Apollo: a sequencing-technology-independent, scalable and accurate assembly polishing algorithm.

Abstract: Motivation: Although third generation sequencing technologies have the ability of sequencing long reads but have large amount of errors so it can have incorrect reads. Because of these errors we used Assembly polishing algorithms to fix these errors by using information from alignments between reads and the assembly but it polish small assembly or certain sequencing technology. **Result:** Because of this problems**,** we use apollo. It's a universal assembly polishing algorithm and accurate to polish an assembly of any size. Our aim of using it is to provide a single algorithm as the only algorithm that can polish large algorithm and uses read sets from all available sequencing technologies.

Introduction: High-throughput sequencing are used to generate a huge value of sequencing data at low cost. it have two significant limitations, the first one can only sequence fragments of the genome. this lead to reconstruct the original full sequence. the second one is that they introduce non-negligible insertion, deletion and substitution errors into reads. It depends on the method for reconstructing the original sequence. HTS is divided into two types: second and third generation sequencing technologies. Second generation sequencing technologies generate the most accurate reads but the length of the read is short. This introduces challenges in both read alignment and de novo genome assembly. In read alignment, a short read can align to multiple locations in a reference well. It should be found a high computational complexity in de nova to identify overlaps between reads. Molecule Real Time and Oxford Nanopore Technologies are able to produce long reads. PacBio reads can have more insertion errors than other error types while insertion errors are the least common errors for ONT reads. Existing solutions that try to solve the problem of errors assemblies when using de novo genome assembly can be divided to two types. First: a typical solution is to correct the errors of long reads by using high coverage reads from the same sequencing technology or additional reads from more reliable second-generation sequencing technologies. There are several available error correction algorithms that use additional reads to locate and correct errors in long reads, LoRDEC, LSC, and LoRMA Second: method for removing errors in an assembly is called assembly polishing, Using the alignments of either long or short reads to the assembly, an assembly polishing method helps to correct the assembly's errors. There are a variety of assembly polishing algorithms that use different methods for detecting and changing dissimilarities in the assembly like Racon, Quiver and Pilon. Many of these assembly polishing algorithms have the drawback of only working for reads from a small number of sequencing technologies. For two reasons, there are scalability issues of using polishing algorithms to polish a broad genome and, as a result, running assembly polishing algorithms several times. first: None of the polishing algorithms can handle massive genomes in a single run because they demand a lot of computing power. Second: splitting a large genome into smaller contigs and running polishing algorithms several times necessitates additional work to compile and combine the multiple findings into a polished large genome assembly. Apollo's machine learning algorithm is based on two key steps: training and decoding. We compare Apollo with Nano polish, Racon, Quiver and Pilon using datasets that are sequenced with different technologies. We first demonstrate that Apollo scales better than other polishing algorithms in polishing assemblies of large genomes using moderate and high coverage readings, using datasets from various sequencing technologies. Apollo is the only algorithm that can use reads from multiple sequencing technologies in a hybrid manner.

Related Work: Different third-generation sequencing technologies result in different error profiles, There are two kinds of existing solutions for overcoming the issue of error prone assemblies by using de novo genome assembly. First, a typical solution is to correct the errors of long reads. Errors are corrected by using high coverage reads. The second method for removing errors in an assembly is called assembly polishing, An assembly polishing process attempts to correct the errors of the assembly using the alignments of either long or short reads to the assembly. There are various assembly polishing algorithms that use various methods for discovering dissimilarities and modifying the assembly like Racon, Pilon. there are scalability problems associated with using polishing algorithms to polish a large genome and, therefore, running assembly polishing algorithms multiple times for two reasons, these assembly polishing algorithms cannot polish large genomes in a single run if the available computational resources are not tremendous. Apollo, that corrects errors in an assembly. when we compare Apollo to other competing algorithms, these experiments show that Apollo usually produces assemblies of similar accuracy to competing algorithms: Nano polish, Pilon, Racon and Quiver. Apollo produces assemblies with less accuracy than that of Racon and Quiver. These experiments are based on ground truth k-mer similarity calculation between an Illumina set of reads and a polished assembly. These comparisons show that Apollo can polish an assembly using reads from any sequencing technology while still generating an assembly with accuracy usually comparable to the competing algorithms