Bioinformatics, 31(19), 2015, 3207–3209
doi: 10.1093/bioinformatics/btv280
Advance Access Publication Date: 3 June 2015
Applications Note

OXFORD

Genome analysis

FinisherSC: a repeat-aware tool for upgrading *de novo* assembly using long reads

Ka-Kit Lam[1], Kurt LaButti[2], Asif Khalak[3,†] and David Tse[1,4,*]

[1]Department of Electrical Engineering and Computer Sciences, UC Berkeley, [2]U.S. Department of Energy Joint Genome Institute, Walnut Creek, [3]Pacific Biosciences, Menlo Park, and [4]Department of Electrical Engineering, Stanford University, Palo Alto, CA, USA

*To whom correspondence should be addressed.
[†]Present address: Samsung SSIC, mHealth, Menlo Park, CA, USA
Associate Editor: John Hancock

Received on November 4, 2014; revised on April 3, 2015; accepted on April 27, 2015

Abstract

We introduce FinisherSC, a repeat-aware and scalable tool for upgrading *de novo* assembly using long reads. Experiments with real data suggest that FinisherSC can provide longer and higher quality contigs than existing tools while maintaining high concordance.
Availability and implementation: The tool and data are available and will be maintained at http://kakitone.github.io/finishingTool/
Contact: dntse@stanford.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

In *de novo* assembly pipelines for long reads, reads are often trimmed or thrown away. Moreover, there is no evidence that state-of-the-art assembly pipelines are data-efficient. In this work, we ask whether state-of-the-art assembly pipelines for long reads have already used up all the available information from raw reads to construct assembly of the highest possible quality. To answer this question, we first collect output contigs from the HGAP (Chin *et al.*, 2013) pipeline and the associated raw reads. Then, we pass them into our tool FinisherSC to see if higher quality assemblies can be consistently obtained after post-processing.

2 Methods

2.1 Usage and pipeline

FinisherSC is designed to upgrade *de novo* assembly using long reads (e.g. PacBio reads). It is especially suitable for data consisting of a single long reads library. Input to FinisherSC are contigs (contigs.fasta) constructed by an assembler and all the raw reads with adaptors removed (raw_reads.fasta). Output of FinisherSC are upgraded contigs (improved3.fasta) which are expected to be of higher quality than its input (e.g. longer N50, longer longest contigs, fewer number of contigs, high percentage match with reference, high-genome fraction, etc). In Figure 1, we show an example pipeline in which FinisherSC can fit. As shown in Figure 1, FinisherSC can be readily incorporated into state-of-the-art assembly pipelines (e.g. PacBio HGAP).

2.2 Algorithm and features

The algorithm of FinisherSC is summarized in Algorithm 1. Detailed description of the algorithm is in the supplementary materials. We summarize the key features of FinisherSC as follows.

- Repeat-aware: FinisherSC uses a repeat-aware rule to define overlap. It uses string graphs to capture overlap information and to handle repeats so that FinisherSC can robustly merge contigs. There is an optional component, X-phaser (Lam *et al.*, 2014), that can resolve long approximate repeats with two copies by using the polymorphisms between them. There is also an optional component, T-solver that can resolve tandem repeat by using the copy count information.
- Data-efficient: FinisherSC utilizes all the raw reads to perform re-layout. This can fill gaps and improve robustness in handling repeats.
- Scalable: FinisherSC streams raw reads to identify relevant reads for re-layout and refined analysis. MUMMER (Kurtz *et al.*, 2004) does the core of the sequence alignment. Although

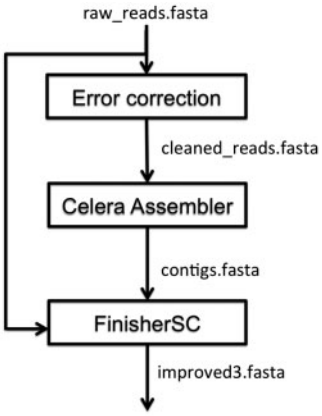

Fig. 1. Pipeline where FinisherSC can fit in

MUMMER is single threaded, we provide an option to segment the files and run multiple MUMMER jobs in parallel. These techniques allow FinisherSC to be easily scalable to high volume of data.

Algorithm 1. Main flow of FinisherSC
Input: contigs.fasta, raw_reads.fasta
Output: improved3.fasta
1. Filter completely embedded contigs
2. Form a string graph with the BEST successors/predecessors as edges
3. Condense the string graph by contracting edges with both in-degree and out-degree being 1
4. Use raw reads to declare potential successors/predecessors of dangling contigs
5. Merge contigs (with gaps filled by reads) when they respectively only have 1 successor/1 predecessor
6. Form a string graph with ALL successors/predecessors as edges
7. Merge contigs with only 1 predecessor or 1 successor and each has no more than two competing edges

3 Results and discussion

3.1 Experimental evaluation on bacterial genomes

We evaluated the performance of FinisherSC as follows. Raw reads were processed according to the pipeline in Figure 1. They were first error corrected and then assembled into contigs by an existing pipeline (i.e. HGAP). Contigs were upgraded using FinisherSC and evaluated for quality with Quast (Gurevich *et al.*, 2013). The data used for assessment are real PacBio reads. These include data recently produced at JGI and data available online supporting the HGAP publication. We compared the assembly quality of the contigs coming out from the Celera assembler (Myers *et al.*, 2000) of HGAP pipeline, the upgraded contigs by FinisherSC and the upgraded contigs by PBJelly (English *et al.*, 2012). A summary of the evaluation is shown in Figure 2. More details can be found in the supplementary materials. We find that FinisherSC can upgrade the assembly from HGAP without sacrifice on accuracy on these test cases. Moreover, the upgraded contigs by FinisherSC are generally of higher quality than those upgraded by PBJelly. This suggests that there is extra information from the reads that is not fully utilized by state-of-the-art assembly pipelines for long reads.

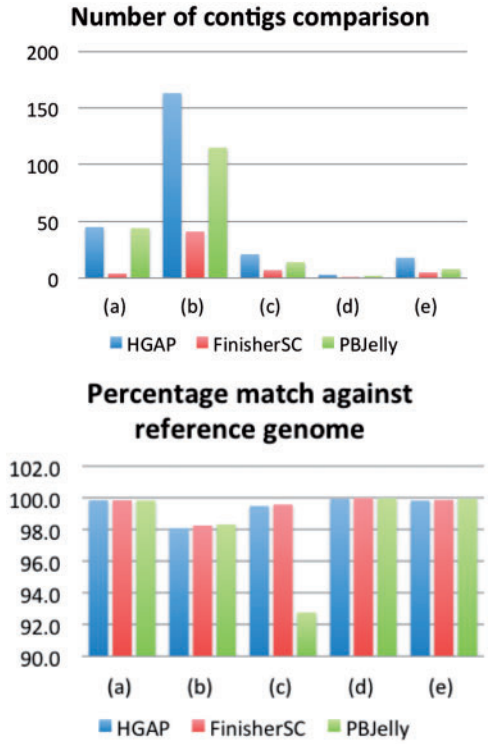

Fig. 2. Experimental evaluation on bacterial genomes. **(a and b)** Pedobacter heparinus DSM 2366 (PacBio long reads from JGI) **(c–e)** *Escherichia coli* MG 1655, Meiothermus ruber DSM 1279, Pedobacter heparinus DSM 2366 (PacBio long reads supporting the HGAP publication)

Table 1. Summary of running time for the experiments on scalability

Genome name	Genome size (Mb)	Size of reads (Gb)	Running time (h)
Caenorhabditis elegans	104	7.65	23
Drosophila	138	2.27	9.4
Saccharomyces cerevisiae	12.4	1.40	0.66

3.2 Experiments on scalability

We tested the scalability of FinisherSC by applying it to handle larger genomes. The data used are the benchmark data available on PacBio Devnet. We run FinisherSC with the option of using 20 threads (-par 20) on a server computer. The server computer is equipped with 64 cores of CPU at clock rate of 2.4–3.3 GHz and 512 GB of RAM. The running time is tabulated in Table 1.

3.3 Discussion

Although FinisherSC was originally designed to improve *de novo* assembly by long reads, it can also be used to scaffold long contigs (formed by short reads) using long reads. For that use case, we note that the contigs formed by short reads can sometimes have length shorter than the average length of long reads. Therefore, we suggest users to filter out those short contigs before passing them into FinisherSC.

Funding

The authors K.K.L and D.T. are partially supported by the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant

agreement CCF-0939370. The work conducted by the U.S. Department of Energy Joint Genome Institute, a DOE Office of Science User Facility, is supported under Contract No. DE-AC02-05CH11231.

Conflict of interest: none declared.

References

Chin,C.-S. *et al*. (2013) Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data. *Nat. Methods,* **10**, 563–569.

English,A.C. *et al*. (2012) Mind the gap: Upgrading genomes with pacific biosciences rs long-read sequencing technology. *PLoS One*, **7**, 47768.

Gurevich,A. *et al*. (2013) Quast: quality assessment tool for genome assemblies. *Bioinformatics*, **29**, 1072–1075.

Kurtz,S. *et al*. (2004) Versatile and open software for comparing large genomes. *Genome Biol*., **5**, R12.

Lam,K.-K. *et al*. (2014) Near-optimal assembly for shotgun sequencing with noisy reads. *BMC Bioinformatics,* **15** (Suppl. 9), S4.

Myers,E.W. *et al*. (2000) A whole-genome assembly of Drosophila. *Science*, **287**, 2196–2204.