JESAM: CORBA software components to create and publish EST alignments and clusters

J. D. Parsons¹,²,* and P. Rodriguez-Tomé¹

¹EMBL-Outstation, The European Bioinformatics Institute (EBI), Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, UK

Received on June 1, 1999; revised on September 30, 1999; accepted on October 13, 1999

Abstract

Motivation: Expressed Sequence Tags (ESTs) are cheap, easy and quick to obtain relative to full genomic sequencing and currently sample more eukaryotic genes than any other data source. They are particularly useful for developing Sequence Tag Sites (STSs for mapping), polymorphism discovery, disease gene hunting, mass spectrometer proteomics, and most ironically for finding genes and predicting gene structure after the great effort of genomic sequencing. However, ESTs have many problems and the public EST databases contain all the errors and high redundancy intrinsic to the submitted data so it is often found that derived database views, which reduce both errors and redundancy, are more effective starting points for research than the original raw submissions. Existing derived views such as EST cluster databases and consensus databases have never published supporting evidence or intermediary results leading to difficulties trusting, correcting, and customizing the final published database. These difficulties have lead many groups to wastefully repeat the complex intermediary work of others in order to offer slightly different final views. A better approach might be to discover the most expensive common calculations used by all the approaches and then publish all intermediary results. Given a globally accessible database with a suitable component interface, like the JESAM software described in this paper, the creation of customized EST-derived databases could be achieved with minimum effort.

Results: Databases of EST and full-length mRNA sequences for four model organisms have been self-compared by searching for overlaps consistent with contiguity. The sequence comparisons are performed in parallel using a PVM process farm and previous results are stored to allow incremental updates with minimal effort. The overlap databases have been published with CORBA interfaces to enable flexible global access as demonstrated by example Java applet browsers. Simple cDNA supercluster databases built as alignment database clients are themselves published via CORBA interfaces browsable with prototypical applets. A comparison with UniGene Mouse and Rat databases revealed undesirable features in both and the advantages of contrasting perspectives on complex data.

Availability: The software is packaged as two Jar files available from: URL: http://corba.ebi.ac.uk/EST/jesam/jesam.html. One jar contains all the Java source code, and the other contains all the C, C++ and IDL code. Links to working examples of the alignment and cluster viewers (if remote firewall permits) can be found at http://corba.ebi.ac.uk/EST. All the Washington University mouse EST traces are available for browsing at the same URL.

Contact: jparsons@ebi.ac.uk

Introduction

CORBA and biological data

The use of CORBA in a biological context was introduced by Hu et al. (1998) and Lijnzaad et al. (1998) where they explain that CORBA can be a good solution to the problem of creating applications for distributed heterogeneous environments. The Internet is the extreme example of both distribution and heterogeneity and, therefore, the hardest to support with conventional client/server systems. Orfali and Harkey (1998) see CORBA as a good middleware solution for the Internet and refer to the creation of the ‘Object Web’ where CORBA hides some of the most complex interoperability issues behind well specified interfaces fronting globally accessible software components. Principal among the benefits that CORBA brings to biological data distribution and interaction, are: the Interface Definition Language (IDL) to define interfaces between objects; scalability via language; location and Operating System independence; a rich set of 15 object services. CORBA is often described as being middleware because it sits between software components or data sources, and lets them all interact without regard to intervening
distance or data conversions. In 1996, the ease with which client/server object-oriented applications could be written, distributed and supported across the Internet increased when Netscape (http://www.netscape.com/) announced (Orfali and Harkey, 1997) that its browsers were all going to include a CORBA Object Request Broker (ORB) alongside the built-in Java Virtual Machine (JVM), and when it subsequently decided to distribute its browsers free of charge.

Sequence databases (for example the EMBL NSDB Stoesser et al., 1998) are currently the most familiar biological databases and are typical by a source for EST sequences. Traditionally these databases are presented and distributed using ‘flat file’ views of the sequence originally submitted, but these static raw views are being superseded by novel interfaces better suited to the needs of academic and industrial researchers. The new browsing interfaces are needed because current research may need to investigate nucleotide or protein sequences grouped by their function, phylogeny, map position, cellular location, expression regulation, disease association, annotation similarity or metabolic pathway etc. To support these new investigative cross-database clients, it is useful to have standardized API level access to both data and services—a requirement where CORBA is starkly appropriate. In summary, ESTs are inherently difficult data to handle (yet extremely useful), so they are a natural target for new efforts to improve data quality, cross database browsing and novel visualization tools.

Expressed Sequence Tag databases and clustering extensions

Of all the biological databases, Expressed Sequence Tag (EST) databases have been the most conspicuous in recent years both because of their prodigious growth rates and because of their increase in intrinsic importance connecting different areas of biological research. Cheap, easy and quick to obtain relative to full genomic sequencing, ESTs currently sample more eukaryotic genes than any other data source. They have great current utility for disease gene hunting (Banfi et al., 1996), mass spectrometer proteomics (Küster and Mann, 1998), and for finding genes and predicting gene structure after genomic sequencing (Bailey et al., 1998; Jiang and Jacob, 1998). The patterns of early public work with human ESTs (Boguski et al., 1993; Hillier et al., 1996; Gerhold and Caskey, 1996) are now being repeated for mouse and other eukaryotic model species. The Mouse Genome Project is advancing quickly (Battey et al., 1999), with a large EST project (Marra et al., 1999) and various maps (Blake et al., 1998; McCarthy et al., 1997) completed, and novel databases available for unique analyses of mammalian development (Davidson et al., 1997; Makalowski et al., 1996).

ESTs have many problems that stem from their means of production. The cDNA library, from which any ESTs are drawn, will sample the levels of expression in a particular tissue at a particular time: rare transcripts will be missed and highly expressed genes will be overly abundant. This latter problem, redundancy, is both wasteful and difficult to handle due to the extreme volume of error-prone data. Redundancy can be reduced in the laboratory using normalization techniques; these are imperfect, however, although additional subtractive techniques are being investigated (Bonaldo et al., 1996).

To address the redundancy problem using computer techniques, a variety of EST clustering programs have been implemented (e.g. Parsons et al., 1992; Boguski and Schuler, 1995; Parsons, 1995; Eckman et al., 1998; Yee and Conklin, 1998). Once clustered, gene fragments can be assembled using sequence assembly programs, or both steps can be combined in large-scale EST assembly and gene indexing protocols. An assembly stage adds two main benefits: first, it produces contigs and consensus sequences which can completely hide EST redundancy; secondly, it should also improve the length and quality of the gene reconstructions beyond that available from any one EST (usually single-pass reads). For some applications, the clustering is all that is required, for example the early stages of STS mapping, but for others sophisticated cluster partitioning (Burke et al., 1998) or complex assemblies are more suitable.

Whilst the redundancy problem can be mitigated, other deficiencies of EST databases are computationally insoluble and most EST-based projects have centred on pragmatic scientific goals cognizant that perfect results are impossible. Early projects that used EST clustering included TIGRs THC database (Adams et al., 1995), Geneexpress (Houlgate et al., 1995), and UniGene (Boguski and Schuler, 1995), but new genomes (e.g. Arabidopsis Rounsley et al., 1996), new methods (e.g. SANBI STACK, http://www.sanbi.ac.za/Dbases.html) and new public data (Aaronson et al., 1996) have kept the field busy with important discoveries and enhanced views (Eckman et al., 1998).

Assembly and Gene Indexing

Assembling gene transcript fragments is more complex than assembling shotgun clones of genomic DNA. One gene does not lead to one EST, nor even a statistically predictable sampling depth and, to complicate things further, there are also variations in the underlying reconstruction target. The cDNA libraries that have been used in the public EST projects were made from many different individuals/genotypes and tissues so the public EST collections show polymorphism and tissue-specific variation. Additionally, errors of many forms (base calling, misnam-
ing, vector and cross-species contamination, chimeras, poor template, etc.) are more likely with broadly sampled complex cDNA collections than with tightly focussed genomic projects which can operate more internal checks and aim for a simpler single target. Tissue variation may take the form of alternative polyadenylation (Gautheret et al., 1998), or alternative splicing (Lopez, 1998). In all cases, the best possible EST comparisons will require knowledge of the likelihood of the various types of error, and the exact tissue-specific origins of the source cDNA library.

Clustering of EST on typical cDNA/EST collections will produce clusters which may vary in size from one to approximately 20 000, which is a size range commonly encountered with larger shotgun assembly projects. Some DNA shotgun assembly programs (necessary for all large-scale genomic sequencing) have been applied to EST assembly; however, there is a significant danger of producing artificial sequences by incorrectly joining, or homogenizing transcripts from different genes. In summary, postulated transcription products will ultimately homogenize transcripts from different genes. In summary, postulated transcription products will ultimately need experimental biological confirmation and the totality of products from a single gene may not fit well into a FASTA format file: instead, a description is needed with the potential to record all of the reconstructed biological complexity.

Clusters of EST can be combined with other sources of gene sequence information (for example full-length mRNA, and genomic DNA) to better sample an organism's entire genome (especially for rare transcripts), to create 'gene indexes', and consensus sequence databases. As gene mapping information accumulates, so the gene indexes lead to the creation of 'gene maps' which link genes and their functions to locations on a physical map.

Gene maps supported by polymorphism information make an extremely valuable resource for positional candidate disease gene hunting.

Introduction to EST clustering algorithms

Computational complexity. A typical EST sequencing project may produce between 50 000 and a few million ESTs which need to be compared against each other (order \( N^2 \)). This formidable scaling problem must be handled by whichever algorithm is chosen and before user considerations such as input requirements and presentation style can be considered. Algorithmic tools with the potential to reduce the comparison complexity include suffix trees, hashes, Finite State Machines, indexing and statistical DNA-word frequency analyses. These methods could be applied sequentially for each sequence versus the rest, or (for some of the algorithms) a simultaneous all-against-all result emerges inside some data structure.

Experimental and biological complexity. Deciding if and how two ESTs are related is complicated by errors in the process of EST manufacture: short, single-pass reads (Hillier et al., 1996), and by many natural causes of variation (Wolfsberg and Landsman, 1997). The program that best estimates the likelihood of combinations of all these complicating events should generate the best set of clusters or contigs, and the best overall result. Given the scale and difficulty of the challenges faced, even the best programs will make many mistakes and be unable to assess the significance of, or just plain miss, much of importance so experimental confirmation is necessary. This should not be a surprise as the same is true of even fully finished genomes (Oliver, 1996).

Comparisons of existing programs

Unfortunately, there are as yet no detailed comparisons of the quality of any current clustered EST databases and direct comparisons of the clustering tools themselves are complicated by potentially important issues such as results presentation style, software portability, memory, disk and CPU requirements. Miller and Powell (1994) published a quantitative comparison of genomic shotgun assembly packages but this work is not directly relevant because all the programs have since changed and the paper did not address issues specific to EST assembly. Burke et al. (1998) did publish two clustering comparisons: first, a subpartitioning of UniGene101, but this used the existing (not independent) UniGene clusters as its starting point, secondly, a contrasting comparison with TIGR Gene Index, but this was summary and non-rigorous using different EST data.

Published references for EST comparison programs show a variety of specified goals and assumptions. The simplest use scripting languages (e.g. Perl) to run and parse sequence database searches by BLASTN (Altschul et al., 1997, http://www3.ncbi.nlm.nih.gov/BLAST/) or FASTA (Pearson et al., 1997, http://www.cs.virginia.edu/brochure/profs/pearson.html). Sequences of EST may be clipped to various levels of quality, vector trimmed, and low complexity masked before being sent for BLASTN/FASTA comparisons. Both BLASTN and FASTA hash the query sequence before scanning a sequence database, whereas JESAM and ACEmbly hash the whole database (indexing) and find hash hits for the query against all database sequences simultaneously before checking for alignments.

Construction of Compugen's EST clusters (LEADS, http://www.cgen.com/solutions/index.html) uses a proprietary non-alignment based algorithm. Other (published) non-alignment algorithms, include EMBLSCAN (Bishop and Thompson, 1984) D-squared Cluster, and RAPID (Miller et al., 1999), try to estimate the significance of discovering oligos (DNA words) in common between two
sequences without the cost of a full alignment.

The simple ideal that all cognate ESTs should cluster together could be achieved easily if all 3' ESTs were anchored to the same region of the cDNA and gene. In this situation (ignoring sequencing errors) any EST from the same gene would have the same basecalled sequence. The original ICAtools (Parsons et al., 1992) were the first tools built to automate this simple analysis. Unfortunately, as described by Gautheret et al. (1998), alternate polyadenylation is common (in humans) so ESTs may not be well anchored and discovering cognate ESTs reverts to solving a peculiar massive shotgun assembly. The anchored EST assumption will, however, still give useful results for strictly relative comparisons of the complexity anchored EST assumption will, however, still give useful results for strictly relative comparisons of the complexity within individual cDNA libraries, especially if transcript heterogeneity is temporally and anatomically restricted (Gautheret et al., 1998; Burke et al., 1998).

**BLAST and FASTA based scripts**

Examples of the BLASTN or FASTA clusters include: INCA (Graul and Sadée, 1997, http://itsa.ucsf.edu/~gram/home/inca/) SEALS (http://www.ncbi.nlm.nih.gov/Walker/SEALS/index.html), WHALE (unpublished), Zymogenetics' REX (Yee and Conklin, 1998) and the anonymous tools used to produce Geneexpress, 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 user-defined rules of when two sequences should actually be considered to overlap. 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).

Typical extra heuristics include (for Geneexpress) 90% overall nucleotide identity in the overlapping region and no more than 35 unmatched (outside local alignment) dangling sequence ends (GeneInfo et al.).

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. The approach taken by Glaxo was quick to implement and effective for simple assemblies, but suffers pauses between each round of assembly, the Greedy assembly is prone to errors with repetitive sequence regions, the static HTML output limits interactivity/editing and alternative BLASTN alignments displayed simultaneously in one alignment may cause confusion for the user.

The original UniGene package (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. These sequence pairs were then 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), assemblies, alignments or consensus sequences are not published. UniGene is available online at http://www.ncbi.nlm.nih.gov/UniGene/index.html with extra descriptions at http://www.ncbi.nlm.nih.gov/Schuler/Papers/ESTtransmap/

**Specialized clustering programs**

The ICAtools and CLEANUP are specialized hash-based-query sequence clustering packages. One of the latest ICAtools, ICAass (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. A variety of specialized cluster-browsing tools assist extraction of non-redundant sequence sets, or allow accelerated database searches where an increased portion of the computation time is spent comparing the query with very similar sequences. All code is ANSI C, runs on many UNIX variants (and ported to MacOS), and is free to academics and industry. It has been used to cluster 180 000 ESTs on one shared 143 MHz UltraSPARC in 9 days. Memory usage scales linearly with database size, but computation time scales quadratically.

Similar to the ICAtools, CLEANUP (Grillo et al., 1996) 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.

**Small-scale assembly programs**

Conventional shotgun sequence assembly relies on overlapping alignment of the called nucleotide bases to recreate a representation of the original DNA progenitor sequence. All the shotgun assemblers referenced except ACEmbly use a hashed-query sequence algorithm for finding matches. Some of the more sophisticated pro-
grams 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. Relevant programs include Gap4 (Bonfield et al., 1995), CAP3 (http://genome.cs.mtu.edu/cap/cap3.html), Phrap and Gene Meyer’s FAK (Meyers, 1995). The conventional shotgun assemblers can be used to assemble typically sized clusters of ESTs.

Gap4 (http://www.mrc-lmb.cam.ac.uk/pubseq/overview.html) implements a simple algorithm (relatively slow) 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 (unpublished, 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 such as alternative splicing etc. Assemblies are fast (6 h for 49,000 sequences on a single 195 MHz R10000); however, a single sequence addition requires all previous calculations to be repeated. An optimized and parallelized version of Phrap is commercially available (http://www.spsoft.com/).

CAP3 (unpublished, http://genome.cs.mtu.edu/cap/cap3.html) is an enhancement of CAP2 (Huang, 1996)—both are relatively slow. CAP3 can utilize ancillary information to guide sequence assembly: mate-pairs, clone size, orientation/location, and basecall error estimates. CAP3 is used at MIPS to assemble UniGene clusters (http://www.mips.biochem.mpg.de/proj/human/).

TIGR Assembler (Sutton et al., 1995) is a relatively fast sequence-assembly program that has been proved when assembling a diverse collection of repetitive microbial genomes. Memory usage for the query-hashing stage is 32 * 4^n 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 Tentative Human Consensus (THC) sequences for all the human ESTs. The THC database is biased towards splitting EST clusters due to this stringent assembly stage (Burke et al., 1998). TIGR Assembler on its own has been used to assemble 186,000 ESTs on a single SPARC processor, with 512 Mb RAM, in 7 days.

ACEmbly (http://alpha.crbm.cnrs-mop.fr/acembly/) 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 ‘zipping’ is fast (<10 s for 600 reads) but prone to errors with repeated sequences. JESAM, the subject of this paper is another hashed database sequence comparison program.

Word-based clustering

Word-based clustering is inherently fast because it omits all alignment steps before clustering. Eliminating initial alignments raises the potential for poor-quality clustering—either through poor stability, or poor selectivity. Word-based clustering uses less information and so has greater error potential, especially when compared to sequence alignments that can use position-dependent sequence quality measures. Good specificity depends on refinements such as statistical weighting schemes where common oligos are given reduced significance. The sensitivity of word-based alignments has not been discussed to any depth in the literature.

The published work on D-Squared Cluster (Hide et al., 1994, http://www.bchs.uh.edu/stemple/D2/d2.html) was preliminary and experimental with only limited exploration of the kinds of optimizations available that might make improvements to the programs specificity. D-Squared Cluster uses windowed (i.e. local) n-mer oligo frequency comparisons between a query sequence and a compressed database. Using a Cray Y/MP-48 supercomputer, 884 bases could be compared against 19 million bases in less than 1 min.

Rapid analysis of Pre-Indexed Data structures (RAPID: Miller et al., 1999; http://www.bioinf.man.ac.uk/rapid/) uses oligo frequencies to alter the significance of oligo matches before scoring particular pairwise 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.

Systems and methods

The JESAM sequence alignment calculation and storage code is written in ANSI C with a C++ wrapper
for the alignment server. Sequence alignments are stored in Berkeley DB libraries supplied by Sleepycat: http://www.sleepycat.com/db/. All the client software is written in JAVA and compiled using Sun Microsystems/Javasoft’s Javac JAVA compilers (http://www.javasoft.com/) and runs with the Java 1.1 class libraries and Java Virtual Machine JVM including the implementation in Netscape Communicator 4.5. The IDL interface specifications were compiled by Object Oriented Concepts’ (URL http://www.ooc.com/) ORBacus IDL to Java and IDL to C++ compilers which have free run-time ORB licenses. Many ORBs and IDL compilers are available free (URL: http://industry.ebi.ac.uk/~corba/) as are Sun’s Java and IDL compilers. Documentation is distributed throughout the Java source code in Javadoc comments.

Algorithm

Sequence overlap detection

A multi-stage pipeline was developed to discover all the arrangements of all pairs of sequences where the alignment could be consistent with the sequences being cognate and contiguous. The algorithm’s first stages reduce the total number of pairwise sequence alignments whilst aiming to maintain overlap sensitivity and alignment accuracy.

To ensure that the published alignments were both biologically useful and mathematically optimal, it was thought necessary to use a dynamic programming algorithm with a sophisticated gap penalty scheme (Altschul and Erickson, 1986). However, computation time makes it impractical to compare all sequences against each other for large EST datasets with workstation implementations of derivatives of the Smith–Waterman algorithm. Specialized hardware was an unacceptable solution due to perceived problems of cost, availability, portability, and ease of algorithm development. The JESAM alignment algorithm therefore uses dynamic programming only for the final alignments relying on the gross overall overlap being easy to find because the goal was only to discover potentially overlapping subsequences, not distant homologues mutated apart through millennia.

Alignment algorithm stages.

Load all non-overlapping 12-mers from all sequences into a hash table
Delete all low-complexity and over-abundant hash-table entries
For each sequence
  Using a 12-mer window sliding in single base steps down query
    Find the positions of all oligo matches retrieved using 12-mer key
  Sort all the oligo matches by relative start positions

(phase) to remove redundancy
Use the representative oligo matches to seed a windowed optimal alignment
Store all alignments consistent with contiguity

For all the alignments described in this paper, an affine gap scoring scheme (+5 match, −15 mismatch, −15 gap-start, −15 per-gap-base) was used with end gaps only penalized if not consistent with contiguity and an overall score of 200 needed for an alignment to be stored. To ensure sensitive alignments are calculated and stored, no predefined repeats are masked. Furthermore, repeat discovery is easier to do accurately after all alignments are calculated.

Supercluster creation

A simple clustering algorithm was chosen to summarize groups of overlapping sequences both as an extension of the alignment database and to prove the ease of writing client CORBA components. The algorithm aims to create a series of independent clusters through recursive calls that follow all alignments for every sequence. Each cluster ultimately contains all the sequences that can be linked by any series of stored overlaps to every cluster member. The clustering is performed without regard for any inconsistent alignment combinations making this algorithm much simpler and less selective than those used to produce highly optimized clustering databases like UniGene. These simple superclusters have a bias towards sensitivity and overclumping that aids further processing (e.g. reduced likelihood of data loss for cluster assembly) and directly offer a useful and easily understood benchmark to contrast against other clustering schemes.

Implementation

JESAM takes novel approaches to both sequence comparison and clustering: alignments are calculated using a PVM process farm, and published via CORBA. A Java program acts as a CORBA client to request alignments, and make sequence clusters and then turns itself into a CORBA server for the cluster data.

Alignment calculation and storage

All the sequences can be loaded from flat files or from a CORBA sequence server. The latter would be more convenient but the former offers higher performance with current sequence server implementations. All sequences are used to extract non-overlapping 12-mers which are recorded in a hash-table of linked lists. Abnormally common oligos and low-complexity oligos may have all their linked lists freed but no predefined repeats are masked. New sequences are retrieved in batches and overlapping 12-mers are used to look up oligo hits in the hash table. Oligo hit results are partitioned between

318
Table 1. Total time taken to align all 123,459 rat ESTs (in EMBL and EMBLnew on 15 September 1999, totaling 48,461,025 bases) against each other and then store the alignments when using different numbers of slaves. The master processes were run on a two-processor (400 MHz) UltraSPARC server and the slaves were run on separate (248 MHz) UltraSPARC servers all running Solaris 2.5 or 2.6.

<table>
<thead>
<tr>
<th>Number of slaves</th>
<th>10</th>
<th>20</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wall time (min)</td>
<td>328</td>
<td>171</td>
<td>107</td>
</tr>
</tbody>
</table>

Table 2. Peak Core Memory Usage (KBytes) for all processes creating the alignments for Table 1. All processes were sampled at 10 min intervals. Figures shown are for the largest memory usage by any instance of each process at any sample time so actual memory usage is less than shown.

<table>
<thead>
<tr>
<th>Number of slaves</th>
<th>10</th>
<th>20</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Master pvmd3</td>
<td>8360</td>
<td>29560</td>
<td>267280</td>
</tr>
<tr>
<td>jesam</td>
<td>26904</td>
<td>26784</td>
<td>25016</td>
</tr>
<tr>
<td>jesamscanner</td>
<td>191672</td>
<td>192656</td>
<td>193288</td>
</tr>
<tr>
<td>jesamdistributer</td>
<td>292912</td>
<td>309656</td>
<td>455432</td>
</tr>
<tr>
<td>Slave pvmd3</td>
<td>18224</td>
<td>23504</td>
<td>61144</td>
</tr>
<tr>
<td>jesamslave</td>
<td>20248</td>
<td>17328</td>
<td>21896</td>
</tr>
</tbody>
</table>

Table 3. Peak Virtual Memory Usage (KBytes) calculated as per Table 2.

<table>
<thead>
<tr>
<th>Number of slaves</th>
<th>10</th>
<th>20</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Master pvmd3</td>
<td>9000</td>
<td>30200</td>
<td>267960</td>
</tr>
<tr>
<td>jesam</td>
<td>27304</td>
<td>27184</td>
<td>27232</td>
</tr>
<tr>
<td>jesamscanner</td>
<td>192048</td>
<td>193032</td>
<td>193744</td>
</tr>
<tr>
<td>jesamdistributer</td>
<td>565128</td>
<td>588208</td>
<td>757264</td>
</tr>
<tr>
<td>Slave pvmd3</td>
<td>18560</td>
<td>24000</td>
<td>61640</td>
</tr>
<tr>
<td>jesamslave</td>
<td>20712</td>
<td>17576</td>
<td>22120</td>
</tr>
</tbody>
</table>

Results and discussion

JESAM Performance

Performing EST comparisons using JESAM can be fast and inexpensive due to the underlying design: storing results reduces computational needs, and PVM can spread the incremental calculations across a network of cheap personal computers, or idle workstations. A total of greater than 120,000 ESTs were aligned and clustered in 5 hours, using a total of 11 processors (UltraSPARC and Compaq Alpha at EBI, all UltraSPARC at Cereon). Scaling efficiency has been tested in detail (Tables 1, 2 and 3) with up to a hundred slaves which do a banded Smith–Waterman-based sequence comparison looking for matches consistent with sequence overlap. The parallel computer architecture and all affine gap penalties can be independently adjusted on the command line. The slaves receive computational tasks via Parallel Virtual Machine (PVM) message transfers which are asynchronous and fully buffered. Buffering is controlled by requiring feedback to the oligo scanner and limiting the number of outstanding comparisons. To spread the network load, slaves ask for batches of sequences from a sequence server only when needed and then cache their set locally (Figure 1). The slaves may need a large amount of virtual memory (to cache sequences: 1 byte per base), but little RAM (typically 10–20 Mb) for the actual comparisons because backtracking memory usage grows linearly with sequence length. The largest memory requirements are for the oligo hash table and the Berkeley DB libraries which both have a random data access pattern: see Tables 2 and 3 for actual figures. Cache sizing is configured on the command line. The minimum cache size is impossible to estimate without experimentation. A larger cache is needed for alignments of very repetitive/redundant sequence sets (more matches), slower disk access and for larger PVM configurations.

The distributed architecture of JESAM provides a flexible framework to allow extension to more computationally intensive sequence comparisons (for example base-call error probabilities) and could also support other costly calculations, such as assembly contig building via directed graph explorations. The PVM slaves running sequence comparisons are typically CPU limited so the performance of the PVM implementation on networks of cheap uniprocessor PCs is equivalent to that of more expensive multi-processor servers.

Updates do not require recomputation of known alignments because all sequence overlaps are automatically stored by the Berkeley DB routines. If a new alignment crosses a user-determined threshold score then it is saved, complete with the position of all indels, starts and ends. The alignments are offered by a CORBA interface provided by a C++ wrapper around the same ANSI C code used to create the alignments (Figure 2).

A Java client recursively steps through the matches to make superclusters which are stored in a serialized Java Hashtable object which loads and saves quickly. The Supercluster server took only one afternoon to write thanks to the alignment data CORBA interface, simple algorithm, and the innate productivity of Java. Both the alignment and supercluster data can be viewed interactively using web-accessible Java applets or larger datasets can be saved to client file systems for use in other file-based scripts etc. The JESAM clusters have been integrated with other databases at the European Bioinformatics Institute, see URL http://corba.ebi.ac.uk/EST for browser demonstrations.
to 40 slaves on a small dataset (120,000 ESTs) but larger datasets allow greater scaling efficiency (PVM daemon, or network communication restrictions will appear with less than 100 slaves with many PVM configurations). The European Bioinformatics Institute can maintain fully updated alignment and cluster databases (with their Internet accessible interfaces) for four model organisms (mouse, rat, *Aspergillus nidulans*, and Zebrafish) using only six distributed processors (two in server, four in separate slaves) for one evening every 2 weeks. These on-line databases are accessible via multiple applets to allow browsing of clustering statistics, pairwise alignments, cluster details, and downloads of FASTA format sequences for individual clusters (Figure 3).

**Clustering comparisons**

A full comparison of all the mentioned EST alignment and clustering programs on multiple representative datasets is a formidable task and certainly beyond the scope of this paper. Instead, samples of UniGene mouse and rat databases were studied by comparison to superclusters and alignments produced by JESAM. The UniGene databases are frequently referenced and important resources that have guided many large-scale experiments, so an investigation into their contents has potentially significant implications. An initial investigation suggested it was impossible to use identical datasets for this comparison because the UniGene databases include sequences which have been intentionally deleted from the EMBL and GenBank (Benson et al., 1998) nucleotide databases and were therefore not publicly retrievable. (A separate ftp site for this information (ftp.ncbi.nlm.nih.gov/repository/UniGene/) was, however, identified in review.) Unfortunately, the non-identical input sequences led to frequent differences in the cluster sets, which complicated and limited the scale of investigations into the basis of the overall cluster discrepancies. It is worth noting that the more intensive EST/genomic comparisons performed by Wolfsberg and Landsman were limited (by the inherent complexity of the underlying data) to analysing just 15 genes and their study concluded that this small sample gave indications of uncharacterized alleles, partial splicing, undocumented splicing, uncoupled clones, and possibly novel biological phenomena!

The UniGene investigation detailed in Tables 4 and 5 was small but it sampled examples of many of the known or expected properties of the UniGene databases, and some of the advantages and disadvantages of JESAM's simplistic superclustering client. UniGene rat excluded 20,000 sequences included in the corresponding JESAM alignments and clusters, and UniGene mouse similarly excluded 150,000 sequences. The two rat databases were generally more similar than the two mouse databases (data not shown). Perhaps most significantly, JESAM both clustered more sequences and created more clusters for both data sets despite the use of genomic clones in UniGene (excluded from JESAM) that helped clustering in two of the 10 mouse clusters examined (but no rat cluster). However, one rat UniGene cluster (Rn. 34368) did probably correctly combine two large divergent cDNAs which may encode alternative transcripts that JESAM rejected as being clearly non-overlapping sequences. The worst feature of the current published JESAM superclusters was demonstrated with the 12 sequences in UniGene Mm. 2477 which were combined into one supercluster of size 116,977 with many other sequences presumably joined on repetitive DNA. This supercluster would not form with a more selective scoring scheme but sensitivity would then be lost.

---

Fig. 1. Diagram showing how PVM links are used to distribute data and calculations across a number of processes which may be running on different processors in different computers.

Fig. 2. Diagrammatic representation of CORBA alignments code generation where an IDL file (centre) is used to create both the server skeleton code (C++), and the client stub code which in this case is Java. Other vendor’s CORBA products could replace the examples shown.
Table 4. Comparison between rat UniGene and EBI Superclusters. First 10 ‘interesting’ (defined by UniGene) clusters taken from largest (best sampled) rat cDNA library: Lib.53, UI-R-C0 containing 7024 sequences. UniGene build 42, uploaded on 17 February 1999 (80 755 sequences, 23 003 clusters including 9520 orphans). EBI superclusters were made on 19 February 1999 (101 383 sequences, 31 863 clusters, including 17 765 orphans). Comparison done on 22 February 1999. NR = not retrievable from EMBL, GenBank, or dbEST (i.e. withdrawn entries, or not available at the time of the study—Nicole Redaschi, personal communication)

<table>
<thead>
<tr>
<th>UniGene cluster ID</th>
<th>UniGene cluster size</th>
<th>Supercluster size</th>
<th>Missed ESTs</th>
<th>Best missed ESTs</th>
<th>Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rn.12107</td>
<td>15</td>
<td>17</td>
<td>AI228202, AI137581, AA997212, −AA963421</td>
<td>AI228202 against AI23460, 124 base overlap with three mismatches</td>
<td>AI228202 was prepended, for unknown reasons, with five extra bases ‘actaa’ on 9 January 1999. These five bases reduce the match quality. AA963421 was NR AA998090 contains a low-complexity region that may have been masked in UniGene. AI029890 was NR so other two form a separate cluster</td>
</tr>
<tr>
<td>Rn.32936</td>
<td>10</td>
<td>11</td>
<td>AA998090</td>
<td>AA998090 against X53725, 138-base overlap with two mismatches</td>
<td>None</td>
</tr>
<tr>
<td>Rn.12821</td>
<td>7</td>
<td>4</td>
<td>−AB029388, −AI045806, −AB029890, AA899962, AI236745, AA54665, AI170199, AA945230, AA043491, −AI029893</td>
<td>AI236745 against AA963979, 75-base overlap with one indel, and one mismatch</td>
<td>Rn.7578 with extra sequence Rn.03576 which was NR AI029893 was NR AB020504 has replaced AB006882</td>
</tr>
<tr>
<td>Rn.11720</td>
<td>8</td>
<td>13</td>
<td>AA998090</td>
<td>AA998090 against X53725, 138-base overlap with two mismatches</td>
<td>None</td>
</tr>
<tr>
<td>Rn.13083</td>
<td>6</td>
<td>7</td>
<td>AB006882</td>
<td>AB006882 against AB020504, 510 base overlap with four mismatches</td>
<td>AB020504 has replaced AB006882</td>
</tr>
<tr>
<td>Rn.12740</td>
<td>10</td>
<td>12</td>
<td>AI407336, AI407876</td>
<td>AI407336 against AI138051, 384 base overlap with one mismatch</td>
<td>Both entered GenBank on 9 February, and EMBL on 11 February 1999 but not in dbEST version used (on 13 February according to release notes) for UniGene rat.</td>
</tr>
<tr>
<td>Rn.11948</td>
<td>6</td>
<td>9</td>
<td>AI406632, AI408422, I409932</td>
<td>AI406632 against AA997064, 454 base overlap with one indel and five mismatches</td>
<td>Too recent to appear in this UniGene release</td>
</tr>
<tr>
<td>Rn.9416</td>
<td>8</td>
<td>33</td>
<td>Rn.2375</td>
<td>AB020424 against U35775, 137 base overlap with one indel and one mismatch</td>
<td>AI029424 appears to join two alternative 3’ ends?</td>
</tr>
<tr>
<td>Rn.8514</td>
<td>6</td>
<td>7</td>
<td>AI407620</td>
<td>AI407620 against AI048791, 392 base overlap with one indel and 15 mismatches</td>
<td>Too recent to appear in this UniGene release</td>
</tr>
<tr>
<td>Rn.34368</td>
<td>4</td>
<td>2</td>
<td>−AI043650, −AA964642</td>
<td>AA998702 matches AI02983 correctly</td>
<td>Sequences share a pattern of local similarities inconsistent with contiguity. UniGene cluster may represent alternative splicing variants.</td>
</tr>
</tbody>
</table>

Biasing towards sensitivity minimizes information loss yet still significantly reduces the problem scale for later cluster partitioning software components. With all the alignments available on-line, it should be relatively easy to develop and test some quite sophisticated partitioning or assembly algorithms so these superclusters should not be seen as final answers in contrast to the other EST cluster databases which cannot be programmatically interrogated.

Extensions

Memory usage could be drastically reduced by switching to a partitioned sequence database: this would both reduce the number of sequences and oligos in memory at any one time and also reduce the subset of sorted alignment results that would need to be cached for random update access. With this enhancement, JESAM could scale easily
Table 5. Comparison between mouse UniGene and EBI Superclusters. First 10 ‘interesting’ (defined by UniGene) clusters taken from largest (best sampled) mouse cDNA library: Lib.30, Soares Mouse Embryo NIME13.5 containing 39,996 sequences. UniGene build 46, uploaded on 15 February 1999 (268,247 sequences, 15,275 clusters including 2,761 orphans). EBI superclusters were made on 23 February 1999 (424,080 sequences, 65,588 clusters including 42,418 orphans). Comparison done on 24 February 1999. NR = not retrievable from EMBL, GenBank, or dbEST at the time of the study.

<table>
<thead>
<tr>
<th>UniGene ClusterID</th>
<th>UniGene Cluster size</th>
<th>EBI Cluster size</th>
<th>Missed ESTs</th>
<th>Best Missed</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mm.2496</td>
<td>25</td>
<td>30</td>
<td>AA048065, AA198448, AA230545, AA967785, W58964, W88178, W98102, L27220, AI425958</td>
<td>AA967785 against AA218283, 432 base overlap with two indels and 29 mismatches</td>
<td>All seven missing ESTs were derived from poor-quality traces and therefore excluded from UniGene. L27220 is genomic DNA. AI425958 was NR</td>
</tr>
<tr>
<td>Mm.11285</td>
<td>14</td>
<td>12</td>
<td>AA052635, −AI427383, −AI430589, −AI431070</td>
<td>AA052635 against AA058152, 140 base overlap with one indel and five mismatches.</td>
<td>AA052635 is only 171 bases long and contains a low complexity region. AI427383, AI430589, and AI431070 were NR</td>
</tr>
<tr>
<td>Mm.3501</td>
<td>11</td>
<td>11</td>
<td>AF013117, AA693148, −AI43091, −AI430109</td>
<td>AF013117 against D49544, 465 base overlap with five mismatches.</td>
<td>AF013117 has been excluded for unknown reasons. AA693148 is annotated as low quality. AI43091 and AI430109 were NR</td>
</tr>
<tr>
<td>Mm.3277</td>
<td>12</td>
<td>6</td>
<td>−AA030514, −AI71281, −AI323427, −AI427499</td>
<td>−</td>
<td>Z22923 is genomic DNA so AA030514, AA171281, AI323427, AI427499 were split off into separate clusters.</td>
</tr>
<tr>
<td>Mm.30219</td>
<td>9</td>
<td>12</td>
<td>AI317572, W78547, W66677, AA235757, AA510343, AI81413, −AI430530, −AI110526, −AI429692</td>
<td>AI317572 against AA049045, 314 base overlap with perfect match</td>
<td>AI317572, W78547, W66677, AA235757, AA510343, AI81413, were excluded for unknown reasons. AI30530 and AI429692 were NR. AA110526 is only 84 bases long</td>
</tr>
<tr>
<td>Mm.20964</td>
<td>11</td>
<td>13</td>
<td>AA450931, W64306, W64209, AA268149, AA170119, −M36777, −W43961</td>
<td>W64306 against W64299, 238 base overlap with two indels.</td>
<td>AA450931, W64306, W64209, AA268149 were excluded for unknown reasons. M36777 and M36778 are two alternatively spliced mRNAs that differ in over 1 kb and are non-overlapping.</td>
</tr>
<tr>
<td>Mm.3127</td>
<td>11</td>
<td>16</td>
<td>AA271310, AA028577, AA028535, AA062181, AI414081</td>
<td>AA271310 against W38289, 263 base overlap with two indels, and eight mismatches.</td>
<td>AA271310, AA028577, AA028535, and AA062181 were excluded for unknown reasons. AI414081 was entered on 11 February 1999</td>
</tr>
<tr>
<td>Mm.2477</td>
<td>12</td>
<td>9</td>
<td>−AI430492, −AI430463</td>
<td>−</td>
<td>Joined on repetitive DNA</td>
</tr>
<tr>
<td>Mm.4758</td>
<td>9</td>
<td>116977</td>
<td>−X86368, −AI430408, −AI429593</td>
<td>−</td>
<td>Possibly matching on low-copy repetitive DNA. X86368 is genomic DNA, AI430408 and AI429593 were NR</td>
</tr>
<tr>
<td>Mm.30224</td>
<td>7</td>
<td>5</td>
<td>−AI430492, −AI430463</td>
<td>−</td>
<td>AI430492 and AI430463 were NR</td>
</tr>
</tbody>
</table>

to any $N^2$ sequence comparison, but also shrink to fit any parallel architecture. If access to the oligo hash table became limiting with large numbers of processors, the single oligo hash table look-up function could be rewritten to use multi-threading, but it would then have to run on a multi-processor computer. If greater speed were needed on very limited hardware, the sequence comparison could be significantly optimized relative to the current generic implementation, and multiple oligo hits per pair of sequences (currently just a single 12-mer) could be required before launching a slave comparison (the existing code currently used to prevent multiple searches on the same diagonal could be easily extended for this purpose).

The mouse EST and mRNA clusters (downloaded directly from the CORBA servers) have been successfully assembled using CAP3 by Juha Muilu (personal communication) as a useful extension of the JESAM superclusters. However, CAP3 ignores the pairwise alignment data so an opportunity still exists for a more integrated, higher performance assembly tool tailored...
to EST data. A suitable post-assembly goal would be something like the Pangea CRAW tools (Burke et al., 1998; Chou and Burke, 1999) which demonstrate how complicated patterns of expression variations can be summarized and presented in a Java applet alongside supporting analyses.

Linking the EST clusters with other databases would make them more valuable. The EMBL outstation European Bioinformatics Institute (EBI) amongst others (including the OMG’s Life Science Research (LSR) and Domain Task Force (DTF); URL: http://lsr.ebi.ac.uk/) are working towards standards for object interfaces and components for many kinds of biological data. Some of the EBI’s efforts include physical and genetic maps (Jungfer and Rodriguez-Tome, 1998), annotated DNA and protein sequences, and sequence traces (Parsons et al., 1999).

JAVA RMI could have offered a CORBA alternative but was not investigated due to the lack of relevant biological standards efforts, frameworks, language independence, services and local support. AcePerl (Stein and Thierry-Mieg, 1998), is another potential alternative.

**Conclusion**

As explained in the introduction, no EST clustering database can be expected to provide a perfect view of an organism’s genes, but the imperfect result is still valuable. The human UniGene database has been a notably effective resource for many large and varied scientific studies. However, this small study shows the mouse and rat UniGene clusters have questionable features: the databases sometimes contain deleted sequences, and conversely may exclude apparently unique and high-quality sequences, and many with perfect matches to those included sequences. The simple JESAM superclusters have problems of their own and consequently complement rather than supersede the UniGene databases but this is a significant role. Furthermore, the public CORBA interface to model organism alignment data offers great potential for future enhancements through new public software components interacting over the Internet, or (since all the code is available and portable) combining public and private data behind corporate firewalls.

The use of CORBA interfaces in the JESAM package requires more software libraries (and greater expertise to install and maintain them) than would a normal file-based Input/Output system but they also offer greater flexibility. Furthermore, by maintaining all information accessible online, it is possible to give users an evidence trail that they can follow all the way from cluster, to alignment, to sequence and even back to the original sequence trace.

Currently, there are a few problems deploying CORBA-based applications over the Internet. These problems include: firewalls blocking the IIOP protocol; the need to download ORB classes to clients due to the lack of a guaranteed local ORB; a lack of support for multiple applet signing which would allow applets to follow
object references to objects on computers other than the original applet’s host. Some of the Java and CORBA problems should decrease once JAVA 2 from Javasoft (URL:http://www.javasoft.com/) becomes a standard component of every client workstation. Amongst the many improvements over Java 1.1 is a built-in CORBA ORB allowing distributed computing to extend out of the Web browser and directly into a user’s general molecular biology application set.

Acknowledgements

The authors are grateful to Tom Flores for helping to start the CORBA element of this project, and would also like to thank David Starks-Browning at the EBI, and the Systems staff at Cereon for their support. This work was funded by EU grant BIO 4 CT 960346.

References


