Methods 79–80 (2015) 3–10

Contents lists available at ScienceDirect

Methods

journal homepage: www.elsevier.com/locate/ymeth

Fast and accurate mapping of Complete Genomics reads

Donghyuk Lee [a,1], Farhad Hormozdiari [b,1], Hongyi Xin [a], Faraz Hach [c], Onur Mutlu [a], Can Alkan [d,*]

[a] Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
[b] Department of Computer Science, University of California Los Angeles, Los Angeles, CA, USA
[c] School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
[d] Department of Computer Engineering, Bilkent University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 2 May 2014
Accepted 13 October 2014
Available online 22 October 2014

Keywords:
Complete Genomics
Read mapping
Gapped reads
High throughput sequencing

ABSTRACT

Many recent advances in genomics and the expectations of personalized medicine are made possible thanks to power of high throughput sequencing (HTS) in sequencing large collections of human genomes. There are tens of different sequencing technologies currently available, and each HTS platform have different strengths and biases. This diversity both makes it possible to use different technologies to correct for shortcomings; but also requires to develop different algorithms for each platform due to the differences in data types and error models. The first problem to tackle in analyzing HTS data for resequencing applications is the read mapping stage, where many tools have been developed for the most popular HTS methods, but publicly available and open source aligners are still lacking for the Complete Genomics (CG) platform. Unfortunately, Burrows-Wheeler based methods are not practical for CG data due to the gapped nature of the reads generated by this method. Here we provide a sensitive read mapper (sirFAST) for the CG technology based on the seed-and-extend paradigm that can quickly map CG reads to a reference genome. We evaluate the performance and accuracy of sirFAST using both simulated and publicly available real data sets, showing high precision and recall rates.

1. Introduction

High throughput sequencing (HTS) technologies continue to evolve in a fascinating speed since their first use for paired-end sequencing in [1], and whole genome shotgun sequencing (WGS) [2]. Many sequencing technologies were developed, including the Roche/454 [3], Illumina [4], AB SOLiD [5], and Pacific Biosciences [6], among others [7]. Many tools for read mapping [8], variant calling [9, 10], and genome assembly [11] have been developed for most of these platforms.

The availability of many different algorithms and their open source implementations helps researchers take more advantage of the data generated by the HTS technologies. However, there are no publicly available algorithms devised to map and analyze data generated by the Complete Genomics [12] (CG) platform. This is mainly due to the fact that the Complete Genomics company offers only sequencing service, and provides its own analysis results to their customers through the use of proprietary software. The only set of algorithms devised to analyze CG data are described by a group of scientists from the company [13], but the implementations of these algorithms remain proprietary and unreleased. Without doubt, the company's own analysis pipeline is fine tuned for data with different characteristics. Still, research projects that incorporate new algorithm development with the analysis of many genomes, such as the 1000 Genomes Project [14], may benefit from the availability of open-source tools and algorithms to further improve the analysis results. Having an open source tool to analyze CG data can enable other researchers to develop new findings that are otherwise not possible to discover.

In this paper, we present a new read mapper, sirFAST, designed for the data generated by the CG platform. Complete Genomics reads present characteristics different from all other HTS technologies, where a read is composed of several subreads (read sections) that may either overlap, or have gaps between each other. We designed sirFAST to efficiently align such reads to the reference assembly, at the presence of the flexible "expected" gaps/overlaps within each read. Due to the fact that the Burrows-Wheeler transform [15] based methods do not scale well with indels, we implemented sirFAST as a hash based seed-and-extend algorithm. Similar to its Illumina and SOLiD based counterparts [16, 17, 18, 19], sirFAST supports both SAM [20] and DIVET [21] file formats. The implementation of sirFAST is publicly available at http://sirfast.sf.net

* Corresponding author.
E-mail addresses: onur@cmu.edu (O. Mutlu), calkan@cs.bilkent.edu.tr (C. Alkan).
[1] Joint First Authors.

http://dx.doi.org/10.1016/j.ymeth.2014.10.012
1046-2023/

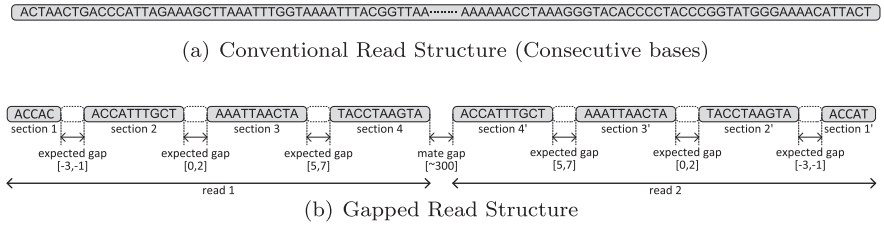

Fig. 1. Conventional read structure vs. Gapped read structure. Note that the read section structures of paired-end reads generated by the Complete Genomics platform are symmetric to each other. Read structure of more recent CG data is changed slightly, however, it is straightforward to provide support for the newer data type using the same algorithms presented in this manuscript.

2. Observation and problem

2.1. Unique characteristics of Complete Genomics reads

The Complete Genomics (CG) platform is based on DNA nanoball technology. Similar to the reads generated by other DNA sequencing technologies, CG provides paired-end reads where each read is 29–35 base-pairs (*bps*), and the fragment size (distance between the two ends) can be between 400 to 1500 bps.

In contrast with the other DNA sequencing technologies that generate reads with consecutive bases (typically 100–8000 bps depending on the platform, as shown Fig. 1(a)), a CG read shows a *gapped read structure* that consists of groups of consecutive bases denoted as *read sections*. There may be either gaps or overlaps between each read sections, denoted as *expected gaps*. Overlaps between read sections are represented as negative gaps. As shown in Fig. 1(b), each read includes four read sections of lengths 5, 10, 10 and 10 bases (35 bps length in total) respectively. Between these read sections, there are expected gaps with sizes that follow a normal distribution with specific mean values (-2, 1 and 6 respectively). As a result, the typical composition of a CG read consists of the following: (i) **first read Section (5 bps)**, (ii) expected gap ($[-3, -1]$ bps), (iii) **second Section (10 bps)**, (iv) expected gap ($[0, 2]$ bps), (v) **third Section (10 bps)**, (vi) expected gap ([5,7] bps), and (vii) **fourth Section (10 bps)**. More recent CG data sets now support slightly different read structure, where 29 bp reads are divided into three read sections of size 10 bp, 9 bp, and 10 bp, respectively[2]. Note that, this difference in the read structure requires only small changes in the implementation of our algorithm, and support for such reads are implemented through a parameter in sirFAST. In the remainder of the paper, we explain our mechanisms based on the assumption that the reads are in the former structure. This is mainly due to the lack of availability of publicly released data generated in this structure. However, we provide simulation-based test results in Table 2.

2.2. The challenge of mapping CG reads

Two paradigms for mapping reads to the reference genome are (1) seed-and-extend, and (2) Burrows-Wheeler Transformation with FM-indexing (BWT-FM) methods [8]. Unfortunately, all existing mappers are designed for mapping consecutive reads to the reference genome, thereby facing challenges or requiring much longer time to map the reads in a gapped read structure (as in CG). Specifically, mapping CG reads with BWT-FM is a harder problem since the expected gaps in CG reads show themselves as indels in an alignment, which increases the run time of BWT-FM search exponentially. Most of the currently available BWT-FM based aligners perform gapped alignment *per read*; instead, they anchor one end with a gap-free alignment, and apply gapped alignment for its mate in regions of close proximity. It is therefore impractical to use BWT-FM based approach for CG reads as both ends contain expected gaps. Although BWT-FM methods outperform seed-and-extend methods in aligning consecutive basepair reads with very low sequence errors, seed-and-extend techniques scale better with high error rate, especially when the errors are indels. We therefore designed our mapper to use the seed-and-extend technique similar to several others available in the literature [8].

Fig. 2 shows the typical operation of a seed-and-extend mapper. The mapper first selects a set of short subsequences of a read as seeds, and quickly queries their locations in the genome using a hash table or a similar data structure. The mapper then tries to extend these initial seed locations by calculating the alignment of the full read against pieces of reference genome at these seed locations. This extension step is commonly called *verification* where the *similarity score* is calculated between the read and the reference. Each insertion, deletion and substitution has a positive score while each matching base has a zero score. A smaller score denotes higher similarity between the read and the reference. The alignment algorithms used in this step are dynamic programming algorithms which tackles insertions and deletions. The complexity of the verification step per seed location is quadratic, which can be reduced to $O(kn)$ if k is an upper bound for allowed indels in the alignment. In the remainder of the paper, we refer to the alignment algorithms as *bp-granularity mapping algorithms*.

3. Methods

We now explain how we map the reads generated by the CG platform to the reference genome using the seed-and-extend methodology (Fig. 3).

- **Selecting Seeds.** The performance and accuracy of seed-and-extend mappers depend on how the seeds are selected in the first stage. The seeds should be selected in a way that guarantees: (1) very high sensitivity to make sure real mapping locations are captured, (2) high specificity to avoid unnecessary alignments, (3) fast seed placement, preferably in $O(1)$ time, and (4) low memory usage. The two main types of seeds are (1) *consecutive seeds* as used in BLAST [22], and (2) *spaced seeds*, first defined in PatternHunter [23]. Spaced seeds carefully select bases from a larger consecutive region of the read to form seeds with gaps in them in order to reduce seed locations. Spaced seeds are considered to be more powerful in terms of higher sensitivity and specificity than consecutive seeds as they provide fewer locations to verify. However, spaced seeds are not suitable for CG reads due to the lack of large consecutive regions in the read (the lengths of expected gaps are flexible and the read sections are only 10-base-long). We, therefore, use *consecutive seeds* of length 10, and implement our seed index using a

[2] http://media.completegenomics.com/documents/DataFileFormats_Standard_Pipeline_2.5.pdf