ITU Department of Computer Engineering
BLG348E Introduction to Bioinformatics
Term Project

2023-2024 Fall

"Question everybody, especially authority,
and choose your mutations carefully."
Mark Mothersbaugh

1 Introduction

For this project, you are going to compare performance of different next generation sequencing (NGS) data processing algorithms using CoSAP (COmparative Sequencing Analysis Platform). CoSAP is an easy yet comprehensive pipeline creation tool for used NGS data. It provides reproducibility and aims to give deeper insight about the powers and limitations of the current tools by allowing users to compare results of different pipelines. A typical variant calling pipeline consists of Read Trimming, Read Mapping, Duplicate Removal-Base Calibration, Variant Calling and Variant Annotation steps. CoSAP provides several tool options for each of these steps as shown in Figure 1.

Each pipeline created for variant calling can have different parameters. CoSAP allows the users to configure these parameters by adding or deleting some steps on the pipeline. For example, one can include or exclude trimming, duplicate handling, or base recalibration steps on a pipeline.

In the scope of this project, you will create pipelines and apply variant calling operations on specific FASTQ files using CoSAP platform. You will work on several mappers and variant calling tools with the aim of creating VCF files. In the end, you will assess the performance of the variant calling operations done with different mappers and variant callers.

Besides the combinations of mappers and callers, you will also focus on base recalibration parameter (using base recalibration or not). Other parameters like trimming and duplicate handling will be fixed (trimming will be done and the duplicates will be marked). You will be using 2 mapper algorithms, namely BWA and Bowtie, and 3 variant calling algorithms, namely SomaticSniper, Mutect, and Strelka. In total, you will have 12 pipelines with different

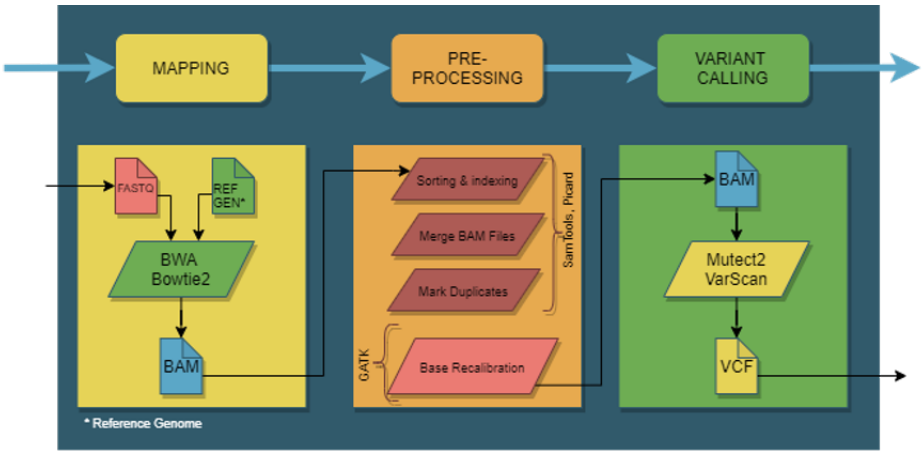

Figure 1: Simplified pipeline structure and file formats produced and accepted by each step of the pipeline.

configurations (2 base-recalibration options x 2 mappers x 3 variant callers).

In the end, you will compare the output of these pipelines and evaluate their performances by employing various metrics. Overall, the project will have four main parts:

1. Understanding of how NGS algorithms work.
2. Generating (1 sample x 2 recalibration conditions x 2 mapper x 3 variant caller) 12 pipelines using docker or local version of CoSAP with different parameters.
3. Comparing the pipelines with different creative visualizations and analyzing them.
4. Presenting the results.

2 Project Steps

2.1 Hardware Requirements

A computer equipped with 16 GB of RAM is recommended for executing the command line codes provided below. It is strongly encouraged to use a Linux operating system on your computer for this purpose. Alternatively, if you are using a Windows-based computer, you have the option to utilize the Windows Subsystem for Linux (WSL) to run the codes. In that case, please make sure

that WSL uses at least 15 GB of RAM.

To commence your work, it's necessary to download multiple data files, collectively occupying 50 GB of disk space. Additionally, during the execution process, intermediate files are generated, consuming approximately 25 GB of disk space. Ensure that your system has a minimum of 80 GB of available free space to accommodate these data and intermediate files.

2.2 Working with Docker

Docker is one option that can be used to run CoSAP. Please go to this link and install the correct docker version for your system by following the instructions. To pull the CoSAP image run the following command:

```
>>$ docker pull itubioinformatics/cosap
```

2.3 Working on local machine

2.3.1 Creating the conda environment

Another option to run the CoSAP is using the local version of CoSAP, which requires Miniconda (minimal installer for conda) to install required packages. Conda is a package and environment management system that allows creation of a standalone Python environment. A Python environment includes all the packages and libraries required for the project which ease the version and requirement management. Conda allows you to create an environment from a premade yml file. Supplied env.yml file contains all the required NGS analysis algorithms and tools for this project.

You can download the lightweight version of Conda from here.

2.3.2 Installing necessary packages

Once you install the Miniconda, clone[1] the CoSAP repository using the following command:

```
>>$ pip install git+https://github.com:MBaysanLab/cosap.git@develop
```

After cloning the repository, go to the CoSAP folder:

```
>>$ cd cosap-develop
```

and run:

```
>>$ make install
```

[1] In case you encounter an error while cloning the project, you can always download the CoSAP project via this link.

If you don't get an error after this command, you prepared your CoSAP environment successfully. If you encounter with an error, carefully read the error message and try to debug it.

After successfully installing everything, activate "cosap environment" via conda using the following command:

```
>>$ conda activate cosap
```

Then, you should install cosap module in your cosap environment (you should be in your cosap folder to run these commands):

```
>>$ pip install .
```

2.3.3 Adding CoSAP as an environment variable

You need to add the paths of CoSAP and CoSAP library as environmant variables. Run the following commands (on a Linux machine) by modifying the paths based on your paths. CoSAP path is the path of the pulled repository. CoSAP Library path is the path of the folder where you will store the data you will download.

```
>>$ export COSAP="cosap-path-here"
>>$ export COSAP_LIBRARY_PATH="cosap-library-path-here"
```

2.4 Downloading the Data

2.4.1 FASTQ Files

You need to download 2 FASTQ files (namely SRR7890850 and SRR7890851). You can get information about samples, sequencing centers and more via this article.

You can use sra-toolkit to download the germline-tumor pair data using the following command:

```
>>$ fastq-dump --split-files --gzip SRR7890850
```

Alternatively, you can download the FASTQ files directly from the web.

Finally you should have four .fq files (ex. SRR7890850_R1.fq, SRR7890850_R2.fq, SRR7890851_R1.fq, SRR7890851_R2.fq).

2.4.2 Reference Genome

From the Google Cloud for genomics data on this link, download the following files and place them under the same directory with the FASTQ files:

1. 1000G_phase1.snps.high_confidence.hg38.vcf.gz
2. 1000G_phase1.snps.high_confidence.hg38.vcf.gz.tbi

3. Homo_sapiens_assembly38.dbsnp138.vcf
4. Homo_sapiens_assembly38.dbsnp138.vcf.idx
5. Homo_sapiens_assembly38.dict
6. Homo_sapiens_assembly38.fasta
7. Homo_sapiens_assembly38.fasta.64.alt
8. Homo_sapiens_assembly38.fasta.64.amb
9. Homo_sapiens_assembly38.fasta.64.ann
10. Homo_sapiens_assembly38.fasta.64.bwt
11. Homo_sapiens_assembly38.fasta.64.pac
12. Homo_sapiens_assembly38.fasta.64.sa
13. Homo_sapiens_assembly38.fasta.fai
14. Mills_and_1000G_gold_standard.indels.hg38.vcf.gz
15. Mills_and_1000G_gold_standard.indels.hg38.vcf.gz.tbi

Downloaded reference genome names should be exactly like above. So, "resources_broad_hg38_v0_" prefix should be deleted.

Alternatively, you can download all reference data with the command below.

```
>>$ docker run -v /path_to_hg38_download_dir/:/cosap_data \
  itubioinformatics/cosap/make download_files
```

2.4.3 Mapper Indices

You should download index files that will be used for mapping operations. The downloaded files must be placed under the same folder which contains data acquired in Sections 2.4.1 and 2.4.2.

For Bowtie mapper, download the index files on this link, for BWA mapper, download these files.

2.4.4 High Confidence BED Files

For further filtering the VCF files, you need to download BED files. Download the high confidence bed file from here and WES data from here.

2.5 Configuring the Pipeline

For each pipeline with different parameters you should configure your pipeline file while running CoSAP. Details about the CoSAP tool are shared on the GitHub repository.

An example Python file which contains a possible configuration is shared on this link. You should modify thif file based on your pipeline preferences and file paths.

For instance, if you want to exclude base recalibration step on your pipeline, you need to delete the related lines on the pipeline.py file.

2.6 Running the Pipelines

You can either use Docker or use your local machine to run the pipeline. In both cases, the output of the pipeline should be VCF files.

2.6.1 Running on the Docker

You can use the following command:

```
>>$ docker run -it -v /your_hg38_bundle_path/:/cosap_data \
    -v /your_directory_with_data/:/workdir -v \
    /var/run/docker.sock:/var/run/docker.sock \
    itubioinformatics/cosap python pipeline.py
```

2.6.2 Running on the local computer

Go to your CoSAP directory, activate CoSAP environment and run the Python script.

```
>>$ cd cosap-develop \
  conda activate cosap \
  python pipeline.py
```

2.7 BED filtering and other filters

After you generate our VCF files, you should use BED filtering to direct the analysis to a genomic region. You will use BED files you have downloaded on Section 2.4.4. For each of the following steps, store the output files in different folders to avoid potential problems.

For ***VCF files created with SomaticSniper variant caller***, apply the following operation in order to fix the headers of the file. The FASTA file on the command has already been downloaded on Section 2.4.2.

```
>>$ gatk UpdateVCFSequenceDictionary -V "input-file.vcf" -R \
    "path/Homo_sapiens_assembly38.fasta" --output \
    "output_path/output-file.vcf"
```

For ***each VCF file created*** (remember, the ones created with SomaticSniper has been modified in the previous step. Use the modified ones.), run the following command.

```
>>$ bcftools +fixploidy "input-file.vcf" > "output-file.vcf"
```

Now, using these VCF files, apply BED filtering via the following command:

```
>>$ vcftools --vcf "output-file.vcf" --bed \
    "/path/High-Confidence_Regions_v1.2.bed" --out \
    "final-vcf-name.vcf" --recode --keep-INFO-all
```

Please remember that you have downloaded 2 BED files in Section 2.4.4. You should run the command above for both of these BED files.

After the BED filtering steps, you should filter your VCF files based on the "FILTER" column in VCF files. For VCF files generated by Mutect and Strelka variant caller, you should keep the "PASS" and "." ones. To eliminate those variants that do not satisfy this condition, you should write a simple `Python` script which reads the VCF files only keeps the variants whose "FILTER" values are "PASS" or ".".

Please be sure that the Mutect files have the filled filter column. If the filter column is empty, you should fix or re-generate it correctly.

For ***VCF files generated by SomaticSniper caller***, you should apply one more filtering step with this file named as snpfilter.pl:

```
>>$ perl snpfilter.pl --snp-file "input-somaticsniper-file.vcf" \
    --out-file "output-somaticsniper.vcf"
```

Don't worry about the filter column being empty in the output file of the command above, it's normal. If you are sure that you run the command without any errors, you can proceed to the next steps with the generated output VCF files. After applying all of the commands above, all of the VCF files are ready for the next step.

3 Analyzing Results

At the end of the implementations, you will write a report about the results you obtained. You should report the performances of your pipelines based on various metrics such as Precision, Recall, F1-Score, and Accuracy. You are encouraged to use visualization tools like heatmaps, box-and-whisker plots, histograms, PCA plots, and Precision-Recall curves. Be creative and use various methods.

You can use a high confidence VCF file as a ground truth for the variant list. The file is shared on this link.

To visualize the overlapping mutations you can use heatmap and Venn diagrams. Here is an example work that uses heatmap.

Your report should have the following sections:

- Abstract: includes the summary of your analyses.
- Introduction: includes information on DNA sequencing, algorithms and datasets you use etc. Add your computer properties such as operating system, RAM, storage, model etc.
- Results: includes your findings in your analyses. You need to include figures and/or tables as visuals. You need to generate at least one Principal Component Analysis (PCA) and one Heatmap in your analysis and include it in your report. Additionally, report the execution times of the pipelines.
- Discussion: includes your interpretations of the results and speculations for future research that should be done.
- References: includes the references you benefited from. They must be written in IEEE citation format.

In addition to the report, you will prepare a presentation with at most 15 pages. The presentation must contain the graphics you have created as a result of your analyses. Make sure that you can complete your presentation in maximum 10 minutes.

4 Bonus

Those of you who extend their research by adding the following analyses will be receiving bonus points:

- The effects of including/excluding trimming and marking/deleting duplicates,
- Using different FASTQ files (SRR7890874-SRR7890883 and SRR7890845-SRR7890844).