Bloocoo, a memory efficient read corrector

Gaëtan Benoit[1], Dominique Lavenier[1], Claire Lemaitre[1] and Guillaume Rizk[1]

[1] Inria/IRISA GenScale, Campus de Beaulieu, 35042 Rennes cedex, France

Next generation sequencing technologies generate a high amount of short DNA sequences, but may contain imperfections. Some applications, such as assembly, yield better results with high quality data, triggering the need for short-read correction software. Most methods proposed in the past do not scale well when dealing with a large dataset, often requiring a very large amount of memory.

Bloocoo is a k-mer-spectrum based read error corrector. It relies on k-mer frequency to discriminate between correct solid k-mer and k-mer containing sequencing errors. This is a traditional approach used by many read correctors. Bloocoo distinguishes itself by requiring an order of magnitude less memory than other state-of-the art correctors, while still providing equivalent correction. This is achieved thanks to a constant-memory k-mer counting algorithm, and a bloom filter to store solid k-mers.

The read correction workflows consists in 3 main steps as follows:
1) k-mer counting.
2) Insertion of solid k-mer in a bloom filter.
3) Multi-stage read correction.

The first stage is the constant-memory k-mer counting step provided by the DSK counter by Rizk et al. It outputs on disk the list of solid k-mer, i.e. k-mers with a high enough abundance.
The second step is the insertion of all solid k-mers in a bloom filter. This probabilistic data structure allows to know if a given k-mer is solid, using only 11 bits per solid k-mer inserted, at the cost of introducing some false positives.

The correction step is a multi-step approach largely similar to the Musket correction algorithm by Liu et al. All k-mers of a read are queried in the bloom filter and classified as solid or non- solid k-mers. Errors are then corrected through multiple iterations of two-sided conservative, one-sided aggressive and voting-based correction algorithm. To neutralize the effect of false positive solid k-mers due to the bloom filter, errors are corrected only if several solid k-mers cover the corrected nucleotide, greatly reducing the risk that all of them are false-positives and induce false correction.

Bloocoo was also designed for fast correction. First, dispatching blocks of reads to several threads is an easy way to parallelize the correction procedure. Secondly, the bloom filter used is a blocked bloom filter, which greatly increases performance thanks to higher cache coherence. Tests were conducted on simulated datasets with different error rates and genomes against state-of the art competitors. For example, with reads from human chromosome 1 with 1% error rate and 70x coverage, Bloocoo completed correction in 1390 sec using 740 MB of memory, while Musket required 5330 sec and 12190 MB of memory. Quality of correction was 90.28 % recall / 96.93 % precision for Bloocoo and 90.92 % recall / 97.86 % precision for Musket. Bloocoo correction is roughly the same as Musket, while taking 16x less memory and 3.8x less time.