later studies on the ESTs tend to follow the same technique (Marra et al., 1999). As a measure of quality they have estimated the frequencies of inverted cDNA inserts by comparing mouse ESTs with mouse mRNA set. 53303 matches, representing 84% of the mRNA set were identified. Most matches were to the correct strand, although as much as 6% of matches were to the complement (wrong) strand. For 4% of matches (2 out of 3) at least 2 ESTs mapped to the same position on the wrong strand, suggesting that the match resulted from some non-random events during library construction. Consequently, it was estimated that only 2% of the wrong strand matches might have resulted from failures in directional cloning or human error (Marra et al., 1999).

1.1.5 Discoveries from the EST databases

As a rich source of discovery EST libraries immediately attracted attention of many scientists. During the last decade hundreds of research projects have employed EST data in some form, either as a primary data or a supplementary source of information. Examples include:

-EST information is used in gene structure prediction (Jiang and Jacob, 1998)

-EST data is used to study alternative splice sites and alternative polyadenylation (Gauntheret et al., 1998)

-EST data is used in analysis of correlated patterns of gene expression (Ewing et al 1999, Heyer et al. 1999)

-EST databases are used for SNP data mining (Picoult-Newberg et al., 1999)

-EST data is used for *in silico* differential display, which allows detecting the genes, specifically expressed in particular tissue (Pietu et al., 1999, Hawkins et al., 1999) or behaving differently in normal and diseased state (Schmitt et al., 1999).

1.2 The importance of EST clustering

With the easy access to technology to generate expressed sequence tags (ESTs), several groups have sequenced from thousands to several hundred thousands of ESTs. Currently the majority of the coding portion is in the form of expressed sequence tags (ESTs), and the need to discover the full length cDNAs of each human gene is frustrated by the partial nature of this data delivery. There is significant value in attempting to consolidate gene sequences as they are produced, in lieu of a yet-to-be-completed reference sequence. Unfortunately, most EST data remains unprocessed, and thus does not provide the important high value sequence consensus information that it contains. The low quality sequence data provided can be much improved on, and in order to achieve quality information, pre-processing, clustering and post-processing of the results is required. One goal of the EST study projects is the construction of gene indices, where all transcripts are partitioned into index classes such that transcripts are put into the same index class if and only if they represent the same gene or gene isoform. Raw ESTs are assigned to clusters according to their sequence similarity or annotation (such as clone information). Accurate gene indexing facilitates gene expression studies as well as inexpensive and early gene sequence discovery through the assembly of ESTs that are derived from genes that have yet to be positionally cloned or obtained directly through genomic sequencing.

1.3 Overview of EST clustering

The early clustering procedure as it was initially described by (Hillier et al., 1996) is two-fold. ESTs are first submitted to a fast pair-wise sequence comparison to build up rough clusters. Then the initial clusters are treated by a slow, but accurate alignment procedure. This general concept of clustering has gained acceptance in the field and most of the ES' related projects do it in some form on the initial stages of data preparation.

EST clustering is performed as a process that utilizes 'clustering information' that is less and less definitive. Initially sequence identity provides a good guide to cluster membership. Shared annotation provides joining information that can be of more variable quality. Thus the number of accurately clustered ESTs is heavily dependent on a strategy that can assign cluster membership based on verifiable criteria; sequence identity is currently the most useful of these. Clustering can be performed with or without sequence consensus generation. It is preferable, although more difficult, to manufacture a consensus sequence from each cluster.

An overview of a modern clustering procedure was undertaken in a tutorial on the EST clustering, presented at ISMB99, Heidelberg (Hide et al, 1999b). Figure 2 depicts the principal stages of the clustering process.

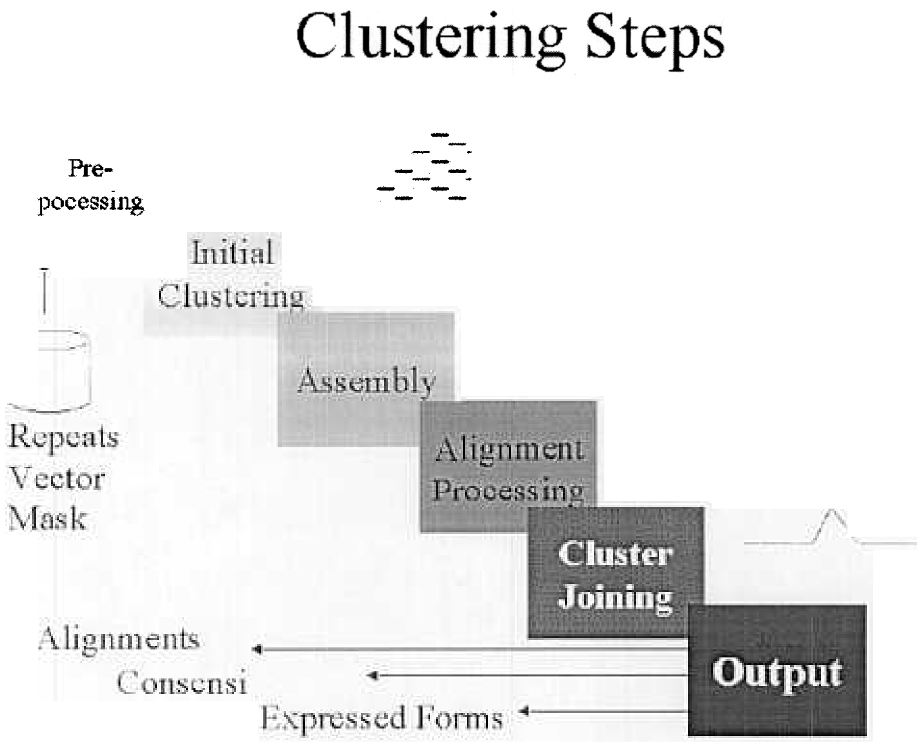

Figure 2. Basic clustering steps.

The steps suggested in EST clustering are as follows:

1.3.1 Preprocessing

Sequences are masked for repeats and vectors, and formatted for the clustering engine. Sequence quality is often assessed at this step. A minimum number of residues are accepted above a known quality threshold. For example, SANBI's STACKPack accepts only masked sequence data above 50bp in length. NCBI discards ESTs with a window of less than 100bp of 'clean' data.

1.3.2 Initial clustering

An initial clustering is performed based on a fast measure of high sequence identity, like D2. ESTs having a high degree of similarity, detected by such fast, although rough measure are grouped in one cluster. Clusters, formed on this stage require further verification.

1.3.3 Assembly

Assembly can be either part of the initial clustering (as used in some high-stringency supervised clustering systems) or separated into clustering followed by assembly performed by a specialist assembly package such as PHRAP (with assessment of residue quality turned off) or CAP3

1.3.4 Alignment processing

Aligned clusters, particularly those generated by a loose clustering engine, need to be processed for errors and alternate forms of expressed sequences. Consensus generation may be a result of this step (as in STACK), or a consensus can be accepted directly from the assembly step.

1.3.5 Cluster joining

Once clustered, clusters and/or cluster consensi can be further associated by additional information contained in annotation, such as clone ID.

1.3.6 Output

Defining an output format for the clustering process is problematical. Information required often includes alignment (alternate splices, polymorphism and error assessment), raw cluster membership, and contextual links. Nonetheless, results must be easily incorporated into existing software packages, which in general have not been designed to support the complexity or evolving nature of clustered EST data.

1.4 What is an EST cluster?

One of the most explicit definitions of an EST cluster is following:

A cluster is fragmented EST data (DNA or protein) and (if known) gene sequence data, consolidated, placed in correct context and indexed by gene such that all expressed data concerning a single gene is in a single index class, and each index class contains the information for only one gene (Burke et al., 1998).

Yet in real life all EST clustering systems available for review have a slightly different meaning or add some specific flavor to the concept of an EST cluster. Even STACK clusters, produced by the authors of this definition, don't fit it exactly: Up until version 2.31 of STACK ignored mRNA and genomic DNA information, even if it was available. STACK has also limited the scope of its clustering engine to a particular tissue prior to index manufacture, built upon tissue clustering. Other systems don't use tissue of origin as one of the clustering criteria, keeping it as additional information only (Bouck et al., 1999). The meaning of an EST cluster also depends on the way the alternative gene variants are apprehended. TIGR Human Gene Index (HGI) separates each alternative variant to a separate cluster (Adams et al. 1995). Unigene keeps all variants in one group as long as they have some common part, but produce no consensus.

1.5 EST clustering and Statistical cluster analysis

By a most general definition cluster analysis is a process of division of a set of objects into smaller subsets, which are uniform in some sense. The number of subsets can be given or unknown at the start of the process (Aivazyan et al., 1989). The term of cluster analysis was initially introduced by Tryon (Tryon, 1939). To formalize the problem of cluster analysis, the objects are represented as points in a corresponding space. In such space, the objects that belong to one cluster are situated on relatively small distance to each other. The fundamental question of a cluster analysis is a choice of statistical metric of similarity or dissimilarity between the objects, which defines the space.

If all objects are sampled from a general set with a common covariation matrix, the most appropriate choice of distance is a Machalanobis-type metric: The other most common metrics are Euclidean or Hamming metric for binary spaces. There are dozens of different metrics and modifications, developed for different cases of data, each developed specifically to make the best possibly representation of the real life meaning of the data and allow a sensible interpretation of the result.

Jnlike many other statistical procedures, cluster analysis methods are mostly used when we do not have any a priori hypotheses, but are still in the exploratory phase of research. In a sense, cluster analysis finds the "most significant solution possible." Hence, another crucial point in cluster analysis is a choice of a measure for clustering quality. This metric is optimized during the clustering procedure and often called a "linkage rule". Most often this metric is built on the basis of some inter-cluster distance measure. If this measure is correctly selected for the particular data, then the further the resulting clusters are from each other the better. A list of most widely used metrics is given on the Statistica web site (http://www.statsoft.com/textbook/). 'omprehensive analysis of various inter-cluster distance measures, as well as clustering algorithms has been performed (Aivazyan et al., 1989).

Clustering of ESTs as it's seen by bioinformatisists is not exactly classical cluster analysis as understood by statisticians. Only a few papers, (Eisen et al., 1998, Heyer et al., 1999) have correctly stated the problem in statistical terms and scrupulously defined all metrics and criteria. These works were devoted to the cluster analysis of genome-wide expression patterns. Although closely related, work of Esisen has pursued differing goals, discrete from çlustering of raw EST data, which prevents a direct comparison to major EST clustering systems such as TIGR Gene Index, Jnigene, STACK, DOTS and others. None of the other works available for the review has a publication or a documentation file, stating the problem in correct statistical terms and defining the objects, the space, the metrics for inter-object and inter-cluster distance and the cluster quality measure. All algorithms currently used for EST are heuristic. Apart from the educational background of the developers, there are a few factors that pre-determine this approach.

In cluster analysis practically all statistical aspects are about classification and justification of classification of the object with complicated relationships. This may be why the word "clustering" is often used in case of ESTs instead of cluster analysis. In the case of clustering the fundamental problem is the selection of a metric that is reduced to a simple binary digit, i.e. sequences either match or don't. ESTs are put in one cluster if they have near identical matches. Essentially, the fragments put together are assumed to belong to the same DNA and possible mismatches are mostly resulting from misreads and represent noise. The inter-cluster distance, as well as inter-object distance is also reduced to binary - in an ideal situation, the clusters of ESTs related to one gene should be infinitely close to each other while those related to different genes are infinitely distant and no intermediate states should make sense. Following the simplification of a distance measure, the problem of clustering quality estimation also reduces to a trivial task: the closer we get to detecting all exactly matching fragments the better. Thus, from the statistical point of view, all available for review systems for clustering ESTs, utilize a trivialized version of cluster analysis. The criteria used for clustering can be traced to the nearest neighbor type of linkage rule (in case of STACK, for example) or furthest neighbor - multiple linkage type (Sanigene, TIGR HGI).

This may not be exactly true in some cases of stringent clustering as stringent clustering involves consensus generation and relies on some form of consensus to represent a whole cluster during the clustering procedure. The majority of research projects, which include of EST clustering, use readily available software. Initially developed for shotgun contig assembly, it works fine on relatively small data sets. For example, TIGR Gene Assembler (Sutton et al., 1995) was used to cluster 5692 EST from poplar *P.tremetula x tremuloides* (Sterky et al., 1998) and CAP2 was used to assemble ESTs for the Prostate Expression Database (Hawkins et al., 1999). For a reliable assembly it's important to make pair-wise alignments in a certain order to keep the resulting consensus representative. Some similarity/dissimilarity measure, or, in other words a distance between the sequences, typically determines this order. Yet in all these cases the distance measure and linkage rule remain hidden inside third-party programs and are not stated as a part of the EST analysis.

1.6 Supervised and unsupervised clustering

Clustering methods can be divided in two general classes, supervised and unsupervised clustering. In supervised clustering, objects are classified with respect to known reference objects, or "seeds". In unsupervised clustering, no pre-defined objects are used and the number of resulting clusters is typically unknown until the end of the clustering procedure. Some EST clustering systems are strictly or partly supervised, like TIGR Gene Index and IMAGene, some are totally unsupervised like STACK and some use a combination of two approaches, like Unigene. In the systems, using some form of supervision, a genomic DNA and/or an mRNA sequence data is as a core used to build up ESTs.

1.7 Stringency of EST clustering

Stringency is a single parameter, which in some form can be traced in most of the systems for EST clustering. Although it can be rarely found as a single parameter, stringency is always present in description of the clustering process as a set of criteria used to detect matching ESTs. More stringent clustering occurs when more strict rules are applied to assign an EST to a particular cluster. Stringency is closely related to the problem of inter-object and inter-class distance. More stringent clustering typically produces smaller number of clusters with shorter consensus sequence (if produced at all), but each cluster has much better quality. Less stringent or "loose" clustering tends to produce bigger clusters at a cost of possible inclusion of paralogous expressed genes and lower fidelity data.

Clustering systems using BLAST as sequence comparison engines often use score, generated by BLAST as a stringency measure. In (Schmitt et al., 1999) authors define stringency level as $E<10^{-4}$ and 95% of sequence identity. In (Wolfsberg and Landsman, 1997), stringency level is chosen at $P<10^{-87}$ plus additional manual selection of matching ESTs with higher P values. (Hishiki et al., 2000) have used FASTA (Pearson and Lipman, 1988) to compare ESTs and percentage of identity as a measure of stringency. They consider EST matching if they have 95% of identity in an overlap longer then 50bp or 70% of tag length and the overlap started with a GATC. (Gauntheret et al., 1998) have chosen to use a combination of BLAST score and identity percentage from FASTA alignment. Any pair of ESTs scored over 150 with <10 mismatched positions at either extremity and >95% identity was clustered.

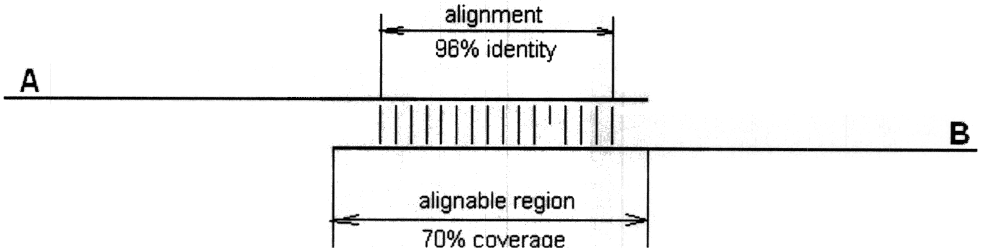

Figure 3. Constraints placed on the alignment quality and coverage of the aligned region to reduce problems caused by chimerical sequences. Data from Wagner et al. Abstracts 1999 Genome Sequencing and Biology, P340, Cold Spring Harbor Laboratory.

1.8 Contaminating sequences

The first study of inclusion of contaminating sequences in public database (Genbank) in 1992 has already stated that the problem grows faster than the size of the database itself (Lamperti et al., 1992). Vector fragments were found in 0.23% of all sequences available at the time. Improved methods of vector identification, applied in GSDB (Harger et al., 1998) reveal slightly more vectors in the same data set - 0.3%. This was also confirmed by (Miller et al., 1998), who studied vector contamination dynamics in Genbank from 1982 to 1996 (partial). The percentage of contaminated sequences shows a persistent increase and ranges from 0% in 1982 and 1984 to 0.57% in 1995 Surprisingly, the percentage of contaminated ESTs over the years is lower then the average for Genbank (ranging from 0. to 0.3 per cent) and even decreases in period from 1992 to 1996, despite of a dramatic increase of available sequences (from 7345 in 1992 to 493816 in 1996). In the classic work on analysis of Human ESTs (Hillier et al., 1997) have given a quantitative estimation of the level of contamination in the EST libraries. To assay library quality, EST sequences were screened against databases of bacterial sequences, mitochondrial sequences and vector sequences. All libraries contained mitochondrial sequences ranging from a high of 16% of ESTs to a low of less then 1% of ESTs. Some libraries are found to contain as much as 20% of bacterial contamination. A search of entire EST subset of EMBL against vector database performed by (Miller et al., 1999) gave an estimated contamination level of 0.27% which is generally in line with the previous studies (Lamperti et al., 1992, Harger et al., 1998).

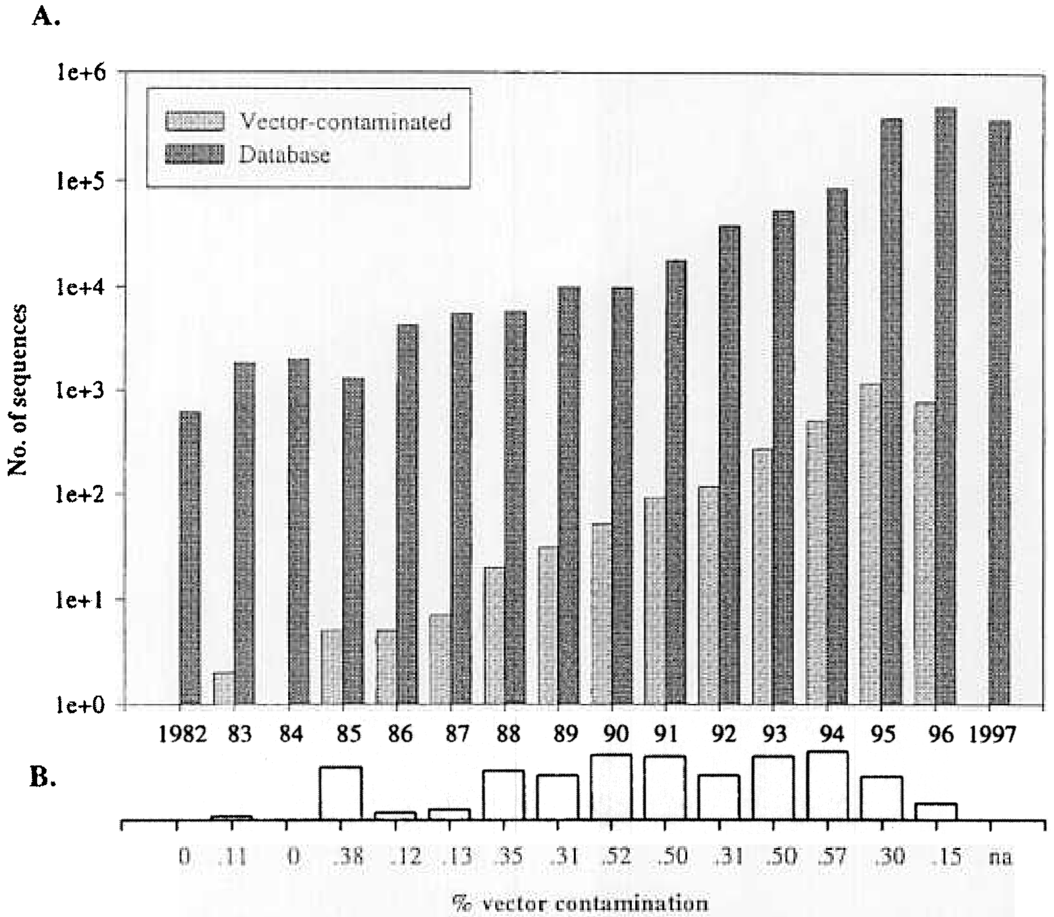

Figure 4 (A) Yearly number of sequences and vector-contaminated sequences (*y*-axis) submitted to Genbank from 1982 to 1997 (*x*-axis). **(B)** The percent of vector contamination in the database for each of the years indicated in (A). No vector-contaminated sequences were found for 1982 and 1984. Data for 1996 are partial (trough June-August) and data for 1997 have not been analyzed. From Seluja et al., (1999)

Γhe most comprehensive collection of cloning vectors is VectorDB, which can be found at http://vectordb.atcg.com/vectordb. Although this site has search facility, there is no single file with a collection of possible contaminating sequence samples for download. Having such a collection is important for local searches and cleaning of an ES′ dataset for further analysis. A collection of cloning vectors, specifically prepared for this purpose is available from NCBI ftp site: ftp://ncbi.nlm.nih.gov/blast/db/vector.Z. NCBI also maintains an online web-based tool for searching and masking out various contaminating sequences: http://www.ncbi.nlm.nih.gov/VecScreen/VecScreen.html. VecScreen pages of NCBI web site contain comprehensive documentation for the search engine and database together with a thorough description of possible sources and the potential consequences of contamination. A good collection of links to cloning vector-related resources in the Internet can be found at http://www.biosupplynet.com/cfdocs/btk/btklist.cfm?category=67.

1.8.1 Repeats

Unlike cloning vector fragments, repeats cannot be regarded as contaminating sequences. All of the human genome can be roughly divided in two fractions of DNA: repetitive and unique sequence. Traditionally, a portion of the unique fraction is thought to comprise the obvious functional constituents of our genome, including exons, introns, and regulatory DNA elements. Interest of biologists is sharply focused on these parts of genome and clustering of EST pursues the goal to unite pieces of unique sequences contained in tags together. With the exception of telomeric and centromeric repeat sequences, the functional significance of the vast majority of the repetitive fraction is less clear. Presence of the repetitive sequence fraction in the ESTs obscures the similarity of the unique parts and represents a serious challenge for clustering. On the other hand, clustering can be used as a tool to detect new repeats, not characterized yet by other methods.

Since the early experiments of re-association kinetics of single-stranded human DNA (Britten and Kohne 1968), various gradations of repetitiveness have always been recognized on the basis of both the copy number and the degree of sequence similarity. The number of repeats range from the prolific (LINES, SINES, -satellite, etc., in the 100,000's) to the relatively few. By virtue of the fact that multi-gene families exist, genes themselves may be repetitive in nature. Many of the most well studied members of gene families (hemoglobins and HOX genes), however, appear to be sufficiently divergent (Ohno, 1999) or localize to discrete clusters of tandem arrays (rRNA genes, HLA genes, immunoglobulin gene segments). These are often distinguished based on the sequence divergence of individual members or their clustered position within the human genome. The term "unique" DNA, therefore, is relative, determined largely by what we already know about any given genome. The basic paradigm regarding the repetitive and unique nature of DNA sequence underlies any effort to sequence a genome. In fact, the reason that any genome can be sequenced and assembled is that there is sufficient unique sequence interdigitated among the repetitive fraction, the repetitive fraction is sufficiently divergent, and/or the repetitive fraction can be distinguished as such (Eichler, 1998). In case of EST clustering, distinguishing the repetitive fraction is just as vital. Detection of paralogous genes and assigning of their tags to one or separate clusters is determined by clustering stringency (see above). High-copy interspersed repeats (LINE, SINE) must be excluded from consideration.

In newly sequenced genomes, such as plant and other eukaryote systems, repeat sequences represent a common and frustrating clustering problem. Repeat databases provide a resource against which repeats can be detected. The repeat databases are dependent on continuing curation and detection of novel repeats in genomes and thus provide a valuable resource. Since the early 90's, the most comprehensive (and practically unchallenged) repeat collection – Repbase, has been supported by Genetic Information Research Institute (http://www.girinst.org) (Jurka et al., 1992, 1998).

1.8.2. Microsatellite repeats and low complexity sequences

Microsatellites are tandemly repeated sequences of 2-6 bp (Tautz 1993). They have been used extensively for genetic mapping and forensic and population studies. However, much remains unknown about the possible functions microsatellites may have in the genome and about their patterns of sequence variation and mutation. Variation in the sequence of microsatellite alleles may affect the interpretation of genetic mapping and population studies in which microsatellites are used. Microsatellite repeats are remarkably variable by number of copies, small deletions, insertions and single base mutations inside the repeat (Bull et al. 1999). This variability and multiplicity of the microsatellite repeats makes their recognition by comparison to a sample, stored in a database, ineffective.

Low complexity sequence is a more general term for stretches of DNA with or without detectable repetitive structure. Lack of a certain consensus makes them impossible to detect by comparison to a sample. But like interspersed repeats, microsatellite repeats and low complexity regions also have the potential to provide an artifactual basis for cluster membership. The problem is more significant for strategies that employ alignable similarity in the first pass cluster assignment. Word based cluster assignment can be modified to provide low weight to low complexity words. The latest version of BLAST (Altshul et al., 1997) widely used as a sequence comparison engine in EST clustering, is capable of filtering out low complexity sequences. In the new EST clustering applications being developed at SANBI (see Chapter 2) and elsewhere, there is no need for masking of low-complexity DNA as such regions tend to have a highly redundant oligonucleotide composition. Because these sequence comparison algorithms scale oligonucleotides according to their potential information content, highly redundant oligos are given very low weight and low-complexity regions are therefore excluded from consideration.

1.9 Masking strategies

The most effective method to remove contaminants is to compare each read against a reference database of repeats RepBase (J.Jurka et al.1998) and vector sequences (VecBase, http://vectordb.atcg.com or vector collection at NCBI, ftp://ncbi.nlm.nih.gov/blast/db/vector.Z) using an algorithm that is reasonably fast and accurate. XBLAST (NCBI tools, ftp://ncbi.nlm.nih.gov/toolbox/ncbi_tools) and Cross-Match, an implementation of the Smith-Waterman-Gotoh algorithm developed by Phil Green have been used successfully, with Cross-match demonstrating greater flexibility and sensitivity than XBLAST. DUST is used for masking repetitive sequences at NCBI (http://www.ncbi.nlm.nih.gov/UniGene/build.html). It's also able to reveal and mask low-complexity sequences. Another recent development, RepeatMasker (Smit, AFA & Green, P. unpublished results), available from Washington University (http://ftp.genome.washington.edu/RM/RepeatMasker.html) is also able to mask huge amounts of data and recognize low-complexity DNA, for example stretches consisting of >84% of CA or >87 GT (default settings). There is also an additional program called "sputnik" (http://bozeman.mbt.washington.edu/sputnik.html) designed to reveal potential simple repeats, often annotated like (CGAA)n. From the algorithmic point of view RepeatMasker is a convenient wrap-around, based on the same CrossMatch of Phil Green. There are also other programs, specifically designed for masking repetitive sequences, like Censor (J.Jurka et al., 1996), available from Genetic Information Research Institute (http://www.girinst.org). Censor is available via the Web interface or mailing server and allows "censoring" (i.e. masking) of query sequence(s) against databases of human, rodent and plant repeats. Although this software is developed by the same organization as the database of repetitive sequences (Repbase), censor is not a widely used masking tool for EST clustering. This may be as a result of complicated procedure of obtaining source code, a lack of current maintenance and performance on huge data sets.

In cases where a direct identity is found with a repeat or vector subsequence a 'mask residue' can be substituted into the read. The resulting runs of NNNNs or XXXXXs will be ignored by most clustering engines (Figure 3a and 3b). A problem arises when an EST library is presented that is from a novel organism for which the repeats have not been characterized. In this instance it may be necessary to employ 'blind' repeat masking if an algorithm is available. Repeat masking is necessary if the repeats are large enough to represent a source for artifactual contamination. An exploitable feature of sequence contamination in loose unsupervised clustering is that, if the tools work as intended, then the contaminated sequences *and all related sequences* will be clustered together. There is no automatic method to identify a contaminated cluster, but once it is identified only that cluster needs to be decomposed into its original sequences and re-processed (no other cluster will be affected by the sequences in the contaminated cluster).

```
>T27784 g609882 | T27784 CLONE_LIB: Human Endothelial cells. LEN:
337 b.p. FILE gbest3.seq 5-PRIME DEFN: EST16067 Homo sapiens cDNA
5' end
AAGACCCCCGTCTCTTTAAAAATATATATATTTTAAATATACTTAAATATATATT
TCTAATATCTTTAAATATATATATATATTTNAAAGACCAATTTATGGGAGANTTG
CACACAGATGTGAAATGAATGTAATCTAATAGANGCCTAATCAGCCCACCATGTT
CTCCACTGAAAAATCCTCTTTCTTTGGGGTTTTTCTTTCTTTCTTTTTTGATTTT
GCACTGGACGGTGACGTCAGCCATGTACAGGATCCACAGGGGTGGTGTCAAATGC
TATTGAAATTNTGTTGAATTGTATACTTTTTCACTTTTTGATAATTAACCATGTA
AAAAATG
```

```
>T27784 g609882 | T27784 CLONE_LIB: Human Endothelial cells. LEN:
337 b.p. FILE gbest3.seq 5-PRIME DEFN: EST16067 Homo sapiens cDNA
5' end
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxTATTTNAAAGACCAATTTATGGGAGANTTGCA
CACAGATGTGAAATGAATGTAATCTAATAGANGCCTAATCAGCCCACCATGTTCTC
CACTGAAAAATCCTCTTTCTTTGGGGTTTTTCTTTCTTTCTTTTTTGATTTTGCAC
TGGACGGTGACGTCAGCCATGTACAGGATCCACAGxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxAACCATGTAAAAAT
G
```

Figure 5. Example of an EST sequence in FASTA format and same sequence after masking vs. Repbase and Vecbase.

1.10 EST Clustering methods

Small projects, clustering of a few dozens or hundreds or even thousands of ESTs are indeed 'trivial' and can be approached by standard tools of contig assembly or even multiple alignment. The real challenge is the amount of data, available now in public and private databases and waiting to be clustered. Taking into account these millions of ESTs, obtaining the trivial binary distance between fragments is far from a trivial job even for available supercomputers. Modern tools of sequence comparison (Smith-Waterman, FASTA, BLAST) are mostly built for a different purpose: searching. They are all different variations of an alignment algorithm, i.e. correct position of sequence elements (nucleotides or groups of nucleotides) against each other maximizes some score. The purpose of this process is to detect and measure quantitatively the similarity (distance) between any 2 sequences compared. Smith-Waterman is the most exhaustive and computationally expensive tool, deriving the best sensitivity and detecting weak similarities. FASTA and BLAST are less sensitive and trade some sensitivity for speed. As mentioned previously, the distance measure in EST clustering is reduced to binary, it is therefore only necessary to detect a near or perfect match. Extension penalties and gapping manipulation become less important in an initial assessment of pair-wise identity. It is therefore important to 'head for speed' over sensitive comparison. Use of a banded Smith-Waterman on already compared clusters is an approach that is tenable for further consensus generation.

1.10.1 Clustering with contig assembly tools

Conventional "shotgun" sequencing produces a large number of fragments, highly redundant and with various degree of overlapping. The following assembly relies on overlapping alignment of the called nucleotide bases to recreate a representation of the original DNA sequence. This makes the task of shotgun sequence assembly very close to EST clustering, even though it was originally developed for genomic DNA sequencing. Algorithmically most of the shotgun assembly tools are build to the same principal schema as early EST clustering tools: they are generally two-fold, implementing a fast procedure for rough detection of sequence matches and thorough post-processing with accurate, but slow alignment and consensi generation. All the shotgun assemblers referenced except ACEmbly use a hashed query sequence algorithm for finding initial matches. Some of the more sophisticated programs are able to make use of ancillary information such as basecall probability estimates, clone orientation and length estimates to improve the quality of the contigs produced, especially in the presence of repetitive DNA. These developments make shotgun assembly tools even more similar to EST clustering systems. Relevant programs include Gap4, (Bonfield et al, 1995), CAP3 (http://genome.cs.mtu.edu/cap/cap3.html) and Phrap.

Gap4 (http://www.mrc-lmb.cam.ac.uk/pubseq/overview.html) implements a simple and relatively slow algorithm where new shotgun reads are compared against contig consensus sequences looking for hash hits that extend the contigs in a greedy fashion. The program is memory efficient and scales to megabase-sized YAC assembly projects with more than 10,000 reads. Gap4 can read and extend contigs produced by other programs e.g.. Phrap, and CAP2.

Phrap (http://bozeman.genome.washington.edu/index.html) has incorporated error probability information deeper into assembly and editing than any other assembly package. Alignments are stored in detail, as a directed graph in RAM, leading to large memory requirements (1 GB for a YAC assembly). A consensus sequence is extracted by using a mosaic approach where the best read in any multiple alignment is directly copied to the result. Phrap has no specific features to handle EST idiosyncrasies like alternative splicing etc. Assemblies are relatively fast, however a single sequence addition requires all previous calculations to be repeated. An optimized and parallelised version of Phrap is commercially available (http://www.spsoft.com/).

CAP3 (Huang, 1999; http://genome.cs.mtu.edu/cap/cap3.html) is a third generation of CAP family, after CAP (Huang, 1992) and CAP2 (Huang, 1996). This program is relatively slow, but tends to produce longer consensi contigs with fewer internal errors. CAP3 can utilize ancillary information to guide sequence assembly: mate-pairs, clone size, orientation/location, and basecall error estimates. CAP3 is used at MIPS (Munich Information Centre for Protein Sequences) to assemble Unigene clusters (http://www.mips.biochem.mpg.de/proj/human)

PEDB: the Prostate expression database contains clustered ESTs and full-length cDNA derived from more than 40 human prostate cDNA libraries, which represent a wide spectrum of normal and pathological conditions. Contaminating sequences and repeats are removed on the pre-processing stage using a core program called AnalDemon (http://www.mbt.washington.edu/PEDB/software). AnalDemon first employs CrossMatch (P.Green, unpublished) to screen the ESTs for vectors and regions similar to complete *E.coli* genome. Then RepeatMasker (Smit, unpublished) is used for masking interspersed repeats and low-complexity regions. EST assembly and clustering is done by CAP2 (Huang, 1996). All resulting clusters are annotated by searching the Unigene, Genbank and dbEST using BLAST. All programs, used in production of PEDB are hold together by a system of Perl5 scripts. The database is accessible via the web at http://www.mbt.washington.edu/PEDB. The differential expression of each EST species can be viewed across all libraries using a Virtual

Expression Analysis Tool (VEAT), a graphical user interface written in Java for inter- and intra-library species comparisons.

TIGR Assembler, is a relatively fast sequence assembly program that has been proven in assembly of a diverse collection of repetitive microbial genomes (Sutton et al, 1995). It employs a standard rapid oligonucleotide content comparison to reduce search time. Pair-wise comparisons generate a list of potential (end-)overlaps. Non-repeat fragments seed subsequent assembly of listed overlaps. Memory usage for the query hashing stage is 32 * 4n bytes where n is the word size (oligo size) e.g. 512 MB for 12-mer oligos. Four criteria control the match decision: minimum length of overlap, maximum number of local errors, percentage of best possible score, and maximum length of overhang. TIGR use a combination of BLASTN and TIGR Assembler to produce their own consensus sequences. This combination is relatively slow: multiple-months to produce THC (Tentative Human Consensus) sequences for all the human ESTs (Quackenbush et al., 2000). The THC database is biased towards splitting EST clusters due to this stringent assembly stage (Burke et al., 1998). The strictness of matching criterion has the advantage of often preventing chimerism and contamination from tainting index groups but results in a more fragmented representation of the data that is less able to incorporate error prone sequence. This strictness often disallows the combination of sequences with sufficient diversity so that sufficiently divergent ESTs that sample alternative splice forms of the same gene are kept in different assemblies but they are linked as being splice variants in those cases where the ESTs match sequenced genes with known isoforms in EGAD (Expressed Gene Anatomy Database).

ACEmbly (http://alpha.crbm.cnrs-mop.fr) is a fast hashed-database program where oligos are sampled from all sequences and used to prime alignments and then assemblies. ACEmbly fragment ordering is initially based on a distance metric calculated from the number of oligos two sequences have in common divided by their total number of sampled oligos. This procedure is fast, but prone to errors with repeated sequences.

To date the majority of research projects that involve EST clustering employ readily available shotgun assembly tools. These tools are very accurate, well established and developed and work very well on small-scale data from hundreds to tens of thousands of ESTs. The conventional shotgun assemblers are also widely used to assemble clusters of ESTs in specialized EST clustering systems.

1.10.2 Alignment scoring methods: BLAST and FASTA

BLAST is an algorithm efficiently implemented on many platforms. Although not developed specially for clustering, BLAST sequence comparison is widely used initially in EST clustering because it's readily available and flexible enough to be tuned for the task with a change of default parameters (Hide et al., 1999). Using the standard BLAST II application, available from NCBI anonymous FTP, it is possible to set up a stringent match set of parameters as follows:

-e expectation value set to 0

-G cost to open a gap can be increased,

-E cost to extend a gap can be increased,

-q mismatch penalty increased

-r match reward increased

-a number of processors to use can be adjusted,

-W (word size, found on NCBI web) set up for longer words (default 11)

FASTA is less widely used, but can be also applied for the same purpose. Generally, it allows the same type of variation in parameters as BLAST - increasing the k-tup parameter to increase speed and raising a threshold to pick up only the strongest similarity.

There are many clustering systems employing conventional sequence comparison tools and this number is growing.

Examples of systems, using BLASTN or FASTA for clustering include, but not limited to: INCA (Graul and Sadee, 1997, http://itsa.ucsf.edu/~gram/home/inca/) SEALS (http://www.ncbi.nlm.nih.gov/Walker/SEALS/), Zymogenetics' REX (Yee and Conklin, 1998) and the anonymous tools used to produce Gene-express, and the Merck Gene Index. The BLASTN/FASTA algorithms were designed for searches for local similarity and homology, which is a different goal from a search for alignments consistent with overlap/contiguity. In consequence, the script wrappers put around the homology searches have three functions: to run the search, to parse the search results and to enforce the clustering stringency parameter (or set of criteria), discussed above. These extra match criteria cannot be handled directly by the database searching programs so later, more sensitive and slower pair-wise sequence alignments may be run to improve the comparison accuracy (Merck Gene Index) (Williamson et al., 1995).

Glaxo's "Dynamic" assembler (Gill et al, 1997) uses interactive rounds of BLASTN searches of EST databases to identify possible overlapping sequences, which are fed to Gap (Bonfield et al, 1995) for Greedy assembly. This approach was quick to implement, and effective for simple assemblies but suffers from pauses between each round of assembly. Greedy assembly is prone to errors with repetitive sequence regions; static HTML output limits interactivity/editing and alternative BLASTN alignments displayed simultaneously in one alignment may cause confusion for the user.

The original UniGene implementation (Boguski and Schuler, 1995, Schuler et al, 1996) relied on a BLASTN-like hash based search to identify pairs of sequences with at least two 13-mer words separated by no more than two bases. Then these sequence pairs were aligned more accurately within a 10-base window either side of the diagonal identified by the oligo match. The alignment was scored +1 for a nucleotide identity, -2 for a mismatch, -1 for a gap, and 0 for an N ambiguous base. An alignment quality was calculated by dividing the score by the alignment length and had to exceed

0.91 to be accepted. An extra constraint was that the aligned region had to extend within 35 bases of the end of either sequence. Unigene itself only accepts data of known good quality, (high quality base count, presence of poly-A tail etc.) so not all ESTs that might pass the above alignment criteria are ever checked for matches. Explicit lists of excluded sequences (>200,000) are not published nor assemblies, alignments or consensus sequences. Sources for Unigene include Genbank Genomic, dbEST and Genbank mRNAs. Genbank Genomic sequences are electronically spliced exons (vGenes). Initial clustering is performed by comparing the set of gene sequences (mRNA or genomic sequences, many of which are complete CDSs) with itself. Sequence pairs that are sufficiently similar are grouped together to form initial clusters. EST to gene links and EST-to-EST links are added to these clusters. The set of ESTs is compared with the set of genes using WHALE (http://nucleus.cshl.org/meetings/98genome_absstat.htm), and sufficiently similar sequence pairs are added to the clusters. Unigene is available online at http://www.ncbi.nlm.nih.gov/UniGene with extra descriptions at http://www.ncbi.nlm.nih.gov/Schuler/Papers/ESTtransmap/

IMAGEne is focused on clustering only IMAGE clones of known genes (Cariaso et al., 1999). A special Perl script running at the pre-processing stage selects human IMAGE clones from all Genbank dbEST. Currently IMAGE clones comprise over 75% of all human dbEST sequences. The pre-processing stage also includes formatting of initial FASTA files into BLAST-searchable database with the *pressdb* utility. No repeat masking is reported in the paper, mentioning only the efforts to introduce one.

For the clustering procedure, IMAGEne uses a combination of BLAST and FASTA with wraparound scripts. Matches are detected by BLAST and extracted from the BLAST-searchable database. The speed of BLAST is balanced by the quality of FASTA, as only matches, confirmed by FASTA are accepted. The next stage of processing is alignment by SIM (http://globin.cse.psu.edu). A Perl script is used to compare each EST to its associated gene. These alignments are then constructed into a multiple alignment table, in which the known gene serves as a consensus sequence. Since IMAGEne is intended as a tool for re-arraying, its ability to pick the best clone is crucial. All clones within a cluster are sorted by preference; the highest one is considered the tentative candidate for Master Array. The factors affecting the preference are: coverage of the coding area, reliability rating of the library and the length of the clone. Results of IMAGEne processing are available for browsing and BLAST search via the web (http://www-bio.llnl.gov/imagene/bin/search).

BodyMap, a human and mouse gene expression database (Hishiki et al., 2000) is constructed from 3' EST data, produced locally at Osaka University and University of Tokyo. As of July 1999, BodyMap contained 159 429 human and 123 468 mouse ESTs. Before clustering sequences are rigorously pre-processed to maintain the quality of data. All sequences with >5% Ns, not starting with GATC or having more then one GATC are eliminated. Sequences, having >90% in an overlap longer then 50bp or 70% of the EST length with libraries of vector and ribosomal sequences are also eliminated. The lengths of ESTs in BodyMap are determined primarily by the location of *Mbo*I sites in the cDNAs, and those sequences with GATC site located within 20bp of the polyA tail are separated out because they are considered too short for gene identification. Finally, repetitive sequences are masked out after comparison to Repbase using BLAST. Clustering of the pre-processed ESTs is done based on FASTA search results with very high stringency. Two ESTs are considered belonging to one cluster only if they have >95% identity in an overlap longer then 50bp or 70% of the tag length and the overlap starts with GATC. Each cluster during the clustering process is represented by one sequence. Approximately 1000 sequences are selected from each library and compared to each other to form the "primary" clusters. All ESTs, similar enough to the representative sequences of the "primary" clusters form "secondary" clusters. As a result of the similarity searches, all clusters are indicated with their recurrence in all libraries in BodyMap, their identity in Genbank and their corresponding Unigene sequences. All data are stored in a relational database (Sybase). BodyMap serves as computerized multi-tissue mRNA. It can be searched by nucleotide sequence, keywords, Genebank accession numbers and Unigene IDs. BodyMap is accessible via the web at http://bodymap.ims.u-tokyo.ac.jp.

“An encyclopedia of mouse genes” (Marra et al., 1999) contains a clustered set of 295 053 mouse ESTs. Repeats are masked with RepeatMasker program (A. Smit, unpublished). Clustering is performed using BLAST2 (http://blast.wustl.edu) to compare all ESTs to each other. All similarities with P values better then 10^{-99} were evaluated to ensure they met 97% identity and >50bp overlap thresholds. Only those ESTs matching consistently with all cluster members were considered, making an encyclopedia of mouse genes an example of super-stringent clustering approach. Then the EST clusters were compared to known mouse mRNA sequences using the same BLAST2, as was used for comparison between ESTs. As about 39% of the original ESTs were from libraries, prepared from embryonic tissue and 59% of original ESTs were from later stages of development, the resulting database offers a broad overview of genes, differentially expressed in ontogenesis. The database has also been used for massive inter-genomic comparison to *Homo sapiens* and *Caenorhabditis elegans*, both at nucleotide and protein level.

EbEST is designed to automate EST-bases analysis on uncharacterized human genomic sequences (Jiang and Jacob, 1998). It aims at facilitating gene discovery by extracting as much gene structure information as possible from ESTs. The principal part of EbEST is EST clustering, which serves mainly to reduce the redundancy by separating all ESTs into non-overlapping clusters. Pre-processing includes masking repeats with RepeatMasker (Smit, unpublished). Initial matches between ESTs are detected by BLAST. The stringency is empirically determined, >90% identity and over 100bp overlap or >95% identity and more then 60bp overlap. Following gapped alignment is done with CrossMatch, which is based on modified Smith-Waterman algorithm (P. Green, unpublished). EbEST is currently provided as a web server (http://EbEST.ifrc.mcw.edu)

1.10.3 Purpose-built alignment based clustering methods

The field has recently seen the emergence of many new algorithms in development, but dedicated production algorithms are still few. We will not attempt at this time, to review the entirety, as there is a growing level of new material that has been very recently appearing.

ICA tools were developed as part of the UK human genome-mapping project, (Alwen, 1990). This group of algorithms was one of the first to become available, and has been used at major sequencing centers and is useful for data reduction. According to Jeremy Parsons, the ICA tools are a set of programs that could be of use to anyone doing medium-to-large scale DNA sequencing projects. The system has several tools but was originally designed for database redundancy and adapted for genomic fragments and finally EST fragments The ICAtools is a specialized hashed-query sequence clustering package. ICAass (one of the latest ICAtools - http://www.ebi.ac.uk/~jparsons) uses a FASTA-like search but uses an asymmetric scoring scheme designed to measure redundancy, rather than directly discover overlaps. *ICAss* uses a BLASTN type of algorithm to perform 'database pruning' to assess whether one sequence is a sub-set of another. *N2tool constitutes* a dedicated clustering tool that relies on indexing. It uses an indexed file format and local alignment to compare all the submitted sequences with each other to find those, which share any region of similarity. *ICAtool* indexes DNA sequences into clusters, which share local sequence similarities. *ICAass* takes a size-sorted (longest first) file of sequences and searches for those sequences that are approximately repeated within the length of another. *ICAmatches* attempts to explain why sequences have been clustered together by using novel sequence alignment. *N2tool* has been utilized in the Washington University Merck EST manufacture, for identification of artifactual sequences. A new generation of these tools is being developed and employed at EBI, and employs CORBA and Java for database interrogation (http://corba.ebi.ac.uk/EST/jesam/jesam_paper.html)just (Parsons and Rodriguez-Tome, 2000). All code is ANSI C, runs on many UNIX variants (and ported to MacOS), and is free to academics and industry. Memory usage scales linearly with database size, but computation time scales quadratically.

CLEANUP (Grillo et al, 1996) is similar to the ICAtools but performs a unique hashed-query pair-wise sequence comparison, which makes it slower than ICAass. Query sequences are encoded as hashed oligos along with one base mutated versions of the oligos to enhance query sensitivity. The largest published data set: 2400 Drosophila sequences, was self-compared in 160 seconds. All code is written in C and publicly available.

1.10.4 Non-alignment based scoring methods

RAPID (Rapid analysis of Pre-Indexed Data structures; Miller et al., 1999 and http://www.bioinf.man.ac.uk/rapid/) uses oligo frequencies to alter the significance of oligo matches before scoring particular pair-wise matches. No clustering is offered but the program has been tested on the analogous problem of removing redundancy and contamination from DNA sequence databases. RAPID uses Receiver Operator Characteristic (ROC) curves to tune parameters such as oligo size to get database search results with maximal specificity to the query. Rapid is memory intensive but relatively fast.

D2-cluster (Hide et al., 1994, 1997) is a word multiplicity comparison method that utilizes an agglomerative algorithm that has been specifically developed for rapidly and accurately partitioning transcript databases into index classes by clustering ESTs and full-length sequences according to minimal linkage or “transitive closure” rules. Agglomerative clustering method means that every sequence begins in its own cluster and the final clustering is constructed through a series of mergers that may be described in terms of minimal linkage, sometimes called single linkage or “transitive closure". The term transitive closure refers to the property that any two sequences with a given level of similarity will be in the same cluster, hence A and B are in the same cluster even if they share no similarity but there exists a sequence C with enough similarity to both A and B. The criterion for joining clusters is the detection of two sequences that share a window of (Window_Size) bases that is (Stringency) percent or more identical. The only criterion for clustering is sequence overlap and source or annotation information is not used. To detect the overlap criterion we use the d2 algorithm and set parameters and threshold values as described in (Torney et al, 1990; Hide, et al, 1994; Wu et al, 1997). The initial and final state of the algorithm is a partition of the input sequences where each sequence is in a cluster and no sequence appears in more than one cluster. D2-cluster uses an approach of word matching within a window, together with a measure of the *multiplicity* (if any) of that word within a window. The principal concept is that it doesn't attempt an alignment, not even in a reduced form. The results of comparison are derived directly from the comparison of word composition (word identity and multiplicity) of 2 sequence windows. Thus, the algorithm can be significantly faster than BLAST. Speed comes with a price: to collect significant statistics, the fragments must be long enough (about 100 bp) and only very high similarities can be detected (above 90% identity within a window). D2-cluster is used to produce initial loose clusters in STACK clustering system. We have determined that results of d2_cluster alone are between 8% and 20% less fragmented than Unigene (Burke et al., 1999) and the STACK datasystem produces clean clusters that are 16% less fragmented than Unigene (Miller et al, 1999).

The ESTate system for EST clustering has been developed at HGMP (http://www.hgmp.ac.uk) by Guy Slater as part of his Ph.D. research project. ESTate includes a number of tools for EST analysis, 2 of them specifically developed for clustering. *Precluster* is a fast word-scoring program, employing VFSMs (Virtual Finite State Machines) and an efficient word-matching algorithm to enable all-against-all matching in of a group of sequences in sub-quadratic time. The scores generated are simply the number of matching words between sequences. The results (which may be large) are written out in a compressed binary format, and can later be used by for EST clustering by the program *estcluster*. *Estcluster* uses a graph theory approach to allow rapid generation of high quality clusters.

1.10.5 Pre-indexing methods

The size of the datasets has effectively precluded their use on workstations architectures. Indexing is one approach that allows for less computationally intense operations. In the field of exact string matching techniques have been developed that preprocess text in such a way that, upon searching a pattern, only a small parts of that text actually need to be explicitly accessed. Since in the clustering applications one imposes a very stringent match criterion, the hope is to draw on these techniques in order to further improve the efficiency of database searching algorithms. Candidates are sophisticated indexing data structures like suffix trees (McCreight, 1976), suffix arrays (Manber and Myers, 1993) or Patricia trees (Knuth, 1973), that allow performance of queries in time proportional to the length of the query string while being as independent as possible of the size of the searched text.

Quite a few attempts have been made to adapt these techniques to the similarity searches needed for biological purposes. Martinez (Martinez, 1983), for example, gave the first application of a position tree in molecular biology. This data structure requires about 16 times the space needed to store the original data which clearly may create serious problems when applied to large data collections. The first sub-linear expected time algorithms were proposed by Chang and Lawler (Chang and Lawler, 1994) and Ukkonen (Ukkonen, 1992). They use a suffix tree of the query sequence but still scan the database.

Myers (Myers, 1994) suggested a sub-linear (in the database size) search algorithm that is centered on an index built on the database sequences. The query sequence is split into small subsequences and words from a neighborhood of these are located in the index. The IBM product FLASH (Califano and Rigoutsos, 1993) takes advantage of that large "probabilistic" index where not all k-tuples are considered, but only some of them are randomly chosen. They report a 18GB index for a 100 million-residue database, which makes such an approach impractical for large databases.

A few publicly available tools are currently in development and will be available in the near future. For instance, 'QUASAR' was announced at RECOMB99(Burkhardt, et al 1999). It is termed ' Q-gramm Alignment based on suffix arrays'. This algorithm is designed to quickly detect sequences with high similarity to the query in a context where many searches are conducted on one database. The database is presented in the pre-processed (indexed) form and similarity detection is based on the exact matching of short sub-strings (q-gramms) (Burkhardt, et al 1999).

The speed of sequence comparison comes with a price. Pre-indexed comparison relies heavily on the principle that similar sequences are built of similar oligonucleotides. This is generally true, but makes this comparison very imprecise. All indexing-based clustering algorithms without exception include a thorough pair-wise alignment on some stage. QUASAR, for example, employs slightly modified publicly available code from NCBI tools package (Burkhardt, personal communication). All pre-indexing systems try to find the optimal balance between speedy rough comparison and reliable, but slow "conventional" alignment tools. On the other hand, sub-linear computation complexity, promised by pre-indexing, is the most promising way to make a breakthrough in performance. This addition is not necessarily an alternative; it can be used as an additional module in existing systems. Most probably all of them will include some form of pre-indexing in the years to come.

Chapter 2

Challenges of EST clustering and choice of strategy

Recently, the growth rate of sequence data in genomic databases has already surpassed the growth rate of computer performance, mostly due to introduction of high throughput sequencing technologies like EST manufacture and to a multiplicity of genome sequencing projects. So far the existing clustered EST databases manage to cope with the deluge of data, but only with a help of high-end computer facilities. Projection of this situation into the future suggests that the progress of computer hardware design may not be sufficient. New algorithms are required to mitigate this problem.

2.1 Computational bottlenecks

The whole process of EST clustering is staged. The stages as they are found in contemporary clustering systems are reviewed in Chapter 1 (Figure 2). Let's make a brief review of this process from the computational point of view:

2.1.1 Preprocessing

Data apprehension, reformatting, quality assessment and temporal storage don't require high computational power. Big data sets may demand high volume storage facilities and fast disk systems. Repeat and vector masking, typically included in this stage, requires much more computation. To perform masking at least n_{rv} times N_{est} acts of sequence comparison are required, where n_{rv} is a number of samples in vector and repeat data sets and N_{est} is a number of ESTs to be masked. Unlike the EST data, the number of known repeats and cloning vectors, used in sequencing projects is relatively stable and grows slowly. Computational complexity of masking grows linearly with the number of ESTs. The problem is compounded by the growing number of 'novel' genomes for which repeats have not yet been characterized.

2.1.2 Initial clustering

Clustering requires a number of inter-sequence comparisons, estimated as N_{est} times N_{est}. Even though pre-indexing algorithms allow $\mathrm{N}^2_{\mathrm{est}}$ comparisons to be performed as a sub-quadratic, this stage remains the most computationally expensive. In the case of linear- and super-linear sequence comparison algorithms (based on some form of pair-wise alignment) the time required to complete N^2_{est} comparisons depends quadratically on the total length of the initial EST data set. Pre-indexing methods rely on word comparison and require less then $c = k\,N^2_{bp}$ time. In any case N^2_{est} inter-object (EST) distance measures are required for cluster analysis.

2.1.3 Assembly

Assembly is performed on a larger number of much smaller data sets partitioned from the initial data on the initial clustering stage. Computation complexity of cluster assembly stage can be estimated as

$c = k\sum_{i=2}^{m}\left(n_{clu}\right)_i i^2$, where n_{clu} is number of clusters consisting of i ESTs.

Each cluster assembly requires multiple applications of computationally expensive precise alignments, but the small size of clusters compared to the initial number of ESTs makes this computation relatively fast. Typically, after initial clustering only up to 30% of all initial ESTs are assigned to clusters, while most of initial ESTs remain singletons (see Fig.14 for example). Singletons are automatically excluded from cluster assembly. Less stringent initial clustering produces bigger clusters, but has a high level of false positive results. Cluster assembly can easily rectify this, but the time required to complete this stage grows quadratically with cluster size. Each cluster can be processed independently from the others; hence, this stage is easily optimized for parallel processing but requires careful memory management.

2.1.4 Alignment processing

Alignment processing doesn't involve heavy computations. It merely processes the results (usually text files), produced on the earlier stages.

2.1.5 Cluster joining and Output

Neither joining of clusters nor output involves significant computations. Cluster joining is performed according to the information contained in annotation, for example clone ID. This involves only single-read text processing.

Although the layout of stages may vary in different clustering systems, the most computationally challenging step is one that includes the highest number of inter-sequence comparisons. Hence, the most significant improvement to the clustering performance can be done if the algorithm of pair-wise sequence comparison is improved.

2.2 Choice of strategy for clustering system

Many of the systems reviewed in Chapter 1 are based on the purpose-built algorithms, which is a clear indication that sequence comparison is considered as a bottleneck of EST clustering by many developers.

Some latest developments in the EST clustering systems give preference to fast sub-linear algorithms. The sheer speed of sequence comparison is the obvious advantage of this family of algorithms. There are also some disadvantages. First of all, the speed comes at a price of precision. A higher false-positive rate is compensated by a thorough alignment at the subsequent cluster assembly stage. False-negative cases, once dropped, are hard to detect. To cut the probable loss of matching sequence pairs, clustering programs have to be tuned to higher sensitivity. As a result, the slow cluster assembly stage gets overloaded with sequences that can't be aligned (i.e. False-positives). Many fast sub-linear algorithms require an extensive pre-clustering procedure. Apart from being resource consuming, this stage is also a difficult parallel optimization. Pre-calculated index tables need to be adjusted with each update of the initial data set. A clustering system, built around a fast, but imprecise sub-linear algorithm would require lots of compensatory mechanisms and additional routines. Complex structure obstacles further development of the clustering system as a whole.

In spite of the seemingly obvious choice of the fastest possible algorithm we have chosen another strategy. Although the linear class algorithms are generally slower then sub-linear, they have some advantages as a basis of an EST clustering system. A more precise algorithm would produce much less false-positive and false-negative results in the first place. Sensitivity to small regions of local similarity can improve quality of results by detecting short, but non-random overlaps between EST fragments and reach much longer resulting consensi. Detecting small similarity regions, even if accompanied by much longer non-similar stretches, is crucial for detecting alternative gene variants.

The best algorithm to serve the purpose of EST clustering would be an algorithm that:

- Is faster than contemporarily alignment searching algorithms;
- More precise that contemporary work-scoring algorithms;
- Sensitive to short areas of similarity;
- Produces a metric that can be used as inter-object distance measure for clustering;
- Has a good potential for future development, like parallel optimization.

The next Chapter 3 describes the new sequence comparison algorithm, developed to meet the specific demands of EST clustering.

Chapter 3.

Linear algorithm for sequence comparison.

3.1 General Idea

The general idea of the proposed fast linear algorithm is based on the following observation: Suppose we have a query polymeric sequence (DNA, for example) S_q. We are to compare this query sequence to a sequence S_b – say, taken from a data bank – to reveal their possible similarity. Suppose we know all overlapping subsequences (words) of a certain length, which are found in S_q and S_b. There is a non-zero probability that a particular word, found in S_q, also happens in S_b. If sequences S_q and S_b are similar, the probability of a word to be found in both sequences is higher, otherwise it's lower. This is the fundamental idea of oligo-based sequence comparison algorithms. But coincidence of words is a weak overall measure of similarity. Clearly, the more coincident words between two sequences, the more probable the similarity of the two sequences. Finally, some algorithms rely partly on multiple word coincidence (like one used in D2_CLUSTER) (Torney et al., 1990).

If we compare two sequences of different length, the probability of finding a coincident word is different, i.e. in a short sequence $S_{b,}$ each word coinciding with a word from S_q, makes a bigger contribution towards a decision that two sequences are similar. In an extreme case, if the length of S_b is equal to the word length, single (and the only) coincident word is enough to say that S_q has a 100% local similarity to S_b. Ideally, the shorter both sequences are, the more informative their word coincidence is.

But, in case of single-word or a few-word sequence S_b, there is no limitation on the size of S_q.

Words, coincident between S_b and S_q, and found within a short range of each other are presumed to be more informative. To consider only such words we introduce a short frame sliding by one position along S_b. Then we compare the words found in S_q to the words found in each frame of S_b. The frame containing no or small number of coinciding words is most probably unrelated to S_q. Naturally, the frame containing many or even all words coinciding to S_q most probably indicates a region of local similarity.

This assumption is true only in cases where only a fraction of all possible words are realized in S_q. Otherwise if all or almost all possible variants were found in S_q, any word, found in any frame of S_b would be coincident and their input would be non-informative. Thus, there is some limitation on the length of S_q and, correspondingly, the length of words, used in comparison, so that words remain reasonably rare.

Finally, we can conclude, that a high number of words, coincident between S_q and a short frame in S_b, indicate a probable zone of local similarity, providing that the words are informative (not too abundant).

3.2 Weighted hash-table.

One of the most appropriate ways to store and access the information about words of definite length found in a sequence (S_q) is by use of a hash table (H). A hash table is a two-dimensional rectangular table with number of rows equal to the number of all possible oligos on length n, and two columns. For example, for 8-tuples (words of 8 letters), the number of rows would be 65536. One of the columns contains a number (hash value) of a word; the other column bears the value that corresponds to that word.

Numbering of the oligos is done the simplest way. Each letter of the 4-letter code is numbered, A=0, T=1, G=2, C=3. Then each word of length n is presented as a base4 number. To obtain the hash value of the word we only need to convert base4 to a decimal number.

In fact, as long as all hash values are placed in ascending order, the hash table H is realized as a linear array, where the offset itself is a hash value. Each element of this array thus contains a value, associated with a corresponding oligonucleotide.

Initially we fill values corresponding to words in the table H with estimated frequencies of the words, found in S_q.

$H_i=N_i/LS_q$

N_i – number of words with hash value i, found in S_q;

LS_q – length of S_q

Then all values of the hash table H are differentially weighted. The purpose of this weighting is to assign different weight to the words with different information content. Obviously, some words, like those containing just repeats of one letter, give us much less information than those of greater complexity. Words abundant in a sequence S_q are likely to be less informative than words found only once or a few times in a sequence. Also, words scattered along the query sequence S_q in a more or less uniform way can tell us much less about local similarity, then those found in some parts of S_q only.

Various weighting can be applied in different implementations of this algorithm. In an implementation used for EST clustering we limited weighting down to only 2 criteria:

a) All words, composed of only one letter, found in S_q, assigned 0 weight.

b) Weights of all words, found in Sq more than 5 times are lowered. The degree may vary, but in this work empirically selected division by 2 was applied.

After a weighting procedure, we have an initial hash table, H, transformed to a weighted hash table H_w that is used further in sequence comparison.

3.3 Similarity function profile

To perform an initial comparison of sequence S_b to sequence S_q, we apply a short frame W to S_b, starting from the first letter. Each word of length n, found in W is compared to the hash table H_w and the results are summarized, i.e. add weights from H_w, corresponding to hash values of words, found in W:

$$F(W)=\sum_{i=0,w}^{w}(H_{wi})$$

w - width(number of words) of frame W

Then we slide the frame W repeatedly along S_b by one letter at a time, calculating a similarity function $F(W_i)$ for each frame Wi. As a result, we have a profile of an heuristic local similarity function $F(W_i)$ (Fig. 6). Higher values of the local similarity function correspond to regions with higher probability of local similarity (more informative coincident words), while lower values correspond to regions with less coinciding words or less informative coincident words, hence less probability of local similarity.

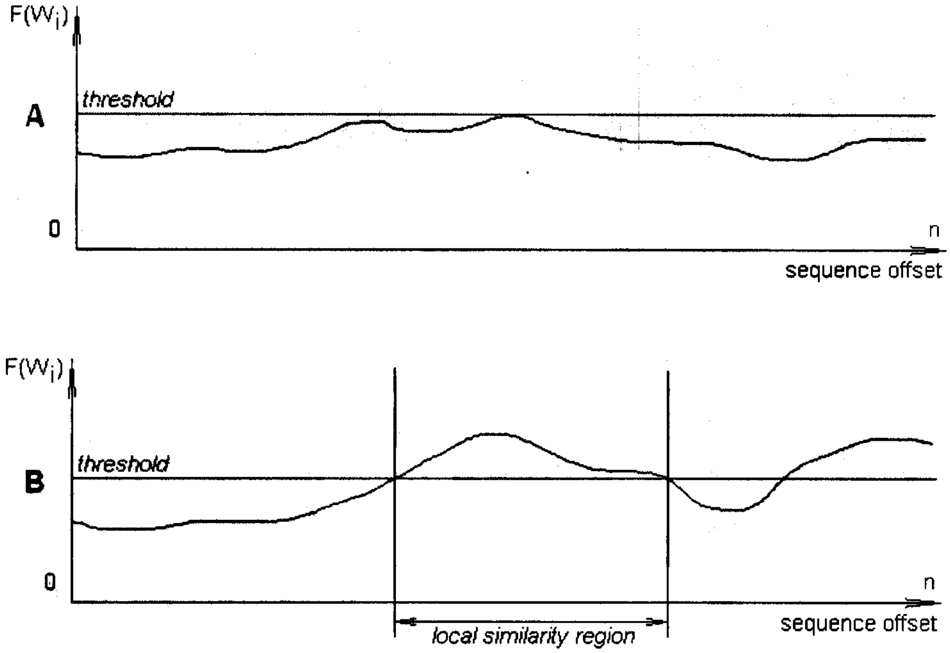

Figure 6. Example of a local similarity function profile along the test sequence. **A** shows a similarity function for two unrelated sequences; **B** shows an example of a similarity function for two sequences with local similarity region.

3.4 Local similarity function value distribution

Generally, the profile of the heuristic local similarity function $F(W_i)$ is sufficient to detect local similarity regions. To do it, we only need to establish a threshold value for $F(W_i)$, as it proposed in the original algorithm (Strelets et al., 1994). This approach has a few significant drawbacks, especially if applied in a sequence-clustering context. For instance, it's hard to estimate a general similarity between sequences. One single figure defining a distance between two given sequences is essential for EST clustering. The original algorithm detects only short regions within 2 sequences, which might be similar. There is no defined relationship between the number of potential similarity regions where the similarity function exceeds the threshold or the value of the similarity function above the threshold and the similarity score. Selection of a threshold value in the original algorithm is also empirical. For example, we can compare 2 nucleotide sequences and find that there are 3 regions where the similarity function value is above the selected threshold. This discovery still leaves open the question whether the sequences are related, i.e. their similarity can be confirmed by alignment with certain percentage of identity. How many local similarity regions are enough to put two sequences in one cluster? How long should these regions be and by how much should they exceed the threshold? Can the probability of making the right choice (in other words the probability of missing the existing similarity or detecting the similarity of unrelated pair of sequences) be estimated?

To overcome these difficulties we propose to transform the similarity function $F(W_i)$ values to a distribution form (DF). This distribution can be easily calculated in categorized form, with the number of base pairs within a frame W taken as a number of categories (Fig. 7). Then, using this approach we can compare the distribution obtained in a sequence comparison experiment (DF_{ex}) with some general distribution DF_g containing the expected values of the similarity function $F(W_i)$. Probabilities of sequence similarity and selection threshold can be estimated based on the proximity of those distributions.

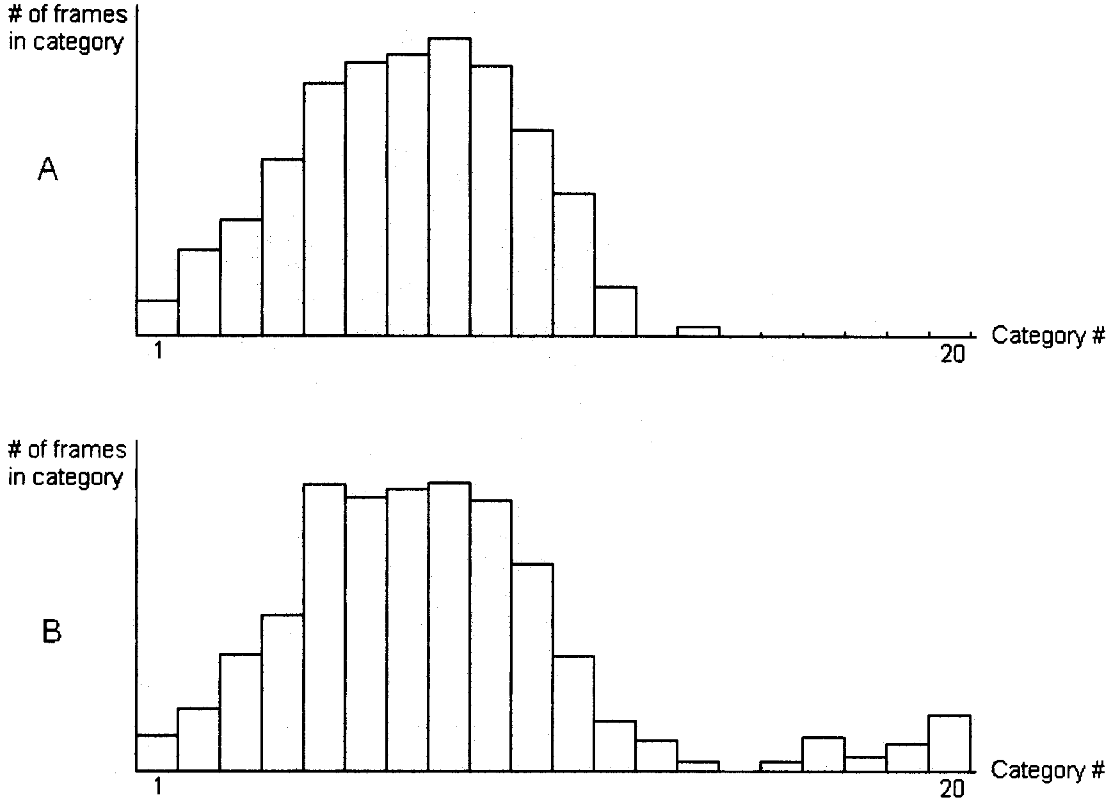

Figure 7. Examples of categorized distribution (DF_{ex}) of similarity function $F(W_i)$ values in case of unrelated sequences (**A**) and sequences with a similarity region (**B**).

As a first step we need to estimate the general distribution. For this purpose we have conducted the following experiment. A sequence was taken from dbEST database. The sequence is chosen randomly, the only criterion used is that it should represent a typical EST, i.e. have a typical length of 300-400 base pairs and have no long single letter stretches. Such stretches, for example resulting from repeat masking can't affect the local similarity function, because all single letter words are assigned 0 weight on the preprocessing stage. Although not involved in the similarity search process, single letter stretches may reduce the effective sequence length. For each selected sequence a random counterpart was generated using a simple technique (i.e. keeping the ratio of base pairs, found in the original sequence). Then the original sequence was compared and the distribution DF_{ex} obtained. This process was iterated 100000 times. Then the DF_g was estimated as

$$DF_g(i)= med(DF_{ex}(i))$$

Once the general expected distribution for non-similar sequences has been estimated, the same distribution for sequences, containing a region of significant local similarity (DF_{gs}) was produced. This distribution was generated by conducting the same experiment as it was done previously for DF_g, but with one difference. After we generate a random counterpart for a sequence, we introduce a short local similarity region by copying a contiguous stretch of n_r base pairs at a random location from one sequence to another. The length of this local similarity patch and the number of possible mismatches may vary in accordance to the sensitivity of the sequence comparison we'd like to achieve. To select a threshold specific for EST clustering we've introduced local similarity patches of length 40 base pairs with no mismatches.

3.5 Weight factors and general similarity index.

Having two sets of experimental data, one set containing no signal (local similarity region) and the other set, containing the signal, we have a classic situation of two contrasting sets for discriminant analysis. Discriminant analysis is used to determine which variables discriminate between two or more naturally occurring groups. In this work we've generated 100 000 objects in each set. As long as we have experimental distributions $DF_g(i)$ and $DF_{gs}(i)$ already in categorized form, we can compare them as Euclidean vectors of size w. To convert variation of different categories of DF_g and DF_{gs} to one scale we normalize it by standard deviation:

$$DF(i)_{norm} = DF(i)/\sigma(DF(i))$$

Then we stretch an imaginary line through the centers of the two contrasting sets. The general expected distributions DF_g and DF_{gs} can be taken as the centers of those two sets. We project all the objects of each set to this imaginary line by a simple formula:

$$X(i) = \frac{\sum_{i=0}^{w} DF(i) * (DF_g(i) - DF_{gs}(i))}{\sqrt{\sum_{i=0}^{w} (DF_g(i) - DF_{gs}(i))}}$$

Having projections of all objects of both sets on the single line we can estimate and plot the distribution of probability of X(i) along the line. For $X_g(i)$ and $X_{gs}(i)$ we have two partly overlapping distributions (Fig. 8). The degree of overlapping depends on the conditions of experiment, like the length of oligonucleotide chosen or the size of introduced similarity region. Given two distributions, we can select the threshold value, which separates them. We can also make a rough estimation of the expected numbers of false positive and false negative results, evaluating the relative parts of distributions $X_g(i)$ and $X_{gs}(i)$, that appear on the opposite sides of the threshold. In this work we have selected the threshold to make the algorithm more sensitive. For the threshold we've taken the leftmost value of 100 000, obtained in the experiment. Nevertheless, the number of false-positive values (cases, having no introduced similarity region, but having projection value above the selected threshold) was reasonably low – under 4% of the total number of negative cases (100 000).

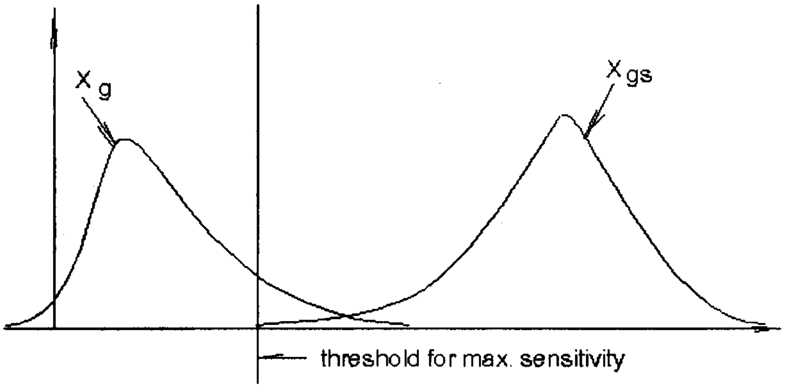

Figure 8. Partly overlapping distributions X_g (generalized for non-similar sequences) and X_{gs} (generalized for sequences with introduced similarity region).

The projections of obtained distributions are single-figure measures, shown to be enough to characterize similarity of a given pair of sequences. In our further work we use such a projection as a general similarity index. The linear coefficients in the equation of the line, coming through the points DF_g and DF_{gs} can be used as weight factors. We only need to take into account the scaling factors, i.e. standard deviation values we applied to the data to equalize variation scales in different categories. In this adjusted form the weight coefficients can be applied to any new distribution DF(W_i) in order to calculate the general similarity index, which can be compared to the threshold.

Statistic analysis of the simulated data proves the estimation, given on Figure 8. We don't know exactly the canonic form of neither X_g nor X_{gs}. High values of skewness and kurtosis suggest it's different from normal. High kurtosis is even good for our purpose, as those two distributions have higher peaks and tend to have very thin overlapping ends. Table 1 shows some basic descriptive statistic for the distributions X_g and X_{gs}.

	Median	Mean	Std. Dev.	Skewness	Kurtosis
Xg	0.038	0.046	0.036	6.119	47.677
Xgs	0.366	0.429	0.175	3.493	14.392

Table 1. Descriptive statistics for generalized distributions X_g and X_{gs}, calculated from the 1000 simulations.

The purpose of generation of distribution X_g and X_{gs} is to learn to distinguish the samples that belong to one of them. From this point of view it is more significant to know how much these distribution overlap, regardless of the distribution type. Figure 9 illustrates the degree of overlapping for two sets of simulated data (with and without similarity region).

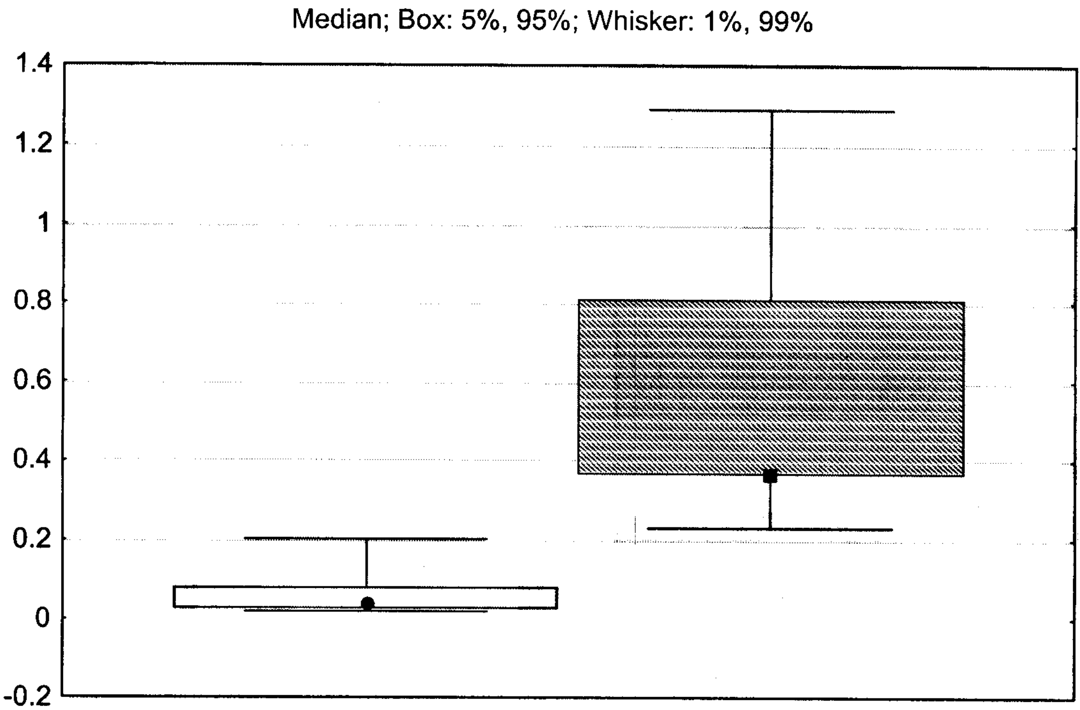

Figure 9. Box and whiskers plot of simulated Xg (left) and Xgs (right) distributions, 1000 samples each. Medians are plotted as dots, boxes contain 90% of the samples. Two sets overlap by less then 1% of each (1% margins are plotted as whiskers).

3.6 Summary of the algorithm

Description of the algorithm development and the code, implemented to perform experiments and produce pre-calculated linear coefficients are quite extensive (over 10 pages of text and over a 1000 lines of C code correspondingly). Nevertheless when comparing two sequences, the algorithm is very compact and can be described in a few steps:

- Take a query sequence
- Calculate query sequence hash table
- Process hash table to adjust the weights

- Slide a short frame along the test sequence and calculate the distribution of local similarity function values
- Calculate the general similarity index as scalar multiplication of local similarity function distribution and pre-calculated weight factors
- Compare the general similarity index to the threshold value

3.7 Sensitivity and computational complexity estimation

There are a number of factors that affect the sensitivity of the algorithm. First of all, it depends on the word length selected for analysis. But the dependence doesn't conform the general rule of FASTA and BLAST (Pearson and Lipman, 1988, Altschul et al., 1990) - the shorter word, the higher sensitivity. Using relatively long words (of length 8 bp or more) makes the algorithm less sensitive to the similarity regions broken by single nucleotide substitutions because each substitution can change up to 8 oligonucleotides. Such regions may be very similar when aligned, but share little or no common 8-tuples. This problem is general for the algorithms relying on fast word comparison. But, unlike alignment algorithms, this algorithm also loses sensitivity when words that are too short are selected. Shorter words are more abundant - hence the probability of finding them within a short distance for stochastic reason is also higher. Using shorter words reduces the signal to noise ratio and increases the number of false-positive cases. Using the weighted hash table can help the problem. This is another factor affecting the algorithm sensitivity. Using more sophisticated weighting scheme, selecting most informative words may provide a reasonable signal to noise ratio even with the shorter word length (5-6 bp), which reduces substitution intolerance.

Another way to increase sensitivity to weak similarities is to consider imperfect matches. This would mean that if no perfectly matching words are found between the frame (W) and the weighted hash table (H_w), the words, different by one or more letters could also contribute to the similarity function (F_w). With this feature introduced this algorithm could be as sensitive as BLAST, which also counts the imperfect word matches. But for quick detection of matching ESTs this excess sensitivity is not required, while additional calculations may slow down the process. For the sake of speed this possible extension of the algorithm is not implemented.

The sensitivity of the algorithm is has thus been pre-determined at the development phase. The algorithm uses weight factors and a threshold value pre-calculated in experiments with model sequences with and without introduced local similarity area. By varying parameters of the experiments, like similarity area length and number of mismatches, we can develop sets of weight factors, tuned for different levels of sensitivity. Particularly, for use in an EST clustering application, weight factors were optimized to detect overlapping regions of at least 40 base pairs with no deletions and up to 2 mismatches (95% identity). This stringency level was chosen based on the experience of other EST clustering systems (See Chapter 1.7). Selection of a threshold value can serve the same purpose. Using a particular combination of weight factors and threshold gives us the ability to estimate sensitivity in terms like "we are able to recognize sequence pairs, having similarity regions of length X, containing up to Y possible mismatches with about N% of false positive and M% of false negative".

This algorithm, like its prototype, also has a linear computation complexity in cases where one sequence is compared against many i.e. the computation time depends linearly on the database size. This is a typical situation for any massive database search. The query sequence is processed only once. The same hash table (H_w) can be reused many times. This makes the general computation complexity dependent on the database size only.

Sliding a short frame by one position through the tested sequence is also very inexpensive from the computational point of view. It requires only a few more operations then required for hashing all the words in a sequence. Calculation of a local similarity function adds very little, since it takes only few simple integer arithmetic operations for each position of a sliding frame. The algorithm can be implemented very effectively using a sliding frame technique. Only the effect of incoming and leaving positions for each frame shift is calculated. Consequently, the speed of this algorithm doesn't depend on the length of words used for comparison.

3.8 Algorithm parameters and evaluation

The algorithm was developed and optimized specifically for certain types of applications. First of all it should provide a sequence comparison mechanism for an EST clustering project. In some sense comparing ESTs to each other is easier than many other cases of similarity search. We only need to select pairs that are in fact identical or almost identical copies of each other or overlapping fragments of the same sequence. Excessive sensitivity may bring to one cluster some evolutionary or functionally related sequences instead of copies of the same sequence. To establish a sensitivity landmark we've taken the results of D2_cluster program (Hide et al, 1994). By the properties of the sequence comparison algorithm, used in D2, it recognizes sequences having over 96% identity in 150bp span. The algorithm we develop was targeted to get the results at least not worse then those achieved by D2_cluster.

The weight factors were calculated on the generated training contrast sets of size 100 000 each. Real ESTs were randomly picked from dbEST for the initial sequence. The non-similar counterpart sequence was generated randomly with the same base pair ratio as in the original sequence. For the set, containing positive signal, a region of 40 base pairs was copied from the original sequence to it's non-similar counterpart at random location, starting from 0 to n-40 (where n is the length of the sequence). Two single nucleotide mismatches were introduced within the similarity region.

The length of the frame W is the principal parameter that has to be optimized for the best performance. The whole algorithm development is largely heuristic and there is no formula to designate a connection between frame length and sensitivity. Nevertheless, the optimal frame length for detection a given type of local similarity region (40 base pairs) can be selected by a Monte-Carlo approach, as it was done for pre-calculated weight factors (see 3.4 and 3.5). To perform this experiment we selected a measure to assess the quality of resolution between two generalized distributions X_g and X_{gs}, obtained in a computer experiment with a given length of frame W. This measure doesn't need to be absolute, but it must distinguish less overlapping distributions from more overlapping. We used the percentage of objects, found in the overlapping part of distributions X_g and X_{gs} as a measure of resolution quality. All frame lengths from 16 to 40 base pairs were repeatedly tested and the best result for each value is saved, along with the corresponding set of weight factors. The best results achieved with the frame length 22 (oligonucleotide length 6) and this frame length was further used in EST clustering applications.

3.9 Implementation

The algorithm is implemented in a form of a set of subroutines and data structures, assembled in a range of applications. All development was done with a standard C and primary implementation is also done in C (ANSI standard). The code is written in order to provide both effectiveness and portability. For the sake of portability potentially platform-dependent operations (like bit-field operations) were avoided. To utilize this code in EST clustering application C++ classes were developed to wrap around the basic C code.

3.10 Applications

A number of applications have been developed to take advantages of the algorithm:

3.10.1 Search for similar sequences in a databank

The method is applicable to searching of databases for near identical or identical matches. This implementation strategy can take advantage of the new algorithm's speed. Sensitivity of the algorithm can be a problem in the cases of distantly related sequences, especially if mutations are distributed evenly, leaving no conservative regions (see 3.7). This problem can be solved by a compensatory mechanism at the application level. The method can make use of two algorithms: it performs the preliminary search with the new fast algorithm and then qualifies the selected sequences by a thorough alignment (Strelets et al., 1992). The new implementation follows the same strategy, only enhanced by introduction of the new sequence comparison algoritm.

3.10.2 Masking repetitive and vector sequences from the data set

A masking implementation takes two data sets as initial data, a set of raw sequences and a set of the "junk" sequences i.e. cloning vectors, repeats or other sequences, which should be replaced in the raw sequence by a neutral symbol. The resulting file contains the same sequences, as in initial file, where all occurrences of local similarity regions are substituted by 'X' letters. This application doesn't need a high sensitivity to weak similarities. Instead, only highly similar regions need to be revealed. Masking a set of sequences from the potential contamination by fragments of cloning vectors and repetitive sequences is a required procedure before similarity search or clustering of nucleotide sequences. In many cases, like EST clustering, the volume of data makes masking a tough job. High performance of the new algorithm makes it a good candidate for this purpose.

3.10.3 Clustering of ESTs.

Two applications, implementing two different clustering strategies are developed. These strategies were described in the ISMB99 tutorial on EST clustering (Hide et al. 1999) as “loose” and “stringent”. “Loose” clustering relies on third-party software for cluster assembly and consensus generation. This allows using very low stringency threshold on the clustering stage and achieving higher sensitivity. “Stringent” approach includes thorough alignment and consensus generation on the clustering stage to produce verified clusters in the first place. These implementations are discussed in Chapter 4.

Chapter 4

Application to Clustering

We have explored two different strategies in clustering, both "loose" and "stringent", as described above. Thus, we developed two separate applications for each of the strategies.

4.1 Stringent clustering

In this approach each original EST sequence read from input file is placed in a separate cluster. The process starts with a number of clusters equal to the number of initial ESTs, each cluster containing one sequence only. The clusters are represented by complex data structures. The principal data fields of these clusters include the list of the original ESTs, the "searchable" sequence and the consensus sequence (Fig. 10). The "searchable" sequence represents the cluster in comparison between clusters: when a cluster is a singleton it's a copy of the only member sequence, otherwise it can be either a copy of the consensus sequence or a concatenation of the non-redundant parts of the member sequences (in case of chimerical or alternatively processed sequences).

```
typedef struct{
   SEQUENCE *searchable_seq;
   SEQUENCE *searchable_seq_complement;
   OLIGOTABLE *table;
   OLIGOTABLE *table_complement;

   struct SEQLIST *list;
   struct SEQLIST *current;
   int number;
   float threshold;
}CLUSTER;

struct CLUSTERLIST{
   struct CLUSTERLIST *prev;
   struct CLUSTERLIST *next;
   CLUSTER c;
};
```

Figure 10. The principal data structures of the stringent clustering program.

All clusters are arranged in a bi-directional dynamic list. All members of the list are repeatedly compared against each other. When clusters are compared, only "searchable" sequences are used. Comparison is performed by application of the fast linear algorithm described earlier in this work. Each time a match is found two clusters are merged, their member lists are concatenated and the "searchable" sequence is renewed to make better representation for all members. In an ideal situation this sequence is simply the consensus, but in the case of ESTs it's often quite different. To produce both "searchable" and consensus sequence a pair-wise alignment procedure is used. Currently the C code from (Strelets et al., 1992) is used. Each cluster in the dynamic list is compared against the rest of the list in a cycle. This cycle is repeated until no match is found for any of the clusters. Because the list of clusters shrinks with every cluster merge, the main cycle shortens and the clustering process accelerates.

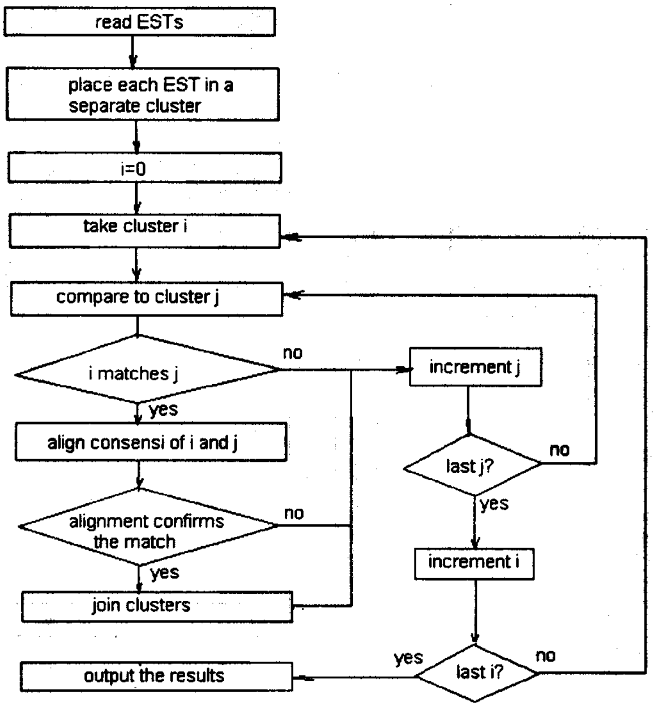

Figure 11. Basic algorithms of the stringent clustering program

The stringent clustering strategy has certain advantages as well as complications. First of all, it produces assembled clusters with consensi and doesn't require further computationally extensive cluster assembly stage. With a typical occurrence of matching sequences in big EST datasets this method is also faster: it accelerates with every match found. But the main advantage of this approach is scalability. The whole set of initial EST data could in principal be divided onto any number of small datasets and processed separately and asynchronously on separate CPUs. After the clustering process in each subset is completed, they could be merged and the process repeated on the merged sets to pick up possible matches between subsets. To optimize this parallel processing, the subset size should be chosen as large enough to contain a reasonable number of matches in order to benefit from data shrinkage during the clustering and small enough to be processed by a given machine in a reasonable time. The details of this approach have not been explored, but the implementation would be amenable to a mixed or single architecture environment.

4.2 Loose clustering

In this approach we read all initial EST sequences and place each one in a separate cluster. All clusters are arranged in a dynamic list and represented by a data structure, similar to that of stringent clustering described above. Then every cluster is compared against every other and the results of comparison are stored in a form of a match list, where every record contains numbers of sequences in a matching pair. Once comparison is done, clusters, or connectivity groups are extracted from the pair list by a recursive procedure. This approach is practically the same as that of D2_cluster, which is currently used in production of the STACK database. This approach also has a number of complications - it requires subsequent cluster assembly, it's slower than the "stringent" implementation and takes more effort for parallel optimization, to name a few. But the main advantage is that it produces results, very similar to those of D2_cluster, thus it's very easy to quantitate and qualify in terms of already established criteria. As such, the algorithm could be quickly implemented in existing systems such as the STACK_pack system, effectively, a plug-in to replace D2-cluster. Thus an improvement in clustering could be performed with relatively little expense in terms of extra development.

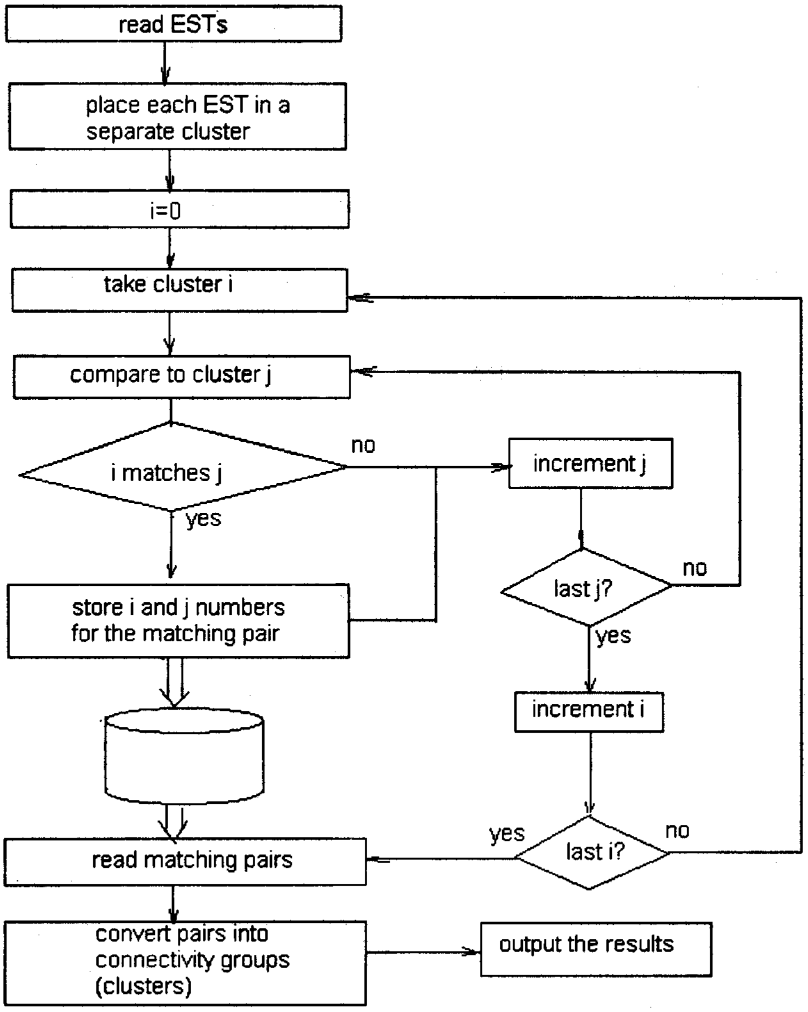

Figure 12. Basic algorithm for loose clustering program

4.3 Parameter space evaluation

Most of the parameters for clustering have been determined statistically during development of the sequence comparison algorithm (see 3.8). The only parameter which remains to be set for clustering application is the threshold value for the general similarity index. This parameter affects the stringency of clustering. A higher threshold value results in more stringent clustering with fewer and smaller clusters and higher

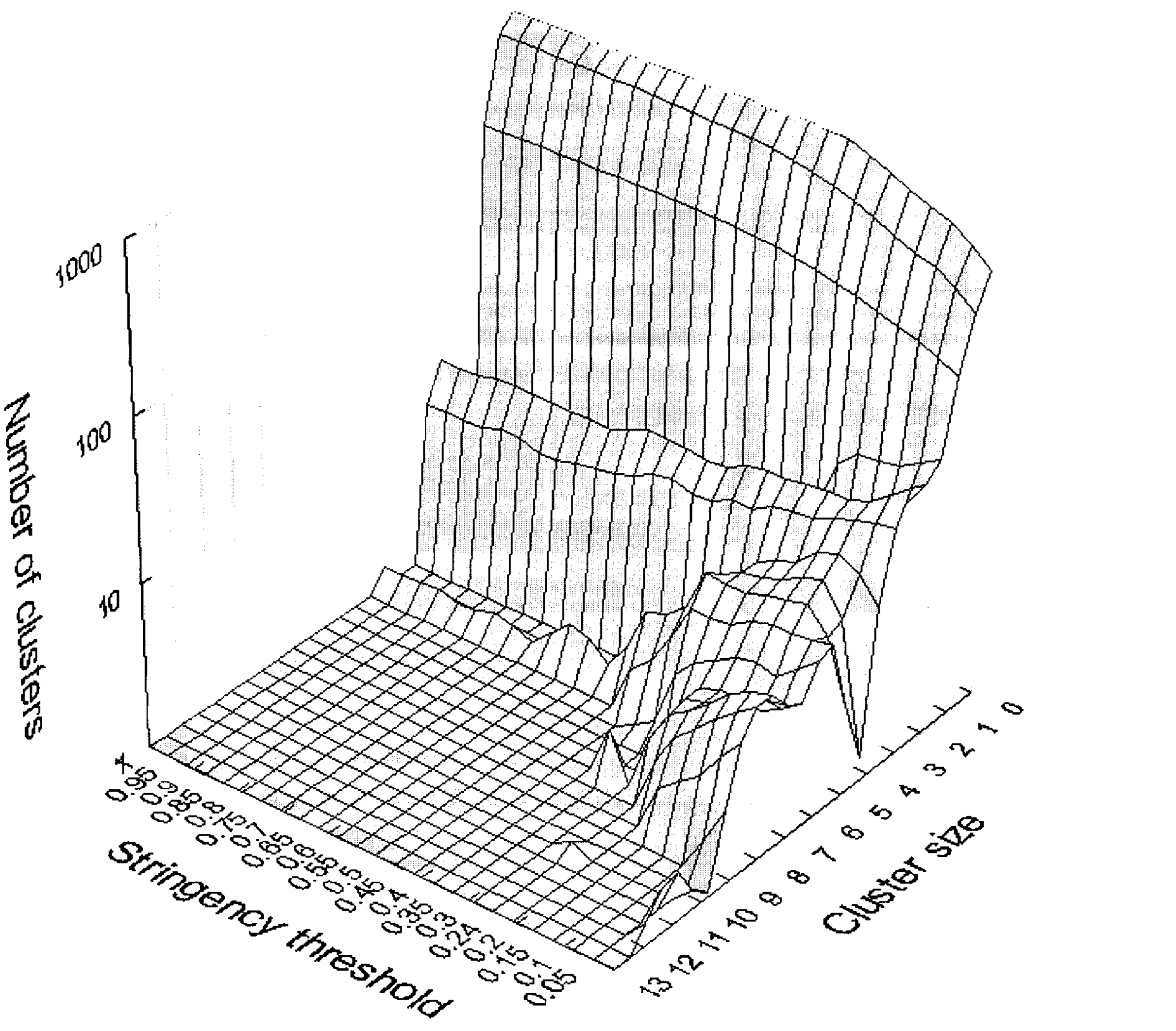

Figure 13 Effect of stringency threshold variation on the clustering results. The only free parameter in both loose and stringent clustering applications is the stringency threshold. This picture shows distribution of cluster size (axis X – categories by cluster size from 1 to 13, axis Z – number of clusters of certain size, decimal logarithmic scale) after clustering of 885 human EST with a threshold value (parameter) ranging from 0.05 to 0.9 (axis Y) with a step of 0.05.

similarity to cluster consensus within clusters. Lower threshold results in bigger number of bigger clusters, for the price of possibly less representative consensus sequence. Any specific threshold is more or less arbitrary, setting a point of equilibrium between higher degree of clustering and higher cluster quality. In terms of cluster analysis, for this clustering algorithm the distance between clusters is the distance between nearest objects (ESTs) that belong to different clusters. Consequently, one single parameter affects both inter-object (EST to EST within a cluster) and inter-cluster distance. This parameter (stringency threshold) can have values between 0 and 1. When stringency threshold approaches 0 every pair of sequences produce general similarity index above exceeding the threshold. As a consequence, all initial sequences stick together in one cluster. In the opposite case, when stringency is too high (approaching 1) even slightly dissimilar sequences produce similarity index below threshold. The result of over-stringent clustering is all singletons. The effect of stringency threshold on the number of clusters and cluster size distribution is given on Fig. 13.

To calibrate against and existing system, we used a set of EST clusters produced by *D2_cluster*. 10 raw clusters ranging from 2 to 882 EST were assembled using CAP3 (http://hercules.tigem.it). Then all clusters were used as initial data sets for stringent clustering and processed repeatedly with a threshold value running from minimal value, resulting in one cluster to the maximal value, producing only singletons. The threshold value, corresponding to the number of clusters, nearest to the number of contigs, generated by CAP3 was taken. The same threshold value was also used for loose clustering.

There are two measured statistics, which may help to optimize the stringency parameter. The stringent clustering application calculates the time spent in the alignment procedure relative to the general processing time. Under normal conditions i.e. clustering raw EST data with optimal stringency settings, this figure should be small. If the amount of time spent in the alignment procedure rises above the acceptable limit, this means that either the data set is enriched with matching sequences or that initial fast sequence comparison produces too many false-positive matches and stringency parameter has to be adjusted. Another measured statistic is percentage of initially detected matches confirmed by following thorough pair-wise alignment. This measure characterizes the rate of false-positive matches on the fast comparison stage. If ratio approaches 100% the stringency is too high and a considerable number of matches may be lost (Fig. 14). The stringency should be set so that percentage of confirmed matches stay low while the time, spent in alignment remains within acceptable limits.

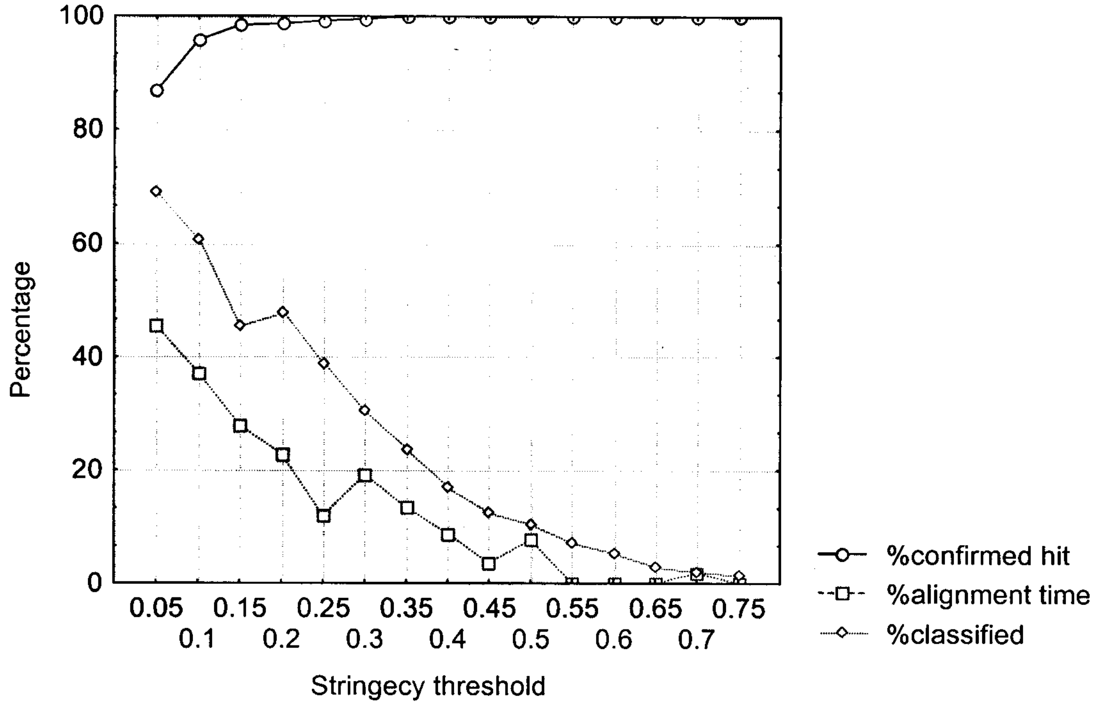

Figure 14. Effect of parameter (stringency threshold) variation on percentage of similarities confirmed by alignment, relative time spent on pair-wise alignment and percentage of initial ESTs assigned to clusters (non-singletons).

4.4 Estimation of clustering performance

Performance of the clustering programs was measured on the same data set, which was used for D2_cluster. In fact, the algorithm was tested on many different data sets, but this one particular data set was chosen for evaluation because it makes possible to compare the new clustering algorithm to the other existing systems. This data set contains the first 10000 ESTs from the eye tissue subset, prepared for STACK_PACK system (ftp://ftp.sanbi.ac.za/pub/STACK/benchmarks/benchmark10000.seq.gz). Unfortunately, the assessment of clustering quality is not yet established. Benchmark10000 (10000 ESTs from eye tissue set) is only a piece of real data big enough to represent variations of EST sequence length, repeat and vector sequence contamination, typical data quality and redundancy. Nevertheless, benchmark10000 is already established *de facto* standard benchmark for EST clustering software. Until now it's the only test data set for which results of more then one clustering system are publicly available. The results of testing of different programs are published on Internet and can be found at http://www.sanbi.ac.za/benchmark. The dataset itself can also be downloaded from the same URL. At least one other clustering system, ESTate, was also tested on the same benchmark10000 dataset (G. Slater, P. Van Heusden, personal communication). This data set can well serve as a basis to compare the relative speed of different clustering software. Although we don't have means to compare the quality of clustering, we suggest that as long as all tested programs produce similar results, their clustering quality is more or less the same. At least one of the tested programs, D2_cluster, has been already published as producing if not absolutely correct, at least valid and useful results from the scientific point of view. Thus, the other EST clustering systems are expected to produce a similar cluster structure. As a quick estimation of similarity of the results we propose a distribution of number of ESTs per cluster. The similarity between distributions doesn't guarantee the identity of clustering results. On the other hand, dissimilarity of distributions would be a good indicator of differences in resulting cluster structure. Our experiments show a remarkable correspondence between the result of D2_cluster and the new clustering programs, while the new programs are always significantly faster. Figure 15 shows distributions of cluster size in raw D2_cluster output, D2_cluster output after cluster assembly stage and the new stringent clustering program output.

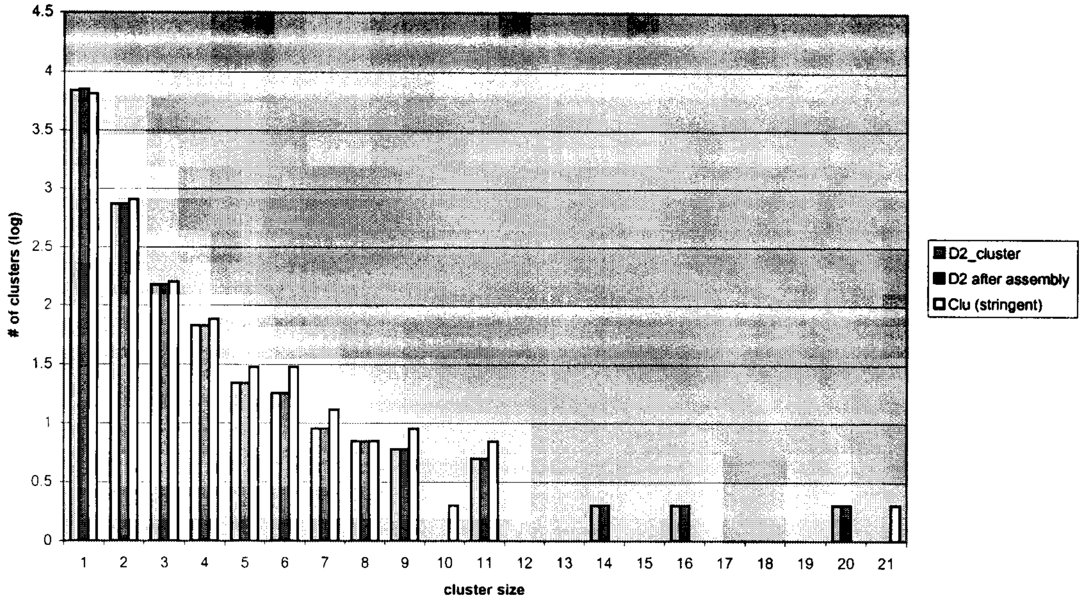

Figure 15. Comparison of cluster size distribution in first 10 000 human EST from eye tissue set.

The number of singletons generated by D2_cluster is 7006, while new program generates 6531. The biggest cluster defined by D2_cluster has 56 ESTs, but this cluster is not confirmed (defined as producing a single contig) by the following cluster assembly, as well as all other clusters with more then 20 members. Three biggest clusters produced by Clu have 66, 69 and 123 members. The new stringent clustering program (Clu) produces results with a very similar cluster size distribution (see Fig. 15). The difference is that Clu tends to produce more clusters, clusters of bigger size and leave less singletons. Quick comparison of the cluster contents shows that new clustering program (Clu) is more sensitive, especially in shorter sequences. The results of such comparison are shown in Table 2. Clusters are picked blindly by their numbers as they appear in the D2_cluster output. In both cases of disagreement between D2_cluster and Clu results all ESTs are longer than 100 base pairs (default frame length parameter in D2_cluster), but sequences missed by D2_cluster are shorter than the others in the same cluster.

Stack Cluster #	size	ESTs	Clu Cluster #	size	ESTs	difference
1	8	T27877 H37900 H38651 H38682 H84662 H85197 H89941 H84148	3145	8	T27877 H37900 H38651 H38682 H84662 H85197 H89941 H84148	
2	2	T27878 AA489885	7763	2	T27878 AA489885	
3	2	T27889 AA176889	5505	2	T27889 AA176889	
4	2	T27893 H84548	1040	2	T27893 H84548	
5	3	T27897 H37921 H40706	2240	4	T27897 H37921 H40706 H92170	H92170
6	6	T27899 H87764 H86519 AA057721 AA167121 AA489902	7780	6	T27899 H87764 H86519 AA057721 AA167121 AA489902	
7	2	T27904 AA063476	4532	4	T27904 AA063476 H40639 H38672	H40639 H38672
8	4	T27908 H37775 H85549 H86568	4051	5	T27908 H37775 H85549 H86568 H40669	
9	3	T27910 H80800 AA062794	4444	3	T27910 H80800 AA062794	
10	6	T27914 AA063475 AA057847 AA174102 AA219283 AA219467	6434	6	T27914 AA063475 AA057847 AA174102 AA219283 AA219467	

Table 2. Comparison of cluster contents between D2 and Clu (new clustering program) results. 10 clusters, produced by D2 from the benchmark10000 dataset with numbers from 1 to 10 are compared against corresponding Clu clusters. Due to the differences in algorithms, clusters containing the same ESTs have different numbers. In two cases of 10 (clusters #5 and #7) Clu clusters are bigger. Following alignment (available from the author upon request) confirms that additional ESTs belong to the corresponding clusters and are clearly alignable.

The data for performance comparison was obtained in the series of experiments by the author and Antoine van Gelder in September 1999 (A. van Gelder, personal communication). Generally, we can conclude that it's reasonable to suggest that results, produced by new program are similar to those of D2_cluster and can be equally useful for further research.

		Clustering program		
		D2_cluster	Loose Cluster	Stringent Cluster
Platform	SGI Origin2000 (R10000/180MHz)	12302	5660	-
	Pentium II/450	17014*	6021	5388
	Compaq DS20 (EV6/500MHz)	5655	2520	-

*Data for 400MHz CPU

Table 3. Comparison of the clustering software performance. The performance is measured in number of seconds required to complete clustering process. In case of Stringent clustering this also includes cluster assembly. First 10 000 human EST from eye tissue set are used as initial data.

4.5 Further improvements

Both programs are far from perfection and have good long-term development perspectives. First of all, there is a big gap between a prototype implementation of a new algorithm and a working clustering system. The overall performance and scientific value of the clustering results are determined only in part by the clustering program. EST pre-processing and further alignment analysis is just as crucial. Thus, an algorithm has much better chance to become a production engine if it can be introduced in a system, which is already working, and proven to give valuable results. From this point of view, the loose clustering program has good potential for introduction under the STACK_PACK system as a plug-in to substitute for D2_cluster. This would require only minor changes in the input and output data formats. Further optimization is also required on the last stage of clustering (see Figure 11), transforming a list of matching pairs into a connectivity group. This stage is memory demanding, due to the need of temporal storage of a huge array of matching EST pairs. Optimization of sorting and processing of this array can significantly reduce the overall computation time.

Stringent clustering would require more efforts to incorporate into any of existing systems, including STACK_PACK, but this is not impossible. Current implementation doesn't keep the alignments of clusters for analysis and generates a cluster consensus based on unsorted pair-wise alignments only. This program performs both clustering and cluster assembly, but the quality of results is significantly diminished by its' implementation. Sequential pair-wise alignment is only tolerable in a prototype-level testing, as it can't compete in accuracy with the tools, specifically developed for cluster assembly, like PHRAP, used in STACK_PACK. Improvement of the consensus generation and introduction of multiple alignment is the priority in father development of the stringent clustering application. The currently implemented prototype version uses readily available code for pair-wise nucleotide sequence alignment and consensus generation (Strelets et al., 1992). Sequential pair-wise alignment of low-quality EST tends to reduce consensus quality and is significantly affected by the order of alignments. Re-implementation with more sophisticated multiple alignment to generate more representative consensus can correct this problem.

Another major improvement can make a major performance boost in both stringent and loose clustering approaches. A pre-processing stage can be easily introduced without any significant change in the algorithms. This stage can be based on any of the sub-linear algorithms, discussed in 1.3.2.6. Unlike the systems initially built around fast sub-linear sequence comparison, our system doesn't experience the problem of excessive false matches, is not supported by further assembly and doesn't rely on extensive pre-processing. The additional fast comparison stage should aim not to detect the matches, but rather to cut most obviously non-matching ESTs from further comparison.

Chapter 5

Conclusion

Why are linear sequence comparison algorithms still in implementation despite availability of faster sub-linear algorithms? As it was noted in 1.3.2.6, super-fast algorithms, based on some form of pre-indexing, are extremely attractive for EST clustering because of their low computational cost. Traditional algorithms of sequence comparison are based on the concept of pair-wise alignment, which require the query to be aligned, even in the most reduced form, to each of the sequences of the database. This means that the entire database needs to be pumped through the CPU as many times as the number of queries. Sub-linear algorithms make it possible to go over the "sound barrier" of sequence comparison and make overall computing time less dependent from the database size. Yet, a clustering system based on the linear algorithm of sequence comparison still has a few significant advantages. First of all, it's more sensitive to local similarities, and gives a better quality results. In addition it doesn't require database pre-processing. This advantage becomes clear if we consider the requirement to keep the data in the clustered database up to date. The amount of EST data tends to grow very fast and each update would require an update of pre-indexed database. The fast linear algorithm, described in chapter 2 is well suited for dynamic updates, without extensive re-indexing of the whole data set.

This algorithm retains the sensitivity of alignment-searching algorithms, like FASTA, while reducing computation to only a few integer operations per base pair. This is close to the theoretical maximum of linear algorithm performance and only limited by the ability of contemporary CPUs to pump the data through. Any further significant improvement of speed can be achieved only by introduction of a sub-linear algorithm, based on pre-indexing of the data set. On the other hand, introduction of a sub-linear "booster" is one of the main development perspectives of the clustering application. In our opinion it's reasonable to base a clustering system on a fast linear algorithm, leaving imprecise sub-linear algorithmic applications to a secondary role.

This algorithm developed by the author can be used in both of the present EST clustering strategies. In order to prove the applicability and estimate the performance two applications were developed. Loose clustering application is easier to introduce to the existing STACK_PACK system, while stringent clustering program has bigger potential for further development.

The eclectic nature of existing EST clustering systems leads to contamination and errors. Despite thorough masking, fragments of repeats are still found in the STACK clusters and can be easily detected by BLAST search. Different programs, utilized in STACK_PACK system use different algorithms of sequence comparison and, consequently, different similarity measures. Each program (CROSSMATCH, D2_cluster, PHRAP) utilises a slightly different meaning of sequence similarity and this one of the potential breaches, letting contaminated sequences into clustering. Unification of sequence comparison by employment of the same sequence comparison algorithm on all stages can solve the problem. This was the main idea behind the development of a repeat and vector masking program, as well as the development of the stringent clustering application. If matches, detected by the clustering program have the same sense as in the cluster assembly stage, this would reduce the redundant sequence comparison, excluding situations where the cluster, reported by clustering program is broken into pieces by the assembly stage. If sequence similarity has the same *sense* through all the system, it would simplify the system architecture and, finally, improve its' overall performance.

Summary

Expressed sequence tag database is a rich and fast growing source of data for gene expression analysis and drug discovery. Clustering of raw EST data is a necessary step for further analysis and one of the most challenging problems of modern computational biology. There are a few systems, designed for this purpose and a few more are currently under development. These systems are reviewed in the "Literature and software review". Different strategies of supervised and unsupervised clustering are discussed, as well as sequence comparison techniques, such as based on alignment or oligonucleotide compositions.

Analysis of potential bottlenecks and estimation of computation complexity of EST clustering is done in Chapter 2. This chapter also states the goals for the research and justifies the need for new algorithm that has to be fast, but still sensitive to relatively short (40 bp) regions of local similarity.

A new sequence comparison algorithm is developed and described in Chapter 3. This algorithm has a linear computation complexity and sufficient sensitivity to detect short regions of local similarity between nucleotide sequences. The algorithm utilizes an asymmetric approach, when one of the compared sequences is presented in a form of oligonucleotide table, while the second sequence is in standard, linear form. A short window is moved along the linear sequence and all overlapping oligonucleotides of a constant length in the frame are compared for the oligonucleotide table. The result of

comparison of two sequences is a single figure, which can be compared to a threshold. For each measure of sequence similarity a probability of false positive and false negative can be estimated. The algorithm was set up and implemented to recognize matching ESTs with overlapping regions of 40bp with 95% identity, which is better than resolution ability of contemporary EST clustering tools.

This algorithm was used as a sequence comparison engine for two EST clustering programs, described in Chapter 4. These programs implement two different strategies: stringent and loose clustering. Both are tested on small, but realistic benchmark data sets and show the results, similar to one of the best existing clustering programs, D2_cluster, but with a significant advantage in speed and sensitivity to small overlapping regions of ESTs. On three different CPUs the new algorithm run at least two times faster, leaving less singletons and producing bigger clusters. With parallel optimization this algorithm is capable of clustering millions of ESTs on relatively inexpensive computers. The loose clustering variant is a highly portable application, relying on third-party software for cluster assembly. It was built to the same specifications as D2_cluster and can be immediately included into the STACKPack package for EST clustering. The stringent clustering program produces already assembled clusters and can apprehend alternatively processed variants during the clustering process.

References:

Aaronson, J.S., B. Eckman, R.A. Blevins, J.A. Borkowski, J. Myerson, S. Imran, and K.O. Elliston. 1996. Toward the Development of a Gene Index to the Human Genome: An Assessment of the Nature of High-throughput EST Sequence Data. Genome Research 6:829-845.

Adams, M.D., J.M. Kelley, J.D. Gocayne, M. Dubnick, M.H. Polymeropoulos, H. Xiao, C.R. Merril, A. Wu, B. Olde, R.F. Moreno, A.R. Kerlavage, W.R. McConbie, and J.C. Venter. 1991. Complementary DNA Sequencing: Expressed Sequence Tags and Human Genome Project. Science 252: 1651-1656.

Aivazyan, S., Buchstaber, V., Yenyukov, I. and Meshalkin, L. 1989. Classification and Reduction of Dimensionality. Finansy i statistika, Moscow (in Russian).

Altschul, S.F., Gish, W., Miller, W., Myers, E.F., and Lipman, D.J. 1990. Basic local alignment search tool. J. Mol. Biol., 215: 403-410.

Alwen, J. United Kindom human genome mapping project: development, components, coordination and management, and international links of the project. 1990. Genomics. 6:386-388.

Baldo, M.F., G. Lennon, and M.B. Soares Normalization and Subtraction: Two Approaches to FacilitateGene Discovery. Genome Research 1996. 6:791 - 806.

Benson D.A., M.S. Boguski, D.J. Lipman, and J. Ostell. 1994. GenBank. Nucleic Acids Research 22: 3441-3444.

Boguski M.S., Lowe T.M., Tolstoshev C.M. 1993. dbEST-database for "expressed sequence tags". Nature Genetics 4(4):332-3

Boguski, M.S., Schuler, G.D. 1995. ESTablishing a Human Transcript Map. Nature Genetics 10:369-371.

Bouck J., Yu W., Gibbs R., Worley K. 1999. Comparison of gene indexing databases. Trends Genet. 15(4):159-162.

Bonfield, J.K., Smith, K.F., Staden, R. 1995. A new DNA sequence assembly program. Nucleic Acids Res. 23(24):4992-4999.

Britten, R.J., Kohne D.E., 1968. Repeated sequences in DNA. Hundreds of thousands of copies of DNA sequences have been incorporated into the genomes of higher organisms. Science. 161(841):529-40.

Bull, L.N., Pabon-Pena, C.R., Freimer, N.B. 1999. Compound microsatellite repeats: practical and theoretical features. Genome Res. 9(9):830-838.

Burkhardt, S., Crauser, A., Ferragina, P., Lenhof, H-P., Rivals, E., Virgon, M 1999. Q-gramm Based Database Searching Using a Suffix Array (QUASAR), Proceedings of the 3rd International Conference on Computational Molecular Biology (RECOMB99), Lyon, France. 77-83.

Burke, J.P., H. Wang, W. Hide, and D. Davison. 1998. Alternative Gene Form Discovery and Candidate Gene Selection from Gene Indexing Projects. Genome Research 8: 276-290.

Califano, A., Rigoutsos, I. 1993. FLASH: a fast look-up algorithm for string homology. ISMB. 1:56-64.

Cariaso, M., Folta, P., Wagner, M., Kuczmarski, T., Lennon, G. 1999. IMAGEne I: clustering and ranking of I.M.A.G.E. cDNA clones corresponding to known genes. Bioinformatics. 15(12):965-973.

Chang, W., Lawler, E. 1994. Sublinear approximate string matching and biological applications. Algorithmica, 12(4/5):327-344.

Eichler, E.E. 1998. Masquerading repeats: paralogous pitfalls of the human genome. Genome Res. 8(8):758-62.

Eisen, M., Spellman, P., Brown, P. and Botstein, D. 1998. Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. USA 95:14863-14868.

Ewing, R.M., Kahla, A.B., Poirot, O., Lopez, F., Audic, S., Claverie, J.M. 1999. Large-scale statistical analyses of rice ESTs reveal correlated patterns of gene expression. Genome Res. 9(10):950-9.

Gauntheret, D., Poirot, O., Lopez, F., Audic, S. and Claverie, J.-M., 1998. Alternative Polyadenylation in Human mRNAs: A Large-Scale Analysis by EST Clustering. Genome Res. 8:524-530

Gill, R.W., Hodgman, T.C., Littler, C.B., Oxer, M.D., Montgomery, D.S., Taylor, S., Sanseau, P. Advanced Technology & Informatics Unit, Glaxo-Wellcome Medicines Research Centre, Stevenage, Heris, UK. rwg23944@ggr.co.uk

Graul, R.C., Sadee, W. 1997. Evolutionary relationships among proteins probed by an iterative neighborhood cluster analysis (INCA). Alignment of bacteriorhodopsins with the yeast sequence YRO2. Pharm Res. 14(11):1533-41.

Green, P. 1994-1996. PHRAP http://boseman.mbt.washington.edu/phrap.docs/phrap.html

Grillo, G., Attimonelli, M., Liuni, S., Pesole, G., 1996. CLEANUP: a fast computer program for removing redundancies from nucleotide sequence databases. Comput. Appl. Biosci. 12(1):1-8.

Harger, C., Skupski, M., Bingham, J., Farmer, A., Hoisie, S., Hraber, P., Kiphart, D., Krakowski, L., McLeod, M., Schwertfeger, J., Seluja, G., Siepel, A., Singh, G., Stamper, D., Steadman, P., Thayer, N., Thompson, R., Wargo, P., Waugh, M., Zhuang, J.J., Schad, P.A. 1998. The Genome Sequence DataBase (GSDB): improving data quality and data access. Nucleic Acids Res. 26(1):21-6.

Hawkins, V., Doll, D., Bumgarner, R., Smith, T., Abajian, C., Hood, L., Nelson, P.S. 1999. PEDB: the Prostate Expression Database. Nucleic Acids Res. 27(1):204-8.

Heyer, L.J., Kruglyak, S., Yooseph, S. 1999. Exploring expression data: identification and analysis of coexpressed genes. Genome Res. 9(11):1106-15.

Hide, W., J. Burke, D. Davison 1994. Biological Evaluation of d^2, an Algorithm for High-Performance Sequence Comparison, Journal of Computational Biology 1(3):199-215.

Hide, W., J. Burke, A. Christoffels, Miller, R. 1997. A novel approach towards a consensus representation of the expressed human genome, Genome Informatics, Universal Academy Press Inc. Tokyo, Japan. ISSN 0919-9454, 187-196.

Hillier, L., Clark, N., Dubuque, T., Elliston, K., Hawkins, M., Holman, M., Hultman, M., Kucaba, T., Le, M., Lennon, G., Marra, M., Parsons, J., Rifkin, L., Rohlfing, T., Soares, M., Tan, F., Trevaskis, E., Waterston, R., Williamson, A., Wohldmann, P., Wilson, R. 1996. Generation and Analysis of 280,000 Human Expressed Sequence Tags. Genome Research 6:807-828.

Hishiki, T., Kawamoto, S., Morishita, S., Okubo, K., 2000. BodyMap: a human and mouse gene expression database. Nucleic Acids Res. 28(1):136-8.

Houlgatte, R., Mariage-Samson, R., Duprat, S., Tesslier, A., Bentolila, S., Lamy, B., Auffray, C. 1995. The GenExpress Index: A Resource for Gene Discovery and the Genic Map of the Human Genome. Genome Research 5: 272-304.

Huang X. 1996. An improved sequence assembly program. Genomics. 33(1):21-31.

Jiang, J., Jacob, H.J. 1998. EbEST: an automated tool using expressed sequence tags to delineate gene structure. Genome Res. 8(3):268-75.

Jurka, J. 1998. Repeats in genomic DNA: mining and meaning. Curr. Opin. Struct. Biol. 8:333-337.

Jurka, J., Klonowski, P., Dagman, V., Pelton, P. 1996. CENSOR - a program for identification and elimination of repetitive elements from DNA sequences. Computers and Chemistry 20(1):119-122.

Kapros, T., Robertson, A.J., Waterborg, J.H. 1994. A simple method to make better probes from short DNA fragments. Mol. Biotechnol. 2(1):95-8.

Knuth, D. 1973. The Art of Computer Programming (Vol. III): Sorting and Searching. Addison-Wesley, Reading, MA.

Lamperti, E.D., Kittelberger, J.M,, Smith, T.F., Villa-Komaroff L. 1992. Corruption of genomic databases with anomalous sequence. Nucleic Acids Res. 20(11):2741-7.

Lennon, G.G., Auffray, C., Polymeropulos, M. and Soares, M.B. 1996. The I.M.A.G.E. Consortium: An integrated molecular analysis of genomes and their expression. Genomics 33:151-152.

Manber, U., and Myers, E. 1993. Suffix Arrays: a new method for on-line string searches. SIAM Journal on Computing, 22(5): 935-948.

Marra, M., Hillier, L., Kucaba, T., Allen, M., Barstead, R., Beck, C., Blistain, A., Bonaldo, M., Bowers, Y., Bowles, L., Cardenas, M., Chamberlain, A., Chappell, J., Clifton, S., Favello, A., Geisel, S., Gibbons, M., Harvey, N., Hill, F., Jackson, Y., Kohn, S., Lennon, G., Mardis, E., Martin, J., Waterston, R., et al. 1999 An encyclopedia of mouse genes. Nat Genet. 21(2):191-194.

Martinez, H. 1983. An efficient method for finding repeats in molecular sequences. Nucleic Acid Research, 11(13):4629-4634.

McCraight, E. 1976. A space-economical suffix tree construction algorithm Journal of the ACM, 23(2):262-272.

Miller, G., Fuchs, R., Lai, E. 1997. IMAGE cDNA clones, UniGene clustering, and ACeDB: an integrated resource for expressed sequence information. Genome Res. 7(10):1027-32.

Miller, R.T., Christoffels, A.G., Gopalakrishnan, C., Burke, J., Ptitsyn, A.A., Broveak, T.R., Hide, W.A. 1999. A comprehensive approach to clustering of expressed human gene sequence: the sequence tag alignment and consensus knowledge base. Genome Res. 9(11):1143-55.

Myers, E., 1994. A sublinear algorithm for approximate keyword searching. Algorithmica, 12(4/5):345-374.

Ohno S. 1999. Gene duplication and the uniqueness of vertebrate genomes circa 1970-1999. Semin Cell Dev Biol. 10(5):517-522.

Parsons, J.D., Rodriguez-Tome P. 2000. JESAM: CORBA software components to create and publish EST alignments and clusters. Bioinformatics. 16(4):313-325.

Pearson, W.R. and Leapman, D.J. 1988. Improved tools for biological sequence comparison. Proc. Natl. Acad. Sci. USA, 85:2444-2448.

Picoult-Newberg, L., Ideker, T.E., Pohl, M.G., Taylor, S.L., Donaldson, M.A., Nickerson, D.A., Boyce-Jacino, M. 1999. Mining SNPs from EST databases. Genome Res. 9(2):167-74.

Pietu, G., Mariage-Samson, R., Fayein, N.A., Matingou, C., Eveno, E., Houlgatte, R., Decraene, C., Vandenbrouck, Y., Tahi, F., Devignes, M.D., Wirkner, U., Ansorge, W., Cox, D., Nagase, T., Nomura, N., Auffray, C. 1999. The Genexpress IMAGE knowledge base of the human brain transcriptome: a prototype integrated resource for functional and computational genomics. Genome Res. 9(2):195-209.

Quackenbush, J., Liang, F., Holt, I., Pertea, G., Upton, J. 2000. The TIGR gene indices: reconstruction and representation of expressed gene sequences. Nucleic Acids Res. 28(1):141-145.

Smitt, A., Specht, T., Beckmann, G., Dahl, E., Pilarsky, C., Hinzmann, B., Rosenthal, A. 1999. Exhaustive mining of EST libraries for genes differentially expressed in normal and tumor tissues, Nucleic Acid Research, 27(21):4251-4260.

Smith, T.F., Waterman, M.S. 1981. The Identification of common molecular subsequences. Jour. Mol. Biol., 147:195-197.

Sterky, F., Regan, S., Karlsson, J., Hertzberg, M., Rohde, A., Holmberg, A., Amini, B., Bhalerao, R., Larsson, M., Villarroel, R., Van Montagu, M., Sandberg, G., Olsson, O., Teeri, T.T., Boerjan, W., Gustafsson, P., Uhlen, M., Sundberg, B., Lundeberg, J. 1998. Gene discovery in the wood-forming tissues of poplar: analysis of 5, 692 expressed sequence tags. Proc. Natl. Acad. Sci. U S A. 95(22):13330-13335.

Strelets, V.B., Ptitsyn, A.A., Milanesi, L., Lim, H.A. 1994. Data bank homology search algorithm with linear computation complexity. Comp. Appl. Biosci., 10(3):319-322.

Streletc, V.B., Shindyalov, I.N., Kolchanov, N.A., Milanesi, L. 1992. Fast, statistically based alignment of amino acid sequences on the base of diagonal fragments of DOT-matrices. Comp. Appl. Biosci., 6:529-534.

Sutton, G., White, O., Adams, M.D., Kerlavage, A.R. 1995. TIGR Assembler: A New Tool for Assembling Large Shotgun Sequencing Projects. Genome Science and Technology 1:9-18.

Tautz D. 1993. Notes on the definition and nomenclature of tandemly repetitive DNA sequences. EXS. 67:21-28.

Torney, D.C., Burks, C., Davison, D., and Sirotkin, K.M. (1990) Computation of d^2. A measure of sequence dissimilarity. In Bell, G. and Marr, T., eds., Computers and DNA, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, New York.

Tryon, R.C., 1939. Cluster Analysis. Ann. Arb., Edw. Brothers.

Ukkonen, E., 1992 Approximate string-matching with q-gramms and maximal matches. Theoretical Computer science, 92(1):191-211.

Vasmatzis, G., Essand, M., Brinkmann, U., Lee, B., Pastan I., 1998. Discovery of three genes specifically expressed in human prostate by expressed sequence tag database analysis. Proc. Natl. Acad. Sci. USA 95(1):300-304 .

Vingron, M., Hoheisel, J., 1999. Computational aspects of expression data. J Mol Med. 77(1):3-7.

Williamson, A. R., Elliston, K.O., Sturchio, J.L., 1995. The Merck Gene Index, a public resource for genomic research. J. NIH Res., 7: 61-63.

Wolfberg, T.G. and D. Landsman. 1997. A comparison of expressed sequence tags (ESTs) to human genomic sequences. Nucleic Acids Research 25(8):1626-1632.

Wu, T.J., Burke J.P., Davison D.B., 1997. A Measure of DNA Sequence Dissimilarity Based on Mahalanobis Distance Between Frequencies of Words. Biometrics 53:1431-1439.

Yee, D.P., Conklin, D., 1998. Automated clustering and assembly of large EST collections. ISMB. 6:203-11.