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

Jeremie S. Kim¹,², Can Firtina¹, Damla Senol Cali², Mohammed Alser¹, Nastaran Hajinazar¹,⁴, Can Alkan³,‡, and Onur Mutlu¹,²,³,‡

¹ Department of Computer Science, ETH Zurich, Zurich 8006, Switzerland
² Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA
³ Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey
⁴ Department of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada
‡ Corresponding authors: calkan@cs.bilkent.edu.tr and onur.mutlu@inf.ethz.ch

Abstract. As genome sequencing tools and techniques improve, researchers are able to incrementally assemble more accurate reference genomes. A more accurate reference genome enables increased accuracy in read mappings, which provides more accurate variant information and thus health data on the donor. Therefore, read data sets from sequenced samples should ideally be mapped to the latest available reference genome. Unfortunately, the increasingly large amounts of available genomic data makes it prohibitively expensive to fully map each read data set to its respective reference genome every time the reference is updated. Several tools that attempt to reduce the procedure of updating a read data set from one reference to another (i.e., remapping) have been published. These tools identify regions of similarity across the two references and update the mapping locations of a read based on the locations of similar regions in the new reference genome. The main drawback of existing approaches is that if a read maps to a region in the old reference without similar regions in the new reference, it cannot be remapped. We find that, as a result of this drawback, a significant portion of annotations 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 moving alignments from one genome to another. AirLift can reduce 1) the number of reads that need to be mapped from the entire read set by up to 99.9% and 2) the overall execution time to remap the reads between the two most recent reference versions by 6.94×, 44.0×, and 16.4× for large (human), medium (C. elegans), and small (yeast) references, respectively.

Code Availability. The AirLift source code is available at https://github.com/CMU-SAFARI/AirLift.

Keywords: Genome Mapping, Genome Assembly, Remapping, Crossmapping, LiftOver

1 Introduction

Reference genomes are inaccurate and do not perfectly represent the average healthy individual of the population for a variety of reasons. First, reference genomes are derived primarily from individuals that do not necessarily represent the population and are missing a substantial amount of sequences [17, 23]. Second, they are constructed using imperfect sequencing technologies that result in error-prone reads [16]. Third, the resulting reads (i.e., read set) are assembled into a reference genome using imperfect assembly tools [6,25]. As genome sequencing tools 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) reviews minor updates to the human reference genome for release every three months and releases major updates every few years. These updates are critical to the accuracy of the reference genome with the latest reference genome having the most complete and most accurate annotations and sequences. Therefore, the original locations that each read was likely sequenced from should be found (i.e., read mapping) using the latest reference genome of its species to maintain an accurate downstream genome analysis [12] to obtain the most accurate health data on the sample.

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 1000 Genomes Project was completed using human reference genome build 37
After the next version of the reference (GRCh38) became available, each read set was mapped again to the new reference [35]. 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 expensive [8,19] (e.g., 15 minutes for aligning 3,000,000 short reads or 0.1× coverage of the human genome) and heavily relies on an expensive alignment algorithm. Second, the number of sequenced samples (i.e., read sets) doubles approximately every 8 months [7,30], and the rate of growth will only increase as long and ultra long sequencing technologies enable building assemblies with better contiguity (e.g., Nanopore [22]) and become more cost effective. Third, researchers are finding that reference genomes should be more comprehensive in representing diverse populations and ethnic groups [4,5,13,21,23,24,31,32]. This may lead to having multiple reference genomes (or alternate subsequences) representing the same species to which each read set must be mapped in order to correctly identify the genome variants.

To reduce the overhead in fully mapping a read set to a new reference genome, several prior tools [9,10,18,26,27,29,33,34] can be used to quickly update or remap the locations of the existing mappings of the reads (i.e., coordinates) in the original (old) reference genome to a different (new) reference genome at the cost of coverage (i.e., the percentage of the new reference genome that reads map to). In the remainder of the paper we collectively refer to such methods as remapping tools. Existing state-of-the-art remapping tools cannot provide high coverage when converting the mapping coordinates from one reference genome to another as they do not account for reads that map to regions in the old reference that are repetitive or have significant changes in the new reference genome. We observe that because of this limitation, state-of-the-art remapping tools can miss up to 7% of gene annotations when remapping reads from human reference genome build 16 to the latest human reference genome build (GRCh38). This limitation requires researchers 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 a technique that substantially 1) reduces the execution time to remap a read set from an (old) reference genome to a (new) reference genome, and 2) provides high coverage by also accounting for reads that map to regions that are repetitive or significantly different in the new reference genome. To this end, we propose AirLift, a methodology that leverages the similarity between two reference genomes to reduce the execution time to map a read set from one reference genome to another while maintaining a high coverage similar to fully mapping a read set to the new reference.

The key idea is to exploit the similarity between a pair of reference genomes in order to identify the cheapest method for moving reads. AirLift selects the cheapest method depending on the region that the read aligns to in the old reference and how that region relates to the new reference. This is done by generating lookup tables for each pair of reference genomes (old and new) that map locations of similar regions (with high error acceptance rates) between the references. The creation of these lookup tables is a one time effort and once created, they can be reused for any amount of reads. AirLift then uses these lookup tables to categorize all reads and remaps them accordingly.

We evaluate AirLift by comparing against BWA-MEM [14] fully mapping a read set to the new reference across various versions of the human (i.e., GRCh37 and GRCh38), Caenorhabditis elegans (i.e., ce1, ce2, ce4, ce6, ce10, and ce10), and yeast (i.e., sacCer1, sacCer2, and sacCer3) reference genomes. Based on our evaluation we provide two major results. First, we show that AirLift reduces the overall number of reads that needs to be remapped from the original read set by up to 99.9%. Second, AirLift reduces the overall runtime required to remap the reads from the old reference genome to the new reference genome by 6.94×, 44.0×, and 16.4× for large (human), medium (C. elegans), and small (yeast) reference genomes, respectively, when we use the reduced read set suggested by AirLift. We conclude that AirLift significantly reduces the overhead of updating the already mapped reads to a new reference genome while still accounting for the significant changes in the new reference genome.

2 Currently Available Remapping Tools

To the best of our knowledge, this is the first work that provides a comprehensive remapping of reads from one reference genome to another. While there are many works that are considered remapping tools, none of them provide a comprehensive mapping to a new reference genome. We explain the subtle differences between each of the remapping tools below.

**UCSC LiftOver.** One of the most commonly used remapping tools is UCSC LiftOver [29]. UCSC LiftOver uses a chain file [1] between two different assemblies of a genome to convert the coordinates from one
assembly to the assembly of the other genome. UCSC LiftOver suffers from three major shortcomings. First, UCSC LiftOver functionality is limited to the genomes whose assemblies are provided by the UCSC Genome Browser [28], hence, making it impossible to remap genomes whose assemblies are not yet included in the tool. Second, the tool only converts the coordinates of regions within the old reference genome that are highly similar to regions within the updated reference genome and ignores regions with significant variance (as we show in 3), which prevents a comprehensive remapping of the coordinates. Third, UCSC LiftOver only supports BED-format (i.e., browser extensible data) input files which limits its usage even further.

**CrossMap.** One alternative to UCSC LiftOver is CrossMap [33, 34]. CrossMap follows a similar approach with UCSC LiftOver and uses chain files to convert mappings from an older reference genome to a newer reference genome. Compared to UCSC LiftOver, CrossMap supports a larger set of input file formats, such as BAM, SAM, or CRAM, BED, Wiggle, BigWig, GFF (i.e., general feature format) or GTF (i.e., gene transfer format), and VCF (i.e., variant call format) [33, 34]. Unfortunately, CrossMap suffers from similar limitations as UCSC LiftOver.

**NCBI Genome Remapping Service.** Another alternative is NCBI Genome Remapping Service [18], which also remaps the annotations from one genome assembly to another. NCBI Remap has support for a larger set of input/output file formats, such as BED, GFF, GTF, and VCF. NCBI Remap can also perform cross species remapping for a limited number of organisms. However, as with UCSC LiftOver, NCBI Remap is limited by the provided assemblies.

**Segment liftover.** Segment liftover [9,10] is another tool that is designed to map coordinates of one genome assembly to another genome’s assembly while maintaining the integrity of the genome segments that are not continuous anymore in the target assembly.

**Galaxy.** Galaxy [11, 26] is a web-based platform, which has LiftOver as part of its toolset. This tool is based on UCSC LiftOver [29] and the chain files provided by UCSC Genome Browser [28]. Thus, Galaxy also suffers from similar limitations as UCSC LiftOver.

**PyLiftover.** PyLiftover [27] is a Python implementation of a limited version of UCSC LiftOver. PyLiftover does not convert ranges (i.e., only converts point coordinates) between different assemblies, and it does not support BED-format input files.

**Bazam.** Bazam [20] is another tool which remaps short paired reads by optimizing memory usage while providing high parallelism. However, Bazam only targets the steps where reads are read from a BAM or CRAM file (i.e., read extraction) and sent to an aligner (e.g., BWA [15]). Eventually, all the reads are remapped to the new reference genome, which is inefficient.

### 3 Motivation and Goal

Repeating a genomic study using another version of the reference genome is computationally very expensive. A faster and more convenient way to achieve this is to “remap” the mapping locations from the older reference genome to its updated version [9,10,18,26,27,29,33,34]. We evaluate the effectiveness of one of the state-of-the-art tools, UCSC LiftOver [29], in updating the mapping information from one version of the human reference genome to another version. We present the evaluation result in Figure 1, where we show the amount of information lost when remapping from one human reference genome version (x-axis) to the latest human reference genome version (hg38). The y-axis shows the percentage of annotations (labeled and marked with unique colors) missed when remapping using UCSC LiftOver. We make two key observations based on Figure 1. First, we observe that a significant portion (> 7%) of the genes and transcripts are lost when simply using an available crossmap tool between hg16 and hg38. Second, the percentage of the missed annotations decreases as the difference in versions becomes smaller, but even when lifting annotations between hg19 (released in 2009) and hg38 (released in 2013), 4.47% of genes are “lost” in hg38.

Table 1 contains the exact values of each lost annotation (in percentages) when using UCSC LiftOver from hg16, hg17, hg18, and hg19 (rows) to hg19 and hg38 (columns). As the output of lifting annotations from one reference to another is used in downstream genome analysis, we believe the speed and accuracy of lifting annotations, and coverage of the new reference genome are all crucial. However, prior works mainly focus on the speed at the cost of both accuracy and coverage. These crossmap tools are often very inaccurate and can only lift mappings or annotations for regions with minor changes [35]. Therefore, if researchers want a comprehensive study using a new reference genome, they must map the entire read data set to the new reference genome rather than rely on the results of such crossmap
Fig. 1. Percentage of different annotations missed when remapping reads from an old reference (x-axis) to the latest reference (hg38), using UCSC LiftOver [29].

Table 1. Lost information when crossmapping across reference genome assemblies using UCSC LiftOver.

| Old Reference | hg19 | hg38 |
|---------------|------|------|
| gene          | 3.07 | 7.03 |
| exon          | 0.92 | 2.13 |
| stop codon    | 0.79 | 2.16 |
| CDS           | 0.77 | 2.07 |
| start codon   | 0.72 | 7.06 |
| transcript    | 2.92 | 2.41 |

Between each pair of reference genomes, we indicate the exact values of specific annotation types (e.g., gene, exon, stop codon, CDS, start codon, transcript) that are “lost” when using UCSC LiftOver [29] on a read data set from an old reference (rows) to a new reference (columns). Briefly, 3.07% of the gene model coordinates in hg16 assembly are not found in hg19, where the loss rate of genes reaches 4.47% between the most recent two assembly versions (hg19 and hg38).

Our goal is to produce a method for quickly remapping the reads of a sample from one reference genome to an updated version of the reference genome or another similar reference genome.

4 AirLift

In this section, we describe AirLift, our technique for quickly mapping a read set from one reference genome to another. The key idea behind AirLift is to generate fixed lookup tables (LUTs) for each pair of reference genomes (old and new) that map locations of similar regions between the references. For a read that maps to a location in the old reference genome, we can query the LUT to quickly identify potential locations for mapping in the new reference genome. Depending on where the read mapped to in the old reference, we update the mapping location using different methods. We next define these regions, show how to generate the fixed lookup tables, and then explain how to use these LUTs to quickly and comprehensively remap a read set.

4.1 Reference Genome Regions

We compare two reference genomes with large sequences (i.e., regions). We identify four types of regions (shown in Figure 2) that fully describe the relationship between two reference genomes:

1. A constant region is a region of the genome which is exactly the same in both old and new reference genomes (blue).
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 reasonable error rates (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 (red).

4. A new region is a region in the new reference genome that does not map to any region in the old reference genome (green).

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

### 4.2 Generating Lookup Tables for AirLift

We propose to generate lookup tables (LUTs) to aid in the efficient mapping of reads from one reference genome to another reference genome. Figure 3 shows the methodology for creating the LUTs. Starting with the old and new reference genomes, we must first either acquire an available chain file \([1]\) (e.g., \([2]\)) or \((1)\) generate our own. We create our chain file by running global alignment without errors between the two reference genomes. This chain file shows where exact sequences from the old reference genome can be found in the new reference genome. We refer to regions that match perfectly across the old and new reference genome to be constant regions (blue). Next, we \((2)\) extract seeds (i.e., smaller subsequences) from regions in the new reference that do not align exactly (non-blue regions). Note that these seeds \((a)\) are the same length \((N)\) as the reads that we want to remap, and \((b)\) are completely overlapping sequences and starting \(N−1\) base pairs before each region, a seed begins at each location and ends \(N\) base pairs later (providing \(NX\) coverage on the region). Next, we \((3)\) align the extracted seeds to the old reference genome to identify regions of approximate similarity across the reference genomes. Note that this alignment can be done with any read mapper. The regions (in the old reference genome) that the extracted seeds align to and the regions (in the new reference genome) that the aligned seeds were extracted from are considered updated regions (orange). Since it is an approximate mapping, we indicate differences between the updated regions with black bars. While we describe in more detail how we use these regions later, 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 guarantee a comprehensive mapping between updated regions, we map the extracted reads with an error rate of \(2e\), where \(e\) is the acceptable error rate for an alignment to be considered a match. Figure 4 shows the worst-case example.
where a read (of length 20) aligns to a subsequence in the updated region of the old reference genome with an \( e = 5\% \) error rate (mismatch on the 9\(^{th} \) base pair), and also aligns to a subsequence in the updated region of the new reference genome with an \( e = 5\% \) error rate (mismatch on the 16\(^{th} \) base pair). In the case where we only have the alignment information between the read and a subsequence from the updated region of the old reference genome, we can quickly identify potential mapping locations in the new reference genome (with an error rate of \( e \)) given mappings between the updated regions of the old and new references with an error rate of \( 2e \). We also note that there are regions from the old reference where extracted seeds do not map anywhere in the new reference and regions in the new reference where no extracted seeds map to. We must (4) check the alignments of all the extracted seeds to determine which bucket each region falls into.

We refer to a region whose extracted seeds do not map to the old reference genome at all as a new region, since the region or anything similar to the region does not exist in the old reference genome. We refer to a region in the old reference, which has no seeds mapping to it as a retired region, since the region or anything similar does not exist in the new reference genome. Next, we (5) check to see whether regions within our recently-identified retired regions can be mapped to constant regions, since we had only previously checked them against the non-constant regions. We extract overlapping seeds from the retired region and (6) map them to the constant regions in the old reference genome. For any seeds that result in a match, we add the respective constant region in the new reference to the updated region and relabel the retired region in the old reference as an updated region. Following these procedures, we generate two lookup tables that aid in remapping a read set from the old reference to the new reference. (7) shows the constant regions lookup table (LUT) which essentially follows the format of a standard chain file describing how large regions map directly between reference genomes. (8) shows the updated regions lookup table (LUT), which contains the mappings of each seed location from updated locations in the old reference genome to the mapping locations in the new reference genome. We can use both of these LUTs (i.e., constant regions LUT and updated regions LUT) to re-map all reads from the old reference to the new reference quickly and comprehensively. Since it is possible that each location in the old reference genome can map to multiple locations in the new reference genome, these lookup tables are organized as maps with the keys being the location of the read in the old reference genome, and the value is a list of locations that the location can map to in the new reference genome. We query these lookup tables with the location of a read that was mapped to the old reference genome to quickly obtain a list of potential locations in the new reference genome to map the read to.

### 4.3 AirLifting a Read

We identify four independent cases for AirLifting reads from the old reference genome to the new reference genome that we must handle to fully map a read set (highlighted in Figure 5): (1) a read that maps to a constant region in the old reference genome, (2) a read that maps to an updated region in the old reference genome, (3) a read maps to a retired region in the old reference genome, and (4) a read that never mapped anywhere in the old reference genome. For a read falling in case (1), 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 a read falling in case (2), we simply query the updated regions LUT, and align the read to the returned locations. We then return locations that align with an error rate smaller than \( e \). For a read falling in case (3), we know that it will not map anywhere to the
reference genome, so we can mark it as an unmapped read. For a read falling in case (4), since we have no prior knowledge about the read other than the fact that it never mapped anywhere in the old reference genome, we must fully map the read to each new and updated region in the new reference genome (using your preferred read mapper), since we know that the constant regions did not result in matches in previous attempts. We next discuss our methodology for evaluating AirLift.

5 Evaluation Methodology

Evaluated Read Mappers. We evaluate AirLift using BWA-MEM [14] and Bazam [20]. Note that Bazam is not a standalone read mapper, instead it facilitates fast extraction of reads from an input BAM file and utilizes BWA-MEM for the actual mapping in a streaming fashion.

Evaluation System. We run our entire toolchain on a server with 24 cores (2 threads per core, Intel Xeon Gold 5118 CPU @ 2.30GHz), and 192GB of the memory. We assign 32 threads to all the tools we use and collect their runtime and memory usage using time command in Linux with -vp options. We report runtime and peak memory usage of our evaluations based on these configurations.

Evaluated Reference Genomes. We study the effects of AirLift on versions of reference genomes of varying size across 3 species (i.e., human, C. elegans, yeast) as shown in Table 2. We study a mix of species to show the effects of AirLift on reference genomes of varying sizes.

Table 2. Details of the reference genomes that we use in our experiments.

| Species  | Version | Bases        | non-N Bases  | Release Date       |
|----------|---------|--------------|--------------|--------------------|
| Human    | hg19    | 3,137,144,693| 2,897,293,955| 2009-02-27         |
| Human    | hg38    | 3,209,286,105| 3,049,316,098| 2013-12-24         |
| C. elegans| ce1     | 100,264,180  | 100,264,085  | 2003-05-02         |
| C. elegans| ce2     | 100,291,769  | 100,291,761  | 2004-03-01         |
| C. elegans| ce4     | 100,281,244  | 100,281,244  | 2007-01-01         |
| C. elegans| ce6     | 100,281,426  | 100,281,426  | 2008-05-01         |
| C. elegans| ce10    | 100,286,070  | 100,286,070  | 2012-04-13         |
| C. elegans| ce11    | 100,286,401  | 100,286,401  | 2013-02-07         |
| Yeast    | SacCer1 | 12,156,302   | 12,156,302   | 2001-10-01         |
| Yeast    | SacCer2 | 12,162,995   | 12,162,995   | 2008-06-01         |
| Yeast    | SacCer3 | 12,157,105   | 12,157,105   | 2014-12-17         |

Evaluated Read Data Sets. We use DNA-seq data sets from four different samples (as shown in Table 3).

Table 3. Our read data sets that we use in our experiments can be accessed via NCBI using the accession number.

| Data Set          | Accession               | Details                                        |
|-------------------|-------------------------|------------------------------------------------|
| Human NA12878 - Illumina | ERR194147 | 795,505,905 paired-end reads (101bps each, 50× coverage) |
| Human NA12878 - Illumina | ERR262997 | 643,097,275 paired-end reads (101bps each, 40× coverage) |
| C. elegans N2 - Illumina      | SRR3536210 | 78,696,056 paired-end reads (101bps each, 150× coverage) |
| Yeast S288C - Illumina       | ERR1938683 | 3,318,467 paired-end reads (150bps each, 82× coverage) |
6 Evaluation Results

6.1 AirLift Map Study

We first show our findings on how two reference genome versions relate to each other. Table 4 shows the region sizes (i.e., constant, updated, retired, new) that each pair of reference genomes are comprised of (as a percentage of all the regions combined). The values in parenthesis show the percentage of reads out of the entire read set (mapped to the old reference genome) that fall in each region of the old reference genome (i.e., constant, updated, retired). We note that the closer the version numbers between the pair of references are to each other, 1) the larger the constant region is, and 2) the smaller the updated region is. This is intuitive as each reference genome version releases incremental changes to update missing and inaccurate sequences, so the similarity between subsequent releases would likely be higher than between releases further apart. We also observe, as expected, that the percentage of reads that map to a region in the reference is correlated with the region size (i.e., larger regions have a larger percentage of the read set mapped to that region). As the method for remapping a read depends on the type of region it is mapped to in the old reference, we can estimate the execution time of using AirLift on an entire read set with the proportions of reads that fall in different regions (in Table 4). Since the most expensive method for remapping in AirLift (i.e., alignment) is employed only for reads that mapped to updated and retired regions of the old reference, we can expect, based on the significantly small proportion of reads (e.g., between 0.36 and 13.68%) in the updated and retired regions of Table 4, a significant reduction in the mapping time.

We next plot the actual reduction (y-axis) we observe in the number of reads for the pairs of reference genomes (x-axis) that we examine in Figure 6. We make two observations. First, we observe that the reduction in the read set is significant, between 86% (for hg19hg38) and 99.967% (for ce4ce6). This is the main contributor to our performance improvement, as the number of alignments performed is significantly reduced. Second, we observe from the C. elegans samples that remapping a read set between subsequent genome version (e.g., ce1ce2, ce2ce4, ce4ce6) results in a significantly reduced read set. We conclude from this figure that we can exploit the high similarity between reference genome versions to significantly reduce the number of reads that we need to map for a comprehensive read set mapping.

| Species | Remapping a read set | Constant (%) | Updated (%) | Retired (%) | New (%) |
|---------|----------------------|--------------|-------------|-------------|---------|
|         | From | To     |    |
| Human   | hg19 | hg38   | 84.71 (86.31) | 2.83 (13.64) | 7.12 (0.045) | 3.44 |
| ce1     |      | ce10   | 98.85 (99.39) | 0.63 (0.61) | 0.01 (0.004) | 0.51 |
| ce2     |      | ce10   | 98.95 (99.45) | 0.58 (0.55) | 0.01 (0.004) | 0.46 |
| ce4     |      | ce10   | 99.23 (99.60) | 0.40 (0.40) | 0.01 (0.004) | 0.33 |
| ce6     |      | ce10   | 99.31 (99.64) | 0.36 (0.36) | 0.01 (0.004) | 0.32 |
| C. elegans | ce1  | ce10   | 95.78 (97.90) | 2.51 (2.09) | 0.01 (0.012) | 1.70 |
| cc2     |      | ce11   | 95.87 (97.94) | 2.47 (2.05) | 0.01 (0.011) | 1.65 |
| cc4     |      | ce11   | 96.12 (98.13) | 2.31 (1.85) | 0.01 (0.012) | 1.56 |
| cc6     |      | ce11   | 96.16 (98.13) | 2.27 (1.86) | 0.01 (0.012) | 1.54 |
| cc10    |      | ce11   | 96.74 (98.48) | 1.92 (1.51) | 0.01 (0.006) | 1.33 |
| SacCer1 | SacCer2 |         | 97.17 (98.64) | 1.70 (1.34) | 0.11 (0.018) | 1.02 |
| SacCer1 | SacCer2 |         | 88.88 (93.87) | 7.86 (6.11) | 0.12 (0.024) | 3.14 |
| SacCer2 | SacCer3 |         | 90.51 (95.00) | 6.53 (4.98) | 0.15 (0.024) | 2.81 |

We show for our selected species’ reference genomes, human (large), C. elegans (medium), yeast (small) how versions of the reference genome (row) are comprised of distinct regions (i.e., constant, updated, retired, new) in relation to a more recent version of the species. Each cell contains the percentages of the reference genome pair that each region (columns) comprises. The value in parentheses is the number of reads as a percentage of the read sets (used for the species) that originally mapped to the region in the old reference genome. Note 0% of the read set is mapped to the new region, since new regions do not exist in the old reference genome.
Fig. 6. **AirLift read reduction results.** We show the percentage of reads (out of the original read set) that we need to align to the new reference genome in order to account for the reads that state-of-the-art remapping tools do not translate. The x-axis sweeps various pairings of reference genomes where the naming convention is the old reference followed by the new reference. The y-axis is the percentage of reads that we must map to the new reference genome, and the specific values are written above each bar.

### 6.2 AirLift Runtime

We next look at how using 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 7 plots the speedup (y-axis) in execution time for mapping a read set to a new reference genome when using AirLift from an old reference genome compared to fully mapping the entire read set to the new reference genome. The execution time of AirLift is calculated as follows:

\[
T_{\text{AirLift}} = T_{\text{read extraction}} + T_{\text{map retired reads}} + T_{\text{map updated reads}} + T_{\text{lift constant reads}}
\]  

where \(T_{\text{read extraction}}\) is the time to extract the reads from the read set into subsets for each type of region they map to in the old reference genome, \(T_{\text{map retired reads}}\) and \(T_{\text{map updated reads}}\) are the times to map reads from retired and updated reads to the new reference genome, and \(T_{\text{lift constant reads}}\) is the time to directly shift coordinates from the old reference to the new reference based on the chain file that we generate. Note that we ignore the time to generate the LUTs since they are generated once per pair of reference genomes and can be used to remap any number of reads. We then divide the time to map the full read set to the new reference genome by \(T_{\text{AirLift}}\) to get the speedup that we plot. The x-axis sweeps various reference genome pairs that we use AirLift with, where the naming convention is the old reference followed by the new reference. We make three observations. First, we find that the speedup AirLift provides is inversely
Author's Suppressed Due to Excessive Length

proportional to the percentage of reads that mapped to the updated and retired regions of the old reference genome. As the percentage of reads in updated and retired regions are directly proportional to the sizes of the updated and retired regions (Table 4), which are generally correlated with the distance of the genome version numbers (within a species’ reference genome), we can generally claim that AirLift will perform better in reference genomes whose versions are closer. Second, when performing AirLift from the second most recent genome version to the most recent genome version across our selected species, AirLift provides $6.94 \times$, $44.0 \times$, and $16.4 \times$ speedup for large (human), medium (C. elegans), and small (yeast) reference genomes. Third, for a pair of references whose updated and retired regions are very small such as ce1ce2 ($\approx 0.053\%$) and ce4ce6 ($\approx 0.033\%$), AirLift can provide $188 \times$ and $202 \times$ speedup, respectively. We conclude that AirLift can significantly improve the time to remap 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.

7 Conclusion

In this work, we propose AirLift, a technique for comprehensively mapping a read data set that had previously been mapped to an older reference genome to a newer reference genome. AirLift exploits the similarity across reference genome versions to generate maps that describe the similarity across reference genomes and uses these to quickly identify locations that reads should directly be translated to or mapped to. AirLift is the first tool that enables a comprehensive mapping of reads from an old reference genome to a new reference genome, as prior state-of-the-art tools only translate alignments between regions with high similarity. When compared against the baseline of fully mapping a read data set to the new reference genome, we find that AirLift can reduce the overall number of reads that needs to be remapped from the original read set by up to 99% and reduce the overall runtime required to remap the reads from the old reference genome to the new reference genome by $6.94 \times$, $44.0 \times$, and $16.4 \times$ for large (human), medium (C. elegans), and small (yeast) reference genomes, respectively. We conclude that AirLift substantially reduces the overhead of remapping a read set to a new reference while still accounting for the significant changes in the new reference.

References

1. “Chain Format,” https://genome.ucsc.edu/goldenPath/help/chain.html.
2. “Sequence and Annotation Downloads,” http://hgdownload.soe.ucsc.edu/downloads.html.
3. 1000 Genomes Project Consortium, “A Global Reference for Human Genetic Variation,” Nature, vol. 526, no. 7571, p. 68, 2015.
4. S.-M. Ahn et al., “The First Korean Genome Sequence and Analysis: Full Genome Sequencing for a Socio-ethnic Group,” Genome Research, vol. 19, no. 9, pp. 1622–1629, 2009.
5. I. S. Al-Mssallem et al., “Genome Sequence of the Date Palm Phoenix dactylifera L,” Nature Communications, vol. 4, p. 2274, 2013.
6. C. Alkan et al., “Limitations of Next-Generation Genome Sequence Assembly,” Nature Methods, vol. 8, no. 1, p. 61, 2011.
7. 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.
8. S. Canzar and S. L. Salzberg, “Short Read Mapping: An Algorithmic Tour,” Proceedings of the IEEE, vol. 105, no. 3, pp. 436–458, 2015.
9. B. Gao, “Segment Liftover,” https://pypi.org/project/segment-liftover/.
10. B. Gao et al., “Segment_Liftover: A Python Tool to Convert Segments Between Genome Assemblies,” F1000Research, vol. 7, 2018.
11. B. Giardine et al., “Galaxy: A Platform for Interactive Large-scale Genome Analysis,” Genome Research, vol. 15, no. 10, pp. 1451–1455, 2005.
12. Y. Guo et al., “Improvements and Impacts of GRCh38 Human Reference on High Throughput Sequencing Data Analysis,” Genomics, vol. 109, no. 2, pp. 83–90, 2017.
13. T. Huang et al., “Genetic Differences among Ethnic Groups,” BMC Genomics, vol. 16, no. 1, p. 1093, 2015.
14. H. Li, “Aligning Sequence Reads, Clone Sequences and Assembly Contigs with BWA-MEM,” arXiv:1303.3997, 2013.
15. H. Li and R. Durbin, “Fast and Accurate Short Read Alignment with Burrows-Wheeler Transform,” Bioinformatics, vol. 25, no. 14, pp. 1754–1760, 2009.
16. X. Ma et al., “Analysis of Error Profiles in Deep Next-Generation Sequencing Data,” *Genome Biology*, vol. 20, no. 1, p. 50, 2019.
17. S. Mallick et al., “The Simons Genome Diversity Project: 300 Genomes from 142 Diverse Populations,” *Nature*, vol. 538, no. 7624, p. 201, 2016.
18. NCBI, “NCBI Genome Remapping Service,” https://www.ncbi.nlm.nih.gov/genome/tools/remap.
19. M. Ruffalo et al., “Comparative Analysis of Algorithms for Next-Generation Sequencing Read Alignment,” *Bioinformatics*, vol. 27, no. 20, pp. 2790–2796, 2011.
20. S. P. Sadedin and A. Oshlack, “Bazam: A Rapid Method for Read Extraction and Realignment of High-Throughput Sequencing Data,” *Genome Biology*, vol. 20, no. 1, p. 78, 2019.
21. S. C. Schuster et al., “Complete Khoisan and Bantu Genomes from Southern Africa,” *Nature*, vol. 463, no. 7283, p. 943, 2010.
22. D. Senol Cali et al., “Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions,” *Briefings in Bioinformatics*, vol. 20, no. 4, pp. 1542–1559, 2019.
23. R. M. Sherman et al., “Assembly of a Pan-genome from Deep Sequencing of 910 Humans of African Descent,” *Nature Genetics*, vol. 51, no. 1, p. 30, 2019.
24. H. G. Shukla et al., “hg19KIndel: Ethnicity Normalized Human Reference Genome,” *BMC Genomics*, vol. 20, no. 1, p. 459, 2019.
25. K. M. Steinberg et al., “Building and Improving Reference Genome Assemblies,” *Proceedings of the IEEE*, vol. 105, no. 3, pp. 422–435, 2017.
26. The Galaxy Team, “Galaxy,” https://www.usegalaxy.org.
27. K. Tretyakov, “Pyliftover,” https://pypi.org/project/pyliftover/.
28. UCSC, “UCSC Genome Browser: Sequence and Annotation Downloads,” http://hgdownload.soe.ucsc.edu/downloads.html.
29. UCSC, “UCSC LiftOver: Lift Genome Annotations,” https://genome.ucsc.edu/cgi-bin/hgLiftOver.
30. T. Ulrich, “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.
31. J. Wang et al., “The Diploid Genome Sequence of an Asian Individual,” *Nature*, vol. 456, no. 7218, p. 60, 2008.
32. P. Xu et al., “Genome Sequence and Genetic Diversity of the Common Carp, Cyprinus carpio,” *Nature Genetics*, vol. 46, no. 11, p. 1212, 2014.
33. H. Zhao et al., “CrossMap: A Versatile Tool for Coordinate Conversion Between Genome Assemblies,” *Bioinformatics*, vol. 30, no. 7, pp. 1006–1007, 2013.
34. 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.
35. X. Zheng-Bradley et al., “Alignment of 1000 Genomes Project Reads to Reference Assembly GRCh38,” *GigaScience*, vol. 6, no. 7, pp. 1–8, 2017.