2021 년 8 월

석사학위 논문

Constructing an ethnic-specific variant calling workflow based on a systematic comparison of multiple pipelines

조선대학교 대학원

글로벌바이오융합학과

박 현 슬

Constructing an ethnic-specific variant calling workflow based on a systematic comparison of multiple pipelines

개인 유전체 분석 파이프라인의 체계적 비교연구를 통한 인종 특이적 분석법 구축

2021 년 8 월 27 일

조선대학교 대학원

글로벌바이오융합학과

박 현 슬

Constructing an ethnic-specific variant calling workflow based on a systematic comparison of multiple pipelines

지도교수 김정수

이 논문을 이학 석사학위 신청 논문으로 제출함.

2021년 4월

조선대학교 대학원

글로벌바이오융합학과

박현슬

박현슬의 석사학위논문을 인준함.

위원장 조선대학교 교수 이건호 (인)

위원 조선대학교 조교수 김석준 (인)

위원 조선대학교 조교수 김정수 (인)

2021 년 5 월

조선대학교 대학원

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Constructing an ethnic-specific variant calling workflow based on a systematic comparison of multiple pipelines

Hyeonseul Park

Advisor: Prof. Jungsoo Gim, Ph.D.

Department of Integrative Biological Sciences

Graduate School of Chosun University

Next Generation Sequencing (NGS) is an experimental method that can read genome at high speed with low cost, while the sanger sequencing method used for the Human Genome Project (HGP), which took 13 years. NGS amplifies fragmented DNA by PCR and reads a sufficient amount of sequence through sequencing. The bases read by the sequencer, called "reads" are analyzed through three steps. The first step is the Quality Control (QC), which shows the quality of the reads and filters out the low-quality reads. The second is the alignment, which maps the reads to the reference genome. In the final step, variant calling analysis is performed to extract bases that differ between the mapped reads and the reference genome.

There have been developed many methods for the alignment and the variant call of human with different algorithm. Their performance, however, has been studied using the reference genome based on Caucasian and Caucasian data called “NA12878”. Considering genetic differences, questions have been raised whether the methods optimized for Caucasians to show the same results for East Asians genetic differences.

In this study, I analyzed the performance difference of each method using Whole Genome Sequencing (WGS) data of NGS. The WGS data used for comparing are Caucasian (NA12878) and Korean data. In case of NA12878 data, the performance is compared with the known correct answer (gold standard), and for Korean data, performance is compared with the data of the Korean microarray chip. Among the various methods that were used in the previous report, I selected 2 alignment methods (BWA-mem, NovoAlign) and 4 variant call methods (GATK4, Strelka2, DeepVariant, Samtools) that are known to perform well. The analyzable 8 combinations were compared. As a result of analyzing NA12878, the run time required for variant call in Samtools was long, and the recall was relatively low. Comparing the required time, the run time for the combination of BWA-mem and Strelka2 was the shortest. For Single Nucleotide Polymorphism (SNP) calling, which was regarded as a criterion for evaluating performance, the run time for the combination of BWA-mem and Strelka2 was the highest, and for recall, the combination of NovoAlign and GATK4 was the highest. In INDEL calling, precision was highest in the BWA-mem and Strelka2 combination showed very high precision, and recall precision was highest in the NovoAlign and Deep Variant combination. Therefore, in the combination for analyzing Korean genome, the variant call was performed with the combination excluding Samtools.

The Markduplicate acting between alignment and variant call caused an overall time difference. The Korean sequencing data did not perform the PCR step, so there was no significant difference when comparing the performance with and without the Markduplicate step. Therefore, if the PCR step has not been performed, Markduplicate step can be skipped.

When comparing the variants resulting from each pipeline with the variants of chip data using Korean sequence data, there is no large difference in the number of matching variants for each pipeline. However, among the pipeline combinations, the number of variants that matched the NovoAlign and GATK4 combinations was the highest from the data of all Koreans. Conclusively, the BWA-mem and Strelka2 combination is fastest to call variants and the NovoAlign and GATK4 combination is highly concordance with chip data and call many variants.

When using the Korean reference genome instead of the existing reference genome in looking only at the BWA-mem and Strelka2 combinations, the SNP recall and INDEL precision and recall values were low, and there were few variants that matched the chip data. Therefore, it took a result that even if using Korean sequence data, alignment and variant call using existing reference genome would perform better.

I. INTRODUCTION

With the start of the Human Genome Project (HGP) in 1990, it became possible to read all of the human genome. (Collins et al., 2003) With the end of the HGP in 2003, scientists wanted to read genomes cheaper and faster than before, and Next Generation Sequencing (NGS) technology emerged. (Venter et al., 2001)

If sanger sequencing, the method used in HGP, could analyze 1000 bases in one experiment, NGS could analyze from 1 million to 1 billion bases. The fast and highly readable NGS allowed us to compare normal genetic variation with individuals, races, and patients in the human genome. There are several ways to confirm genetic variation, but in this paper, I analyzed the data using Whole Genome Sequencing (WGS). WGS data analysis is largely divided into quality control, alignment, and variant call stages, and there are several analysis methods for each stage. Since each analysis method has different advantages and disadvantages, its performance is also different, and there are many benchmark papers to find the optimal combination. The data used for performance evaluation was mainly Caucasian, NA12878. Due to ethnic differences, it was questioned whether the pipeline claimed in the papers using Caucasian as a performance evaluation was significant in other racial groups as well. Therefore, in this paper, I would like to find a combination of an alignment tool and a variant caller that better detect genetic variation when using Korean data.

There were no benchmark papers using Korean data. In 2018, dozens of Korean variom databases were created and vcf files were created using BWA and GATK UnifiedGenotyper (Jungeun Kim et al., 2018). Other papers using Korean data only used BWA and GATK. This paper looked for a pipeline that accurately and quickly finds genetic differences in Korean data. Compared to using one alignment tool and variant caller, I propose a pipeline that can be used according to the researcher's purpose by viewing each result using various tools.

I-1. Human Genome Project

After knowing that DNA constitutes a organisms, the HGP began with the idea that humans could learn a lot about organisms if they knew all the DNA that an organism has.

The project, which started in 1990, is based on the sanger sequencing method developed in 1977. However, since it was necessary to sequencing a large amount at high speed, a lot of effort was made to develop sequencing technology.

HGP was implemented from 1990 to 2003 for the purpose of identifying the sequence of about 3 billion nucleotides base pairs in the human genome, and 3 billion human genomes were repeatedly read to reduce errors and obtain accurate sequences. In December 1995, a physical map showing the actual physical location of the chromosomes was created. In 1998 Single Nucleotide Polymorphism (SNP) initiative begin. (Collins et al., 2003)

Twenty sequencing centers from around the world formed the International Human Genome Sequencing Consortium to participate in this project. Each center sequencing the fragmented genome of a single person and combining them to complete the sequence was sequenced with an accuracy of 99.99%, including about 99% of the gene-containing region of the human genome. After the completion of HGP, a project to develop a technology capable of sequencing the human genome for $1000 was held. In 2007, the cost of DNA sequencing decreased by 10 times compared to 2001. The cost of DNA sequencing has dropped sharply since 2008, a 100-fold decrease compared to 2007. The sequencing method used at this time was not an improvement of the existing sanger sequencing, but a sequencing technology of a completely different concept was developed, and was called NGS.

I-2. Next Generation Sequencing

With the development of sequencing technology to read genomes, I can read many sequences at a faster rate at a lower cost than before through the technology of NGS for each genome. The NGS process has three major steps, it cuts the DNA into constant and many pieces. Oligonucleotides with a specific sequence are attached so that the sequencing device can recognize these cuts. And this step is called creating a library. This is followed by reading the nucleotide sequence of the library DNA strand with a sequencing device, and finally analyzing the data generated by the sequencing device. These three major steps are the steps to perform NGS. WGS is the process of sequencing DNA using NGS technology without any additional work to obtain data for all DNA. Because WGS analyzes the entire genome, it can analyze not only exons, but also introns and untranslated regions. (식품의약품안전평가원, 2019)

Among the first of NGS steps, a PCR amplification step is included in the library preparation step. The PCR amplification step occurs between the step of attaching the adapter to the cut piece and the step of validation of the library. Why is a PCR amplification step necessary? This step is performed to enrich the DNA fragment attached to the adapter and increase the concentration of the library, but there is a problem that some sequences appear a lot due to uneven amplification or the composition of the base is biased to one side, causing sequencing to occur. To alleviate this problem, a PCR-free library construction kit was created that eliminates the PCR process.

The equipment for sequencing is also various. Among the sequencing equipment, the most representative Illumina equipment records the base sequence by changing fluorescence through bases with different fluorescence attached. Ion Torrent attaches a DNA library to the bead and enters it one by one in the minute hole, and has a semiconductor chip circuit, which makes it a complementary base. It detects that the pH is lowered due to hydrogen ions released during the bonding process. And Oxford Nanopore. The whole genome sequencing of Korean data made in this paper was performed using Illumina equipment.

Due to the low price of NGS, fast speed, and the advantage of being able to read many sequences, it was possible to read the genome sequence of people, and it was confirmed that a variant exists in the genome sequence for each individual, and this variant is not affected by various diseases and phenotypes. It turns out that there is a connection. As such, the diversity of genotypes can determine not only physical characteristics, but also individual differences in diseases, which can lead to personalized medicine. Therefore, humans tried to find genetic variants that could be different for each individual, and tools with different algorithms to find variants were created.

I-3. Bioinformatics analysis of NGS data

There are three main steps, from reading material to finding variations. There is a Quality Control (QC) step that adjusts the quality of the read base, an alignment step that maps the read base to a reference, and a variant calling step that searches for Single Nucleotide Variants (SNVs) when the reference and base are different.

The bases obtained through this sequencing process are analyzed in the form of a FASTQ file. The FASTQ format is composed of text and includes a nucleotide sequence and a quality score corresponding to the nucleotide sequence. One sequenced read includes a total of 4 lines, from the line indicating information about the data starting with the '@' character of FASTQ, the line indicating the base sequence, and the quality information consisting of ASCII characters. Using the quality information on the base sequence of FASTQ, a contamination sequence or a low-quality sequence is found and the quality of the sequence is increased through a preprocessing process, and this process is called the QC step.

After obtaining a sequence of good quality through QC, a step called mapping or alignment is performed. This step is to align the reads to the reference genome. At this time, the reference genome is data created by sequencing by receiving DNA from several donors, and mainly hg19 or hg38 is used.

After mapping the sequence to the reference genome, analyze the variant. This process is called variant calling, and it is a step in which the reference genome and the mapped reads are different from the reference genome.

I-4. Pipelines with different algorithms

In this paper, I used a pipeline whose performance has been verified in several papers. BWA-mem and NovoAlign were selected as Aligner. BWA is an algorithm that performs alignment using Burrow-Sheeler Transform (BWT) and Suffix array, and NovoAlign uses hash table for indexing of reference fasta and Needleman-Wunsch algorithm for alignment scoring. As variant callers, I chose GATK4, Strelka2, DeepVariant, and Samtools.

In GATK4, HaplotypeCaller finds variants in four main processes. A place where reference and other alleles appear repeatedly is defined as ActiveRegion. After that, haplotypes that might come out of the data are extracted and the haplotypes are mapped back to the reference using the Smith-Waterman algorithm to identify potential variants. After sorting and comparing the reference and haplotype, pairwise mapping the Reads of ActiveRegion to each haplotype using the PairHMM algorithm. The genetic variant is found by selecting the highest probability of the target allele for each read in the read and haplotype matrix.

Strelka2's germline calling model uses haplotype identification. This method uses a fast k-mer ranking approach to simple loci and local assembly to complex or repetitive regions.

DeepVariant is used after training a TensorFlow-based image classification model.

Samtools uses the Base Alignment Quality (BAQ) algorithm. This algorithm is the Phred-scaled probability of a read base being misaligned. Configure the profile HMM for BAQ calculation. Then calculate BAQ and call SNP.

I-5. Suitable pipeline by ethnic difference

In humans, 99.9% of the bases in the entire genome are similar. Of these, it is the remaining 0.1% that makes the difference, and more than 90% of these 0.1% bases are SNPs. Therefore, most of the cases in which the reference genome and sequence are different through variant calling are SNPs, and only about 15% of them were estimated to be different depending on the population. Different SNPs may exhibit different skin or hair color, in addition to susceptibility to diseases between races. For this reason, there may be differences in performance between races when searching for variants using different alignment tools and variant callers.

The reference fasta file, which is mainly used for analysis, is close to Caucasian. East Asians and Caucasians can see that their SNPs are different just by looking at their appearance. Therefore, it has been thought that the use of a reference appropriate to the race in the WGS analysis improves the accuracy of the analysis.

The data frequently used in Benchmark papers is NA12878, which is Caucasian. Since NA12878 has been studied a lot, the correct answer for variant exists. Therefore, there is gold standard data that can be compared when evaluating performance. When comparing the performance of several pipelines for the NA12878, I selected a pipeline that showed good performance. Combination BWA-mem and Samtools (Sohyun Hwang et al., 2015), BWA-mem and DeepVariant, GATK4 (Anna Supernat et al., 2018), BWA-mem and Strelka2 (Jiayun Chen et al., 2019), BWA-mem and DeepVariant, NovoAlign and DeepVariant, BWA-mem and Samtools, NovoAlign and Samtools (Manojkumar Kumaran et al., 2019). All pipelines that showed good performance in the above benchmark paper used NA12878 as subjects. Since NA12878 is Caucasian data, it was questioned whether the above results were significant in Korean as well.

Since there is a difference between ethnic, a Korean reference was made by assembling a reference fasta file in Korea, with the idea that different references for each race should be used when analyzing. Korean references were made in 2009 and 2015. References made in 2009 were based on hg19 to measure the genetic variation and location of Koreans. References made in 2015 are new It is said that the technique was applied.

I-6. Chip data for comparison of Korean data

There are two methods of producing genomic information, and there are two methods of using the NGS and genome chips. Since NGS reads a large amount of information, it has a disadvantage that it requires high-performance equipment. However, since the genome chip requires a lower experimental cost and a lower level of computing power than NGS analysis, it can be used more efficiently than NGS when researching on known genomic information. In the genome chip method of obtaining genomic information using microarray, probes related to known genomic information are arranged at high density on a small substrate, and genomes complementarily bonded thereto can be observed.

Korea National Institute of Health (KNIH) has established a community-based cohort targeting Koreans in Ansan and Anseong since 2001. This cohort was followed up every two years. People in the cohort were analyzed using commercial chips. However, in several studies, the limitations of existing research techniques using commercial chips were pointed out. Existing commercial chips were also designed around Caucasians, so their genome representation for Asians was low. (Wong et al. 2013) For this reason, in 2014, through the Korean chip business, a “Korean customized genome chip” was developed. 384 Koreans were analyzed in this Korean genome chip, and Affymetrix's Axiom Genotyping analysis method was used. (국립보건연구원, 2018)

I-7. Research question & Purpose

The reference genome used in the sequencing step or alignment step is mainly composed of Caucasian genomes. In addition, I thought that alignment tools or variant callers made accordingly would be tailored to Westerners. If so, what alignment tool and variant caller should be used for Asian and Korean, especially when using the same reference genome, to analyze more accurately and quickly? This paper began with these questions. In the combination of known alignment tools and variant caller, I studied which pipeline would be better for analyzing Korean, and I would like to propose a combination.

II. WORKFLOW OF ANALYSIS

II-1. NA12878 data for selecting variant caller

The FASTQ file of the Caucasians NA12878 sample was obtained from the SRA database through data called SRR8454589. This data is sequencing NA12878 close to 30x using Illumina Novaseq. (Chen et al., 2019) Usually, SRR data is received by fastq-dump, but when the size of SRR data is more than 20GB, it is difficult to receive it through fastq-dump. Since the NA12878 data was 28GB, I downloaded it using the -max-size option in prefetch. The downloaded file is paired-end, so use the -split-files option of fasterq-dump to divide the file.

FastQC was performed to see the quality information on the base of the downloaded file, and trimming was not performed because the quality was good.

The reference fasta file is used in the alignment and variant calling steps. The reference fasta file was used by receiving the hg38 version through the bucket.

II-2. Using two alignment tools

When using BWA-mem (version 0.7.17), index for reference fasta is first performed and alignment is performed. Threads can be given through the -t option. After alignment, it was changed to bam using Samtools (version 1.10), and index was performed for calling.

When NovoAlign is used, the reference fasta file is also indexed, and indexing proceeds quickly because the hash table is used for indexing. However, since there is no option to give a thread, it takes longer than BWA. The -c option speeded up a bit, but the higher the number, the lower the mapping rate, so I excluded it from the option. The resulting Sequencing Alignment Map (SAM) file is changed to bam using Samtools, and index is performed for calling.

II-3. Identifying variants using four variant calling tools

By referring to existing papers, I selected four variant call tools that claimed to have good performance in each paper.

GATK4 (version 4.1.8) outputs Baserecalibrator, ApplyBQSR, Haplotypecaller, and FilterVCF in parallel on Linux, and merges the results into one VCF file through MergeVCF. After that, the file released as g.vcf by HaplotypeCaller was created as GenotypeGVCFs and used. (Figure III-1.2) GATK4 didn't have the option to run fast by giving threads, but I checked that it runs fast using Java memory. (Heldenbrand et al., 2019)

Strelka2 has changed options when calling NA12878 and calling Korean data. When calling NA12878, the default option was used, but when calling Korean data, it was confirmed that the speed was too slow. Therefore, to solve this problem, I put a BED (Browser Extensible Data) file to speed it up. BED file was received from Strelka2's GitHub.

DeepVariant is a Docker program, and the other values excluding thread were used by default. Docker was installed only if you had a root account on Linux. After being installed by root, I took the docker image and installed DeepVariant. When you run DeepVariant, it runs as root in Linux, so be careful with the root memory.

The Samtools pipeline was created by referring to the paper with the code. (Cornish et al., 2015) The pipeline included GATK's Realignment, BaseRecalibrator, PrintReads, and Samtools Mpileup. Prior to Samtools Mpileup, GATK's Realign, BaseRecalibrate, and PrintReads steps were executed first, and all of the above steps were included in the Samtools pipeline time. I tried to use Samtools Mpileup, but in the manual (http://www.htslib.org/doc/samtools-mpileup.html), now Samtools Mpileup can generate VCF, but the function is deprecated and will be removed in the future. Because it was planned, I used the BCFtools Mpileup recommended in the manual. (Figure III-1.2)

II-4. Data for comparing results

To compare the performance of each pipeline for NA12878, I received NA12878.vcf from Illumina ftp, which I think is the correct answer for NA12878. Afterwards, hap.py was used to evaluate the performance of the eight pipelines.

For microarray chip data, GARD cohort data and the Korean Genome and Epidemiology Study (KoGES) Ansung, Ansan data were combined, and each Axiom probe ID was changed to rsID. It was then divided according to the MAF value of the data. After that, the chip data and the rsID of the VCF file were compared.

II-5. Performance measure of variant calling pipelines in NA12878

To compare the results of NA12878 WGS data using 8 pipelines and NA12878.vcf, I calculated Precision and Recall using False Positive (FP), False Negative (FN), and True Positive (TP). (Figure III-1.4)

FP : The variants resulting from the pipeline, but not in the gold standard NA12878.vcf

FN : There are variants in NA12878.vcf, the gold standard, but not the variants resulting from the pipeline.

TP : The variants resulting from the pipeline and also exist in the gold standard NA12878.vcf

Precision : TP/(TP + FP) , Proportion of variants resulting from the pipeline that are also present in the Gold standard

Recall : TP/(TP + FN) , Among the variants in the gold standard, the proportion of the resulting variants by the pipeline

II-6. Replace reference with Korean genome

Korean references were made in 2009 and 2015. In 2009, it was created based on the hg19 reference, and in 2015, a new technique was applied to integrate genome information of dozens of Koreans.

I mapped NA12878 data to Korean reference and even performed variant calling. Among the callers, the BaseRecalibrator or HaplotypeCaller of the GATK4 process requested a vcf file for the reference fasta. Therefore, the vcf file corresponding to the Korean reference was needed. The site that received the Korean reference fasta file did not have a vcf file (ftp://koref.biodisk.org/KOREF1.0.r20150820/), so I tried a liftover to change the vcf file using hg38 reference to Korean reference, but it did not work. In the GATK4 pipeline, because of this problem, the BaseRecalibrator process, which required the vcf file, was skipped, and in the HaplotypeCaller process, which was selectively needed, the vcf file did not need to be inserted, so the option was removed.

II-7. Concordance comparison with chip data

WGS analysis was performed on three Korean subjects. Each subject was sequenced using a PCR free library.

The microarray data used to compare the pipeline and subjects is the data created by the KoGES Ansung, Ansan data and GARD cohort data combined. This is a combination of two cohort study data to increase MAF (Minor Allele Frequency), and was used through the "—merge" option using plink.

The accuracy of the SNP matching of the microarray data and the VCF file generated through the variant call process was compared. As a method of comparing the matching accuracy, the microarray data and the SNP of the VCF file were compared with rsID, respectively. For VCF files, I used bcftools annotations to give the variant an rsID, for microarray data, I matched the Axiom probe ID in the Axiom annotation file to the Axiom probe ID in the bim file and attached the rsID to match the probe ID. The microarray data is composed of plink format (bed, bim, fam), and among them, the minor and major columns of the bim file were used to compare with ref and alt of the vcf file. Since the major of the microarray data may not be the ref of the VCF file, alleles of the major and reference (ref) coincide, and all the minor and alt (alternative) alleles match, and all variants corresponding to the opposite cases were selected. After that, among the cases where the rsID of the microarray data and the rsID of the VCF file match, the ratio and number that match the major and minor of the microarray data were used for comparison.

$$\frac{VCF's\ rsID\ and\ Microarray\ data's\ rsID\ match}{Match\ rs\ ID\ in\ two\ files} \times 100$$

II-8. Comparison of concordance with and without Markduplicate

To compare the SNP accuracy according to whether or not Markduplicate is applied, duplicates are removed using Picard's Markduplicates after the alignment process. First, the removal of duplicates from the NA12878 data and the non-removal of duplicates were compared with the gold standard NA12878.vcf data, and then applied to the Korean data. In the Korean data, as in II-6, it was checked whether the minor and major alleles of the microarray data match the ref and alt of the VCF file, respectively.

II-9. Primer design for validation sequencing

To compare the performance of each pipeline in Korean data more deeply, validation was used, and Sanger sequencing was performed as a validation method. Prior to Sanger sequencing, a primer preparation step was performed. In the primer production step, primers are made on both sides of the variant, and when production is completed, sequencing proceeds in one or both directions in the forward or reverse direction. In this case, some of the variants were in the repeat or poly region, and it was judged that sequencing was difficult, so I replaced them with other variants. When sequencing is in progress, if there are many repeats or poly sections, it is said that there are many cases where the other primer cannot be read and is stopped while sequencing is in progress. Therefore, it is recommended to change the variant in this case because it may be read incompletely.

II-10. Validation variant sets

Validation sets were created using variants that appear when compared to microarray data. Use these sets to see if the variants are definitely coming out using sanger sequencing.

The MAF (Minor Allele Frequency) value of the chip data was divided into 0.01 or less, 0.01 to 0.05 or less, 0.05 to 0.1 or less, 0.1 to 0.2 or less, and more than 0.2. And the chip data variants of the MAF range and the variants of the VCF file resulting from each pipeline were compared, and the variants were compared based on rsID.

By comparison, the matching variants were identified using the SNP Nexus to determine the frequency of the variants divided by MAF in East Asian and European in 1000 genome project.

150 variants for validation sequencing were selected through the following selection criteria.

- $|MAF_{Asian}\ -\ MAF_{European}| = \Delta\mathrm{MAF}$, ΔMAF in large order
- $|MAF_{Asian}(< 0.05)\ -\ MAF_{European}(> 0.05)|$ is in the largest order
- EAS and EUR frequency of SNP Nexus result file is None or variants not shown in the result
- When there is a large difference between the MAF value of Chip data (Korean) and the value of EAS frequency
- In case of common MAF (MAF> 0.05) in chip data, but not from all 6 pipelines in VCF file

- In case of rare MAF (MAF <0.05) in chip data, but not from all 6 pipelines in VCF file
- When comparing VCF files, the SNP of ALT is different.
  - ■ In the case of different SNPs of ALT in only one pipeline in 6 pipelines.
  - ■ In the case of different SNPs of ALT in two pipelines in 6 pipelines.
  - ■ If there are 2 SNPs in ALT out of 6 pipelines
- In the case of overlap among the Markduplicate variants that appear in both the combination of BWA and Caller and the combination of NovoAlign and Caller according to the presence or absence of Markduplicate

Sanger Sequencing was performed using 1ml of buffy coat to identify 150 variants for subject1 with the above criteria. 40ml of blood was drawn and the buffy coat was concentrated to 10ml and stored, and 1ml of the buffy coat was used.

III. RESULTS

III-1. Performance comparison of pipelines using NA12878 data

The workflow using NA12878 is summarized in Figure III-1.1. The NA12878 data sequenced with Novaseq was made into a VCF file through a combination of Aligners BWA and NovoAlign and 4 different Callers. These VCF files were compared with NA12878.vcf, which is known as the gold standard through hap.py.

When comparing the run time, there was a difference in the Aligner. BWA-mem took 54 minutes for reference fasta file indexing and 1 hour 35 minutes for read align. In NovoAlign, indexing was as short as 1 minute, while read align took 5 hours. In Aligner, the run time of BWA-mem was fast. (Figure III-1.3) The SAM file generated through the alignment process was converted to a bam file through the Samtools view, and it took about 6 minutes. The Bam file needs the process of sorting the reads before entering the calling stage, and it was sorted using Samtools sort. Samtools sort took 15 minutes when I gave the thread 64.

The time of the calling process, excluding the alignment time, was compared for each pipeline. Run time of Strelka2 was faster among callers. In particular, when the combination of BWA-mem and Strelka2 set the thread to 32, it took only 40 minutes. In GATK4, there was no significant change in the run time according to the thread, and in Strelka2 and DeepVariant, the run time was shorter as the thread became larger. The run time of Samtools was the longest, BWA-mem-Samtools took 39 hours 48 minutes, NovoAlign-Samtools took 43 hours 26 minutes. (Figure III-1.7)

Strelka2 had a difference in run time depending on the presence or absence of a BED file. When comparing only the speed of Strelka2, in the pipeline using BWA and Strelka2, the higher the number of threads without using the BED file, the faster it was. On the other hand, in the pipeline using NovoAlign and Strelka2, when the BED file was not used for the Strelka2 option, it was about 5-6 times slower than the BWA-Strelka2 combination. When the BED file is used for the Strelka2 option, BWA – Strelka2 is about 15 minutes faster, and the NovoAlign – Strelka2 combination is about 5 hours faster. (Figure III-1.8)

The performance was compared using the precision and recall of the SNP and INDEL in NA12878. Precision is the ratio of variants in the gold standard among the variants produced by the pipeline, and recall is the ratio of variants in the gold standard that are also released by the pipeline.(Figure III-1.4) Among variants, it was confirmed that there are many cases where the gold standard data and the variant from the pipeline match when mapped to NovoAlign. (Figure III-1.5) This is because the variants mapped and called by NovoAlign appear more than those by BWA-mem. (Table III-1.1) In SNP, the precision was high in BWA-Strelka2, and the recall was high in Novoalign-GATK4. In INDEL, precision was high in BWA-Strelka2, and recall was high in NovoAlign-DeepVariant. (Table III-1.1, Figure III-1.6)

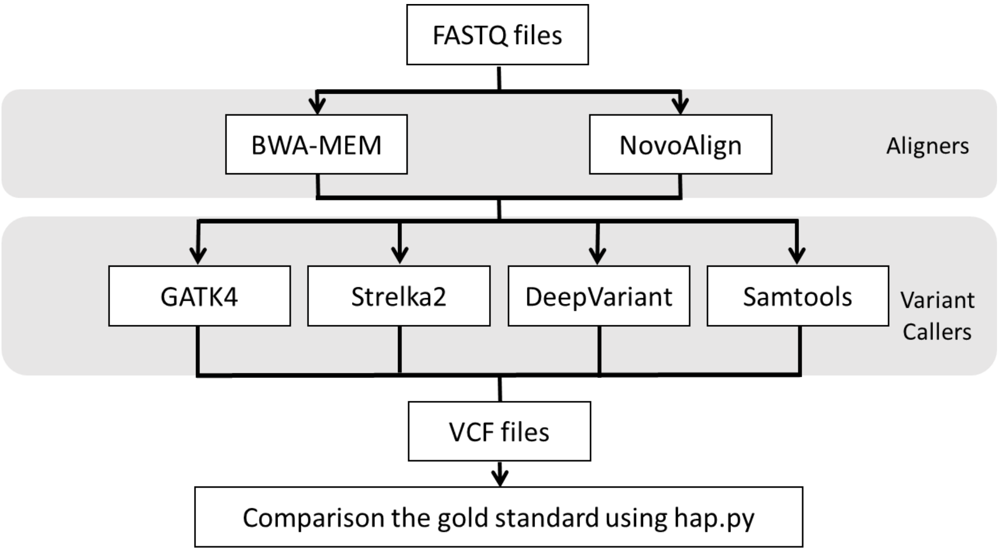

Figure III-1.1. Workflow performed for comparison from NA12878 data. Shows the process from FASTQ files to creating and comparing VCF files. Mapping aligner and variant caller calling variant are shown, and bam files created by mapping are created as vcf files through different variant callers. Compare the gold standard with the resulting vcf file from each pipeline through a python script called hap.py.

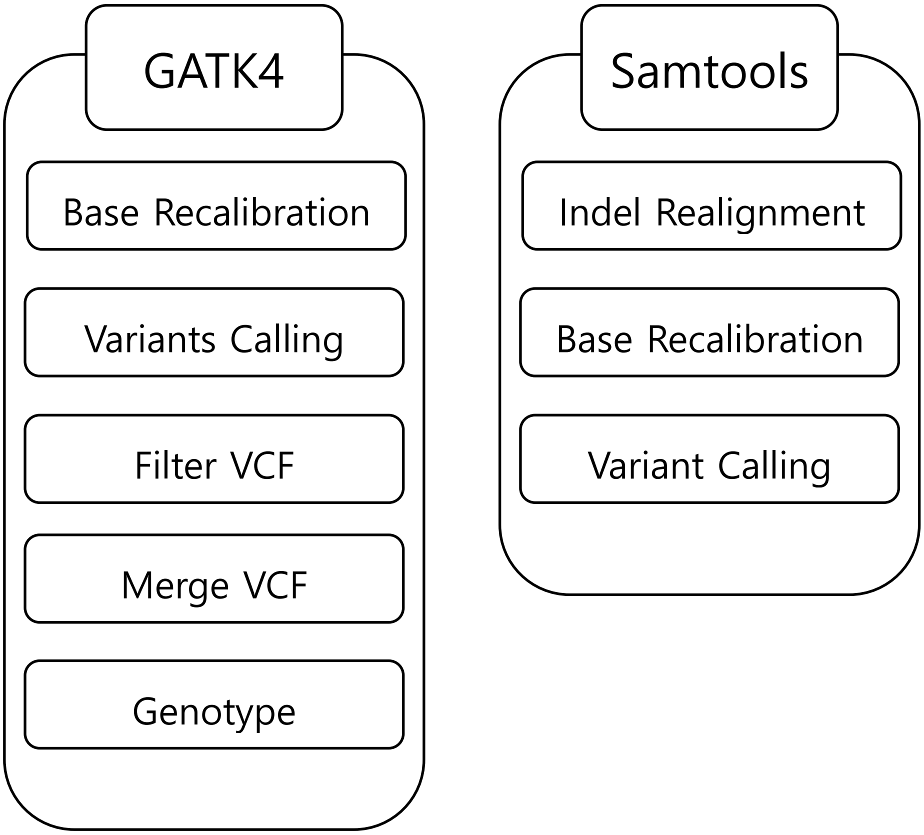

Figure III-1.2. GATK4 , Samtools detail workflow options. There are several processes in the analysis of GATK4. The base recalibration step, the haplotype caller step for variant calling, the step for filtering the quality of the created vcf, the step for recombining due to parallelization, and the step for genotype. Samtools uses the options of GATK3 until variant calling using Samtools mpileup. After using indel realignment and base recalibration, perform variant call.

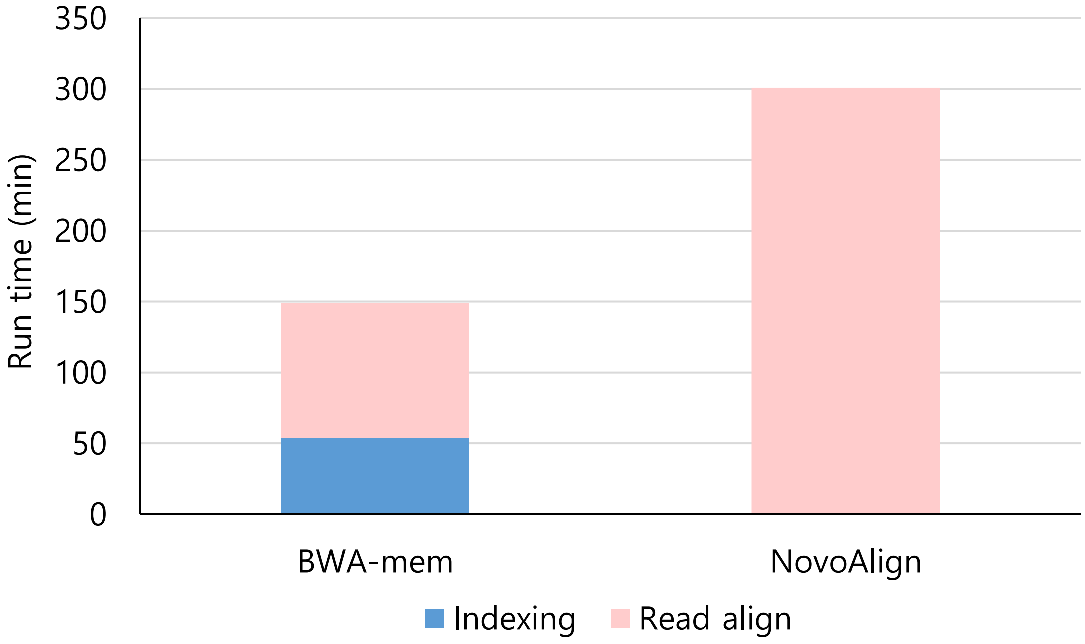

Figure III-1.3. Run time of Aligner. The x-axis represents the Aligner, and the y-axis represents the run time in minutes. Aligner is divided into the step of indexing the fasta file and the step of read aligning, and each time is indicated separately. (blue = Indexing, pink = Read align)

Table III-1.1. Performance of variants in NA12878 data

Aligner	Caller	SNP			INDEL		
		Number of variants	Precision	Recall	Number of variants	Precision	Recall
BWA-mem	GATK4	3,818,797	0.99444	0.965177	958,154	0.858638	0.906434
	Strelka2	3,620,081	0.998313	0.960672	771,602	0.965633	0.908309
	DeepVariant	3,718,909	0.997412	0.967126	1,005,570	0.875075	0.922773
	Samtools	3,710,308	0.996516	0.957156	593,501	0.725275	0.550751
NovoAlign	GATK4	4,006,335	0.992041	0.989193	1,020,127	0.855729	0.925955
	Strelka2	3,754,790	0.996569	0.977837	790,155	0.964295	0.920587
	DeepVariant	3,818,663	0.924673	0.981983	1,034,230	0.530872	0.936176
	Samtools	3,850,381	0.995963	0.981362	655,763	0.72195	0.579427

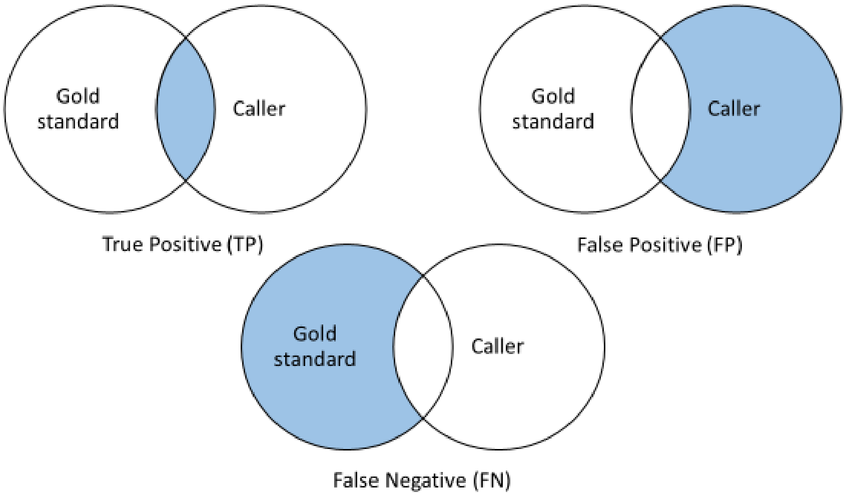

Figure III-1.4. Definition for performance evaluation. The definition used to evaluate performance is expressed as a Venn diagram. The colored areas represent True Positive, False Positive, and False Negative, respectively.

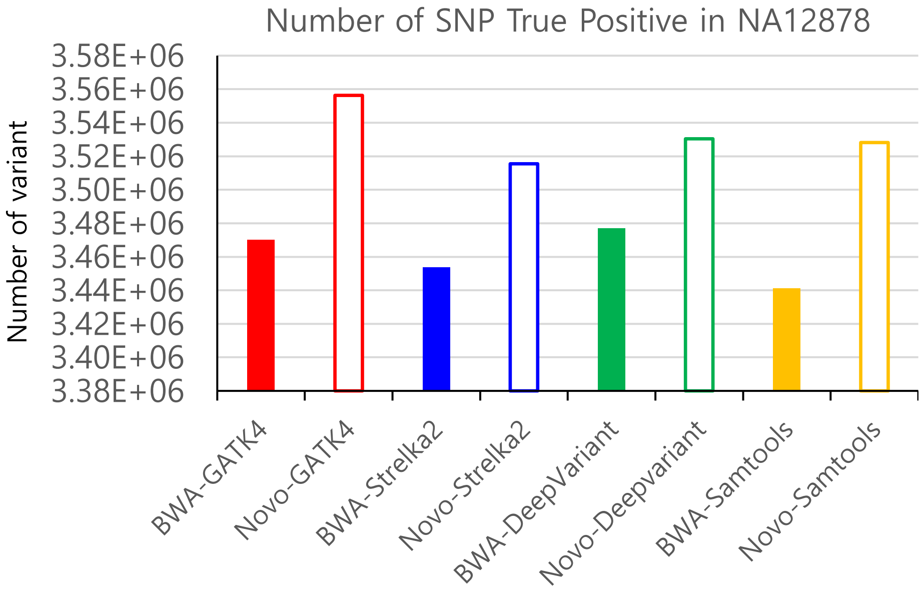

Figure III-1.5. Number of SNP True Positive in NA12878 data. The x-axis represents the pipeline that combines Aligner and Callers, and the y-axis represents the number of variants belonging to true positives when comparing SNPs. If the color is filled, it is mapped with BWA-mem, and if it is not filled, it is mapped with NovoAlign. Each color represents a caller, red for GATK4, blue for Strelka2, green for DeepVariant, and yellow for Samtools.

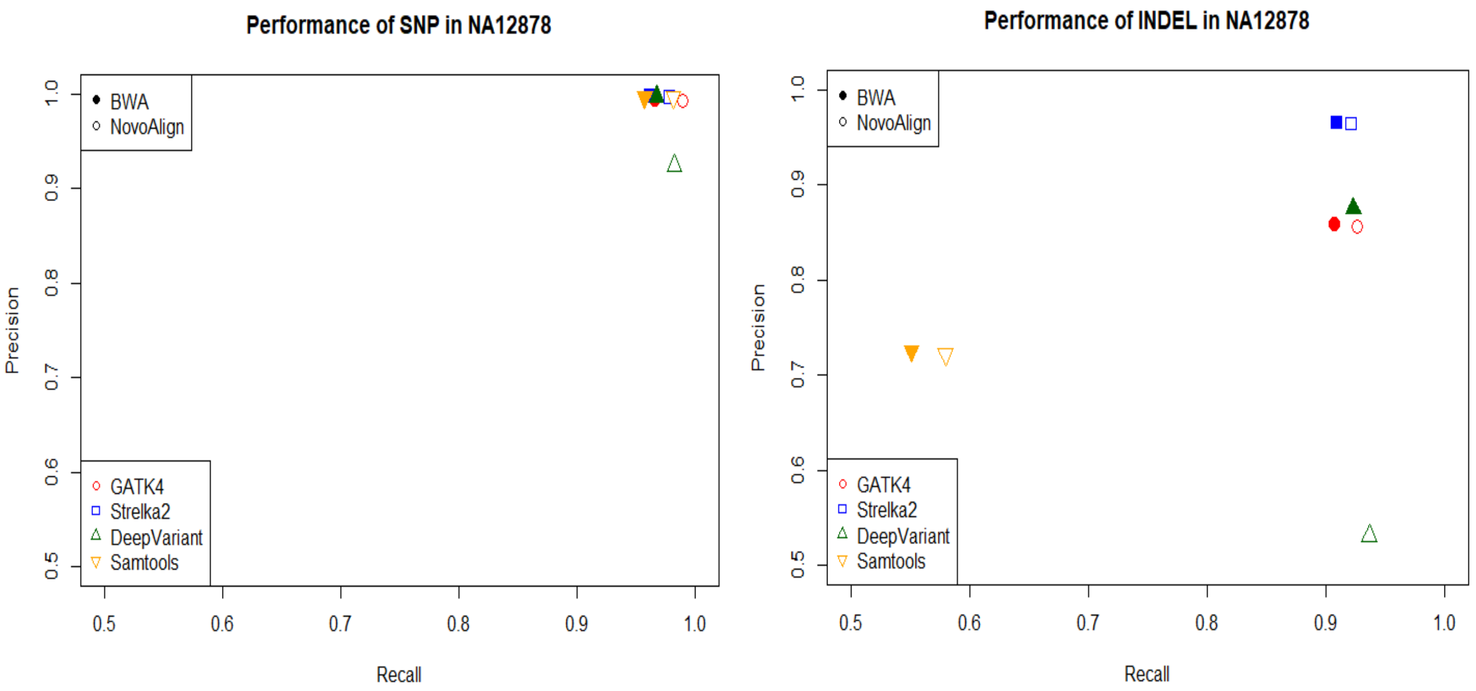

Figure III-1.6. Performance in each pipelines for NA12878 data. The x-axis is recall, and the y-axis is precision. If the figure is filled with color, BWA-mem is used among Aligners, and if there is only a border, NovoAlign is used. Each caller has a different shape and color. Red is GATK4, blue is Strelka2, green is DeepVariant, and yellow is Samtools.

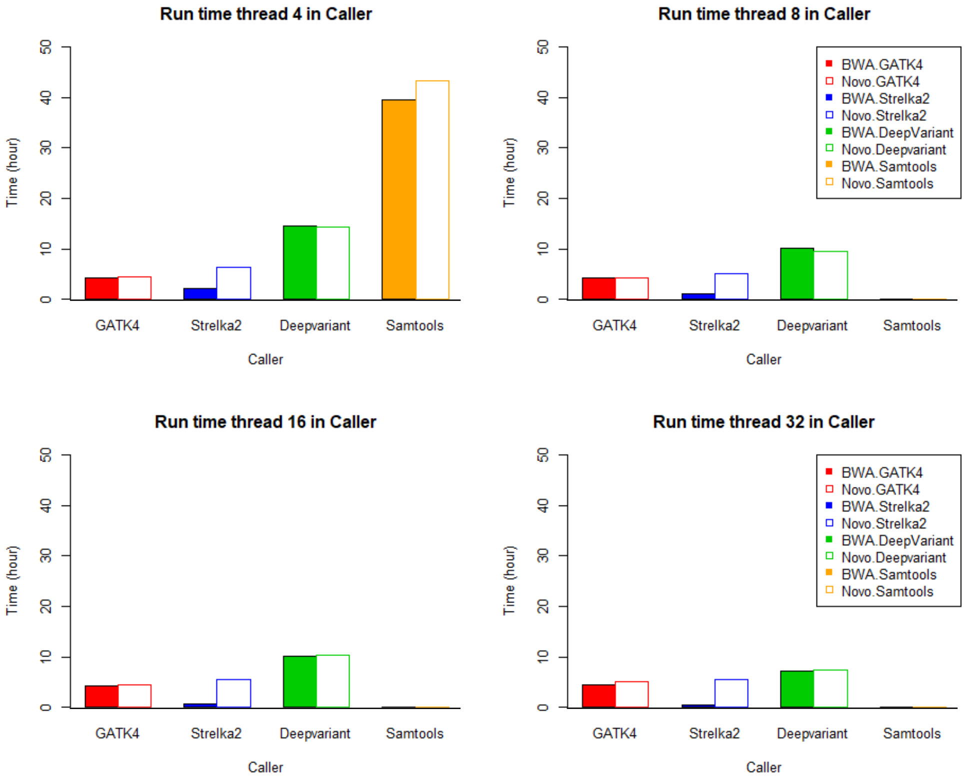

Figure III-1.7. Run time per thread in Caller. The x-axis represents the caller, the y-axis represents the caller's runtime, and the unit is hour. When the color is filled, the bam file obtained by mapping with BWA-mem is called, and when the color is not filled, it is the run time when the bam file obtained by mapping by NovoAlign is called. Colors were different for each caller to compare, red for GATK4, blue for Strelka2, green for DeepVariant, and yellow for Samtools.

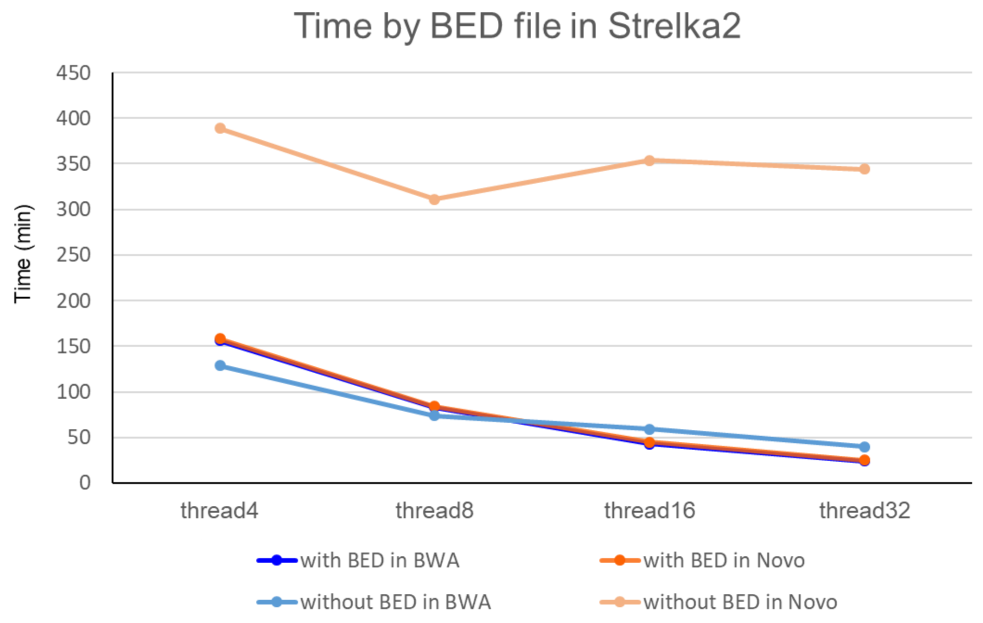

Figure III-1.8. Strelka2's run time for the use of BED file. The x-axis is the thread and the y-axis are the run time, and the unit of this y-axis is min. Blue when calling with a bam mapped with BWA-mem, and orange when calling with a bam mapped with NovoAlign. If the color is dark, the BED file is inserted in the Strelka2 option, and if the color is light, the BED file is not inserted as an option.

III-2. Performance when using Korean reference

First, alignment and variant calls were performed using the Korean reference genome for NA12878 data. After that, the performance was compared with the gold standard NA12878.vcf. The pipeline used for this analysis used a combination of BWA-mem and Strelka2 which had short run times. As a result, SNP recall was 0.490368 and SNP precision was 0.936296, INDEL recall was 0.425591 and INDEL precision was 0.388198. (Table III-2.1)

Second, for Korean sequencing data, alignment and variant calls were performed using Korean reference. A pipeline of the combination of BWA-mem and strelka2 was used, and the resulting variant was compared with that of the Korean microarray chip. Looking at the number of exact matches, it can be seen that the number of calls using the standard reference genome is more than twice as many as calling using the Korean reference genome. (Table III-2.2)

Table III-2.1. Performance when using Korean reference for NA12878 data

Aligner	Caller	SNP			INDEL		
		Number of variant	Precision	Recall	Number of variant	Precision	Recall
BWA	Strelka2	3,595,172	0.936296	0.490368	572,728	0.388198	0.425591

Table III-2.2. Concordance when using Korean reference for Korean WGS data

Concordance					
Pipeline	**MAF<=0.01**	**0.01<MAF<=0.05**	**0.05<MAF<=0.1**	**0.1<MAF<=0.2**	**0.2 <MAF**
BWA–Strelka2	0	8,992 (94.04%)	10,011 (87.02%)	22,439 (81.165%)	78,418 (77.69%)

III-3. Performance comparison with and without Markduplicate step in NA12878 data

The TruSeq PCR-Free Library Prep Kit was used for Korean WGS data. Since the PCR process was omitted, I assumed that there were few duplicates and compared the procedure to remove the duplicates, with and without Markduplicate step.

When performing the Markduplicate step, Picard's Markduplicate was used, and the speed was also checked by changing the java option. I used the Java -Xmx8g and -Xms8g, -Xmx32g and -Xms32g options to see the difference in speed depending on the options. It was expected that the more memory used, the faster it would be, but the case of using "8g" of memory was faster. (Table III-3.1)

Markduplicate is a step that is performed after alignment and before calling, so if this step is excluded, you will be able to quickly create a VCF file. When I ran Markduplicate, it took 4 hours and 29 minutes for NovoAlign's output file and 3 hours and 53 minutes for BWA's output file.

Using the NA12878 data, I examined the performance according to with or without of the Markduplicate process. Looking at the performance according to the presence or absence of the Markduplicate process, it can be seen that there is no significant difference between precision and recall. (Table III-3.2 and Figure III-3.1)

Table III-3.1. Run time of Markduplicate according to java options

	Do Markduplicate	
Java options	**In BWA**	**In NovoAlign**
Java -Xmx 8g -Xms 8g	233 min	269 min
Java -Xmx 32g -Xms 32g	288 min	283 min

Table III-3.2. Performance according to Markduplicate in NA12878

Markduplicate	Aligner	Caller	SNP		INDEL	
			Precision	Recall	Precision	Recall
No Markduplicate	BWA	GATK4	0.993911	0.965275	0.852968	0.905847
		Strelka2	0.997969	0.96067	0.963374	0.908895
		DeepVariant	0.997326	0.967179	0.877585	0.923899
	NovoAlign	GATK4	0.990995	0.98929	0.849226	0.925462
		Strelka2	0.996392	0.978105	0.961986	0.921318
		DeepVariant	0.996924	0.982084	0.875412	0.937378
Markduplicate	BWA	GATK4	0.99444	0.965177	0.858638	0.906434
		Strelka2	0.998313	0.960672	0.965633	0.908309
		DeepVariant	0.997412	0.967126	0.875075	0.922773
	NovoAlign	GATK4	0.992041	0.989193	0.855729	0.925955
		Strelka2	0.996569	0.977837	0.964295	0.920587
		DeepVariant	0.924673	0.981983	0.530872	0.936176

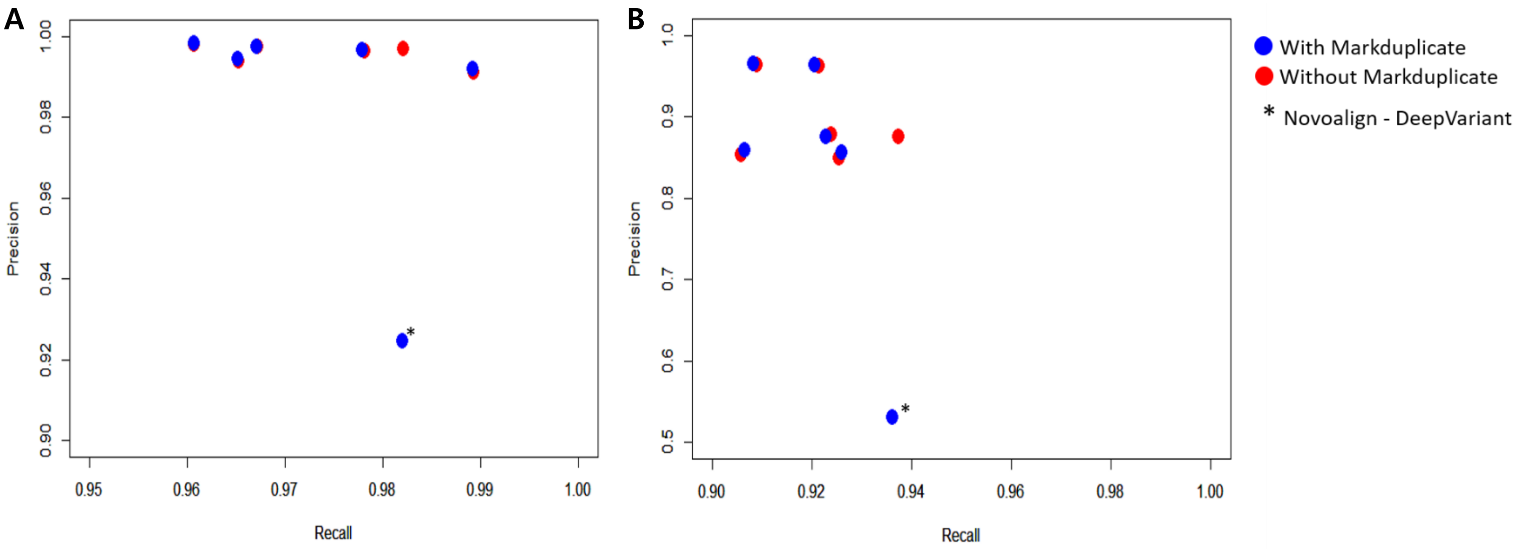

Figure III-3.1. Performance of pipelines with or without Markduplicate. The X-axis represents recall and the y-axis represents precision. The graph shows the performance of the pipeline according to the presence or absence of Markduplicate. The blue circle indicates the case where the Markduplicate process was executed, and the red circle indicates the case where the Markduplicate process was not executed. The performance when comparing SNP is (A), and the performance when comparing INDEL is (B). The NovoAlign-DeepVariant pipeline, which is a case of low precision, is marked separately with *.

III-4. Concordance comparison of pipelines using Korean data

Through the NA12878 data, callers to be used for Korean data analysis were selected. Korean data were analyzed using GATK4, Strelka2, and DeepVariant. (Figure III-4.1)

Three Korean subjects were analyzed using a pipeline selected through NA12878 data analysis.

In the analysis through the NA12878 data, it was confirmed in result III-3 that the difference in the number of variants appeared according to with or without Markduplicate was not significant. Therefore, it was analyzed that there is no difference in the number of variants in Korean data whether Markduplicate with or without. (Table III-4.1)

Table III-4.1 shows the results for one subject, and each column is divided based on the MAF value. The values in the table represent the number and percentage of the microarray chip data and variants of the VCF file matched across six pipelines. There was little difference in number and ratio.

The values in table III-4.1 represent the number of variants that match exactly. The ratio represents the ratio of exactly the same variant while having the same rsID. Looking at Figure III-4.2, which analyzed subjects, the combination of NovoAlign and GATK4 showed the highest ratio in all the ranges divided by MAF. Overall, when mapping with NovoAlign, many variants appeared. Although there is not much difference in the ratio for each pipeline, the combination of NovoAlign-GATK4 in the figure divided by MAF value has a high ratio in three subjects, so if you want to see many variants, it is suggested to select this combination. In addition, there was no difference in speed compared to the analysis of NA12878 data. So, if you want a quick analysis, I suggest a combination of BWA-mem and Strelka2.

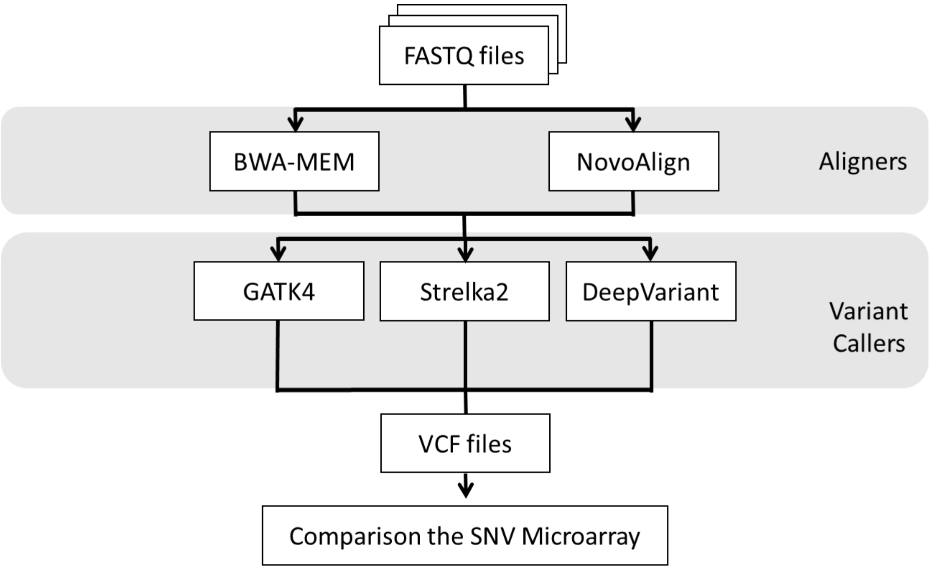

Figure III-4.1. Workflow performed for comparison from Korean data. Using three Korean people, the results were compared with the selected pipeline and the chip data obtained through the microarray. Above is a workflow for comparing Korean data.

Table III-4.1. Accuracy of variants in Subject1

Subject1						
Markduplicate	**Pipeline**	**MAF<=0.01**	**0.01<MAF<=0.05**	**0.05<MAF<=0.1**	**0.1<MAF<=0.2**	**0.2<MAF**
With Markduplicate	BWA–GATK4	1 (50%)	20,674 (83.93%)	21,928 (80.59%)	45,196 (78.05%)	151,430 (76.68%)
	NovoAlign–GATK4	1 (50%)	21,179 (84.09%)	22,461 (80.73%)	46,244 (78.20%)	155,257 (76.89%)
	BWA–Strelka2	1 (50%)	20,668 (83.94%)	21,927 (80.57%)	45,185 (78.06%)	151,350 (76.68%)
	NovoAlign–Strelka2	1 (50%)	21,034 (84.035%)	22,300 (80.66%)	45,829 (78.11%)	153,635 (76.75%)
	BWA–DeepVariant	1 (50%)	20,685 (83.94%)	21,932 (80.56%)	45,205 (78.05%)	151,449 (76.68%)
	NovoAlign–DeepVariant	1 (50%)	21,043 (84.03%)	22,311 (80.67%)	45,840 (78.112%)	153,689 (76.75%)
Without Markduplicate	BWA–GATK4	1 (50%)	20,676(83.94%)	21,933 (80.56%)	45,197 (78.05%)	151,425 (76.68%)
	NovoAlign-GATK4	1 (50%)	21,179 (84.09%)	22,463 (80.73%)	46,241 (78.20%)	155,250 (76.89%)
	BWA–Strelka2	1 (50%)	20,669 (83.94%)	21,930 (80.57%)	45,183 (78.06%)	151,337 (76.67%)
	NovoAlign–Strelka2	1 (50%)	21,035 (84.035%)	22,303 (80.67%)	45,828 (78.11%)	153,629 (76.74%)
	BWA–DeepVariant	1 (50%)	20,683 (83.94%)	21,933 (80.57%)	45,198 (78.05%)	151,414 (76.68%)
	NovoAlign–DeepVariant	1 (50%)	21,040 (84.03%)	22,309 (80.67%)	45,831 (78.115%)	153,646 (76.75%)

Table III–4.2. Accuracy of variants in Subject2

Subject2						
Markduplicate	**Pipeline**	**MAF<=0.01**	**0.01<MAF<=0.05**	**0.05<MAF<=0.1**	**0.1<MAF<=0.2**	**0.2<MAF**
Without Markduplicate	BWA–GATK4	1 (50%)	20,738 (84.04%)	21,852 (80.23%)	44,365 (77.76%)	151,947 (76.74%)
	NovoAlign-GATK4	1 (50%)	21,214 (84.155%)	22,410 (80.43%)	45,568 (77.925%)	156,292 (76.99%)
	BWA–Strelka2	1 (50%)	20,747 (84.05%)	21,874 (80.24%)	44,425 (77.77%)	152,138 (76.76%)
	NovoAlign–Strelka2	1 (50%)	21,061 (84.11%)	22,207 (80.33%)	45,117 (77.82%)	154,494 (76.85%)
	BWA–DeepVariant	1 (50%)	20,761 (84.06%)	21,881 (80.24%)	44,457 (77.77%)	152,211 (76.766%)
	NovoAlign–DeepVariant	1 (50%)	21,066 (84.11%)	22,211 (80.33%)	45,135 (77.82%)	154,530 (76.85%)

Table III-4.3. Accuracy of variants in Subject3

Subject3						
Markduplicate	**Pipeline**	**MAF<=0.01**	**0.01<MAF<=0.05**	**0.05<MAF<=0.1**	**0.1<MAF<=0.2**	**0.2<MAF**
Without Markduplicate	BWA–GATK4	1 (50%)	20,722 (84.198%)	21,890 (80.61%)	44,284 (77.88%)	151,413 (76.65%)
	NovoAlign-GATK4	1 (50%)	21,170 (84.31%)	22,503 (80.82%)	45,447 (78.07%)	155,435 (76.885%)
	BWA–Strelka2	1 (50%)	20,718 (84.206%)	21,888 (80.62%)	44,259 (77.89%)	151,327 (76.65%)
	NovoAlign–Strelka2	1 (50%)	21,031 (84.27%)	22,260 (80.69%)	45,022 (77.97%)	153,669 (76.73%)
	BWA–DeepVariant	1 (50%)	20,728 (84.21%)	21,890 (80.62%)	44,284 (77.89%)	151,409 (76.65%)
	NovoAlign–DeepVariant	1 (50%)	21,035 (84.27%)	22,256 (80.70%)	45,021 (77.97%)	153,690 (76.73%)

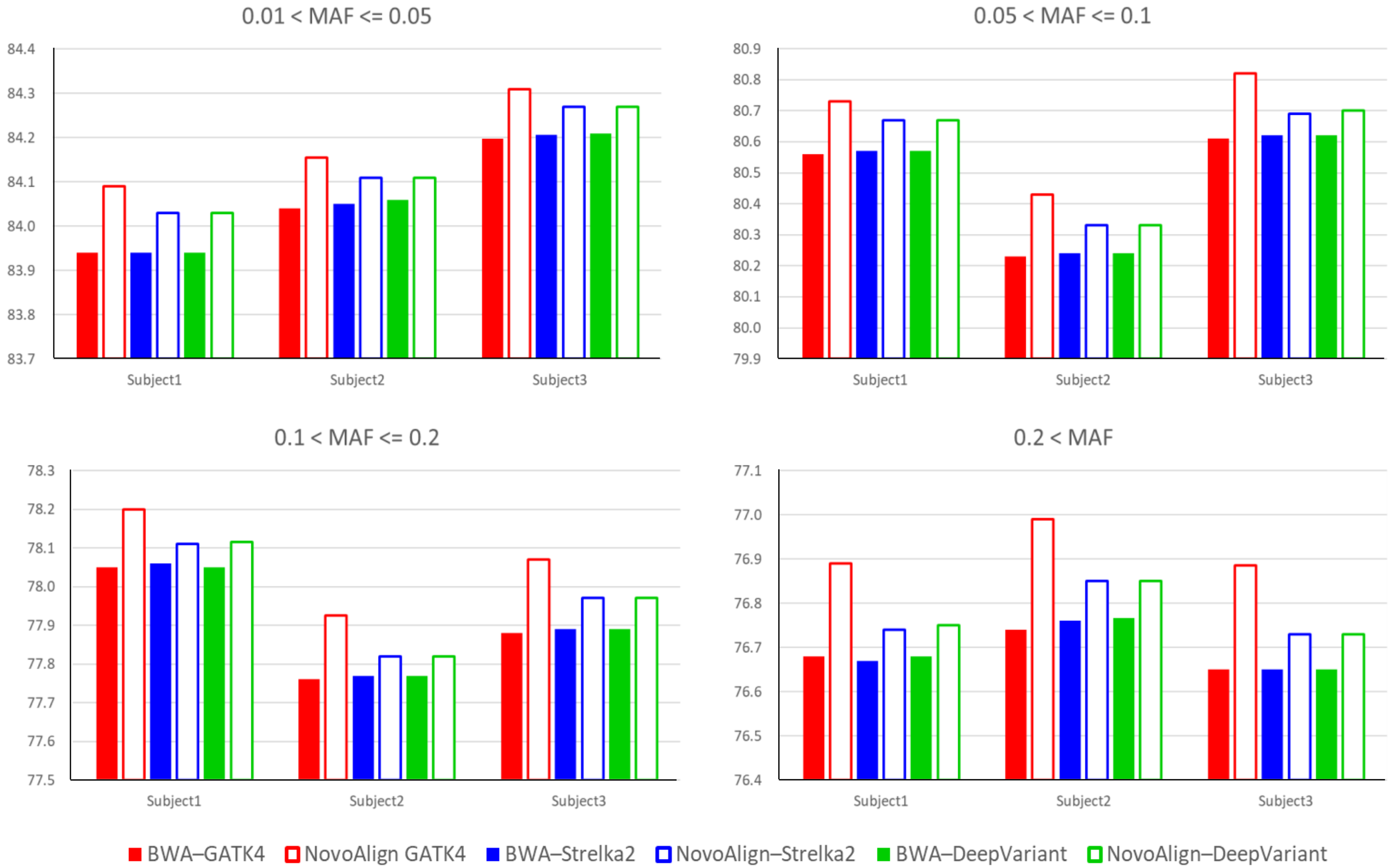
0.01 < MAF <= 0.05
84.4
84.3
84.2
84.1
84.0
83.9
83.8
83.7
Subject1
Subject2
Subject3
0.05 < MAF <= 0.1
80.9
80.8
80.7
80.6
80.5
80.4
80.3
80.2
80.1
80.0
79.9
Subject1
Subject2
Subject3
0.1 < MAF <= 0.2
78.3
78.2
78.1
78.0
77.9
77.8
77.7
77.6
77.5
Subject1
Subject2
Subject3
0.2 < MAF
77.1
77.0
76.9
76.8
76.7
76.6
76.5
76.4
Subject1
Subject2
Subject3
■ BWA–GATK4
□ NovoAlign GATK4
■ BWA–Strelka2
□ NovoAlign–Strelka2
■ BWA–DeepVariant
□ NovoAlign–DeepVariant

Figure III-4.2. Variant matching rate of pipeline and chip data by MAF range. Table III-4.1, Table III-4.2, Table III-4.3 as a graph. The x-axis is the subject, the y-axis is the coincidence rate between the chip data and the pipeline result data, and the unit is percent (%). The four graphs were divided according to the MAF range. Subject1 in the graph represents the match rate when the Markduplicate process was not performed. If the color is filled, it is mapped with BWA-mem, and if it is not filled, it is mapped with NovoAlign. Each color represents a caller, red for GATK4, blue for Strelka2, and green for DeepVariant.

IV. DISCUSSION

In the analysis using the well-known data, NA12878, the BWA-mem - Strelka2 combination was faster than the other seven pipelines. Looking at the speed by caller, Strelka2 was the fastest, followed by GATK4. Since there was no option to speed up GATK4, the speed was reduced by dividing it by chromosome, analyzing it in a parallel manner, and then combining it again. However, this has the disadvantage that it consumes a lot of memory or capacity of the server. DeepVariant needed a docker, and this docker had to have root privileges to install. Due to the nature of Docker, an image is created, but if it is not deleted, the server's capacity occupied from 20G to 30G. Samtools mpileup used Indel Realignment as a step to better call the variant at the front of the pipeline in this paper, and the run time of this process was long. Therefore, the overall time of the Samtools mpileup pipeline was the longest. If you use only Samtools mpileup (BCFtools mpileup) without using it like the pipeline in this paper, the speed is similar to that of DeepVariant.

Looking at Table III-1.1, the number of variants differed according to the Aligner. It can be seen that the number of variants to be called is more than 100,000 in the case of mapping from SNP to NovoAlign than the case of mapping with BWA-mem. For this reason, it could be said that when NovoAlign is used as an aligner, many variants are called. However, calling a lot of variants does not mean that the performance is good. In Result III-1, when mapping with NovoAlign, recall was higher than that of BWA-mem. On the other hand, when mapping with BWA-mem, the precision was higher. When comparing the case of the same caller, there was no significant difference in precision except for the NovoAlign-DeepVariant mentioned earlier. On the other hand, there was a difference in recall.

By combining various results, the Samtools pipeline, which was slow and the INDEL performance was poor, was excluded from the Korean data analysis pipeline. Table III-

1.1 shows that the precision of NovoAlign-DeepVariant is low in INDEL as well as Samtools. Therefore, I tried to exclude this pipeline with Samtools, but the recall was higher in SNP and INDEL than Samtools, and the pipeline was changed according to the presence or absence of Markduplicate, so I used it for Korean data analysis to study more. The reason I chose a pipeline with high recall is the case that there is a variant from the pipeline that I performed among the variants in the gold standard data. Precision differs from recall and perspective because the variant from the pipeline is also in the gold standard. The reason why NovoAlign-DeepVariant was selected based on recall in this paper was because it focused on how well the variant, which is the correct answer, was found.

As mentioned in Introduction I-4, in humans, 99.9% of the bases in the whole genome are similar, and it is 0.1% that makes the difference. Due to racial differences, 15% of SNPs are different, and Korean references were made due to this problem. However, when comparing, the number of matching variants was higher in the case of using the existing standard reference genome than in the case of using the Korean reference. Therefore, even if Korean sequencing data is used, it seems to be better to use the standard reference genome than the Korean reference genome.

In the study on the presence or absence of Markduplicate, there was no difference in most pipelines when using NA12878 data. There was a big change only in the case of NovoAlign-DeepVariant, but surprisingly, it was confirmed that the performance improved significantly when without Markduplicate than when with Markduplicate. However, as can be seen in Table III-4.1, in Korean data, not only NovoAlign-DeepVariant, but also most of the pipelines showed little change in the number of variants according to Markduplicate. Since there is no perfect answer, there may be a limit to comparing only the variant of the chip data, but it was possible to compare by finding the variants in the corresponding range using the minor allele frequency.

As mentioned in Result III-4, there is no significant difference. However, among the pipeline combinations, the number of variants that matched the NovoAlign and GATK4 combinations was the highest from the data of all Koreans. In conclusion, if you want to check variants quickly, use the BWA-mem and Strelka2 combination, and if you want to detect many variants and use a pipeline that is highly concordance with chip data, use the NovoAlign and GATK4 combination.

V. 초 록

개인 유전체 분석 파이프라인의 체계적 비교연구를 통한 인종 특이적 분석법 구축

박현슬
지도교수 : 김 정 수
글로벌바이오융합학과
조선대학교 대학원

차세대 염기서열 분석법(Next Generation Sequencing, NGS)라고 불리는 시퀀싱 방법은 13년이 걸렸던 인간 게놈 프로젝트(Human Genome Project, HGP)에서 사용한 생어 시퀀싱(sanger sequencing) 기법과 달리 빠르고 저렴한 비용으로 인간 유전체를 읽을 수 있는 분석법이다.

NGS는 조각 낸 DNA를 중합효소 연쇄 반응(PCR)을 통해 증폭시키고 충분한 양이 된 서열들을 시퀀싱 기법을 통해 읽게 된다. 읽은 염기서열을 “reads” 라고 부르며 크게 3단계의 분석 단계를 거친다. 읽은 서열의 퀄리티를 나타내고, 퀄리티가 높은 reads만을 선별하는 품질관리(Quality Control, QC) 단계와 선별된 reads를 참조 유전체에 맵핑 시켜주는 정렬(alignment) 단계, 그리고 맵핑 된 reads와 참조 유전체에서 차이가 나는 염기를 추출해주는 변이 검출(variant call) 단계로 분석이 진행된다. 정렬과

변이 검출 단계는 사용할 수 있는 많은 방법들이 존재하며, 이 방법들은 저마다 다른 알고리즘을 가지고 있다.

정렬과 변이 검출 단계에서 사용되는 참조 유전체는 서양인을 기반으로 만들어져 있으며, NGS 데이터 분석 방법의 성능에 대한 여러 연구들도 “NA12878” 서양인 데이터를 사용하였다. 하지만 서양인에게 최적화 되어있는 방법들이 유전적 차이가 존재하는 동양인에서도 같은 결과를 보일지에 대해 의문이 제기되었다.

본 연구에서는 NGS를 이용한 전장유전체 시퀀싱(Whole Genome Sequencing, WGS) 데이터를 이용하여 각 방법의 성능 차이에 대한 분석을 진행하였다. 비교에 사용된 WGS 데이터는 서양인(NA12878) 데이터와 한국인 데이터이며 NA12878 데이터의 경우에는 알려진 정답(gold standard)과 성능을 비교하였고, 한국인 데이터의 경우에는 한국인을 대상으로 한 마이크로어레이 칩 데이터를 사용하여 비교했다.

다양한 방법들 중 이전 보고에서 성능이 좋았던 2개의 정렬 방법(BWA-mem, NovoAlign)과 4개의 변이 검출 방법(GATK4, Strelka2, DeepVariant, Samtools)을 선택하여 8개의 분석 가능한 조합을 비교했다.

NA12878을 분석한 결과 Samtools로 변이 검출을 했을 때 소요시간이 길었고, 재현율이 상대적으로 떨어졌다. 소요시간을 비교하면 BWA-mem과 Strelka2 조합의 소요시간이 가장 짧았다. 성능을 평가하는 기준으로 삼았던 정밀도와 재현율에서 단일 염기 다형성(Single Nucleotide Polymorphism, SNP)은 BWA-mem 과 Strelka2의 조합이 가장 높았고, 재현율은 NovoAlign과 GATK4의 조합이 가장 높았다. 염기의 삽입과 결실 돌연변이(Insertion Deletion, INDEL) 정밀도는 BWA-mem과 Strelka2 조합에서 가장 높았고,

재현율은 NovoAlign과 DeepVariant 조합에서 가장 높았다. 따라서 한국인 유전체를 분석하는 조합으로는 변이 검출 단계에서 Samtools를 제외한 조합으로 분석을 진행하였다.

정렬과 변이 검출 과정 사이에 중복되는 reads를 표시하는 단계(Markduplicate)에 의해 소요시간의 차이가 생겼다. 한국인 시퀀싱 데이터는 PCR 단계를 수행하지 않았기 때문에 Markduplicate 단계를 수행하지 않은 경우와 수행한 경우의 성능을 비교했을 때, 큰 차이가 나지 않았다. 따라서 PCR 단계를 수행하지 않는 경우에는 Markduplicate 과정을 하지 않아도 된다.

한국인 시퀀싱 데이터를 사용하여 각 파이프라인 결과로 나오는 변이와 칩 데이터의 변이를 비교했을 때, 파이프라인 별로 일치하는 변이의 개수 차이가 크게 나지 않았다. 하지만 파이프라인 조합들 중에서도 모든 한국인 데이터에서 NovoAlign과 GATK4의 조합이 일치하는 변이의 개수가 가장 많았다. 결론적으로 빠른 속도로 변이를 확인하고 싶다면 BWA-mem과 Strelka2 조합을 사용하고, 많은 변이를 검출하며 칩 데이터와의 일치성이 높은 파이프라인을 사용하고 싶다면 NovoAlign과 GATK4 조합을 사용하는 것이 좋다.

기존의 참조 유전체 대신 한국인 참조 유전체를 사용한 경우 BWA-mem과 Strelka2 조합만을 봤을 때, SNP의 재현율과 INDEL의 정밀도, 재현율 값이 낮았고, 칩 데이터와 일치되는 변이가 적었다. 따라서 한국인 시퀀싱 데이터를 사용하더라도 기존의 참조 유전체를 사용하여 정렬과 변이 검출을 하는 것이 더 성능이 좋다는 결론이 도출되었다.

VI. REFERENCES

Chen, J., Li, X., Zhong, H., Meng, Y., & Du, H. (2019). Systematic comparison of germline variant calling pipelines cross multiple next-generation sequencers. *Sci Rep, 9*(1), 9345. doi:10.1038/s41598-019-45835-3

Chen, X., Dong, Y., Huang, Y., Fan, J., Yang, M., & Zhang, J. (2021). Whole-genome resequencing using next-generation and Nanopore sequencing for molecular characterization of T-DNA integration in transgenic poplar 741. *BMC Genomics, 22*(1), 329. doi:10.1186/s12864-021-07625-y

Cho, Y. S., Kim, H., Kim, H. M., Jho, S., Jun, J., Lee, Y. J., . . . Bhak, J. (2016). An ethnically relevant consensus Korean reference genome is a step towards personal reference genomes. *Nat Commun, 7*, 13637. doi:10.1038/ncomms13637

Collins, F. S., Morgan, M., & Patrinos, A. (2003). The Human Genome Project: lessons from large-scale biology. *Science, 300*(5617), 286-290. doi:10.1126/science.1084564

Cornish, A., & Guda, C. (2015). A Comparison of Variant Calling Pipelines Using Genome in a Bottle as a Reference. *Biomed Res Int, 2015*, 456479. doi:10.1155/2015/456479

Heldenbrand, J. R., Baheti, S., Bockol, M. A., Drucker, T. M., Hart, S. N., Hudson, M. E., . . . Mainzer, L. S. (2019). Recommendations for performance optimizations when using GATK3.8 and GATK4. *BMC Bioinformatics, 20*(1), 557. doi:10.1186/s12859-019-3169-7

Huang, T., Shu, Y., & Cai, Y. D. (2015). Genetic differences among ethnic groups. *BMC Genomics, 16*, 1093. doi:10.1186/s12864-015-2328-0

Hwang, K. B., Lee, I. H., Li, H., Won, D. G., Hernandez-Ferrer, C., Negron, J. A., & Kong, S. W. (2019). Comparative analysis of whole-genome sequencing pipelines to minimize false negative findings. *Sci Rep, 9*(1), 3219. doi:10.1038/s41598-019-39108-2

Hwang, S., Kim, E., Lee, I., & Marcotte, E. M. (2015). Systematic comparison of variant calling pipelines using gold standard personal exome variants. *Sci Rep, 5*, 17875. doi:10.1038/srep17875

Jeon, S., Bhak, Y., Choi, Y., Jeon, Y., Kim, S., Jang, J., . . . Bhak, J. (2020). Korean Genome Project: 1094 Korean personal genomes with clinical information. *Sci Adv, 6*(22), eaaz7835. doi:10.1126/sciadv.aaz7835

Jeon, Y., Jeon, S., Blazyte, A., Kim, Y. J., Lee, J. J., Bhak, Y., . . . Bhak, J. (2021). Welfare Genome Project: A Participatory Korean Personal Genome Project With Free Health Check-Up and Genetic Report Followed by Counseling. *Front Genet, 12*, 633731. doi:10.3389/fgene.2021.633731

Kim, Y., Han, B. G., & Ko, G. E. S. g. (2017). Cohort Profile: The Korean Genome and Epidemiology Study (KoGES) Consortium. *Int J Epidemiol, 46*(2), e20. doi:10.1093/ije/dyv316

Kishikawa, T., Momozawa, Y., Ozeki, T., Mushiroda, T., Inohara, H., Kamatani, Y., . . . Okada, Y. (2019). Empirical evaluation of variant calling accuracy using ultra-deep whole-genome sequencing data. *Sci Rep, 9*(1), 1784. doi:10.1038/s41598-018-38346-0

Kumaran, M., Subramanian, U., & Devarajan, B. (2019). Performance assessment of variant calling pipelines using human whole exome sequencing and simulated data. *BMC Bioinformatics, 20*(1), 342. doi:10.1186/s12859-019-2928-9

Li, H., & Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics, 25*(14), 1754-1760. doi:10.1093/bioinformatics/btp324

Miller, J. R., Koren, S., & Sutton, G. (2010). Assembly algorithms for next-generation sequencing data. *Genomics, 95*(6), 315-327. doi:10.1016/j.ygeno.2010.03.001

Park, S. T., & Kim, J. (2016). Trends in Next-Generation Sequencing and a New Era for Whole Genome Sequencing. *International Neurourology Journal, 20*, 76-83. doi:10.5213/inj.1632742.371

Roepman, P., de Bruijn, E., van Lieshout, S., Schoenmaker, L., Boelens, M. C., Dubbink, H. J., . . . Cuppen, E. (2021). Clinical validation of Whole Genome Sequencing for cancer diagnostics. *J Mol Diagn*. doi:10.1016/j.jmoldx.2021.04.011

Salavert, J., Tomas, A., Tarraga, J., Medina, I., Dopazo, J., & Blanquer, I. (2015). Fast inexact mapping using advanced tree exploration on backward search methods. *BMC Bioinformatics, 16*, 18. doi:10.1186/s12859-014-0438-3

Supernat, A., Vidarsson, O. V., Steen, V. M., & Stokowy, T. (2018). Comparison of three variant callers for human whole genome sequencing. *Sci Rep, 8*(1), 17851. doi:10.1038/s41598-018-36177-7

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., . . . Zhu, X. (2001). The sequence of the human genome. *Science, 291*(5507), 1304-1351. doi:10.1126/science.1058040

Wong, L. P., Ong, R. T. H., Poh, W. T., Liu, X., Chen, P., Li, R., . . . Teo, Y. Y. (2013). Deep Whole-Genome Sequencing of 100 Southeast Asian Malays. *American Journal of Human Genetics, 92*(1), 52-66. doi:10.1016/j.ajhg.2012.12.005

Zhang, J., Chiodini, R., Badr, A., & Zhang, G. (2011). The impact of next-generation sequencing on genomics. *J Genet Genomics, 38*(3), 95-109. doi:10.1016/j.jgg.2011.02.003

국립보건연구원. (2018). 한국인칩사업 백서.

식품의약품안전평가원. (2019). Next Generation Sequencing 기반 유전자 검사의 이해.

VII. APPENDIX

Share the code I used. This code is also available at https://github.com/BIjoy92/WGS.

- Aligner : BWA-mem

```
#!/bin/bash
-
-   source hg38_pipeline.cfg
-
-   bwa index ${ref_fasta}
-
-   bwa mem ${ref_fasta} $1_R1.fastq.gz $1_R2.fastq.gz -R
    '@RG\tID:'$2'\tSM:'$2'\tPL:ILLUMINA' -t 64 >
    ${BWA_output}/$2_BWA.sam
-
-   samtools view -@ 64 -Sb ${BWA_output}/$2_BWA.sam >
    ${View_path}/$2_BWA.bam
-
-   samtools sort -@ 64 ${View_path}/$2_BWA.bam -o
    ${View_path}/$2_BWA_sort.bam
-
-   samtools index -@ 64 ${View_path}/$2_BWA_sort.bam
```

- Aigner : NovoAlign

```
-   #!/bin/bash
-
-   source hg38_pipeline.cfg
-
-   novo_path='Novoalign/novocraft'
-
-   ${novo_path}/novoindex ${ref_fasta_path}/
    hg38_v0_Homo_sapiens_assembly38.nix ${ref_fasta}
-
-   ${novo_path}/novoalign -d
    ${ref_fasta_path}/hg38_v0_Homo_sapiens_assembly38.nix -f
    $1_S1_L001_R1_001.fastq.gz $1_S1_L001_R2_001.fastq.gz -o
    SAM $'@RG\tID:Sub1\tPL:illumina\tSM:Sub1' >
    $Novo_output/$1_Novo.sam
```

```
-
-   samtools view -@ 64 -Sb ${Novo_output}/$1_Novo.sam >
    ${View_path}/$3_$1_Novo.bam
-   samtools sort -@ 64 ${View_path}/$3_$1_Novo.bam -o
    ${View_path}/$3_$1_Novo_sort.bam
-   samtools index -@ 64 ${View_path}/$3_$1_Novo_sort.bam
```

- Caller : GATK4

```
-   #!/bin/bash
-
-   source hg38_pipeline.cfg
-
-   ## BaseRecalibrator
-
-   baserecal_start=`date +%Y.%m.%d.%H:%M`
-
-   list="1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
    21 22 X Y M"
-
-   for i in $list
-   do
-
-
    outfile=$GATK4_path/1.BQSR/$3_$1_$2_recal_data_$i.table
-          gatk --java-options "-Xmx8G -XX:+UseParallelGC -
    XX:ParallelGCThreads=8 -XX:-UsePerfData" BaseRecalibrator
    \
-    -R $ref_fasta \
-    -I ${View_path}/$1_$2_sort.bam \
-    -O $outfile \
-    -L chr$i \
-    --known-sites ${dbsnp_vcf} \
-    --known-sites ${known_indels} \
-    --known-sites ${Mills_indels} \
-    --tmp-dir ${tmp_dir} &
-
-   done
-
-   wait
-
-   list="1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
    21 22 X Y M"
```

```
-
-   for i in $list
-   do
-           gatk --java-options "-Xmx8G -XX:-UsePerfData"
    GatherBQSRReports \
-           -I $GATK4_path/1.BQSR/$3_$1_$2_recal_data_$i.table
    \
-           -O
    $GATK4_path/1.BQSR/$3_$1_$2_GatherBQSR_output.table
-   done
-
-   mkdir $GATK4_path/2.ApplyBQSR
-
-   rm -rf $GATK4_path/1.BQSR/$3_$1_$2_recal_data_*.table
-
-   for i in $list
-   do
-          ## Apply Base Quality Score Recalibration model
-
-           gatk --java-options "-Xmx8G -XX:-UsePerfData"
    ApplyBQSR \
-    -R $ref_fasta \
-    -I ${View_path}/$1_$2_sort.bam \
-    -L chr$i \
-    -bqsr-recal-file
    $GATK4_path/1.BQSR/$3_$1_$2_GatherBQSR_output.table  \
-    -O $GATK4_path/2.ApplyBQSR/$3_$1_$2_BQSR_chr$i.bam &
-
-   done
-
-   wait
-
-   mkdir $GATK4_path/3.GatherBam
-
-   find -name "$3_$1_$2_BQSR_chr*.bam" > $3_$1_$2_bqsr.list
-
-   java -Xmx8g -Djava.io.tmpdir=$tmp_dir -jar
    $picard/picard.jar GatherBamFiles I=$3_$1_$2_bqsr.list
    O=$GATK4_path/3.GatherBam/$3_$1_$2_gather_BAM.bam
-
-   samtools index -@ 64
    $GATK4_path/3.GatherBam/$3_$1_$2_gather_BAM.bam
-
-   ## Haplotype Caller
-
-   mkdir $GATK4_path/4.GATK4_HaploCall
```

```
-
-   list="1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 X Y M"
-
-   for i in $list
-
-   do
-
-         gatk --java-options "-Xmx8g -XX:-UsePerfData" HaplotypeCaller -R $ref_fasta --dbsnp ${dbsnp_vcf} -I $GATK4_path/3.GatherBam/$3_$1_$2_gather_BAM.bam -O $GATK4_path/4.GATK4_HaploCall/$3_$1_$2_chr$i.g.vcf.gz -L chr$i -ERC GVCF -ip 100 --pcr-indel-model NONE -G StandardAnnotation -G AS_StandardAnnotation --RF OverclippedReadFilter --filter-too-short 25 --max-alternate-alleles 3 --pairHMM AVX_LOGLESS_CACHING_OMP --native-pair-hmm-threads 8 --tmp-dir $tmp_dir &
-
-   done
-
-   wait
-
-   ## filter VCF
-
-   mkdir $GATK4_path/5.FilterVCF
-
-   list="1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 X Y M"
-
-   for i in $list
-   do
-
-         java -Xmx8g -Xms8g -Djava.io.tmpdir=$tmp_dir -jar $picard/picard.jar FilterVcf I=$GATK4_path/4.GATK4_HaploCall/$3_$1_$2_chr$i.g.vcf.gz O=$GATK4_path/5.FilterVCF/$3_$1_$2_filter_chr$i.g.vcf.gz &
-
-   done
-
-   wait
-
-   ## Merge VCF
-
-   find -name "$3_$1_$2_filter_chr*.g.vcf.gz" > $GATK4_path/6.MergeVCF/$3_$1_$2_vcf.list
-
```

```
merge_start=`date +%Y.%m.%d.%H:%M
`
java -Xmx8g -Xms8g -jar $picard/picard.jar MergeVcfs \
 I=$GATK4_path/6.MergeVCF/$3_$1_$2_vcf.list \
 O=$GATK4_path/6.MergeVCF/$3_$1_$2_mergeVCF.g.vcf.gz

## Genotype gvcf

mkdir $GATK4_path/7.Genotype

gvcf_start=`date +%Y.%m.%d.%H:%M`

gatk --java-options "-Xmx8g -XX:-UsePerfData"
GenotypeGVCFs \
 -R $ref_fasta \
 -V $GATK4_path/6.MergeVCF/$3_$1_$2_mergeVCF.g.vcf.gz \
 -O $GATK4_path/7.Genotype/$3_$1_$2_gvcf.vcf.gz \
 --tmp-dir $tmp_dir

tabix -p vcf $GATK4_path/7.Genotype/$3_$1_$2_gvcf.vcf.gz
```

- Caller : Strelka2

```
#!/bin/bash

source hg38_pipeline.cfg

BED_path=WGS/
mkdir ${strelka2_path}/$3_$1_$2_output

call_start=`date +%Y.%m.%d.%H:%M`

configureStrelkaGermlineWorkflow.py \
--bam ${View_path}/$1_$2_sort.bam \
--callRegions ${BED_path}/without_decoy.hg38.bed.gz \
--referenceFasta ${ref_fasta} \
--runDir
${strelka2_path}/$3_$1_$2_output/strelkaGermlineWorkflow

${strelka2_path}/$3_$1_$2_output/strelkaGermlineWorkflow/r
unWorkflow.py -m local -j 32
```

- Caller : DeepVariant

```
-   #!/bin/bash
-
-   source hg38_pipeline.cfg
-
-   BIN_VERSION="0.10.0"
-   N_SHARDS="32"
-
-   mkdir $BASE/output
-   OUTPUT_DIR="$BASE/output"
-
-   docker run \
-   -v "${INPUT_DIR}":"/input" \
-   -v "${OUTPUT_DIR}:/output" \
-   google/deepvariant:"${BIN_VERSION}" \
-   /opt/deepvariant/bin/run_deepvariant \
-   --model_type=WGS \
-   --ref=/input/hg38_v0_Homo_sapiens_assembly38.fasta \
-   --reads=/input/$1_$2_sort.bam \
-   --output_vcf=/output/$3_$1_$2_Deepvariant.output.vcf.gz \
-   --output_gvcf=/output/$3_$1_$2_Deepvariant.output.g.vcf.gz
    \
-   --num_shards=$N_SHARDS
-
-   docker system prune -f
```

-

- Caller : Samtools

```
-   #!/bin/bash
-
-   source pipeline.cfg
-
-   ## RealignerTargetCreator
-
-   mkdir ${samtools_path}/1.realign
-
-   java -Xmx4g -Xms4g -jar ${GATK3}/GenomeAnalysisTK.jar -T
    RealignerTargetCreator \
-    -R ${ref_fasta} \
-    -I ${Sort_path}/$2_Markdup_sort.bam \
-    -nt 4 \
-     -known ${Mills_indels} \
-     -known ${dbsnp_vcf} \
-     -known ${known_indels} \
-     -o
    ${samtools_path}/1.realign/$1_$2_realignment_targets.inter
    vals
-
-    # Realigner
-
-    java -Xmx4g -Xms4g -jar ${GATK3}/GenomeAnalysisTK.jar -T
    IndelRealigner \
-     -R ${ref_fasta} \
-     -I ${Sort_path}/$2_Markdup_sort.bam \
-     -targetIntervals
    ${samtools_path}/1.realign/$1_$2_realignment_targets.inter
    vals \
-     -known ${Mills_indels} \
-     -known ${dbsnp_vcf} \
-     -known ${known_indels} \
-     -o ${samtools_path}/1.realign/$1_$2_realign.bam
-
-    ## BaseRecalibrator
-
-    java -Xmx4g -Xms4g -jar ${GATK3}/GenomeAnalysisTK.jar -T
    BaseRecalibrator \
-     -R ${ref_fasta} \
-     -nct 4 \
-     -knownSites ${dbsnp_vcf} \
-     -knownSites ${known_indels} \
-     -knownSites ${Mills_indels} \
-     -I ${samtools_path}/1.realign/$1_$2_realign.bam \
-     -o ${samtools_path}/2.BaseRecal/$1_$2_recal.table
-
-    ## PrintReads
-
-    java -Xmx4g -Xms4g -jar ${GATK3}/GenomeAnalysisTK.jar -T
    PrintReads \
-     -R ${ref_fasta} \
-     -I ${samtools_path}/1.realign/$1_$2_realign.bam \
-     --BQSR ${samtools_path}/2.BaseRecal/$1_$2_recal.table \
-     -o ${samtools_path}/2.BaseRecal/$1_$2_recal.bam \
-     -nct 4
-
-    ## samtoolls mpileup
-
```

```
-   bcftools mpileup --threads 4 -Ou -f ${ref_fasta}
    ${samtools_path}/2.BaseRecal/$1_$2_recal.bam | bcftools
    call --threads 4 -vmO b -o
    ${samtools_path}/3.Samtools/$1_$2_mpileup.bcf
-
-   bcftools view --threads 4
    ${samtools_path}/3.Samtools/$1_$2_mpileup.bcf >
    ${samtools_path}/3.Samtools/$1_$2_samtools_raw.vcf
-
-   bgzip -c
    ${samtools_path}/3.Samtools/$1_$2_samtools_raw.vcf >
    ${samtools_path}/3.Samtools/$1_$2_samtools_raw.vcf.gz
-
-   tabix -p vcf
    ${samtools_path}/3.Samtools/$1_$2_samtools_raw.vcf.gz
-
-   bcftools filter --threads 4 -O z -o
    ${samtools_path}/3.Samtools/$1_$2_mpileup_filter.vcf.gz -s
    LOWQUAL -i '%QUAL>=20'
    ${samtools_path}/3.Samtools/$1_$2_samtools_raw.vcf.gz
```