FQSqueezer: *k*-mer-based compression of sequencing data

Abstract

Modern sequencing instruments create a massive amount of data that must be processed. the advanced data compression of FASTQ files has gotten a lot of attention. Current algorithms, on the other side, are still flawed, and the best tools create very large archives. FQSqueezer is a novel data compression algorithm capable of processing single- and paired-end reads of varying lengths. It is based on the well-known prediction by partial matching and dynamic Markov coder algorithms from the world of general-purpose compressors. The compression ratios are often tens of percent higher than what state-of-the-art software can provide. Large memory and time requirements are disadvantages of the proposed system.

Introduction

Genome sequencing has progressed to the point that it is now a mature technology with various medical applications. Illumina's instruments generate the vast majority of available data at a low cost. The sequenced reads are short but of high quality. The PacBio or Oxford Nanopore 3rd generation instruments can produce much longer readings, but at a much lower throughput and with much lower efficiency.

The big numbers provided directly lead to the inference that just storing and transferring sequenced (and mapped) reads would cost a lot of money in the near future, and may be a major factor in overall sequencing costs .The use of gzip, a general-purpose compressor, was an obvious first move. The key issue with gzip is that it was developed primarily for textual data with redundancy forms that are similar to textual data. Unfortunately, read sets rarely contain items like data repetitions over short distances.

The next move was to create specialized algorithms that took into account the different types of redundancies that FASTQ files had. the best algorithms were able to reduce the space needed for DNA bases by about 5 times. it appeared that quality value compression was even more difficult quality scores became responsible for a dominant part of the compressed archives. One of the simplest strategies was to reduce the resolution of quality scores. Illumina in their HiSeq sequencers restricted the quality scores to eight values, and then in the NovaSeq instruments to just four values. the quality of sequencing is currently very good and the prices of sequencing are low. it is easier (and cheaper) to perform sequencing with a bit larger coverage than store high-resolution quality scores.

In this article, we propose a novel compression algorithm for FASTQ files. The main novelty is in the compression of DNA bases, for quality scores and read identifiers we follow similar strategies. Our algorithm, FQSqueezer, make use of the ideas from the prediction by partial matching (PPM) and dynamic Markov coder (DMC) general-purpose methods. the PPM-like strategy to sequencing reads would be, very hard and likely unsuccessful. There are four main reasons for that. First, PPM algorithm should construct a dictionary of all already seen strings of length up to some threshold. Second, the PPM algorithms often need many accesses to the main memory. Third, sequenced data contain errors that should be corrected. Fourth, the PPM algorithms usually learn slowly.

To address these issues, we created a collection of fixed-k dictionaries for k-mers (k-symbol long substrings) We use a much more aggressive approach to estimating the likelihood of symbols appearing. As a result, compression is greatly improved. Finally, we perform some kind of error correction when storing k-mers in dictionaries.

Proposed ideas are that longer (assembled) fragments are of much better quality than reads, so small differences between very similar fragments are usually due to variations between organisms. In sequencing datasets, we usually work with reads originating from a single genome and differences are in majority due to sequencing errors. A compressor of sequencing data should be designed in a different way than a compressor of assembled genomes or its parts. We greatly developed the previously listed ideas and suggested new approaches in this article. The most significant are the organizing of large dictionaries and the design of PPM-like probabilistic estimation. We designed the entire FASTQ compressor system.

The main asset of FQSqueezer is its compression ratio. It is a few times slower than the mentioned competitors in compression and much slower in decompression. Our tool has, however, also some drawbacks in terms of speed and memory usage.