Nanopore Sequencing Technology and Tools: Computational Analysis of the Current State, Bottlenecks, and Future Directions

Damla Senol[1], Jeremie Kim[1], Saugata Ghose[1], Can Alkan[2] and Onur Mutlu[3,1]
[1] *Carnegie Mellon University, Pittsburgh, PA, USA* [2] *Bilkent University, Bilkent, Ankara, Turkey* [3] *ETH Zürich, Zürich, Switzerland*

Nanopore sequencing, a promising single-molecule DNA sequencing technology, exhibits many attractive qualities and, in time, could potentially surpass current sequencing technologies. Nanopore sequencing promises higher throughput, lower cost, and increased read length, and it does not require a prior amplification step. Nanopore sequencers rely solely on the electrochemical structure of the different nucleotides for identification, and measure the ionic current change as long strands of DNA (ssDNA) pass through the nanoscale protein pores.

Biological nanopores for DNA sequencing were first proposed in the 1990s, but were only made commercially available in May 2014 by Oxford Nanopore Technologies (ONT). The first commercial nanopore sequencing device, MinION, is an inexpensive, pocket-sized, high-throughput sequencing apparatus that produces real-time data using the R7 nanopore chemistry. These properties enable new potential applications of genome sequencing, such as rapid surveillance of Ebola, Zika or other epidemics, near-patient testing, and other applications that require real-time data analysis. This technology is capable of generating very long reads (~50,000bp) with minimal sample preparation. Despite all these advantageous characteristics, it has one major drawback: high error rates. In May 2016, ONT released a new version of MinION that uses a nanopore chemistry called R9. Although R9 improves data accuracy over R7, the error rate remains high. To take advantage of the real-time data produced by MinION, the tools used for nanopore sequence analysis must be fast and must overcome high error rates.

Our goal in this work is to comprehensively analyze current publicly available tools for nanopore sequence analysis, with a focus on understanding the advantages, disadvantages, and bottlenecks of them. It is important to understand where the current tools do not perform well in order to develop better tools. To this end, we analyze the multiple steps and tools in the nanopore genome analysis pipeline; and also provide some guidelines for determining the appropriate tools for each step of the pipeline and the corresponding parameters of them.

The first step, *basecalling*, translates the raw signal output of MinION into nucleotides to generate DNA sequences. *Metrichor* is the cloud-based basecaller of ONT; while *Nanocall* and *Nanonet* are publicly available nanopore basecallers. Overlap-layout-consensus (OLC) algorithms are used for nanopore sequencing reads since they perform better with longer error-prone reads. The second pipeline step finds read-to-read overlaps. *Minimap* and *GraphMap* are the commonly used tools for this step. After finding the overlaps, OLC-based assembly algorithms generate an overlap graph, where each node is a read and each edge is an overlap connecting them. The third pipeline step, genome assembly, traverses this graph, producing the layout of the reads and then constructing the draft assembly. *Canu* and *Miniasm* are the commonly used error-prone long-read assemblers. In order to increase the accuracy of the assembly, further polishing may be required. The first step of polishing is mapping the raw basecalled reads to the generated draft assembly from the previous step. The most commonly used long read mapper is *BWA-MEM*. After aligning the basecalled reads to the draft assembly, the final polishing of the assembly can be performed with *Nanopolish*.

We analyze the aforementioned nanopore sequencing tools with the goals of determining their bottlenecks and finding improvements to these tools. First, we compare the performance of the chosen tools for each step in terms of accuracy and speed. After the basecalling, read overlap finding, and assembly steps, the generated draft assemblies are compared with their reference genome; and the coverage of and identity with the reference genome are used to gauge their accuracy. For two of the draft assemblies, read mapping and polishing steps are further applied and the generated polished sequences are compared similarly. The execution time of each tool is recorded in order to compare the performance of the tools. Second, we analyze the first two steps of the pipeline in detail in order to assess the scalability of these tools. The performance of each basecaller and each read overlap finder as we vary the thread count is analyzed; wall clock time, peak memory usage, and parallel speedup are the metrics used for comparison. We present our key results in this work, and we expect future work to examine other stages of the pipeline and provide end-to-end results and analyses.