Genome analysis

FastRemap: a tool for quickly remapping reads between genome assemblies

Jeremie S. Kim 1,*, Can Firtina 1, Meryem Banu Cavlak 1, Damla Senol Cali 2, Can Alkan 3 and Onur Mutlu 1,3

1Department of Computer Engineering, ETH Zurich, D-ITET, Zurich 8006, Switzerland, 2Department of Computer Engineering, Bionano Genomics, San Diego, CA 92121, USA and 3Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

*To whom correspondence should be addressed.

Associate Editor: Inanc Birol

Received on January 18, 2022; revised on July 7, 2022; editorial decision on August 8, 2022; accepted on August 10, 2022

Abstract

Motivation: A genome read dataset can be quickly and efficiently remapped from one reference to another similar reference (e.g., between two reference versions or two similar species) using a variety of tools, e.g., the commonly used CrossMap tool. With the explosion of available genomic datasets and references, high-performance remapping tools will be even more important for keeping up with the computational demands of genome assembly and analysis.

Results: We provide FastRemap, a fast and efficient tool for remapping reads between genome assemblies. FastRemap provides up to a 7.82 × speedup (6.47 ×, on average) and uses as low as 61.7% (80.7%, on average) of the peak memory consumption compared to the state-of-the-art remapping tool, CrossMap.


Contact: jeremiekim123@gmail.com

1 Introduction

Genome read datasets that have already been mapped to one reference must often be remapped to another reference. This is especially important for cases where (i) a more accurate reference genome is released or (ii) the read dataset was previously mapped to a reference genome of the wrong species. Several remapping tools (Gao et al., 2018, pyliftover; Kuhn et al., 2013, genome.ucsc.edu/cgi-bin/hgLiftOver; Zhao et al., 2014, crossmap.sourceforge.net; ncbi.nlm.nih.gov/genome/tools/remap; pyliftover) exist for this purpose and provide a means for remapping a read dataset from one (source) reference to another similar (target) reference (e.g., between two reference versions or two similar species). CrossMap (Zhao et al., 2014) is among the most commonly used since it supports remapping commonly-used BAM/SAM files as well as many other data file formats used in genome analysis (e.g., BED, VCF). CrossMap provides significant speedup compared to fully mapping a read dataset to the target reference using a conventional read mapping (not remapping) tool [e.g., BWA-MEM (Li, 2013)], since CrossMap exploits the knowledge of shared (i.e., identical) genomic regions and their positions within each of the source/target pair of references in order to quickly update a read’s mapping position. If a read maps within a shared region of the source reference, the read will map within the same shared region (with the same offset into the region) of the target reference, and thus a remapping tool can infer the read’s mapping position in the new reference.

Despite its wide use in the area (Luu et al., 2020), CrossMap has several important shortcomings. First, it is not optimized for performance, leaving much to be desired in terms of remapping speed. We believe speed of remapping tools is critical due to continued explosion of available genomic data, frequent improvements to existing references and the prohibitive computational power that would be required to completely rely on read mapping tools to repeatedly map all read datasets to the most recent reference genome versions. Second, we find that the immediate BAM/SAM output of CrossMap is incompatible with downstream analysis tools [e.g. GATK HaplotypeCaller (McKenna et al., 2010), Strelka2 (Kim et al., 2018), Platypus (Rimmer et al., 2014)]. Enabling compatibility is critical to accurately understand the final mapping results.

In order to address the key limitations of CrossMap, in this work, we provide FastRemap, a new C++ implementation of the BAM/SAM remapping component of CrossMap with key modifications that enable downstream analysis on its immediate outputs. In our evaluation of FastRemap and CrossMap on three different sizes of reference genomes (i.e. human, Caenorhabditis elegans and yeast), we find that FastRemap provides up to 7.82 × speedup (6.47 ×, on average) and uses as low as 61.7% (80.7%, on average) of the peak memory consumption compared to the state-of-the-art remapping tool, CrossMap.
FastRemap is open source and all scripts needed to replicate the results in this paper can be found at https://github.com/CMU-SAFARI/FastRemap. FastRemap is also available at conda in the biocconda channel as fastremap-bio (Gruning et al., 2018).

2 Features and methods

To remap reads from one (source) reference to another (target) reference, FastRemap relies on a chain file (specific to the pair of references), which indicates regions that are shared between the two references. The chain files that we use are available on the UCSC genome browser (hgdownload.soe.ucsc.edu/downloads.html). We model our codebase off of CrossMap, such that it is easily extensible to other data file formats. We currently support remapping reads in the most commonly used formats (i.e., SAM/BAM, BED). However, we plan to extend support for additional file formats (e.g., VCF, GTF/GFF, BigWig, MAF).

Libraries. We use the Seqan2 library (Knut et al., 2017) for manipulating genomic data and zlib (github.com/madler/zlib) for enabling the input and output of compressed files (i.e., BAM format).

Upgrades to the CrossMap implementation. In addition to providing a faster C++ implementation, we make two key changes to the CrossMap code logic to enable accurate and immediate downstream analysis on FastRemap’s output. First, CrossMap removes the supplementary alignment flag when remapping BAM/SAM files. These incorrect flag values cause Picard’s MarkDuplicates tool (gatk.broadinstitute.org/hc/en-us/articles/360037052812-MarkDuplicates-Picard) to fail, which prevents further downstream analysis beyond the MarkDuplicates step (e.g., GATK HaplotypCaller for variant calling). FastRemap correctly sets the supplementary alignment flag when remapping BAM/SAM files, enabling downstream analysis tools to correctly process the results. Second, FastRemap outputs reads that are either (i) unmapped in the target reference or (ii) have failed to remap into a separate BED file, which can be referenced for further analysis. This allows the user to immediately identify and process these reads from the BED file while minimizing the disk space requirements for storing these reads.

3 Results

Before discussing our evaluation results of peak memory consumption and performance, we describe our methodology for evaluation.

3.1 Evaluation methodology

Remapping tools. We compare FastRemap to the state-of-the-art remapping tool CrossMap and also full mapping with BWA-MEM.

Evaluated reference genomes. We evaluate FastRemap with several versions of reference genomes of varying size across three species (i.e., human, C. elegans, yeast).

Evaluated read datasets. We use publicly available DNA-seq read sets for each reference under examination, which are summarized in Table 1.

Evaluation system. We run FastRemap on a state-of-the-art server with 64 cores (2 threads per core, AMD EPYC 7742 @ 2.5 GHz), and 1 TB of main memory. We collect the end-to-end wallclock runtime (i.e., total execution time from the program’s start to finish from the perspective of the user) and memory usage using the time command in Linux with --et flags. We report the wallclock runtime (in seconds) and peak memory usage (in megabytes) of our evaluations based on these configurations.

3.2 Evaluation results

Figure 1 shows the evaluation results of FastRemap and CrossMap, as compared to full mapping using BWA-MEM, for wallclock runtime (Fig. 1, top graph) and peak memory usage (Fig. 1, bottom graph) across various pairs of reference versions for human (hg), C. elegans (ce) and yeast (sacCer). The reference pairs are denoted as source → target on the shared x-axis.

We make three key observations. First, FastRemap provides significant wallclock speedup (i.e. between 5.88× and 7.82×) compared to CrossMap. This speedup mainly comes from FastRemap’s utilization of four threads in the Seqan code segments, whereas CrossMap’s code is purely single-threaded. Second, FastRemap requires a smaller peak memory footprint (between 61.7% and 88.0% of that required by CrossMap) compared to CrossMap’s, which is likely due to the smaller data structures that are used by C++ compared to Python. FastRemap’s lower memory usage provides higher scalability than CrossMap when running many instances in parallel. Third, compared to fully mapping the read set using BWA-MEM, FastRemap is significantly faster (150.3× on average) and much more memory efficient (requiring only 0.4% of the memory footprint of BWA-MEM on average).

We conclude that FastRemap provides significant performance, memory and functional benefits over the state-of-the-art CrossMap tool when remapping BAM/SAM files. We hope that FastRemap enables academia and industry to develop powerful analysis pipelines for quickly studying the effects of mapping a read dataset to a different reference genome [e.g. AirLift (Kim et al., 2019)].

Funding

Gifts and funding from Intel and VMware to Onur Mutlu’s SAFARI Research Group.

Conflict of Interest: none declared.

Data availability

The data underlying this article are available in NCBI’s Genome Database, at https://www.ncbi.nlm.nih.gov, and can be accessed with accession numbers ERR194147, ERR262997, SRR3536210, and ERR1938683. Reference genomes (i.e., hg16, hg17, hg18, hg19, hg38, ce2, ce4, ce6, ce10, ce11, sacCer1, sacCer2, and sacCer3) can be accessed at https://hgdownload.soe.ucsc.edu/downloads.html.

References


