bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

RESEARCH

AirLift: A Fast and Comprehensive Technique for Remapping Alignments between Reference Genomes

Jeremie S. Kim[1], Can Firtina[1], Meryem Banu Cavlak[2], Damla Senol Cali[3], Nastaran Hajinazar[1,4], Mohammed Alser[1], Can Alkan[2] and Onur Mutlu[1,2,3*]

[*]Correspondence:
omutlu@gmail.com
[1]ETH Zurich, Rämistrasse 101, Zürich, Switzerland
Full list of author information is available at the end of the article
[†]Equal contributor

Abstract

As genome sequencing tools and techniques improve, researchers are able to incrementally assemble more accurate reference genomes, which enable sensitivity in read mapping and downstream analysis such as variant calling. A more sensitive downstream analysis is critical for a better understanding of the genome donor (e.g., health characteristics). Therefore, read sets from sequenced samples should ideally be mapped to the latest available reference genome that represents the most relevant population. Unfortunately, the increasingly large amount of available genomic data makes it prohibitively expensive to fully re-map each read set to its respective reference genome every time the reference is updated. There are several tools that attempt to accelerate the process of updating a read data set from one reference to another (i.e., remapping) by 1) identifying regions that appear similarly between two references and 2) updating the mapping location of reads that map to any of the identified regions in the old reference to the corresponding similar region in the new reference. The main drawback of existing approaches is that if a read maps to a region in the old reference that does not appear with a reasonable degree of similarity in the new reference, the read cannot be remapped. We find that, as a result of this drawback, a significant portion of annotations (i.e., coding regions in a genome) are lost when using state-of-the-art remapping tools. To address this major limitation in existing tools, we propose AirLift, a fast and comprehensive technique for remapping alignments from one genome to another. Compared to the state-of-the-art method for remapping reads (i.e., full mapping), AirLift reduces 1) the number of reads (out of the entire read set) that need to be fully mapped to the new reference by up to 99.99% and 2) the overall execution time to remap read sets between two reference genome versions by 6.7×, 6.6×, and 2.8× for large (human), medium (C. elegans), and small (yeast) reference genomes, respectively. We validate our remapping results with GATK and find that AirLift provides similar accuracy in identifying ground truth SNP and INDEL variants as the baseline of fully mapping a read set.

Code Availability. AirLift source code and readme describing how to reproduce our results are available at https://github.com/CMU-SAFARI/AirLift.

Keywords: Genome Read Mapping; Genome Assembly; Remapping; Crossmap; LiftOver

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

1 Introduction

Reference genomes are inaccurate and do not perfectly represent the average healthy individual of a species for a variety of reasons [1,2]. First, reference genomes are constructed using imperfect sequencing technologies that result in error-prone reads [3]. Second, the sequenced reads of an individual (i.e., *read set*) are assembled into a reference genome using imperfect assembly tools [4,5]. As genome sequencing technology and assembly algorithms improve, and as more sequenced samples become available, researchers are able to incrementally assemble more accurate reference genomes. As an example, the Genome Reference Consortium (GRC) releases minor updates to the human reference genome every three months and major updates every few years [6, 7]. Very recently, significant advances have resulted in a novel full telomere to telomere reference [8]. These updates are *critical* to the accuracy of the reference genome as they enable the latest reference genome to provide the most accurate and complete representation of the reference's respective population. Therefore, a read set should be mapped to the latest and most relevant reference genome to obtain the most accurate downstream genome analysis results [9].

Currently, the best way to adapt an existing genomic study (i.e., read sets from many samples) to a new reference genome is to re-run the *entire* analysis pipeline using the new reference genome. For example, the original analysis of the read sets from the 1000 Genomes Project was completed using the human reference genome build 37 (GRCh37) [10]. After the next version of the reference (GRCh38) became available, each read set from the 1000 Genomes Project was mapped again to the new human reference genome (GRCh38) [11]. Unfortunately, this approach is *computationally very expensive* and does not scale to large genomic studies that include a large number of individuals for three key reasons. First, mapping even a *single* read set is computationally expensive [12,13] (e.g., 75 hours for aligning 300,000,000 short reads, which provides 30× coverage of the human genome) as it heavily relies on a computationally-costly alignment algorithm [14, 15]. Second, the number of available read sets doubles approximately every 8 months [16, 17], and the rate of growth will continue to increase as sequencing technologies continue to become more cost effective and sequence with higher throughput [18]. Third, researchers are beginning to use highly-specific reference genomes that better represent diverse populations and ethnic groups [2,19,20,21,22,23,24,25]. This may result in the need to map each read set to *multiple* reference genomes that represent various populations within the same species in order to correctly identify the genome donor's genetic variations (i.e., differences from the most relevant reference genome).

To reduce the large overhead of *fully mapping* a read set to a new reference genome, several existing tools [26,27,28,29,30,31,32,33] can be used to quickly *remap* the reads (i.e., update a read's alignment location from the original (old) reference to another (new) reference). In the remainder of this paper, we collectively refer to such methods as *remapping tools*. At a high level, state-of-the-art remapping tools rely on *chain files* (described in Supplementary Section S2), which identify and list *constant regions*, i.e., genome sequences that appear in both old and new references (e.g., regions A and B in Figure 1) and their positional offsets into each reference genome. A remapping tool uses a chain file to identify reads whose original mapping locations in the old reference is sufficiently contained within constant regions and

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

quickly updates the alignment location of each read according to how the location of the constant region containing it changes between the old and new references. For example, Read 2 in Figure 1 can be quickly remapped by shifting its location by 5 base pairs from the old reference to the new reference.[1]

Unfortunately, these remapping tools 1) are not *comprehensive* in remapping a read set, meaning that they *cannot* remap a significant proportion of reads due to the limitations of using a chain file (e.g., a chain file only contains information about genome sequences that appear exactly the same between two references and their positional offsets into each reference), 2) are not *accurate*, meaning that some remapped reads do *not* align to the sequence they are remapped to in the new reference genome within the acceptable error rate, and 3) result in output on which downstream analysis cannot be performed (i.e., do not provide an end-to-end BAM-to-BAM[2] remapping solution). We identify two key limitations that we illustrate in Figure 1. First, since each *deleted region* (i.e., a region that does not appear in the new reference) does not have a corresponding region in the new reference, chain files *cannot* provide information on how to remap reads that had originally mapped to a deleted region. This is because, by definition, a deleted region has no similar regions in the new reference. For example, Read 1 in Figure 1 maps to a deleted region in the old reference and therefore cannot be remapped to the new reference to any extent. Second, state-of-the-art remapping tools *only* consider the degree of similarity between a read and the constant regions (from the chain file) in the old reference, without considering the changes in the new reference when remapping the read to the new reference. Therefore, remapping can result in a poor degree of similarity between the read and the new reference. As an example, Read 3 in Figure 1 maps to the old reference with high similarity (i.e., 4 deletions between base pairs 1375 and 1379; $< 5\%$ error rate), so it is remapped to the new reference at a location corresponding to the read's original mapping in the old reference. This remapping does not account for differences that appear in the new reference (e.g., 20 insertions between base pairs 1380 and 1400) and result in a high error rate (i.e., $> 5\%$).

Due to these limitations, existing remapping tools are unable to comprehensively remap a read set from one reference to another. We observe that state-of-the-art remapping tools miss at least 7% of gene annotations when remapping reads from an older human reference genome (hg16) to its latest version (GRCh38), as shown in Supplementary Table S1 and Supplementary Figure S1. These limitations require researchers and practitioners to re-run the *full* genome analysis pipeline for each read set on an updated reference genome for a comprehensive study.

Our **goal** is to provide the first read remapping technique across (reference) genomes 1) that *substantially* reduces the time to remap a read set from an old (i.e., previously mapped to) reference genome to a new reference genome, 2) that is *comprehensive* in remapping a read set, i.e., attempts to remap *all* reads in a read set, 3) provides *accurate* remapping results, i.e., provides alignments with error rates below a specified acceptable error rate, and 4) provides an end-to-end

[1] These tools are described in more detail in Supplementary Section S1.
[2] A BAM file is the binary version of a SAM file. A SAM file is a tab-delimited text file that contains sequence alignment data [34].

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

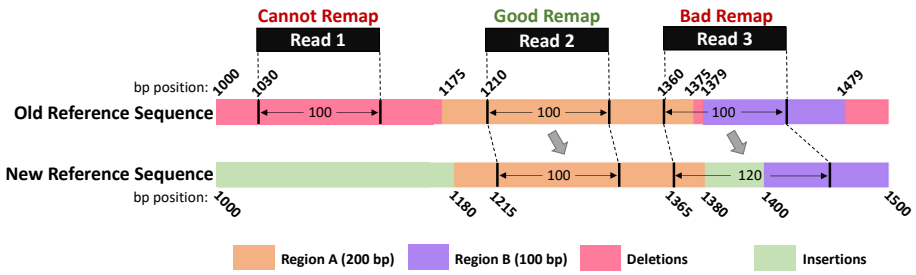

Figure 1: Limitations of Existing Remapping Tools. Existing remapping tools correctly remap reads that mapped *completely within* a region indicated by the chain file (e.g., Read 2). However, these tools 1) cannot remap reads that mapped within a region in the old reference that does not appear in the new reference (e.g., Read 1) and 2) may incorrectly remap reads that align to multiple constant regions in the old reference (e.g., Read 3).

BAM-to-BAM remapping solution on which downstream analysis can be immediately performed. To this end, we propose *AirLift*, the first methodology and tool that leverages the similarity between two reference genomes to satisfy our goal. Specifically, AirLift greatly reduces the time to perform end-to-end BAM-to-BAM remapping on a read set from one reference genome to another while maintaining high accuracy and comprehensiveness that is comparable to *fully mapping* the read set to the new reference.

We evaluate AirLift and demonstrate that AirLift satisfies the four design goals of an effective remapping tool by comparing it against state-of-the-art remapping tools and the previous best method of *fully mapping* a read set to a new reference with BWA-MEM [35] across various versions of the human, C. elegans, and yeast references (summarized in Table 1). We demonstrate that AirLift can identify SNPs and Indels with precision and recall similar to full mapping (via GATK HaplotypeCaller [36]) while providing 2.6× to 6.7× speedup over *fully mapping* a read set to the new reference genome.

Proposal	Year	Fast	Comprehensive	Accurate	BAM-to-BAM	Memory Usage
CrossMap [30]	2014	✓	✗	✗	✗	low
LiftOver [26]	2014	✓	✗	✗	✗	low
Full Mapping (BWA-MEM [35])	2013	✗	✓	✓	✓	high
AirLift	2021	✓	✓	✓	✓	high

Table 1: AirLift vs. existing state-of-the-art remapping tools.

2 AirLift

In order to accurately and comprehensively remap a read set, AirLift 1) categorizes and labels each region (i.e., a contiguous sequence within a genome) in the old reference genome depending on its degree of similarity to the most similar region in the new reference and 2) remaps each read from the old reference to the new reference according to the label of the region in the old reference that the read had been originally mapped to.

For each pair of references that AirLift remaps reads between, we must first construct an *AirLift Index*, i.e., a set of lookup tables (LUTs), in a one-time prepro-

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

cessing step. AirLift queries the AirLift Index with a read and its original mapping location in the old reference (from the BAM file) to efficiently identify the region and the label of the region that the read mapped to in the old reference. This information is then used to identify potential mapping locations of the read in the new reference (based on regions in the new reference that are similar to the region that the read mapped to in the old reference).

We next define these regions, show how to generate the *AirLift Index*, and then explain how to use the *AirLift Index* to quickly remap a read set with high genome coverage.

2.1 Reference Genome Regions

We identify four categories of regions that fully describe the relationship between two reference genomes, old and new (shown in Figure 2):

1. A *constant region* is a region of the genome which is exactly the same in both old and new reference genomes (colored in blue). The start and end positions of a constant region are not necessarily the same in the old and new reference genomes.
2. An *updated region* is a region in the old reference genome that maps to at least one region in the new reference genome within a reasonable error rate, i.e., differences from the old reference (colored in orange with some differences marked with black bars).
3. A *retired region* is a region in the old reference genome that does *not* map to any region in the new reference genome (colored in pink).
4. A *new region* is a region in the new reference genome that does *not* map to any region in the old reference genome (colored in green).

We next describe how we identify and use these regions to quickly and comprehensively remap a read set.

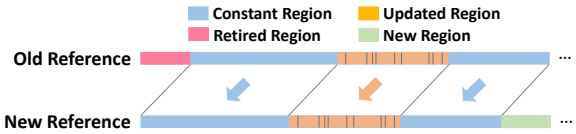

Figure 2: An example pair of reference genomes (old and new) with regions labeled (as constant, updated, retired, and new regions) and associated with each other according to their degrees of similarity. Regions that are associated with (i.e., similar to) each other are indicated with an arrow.

2.2 The AirLift Index

The *AirLift Index* is comprised of two lookup tables (LUTs), each of which has a one-time construction cost for any pair of reference genomes. The LUTs describe regions of similarity between a pair of reference genomes, which can then be used to quickly remap reads between the references.

The first LUT, i.e., *constant regions LUT*, associates each constant region in the old reference with its respective region in the new reference genome. AirLift queries this *constant regions LUT* with a location (of a previously-mapped read) in the old

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

reference to quickly find a list of corresponding locations in the new reference that have the same genome sequence. AirLift uses this list of locations to update the mapping of the read, as we explain in more detail in Section 2.4.

The second LUT, i.e., *updated regions LUT*, associates each updated region in the old reference with its respective region in the new reference genome. AirLift queries this *updated regions LUT* with a location (of a previously-mapped read) in the old reference to quickly find a list of corresponding locations in the new reference that have similar genome sequences. AirLift uses this list of locations to update the mappings of the read, as we explain in more detail in Section 2.4.

Once constructed, the *AirLift Index* is used to aid in the efficient mapping of any number of reads from one reference genome to another reference genome. We next explain how to label regions in the reference and construct the *AirLift Index*.

2.3 Categorizing Regions of Similarity and Constructing the AirLift Index

The *AirLift Index* is constructed via eight key steps, as we show in Figure 3.

(1) First, we want to identify all regions (i.e., genome sequences) that appear exactly the same in both the old and the new reference genomes. To do so, we use a chain file (described in Supplementary Section S2), which can be generated via BLAT [37] with exact matching (no errors allowed) global alignment. In Figure 3, we indicate the constant regions in blue.

(2) In order to label the remaining regions in the new reference, we first extract seeds (i.e., smaller subsequences) from regions in the old reference that do *not* map exactly to the new reference (non-blue regions). Note that these seeds **a)** are the same length (N) as the reads that we want to remap, and **b)** are overlapping seeds, i.e., completely overlap with each other such that a seed begins at each base pair within each (non-blue) region and starting $N-1$ base pairs before each (non-blue) region. This is to ensure that AirLift completely accounts for all possible mapping locations including sequences that may be partially included in a constant region.

(3) Next, we map the extracted seeds (from Step 2) to the new reference genome to identify regions of approximate similarity across the reference genomes. Note that this step can be done with *any* read mapper. We label as an *updated region* (colored

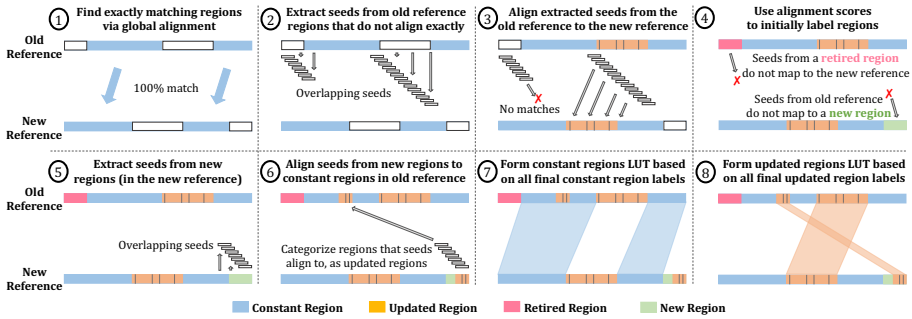

Figure 3: AirLift uses eight key steps to identify and label regions in the old and new reference genomes as *constant*, *updated*, *retired*, or *new* in order to efficiently map any number of reads from an old reference genome to a new reference genome.

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

in orange) 1) any continuous segment of base pairs that any seed has mapped to in the new reference or 2) any continuous segment of seed locations in the old reference whose seeds have mapped to the new reference. Since it is an approximate mapping, we indicate differences between the updated regions in Figure 3 with black stripes. These differences are accounted for by the resulting chain file.

While we describe in more detail how we use these regions in Section 2.4, we can quickly tell that if a read mapped to an updated region in the old reference genome, there is a high chance that the read will map to the respective updated region in the new reference genome. In order to comprehensively identify all possible locations in the new reference that a read can map to just by examining the read's mapping location in the old reference, we map seeds from the new reference using an error rate of $2e$, where e is the acceptable error rate for a successful alignment. Due to our usage of a conservative error rate ($2e$), we are still able to find every potential mapping with an alignment score within the acceptable error rate (Explained in Supplementary Section S4).

(4) We find regions in the old reference where seeds (extracted from Step 2) do *not* align to and label them as *retired regions*, since the region or anything similar does not exist in the new reference genome. Similarly, we find regions in the new reference whose seeds do not map to the old reference genome and label them as *new regions*, since the region or anything similar to the region does not exist in the old reference genome.

(5) Next, we check to see whether regions within the recently-identified new regions can be approximately aligned to constant regions in the old reference, since we had only previously attempted mapping them to the non-constant regions (in Step 3), and constant regions were only identified with *exact* matching. We do this by first extracting overlapping seeds from the new regions.

(6) We then map the extracted overlapping seeds (from Step 5) to the constant regions in the old reference genome. For any seeds that result in a successful alignment, we 1) additionally label the corresponding segment of the constant region as an updated region and 2) relabel the corresponding segment of the new region as an updated region. We can now consider each of these regions as updated regions, since this step has resulted in identifying an associated similar region in the other reference. This step is necessary to ensure that all regions in the old reference are checked for similarity to all regions in the new reference, enabling a comprehensive mapping for reads that map to *any* region in the old reference.

(7) We show the associated constant regions between the two references within the areas shaded in blue and use this information to create a *constant regions LUT*, which can be queried with a location in the old reference to obtain locations in the new reference that contain the exact same sequence. We encode the mapping with the chain file format (described in Supplementary Section S2).

(8) We show the associated updated regions between the two references within the areas shaded in orange and use this information to create the *updated regions LUT*, which can be queried to immediately return candidate locations in the new reference that a read should be aligned to. We encode the mapping and account for the minor differences using the chain file format.

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

2.4 Using AirLift to Remap a Read

AirLift follows the procedure illustrated in Figure 4 to comprehensively and accurately remap a read set. AirLift first identifies the label of the region that the each read had originally mapped to in the old reference using a series of steps (described in Section 2.4.1). Depending on the label, AirLift remaps each read using one of four independent cases (described in Section 2.4.2), depending on the label of the region that the read originally mapped to within the old reference: **(1)** a read that mapped to a *constant region*, **(2)** a read that mapped to an *updated region*, **(3)** a read that mapped to a *retired region*, and **(4)** a read that *never mapped* to any location in the old reference genome (i.e., an unmapped read).

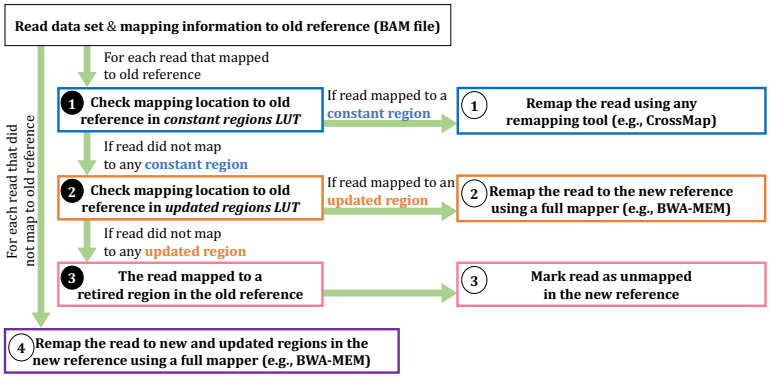

Figure 4: Using AirLift to remap a read set. AirLift remaps each read differently depending on the label of the region in the old reference that the read had originally mapped to: constant, updated, retired, or unmapped.

2.4.1 Determining how to Remap each Read

To determine which case AirLift should apply when remapping a read, AirLift performs the following steps on each read in the read set that originally mapped to any location in the old reference. First, AirLift checks the read's mapping location to the old reference in the *constant regions LUT* (❶ in Figure 4). If the mapping location returns an associated location in the new reference, the read had been originally mapped to a constant region in the old reference and AirLift remaps the read via Case (1) (described in Section 2.4.2).

If the *constant regions LUT* does not return a location in the new reference, AirLift next checks the read's mapping location to the old reference in the *updated regions LUT* (❷ in Figure 4). If the mapping location returns an associated location in the new reference, the read had been originally mapped to an updated region in the old reference and AirLift remaps the read via Case (2) (described in Section 2.4.2).

If the *updated regions LUT* does not return a location in the new reference, the read had been originally mapped to a retired region in the old reference (❸ in Figure 4). This is because an old reference is only comprised of constant, updated, and retired regions, and AirLift already determined that the read was not originally mapped to a constant or updated region. AirLift handles such reads via Case (3) (described in Section 2.4.2).

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

In order to be comprehensive in remapping a read set, AirLift also considers the reads that were unmapped in the old reference and attempts to remap them to the new reference using Case (4) (described in Section 2.4.2).

2.4.2 Remapping each Read

Case 1: For a read that had originally mapped to a *constant region*, we simply translate the mapping locations according to the offset in the specific constant region from the old reference to the new reference. Since this is the extent of existing state-of-the-art remapping tools capabilities, we can perform this step with any of these tools (e.g., *LiftOver*, *CrossMap*) for any read that is fully encapsulated within a chain file interval. In our analysis, we use *CrossMap*[3], since it outputs BAM files that can be used for downstream analysis (e.g., variant calling) for validating our results. The chain file represents only regions that are exact matches, so remapped reads will perfectly match to regions in the new reference genome as well.

Case 2: For a read that maps to an *updated region*, we first query the *updated regions LUT* to quickly obtain a list of locations in the new reference genome that are similar (within a $2e$ error rate) to the location that the read mapped to in the old reference genome. We can then use any aligner to align the read to all locations returned by the *updated regions LUT* and return the locations in the new reference genome that align with an error rate smaller than a user defined error rate (e).

Case 3: For a read that maps to a *retired region* (in the old reference genome), we already know that the read will not map anywhere in the new reference genome, since retired regions are not similar to any region in the new reference genome. Therefore, we can mark that read as an unmapped read in the new reference genome.

Case 4: For a read that *never mapped anywhere* in the old reference genome, we know that the read will not map to any constant region in the new reference genome. However, there is a chance that the read can align to updated or new regions in the new reference genome. Therefore, we must fully map the read to each new and updated region using any read mapper.

3 Evaluation

Before showing our evaluations of AirLift's execution time (in Section 3.2), memory usage (in Section 3.3), and accuracy and comprehensiveness (in Section 3.4), we describe our methodology for evaluation.

3.1 Evaluation Methodology

AirLift Tools. We evaluate AirLift using 1) *CrossMap* [27,30] to quickly move all reads that map to constant regions in the old reference and 2) *BWA-MEM* [35] to map reads when constructing the *AirLift Index* and when fully mapping all other reads that do not map to constant regions (i.e., reads that map to updated regions and never mapped to the old reference), according to the *AirLift Index*.

Evaluated Remappers. We evaluate two state-of-the-art remappers, *CrossMap* [27, 30] and *UCSC LiftOver* [26] to compare against AirLift. Note that these two remappers do *not* provide a *comprehensive* or *accurate* solution to remapping reads from

[3] We make some necessary modifications to the *CrossMap* code such that its output is compatible with GATK (See Supplementary Section S5).

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

one reference to another. Due to the limitations of prior remappers (described in Supplementary Section S3), we evaluate and compare against the only comprehensive and accurate baseline of *fully mapping* the read set (from scratch without using any prior mapping information) to the new reference genome with *BWA-MEM* [35].

Evaluated Reference Genomes. We evaluate AirLift with several versions of reference genomes of varying size across 3 species (i.e., human, C. elegans, yeast) as shown in Supplementary Table S2.

Evaluated Read Data Sets. We use DNA-seq read sets from four different samples of the set of species whose reference genomes we examine (as shown in Supplementary Table S3).

GATK Variant Calling Evaluation. We evaluate AirLift remapping results via variant calling with *GATK HaplotypeCaller* [36] by following the best practices [38], VCFtools [39] to filter variant calling files based on a minimum quality score of 30 (i.e., −−minQ 30), and use the *hap.py* tool (https://github.com/Illumina/hap.py) to benchmark the variant calling results.

Evaluation System. We run AirLift on a server with 64 cores (2 threads per core, AMD EPYC 7742 @ 2.25GHz), and 1TB of the memory. We assign 32 threads for C. elegans and yeast and 48 threads for human genomes when running all tools and collect their runtimes (usr and sys) and memory usage using the `time` command in Linux with `-vp` flags. We report the runtime (in seconds) and peak memory usage (in megabytes) of our evaluations based on these configurations.

AirLift Evaluation Plots. In each AirLift evaluation plot, we show on the x-axis, both the old reference genome (below) and the new reference genome (above) used in the evaluation. Note that in our evaluations of AirLift, we *only* consider the remapping stage (as other stages are preprocessing stages that are performed once for each pair of reference genomes). We show the execution times of the preprocessing stage in Supplementary Table S4.

3.2 AirLift Execution Time

We first demonstrate how AirLift reduces the time to map a set of reads to an updated reference genome by reducing the number of reads that we must map. Figure 5 plots the execution times (y-axis) for mapping a read set to a new reference genome using three different remapping tools, *CrossMap*, AirLift, and *LiftOver* compared to the baseline of *fully remapping* the entire read set from an old reference genome to the new reference genome. We provide the speedup of AirLift over fully mapping the read set to the new reference (i.e., $T_{\text{Full Mapping}}/T_{\text{AirLift}}$) above each bar.

The execution time of AirLift is calculated as the sum of the execution times for performing each of the cases (described in Section 2.4.2) as follows:

$$T_{\text{AirLift}} = T_{\text{constant_reads}} + T_{\text{updated_reads}} + T_{\text{retired_reads}} + T_{\text{unmapped}} \quad (1)$$

where $T_{\text{constant_reads}}$ is the time to translate all reads that originally map to a constant region in the old reference, $T_{\text{updated_reads}}$ is the time to map all reads that originally mapped to an updated region in the old reference, $T_{\text{retired_reads}}$ is the time to map all reads that originally mapped to a retired region in the old reference, and T_{unmapped} is the time to map all reads that never mapped anywhere in the

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

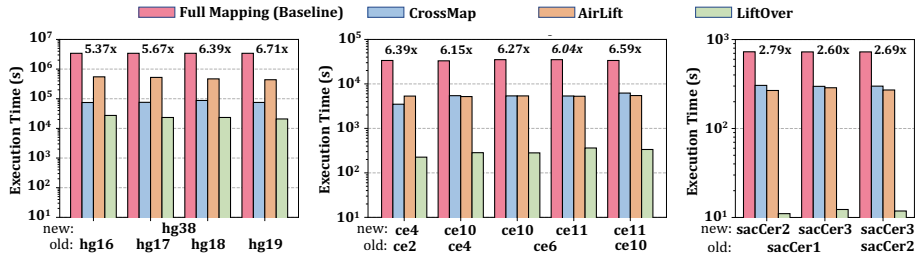

Figure 5: **AirLift execution time results.** We show the execution time (log-scale y-axis) of running three remapping tools, *CrossMap* (blue), AirLift (orange), and *LiftOver* (green) on a read set to a new reference genome against the baseline (red) of *fully mapping* a read set to the new reference genome. We plot the execution times of each tool for various pairs of reference genomes (x-axis; where the old reference is at the bottom and the new reference is above the old reference) in three separate plots for different sizes of reference genomes, i.e., large (human), medium (C. elegans), small (yeast). We indicate the speedup of AirLift against the *full mapping* baseline above each grouping of bars, since AirLift and the baseline are the only comprehensive and accurate remapping techniques available.

old reference. The exact execution time breakdowns for each of these four cases are shown in Supplementary Table S6). We also provide the number of reads that AirLift must remap in each case for each pair of references in Supplementary Table S7, and the average time per read per case for each pair of references in Supplementary Table S8.

We make three observations based on Figure 5 and the supplementary tables. First, AirLift consistently provides significant speedup over the baseline (of *fully mapping* a read set) across all tested pairs of references, ranging from 2.60× (`sacCer1`→`sacCer2`) up to 6.7× (`hg19`→`hg38`). Second, AirLift execution time is largely comprised of the time to remap reads that originally mapped to the constant region in the old reference. This is because the number of reads remapped by AirLift are mostly (i.e., between 86.57% for `hg16`→`hg38` and 98.47% for `ce10`→`ce11`) comprised of reads that originally mapped to a constant region. Third, AirLift execution time is significantly lower than the full mapping baseline since the average time to remap a read from the constant regions is significantly lower than the average time to fully map a read. This is because AirLift can very efficiently remap reads from constant regions. Fourth, remapping a read set with AirLift between a pair of references with a smaller constant regions size results in a higher execution time. Therefore, AirLift performs faster when remapping reads between pairs of references that are more similar to each other.

We conclude that AirLift significantly improves the execution time for *comprehensively* and *accurately* remapping a read set from an old reference to a new reference compared to the baseline of *fully mapping* the read set to the new reference.

3.3 AirLift Memory Usage

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

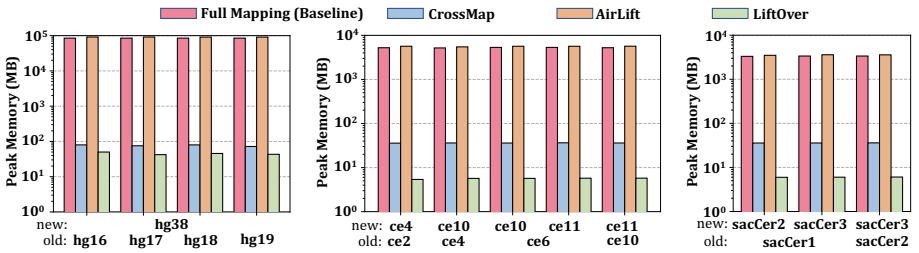

Figure 6: **AirLift memory usage results.** Peak memory usage results for each of the remapping tools during remapping.

Figure 6 plots the peak memory usage in MB (y-axis) across the remapping tools (i.e., *CrossMap*, AirLift, and *LiftOver*) and baseline full mapping method (i.e., *BWA-MEM*) for our set of evaluated reference pairs (x-axis). We find that across all tested reference pairs, AirLift has similar peak memory requirements as our *full mapping* baseline, *BWA-MEM*. This is because AirLift relies on *BWA-MEM* to remap a portion (i.e., up to 16.61%) of the read set, which is large enough to require the same amount of memory as mapping the full read set.

3.4 GATK Variant Calling Results

To demonstrate that AirLift provides similar mapping results as a *full mapper* (baseline) and it is much more comprehensive and accurate than *CrossMap* and *LiftOver*[4], we perform downstream analysis (i.e., variant calling). We use the GATK HaplotypeCaller tool to call variants from both the 1) *full mapping* BAM file and 2) Airlift-generated BAM file. We use the hap.py tool to benchmark 1) the AirLift variant calls against full mapping, 2) the AirLift variant calls against the gold standard (i.e., ground truth), and 3) full mapping variant calls against the ground truth, if the ground truth is available. We use the variant calling ground truth from the Platinum Genomes [40] and Genome in a Bottle [41] for the human NA12878 sample. We only benchmark Airlift against *full mapping* for the C. elegans and yeast data sets, since we do not have the ground truth for these species. We report the precision and recall results for the SNPs and insertion-deletions (indels) as calculated by hap.py (https://github.com/Illumina/hap.py).

Table 2 shows the variant calling results for human, C. elegans, and yeast genomes, respectively. Each row contains quality measurements of identifying single nucleotide polymorphisms (SNPs) and insertion-deletions (indels) for a pair of reference genomes in terms of precision and recall (written as 'precision score(%)/recall score(%)'). For the human results, we show the precision and recall scores of *full mapping* when identifying the set of SNPs and indels compared against the set of SNPs and indels that the ground truth reports, to demonstrate how AirLift compares against *full mapping* when identifying ground truth SNPs and indels. The

[4]The GATK HaplotypeCaller tool cannot analyze the outputs of *CrossMap* or *LiftOver* since their outputs are not compatible with downstream analysis tools (as described in Supplementary Section S5). Therefore, we do not analyze the outputs of *CrossMap* or *LiftOver* in this section.

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

columns are separated to show separate precision and recall scores for identifying the set of SNPs and indels when compared against the set of SNPs and indels that *full mapping* identifies (vs. Full Mapping) and the ground truth reports (vs. Ground Truth; only available for human results).

Table 2: GATK Variant Calling Results for Human, C. elegans, and Yeast Genomes

Remap Technique	**Read Sets** from	**Read Sets** to	**vs. Full Mapping** SNP (%)	**vs. Full Mapping** Indel (%)	**vs. Ground Truth** SNP (%)	**vs. Ground Truth** Indel (%)
Full Mapping	-	hg38	-	-	97.73/99.25	81.46/96.27
AirLift	hg16	hg38	92.69/92.31	85.01/87.14	95.05/97.73	76.22/94.12
	hg17		92.52/95.27	84.67/88.61	94.58/98.45	75.93/94.76
	hg18		93.10/95.33	85.10/88.64	95.00/98.49	76.25/94.81
	hg19		93.77/95.61	85.28/89.02	95.47/98.64	76.22/95.03
AirLift	ce2	ce4	90.82/97.29	96.97/97.66	-	-
	ce4	ce10	91.06/96.96	96.81/97.30	-	-
	ce6		91.11/97.00	96.81/97.33	-	-
	ce6	ce11	90.01/96.12	95.86/96.18	-	-
	ce10		90.03/96.48	95.90/96.44	-	-
AirLift	sacCer1	sacCer2	95.30/98.82	95.83/94.74	-	-
	sacCer1	sacCer3	86.35/94.27	90.38/88.65	-	-
	sacCer2		87.03/91.19	91.14/88.65	-	-

GATK results of the read sets from all evaluated species remapped by AirLift from an older reference version (e.g., hg16, hg17) to a more recent reference version (e.g., hg38) and for fully mapping (via BWA-MEM) the read set to the latest human reference version (since we only have ground truth GATK values for the human reference). For each read set remapped by AirLift, we show the precision(%)/recall(%) results of identifying SNPs and indels compared to 1) *full mapping* and 2) the ground truth. We also show the results of fully mapping the read set to hg38 compared to the ground truth. All results were obtained using GATK HaplotypeCaller [36] and hap.py.

We make two key observations. First, we observe that AirLift is able to identify SNPs reported by *full mapping* with high precision and recall scores (as shown under the first column, *vs. Full Mapping*). This is because AirLift 1) identifies all possible mapping locations for each read in the read set similarly to the *full mapping* approach, 2) comprehensively maps each read accordingly, and 3) reports accurate alignment results (i.e., alignments with error rates below a specified acceptable error rate) unlike existing remapping tools. Second, we observe that AirLift indentifies SNPs and indels reported by the ground truth with precision similar to *full mapping*. We observe this by comparing the results in the first row (i.e., *Full Mapping*) against the AirLift results directly underneath them (e.g., 95.47%/98.64% precision/recall values for identifying SNPs when full mapping to hg38 compared to 97.73%/99.25% when using AirLift between `hg19`→`hg38`; only available for human results). We note the small variation across precision and recall values in the table and attribute them to two main factors. First, since AirLift performs the most efficient method for remapping a read in the case that multiple methods are available (i.e., a read that maps to a constant region and updated region will be treated as a read in a constant region), AirLift may report mapping results that do not necessarily result in the best alignment score. Second, these discrepancies may occur as a result of genomic repeats and reproducibility issues in BWA-MEM [42]. We argue that these alignment differences do not cause a significant loss in variant calling quality, as AirLift precision and recall results for SNPs and indels are very similar to full mapping (when both are benchmarked against the ground truth).

We have shown in our evaluations against existing state-of-the-art remapping tools, that AirLift can comprehensively and accurately remap a read set from one reference genome to another at high speeds (i.e., up to 6.7× faster than our *full*

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

mapping baseline). Since AirLift accomplishes our four goals of remapping a read set quickly, comprehensively, accurately, and end-to-end, providing a BAM-to-BAM result that can be immediately used in downstream analysis, we conclude that AirLift is a viable tool to be used as a quick alternative to fully mapping a read set when it had previously been mapped to a similar reference genome.

4 Conclusion

We introduce AirLift, a methodology and tool for quickly, comprehensively, and accurately remapping a read data set that had previously been mapped to an older reference genome to a newer reference genome. AirLift is the first tool that provides BAM-to-BAM remapping results of a read data set on which downstream analysis can be immediately performed. The key idea of AirLift is to construct and and use an *AirLift Index*, which exploits the similarity between two references to quickly identify candidate locations that the read should be remapped to based on its original mapping in the old reference. We compare AirLift against several existing remapping tools, CrossMap and LiftOver, which we demonstrate have several major limitations. These tools either do *not* provide accurate and comprehensive remapping results or do not result in remapping results on which downstream analysis can be immediately performed (summarized in Table 1). We compare AirLift against the only comprehensive and accurate method of *fully mapping* a read data set to the new reference using BWA-MEM, and find that AirLift significantly reduces the execution time by 6.7×, 6.6×, and 2.8× for large (human), medium (C. elegans), and small (yeast) reference genomes, respectively. We validate our results against the ground truth and show that AirLift identifies similar rates of SNPs and Indels as the full mapping baseline. We conclude that AirLift is the first comprehensive and accurate remapping tool that substantially reduces the execution time of remapping a read data set, while providing end-to-end BAM-to-BAM results on which downstream analysis can be performed. We look forward to future works that take advantage of as well as improve AirLift for various genomic analysis studies.

Availability of data and materials

The Human NA12878 illumina read data set is publicly available (Accession number ERR194147 and ERR262997). The C. elegans N2 illumina read data set is publicly available (Accession number SRR3536210). The Yeast S288C illumina read data set is publicly available (Accession number ERR 1938683).

Competing Interests

The authors declare that they have no competing interests.

Author details

[1]ETH Zurich, Rämistrasse 101, Zürich, Switzerland. [2]Bilkent University, Bilkent, Ankara, Turkey. [3]Carnegie Mellon University, 5000 Forbes Avenue, 15213, Pittsburgh, Pennsylvania, USA. [4]Simon Fraser University, 8888 University Dr, V5A 1S6, Burnaby, BC, Canada.

References

1. Mallick, S., Li, H., Lipson, M., Mathieson, I., Gymrek, M., Racimo, F., Zhao, M., Chennagiri, N., Nordenfelt, S., Tandon, A., *et al.*: The Simons Genome Diversity Project: 300 Genomes from 142 Diverse Populations. Nature **538**(7624), 201 (2016)
2. Sherman, R.M., Forman, J., Antonescu, V., Puiu, D., Daya, M., Rafaels, N., Boorgula, M.P., Chavan, S., Vergara, C., Ortega, V.E., *et al.*: Assembly of a Pan-genome from Deep Sequencing of 910 Humans of African Descent. Nature Genetics **51**(1), 30 (2019)
3. Ma, X., Shao, Y., Tian, L., Flasch, D.A., Mulder, H.L., Edmonson, M.N., Liu, Y., Chen, X., Newman, S., Nakitandwe, J., *et al.*: Analysis of Error Profiles in Deep Next-Generation Sequencing Data. Genome Biology **20**(1), 50 (2019)
4. Alkan, C., Sajjadian, S., Eichler, E.E.: Limitations of Next-Generation Genome Sequence Assembly. Nature Methods **8**(1), 61 (2011)

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

5. Steinberg, K.M., Schneider, V.A., Alkan, C., Montague, M.J., Warren, W.C., Church, D.M., Wilson, R.K.: Building and Improving Reference Genome Assemblies. Proceedings of the IEEE **105**(3), 422–435 (2017)
6. RefSeq Curation and Annotation of the Human Reference Genome. https://www.ncbi.nlm.nih.gov/refseq/about/human/
7. Genome Reference Consortium Introduction to Patches. https://www.ncbi.nlm.nih.gov/grc/help/patches/#frequency
8. Miga, K.H., Koren, S., Rhie, A., Vollger, M.R., Gershman, A., Bzikadze, A., Brooks, S., Howe, E., Porubsky, D., Logsdon, G.A., et al.: Telomere-to-Telomere Assembly of a Complete Human X Chromosome. Nature (2020)
9. Guo, Y., Dai, Y., Yu, H., Zhao, S., Samuels, D.C., Shyr, Y.: Improvements and Impacts of GRCh38 Human Reference on High Throughput Sequencing Data Analysis. Genomics **109**(2), 83–90 (2017)
10. 1000 Genomes Project Consortium: A Global Reference for Human Genetic Variation. Nature **526**(7571), 68 (2015)
11. Zheng-Bradley, X., Streeter, I., Fairley, S., Richardson, D., Clarke, L., Flicek, P., Consortium, .G.P.: Alignment of 1000 Genomes Project Reads to Reference Assembly GRCh38. GigaScience **6**(7), 1–8 (2017)
12. Ruffalo, M., LaFramboise, T., Koyuturk, M.: Comparative Analysis of Algorithms for Next-Generation Sequencing Read Alignment. Bioinformatics **27**(20), 2790–2796 (2011). doi:10.1093/bioinformatics/btr477
13. Canzar, S., Salzberg, S.L.: Short Read Mapping: An Algorithmic Tour. Proceedings of the IEEE **105**(3), 436–458 (2015)
14. Alser, M., Rotman, J., Taraszka, K., Shi, H., Baykal, P.I., Yang, H.T., Xue, V., Knyazev, S., Singer, B.D., Balliu, B., et al.: Technology Dictates Algorithms: Recent Developments in Read Alignment. arXiv preprint arXiv:2003.00110 (2020)
15. Alser, M., Bingöl, Z., Cali, D.S., Kim, J., Ghose, S., Alkan, C., Mutlu, O.: Accelerating Genome Analysis: A Primer on an Ongoing Journey. IEEE Micro (2020)
16. Broad Communications: Broad Institute Sequences Its 100,000th Whole Human Genome on National DNA Day. https://www.broadinstitute.org/news/broad-institute-sequences-its-100000th-whole-human-genome-national-dna-day
17. Ulrich, T.: Harnessing the Flood: Scaling up Data Science in the Big Genomics Era. https://www.broadinstitute.org/blog/harnessing-flood-scaling-data-science-big-genomics-era
18. Senol Cali, D., Kim, J.S., Ghose, S., Alkan, C., Mutlu, O.: Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions. Briefings in Bioinformatics **20**(4), 1542–1559 (2019)
19. Al-Mssallem, I.S., Hu, S., Zhang, X., Lin, Q., Liu, W., Tan, J., Yu, X., Liu, J., Pan, L., Zhang, T., *et al.*: Genome Sequence of the Date Palm Phoenix dactylifera L. Nature Communications **4**, 2274 (2013)
20. Xu, P., Zhang, X., Wang, X., Li, J., Liu, G., Kuang, Y., Xu, J., Zheng, X., Ren, L., Wang, G., *et al.*: Genome Sequence and Genetic Diversity of the Common Carp, Cyprinus carpio. Nature Genetics **46**(11), 1212 (2014)
21. Ahn, S.-M., Kim, T.-H., Lee, S., Kim, D., Ghang, H., Kim, D.-S., Kim, B.-C., Kim, S.-Y., Kim, W.-Y., Kim, C., *et al.*: The First Korean Genome Sequence and Analysis: Full Genome Sequencing for a Socio-ethnic Group. Genome Research **19**(9), 1622–1629 (2009)
22. Wang, J., Wang, W., Li, R., Li, Y., Tian, G., Goodman, L., Fan, W., Zhang, J., Li, J., Zhang, J., *et al.*: The Diploid Genome Sequence of an Asian Individual. Nature **456**(7218), 60 (2008)
23. Schuster, S.C., Miller, W., Ratan, A., Tomsho, L.P., Giardine, B., Kasson, L.R., Harris, R.S., Petersen, D.C., Zhao, F., Qi, J., *et al.*: Complete Khoisan and Bantu Genomes from Southern Africa. Nature **463**(7283), 943 (2010)
24. Huang, T., Shu, Y., Cai, Y.-D.: Genetic Differences among Ethnic Groups. BMC Genomics **16**(1), 1093 (2015)
25. Shukla, H.G., Bawa, P.S., Srinivasan, S.: hg19KIndel: Ethnicity Normalized Human Reference Genome. BMC Genomics **20**(1), 459 (2019)
26. UCSC: UCSC LiftOver: Lift Genome Annotations. https://genome.ucsc.edu/cgi-bin/hgLiftOver
27. Zhao, Hao and Sun, Zhifu and Wang, Jing and Huang, Haojie and Kocher, Jean-Pierre and Wang, Liguo: CrossMap: Convert Genome Coordinates Between Assemblies. http://crossmap.sourceforge.net/#use-pip-to-install-crossmap
28. Gao, B.: Segment Liftover. https://pypi.org/project/segment-liftover/
29. Gao, B., Huang, Q., Baudis, M.: Segment_Liftover: A Python Tool to Convert Segments Between Genome Assemblies. F1000Research **7** (2018)
30. Zhao, H., Sun, Z., Wang, J., Huang, H., Kocher, J.-P., Wang, L.: CrossMap: A Versatile Tool for Coordinate Conversion Between Genome Assemblies. Bioinformatics **30**(7), 1006–1007 (2013)
31. NCBI: NCBI Genome Remapping Service. https://www.ncbi.nlm.nih.gov/genome/tools/remap
32. The Galaxy Team: Galaxy. https://www.usegalaxy.org
33. Tretyakov, K.: PyLiftover. https://pypi.org/project/pyliftover/
34. SAM/BAM and related specifications. http://samtools.github.io/hts-specs/
35. Li, H.: Aligning Sequence Reads, Clone Sequences and Assembly Contigs with BWA-MEM. arXiv:1303.3997 (2013)
36. McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., DePristo, M.A.: The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. Genome Research **20**(9), 1297–1303 (2010). doi:10.1101/gr.107524.110
37. UCSC: Blat Suite Program Specifications and User Guide. https://genome.ucsc.edu/goldenPath/help/blatSpec.html
38. Auwera, G.A., Carneiro, M.O., Hartl, C., Poplin, R., del Angel, G., Levy-Moonshine, A., Jordan, T., Shakir, K., Roazen, D., Thibault, J., Banks, E., Garimella, K.V., Altshuler, D., Gabriel, S., DePristo, M.A.: From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline. Current Protocols in Bioinformatics **43**(1) (2013). doi:10.1002/0471250953.bi1110s43
39. Danecek, P., Auton, A., Abecasis, G., Albers, C.A., Banks, E., DePristo, M.A., Handsaker, R.E., Lunter, G., Marth, G.T., Sherry, S.T., *et al.*: The variant call format and vcftools. Bioinformatics **27**(15), 2156–2158

bioRxiv preprint doi: https://doi.org/10.1101/2021.02.16.431517; this version posted February 17, 2021. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

(2011)

40. Eberle, M.A., Fritzilas, E., Krusche, P., Källberg, M., Moore, B.L., Bekritsky, M.A., Iqbal, Z., Chuang, H.-Y., Humphray, S.J., Halpern, A.L., Kruglyak, S., Margulies, E.H., McVean, G., Bentley, D.R.: A reference data set of 5.4 million phased human variants validated by genetic inheritance from sequencing a three-generation 17-member pedigree. Genome Research **27**(1), 157–164 (2017). doi:10.1101/gr.210500.116
41. Zook, J.M., Chapman, B., Wang, J., Mittelman, D., Hofmann, O., Hide, W., Salit, M.: Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. Nature Biotechnology (2014). doi:10.1038/nbt.2835
42. Firtina, C., Alkan, C.: On genomic repeats and reproducibility. Bioinformatics **32**(15), 2243–2247 (2016). doi:10.1093/bioinformatics/btw139