Parsing Next Generation Sequencing Data in Parallel Environments for Downstream Genetic Variation Analysis

Mariana Vasquez
Bioinformatics Program,
The University of Texas at El Paso,
El Paso, TX 79968
mvasquez16@miners.utep.edu

Jonathon Mohl
Border Biomedical Research Center
and Computational Science Program,
The University of Texas at El Paso,
El Paso, TX 79968
jemohl@utep.edu

Ming-Ying Leung
Bioinformatics Program,
Border Biomedical Research Center,
Computational Science Program, and
Department of Mathematical Sciences,
The University of Texas at El Paso,
El Paso, TX 79968
mleung@utep.edu

ABSTRACT

With the recent advances in next generation sequencing technology, analysis of prevalent DNA sequence variants from patients with a particular disease has become an important tool for understanding the associations between the disease and genetic mutations. A publicly accessible bioinformatics pipeline, called OncoMiner (http://oncominer.utep.edu), was implemented in 2016 to help biomedical researchers analyze large genomic datasets from patients with cancer. However, the current version of OncoMiner can only accept input files with a highly specific format for sequence variant description. In order to handle data from a broader range of sequencing platforms, a data preprocessing tool is necessary. We have therefore implemented the OncoMiner Preprocessing (OP) program for parsing data files in the popular FastQ and BAM formats to generate an OncoMiner input file. OP involves using the open source Bowtie2 and SAMtools software, followed by a python script we developed for genetic sequence variant identification. To preprocess very large datasets efficiently, the OP program has been parallelized on two local computers and the Blue Waters system at the National Center for Supercomputing Applications using a multiprocessing approach. Although reasonable parallelization efficiency has been obtained on the local computers, the OP program's speedup on Blue Waters has been limited, possibly due to I/O issues and individual node memory constraints. Despite these, Blue Waters has provided the necessary resources to process 35 datasets from patients with acute myeloid leukemia and demonstrated significant correlation of OP runtimes with the BAM input size and chromosome diversity.

Keywords

Next generation sequencing, Genetic sequence variants, Cancer, OncoMiner pipeline, Data preprocessing, Acute myeloid leukemia, High performance computing, Blue Waters, Multiprocessing, Python scripts.

1 INTRODUCTION

Many serious diseases, such as cancer [1], cystic fibrosis [2] and multiple sclerosis [3], are often linked to genetic mutations in the human genome. Identification of mutations prevalent in individuals with a given illness helps establish associations between the mutations and the disease. This is exemplified in acute myeloid leukemia (AML) with the genes DNMT3A, ASKL1, TET2, IDH1 and IDH2 linked to early disease progression [1]. With the recent advances in next generation sequencing (NGS), genomic sequences of an increasing number of cases have been made available. These data have enabled scientists to perform detailed data analyses to look for associations between diseases and DNA mutations, called genetic sequence variants (GSVs).

OncoMiner [4] (http://oncominer.utep.edu) is a publicly accessible bioinformatics pipeline developed at the University of Texas at El Paso (UTEP) for analyzing large genomic datasets from patients with cancer. OncoMiner's functionalities include linking GSVs with published research literature, visualization of their chromosomal locations, and performing statistical comparisons of their occurrence frequencies among different groups of subjects. As OncoMiner was originally developed for analyzing the GSVs from NGS and GSV identification services provided by Otogenetics Corporation, the input files for the OncoMiner pipeline were restricted to a particular format with a set of specific terms describing the type and location of each GSV. In order to utilize OncoMiner on more general datasets coming from other NGS platforms (e.g., the Illumina NextSeq sequencer in the UTEP Genomic Analysis Core Facility), data preprocessing needs to be performed in order to provide an input file that can be passed to OncoMiner for GSV analysis.

The purpose of this project was to develop an efficient program to preprocess NGS data, extract the necessary information and write it in a suitable format to be inputted to OncoMiner for further analysis. NGS data files are typically large, in the order of tens of gigabytes (GBs) and downstream analysis in OncoMiner usually involves multiple samples from the cancer group and the control group. Serial preprocessing of such datasets on our local computers would take excessive amount of time to complete all the tasks. A high performance computing system such as Blue

Waters that allow multiple samples to be processed simultaneously would be essential.

In this paper, we describe the implementation of a program to preprocess NGS data files in the popular FastQ and BAM formats, converting them to files in csv format that can be inputted to the OncoMiner pipeline on three parallel computing platforms. Some background information about NGS data and the OncoMiner input requirements is given in the next section. The Methods section describes the steps taken to complete the data preprocessing using a multiprocessing approach. This preprocessing program is tested with a collection of 35 datasets from patients with AML. The resulting runtimes, speedup, and efficiencies are presented in Section 4, where we also discuss various issues encountered during the parallelization process. The conclusion and our ongoing investigations are given in section 5.

2 BACKGROUND

Human genetic information is stored in DNA molecules contained inside 23 pairs of chromosomes, designated chromosomes 1, 2,…, 22, and X. Each chromosome contains a DNA molecule represented by a very long string of letters from the four-letter alphabet denoting the nucleotide bases adenine (A), cytosine (C), guanine (G), and thymine (T). The lengths of human DNA molecules are in the range of approximately 48 million - 250 million nucleotides. DNA has a double stranded, antiparallel structure. One end of each strand is labeled 5' and the other labeled 3'. Genetic information in DNA can be found on either strand and is always read from the 5' end to 3' end.

There are three common types of point mutations that cause genetic information changes: (1) substitution occurs when a nucleotide is substituted by another; (2) insertions are "extra" nucleotides inserted into the sequence; (3) deletions are the removal of nucleotides. An example of each type is shown in Figure 1. The top line is the reference sequence. A substitution of nucleotide "C" by "A" occurs at position 7 of line 2. In line 3, the nucleotide "A" is inserted after position 25. The four nucleotides "AACC" at position 13 – 16 are deleted in line 4.

```
ATCGGGCCAAAAAACCCCCGCGCGCGAAAAATTTTT  Ref Sequence
ATCGGGACAAAAAACCCCCGCGCGCGAAAAATTTTT  Substitution
ATCGGGCCAAAAAACCCCCGCGCGACGAAAAATTTTT Insertion
ATCGGGCCAAAA____CCCGCGCGCGAAAAATTTTT  Deletion
```

Figure 1. Three types of point mutations in DNA: substitution, insertion, and deletion.

A gene comprises multiple segments of DNA that are necessary to transcribe and translate encoded genetic information into a protein. DNA is transcribed into RNA containing both exons and introns initially. During the process of RNA splicing, introns are removed and the exons are joined to form a continuous, mature mRNA. Except for the small stretch of nucleotides at the 5' end and the 3' end of the mRNA, the rest of the transcript form the coding sequence (CDS) that will be translated into a protein. A description of the organization of DNA transcriptional elements can be found at http://www.scfbio-iitd.res.in/research/orf.html. GSVs within the CDS of a gene can either be synonymous or non-synonymous. Synonymous variants do not change the resulting protein, but non-synonymous variants do. Non-synonymous GSVs can directly affect biological functions and are generally of greater biomedical concern.

NGS is a high-throughput technology for DNA sequencing. It allows for large amounts of sequences to be obtained much faster and cheaper than the traditional Sanger sequencing procedure that produces one sequence at a time. Some NGS platforms can generate up to 20 billion reads or 6 TB of data per run. The results are often stored in FastQ format (Figure 2A), which contains nucleotide sequences and their respective sequencing quality scores. Programs like Bowtie2 [5] and Burrows-Wheeler Aligner (BWA) [6] align the sequences obtained by NGS to a reference human genome and store the results in a sequence alignment map (SAM) file or its binary equivalent BAM file. The SAM format (Figure 2B) contains information about the location and nature of the differences from the reference sequence, and quality scores among other things.

Currently, a number of open-source programs are available for GSV identification and analysis. For example, ANNOVAR [7] is a tool for ANNotation Of genetic VARiants. VEP (Variant Effect Predictor) uses different scoring schemes to evaluate

A

```
@NS500477:4:HHFYGBGX2:1:11101:7339:1083 1:N:0:9
GTTACNATCATCTTGTTTCTTGAGATAGAGCATATGGGCCTATAAAAAAAATTTAAAAAATAAAATTCCACTATTC
+
AAAAA#EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEAEEEEEEEEEEAEEEEEEEAEEEEAEEEEEEEEA
@NS500477:4:HHFYGBGX2:1:11101:3189:1084 1:N:0:9
CGGATNCCTTCTCACTGTACACAGCTGCCTGGCTCCCTCTGGTCCCTGCTTCAGAGGAGGCCTGGGCCTGTGGCC
+
A/AAA#EEEEEEEE/EEEEEE<AEEAEEAE/EAE<EEAEEEAEEEEEEAAEEA/E//EAEAEEE/<EEEEA/EEE
@NS500477:4:HHFYGBGX2:1:11101:9780:1084 1:N:0:9
```

B

```
K00211:113:H52VJBBXX:8:2115:30107:30204   83   chr1 21521 1   100M =    21502 -119 TCCTGTCTTGAGTAATCCGTGGGCCCTAACTCACTCATCCCAACTCTTCACTCACTGCCC
K00211:113:H52VJBBXX:8:2113:22252:2809    147  chr1 21574 1   100M =    21508 -166 ACTGCCCTGCCCCACACCCTGCCAGGGAGCCTCCCGTGGCACCGTGGGGACACAAAGGAA
K00211:113:H52VJBBXX:8:1114:4827:28147    99   chr1 21679 0   92M5D8M   =    21755 181  AGGCTCACGGAAAGTCAGCCTCTCATGCCCCGAGAGCTGAGTGCAAGGGAGAGG
K00211:113:H52VJBBXX:8:1109:30401:16682   163  chr1 21740 1   100M =    21774 134  GTCTGTGCTTCCCATGCAGAAGCACCCCCCTCCCACCCCTGTGCAGTCCGGCCTTCGCGG
K00211:113:H52VJBBXX:8:1123:14478:41739   163  chr1 21750 1   100M =    21845 195  CCCATGCAGAAGCACCCCCCTCCCACCCCTGTGCAGTCCGGCCTTCGCGGCAGACCACCA
K00211:113:H52VJBBXX:8:1114:4827:28147    147  chr1 21755 0   16M5D84M  =    21679 -181 GCAGAAGCACCCCCCTCCCTGTGCAGGCCGGCCTTCGCGGCAGACCACCATACAC
K00211:113:H52VJBBXX:8:1109:30401:16682   83   chr1 21774 1   100M =    21740 -134 ACCCCTGTGCAGTCCGGCCTTCGCGGCAGACCACCATACACCACGTTCCAAGCCACACTG
K00211:113:H52VJBBXX:8:1215:18578:23417   99   chr1 21783 1   100M =    21872 189  CAGGCCGGCCTTCGCGGCAGACCACCATACACCACGTTCCAAGCCACACTGAGGCCTCCC
```

Figure 2. Examples of FastQ (panel A) and SAM (panel B) input files.

var_index	chrom	gene_name	left	right	ref_seq	var_seq1	count	var_score	where_in_transcript	change_type1
11173	chr1	FAM87B	820507	820508	C	G	7	34.1428571	3' untranscribed	
2070956	chr6	TBP	170569705	170569706	T	C	54	31.3888889	CDS	Non-synonymous
349657	chr2	AGBL5	27074846	27074847	G	C	12	32.5833333	5' untranscribed	
1380229	chr7	POLR2J2	102667109	102667110	A	A	8	28.25	CDS	Non-synonymous
1380256	chr7	POLR2J2	102667105	102667106	C	C	6	29.8333333	CDS	Non-synonymous
1380215	chr7	POLR2J2	102667104	102667105	C	C	6	30.8333333	CDS	Non-synonymous
1446619	chr2	POLR1B	112550939	112550940	T	A	65	31.2	CDS	Synonymous

Figure 3. OncoMiner Input (OMI) file example

consequences of genetic mutations. SNPeff [8] is a set of tools for annotating and predicting the effects of GSVs on genes. MuSiC [9] is a package that provides statistical methods to identify significantly mutated genes. Each of these program packages contains their own preprocessing procedures for converting NGS data files to the required format for downstream analysis.

The OncoMiner pipeline [4] was originally developed in support of researchers in the UTEP Border Biomedical Research Center for the investigation of GSVs from a group of patients with leukemia in the El Paso Children's Hospital. The current OncoMiner pipeline provides the necessary tools for literature search, visualization, and statistical analysis with control of false discovery rates in one package, but it can only accept input files in a specific format that comes from the NGS and GSV identification services provided by Otogenetics Corporation. However, as the scope of the study expands and more genome sequencing is now being done at different locations by different sequencers, an additional data preprocessing step is needed to obtain the required information to generate an OncoMiner input (OMI) file.

At a minimum, an OMI file requires 11 data items for each variant as shown in Figure 3. These include a unique numeric identifier (var_index), its position on the reference genome (chrom, left and right), gene name (gene_name), the nucleotides involved (ref_seq and var_seq1), the number of sequences obtained (count) and an averaged sequencing quality score over those sequences (var_score). Additionally, if a variant is within a transcript, the where_in_transcript field states whether the variant falls on a CDS, an intron, or an untranslated region. The field change_type1 tells whether the variant is synonymous or nonsynonymous if it is within a CDS. In the next section, we will describe the steps involved to get the above information from an NGS dataset of an individual in FastQ or BAM format and prepare the OMI file.

3 METHODS

3.1 Overall Workflow

The OncoMiner Preprocessing (OP) program has three components. First, the open-source Bowtie2 aligner [5] is used to align the sequence data in a FastQ file to a reference human genome to produce a BAM file. Second, we use the SAMtools software toolkit [10], along with our file-splitting awk script to sort the aligned data and separate them by chromosomes. Finally, we have developed a mutation-calling (MC) python script to identify GSVs and generate the OMI file for input into OncoMiner. The overall workflow is displayed in Figure 4 and it has been implemented on two Linux-based local computers as well as the Blue Waters system at the National Center of Supercomputing Applications (NCSA).

3.2 Alignment by Bowtie2

An NGS data file, in FastQ format (Figure 2A), would first go through Bowtie2 [5], which aligns sequences in the FastQ file to the reference human genome version GRCh38 (Genome Reference Consortium human build 38), available at the

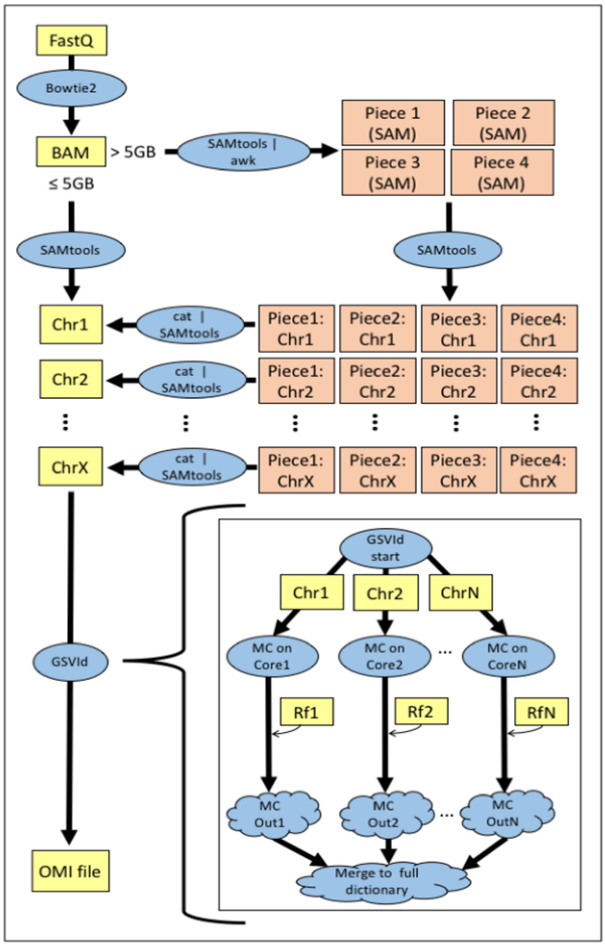

Figure 4. Workflow through which a FastQ or BAM file is processed to become an OncoMiner Input (OMI) file.

```
DDX11L1    NR_046018  chr1  +    11873 14409 14409 14409 3    11873,12612,13220,    12227,12721,14409,
WASH7P     NR_024540  chr1  -    14361 29370 29370 29370 11   14361,14969,15795,16606,16857,17232,17605,17914,18267,24737,29320,    14829,15038,
MIR6859-1  NR_106918  chr1  -    17368 17436 17436 17436 1    17368,     17436,
MIR6859-2  NR_107062  chr1  -    17368 17436 17436 17436 1    17368,     17436,
MIR6859-3  NR_107063  chr1  -    17368 17436 17436 17436 1    17368,     17436,
MIR6859-4  NR_128720  chr1  -    17368 17436 17436 17436 1    17368,     17436,
MIR1302-2  NR_036051  chr1  +    30365 30503 30503 30503 1    30365,     30503,
MIR1302-9  NR_036266  chr1  +    30365 30503 30503 30503 1    30365,     30503,
```

Figure 5. Example of a refflat file.

University of California at Santa Cruz (UCSC) Genome Browser [11]. These alignments are stored as a sequence alignment map (SAM) file as shown in Figure 2B or its binary equivalent BAM file. The SAM file contains readable text with the DNA sequence fragments as well as descriptions that include read lengths, ASCII-encoded quality scores of each nucleotide, chromosomal positions and mutation information in comparison to the reference sequence. The corresponding BAM file contains the same information in binary format, and has a much smaller file size.

3.3 File Sorting using SAMtools

The SAMtools sort and index functions are used on the BAM file generated by the alignment step above in order to group the GSVs on the same chromosome together and sort them by their positions on the chromosome. At this point, the SAMtools view function is used to extract information from the sorted BAM file and write them to 23 readable SAM files each containing the sorted, chromosome specific GSVs. These 23 chromosome specific SAM files are named Chr1, Chr2,…, Chr22, and ChrX in Figure 4, where C*n* stands for chromosome *n* for $n = 1,\ldots, 22$, and ChrX the sex chromosome (generally only chromosome X is sequenced even for a male subject as the chromosome Y is considered sufficiently similar to a portion of chromosome X).

When handling large BAM files on computers with limited memory (e.g., 32 GB), out-of-memory (OOM) errors sometimes occur when running the SAMtools sort function. This problem is circumvented by using an awk script to split a BAM file with size exceeding a threshold (e.g., 5 GB) into four pieces, each of which is then sorted by chromosome as described above. The threshold can be adjusted by the user according to the available amount of memory in the particular machine running the program. The four files associated with each chromosome are then joined back together by the cat command before feeding into the next step for GSV identification.

3.4 GSV Identification and Mutation-Calling

To identify GSVs and build the OMI file for input into OncoMiner, we have developed the mutation-calling (MC) script to parse the information contained in the chromosome specific SAM files from the previous sorting step. Information for most of the required fields in the OMI file, such as the GSV location, nucleotides involved, gene name, can be parsed directly from the SAM files. However, classifying the genomic region type for each variant is not as straightforward. GSVs have to be classified based on information obtained from the UCSC Genome Browser in the form of a refflat file (Figure 5) [12], which contains reference information for the start and end positions of various genes, introns, exons, and untranscribed regions.

Using this information, each GSV can be classified according to the decision tree shown in Figure 6. Each unique GSV is given an identifier consisting of the chromosome number and the position of the GSV within the chromosome. It is then added to the GSV dictionary. For each sequence containing the GSV, the site is counted and the sequence quality is tracked. Once all the sequences have been processed, GSVs that fail to meet minimum quality scores and sequence depth specified by the user are removed.

3.5 Parallelization

Bowtie2 is an open-source program that can run in parallel by simply specifying the number of processors in the command. In contrast, the SAMtools functions are designed to run in serial so we have not attempted any parallelization for them. However, the process of extracting GSV information from the sorted BAM file to produce the chromosome specific SAM files has been parallelized using different processors to extract the information for different chromosomes and write to different files.

For GSV identification, we have created the GSVId function that makes use of the Python multiprocessing module to run the MC script in parallel, distributing the chromosome specific files to run on different cores. Since the first two chromosomes contain the largest number of genes and are likely to take the longest to run, we start with assigning these two chromosomes to two cores first, and then the others to the remaining available cores.

3.6 Implementation and Testing

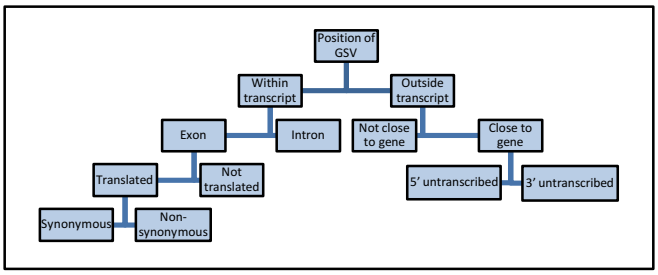

Figure 6. GSVId decision tree. The MC script classifies mutations according to their positions in relation to genetic transcripts. All decisions are based on gene information within the refflat file except for "Close to gene" which is a tunable parameter set as a default of 5000 nucleotides from either beginning or end of transcript.

The OP program has been implemented on two local computers at UTEP. The first one is BioTower, a Dell Precision 5810 containing an Intel Xeon E5-1650 with a 12-core processor and 32 GB memory. The second machine is BinfCompute, a more powerful Dell PowerEdge R730 with 32 cores (dual Intel Xeon E5-2667 processors with 16 cores) and 256 GB memory. Both computers have CentOS 7 as operating system and use the OS default version of Python (v2.7). The required Python modules, Bowtie2, SAMtools and reference files are locally available on these machines.

For initial testing of OP, we used a FastQ file generated by our local DNA sequencer in the Genomics Core Facility in the Border Biomedical Research Center at UTEP. The file was 7.3 GB in size containing 29 million 100-base long sequences spanning all 23 human chromosomes. OP was run on both machines using varying number of cores to check that parallelization of each step has been achieved. The final output file of OP was inputted to OncoMiner to check that a legitimate OMI file was produced.

For further performance testing, we have also implemented OP on the Blue Waters system, a Cray XE/XK hybrid machine composed of AMD 6276 Interlagos processors running the Cray Linux Environment. Our OP program implementation uses only one high memory XE node with 128 GB total memory to process the dataset from one individual. The Blue Waters Python software stack bwpy and PrgEnv-gnu modules have to be loaded in order to use Python and the GNU programming environment respectively.

A collection of BAM files containing the aligned sequence information of 35 patients with AML was obtained from The Cancer Genome Atlas (TCGA, http://cancergenome.nih.gov/) for testing [13]. These file sizes range from 15 to 54 GB. In all the test runs, runtimes were determined using the Linux time command and internally using Python's time module. Internal memory usage was determined using Python's getrusage() function.

We have assessed the performance of the parallelization on the local machines by calculating the efficiency of core usage as $T(1)/[pT(p)]$ where $T(p)$ is the measured runtime using p cores with varying p = 1, 2, 4, 8, 16, and 24. Statistical analysis of efficiencies and runtimes were performed using the functions for t-tests and linear models in the statistical software package R [14].

4 Results and Discussion

4.1 The OP program

The OP program has been successfully implemented on both of our local machines BioTower and BinfCompute. It can take FastQ files as input and produce OMI files as output in the correct format that can then be fed into OncoMiner for downstream analysis. Because many datasets in public databases such as TCGA are already stored in the form of BAM files, our OP program has been set up to also take input in BAM format.

To set a baseline for parallelization performance assessment, we first conducted runtime measurements of OP using the locally generated 7.3 GB FastQ file on a single core in the two local machines. The runtimes of the various steps on BinfCompute are displayed in Table 1 (the run time distribution on BioTower is similar). These results show that about 80% of the total OP runtime is taken by Bowtie2 in the alignment step. Fortunately, Bowtie2 is designed to run in parallel, and the speedup using multiple cores seems quite substantial. While the SAMtools sort and index functions must run serially, parallelization of the extraction process to produce chromosome specific SAM files by the SAMtools view function has reduced the runtime somewhat.

Table 1. Runtimes (in minutes) of different steps of OP on BinfCompute using 1, 8, and 24 cores.

Function	1 Core	8 Cores	24 Cores
Alignment (Bowtie2)	145.16	22.39	10.08
File sorting (SAMtools)	13.12	6.33	5.85
GSVId (MC script)	19.17	3.30	2.56
Others	0.32	0.29	0.29
Total	**177.76**	**32.31**	**18.78**

As the GSV identification part of the OP program was developed entirely by our group, we examined the MC script performance more carefully. Figure 7 displays the runtimes for our test dataset on BioTower and BinfCompute, showing substantial speedups as the number of cores increases from 1 to 8 on both machines. For BioTower with only 32 GB of RAM, the execution speed deteriorated sharply beyond 8 cores as swap memory started to be used. We therefore stopped the BioTower runtime measurements at that point, but continued the measurements on BinfCompute using higher number of cores. Overall, the best average MC script speedup achieved on BioTower was 3.32 using 6 cores, and on BinfCompute was 9.29 using 16 cores.

Aside from the locally generated 7.3 GB FastQ file, we also selected 9 datasets from our AML collection obtained from TCGA to run on the two local machines. These data files were already in BAM format but were much larger than our test dataset. OOM errors were encountered on BioTower during the execution of the SAMTools sort function. Such problems did not occur on BinfCompute though. Given that BioTower has only 32 GB of memory while BinfCompute has 256 GB, this is not surprising. It has, however, suggested that memory requirement is an issue at this point of the OP program.

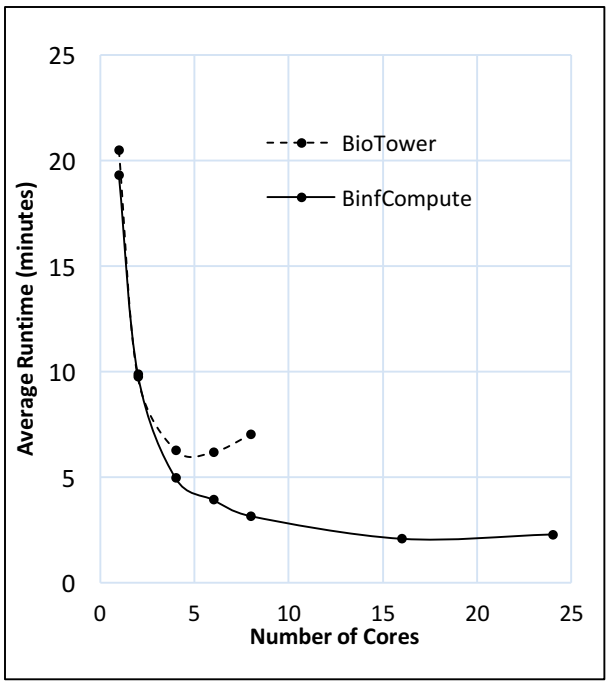

Figure 7. Runtimes for test dataset on two local computers: BioTower and BinfCompute.

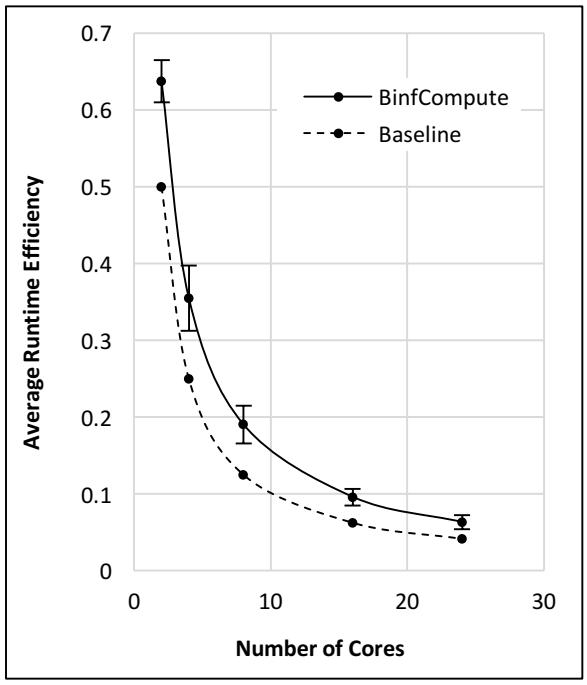

Figure 8. Average efficiencies ± standard deviation for 2, 4, 8, 16, 24 cores on BinfCompute. Dashed curve indicates baseline efficiencies.

With the expectation that the OP program might need to be run on other computers without large amounts of memory, we used awk to split the large BAM files exceeding a cutoff file size into four pieces and let each piece be sorted one at a time. The cutoff file size can be set by the user with consideration to the available memory of the specific computer running OP. For BioTower, we found that a 5 GB cutoff worked well. The splitting slowed down the file sorting step substantially, but OOM errors were avoided.

We further examined the overall parallelization efficiency of the OP program on BinfCompute for the 9 selected AML dataset. If no speedup were achieved at all by the parallelization, the efficiency would have a baseline value of 1/p. Figure 8 shows the average efficiency of the OP program on BinfCompute with p = 2, 4, 8, 16, 24. In each case, a t-test confirmed statistically that the average efficiency was at least 20% higher than the baseline values at significance level $\alpha = 0.05$.

In the test runs of these 9 files, the runtimes required ranged from around 15 minutes to almost 9 hours. It seemed not practical to run the OP program for all the TCGA datasets on our local computers as each run would tie up the computer for a substantial amount of time and prevent others from running their jobs on these machines that were used heavily. We therefore turned to Blue Waters to utilize the allocated resources to complete this project.

4.2 Running OP on Blue Waters

The OP program was installed on Blue Waters using one high memory XE node with 128 GB memory to process each of the 35 datasets from the AML collection. While Blue Waters allows multiple nodes to be used at one time, we decided to process a dataset in a single node, as the shared memory within the same node made it easier to construct the dictionary of GSVs and kept communication time to a minimum. However, we were able to use multiple nodes to independently process multiple datasets simultaneously.

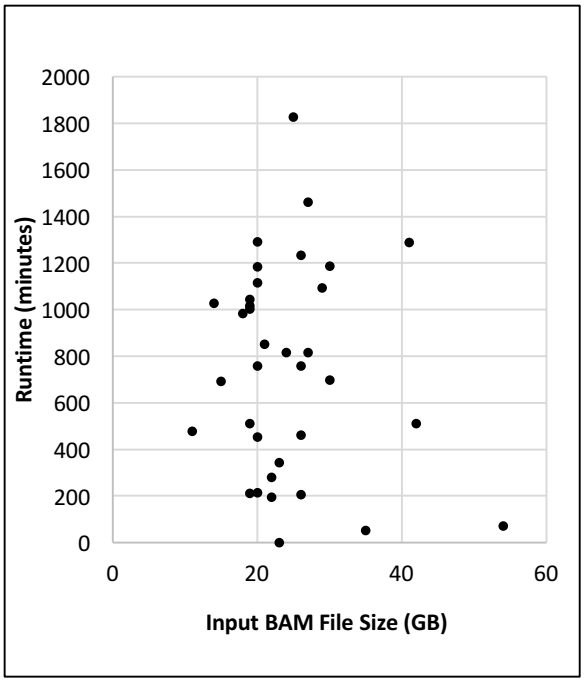

Figure 9. OP runtimes versus input file size for 35 datasets in AML collection

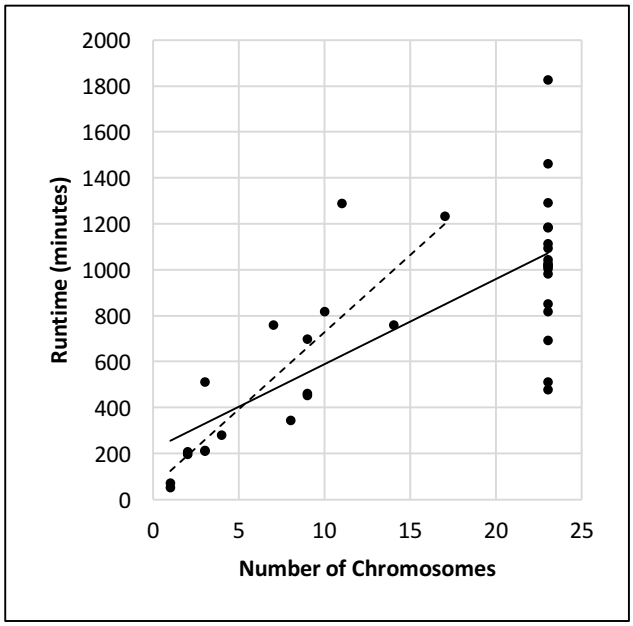

Figure 10. OP runtimes versus the number of chromosomes contained within the files. The solid regression line was determined with all data points while the dashed line used only the data points with no more than 22 chromosomes.

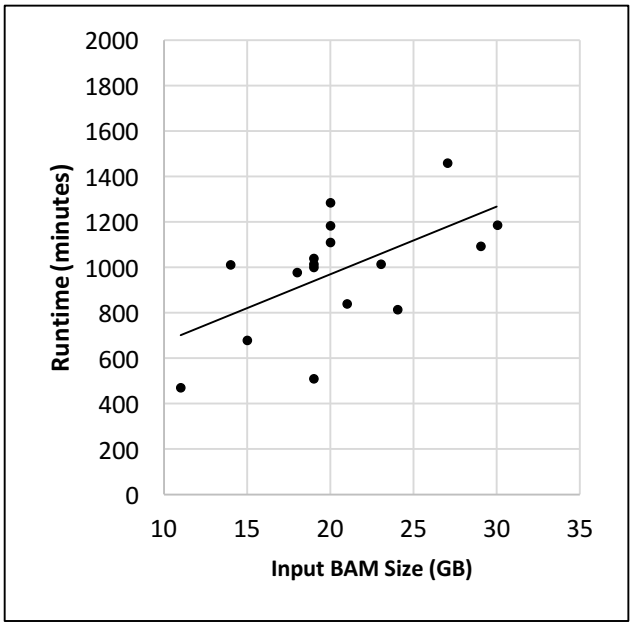

Figure 11. OP runtimes versus input file size for datasets with 23 chromosomes

OOM errors occurred in the MC script on Blue Waters for a few datasets when the node could not provide sufficient memory for processing all the chromosome files at once. In those cases, we had to reduce the number of cores used so that fewer chromosomes were processed at one time. On hindsight after understanding the shared memory constraints, a better approach to parallelize the OP program on Blue Waters would be to couple MPI to our script and allocate a node to process each of the four pieces of one chromosome-specific file, and then join the four dictionaries afterwards. We plan to implement this approach in the next version of OP.

Furthermore, we observed that using more than one core in the node produced no speedup in runtime. After monitoring the memory usage on Blue Waters, the reading and writing (I/O) of the utilized files was believed to be the cause of this lack in speedup because of the data transfer to and from the compute node. This would not have readily been seen on the BinfCompute and BioTower machines using local storage of the data but is a known issue for HPC systems analyzing large datasets [15].

Despite these issues, Blue Waters was the only platform that provided us with sufficient resources to process our complete AML dataset collection. From the recorded runtimes on Blue Waters, we were also able to investigate which characteristics of the datasets would influence the runtimes significantly, as described below.

4.3 Runtime correlates with input file size and chromosome diversity

The runtimes for our 35 AML datasets using one core varied from 53 minutes to over 30 hours. One would expect larger input files to require longer runtime. Surprisingly, a simple linear regression analysis indicated that the correlation r = 0.152 was not significant (p value = 0.385). Looking at the scatter plot in Figure 9, we noticed a few outliers that might have contributed to the unexpected result. For example, the largest input file (54 GB) ran very quickly. On closer examination, we found that this dataset contains GSVs from only one chromosome of the patient.

This prompted us to look into how the number of chromosomes in the input file might affect the OP runtime (Figure 10). This time, regression analysis showed a highly significant linear correlation (r = 0.762, p value = 1.07e-07). The correlation was even stronger (r = 0.858) when only the datasets with no more than 22 chromosomes were considered. This result suggested that chromosome diversity could be an important factor that influenced the OP runtime. The I/O involved in the analysis of individual chromosome files is believed to drive the major difference in runtimes of files with different numbers of chromosomes. Having multiple cores trying to access the larger files from a network drive at the same time could create a bottleneck in the accessibility of the data and thus causing the individual processes to slow down. This would not be as much of a factor when pulling data from a local hard drive.

When the OP runtimes for only those files with complete sets of 23 chromosomes are analyzed, we then see a significant positive

correlation with input file size (see Figure 11, r = 0.509, p value = 0.031). This implies that a strong relationship between input file size and runtime indeed exists once the number of chromosomes is fixed.

To complete the runtime analysis, a multiple regression model was fit to our runtime data with the number of chromosomes and input file size as covariates, producing the regression equation:

$$\text{Runtime} = 94.8 + 20.3 * (\#\text{ chromosomes}) + 22.3 * (\text{file size})$$

with coefficient of determination 0.749, implying that almost 75% of the variations in runtime can be explained by input file size and chromosome diversity together. This regression equation will allow OP runtimes to be estimated when we process new data files in the future.

5. CONCLUSION AND FUTURE WORK

We have implemented the OP program, which comprises the open source Bowtie2 and SAMtools programs, as well as our GSVId function, on two local computers at UTEP and on Blue Waters at the NCSA. The OP program can preprocess NGS data stored in either FastQ or BAM format and obtain the necessary information to produce an OMI file to be inputted to OncoMiner for downstream genetic variation analysis.

We have demonstrated that our multiprocessing parallelization approach in GSVId for the MC script works with reasonable efficiencies on our local computers. However, the same parallelization using multiple cores on one node in Blue Waters did not produce any substantial speedup of the OP program. We have identified possible factors involving memory constraints and I/O issues that limit the OP program's performance on Blue Waters and will continue to develop a better approach using MPI to distribute the analysis of a single dataset to multiple nodes. We also plan to test the script on other high performance platforms with more memory in a single node and internal solid state drives for the I/O intensive portions of the code.

Despite the memory and I/O issues encountered, Blue Waters provided the necessary resources for us to process our entire collection of datasets from 35 patients with AML and showed that OP runtimes were correlated not only with the input data file size, but also with chromosomes diversity.

Aside from the most popular FastQ and BAM formats, genome sequence variant data may also come in other file formats such as the variant call format (VCF). The OP program framework set up in this project now allows us to adapt and extend our code relatively easily to process files in other formats to produce OMI files for input to OncoMiner.

6. REFLECTIONS

The Blue Waters Student Internship Program (BWSIP) allowed me to learn about parallel computing, which I had not been previously introduced to. I am glad to have attended the two-week Petascale Institute 2016 workshop. What I learned during the workshop and internship will be useful in the future for performing bioinformatics analysis. The NCSA and Shodor staff was very helpful and explained the concepts of parallel computing clearly. I had an eye-opening experience in the BW Symposium in May of 2017 as I got to see how HPC was used in so many different fields with methodologies that I had not encountered in bioinformatics related projects.

Due to parallelization of the MC script having been effective on the local machines, memory usage was a suspected cause of static runtime on Blue Waters. The process of ruling out possible causes for the lack of speedup was an invaluable lesson. Looking at the memory usage of the MC script helped me better understand memory related issues in parallel programming and taught me how to monitor the usage at different points in the script. Although the runtime remained static, I now have a better feel for how to check to see if our current suspect, I/O, is behind it. In this, the future work will revolve around machines with higher memory and designed for I/O intensive programs.

The research experience also allowed for a local collaboration at UTEP. While working with my group, I learned about version control and improved on communication with other group members. I learned the value in clarifying tasks at the onset and in keeping track of changes to allow easier modifications during debugging. In conclusion, the internship has not only made me more knowledgeable in the use of HPC systems, but also trained me to become a better researcher.

7. ACKNOWLEDGMENTS

This research is funded in part by the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. This project was also in part supported by NIH Grant #5G12RR007592 from the National Center for Research Resources (NCRR)/NIH to the Border Biomedical Research Center at The University of Texas at El Paso. The results published here are in whole or part based upon data generated by the TCGA Research Network: http://cancergenome.nih.gov/.

8. REFERENCES

[1] Bullinger, L., Dohner, K. and Dohner, H. Genomics of Acute Myeloid Leukemia Diagnosis and Pathways. *J Clin Oncol*, 35, 9 (Mar 20 2017), 934-946.

[2] Corvol, H., Blackman, S. M., Boelle, P. Y., Gallins, P. J., Pace, R. G., Stonebraker, J. R., Accurso, F. J., Clement, A., Collaco, J. M., Dang, H., Dang, A. T., Franca, A., Gong, J., Guillot, L., Keenan, K., Li, W., Lin, F., Patrone, M. V., Raraigh, K. S., Sun, L., Zhou, Y. H., O'Neal, W. K., Sontag, M. K., Levy, H., Durie, P. R., Rommens, J. M., Drumm, M. L., Wright, F. A., Strug, L. J., Cutting, G. R. and Knowles, M. R. Genome-wide association meta-analysis identifies five modifier loci of lung disease

severity in cystic fibrosis. *Nat Commun*, 6 (Sep 29 2015), 8382.

[3] Hilven, K., Vandebergh, M., Smets, I., Mallants, K., Goris, A. and Dubois, B. Genetic basis for relapse rate in multiple sclerosis: Association with LRP2 genetic variation. *Mult Scler* (Jan 1 2018), 1352458517749894.

[4] Leung, M.-Y., Knapka, J. A., Wagler, A. E., Rodriguez, G. and Kirken, R. A. *OncoMiner: A Pipeline for Bioinformatics Analysis of Exonic Sequence Variants in Cancer*. In: Big Data Analytics in Genomics, Wong, K.C. (Ed.), Springer, New York, USA (2016), 373-396.

[5] Langmead, B. and Salzberg, S. L. Fast gapped-read alignment with Bowtie 2. *Nat Methods*, 9, 4 (Mar 4 2012), 357-359.

[6] Li, H. and Durbin, R. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics*, 25, 14 (2009), 1754-1760.

[7] Wang, K., Li, M. and Hakonarson, H. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Res*, 38, 16 (Sep 2010), e164.

[8] Cingolani, P., Platts, A., Wang le, L., Coon, M., Nguyen, T., Wang, L., Land, S. J., Lu, X. and Ruden, D. M. A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. *Fly (Austin)*, 6, 2 (Apr-Jun 2012), 80-92.

[9] McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M. and DePristo, M. A. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res*, 20, 9 (Sep 2010), 1297-1303.

[10] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G. and Durbin, R. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25, 16 (Aug 15 2009), 2078-2079.

[11] Karolchik, D., Baertsch, R., Diekhans, M., Furey, T. S., Hinrichs, A., Lu, Y. T., Roskin, K. M., Schwartz, M., Sugnet, C. W., Thomas, D. J., Weber, R. J., Haussler, D. and Kent, W. J. The UCSC Genome Browser Database. *Nucleic Acids Res*, 31, 1 (Jan 1 2003), 51-54.

[12] Tyner, C., Barber, G. P., Casper, J., Clawson, H., Diekhans, M., Eisenhart, C., Fischer, C. M., Gibson, D., Gonzalez, J. N., Guruvadoo, L., Haeussler, M., Heitner, S., Hinrichs, A. S., Karolchik, D., Lee, B. T., Lee, C. M., Nejad, P., Raney, B. J., Rosenbloom, K. R., Speir, M. L., Villarreal, C., Vivian, J., Zweig, A. S., Haussler, D., Kuhn, R. M. and Kent, W. J. The UCSC Genome Browser database: 2017 update. *Nucleic Acids Res*, 45, D1 (Jan 4 2017), D626-d634.

[13] Ley, T. J., Miller, C., Ding, L., Raphael, B. J., Mungall, A. J., Robertson, A., Hoadley, K., Triche, T. J., Jr., Laird, P. W., Baty, J. D., Fulton, L. L., Fulton, R., Heath, S. E., Kalicki-Veizer, J., Kandoth, C., Klco, J. M., Koboldt, D. C., Kanchi, K. L., Kulkarni, S., Lamprecht, T. L., Larson, D. E., Lin, L., Lu, C., McLellan, M. D., McMichael, J. F., Payton, J., Schmidt, H., Spencer, D. H., Tomasson, M. H., Wallis, J. W., Wartman, L. D., Watson, M. A., Welch, J., Wendl, M. C., Ally, A., Balasundaram, M., Birol, I., Butterfield, Y., Chiu, R., Chu, A., Chuah, E., Chun, H. J., Corbett, R., Dhalla, N., Guin, R., He, A., Hirst, C., Hirst, M., Holt, R. A., Jones, S., Karsan, A., Lee, D., Li, H. I., Marra, M. A., Mayo, M., Moore, R. A., Mungall, K., Parker, J., Pleasance, E., Plettner, P., Schein, J., Stoll, D., Swanson, L., Tam, A., Thiessen, N., Varhol, R., Wye, N., Zhao, Y., Gabriel, S., Getz, G., Sougnez, C., Zou, L., Leiserson, M. D., Vandin, F., Wu, H. T., Applebaum, F., Baylin, S. B., Akbani, R., Broom, B. M., Chen, K., Motter, T. C., Nguyen, K., Weinstein, J. N., Zhang, N., Ferguson, M. L., Adams, C., Black, A., Bowen, J., Gastier-Foster, J., Grossman, T., Lichtenberg, T., Wise, L., Davidsen, T., Demchok, J. A., Shaw, K. R., Sheth, M., Sofia, H. J., Yang, L., Downing, J. R. and Eley, G. Genomic and epigenomic landscapes of adult de novo acute myeloid leukemia. *N Engl J Med*, 368, 22 (May 30 2013), 2059-2074.

[14] R Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. (2017).

[15] Reed, D. A. and Dongarra, J. Exascale computing and big data. *Communications of the ACM*, 58, 7 (2015), 56-68.