Sequence Assembly

In bioinformatics, sequence assembly refers to aligning and merging fragments of a much longer DNA sequence in order to reconstruct the original sequence. This is needed as DNA sequencing technology cannot read whole genomes in one go, but rather reads small pieces of between 20 and 1000 bases, depending on the technology used. Typicallythe short fragments, called reads, result from shotgun sequencing genomic DNA, or gene transcript (ESTs).

The problem of sequence assembly can be compared to taking many copies of a book, passing them all through a shredder, and piecing the text of the book back together just by looking at the shredded pieces. Besides the obvious difficulty of this task, there are some extra practical issues: the original may have many repeated paragraphs, and some shreds may be modified during shredding to have typos. Excerpts from another book may also be added in, and some shreds may be completely unrecognizable.

Genome Assemblers

The first sequence assemblers began to appear in the late 1980s and early 1990s as variants of simpler sequence alignment programs to piece together vast quantities of fragments generated by automated sequencing instruments called DNA sequencers. As the sequenced organisms grew in size and complexity (from small viruses over plasmids to bacteria and finally eukaryotes), the assembly programs used in these genome projects needed to increasingly employ more and more sophisticated strategies to handle:

- Terabytes of sequencing data which need processing on computing clusters.
- Identical and nearly identical sequences (known as repeats) which can, in the worst case, increase the time and space complexity of algorithms exponentially.
- Errors in the fragments from the sequencing instruments, which can confound assembly.

Faced with the challenge of assembling the first larger eukaryotic genomes, the fruit fly Drosophila melanogaster, in 2000 and the human genome just a year later, scientists developed assemblers like Celera Assembler and Arachne able to handle genomes of 100-300 million base pairs. Subsequent to these efforts, several other groups, mostly at the major genome sequencing centers, built large-scale assemblers, and an open source effort known as AMOS was launched to bring together all the innovations in genome assembly technology under the open source framework.

EST Assemblers

Expressed Sequence Tag or EST assembly differs from genome assembly in several ways. The sequences for EST assembly are the transcribed mRNA of a cell and represent only a subset of the whole genome. At a first glance, underlying algorithmical problems differ between genome and EST assembly. For instance, genomes often have large amounts of repetitive sequences, mainly in the inter-genic parts. Since ESTs represent gene transcripts, they will not contain these repeats. On the other hand, cells tend to have a certain number of genes that are constantly expressed in very high amounts (housekeeping genes), which again leads to the problem of similar sequences present in high amounts in the data set to be assembled.

Furthermore, genes sometimes overlap in the genome (sense-antisense transcription), and should ideally still be assembled separately. EST assembly is also complicated by features like (cis-) alternative splicing, trans-splicing, single-nucleotide polymorphism, recoding, and post-transcriptional modification.

De-novo vs. Mapping Assembly

In sequence assembly, two different types can be distinguished:

- De-novo: assembling short reads to create full-length (sometimes novel) sequences (see de novo transcriptome assembly)
- Mapping: assembling reads against an existing backbone sequence, building a sequence that is similar but not necessarily identical to the backbone sequence

In terms of complexity and time requirements, de-novo assemblies are orders of magnitude slower and more memory intensive than mapping assemblies. This is mostly due to the fact that the assembly algorithm needs to compare every read with every other read (an operation that has a complexity of O (n^2) but can be reduced to O (n log(n)).