Efficient storage of high throughput DNA sequencing data using reference-based compression (Summary)

Analysis of DNA sequencing data has a growing need of handling its storage in an efficient manner. It is the first molecular data for which storage costs make up a large portion of the overall analysis costs. Three proposed approaches are to add storage, throw away some data or compress the stored data. Adding storage seems unlikely and human samples are non-renewable so permanent electronic archives are preferred; i.e. the most feasible approach is to compress the stored data.

DNA's natural representation is a string of characters so it can be compressed using generic methods. Its compressibility can benefit from biological components such as repeat content and relationship to existing sequence. Two such works are DNACompress and DNAzip. A new and more efficient method is that of reference-based compression. It is primarily based on compressing whole genome information due to the progressive area of study/research.

The lossless reference-based compression method is based on the idea of efficiently storing identical or near-identical input sequences. The reference genome is used as a compression framework where new sequences which are identical to the reference have little impact on storage. Most reads match the sequence perfectly or near-perfectly so a mapping of the reads against the sequence is taken with the aid of Golomb codes to store relative encoded read positions and Huffman coding to compress length of reads. This technique's efficiency increases proportionally to the read length irrespective of the coverage and it increases proportionally with the coverage as well. It compares well with bzip2 compression and is more efficient than BAM-based storage.

Unmapped reads cannot be stored in the same manner so the idea of a 'useful' compression framework is implemented. These unmapped reads are pooled together to serve as said secondary framework by use of a De Bruijn graph framework. Base quality scores are stored relative to if their positions show variation (if not, they are not stored) along with a user-defined percentage, both being compressed using Huffman coding. The lowest base qualities which are identical to the reference are stored to ensure higher compression at longer read lengths.

All positions are relative encoded and stored as a Golomb code. As for size, exact read positions (encoded using strand and match flags) take up 1 bit each. Inexact matches are stored using a list of variations (read position, variation type and other information) with the variation type taking up 1 or 2 bits. Given any base, a substitution takes up 2 bits. Inserted quality scores take up 2 or 3 bits.

Hence, the compression method described constructs an explicit balance between storage cost and data precision. It provides efficient lossless compression for read alignments with little or no differences to the reference genome at approximately 0.02 bits/base. As for unmapped reads, it is still uncertain whether they should be stored or removed due to storage constraints (the decision being based on the data set being analysed). The implementation is still a prototype for BAM-based input however it can be improved according to some complex details with the necessary effort.