GenMPI: Cluster Scalable Variant Calling for Short/Long Reads Sequencing Data

Tanveer Ahmad, *Member, IEEE*, Joseph Schuchart, *Member, IEEE*, Zaid Al Ars, *Fellow, IEEE*, Christoph Niethammer, *Member, IEEE*, José Gracia, *Fellow, IEEE*, and H. Peter Hofstee, *Member, IEEE*

***Abstract*—Rapid technological advancements in sequencing technologies allow producing cost effective and high volume sequencing data. Processing this data for real-time clinical diagnosis is potentially time-consuming if done on a single computing node. This work presents a complete variant calling workflow, implemented using the Message Passing Interface (MPI) to leverage the benefits of high bandwidth interconnects. This solution (GenMPI) is portable and flexible, meaning it can be deployed to any private or public cluster/cloud infrastructure. Any alignment or variant calling application can be used with minimal adaptation. To achieve high performance, compressed input data can be streamed in parallel to alignment applications while uncompressed data can use internal file seek functionality to eliminate the bottleneck of streaming input data from a single node. Alignment output can be directly stored in multiple chromosome-specific SAM files or a single SAM file. After alignment, a distributed queue using MPI RMA (Remote Memory Access) atomic operations is created for sorting, indexing, marking of duplicates (if necessary) and variant calling applications. We ensure the accuracy of variants as compared to the original single node methods. We also show that for 300x coverage data, alignment scales almost linearly up to 64 nodes (8192 CPU cores). Overall, this work outperforms existing Big Data based workflows by a factor of two and is almost 20% faster than other MPI-based implementations for alignment without any extra memory overheads. Sorting, indexing, duplicate removal and variant calling is also scalable up to 8 nodes cluster. For pair-end short-reads (Illumina) data, we integrated the BWA-MEM aligner and three variant callers (GATK HaplotypeCaller, DeepVariant and Octopus), while for long-reads data, we integrated the Minimap2 aligner and three different variant callers (DeepVariant, DeepVariant with WhatsHap for phasing (PacBio) and Clair3 (ONT)).**

***Index Terms*—MPI, pthreads, scalable, alignment, HPC, clusters, NGS, variant calling, WGS.**

Received 8 July 2023; revised 15 June 2025; accepted 21 July 2025. Date of publication 5 August 2025; date of current version 3 April 2026. This work was supported in part by the Dutch National E-infrastructure with the support of SURF Cooperative. The research conducted in this work is co-supported by HPC-Europa3 which receives funding from the European Union's Horizon 2020 research and innovation programme under Grant 730897. This research was supported by the Exascale Computing Project under Grant 17-SC-20-SC, a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. *(Corresponding author: Tanveer Ahmad.)*

Tanveer Ahmad, Zaid Al Ars, and H. Peter Hofstee are with the Department of Quantum and Computer Engineering, EEMCS, Delft University of Technology, 2628, CD Delft, The Netherlands (e-mail: tahashmipk@gmail.com; Z.Al-Ars@tudelft.nl).

Joseph Schuchart is with the Innovative Computing Laboratory, University of Tennessee, Knoxville, TN 37996 USA.

Christoph Niethammer and José Gracia are with the High-Performance Computing Center Stuttgart (HLRS), 70597 Stuttgart, Germany.

All codes and scripts are available at: https://github.com/abs-tudelft/gen-mpi.

Digital Object Identifier 10.1109/TCBBIO.2025.3595409

I. Introduction

NEXT Generation Sequencing (NGS) technologies can produce high throughput, less erroneous and higher depth/coverage sequencing data at low costs. These high-throughput sequencing technologies are making their way from research to the field in a wide range of applications ranging from clinical diagnostics to agriculture. Depending on the experiment design type, the need of sequencing coverage varies. A typical 300x coverage human genome dataset size exceeds 2.3 TBytes [1]. Processing such large datasets on a single-node can take up to several days. For a number of applications employing higher coverage sequencing data and/or requiring urgent sequencing results, this exceeds acceptable time constraints.

Sequencing coverage influences both accuracy and the sensitivity of genomics analysis. In pediatric brain-tumor studies [2], more than 200x coverage for the tumor sample and more than 100x coverage for the normal sample are collected for better precision for the tumors of concern. Recent studies show that whole exome sequencing (WES)/whole genome sequencing (WGS) helps with diagnosis, decision making, and treatment of fetal diseases [3]. Although WES is normally used to detect fetus anomalies in prenatal and perinatal testing, which only targets protein-coding regions of genes in a genome, this type of sequencing needs urgent and fast sequencing analysis due to the time-critical nature of these tests. Sequencing can also enable finding some rare hereditary disorders and genetic variants associated with specific diseases in newborn screening. A recent study that uses genomic sequencing for newborn screening [4] showed that some of enrolled healthy newborns and children with metabolic diseases or hearing loss exhibited pathogenic variants associated with hereditary breast or ovarian cancer and a pathogenic variant in the gene associated with Lowe syndrome. This shows that sequencing in newborn screening can play a vital role in timely diagnosis and treatment of diseases and ultimately will lead to urgent need of processing genomics data in such time-critical clinical settings. Many consortia, associations and government disease diagnosis and drug regulatory agencies stipulate guidelines/protocols for genomics sequencing as a standard tool in diagnosis and treatment of some fatal diseases [5]. Many medical and diagnosis centers, particularly in developed countries, have started sequencing as a regular practice for prenatal and perinatal testing, newborn screening, genetic and cancerous disease diagnostic and personalized treatments. In the coming decades, as sequencing becomes a

normal practice for human health and other types of research, locally available compute infrastructure to any organization will not be adequate to fulfill the sequencing requirements. At the same time, large computing infrastructure also requires human resources and incurs power and maintenance costs. Mostly, genome sequencing machines produce FASTQ format raw data for a given sample [6]. Mapping this raw data to a reference genome, sorting the resultant Sequence Alignment/Map (SAM) reads according to chromosomes and their positions, and finally removing the duplicate reads are some of the standard practices before actual variant detection. A large number of tools is available for the aforementioned methods but almost all are developed for use in a single compute node context. Scaling these tools for cluster environments for both scalability and reproducibility is an ongoing challenge. In the past decade, many cluster-scaled solutions have been created, almost all these solutions use Big Data frameworks like Apache Spark [7] or Apache Hadoop [8], [9] for distributing and work scheduling. Data formats like Apache Parquet, Apache Arrow, Apache Avro have been explored extensively in conjunction with these frameworks to store and process genomic data efficiently. These frameworks include ADAM [10], SparkGA2 [11], VC@Scale [12] and Halvade [13]. Some existing works which use Apache Arrow for genomics applications include ArrowSAM [14] and [15]. Due to many underlying dependencies, inefficient memory usage, issues related to scalability, cluster deployment challenges as well as incompatible data formats, solutions based on these frameworks are still not widely used in the mainstream Bioinformatics community. While the Message Passing Interface (MPI) has been used previously for parallelization of genomic algorithms on HPC systems like pBWA [16], in this work we explore a comprehensive approach towards using MPI for cluster scale parallelization of genomics applications and complete variant calling workflows. One of the main advantages of using MPI is reducing the large memory consumption incurred by running Big Data pipelines due to their heavy dependence on memory-inefficient virtualization using JVM. Furthermore, the usage of intermediate steps for data type conversion and compression in genomics applications can cause challenges for Big Data pipelines, which increases the difficulty in correctly configuring Spark/Hadoop-based frameworks for WGS analysis.

We expect to benefit from the following advantages that MPI has over traditional Big Data frameworks for genomics workflows.

- Bare-metal performance and linear scalability of existing applications;
- Portable access to low-level network capabilities;
- Little to no extra memory overheads otherwise incurred in Big Data frameworks;
- Efficient MPI I/O performance on parallel file systems (like Lustre [17], GPFS [18]).

In the following, we list the main contributions of this work.

- MPI-based parallelization of BWA-MEM [19] and Minimap2 [20], compressed input FASTQ files can be provided as separate files to parallelize streaming of input while reading of uncompressed FASTQ will be parallelized internally. SAM output is tested on both POSIX and shared MPI I/O.
- Sorting, indexing and duplicate reads removal (if necessary) can be performed through a queue employing low-level network atomic operations (if input is already chunked based on chromosomes) or through MPI based bitonic sorting.
- For short reads, GATK HaplotypeCaller [21], Octopus [22], and DeepVariant [23] are used in combination with the MPI RMA-based queue for parallel chromosomes processing on a cluster.
- For long reads, DeepVariant, DeepVariant with WhatsHap [24] for phasing, and Clair3 [25] variant callers are used in combination with the MPI RMA-based queue for parallel chromosome processing on a cluster.
- Resultant VCFs are merged through Bcftools [26] to generate a single combined VCF (variant calling file) and to insure variants correctness.
- SNP/INDEL accuracy/precision/F1 tests are performed through hap.py [27] against GIAB v4.2.1 [28] benchmark set for the HG002 dataset.

GATK HaplotypeCaller and Octopus are the highest accuracy performing methods for germline short reads, while DeepVariant and Clair3 are the highest accuracy performing methods for germline long reads WGS short-variant calling analysis. The rest of this article is organized as follows. In Section II we describe our implemented methods in detail for both pre-processing and variant calling for short and long reads NGS data, followed by Section III where we compare the methods integrated into this approach with the existing workflows for both performance, accuracy, and scalability. In Section IV, run-time, accuracy, scalability, portability, and cost efficiency are discussed briefly. Finally we conclude this work in Section V.

II. Methods

Methods section elaborates MPI simple yet scalable implementation into existing BWA and Minimap2 alignment algorithms as compared to the QUARTIC MPI based BWA implementation while eliminating the configuration complexity of the Spark/Hadoop based approaches and need additional resources as well as larger memory overheads for intermediate operations. We have constructed both short and long reads based cluster scalable variant calling workflows. For pre-processing of short and long reads NGS data, the *BWA-MEM* [19] aligner with sorting (*Samtools* [29]) and mark duplicate (*Sambamba* [30]) are used for the former, while *Minimap2* [20] with sorting (*Samtools*) is used for the latter. Alignment MPI implementation adopts load-balancing approach which enables almost linear scalability upto number of nodes and cores/threads available on a single node. As shown in Fig. 1, for short reads we use three different variant callers like *Octopus* and *DeepVariant* as well as GATK best practices variant calling pipeline using *HaplotypeCaller*. Similarly, for PacBio long reads, DeepVariant and DeepVariant with chromosomes phasing using *WhatsHap* have been used while *Clair3* variant caller is used for Oxford Nanopore Technologies (ONT) data as recommended by ONT [31]. These workflows can be run through both cluster workload managers like Slurm [32] and PBS [33]. The following sections provide a

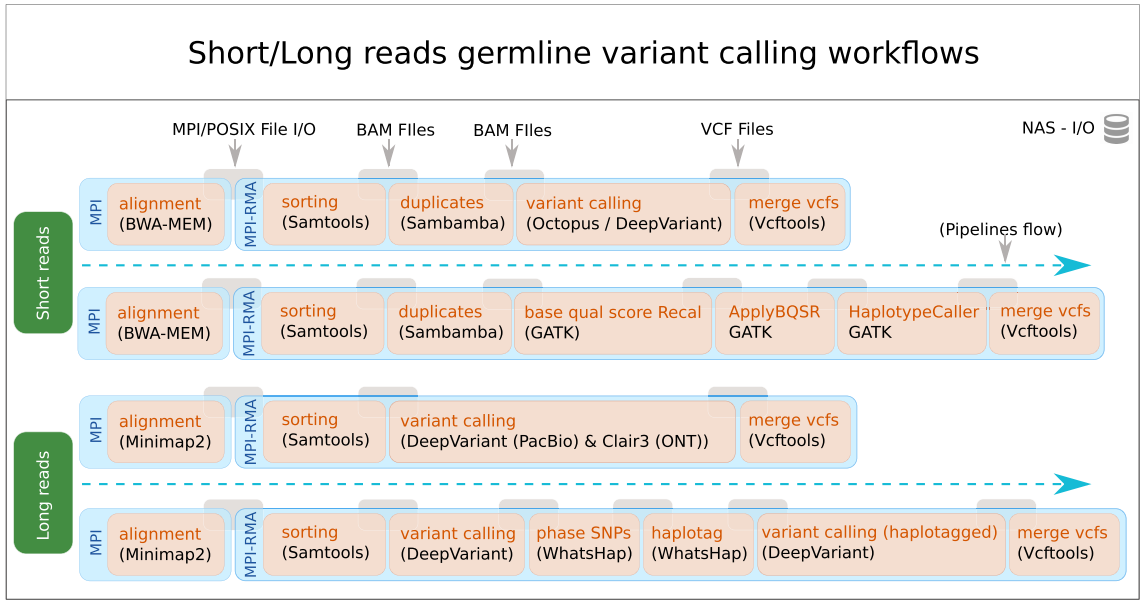

Fig. 1. Architectural description of short and long reads based NGS data variant calling workflows.

more detailed description of both short and long reads workflows implementation.

A. Short-Reads Variant Calling Workflow

In the following subsections, we describe in details how MPI has been integrated with the BWA-MEM short read aligner and how different pre-processing applications are used in the variant calling workflow.

1) Alignment: The complete algorithmic implementation details of integrating the alignment applications is given in pseudo-code representation in Algorithms 1 and 2. Normally, pair-end short reads are used for variant detection, consisting of two FASTQ raw NGS data files. As shown in Algorithm 1, we start MPI after basic parameter initialization. First, we check (Lines 5–10) if the user enabled chromosomes-based output into separate SAM files. By default this option is disabled and only single output SAM file generation is enabled. For compressed FASTQ input files (Lines 13–16), each BWA-MEM process expects equally chunked FASTQ pairs in a directory "parts" in the path of the original FASTQ files. This is done through an extra FASTQ streaming process. For streaming purpose, the user has to start an additional process; we used SeqKit [34] for this purpose. SeqKit is an efficient multi-threaded command line FASTQ/FASTA data manipulation tool. In the case of uncompressed FASTQ files (Lines 18–21), we used gzseek() function which sets the file position to a given offset. We get this offset by dividing the total FASTQ file size to the number of total MPI processes. Through this technique the kseq_read() function may encounter a broken first read, which we simply discard because process $N-1$ will read the last read even if it reaches a break point for total number of bytes it can read, as shown in Algorithm 2 (Lines 1–8) in *process()* function *step-0*. In this way, we ensure that none of the input reads is dropped by any process. We calculate the size of each read and increment the bytes counter until it reaches the required number of bytes for each BWA-MEM process. Afterwards, BWA-MEM starts processing these sequences (Lines 9) in *process()* function *step-1*. Finally, we distinguish the chromosomes id for each read if writing to individual chromosome files is enabled (Lines 10–17) in *process()* function *step-2*. The output files should be written on a parallel file system with either POSIX or shared MPI I/O, depending on the best possible performance scenarios for the user.

2) MPI RMA-Based Chrosome Queue: The MPI RMA interface provides applications with access to advanced features of modern high-performance networks, including direct access to remote memory through puts, gets, and atomic operations [35]. We utilize the atomic fetch-and-add functionality of MPI RMA to assign chromosomes to processes. Processes continuously increment the chromosome counter atomically until the returned value is equal to or larger than the number of chromosomes. Algorithm 3 shows the algorithm for allocating a suitable window and then querying chromosome numbers and processing them until all chromosomes have been assigned to a process. The respective window is allocated using the `"osc_rdma_acc_single_intrinsic"` info key set to true (Lines 1–2), which is supported by Open MPI and allows the implementation to utilize low-level hardware atomic operations by signaling the use of a single data element [36]. Processes continuously increment a variable using an atomic fetch-and-op in the window to acquire the next chromosome to process (Lines 5–7) and stop once the counter exceeds the number of chromosomes available (Line 9).

3) Sorting, Duplicates Removal, Indexing and Variant Calling: As described above, the MPI atomic operations based queue algorithm continuously pools the input chromosomes SAM files and operates on them for sorting, mark duplicate, index generation and variant calling algorithms. We used and integrated the sorting algorithm from Samtools and the mark

Algorithm 1: Part-1: MPI Integration Into **BWA-MEM** and **Minimap2** for Reading Compressed and Uncompressed FASTQ Input Files.

```
// BWA-MEM/Minimap2 basic initialization
MPI_Init(NULL, NULL)                          ▷ MPI initialization
P ← size(totalnumberofprocesses);
N ← rank(processnumber);
                ▷ Check if user needs a single SAM or chromosomes-based SAM file(s) to write output
if chromosomes ← output then
    for CHRM ← 1 to chromosomes do
        files[CHRM] ← fopen(CHRM);
    end for
else
    file ← fopen(outputfile)
end if
 ▷ Compressed or uncompressed input FASTQ file(s) check
if input(compressed (*.gz/*.tar.gz/*.zip)) then
    // FASTQ streaming from an external process
    Seekable ← FALSE;
    parts[n] ← scandir(./parts);
    FASTQ_INPUT ← parts[P];
else
    // SEEK_SET pointer initialization
    Seekable ← TRUE;
    size(bytes) ← Calculatefile(s)size;
    SEEKpointer* ← size(bytes)/P;
    gzseek(file, SEEKpointer*);
end if
```

Algorithm 2: Part-2: MPI Integration Into **BWA-MEM** and **Minimap2** for Processing FASTQ Data and Saving Output to SAM Single or Chromosome-Based File.

```
while (SEQs ≠ ∅)(FASTQ(compressed_file_chunk) ≠ END || FASTQ(SEEK_SET) ≠ END) do
    SEQs ← bseq_read(FASTQ_INPUT)
        // process() -> Step:0
    if Seekable = TRUE then
        bytes+ = Bytescountforeachread(seq, commentandquallength);
        if bytes ≥ max(SEEK_SET) then
            break;
        end if
    end if
    SAM ← mem_process_seqs(SEQs)// process() -> Step:1
    if chromosomes ← output // process() -> Step:2
    then
        for CHRM ← 1 to chromosomes do
            files[CHRM] ← err_fputs(files[CHRM], SAMs);
        end for
    else
        file ← err_fputs(file, SAMs);
    end if
end while
                                  ▷ Close the output SAM file(s)
if chromosomes ← output then
    for CHRM ← 1 to chromosomes do
        files[CHRM] ← fclose(CHRM);
    end for
else
    file ← fclose(outputfile);
end if
MPI_Finalize()                              ▷ MPI finalization
return 0;
```

Algorithm 3: MPI RMA Atomic Operations for Creating a Chromosome Queue for Pre-Procssing and Variant Calling.

```
Input: NumChromosomes – Number of chromosomes
info ← MPI_Info_create();
MPI_Info_set(info,"osc_rdma_acc_single_intrinsic","true")
win ← MPI_Win_allocate(sizeof(int), 1, info)
X ← 0
one ← 1
while true do
    X ← MPI_Fetch_and_op(&one, MPI_SUM, win)
                                        ▷ Next Chromosome
    if X < NumChromosomes then
        process_chromosome(X)           ▷ Process Chromosome (sort, index, duplicates, variant call)
    else
        break ▷ All chromosomes have been assigned, stop.
    end if
end while
MPI_Win_free(win)                       ▷ Free window resources
```

duplicate algorithm from Sambamba because both perform better and have multi-threading capabilities. For variant calling, we use three different tools (GATK HaplotypeCaller, DeepVariant and Octopus) for germline variant calling of short reads. As discussed above, generating SAM files for individual chromosomes in the alignment process is not necessary, since we can also use a single SAM file. If the user wants to use all other pre-processing (sorting, mark duplicate, BAM indexing) and variant calling tools in this workflow, it is also possible to parallelize these by providing the contigs/chromosome numbers, even in the case of using a single SAM file. Finally, the individual VCFs created are merged through Bcftools to produce a final complete VCF file for further downstream analysis.

B. Long-Reads Variant Calling Workflow

In the following subsections, we describe in detail how MPI has been integrated with the long read aligner Minimap2, and how different pre-processing applications are used in the variant calling workflow.

1) Alignment: Circular consensus sequencing (CCS) based PacBio HiFi reads and Oxford Nanopore technologies (ONT) long reads are making their way from research to clinical applications. They provide more in-depth and better consensus particularly in more complex repetitive regions of the genome [37]. The Minimap2 long reads aligner (with or without some additional parameter settings) is mainly being used for both PacBio

and ONT long reads data. The complete algorithmic implementation details of this integration are given in pseudo-code in Algorithms 1 and 2, the detail description of Minimap2 implementation is similar to BWA-MEM as described in Section II-A1.

2) Sorting, Indexing and Variant Calling: For long reads after alignment, only sorting and index generation for BAM is necessary. MPI atomic operations based queue algorithm continuously polls the input chromosomes SAM files, or otherwise contigs/chromosomes number can also be used in case a single SAM file is generated through Minimap2. Then the queue algorithm operates on those files/contigs/chromosomes for sorting, BAM index generation and variant calling algorithms. We used and integrated the sorting algorithm from Samtools. For variant calling, we used three algorithms for germline variant calling of long reads (DeepVariant, DeepVariant with WhatsHap for phasing on PacBio dataset, and Clair3 for ONT dataset). These variant callers are commonly recommended by the corresponding sequencing vendors. Finally, the individual VCFs created are merged using Bcftools to produce a final complete VCF file for further downstream analysis.

III. Results

A. Evaluation Cluster

We used both HLRS Hawk [38] (an HPE Apollo 9000 system at the High Performance Computing Center Stuttgart (HLRS) in Germany) and the SurfSara Snellius [39] (part of the Dutch national supercomputing infrastructure) HPC clusters. Each compute node of Hawk is equipped with a dual socket AMD EPYC 7742 processors (64 cores/socket) running at 2.25 GHz. All nodes are connected through Mellanox HDR200 (interconnect bandwidth 200 Gbit/s) Infiniband adapter. Likewise, on SurfSara Snellius, each compute node is equipped with a dual socket AMD EPYC 7H12 (64 cores/socket) processors running at 2.6 GHz. All nodes are connected through Mellanox HDR100 (interconnect bandwidth 100 Gbit/s) Infiniband adapter. A local storage of 1-TBytes and the same amount of network attached storage is available on both systems.

The parallel file system available on HLRS Hawk is based on Lustre while SurfSara Snellius is equipped with IBM Spectrum Scale (GPFS) [18]. The SLURM Workload Manager is installed on SurfSara Snellius while HLRS Hawk uses PBSPro workload manager and job scheduler.

Additional, we had tried to use NVIDIA Clara Parabricks, but due to some limitations we avoided including it this manuscript, 1) it's not an open source solution 2) NVIDIA Clara Parabricks only provides limited commercial usage to run with limited datasets 3) it requires some specific GPU hardware like higher than A100 which is not available in our cluster. 4) There are no publicly available accuracy numbers published for the NVIDIA Clara solution that we can compare with.

Only DeepVariant based variant calling workflows are ran on SurfSara Snellius GPU nodes.

In the alignment MPI implementation, both BWA and Minimap2 already use a load balancing approach. Seqkit divides the input FASTQ files at runtime equal to the number of nodes we input. For variant calling, we have tested load balancing approach as mentioned in Fig. 11: (Performance and scalability comparisons when using MPI shared I/O and POSIX I/O for (a) single SAM output (b) chromosomes based SAM output and (c) chromosomes regions specific SAM output.) Where we showed that region based approach is scaling up the runtime but at the same time due to IOs it's not that significant to be adopted as we preferred the same accuracy as on a single node BAM pileup for SNPs calling over a minor speedup improvement. But users can select any approach. The accuracy goes a bit low due to BAMs chunks being processed for pileup and it misses some SNPs positions. This is further discussed in Section IV

B. Datasets

We used the Illumina, PacBio HiFi, and ONT HG002 datasets taken from the PrecisionFDA challenge V2 [40]) dataset for variant calling workflows. We also used 300x sequencing coverage WGS data from Genome in a Bottle (GIAB) aligned with novoalign for the Illumina HiSeq 300x reads for NA12878 [41] to analyze the scalability of BWA-MEM aligners. Human genome reference GRCh38 [42] is used as a reference genome. For accuracy comparisons, the GIAB v4.2.1 [28] benchmark set for HG002 dataset is used.

C. Runtime Performance

In this section, we analyze and compare the runtime performance of MPI based aligners (BWA-MEM and Minimap2) with existing state-of-the-art MPI and Apache Spark based implementations. We also benchmark the performance of variant calling workflows using different variant callers on a cluster of up to 8 nodes.

1) Short Reads Alignment: We compare the scalability and runtime performance of GenMPI with those based on Apache Spark (ADAM's Cannoli [10] alignment) and another MPI-based implementation, QUARTIC [43] (mpiBWA). We use different numbers of nodes; 2, 4, 8 and 16 nodes for runtime performance comparisons. ADAM's Cannoli uses the built-in Scala API from the Apache Spark backend for distributing and scheduling data for parallel processing. The Cannoli wrapper encompasses many different aligners and variant callers for distributed processing. Apache Spark based implementations use HDFS or NFS for I/O operations. QUARTIC (mpiBWA) is a distributed BWA-MEM alignment algorithm employing MPI functionality and uses MPI shared I/O for input/output on parallel file system. In Fig. 2 (left bar), we show the BWA-MEM runtime on a single node utilizing all 128 system CPU cores with one thread for each core. We consider this time as an ideal runtime for a single node. When we compare this runtime by increasing the number of nodes in a cluster as shown in Fig. 2, we see QUARTIC and GenMPI perform better than this ideal runtime, while ADAM Cannoli BWA-MEM implementation has less than ideal scalability. We attribute the super-linear scaling of both QUARTIC and GenMPI to scalability issues of BWA-MEM when utilizing all cores on a single node using a single process (128 threads). As a consequence, we use two MPI processes on each node to overcome this inefficiency for all

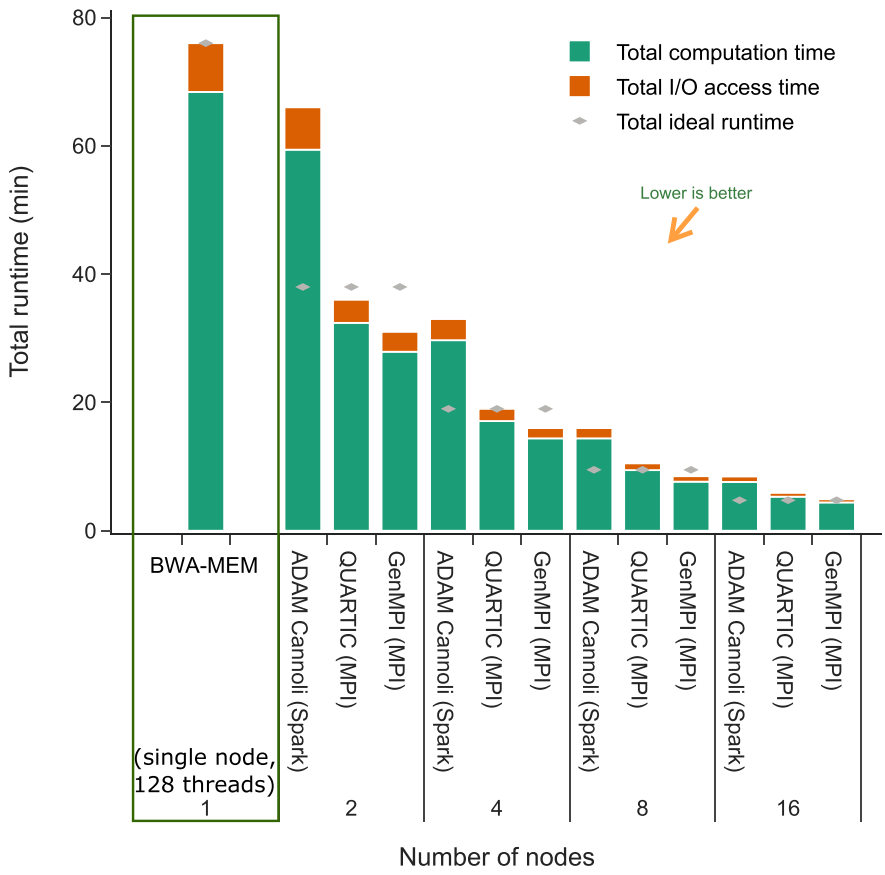

Fig. 2. Run-time and scalability comparisons of ADAM's Cannoli, QUARTIC mpiBWA and this work implementations of BWA-MEM aligner on a cluster of varying number of nodes using HG002 (NA24385) Illumina NovaSeq 35x coverage dataset. Ideal runtime is the runtime assuming ideal scalability on the available nodes.

other runs. Overall GenMPI outperforms QUARTIC by almost 20% in terms of runtime by reducing compute and read/write I/O time, and outperforms Apache Spark based ADAM Cannoli BWA-MEM by 2x in terms of runtime.

2) Short-Reads Variant Calling Workflow: As discussed in the alignment section, we store BWA-MEM SAM output in chromosomes files. Sorting can be performed using existing tools on these chromosome files in parallel on the cluster. We use *Samtools'* sorting algorithm since it is one of the most efficient due to its in-memory and multi-threading functionality. Similarly, for duplicate removal we use *Picard*'s MarkDuplicate compatible algorithm *Sambamba*, which is also multi-threaded. We also use GATK best practices pipeline applications like *Base quality score recalibration* (BQSR), *ApplyBQSR*, and *HaplotypeCaller* afterwards. Moreover, we integrate *DeepVariant* and *Octopus*, both recent and accurate variant callers, in this workflow. Their published results show high accuracy and F_1-score compared to other state-of-the-art variant callers like *GATK4 HaplotypeCaller* [21], *Strelka2* [44], or *FreeBayes* [45].

We compare the total runtime of these workflows on varying numbers of nodes. As shown in Fig. 3, moving from a single node to two nodes a more than 3x runtime speedup is achieved because we allocated multiple MPI processes for pre-processing and variant calling applications on each node as these tools have some single node (128 CPU cores) scalability limitations. Increasing the cluster size from 4 to 8 nodes yields only 70–80% runtime improvements because of poor chromosome load-balance. The same runtime speedup is observed for DeepVariant instead of Octopus. On the other hand, GATK best practices pipeline applications are either slow or single-threaded; therefore we only focus on optimizing the workflow by insuring accuracy while running with maximum efficiency on a single node. As shown in Fig. 4, using alternative application like Samtools/Sambamba instead of Picard (sort and markdup) respectively, and parallel execution of GATK BQSR, ApplyBQSR and HaplotypeCaller applications for chromosomes we have achieved more than 6.5x runtime speedup compared to the baseline. Since GATK processes chromosomes using a single thread, distributing it's execution across multiple nodes is not worthwhile due to the limited number of chromosomes. We have only 25 chromosomes which can be ran in parallel on a single node more efficiently.

3) Long Reads Alignment: GenMPI is the first-ever cluster scale implementation of any long reads aligners. As discussed in Section II, both chromosomes based SAM files and a single SAM file output options are implemented in Minimap2. Fig. 5

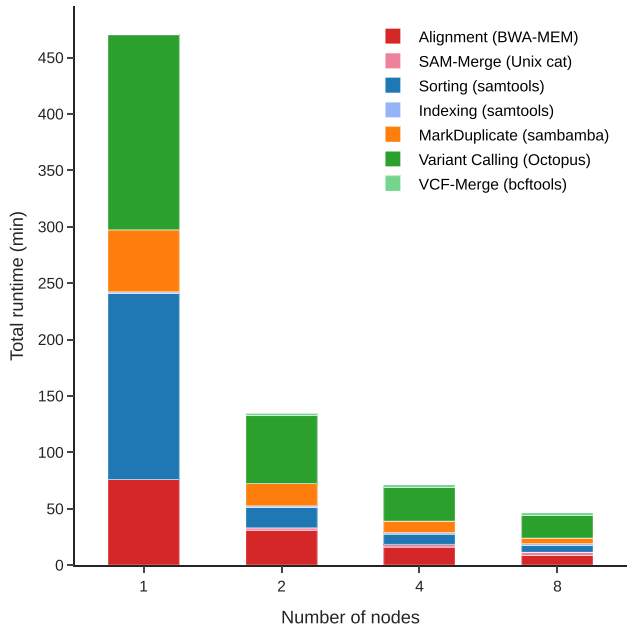

Fig. 3. Run-time and scalability benchmark of GenMPI for BWA-MEM aligner and Octopus variant caller on a cluster of varying number of nodes using HG002 (NA24385) Illumina NovaSeq 35x coverage dataset.

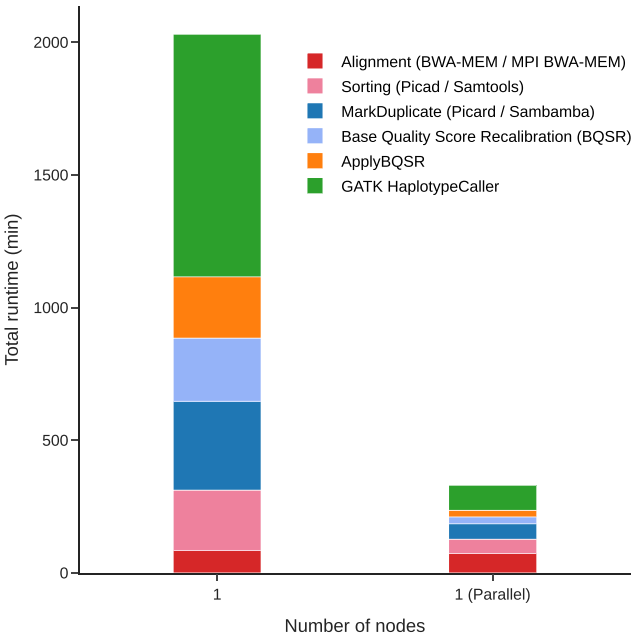

Fig. 4. Run-time and scalability benchmark of GenMPI for BWA-MEM aligner and GATK best practices pipeline on a single node versus alternative parallel single node methods using HG002 (NA24385) Illumina NovaSeq 35x coverage dataset.

shows the total runtime for long-reads aligner, Miniamap2, which exhibits close to linear scalability when increasing the number of nodes in the cluster. The ideal theoretical values for linear scalability are also shown. As shown in the figure, there is a minor performance degradation compared to ideal runtimes that is caused by read/write I/O operations for long reads. Due to the efficient performance of network-attached parallel file system, the overhead of storing SAM data in multiple chromosomes file is minimal compared to a single SAM file generation from all the MPI processes.

4) Long Reads Variant Calling Workflow: Almost all new long reads variant callers only require aligned and sorted reads. Long reads sorting is performed through Samtools 'sort' algorithm. As mentioned before, for PacBio data, we used DeepVariant and DeepVariant with WhatsHap [24]. WhatsHap is used to reconstruct the chromosomes haplotypes and then write out the input VCF augmented with phasing information in its first pass, in the second pass WhatsHap haplotag writes information of reads along with the variants. This information can be used again in DeepVariant for better INDEL detection and accuracy purposes We have represented this additional phasing based approach as DV-WH-DV in rest of this paper. In Fig. 6, we show the scalability results of long reads variant calling workflow (Minimap2, Samtools sorting and DeepVariant) on a cluster where more than 2.25x runtime speedup is achieved from 1 node to 2 node cluster and a total of 5x speedup is achieved on a 8 node cluster. This scalability is again only valid up to 8 nodes since we aim at reproducing the exact same variant output as that of a single node, which means that we do not sub-divide chromosomes into smaller pieces. Similarly, for ONT dataset we used Clair3 which provides better accuracy and performance for both SNP and INDEL variants as compared to other variant callers on ONT data. As shown in Fig. 7 due to some internal scalability limitations of Clair3 we are only able to achieve 1.5x runtime speedup for two nodes cluster as compared to a single node runtime and similar limited scalability trend is shown for more nodes.

D. Performance Evaluation

In this subsection, we present small variants (SNP and INDEL) detection accuracy. The GA4GH small variant benchmarking tool hap.py has been used to compare the variants in all methods. We compared all these statistics on Chr1-22 and X, Y for each dataset. We compared the small variants (SNP and INDEL) detection accuracy for both short and long reads on HG002 (PrecisionFDA challenge V2) dataset against GIAB v4.2.1 benchmark set. We only tested the accuracy metrics for all-benchmarking region of GIAB v4.2.1 benchmark.

1) SNP precision/recall: Fig. 8 shows the accuracy performance of SNPs in terms of precision and recall for short reads methods (GATK HaplotypeCaller, Octopus and DeepVariant) on Illumina dataset and long reads methods (DeepVariant, DeepVariant with WhatsHap and Clair3) on PacBio and ONT datasets respectively. Both DeepVariant and DeepVariant with WhatsHap on PacBio data perform best as compared to other methods while GATK HaplotypeCaller SNP performance is well below all other methods. Overall, the best SNP F_1 score is 0.999283 for PacBio dataset using DeepVariant variant caller, as shown in Table II, while Table III shows the results for ONT dataset.

2) INDELs precision/recall: Similarly in Fig. 9 the accuracy performance in terms of precision and recall of INDELs for short

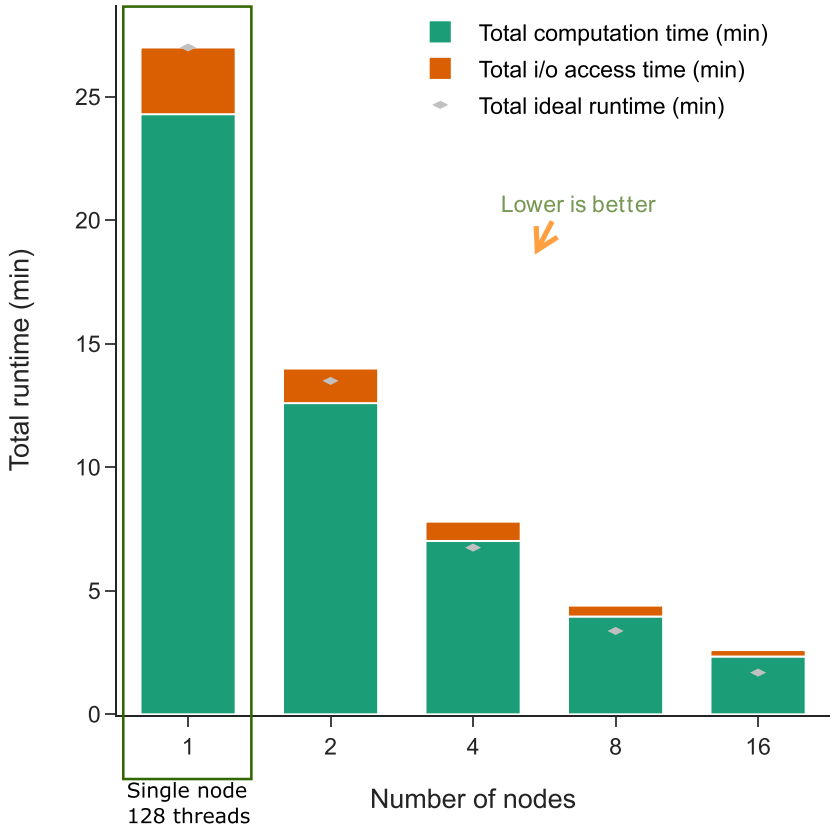

Fig. 5. Run-time and scalability performance of MPI based Minimap2 implementation. HG002(NA24385) PacBio HiFi 35x coverage dataset has been used on a cluster of varying number of nodes.

TABLE I
ACCURACY EVALUATION OF SMALL VARIANTS FOR HG002 (PAIR-END ILLUMINA SHORT READS) AGAINST GIAB HG002 V4.2 BENCHMARKING SET FOR DIFFERENT METHODS ADOPTED IN THIS WORKFLOW

Method	Variant type	Truth total	TP	FN	FP	Recall	Precision	F_1-Score
DeepVariant	INDEL	525469	522169	944037	1238	0.993720	0.997731	0.995721
-	SNP	3365127	3345702	3815162	4125	0.994228	0.998769	**0.996493**
GATK	INDEL	525469	510602	913193	3671	0.971707	0.993147	0.982310
-	SNP	3365127	3261925	3800121	21703	0.969332	0.993393	0.981215
Octopus	INDEL	525469	522366	935701	1195	0.994095	0.997825	**0.995957**
-	SNP	3365127	3339460	3719754	4455	0.992373	0.998662	0.995507

This table shows the SNP and INDEL results for all benchmarking regions.

TABLE II
ACCURACY EVALUATION OF SMALL VARIANTS FOR HG002 (PACBIO) AGAINST GIAB HG002 V4.2 BENCHMARKING SET FOR DIFFERENT METHODS ADOPTED IN THIS WORKFLOW

Method	Variant type	Truth total	TP	FN	FP	Recall	Precision	F_1-Score
DeepVariant	INDEL	525469	518336	970119	7560	0.986425	0.986187	0.986306
-	SNP	3365127	3362077	4054200	1778	0.999094	0.999472	**0.999283**
DV-WH-DV	INDEL	525469	521828	981392	3844	0.993071	0.992977	**0.993024**
-	SNP	3365127	3362239	4060995	2199	0.999142	0.999347	0.999244
Clair3	INDEL	525469	351200	554604	26874	0.668355	0.930726	**0.778016**
-	SNP	3365127	3355502	4148864	10306	0.997140	0.996939	0.997039

This table shows the SNP and INDEL results for all benchmarking regions.

reads methods (GATK HaplotypeCaller, Octopus and DeepVariant) on Illumina dataset and long reads methods (DeepVariant, DeepVariant with WhatsHap and Clair3) on PacBio and ONT datasets respectively has been shown. We have observed for INDEL performance that Illumina short read dataset seems to performs best as compared to other datasets for both DeepVariant and Octopus methods. ONT dataset INDEL performance is lower than any other datasets. Overall the best INDEL F_1 score is 0.995957 for Illumina dataset using Octopus variant caller as shown in Table I.

TABLE III
ACCURACY EVALUATION OF SMALL VARIANTS FOR HG002 (ONT) AGAINST GIAB HG002 V4.2 BENCHMARKING SET FOR DIFFERENT METHODS ADOPTED IN THIS WORKFLOW

Method	Variant type	Truth total	TP	FN	FP	Recall	Precision	F_1-Score
DeepVariant	INDEL	526461	518453	970445	7556	0.985675	0.986567	0.996500
-	SNP	3364534	3362045	4054136	1722	0.996457	0.993443	0.998232
DV-WH-DV	INDEL	525346	522379	98125	3856	0.994348	0.992400	0.989265
-	SNP	3356727	3362532	4060445	2121	0.999223	0.995563	0.999567
Clair3	INDEL	525455	354512	554345	26245	0.668453	0.967826	0.768456
-	SNP	3368124	3356560	414845	104646	0.997140	0.956939	**0.998039**

This table shows the SNP and INDEL results for all benchmarking regions.

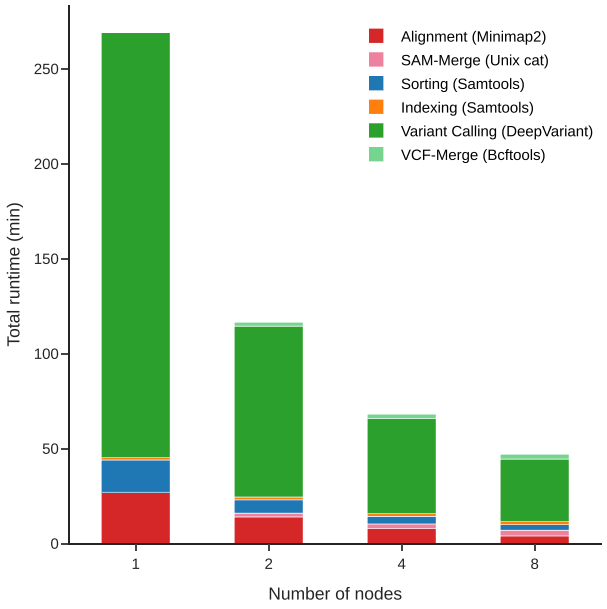

Fig. 6. Run-time and scalability benchmark of this MPI based implementation for Minimap2 aligner and DeepVariant variant caller on a cluster of varying number of nodes using HG002 (NA24385) PacBio HiFi 35x coverage dataset.

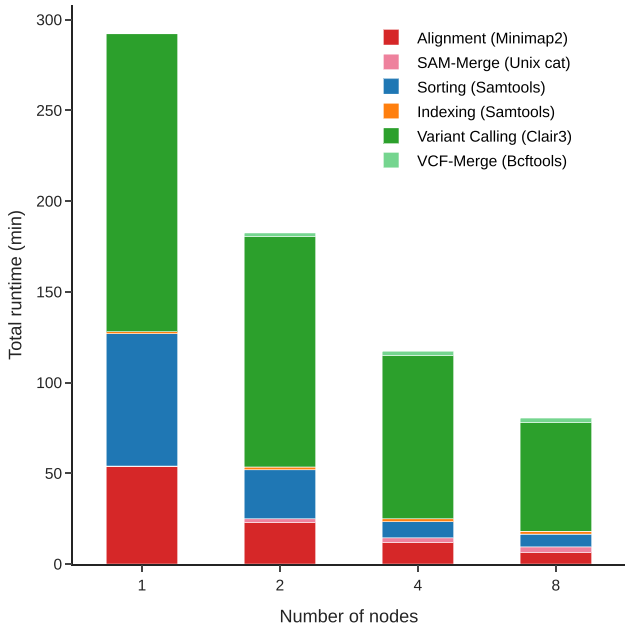

Fig. 7. Run-time and scalability benchmark of this MPI based implementation for Minimap2 aligner and Clair3 variant caller on a cluster of varying number of nodes using HG002 (NA24385) ONT Guppy 3.6.0 dataset.

IV. DISCUSSION

This section further elaborates on the benefits of this approach in a broader context of its applicability in real-time usage on HPC clusters.

A. Runtimes

Total runtimes of the used aligners show a good linear scalability on small to big clusters in both single SAM output as well as for chromosomes based SAM output. Similarly, for complete variant calling workflows almost all methods provide more than 2x to 6x lower runtime when executed on 2 to 8 cluster nodes, respectively. Because GATK best practices pipeline applications are not multi-threaded, cluster scalability is not possible. Nevertheless, we have achieved more than 6.5x speedup when executing in parallel on a single node. Similarly, Clair3 has multi-threading limitations that preclude proper scaling across multiple nodes of a cluster.

B. Accuracy and Reproducibility

Accuracy results for the Illumina short-read dataset using GATK HaplotypeCaller, Octopus, and DeepVariant variant

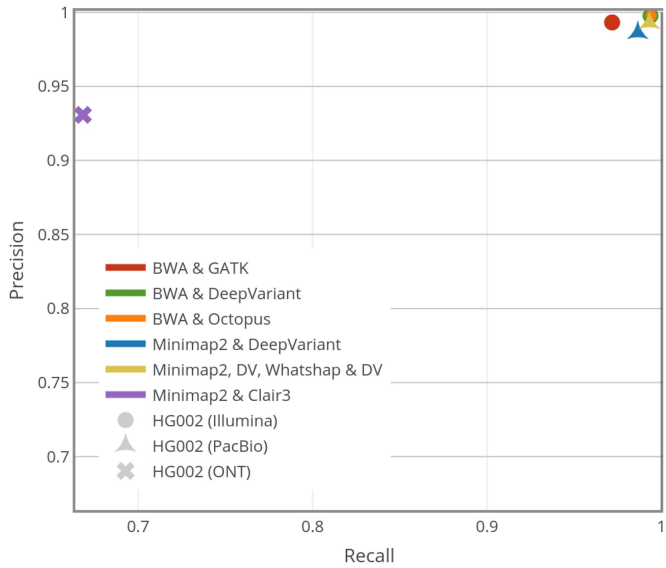

Fig. 8. SNPs performance comparison of different short and long reads variant calling methods on HG002 dataset.

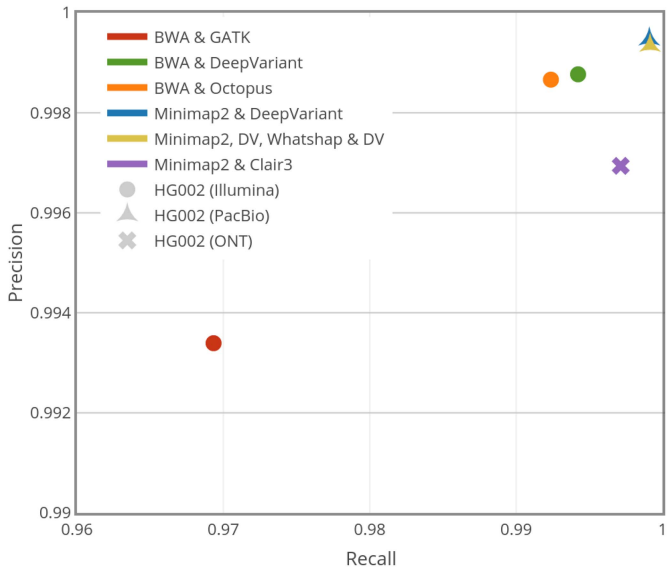

Fig. 9. INDELs performance comparison of different short and long reads variant calling methods on HG002 dataset.

callers are shown in Table I. Similarly, Table II shows accuracy results for the PacBio dataset using the DeepVariant and DeepVariant with WhatsHap variant callers. Table III lists the accuracy results for the ONT dataset using the Clair3 variant caller. These results are obtained through GA4GH small variant benchmarking tool hap.py. The accuracy comparisons between the MPI versions of the pipelines and their single node baseline show that these workflows produce the exact same recall, precision, and F_1-score for both short and long reads based variant calling workflows. This ensures the reproducibility of original BWA-MEM and Minimap2 functionality when using GenMPI.

C. Scalability

The MPI parallelization of both aligners (BWA-MEM and Minimap2) is highly scalable. To further evaluate the scalability of both aligners, we tested the GIAB 300x coverage whole genome sample NA12878 for HG001 with the GenMPI using BWA-MEM. Fig. 11 shows scalability results of aligning almost 2.5 TBytes data in just 10 minutes walltime on a 64 nodes. We show both computation and I/O time for both aligners separately. In both aligners, the resultant graphs show that increasing the number of nodes in the cluster, the runtime of alignments decreases almost linearly. We also observed that I/O time is increasing slightly when increasing the number of nodes but it is not a bottleneck for the scalability.

The only issue is in shared MPI I/O, particularly in writing and reading part of Minimap2 for long reads is a limiting factor for linear scalability. Both MPI I/O and POSIX file system options to output a single SAM file, chromosomes-based SAM output and chromosomes-regions specific SAM output have been implemented as shown in Fig. 10. We have observed significant overhead due to synchronization effects in writing to POSIX I/O as well as MPI I/O when a single output SAM file is written, as shown in Fig. 10(a). For per-chromosome SAM output (Fig. 10(b)), the best suitable option is using POSIX I/O. For the chromosomes-regions specific SAM output option (Fig. 10(c)), we used 128 output files, which provides comparable results to per-chromosome SAM output with POSIX I/O but still affects the overall performance compared to ideal runtime. The reason behind this is inner reads identification and writing loops which direct reads to a specific file. We have tested both MPI I/O blocking and non-blocking I/O operations for writing SAM output. Due to processes synchronization for wiring SAM data chunks to I/O, an extra overhead slows down the writing of results. This overhead can be mitigated by directly integrating aligners with sorting and indexing applications without relying on file I/O. This will be part of our future work (Section IV-G).

D. Portability and Deployment

By using the MPI standard, this workflow is portable and easily deployable to any HPC cluster. A detailed description and quick start guide to run all methods in this approach are given on the project github page. We also tested this implementation with OpenMPI, Intel MPI and HPE MPI flavors and it compiles/runs appropriately.

E. Cost Efficiency

Our cost estimations predict that a significant amount can be saved when opting for public clouds clusters instead of a single large node in the cloud. Particularly in BWA-MEM alignment, clusters utilize maximum system resources resulting in cost saving of more than 50% . Similarly, DeepVariant and Octopus variant callers have some limitation in single node performance for a higher number of cores per socket. More than 20-30% cost can be saved using these variant callers on a cluster with each node having two sockets each with 64 CPU cores.

F. Memory Consumption

MPI based scalable implementations can have a large edge over Apache Spark based variant calling workflows in terms of memory consumption. The MPI implementations do not need extra memory for the platform itself nor to store the data during data shuffling in Spark based implementations.

G. Future Work

This whole workflow implementation (GenMPI) uses storage for intermediate applications I/O read and write operations in form of SAM/BAM files. For future work, integrating preprocessing applications like GenMPI BWA-MEM with sorting (mpiSORT [43]) and index generation would yield further performance gains. We have observed that 5-10% of the total time in GenMPI BWA-MEM and 50% of the total time in mpiSORT are spent on file I/O, which could be mitigated through this approach.

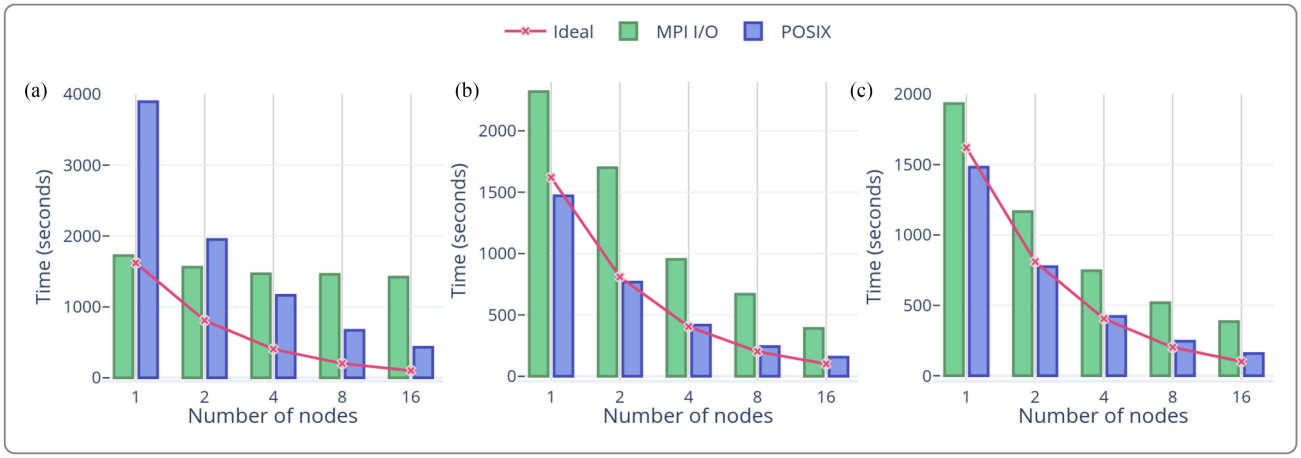

Fig. 10. Performance and scalability comparisons when using MPI shared I/O and POSIX I/O for (a) single SAM output (b) chromosomes based SAM output and (c) chromosomes regions specific SAM output.

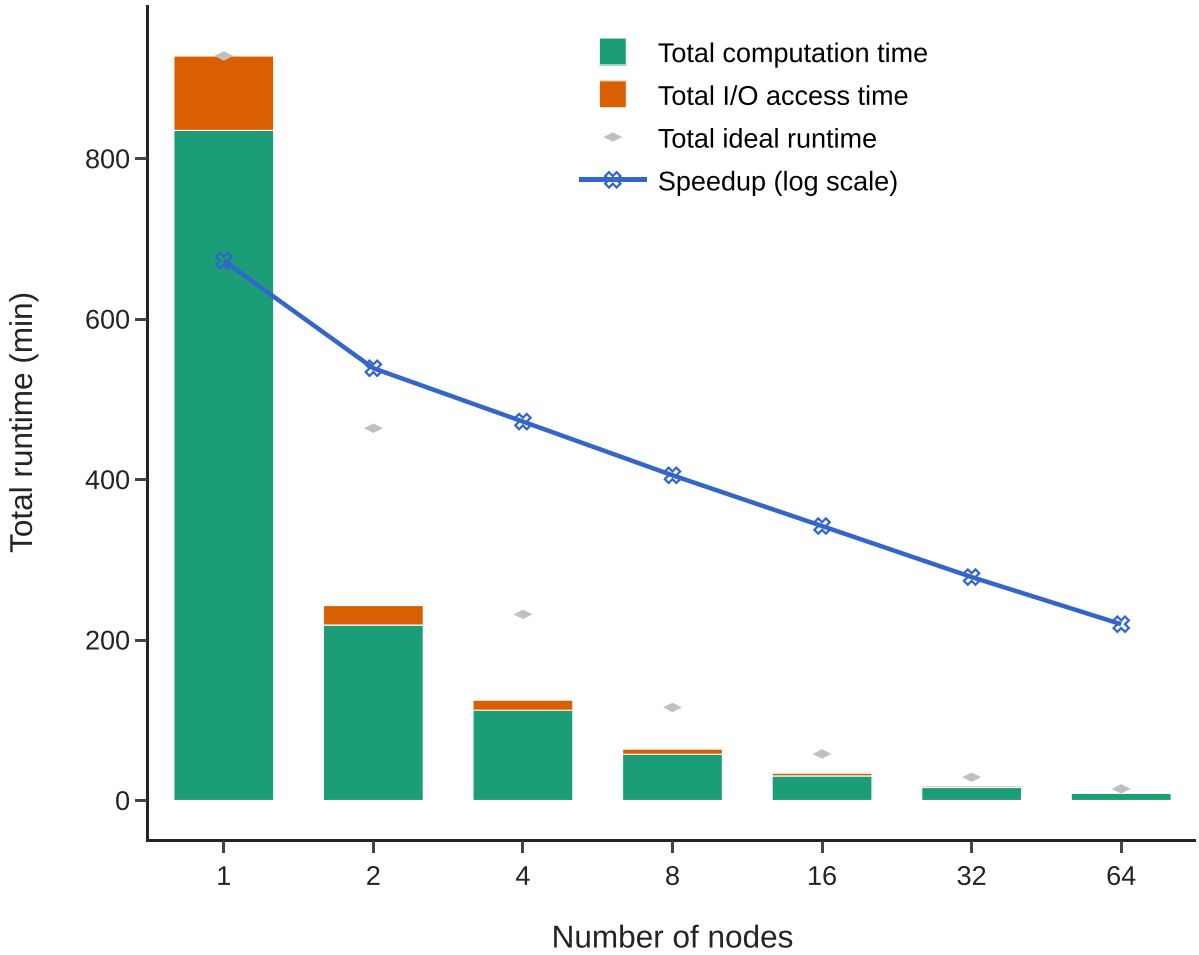

Fig. 11. GenMPI BWA-MEM performance and scalability measurements for 300x sequencing coverage WGS data from Genome in a Bottle (GIAB) aligned with novoalign for the Illumina HiSeq 300x reads for NA12878 [41].

V. Conclusion

In this work, we have implemented GenMPI, an MPI based scalable method for both widely used short and long reads aligners, BWA-MEM and Minimap2, respectively. One of the main goals of GenMPI is to ensure 100% identical variant output compared to the single node baseline. In addition, GenMPI provides a flexible architecture which can be used to integrate a variety of alignment and variant calling tools. Compressed and uncompressed FASTQ input is supported and output can be a

single SAM file or chromosomes-based SAM files generated by MPI processes. This output can be stored on network attached storage through both POSIX and MPI shared I/O, whichever is convenient and efficient for the underlying HPC system. Results show that these implementations outperform existing Apache Spark based implementation of alignment algorithms by 2x and yield a 20% speedup over state-of-the-art MPI implementations of the BWA-MEM algorithm. Likewise, we also integrate pre-processing applications (sorting, indexing, duplicates removal (for short reads)) as well as variant callers like GATK HaplotypeCaller, DeepVariant, Octopus and Clair3 for both short (Illumina) and long reads (PacBio and ONT) datasets. The variant calling workflows are scalable for up to 8 nodes cluster while giving 2x to 6x total runtime speedups.

Particularly, we show that the distribution of chromosomes across aligners is almost linearly scalable, even when tested using 300x coverage datasets on up to 64 nodes cluster for BWA-MEM. At scale, the alignment completes in under 10 minutes walltime. Thanks to the use of MPI, the workflow implementation is portable and easily deployable on any public cloud or private HPC cluster with minimal effort. Memory requirements do not exceed the actual software needs. The final accuracy results (Recall, Precision, and F_1 score) for variant calling have been reproduced with hap.py against latest GIAB benchmarks set and are shown to be identical to those of the baseline pipeline.

V. Author Contributions

Z.A.A, P.H., J.G. and C.N. conceived and supervised this work. T.A. and J.S. implemented and developed MPI-based whole variant calling workflow including MPI integration into aligners. All authors read and approved the final manuscript.

V. Funding

Tanveer Ahmad is recipient of PEEF, Pakistan PhD sponsorship. The research conducted in this work is co-supported by HPC-Europa3 which receives funding from the European Union's Horizon 2020 research and innovation programm under grant agreement No.730897. This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

V. Data Availability Statement

All the codes and scripts are available at: https://github.com/abs-tudelft/gen-mpi. Illumina, PacBio HIFI and ONT HG002 datasets taken from PrecisionFDA challenge V2 [40]). We also used 300x sequencing coverage WGS data from Genome in a Bottle (GIAB) aligned with novoalign for the Illumina HiSeq 300x reads for NA12878 [41]. Human genome reference GRCh38 [42] is used as a reference genome. For accuracy comparisons, GIAB v4.2.1 [28] benchmark set for HG002 dataset is used.

Acknowledgment

We are grateful for the resources provided by the High Performance Computing Center Stuttgart (HLRS). Special thanks to Alexey Cheptsov (HLRS) for all the HPC related support. This work was also carried out in part on the Dutch national e-infrastructure with the support of SURF Cooperative.

References

[1] J. M. Zook et al., "Extensive sequencing of seven human genomes to characterize benchmark reference materials," *Sci. Data*, vol. 3, no. 1, Jun. 2016, Art. no. 160025, doi: 10.1038/sdata.2016.25.

[2] C. N. Kline et al., "Targeted next-generation sequencing of pediatric neuro-oncology patients improves diagnosis, identifies pathogenic germline mutations, and directs targeted therapy," *Neuro- Oncol.*, vol. 19, no. 5, pp. 699–709, Nov., 2016, doi: 10.1093/neuonc/now254.

[3] N. Becher et al., "Implementation of exome sequencing in fetal diagnostics—Data and experiences from a tertiary center in Denmark," *Acta Obstetricia et Gynecologica Scandinavica*, vol. 99, no. 6, pp. 783–790, 2020. [Online]. Available: https://obgyn.onlinelibrary.wiley.com/doi/abs/10.1111/aogs.13871

[4] T. S. Roman et al., "Genomic sequencing for newborn screening: Results of the nc nexus project," *Amer. J. Hum. Genet.*, vol. 107, no. 4, pp. 596–611, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000292972030269X

[5] M. Allegretti et al., "Tearing down the walls: Fda approves next generation sequencing (NGS) assays for actionable cancer genomic aberrations," *J. Exp. Clin. Cancer Res.*, vol. 37, no. 1, Mar. 2018, Art. no. 47, doi: 10.1186/s13046-018-0702-x.

[6] S. Deorowicz and S. Grabowski, "Compression of DNA sequence reads in FASTQ format," *Bioinformatics*, vol. 27, no. 6, pp. 860–862, Jan., 2011, doi: 10.1093/bioinformatics/btr014.

[7] Apache, "Apache spark: Lightning-fast unified analytics engine [accessed: 2nd april 2019]," 2019. [Online]. Available: https://spark.apache.org/

[8] Apache, "Apache hadoop [accessed: 2nd april 2019]," 2019. [Online]. Available: https://hadoop.apache.org/

[9] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1327452.1327492

[10] M. Massie et al., "ADAM: Genomics formats and processing patterns for cloud scale computing," UCB/EECS-2013-207, EECS Department, University of California, Berkeley, Tech. Rep., 2013.

[11] H. Mushtaq, N. Ahmed, and Z. Al-Ars, "SparkGA2: Production-quality memory-efficient apache spark based genome analysis framework," *PLOS ONE*, vol. 14, no. 12, pp. 1–14, Dec., 2019, doi: 10.1371/journal.pone.0224784.

[12] T. Ahmad, Z. Al Ars, and H. P. Hofstee, "VC, scale: Scalable and high-performance variant calling on cluster environments," *GigaScience*, vol. 10, no. 9, 2021, Art. no. giab057, doi: 10.1093/gigascience/giab057.

[13] D. Decap, J. Reumers, C. Herzeel, P. Costanza, and J. Fostier, "Halvade: Scalable sequence analysis with MapReduce," *Bioinformatics*, vol. 31, no. 15, pp. 2482–2488, 2015, doi: 10.1093/bioinformatics/btv179.

[14] T. Ahmad, N. Ahmed, J. Peltenburg, and Z. Al-Ars, "Arrowsam: In-memory genomics data processing using apache arrow," in *Proc. 3 rd Int. Conf. Comput. Appl. Inf. Secur.*, 2020, pp. 1–6.

[15] T. Ahmad, N. Ahmed, Z. Al-Ars, and H. P. Hofstee, "Optimizing performance of gatk workflows using apache arrow in-memory data framework," *BMC Genomic.*, vol. 21, no. 10, Nov. 2020, Art. no. 683, doi: 10.1186/s12864-020-07013-y.

[16] L. X., K. Qiu, P. Liang, and D. Peters, "Speeding up large-scale next generation sequencing data analysis with pbwa," *J. biocomputing*, vol. 1, 2012, Art. no. 2.

[17] Lustre, "Lustre parallel filesystem," 2020. [Online]. Available: https://www.lustre.org/

[18] IBM, "IBM spectrum scale," 2020. [Online]. Available: https://www.ibm.com/products/spectrum-scale

[19] H. Li and R. Durbin, "Fast and accurate short read alignment with Burrows–Wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009, doi: 10.1093/bioinformatics/btp324.

[20] H. Li, "Minimap and miniasm: Fast mapping and de novo assembly for noisy long sequences," *Bioinformatics*, vol. 32, no. 14, pp. 2103–2110, Jul. 2016, [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/27153593

[21] B. Institute, "Genome analysis toolkit [accessed: 11 april 2019]," 2010. [Online]. Available: https://software.broadinstitute.org/gatk/
[22] D. P. Cooke, D. C. Wedge, and G. Lunter, "A unified haplotype-based method for accurate and comprehensive variant calling," *Nature Biotechnol.*, vol. 39, no. 7, pp. 885–892, Jul. 2021, doi: 10.1038/s41587-021-00861-3.
[23] R. Poplin et al., "A universal snp and small-indel variant caller using deep neural networks," *Nature Biotechnol.*, vol. 36, 2018, Art. no. 983, doi: 10.1038/nbt.4235.
[24] M. Martin et al., "Whatshap: Fast and accurate read-based phasing," *bioRxiv*, 2016. [Online]. Available: https://www.biorxiv.org/content/10.1101/085050v2
[25] Z. Zheng et al., "Symphonizing pileup and full-alignment for deep learning-based long-read variant calling," *bioRxiv*, 2021. [Online]. Available: https://www.nature.com/articles/s43588-022-00387-x
[26] P. Danecek et al., "Twelve years of SAMtools and BCFtools," *GigaScience*, vol. 10, no. 2, 2021, Art. no. giab008, doi: 10.1093/gigascience/giab008.
[27] P. Krusche, "Haplotype VCF comparison tools," 2021. [Online]. Available: https://github.com/Illumina/hap.py
[28] GIAB, "Giab v4.2.1 small variant benchmark for hg002-hg004," 2021. [Online]. Available: https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/AshkenazimTrio/
[29] H. Li, "The sequence alignment/map format and samtools," *Bioinformatics*, vol. 25, pp. 2078–2079, Jan., 2009.
[30] A. Tarasov, A. J. Vilella, E. Cuppen, I. J. Nijman, and P. Prins, "Sambamba: Fast processing of NGS alignment formats," *Bioinformatics*, vol. 31, no. 12, pp. 2032–2034, Jun. 2015, [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/25697820
[31] medaka, "Medaka - sequence correction provided by ONT research," 2021. [Online]. Available: https://github.com/nanoporetech/medaka#usage
[32] Slurm, "Slurm workload manager," 2020. [Online]. Available: https://www.schedmd.com/
[33] OpenPBS, "Openpbs workload manager," 2020. [Online]. Available: https://www.openpbs.org/
[34] W. Shen, S. Le, Y. Li, and F. Hu, "SeqKit: A cross-platform and ultrafast toolkit for FASTA/Q file manipulation," *PLoS One*, vol. 11, no. 10, pp. 1–10, 2016, doi: 10.1371/journal.pone.0163962.
[35] T. Hoefler et al., "Remote memory access programming in MPI-3," *ACM Trans. Parallel Comput.*, vol. 2, no. 2, pp. 1–26, Jun. 2015, doi: 10.1145/2780584.
[36] J. Schuchart, A. Bouteiller, and G. Bosilca, "Using MPI-3 RMA for active messages," in *Proc. IEEE/ACM Workshop Exascale MPI*, 2019, pp. 47–56.
[37] A. M. Wenger et al., "Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome," *Nature Biotechnol.*, vol. 37, no. 10, pp. 1155–1162, Oct. 2019, doi: 10.1038/s41587-019-0217-9.
[38] Hawk, "Hpe apollo (hawk): Next-generation HPC system, HLRS," 2021. [Online]. Available: https://www.hlrs.de/systems/hpe-apollo-hawk/
[39] Snellius, "Snellius: The dutch supercomputer," 2021. [Online]. Available: https://userinfo.surfsara.nl/systems/snellius
[40] FDA, "Precisionfda truth challenge V2: Calling variants from short and long reads in difficult-to-map regions," 2019. [Online]. Available: https://precision.fda.gov/challenges/10
[41] GIAB, "Nhgri illumina 300x bam," 2020. [Online]. Available: ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/NA12878/NIST_NA12878_HG001_HiSeq_300x/NHGRI_Illumina300X_novoalign_bams/
[42] UCSC, "Grch38," 2019. [Online]. Available: ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/001/405/GCA_000001405.15_GRCh38/seqs_for_alignment_pipelines.ucsc_ids/GCA_000001405.15_GRCh38_no_alt_analysis_set.fna.gz
[43] N. Frédéric et al., "Quartic: Quick parallel algorithms for high-throughput sequencing data processing," *F1000Research*, vol. 9, Oct., 2020, Art. no. 240.
[44] S. Kim et al., "Strelka2: Fast and accurate calling of germline and somatic variants," *Nature Methods*, vol. 15, no. 8, pp. 591–594, 2018, doi: 10.1038/s41592-018-0051-x.
[45] E. Garrison and G. Marth, "Haplotype-based variant detection from short-read sequencing," 2012. [Online]. Available: https://arxiv.org/abs/1207.3907