Abstract
The rapid development of genomic sequencing technologies in the past decades has outgrown the advances of computing power and therefore requires efficient read mapping algorithms.2 Read mappers align sequenced reads to a reference genome, where a set of reads aligned to the same position can hint at possible mutations. Even though many fast read mappers have been published in the recent years, most of them do not consider common variants of the reference genome. Variant tolerance highly increases accuracy of read mappers when aligning reads against a species’ pangenome. We have developed a new read mapper for variant tolerant alignment by usage of hash based filtering in combination with an alignment algorithm based on dynamic programming. In the first step, we use locality-sensitive hashing (LSH), initially designed for finding similarities in documents, for candidate filtering.2,3 We treat reads and windows of the reference genome as documents, which are compared by LSH. As a result, we obtain an approximative mapping to the reference regions. This leads to a dramatic reduction of the reference length and therefore semi-global alignment becomes feasible. The aligner handles variants like SNPs, insertions and deletions and decides which variants lead to the best alignment. New genetic variants and gene mutations can be found by observing the mismatches from the alignments.

Mapping - LSH algorithm
Utilization for read mapping:
1. Split references into half-overlapping windows. These will provide our documents for locality-sensitive hashing.
2. Calculate set of q-grams for every document.
   In case of genomic variants, the algorithm computes all combinations of q variants inside a window of q characters.
   The q-grams, which are derived from all of these combinations, are added for the corresponding document. If the number of combinations exceeds a certain limit, variants are ignored inside the window.
3. Calculate union of the q-gram sets.
4. Permute q-gram indices.
5. Save index of the first existing q-gram in the current document according to the generated permutation.
6. Create signatures by splitting the signature matrix into bands.
7. Signatures within a band represent the keys for the hash buckets.
8. Create signature matrix for reads analogously.
9. Reads whose signature collide with a signature from a reference document may be interpreted as putatively mapping to the corresponding reference region.

Semi-global Alignment
The semi-global alignment algorithm is based on dynamic programming. Here is an example for a nucleotide string aligned against a read:
Alignment: AATAGACCT
T GAC
Our improvement to the algorithm is that it aligns a read against multiple variants of a reference string.

Variants

\[
\begin{align*}
\text{AGACGTATG} & \quad \text{AGACGTATG} \\
\text{AGCTATGT} & \quad \text{AGCTATGT} \\
\text{AGACGTGT} & \quad \text{AGACGTGT} \\
\text{AGACGTAGT} & \quad \text{AGACGTAGT}
\end{align*}
\]

Since the variants are stored as a difference to the reference, only one (modified) alignment process suffices.

References