Whole Genome Sequencing

El objetivo es definir un procedimiento que procese un numero grande de secuencias WGS en el menor tiempo posible. Para ello,una vez definido el pipeline deberemos automatizar las tareas e integrarlas en el schedule manager del cluster (Como usar el cluster sin morir en el intento).

tl;dr

```
./wgs.pl -o <output_dir> [-cut <list.txt>] [-g] <input_dir>
```

Opciones:

- <input_dir> : (Mandatory) Directorio donde se encuentran todas las secuencias. El script buscara los sujetos y sus archivos dentro de este directorio.
- -o <output_dir> : (Opcional) Directorio donde se escribiran los resultados. En caso de obviarse se escribiran en el directorio desde el cual se lanza el script.
- -cut <list.txt> : (Opcional) Dice al script que analice SOLO los sujetos incluidos en el archivo que se suministra <list.txt>. Este archivo debe ser una listasimple de los sujetos a analizar.
- -g : Indica que no se borren los archivos temporales. Por defecto se borran, a no ser que se ponga este switch.

Pipeline

Primeramente hemos de definir el pipeline que se correra dentro del cluster. Aqui se ha de tener cuidado porque todas las herramientas y archivos han de ser accesibles desde cualquier nodo. En aras del siguiente paso podemos dividir el proceso en varios trozos. Tomemos por ejemplo el sujeto *seq-5*. Aqui los pasos son,

```
/nas/usr/local/bin/bwa mem -t 4 -R
"@RG\tID:V300016291_L01_549\tSM:seq5\tPL:BGI\tPI:380" -M
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_549_1.fq.gz
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_549_2.fq.gz
> /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_0.sam

/nas/usr/local/bin/bwa mem -t 4 -R
"@RG\tID:V300016291_L01_550\tSM:seq5\tPL:BGI\tPI:380" -M
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_550_1.fq.gz
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_550_2.fq.gz
> /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_1.sam

/nas/usr/local/bin/bwa mem -t 4 -R
"@RG\tID:V300016291_L01_551\tSM:seq5\tPL:BGI\tPI:380" -M
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_551_1.fq.gz
```

```
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_551_2.fq.gz
> /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_2.sam

/nas/usr/local/bin/bwa mem -t 4 -R
"@RG\tID:V300016291_L01_552\tSM:seq5\tPL:BGI\tPI:380" -M
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_552_1.fq.gz
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE/seq-5/V300016291_L01_552_2.fq.gz
> /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_3.sam

java -Djava.io.tmpdir=/nas/osotolongo/tmp/ -Xmx8g -jar
/nas/usr/local/bin/picard.jar MergeSamFiles
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_0.sam
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_1.sam
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_2.sam
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_3.sam
O=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5.sam
java -Djava.io.tmpdir=/nas/osotolongo/tmp/ -Xmx8g -jar
/nas/usr/local/bin/picard.jar SortSam
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5.sam
O=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_sorted.bam
SORT_ORDER=coordinate
/nas/software/samtools/bin/samtools index
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_sorted.bam

/nas/usr/local/bin/verifyBamID --vcf
/the_dysk/BGI_exome/reference/hapmap_3.3.hg38.vcf.gz --bam
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_sorted.bam --chip-none
--maxDepth 1000 --precise --verbose --ignoreRG --out
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_verifybam |& grep
-v "Skipping marker"

java -Djava.io.tmpdir=/nas/osotolongo/tmp/ -Xmx8g -jar
/nas/usr/local/bin/picard.jar ValidateSamFile IGNORE=MATE_NOT_FOUND
IGNORE=MISSING_READ_GROUP IGNORE=RECORD_MISSING_READ_GROUP
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_sorted.bam
java -Djava.io.tmpdir=/nas/osotolongo/tmp/ -Xmx8g -jar
/nas/usr/local/bin/picard.jar MarkDuplicates
I=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_sorted.bam
O=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_GATKready.bam
METRICS_FILE=/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_metrics.t
xt QUIET=true MAX_RECORDS_IN_RAM=2000000 ASSUME_SORTED=TRUE
CREATE_INDEX=TRUE
java -jar /nas/usr/local/opt/gatk3/GenomeAnalysisTK.jar -T DepthOfCoverage -
R /the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta -nt 1 -ct 10 -
ct 15 -ct 20 -ct 30 --omitDepthOutputAtEachBase --omitIntervalStatistics --
omitLocusTable -L
/the_dysk/BGI_exome/reference/MGI_Exome_Capture_V5_bis2.bed -I
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_GATKready.bam -o
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_exome_coverage
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
```

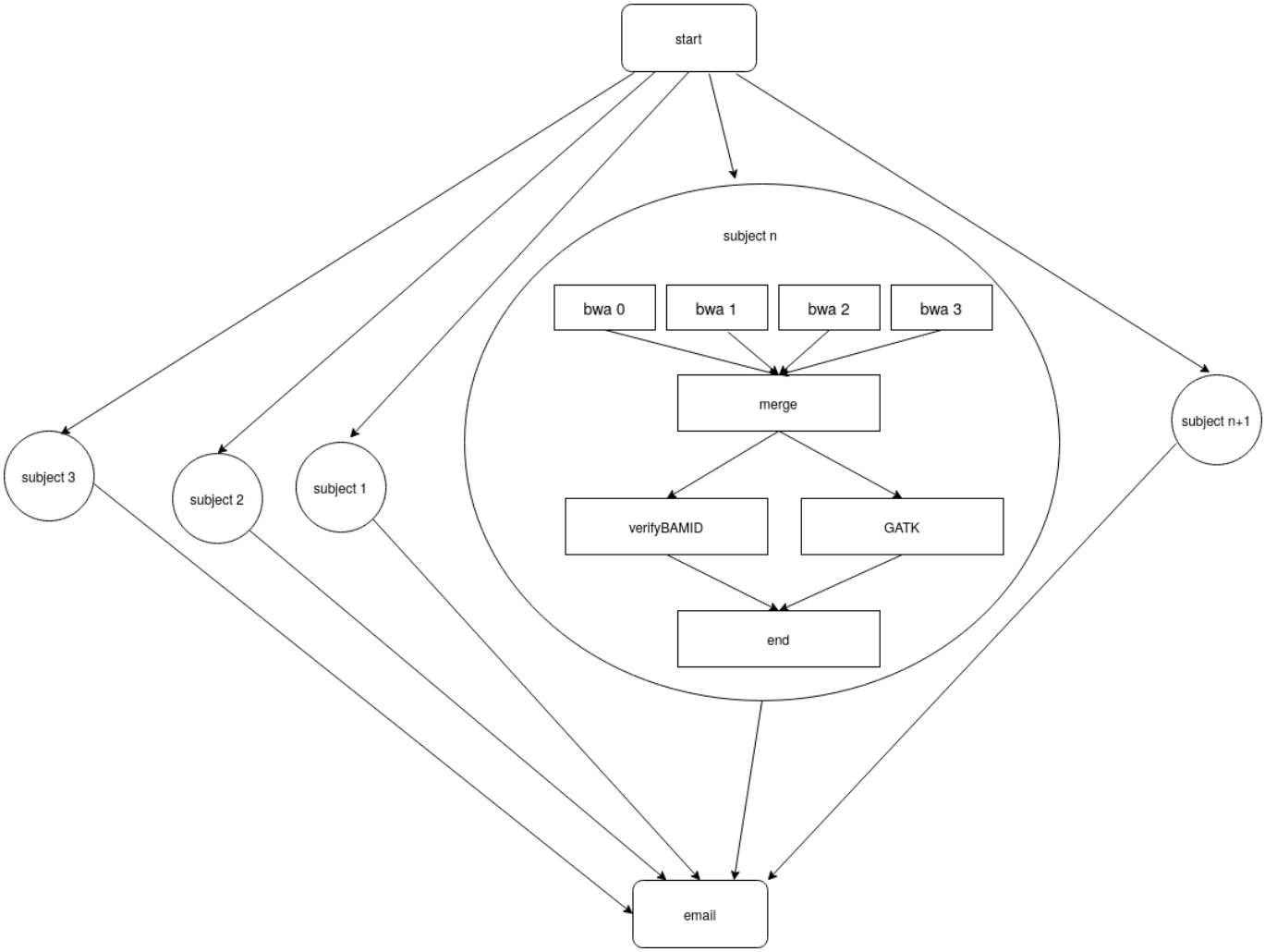
```
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" BaseRecalibrator -I
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_GATKready.bam -R
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta --known-sites
/the_dysk/BGI_exome/reference/Mills_and_1000G_gold_standard.indels.hg38.vcf.
gz --known-sites /the_dysk/BGI_exome/reference/dbsnp_146.hg38.vcf.gz --
known-sites
/the_dysk/BGI_exome/reference/1000G_phase1.snps.high_confidence.hg38.vcf.gz
-O /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table1
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" ApplyBQSR -R
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta -I
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_GATKready.bam -bqsr-
recal-file
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table1 -O
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_recal.bam
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" AnalyzeCovariates -bqsr
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table1 --
plots
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_AnalyzeCovariates.
pdf
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" BaseRecalibrator -I
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_recal.bam -R
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta --known-sites
/the_dysk/BGI_exome/reference/Mills_and_1000G_gold_standard.indels.hg38.vcf.
gz --known-sites /the_dysk/BGI_exome/reference/dbsnp_146.hg38.vcf.gz --
known-sites
/the_dysk/BGI_exome/reference/1000G_phase1.snps.high_confidence.hg38.vcf.gz
-O /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table2
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" AnalyzeCovariates -before
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table1 -
after
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/tmp/seq5_recal_data.table2 -
plots /the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_before-
after-plots.pdf
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G" HaplotypeCaller -R
/the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta -I
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_recal.bam -ERC
GVCF --dbsnp /the_dysk/BGI_exome/reference/dbsnp_146.hg38.vcf.gz -O
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_raw.snps.indels.g.
vcf.gz
singularity run --cleanenv -B /nas:/nas -B /the_dysk:/the_dysk
```

```
/usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G -XX:+UseConcMarkSweepGC"
VariantEval -R /the_dysk/BGI_exome/reference/Homo_sapiens_assembly38.fasta -
L /the_dysk/BGI_exome/reference/MGI_Exome_Capture_V5_bis2.bed -D
/the_dysk/BGI_exome/reference/dbsnp_146.hg38.vcf.gz -O
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_eval.gatkreport --
eval
/the_dysk/BGI_exome/F18FTSEUET0180/WGS/seq-5/results/seq5_raw.snps.indels.g.
vcf.gz
```

Este pipeline se ha dividido en varios bloques que dependen unos de otros. Los 4 primeros son independientes y se pueden correr en paralelo. El numero 5 depende de la terminacion de los 4 primeros. Los bloques 6 y 7 dependen del numero 5 pero pueden correrse independientemente.

Paralelizacion

El montaje correcto del pipeline permite la paralelizacioen del procediemiento dentro de cada secuenciacion, pero ademas, debe paralelizarse el procedimientp total. Es decir, cadda sujeto debe correr en paralelo a los demas. Para ello basta crear un sistema de dependencias como el que se muestra en la figura.



Entonces, una vez definido el pipeline, la secuencia de ejecucion de cada trozo y el modo de paralelizacion seria sencillo definir el flujo de ejecucion si supieramos que archivos iniciales debe cargar cada ejecucion inicial (los *bwa*).

Parsing

Lo primero que ddebemos saber es que el directorio de *input* esta compuesto por una lista de subdirectorios cada uno perteneciente a un sujeto distinto.

```
[osotolongo@detritus HUMehbE]$ pwd
/the_dysk/BGI_exome/F18FTSEUET0180/HUMehbE
[osotolongo@detritus HUMehbE]$ ls
BGI_Data_List_F18FTSEUET0180_filled.pdf                    md5sum.check
seq-1   seq-11  seq-13  seq-15  seq-17  seq-19  seq-20  seq-22  seq-24
seq-26  seq-28  seq-4   seq-6   seq-8
BGI_Sequencing_Analysis_Report_F18FTSEUET0180_HUMehbE.pdf  md5sum.txt
seq-10  seq-12  seq-14  seq-16  seq-18  seq-2   seq-21  seq-23  seq-25
seq-27  seq-3   seq-5   seq-7   seq-9
```

Para cada sujeto hay una lista de ocho archivos que deben parearse segun el nombre de archivo.

```
[osotolongo@detritus HUMehbE]$ ls seq-5/
V300016291_L01_549_1.fq.gz  V300016291_L01_550_1.fq.gz
V300016291_L01_551_1.fq.gz  V300016291_L01_552_1.fq.gz
V300016291_L01_549_2.fq.gz  V300016291_L01_550_2.fq.gz
V300016291_L01_551_2.fq.gz  V300016291_L01_552_2.fq.gz
```

En este ejemplo deben parearse,

- V300016291_L01_549_1.fq.gz y V300016291_L01_549_2.fq.gz
- V300016291_L01_550_1.fq.gz y V300016291_L01_550_2.fq.gz
- V300016291_L01_551_1.fq.gz y V300016291_L01_551_2.fq.gz
- V300016291_L01_552_1.fq.gz y V300016291_L01_552_2.fq.gz

asi que la estructura a seguir es bastante clara. No obstante a que los nombres de archivo varian entre sujetos, siguen la misma estructura.

```
[osotolongo@detritus HUMehbE]$ ls seq-10
V300016281_L01_553_1.fq.gz  V300016281_L01_554_1.fq.gz
V300016281_L01_555_1.fq.gz  V300016281_L01_556_1.fq.gz
V300016281_L01_553_2.fq.gz  V300016281_L01_554_2.fq.gz
V300016281_L01_555_2.fq.gz  V300016281_L01_556_2.fq.gz
```

Lo que voy hacer entonces es definir un *hash* donde se guarda la informacion relativa a cada sujeto (incluidos los patrones en los archivos originales) y a partir de ahi construir los scripts.

Veamos, primero leo el *input dir*

```
my $src_dir = shift;
```

```
$src_dir =~ s/\/$//;
my $find_rule = File::Find::Rule->new;
my @ppaths = $find_rule->maxdepth(1)->directory->in($src_dir);
@ppaths = grep {!/$src_dir$/} @ppaths;
my %ltpaths = map { /.*\/(.*)?$/ => $_ } @ppaths;
my %finfo;
my %lpaths;
```

luego, en dependencia de las opciones puedo reducir la lista o no,

```
if($cfile && -e $cfile && -f $cfile){
        my @cuts = read_file $cfile;
        chomp @cuts;
        foreach my $cut (@cuts){
                if(grep {/$cut/} %ltpaths){
                        $lpaths{$cut} = $ltpaths{$cut};
                }
        }
}else{
        %lpaths = %ltpaths;
}
```

y ahora lleno el *hash* de datos,

```
foreach my $pollo (sort keys %lpaths){
        opendir PD, $lpaths{$pollo} or next;
        my @fqs = grep { !/^\./} readdir PD;
        #$finfo{$pollo}{'name'} = $pollo;
        $finfo{$pollo}{'path'} = $lpaths{$pollo};
        foreach my $fq (@fqs) {
                $_=$fq;
                my @ids  = /^V(\d*?)\_L(\d*?)\_(\d*?)\_\d\.fq\.gz$/;
                $finfo{$pollo}{'fq_name'} = 'V'.$ids[0].'_L'.$ids[1];
                push @{ $finfo{$pollo}{'fq_list'} }, $ids[2];
        }
}
```

Programatic tree

Ya casi esta, ahora por cada sujeto guardado pueden definirse los primeros 4 procesos,

```
        for (my $i=0; $i<4; $i++){
                my $fqid = $finfo{$pollo}{'fq_list'}[2*$i];
                my $orderfile = $outdir.'/bwa_'.$pollo.'_'.$i.'.sh';
                open ORD, ">$orderfile" or die "Couldnt create file";
                print ORD '#!/bin/bash'."\n";
                print ORD '#SBATCH -J sam_'.$pollo."\n";
                print ORD '#SBATCH --time=24:0:0'."\n";
```

```
                print ORD '#SBATCH -o '.$outdir.'/bwa_'.$pollo.'-%j'."\n";
                print ORD '#SBATCH -c 8'."\n";
                print ORD '#SBATCH --mem-per-cpu=4G'."\n";
                print ORD '#SBATCH -p fast'."\n";
                print ORD '#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT'."\n"; #no quieres que te mande email de todo
                print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
                print ORD $bwa.' mem -t 4 -R
"@RG\tID:'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'\tSM:'.$pname.'\tPL:BGI\tPI:
380" -M '.$ref_dir.'/Homo_sapiens_assembly38
'.$finfo{$pollo}{'path'}.'/'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'_1.fq.gz
'.$finfo{$pollo}{'path'}.'/'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'_2.fq.gz >
'.$tmpdir.'/'.$pname.'_'.$i.'.sam';
                print ORD "\n";
                close ORD;
                $gsconv.= 'I='.$tmpdir.'/'.$pname.'_'.$i.'.sam ';
                system("sbatch $orderfile");
        }
```

El numero 5 se hace que dependa de estos 4,

```
        my $orderfile = $outdir.'/merge_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
        print ORD '#SBATCH -J sam_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH -o '.$outdir.'/merge_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 8'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n"; #no
quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $picard.' MergeSamFiles '.$gsconv.'
O='.$tmpdir.'/'.$pname.'.sam'."\n";
        print ORD $picard.' SortSam I='.$tmpdir.'/'.$pname.'.sam
O='.$tmpdir.'/'.$pname.'_sorted.bam SORT_ORDER=coordinate'."\n";
        print ORD $samtools.' index '.$tmpdir.'/'.$pname.'_sorted.bam'."\n";
        close ORD;
        my $order = 'sbatch --dependency=singleton '.$orderfile;
        my $ujobid = `$order`;
        $ujobid = ( split ' ', $ujobid )[ -1 ];
```

El 6 y el 7 dependen del 5,

```
        $orderfile = $outdir.'/verify_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
```

```
        print ORD '#SBATCH -J verify_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH -o '.$outdir.'/verify_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 4'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n"; #no
quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $verifybamib. ' --vcf '.$ref_dir.'/hapmap_3.3.hg38.vcf.gz
--bam '.$tmpdir.'/'.$pname.'_sorted.bam --chip-none --maxDepth 1000 --
precise --verbose --ignoreRG --out '.$resdir.'/'.$pname.'_verifybam |& grep
-v "Skipping marker"'."\n";
        close ORD;
        $order = 'sbatch --dependency=afterok:'.$ujobid.' '.$orderfile;
        my $jobid = `$order`;
        $jobid = ( split ' ', $jobid )[ -1 ];
        push @jobids, $jobid;
```

```
        $orderfile = $outdir.'/validate_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
        print ORD '#SBATCH -J validate_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH -o '.$outdir.'/validate_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 12'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n"; #no
quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $picard.' ValidateSamFile IGNORE=MATE_NOT_FOUND
IGNORE=MISSING_READ_GROUP IGNORE=RECORD_MISSING_READ_GROUP
I='.$tmpdir.'/'.$pname.'_sorted.bam'."\n";
        print ORD $picard.' MarkDuplicates
I='.$tmpdir.'/'.$pname.'_sorted.bam O='.$tmpdir.'/'.$pname.'_GATKready.bam
METRICS_FILE='.$resdir.'/'.$pname.'_metrics.txt QUIET=true
MAX_RECORDS_IN_RAM=2000000 ASSUME_SORTED=TRUE CREATE_INDEX=TRUE'."\n";
        print ORD $gatk3.' -T DepthOfCoverage -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -nt 1 -ct 10 -ct 15 -ct 20 -ct 30
--omitDepthOutputAtEachBase --omitIntervalStatistics --omitLocusTable -L
'.$ref_dir.'/MGI_Exome_Capture_V5_bis2.bed -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -o
'.$resdir.'/'.$pname.'_exome_coverage'."\n";
        print ORD $gatk4.' BaseRecalibrator -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta --known-sites
'.$ref_dir.'/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz --known-sites
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz --known-sites
'.$ref_dir.'/1000G_phase1.snps.high_confidence.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_recal_data.table1'."\n";
```

```
        print ORD $gatk4.' ApplyBQSR -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -bqsr-recal-file
'.$resdir.'/'.$pname.'_recal_data.table1 -O
'.$resdir.'/'.$pname.'_recal.bam'."\n";
        print ORD $gatk4.' AnalyzeCovariates -bqsr
'.$resdir.'/'.$pname.'_recal_data.table1 --plots
'.$resdir.'/'.$pname.'_AnalyzeCovariates.pdf'."\n";
        print ORD $gatk4.' BaseRecalibrator -I
'.$resdir.'/'.$pname.'_recal.bam -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta --known-sites
'.$ref_dir.'/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz --known-sites
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz --known-sites
'.$ref_dir.'/1000G_phase1.snps.high_confidence.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_recal_data.table2'."\n";
        print ORD $gatk4.' AnalyzeCovariates -before
'.$resdir.'/'.$pname.'_recal_data.table1 -after
'.$resdir.'/'.$pname.'_recal_data.table2 -plots
'.$resdir.'/'.$pname.'_before-after-plots.pdf'."\n";
        print ORD $gatk4.' HaplotypeCaller -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -I
'.$resdir.'/'.$pname.'_recal.bam -ERC GVCF --dbsnp
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_raw.snps.indels.g.vcf.gz'."\n";
        print ORD $gatk4_l.' VariantEval -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -L
'.$ref_dir.'/MGI_Exome_Capture_V5_bis2.bed -D
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz -O '.$resdir.'/'.$pname.'_eval.gatkreport
--eval '.$resdir.'/'.$pname.'_raw.snps.indels.g.vcf.gz'."\n";
        close ORD;
        $order = 'sbatch --dependency=afterok:'.$ujobid.' '.$orderfile;
        $jobid = `$order`;
        $jobid = ( split ' ', $jobid )[ -1 ];
        push @jobids, $jobid;
```

y por ultimo, hay un proceso que depende de que terminen todos los demas (TODOS) y lo que hace es limpiar los archivos temporales y enviar un email de finalizacion,

```
my $orderfile = $outdir.'/wgs_end.sh';
open ORD, ">$orderfile";
print ORD '#!/bin/bash'."\n";
print ORD '#SBATCH -J wgs_end'."\n";
print ORD '#SBATCH --mail-type=END'."\n"; #email cuando termine o falle
print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
print ORD '#SBATCH -o '.$outdir.'/wgs_end-%j'."\n";
unless ($debug){
        print ORD "rm -rf $w_dir/*/tmp\n";
}else{
        print ORD ":\n";
}
```

```
close ORD;
my $sjobs = join(',',@jobids);
my $order = 'sbatch --depend=afterok:'.$sjobs.' '.$orderfile;
print "$order\n";
exec($order);
```

La magia completa aqui, en menos de 200 lineas

wgs.pl

```
#!/usr/bin/perl

# Copyright 2020 O. Sotolongo <asqwerty@gmail.com>
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

use strict; use warnings;
use File::Find::Rule;
use File::Slurp qw(read_file);
use Data::Dump qw(dump);

######################################################################
################### rellenar los paths sistema ###################
######################################################################
my $ref_dir = '/the_dysk/BGI_exome/reference';
my $bwa ='/nas/usr/local/bin/bwa';
#my $picard = 'java -jar /nas/usr/local/bin/picard.jar';
my $picard = 'java -Djava.io.tmpdir=/nas/'.$ENV{'USER'}.'/tmp/ -Xmx8g -
jar /nas/usr/local/bin/picard.jar';
my $samtools = '/nas/software/samtools/bin/samtools';
my $verifybamib = '/nas/usr/local/bin/verifyBamID';
my $gatk3 = 'java -jar /nas/usr/local/opt/gatk3/GenomeAnalysisTK.jar';
#my $gatk4 = '/nas/usr/local/opt/gatk4/gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G"';
#my $gatk4_l = '/nas/usr/local/opt/gatk4/gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G -Xmx16G -
XX:+UseConcMarkSweepGC"';
my $gatk4 = 'singularity run --cleanenv -B /nas:/nas -B
/the_dysk:/the_dysk /usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G"';
my $gatk4_l = 'singularity run --cleanenv -B /nas:/nas -B
```

```
/the_dysk:/the_dysk /usr/local/bin/gatk4.simg gatk --java-options "-
DGATK_STACKTRACE_ON_USER_EXCEPTION=true -Xmx16G -
XX:+UseConcMarkSweepGC"';
#####################################################################
#####################################################################
#####################################################################
my $cfile;
my $outdatadir;
my $debug = 0;
@ARGV = ("-h") unless @ARGV;
while (@ARGV and $ARGV[0] =~ /^-/) {
    $_ = shift;
    last if /^--$/;
    if (/^-cut/) { $cfile = shift; chomp($cfile);}
    if (/^-o/) { $outdatadir = shift;}
    if (/^-g/) { $debug = 1;}
}
my $src_dir = shift;
$src_dir =~ s/\/$//;
my $find_rule = File::Find::Rule->new;
my @ppaths = $find_rule->maxdepth(1)->directory->in($src_dir);
@ppaths = grep {!/$src_dir$/} @ppaths;
my %ltpaths = map { /.*\/(.*)?$/ => $_ } @ppaths;
my %finfo;
my %lpaths;

if($cfile && -e $cfile && -f $cfile){
        my @cuts = read_file $cfile;
        chomp @cuts;
        foreach my $cut (@cuts){
                if(grep {/$cut/} %ltpaths){
                        $lpaths{$cut} = $ltpaths{$cut};
                }
        }
}else{
        %lpaths = %ltpaths;
}

# Creo el entorno de ejecucion
my $tmp_shit = '/nas/'.$ENV{'USER'}.'/tmp/';
mkdir $tmp_shit;
my $wdir;
if($outdatadir){
        mkdir $outdatadir;
        $wdir = $outdatadir;
}else{
        $wdir = $ENV{'PWD'};
}
my $outdir = $wdir.'/slurm';
mkdir $outdir;
#Recopilo la informacion de los archivos de entrada
```

```
foreach my $pollo (sort keys %lpaths){
        opendir PD, $lpaths{$pollo} or next;
        my @fqs = grep { !/^\./} readdir PD;
        #$finfo{$pollo}{'name'} = $pollo;
        $finfo{$pollo}{'path'} = $lpaths{$pollo};
        foreach my $fq (@fqs) {
                $_=$fq;
                my @ids  = /^V(\d*?)\_L(\d*?)\_(\d*?)\_\d\.fq\.gz$/;
                $finfo{$pollo}{'fq_name'} = 'V'.$ids[0].'_L'.$ids[1];
                push @{ $finfo{$pollo}{'fq_list'} }, $ids[2];
        }
}
#Creo y ejecuto los procesos
my @jobids;
foreach my $pollo (sort keys %finfo){
        my $udir = $wdir.'/'.$pollo;
        mkdir $udir;
        my $tmpdir = $udir.'/tmp';
        mkdir $tmpdir;
        my $resdir = $udir.'/results';
        mkdir $resdir;
        my $gsconv = "";
        my @ujobids;
        (my $pname = $pollo) =~ s/-//g;
        for (my $i=0; $i<4; $i++){
                my $fqid = $finfo{$pollo}{'fq_list'}[2*$i];
                my $orderfile = $outdir.'/bwa_'.$pollo.'_'.$i.'.sh';
                open ORD, ">$orderfile" or die "Couldnt create file";
                print ORD '#!/bin/bash'."\n";
                print ORD '#SBATCH -J sam_'.$pollo."\n";
                print ORD '#SBATCH --time=24:0:0'."\n";
                print ORD '#SBATCH -o '.$outdir.'/bwa_'.$pollo.'-
%j'."\n";
                print ORD '#SBATCH -c 8'."\n";
                print ORD '#SBATCH --mem-per-cpu=4G'."\n";
                print ORD '#SBATCH -p fast'."\n";
                print ORD '#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT'."\n"; #no quieres que te mande email de
todo
                print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
                print ORD $bwa.' mem -t 4 -R
"@RG\tID:'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'\tSM:'.$pname.'\tPL:BGI
\tPI:380" -M '.$ref_dir.'/Homo_sapiens_assembly38
'.$finfo{$pollo}{'path'}.'/'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'_1.fq
.gz
'.$finfo{$pollo}{'path'}.'/'.$finfo{$pollo}{'fq_name'}.'_'.$fqid.'_2.fq
.gz > '.$tmpdir.'/'.$pname.'_'.$i.'.sam';
                print ORD "\n";
                close ORD;
                $gsconv.= 'I='.$tmpdir.'/'.$pname.'_'.$i.'.sam ';
                system("sbatch $orderfile");
```

```
        }
        my $orderfile = $outdir.'/merge_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
        print ORD '#SBATCH -J sam_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH -o '.$outdir.'/merge_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 8'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n";
#no quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $picard.' MergeSamFiles '.$gsconv.'
O='.$tmpdir.'/'.$pname.'.sam'."\n";
        print ORD $picard.' SortSam I='.$tmpdir.'/'.$pname.'.sam
O='.$tmpdir.'/'.$pname.'_sorted.bam SORT_ORDER=coordinate'."\n";
        print ORD $samtools.' index
'.$tmpdir.'/'.$pname.'_sorted.bam'."\n";
        close ORD;
        my $order = 'sbatch --dependency=singleton '.$orderfile;
        my $ujobid = `$order`;
        $ujobid = ( split ' ', $ujobid )[ -1 ];

        $orderfile = $outdir.'/verify_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
        print ORD '#SBATCH -J verify_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH -o '.$outdir.'/verify_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 4'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n";
#no quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $verifybamib. ' --vcf
'.$ref_dir.'/hapmap_3.3.hg38.vcf.gz --bam
'.$tmpdir.'/'.$pname.'_sorted.bam --chip-none --maxDepth 1000 --precise
--verbose --ignoreRG --out '.$resdir.'/'.$pname.'_verifybam |& grep -v
"Skipping marker"'."\n";
        close ORD;
        $order = 'sbatch --dependency=afterok:'.$ujobid.' '.$orderfile;
        my $jobid = `$order`;
        $jobid = ( split ' ', $jobid )[ -1 ];
        push @jobids, $jobid;
        $orderfile = $outdir.'/validate_'.$pollo.'.sh';
        open ORD, ">$orderfile";
        print ORD '#!/bin/bash'."\n";
        print ORD '#SBATCH -J validate_'.$pollo."\n";
        print ORD '#SBATCH -p fast'."\n";
```

```
        print ORD '#SBATCH -o '.$outdir.'/validate_'.$pollo.'-%j'."\n";
        print ORD '#SBATCH -c 12'."\n";
        print ORD '#SBATCH --mem-per-cpu=4G'."\n";
        print ORD '#SBATCH -p fast'."\n";
        print ORD '#SBATCH --mail-type=FAIL,TIME_LIMIT,STAGE_OUT'."\n";
#no quieres que te mande email de todo
        print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
        print ORD $picard.' ValidateSamFile IGNORE=MATE_NOT_FOUND
IGNORE=MISSING_READ_GROUP IGNORE=RECORD_MISSING_READ_GROUP
I='.$tmpdir.'/'.$pname.'_sorted.bam'."\n";
        print ORD $picard.' MarkDuplicates
I='.$tmpdir.'/'.$pname.'_sorted.bam
O='.$tmpdir.'/'.$pname.'_GATKready.bam
METRICS_FILE='.$resdir.'/'.$pname.'_metrics.txt QUIET=true
MAX_RECORDS_IN_RAM=2000000 ASSUME_SORTED=TRUE CREATE_INDEX=TRUE'."\n";
        print ORD $gatk3.' -T DepthOfCoverage -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -nt 1 -ct 10 -ct 15 -ct 20 -
ct 30 --omitDepthOutputAtEachBase --omitIntervalStatistics --
omitLocusTable -L '.$ref_dir.'/MGI_Exome_Capture_V5_bis2.bed -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -o
'.$resdir.'/'.$pname.'_exome_coverage'."\n";
        print ORD $gatk4.' BaseRecalibrator -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta --known-sites
'.$ref_dir.'/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz --known-
sites '.$ref_dir.'/dbsnp_146.hg38.vcf.gz --known-sites
'.$ref_dir.'/1000G_phase1.snps.high_confidence.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_recal_data.table1'."\n";
        print ORD $gatk4.' ApplyBQSR -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -I
'.$tmpdir.'/'.$pname.'_GATKready.bam -bqsr-recal-file
'.$resdir.'/'.$pname.'_recal_data.table1 -O
'.$resdir.'/'.$pname.'_recal.bam'."\n";
        print ORD $gatk4.' AnalyzeCovariates -bqsr
'.$resdir.'/'.$pname.'_recal_data.table1 --plots
'.$resdir.'/'.$pname.'_AnalyzeCovariates.pdf'."\n";
        print ORD $gatk4.' BaseRecalibrator -I
'.$resdir.'/'.$pname.'_recal.bam -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta --known-sites
'.$ref_dir.'/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz --known-
sites '.$ref_dir.'/dbsnp_146.hg38.vcf.gz --known-sites
'.$ref_dir.'/1000G_phase1.snps.high_confidence.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_recal_data.table2'."\n";
        print ORD $gatk4.' AnalyzeCovariates -before
'.$resdir.'/'.$pname.'_recal_data.table1 -after
'.$resdir.'/'.$pname.'_recal_data.table2 -plots
'.$resdir.'/'.$pname.'_before-after-plots.pdf'."\n";
        print ORD $gatk4.' HaplotypeCaller -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -I
'.$resdir.'/'.$pname.'_recal.bam -ERC GVCF --dbsnp
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz -O
```

```
'.$resdir.'/'.$pname.'_raw.snps.indels.g.vcf.gz'."\n";
        print ORD $gatk4_l.' VariantEval -R
'.$ref_dir.'/Homo_sapiens_assembly38.fasta -L
'.$ref_dir.'/MGI_Exome_Capture_V5_bis2.bed -D
'.$ref_dir.'/dbsnp_146.hg38.vcf.gz -O
'.$resdir.'/'.$pname.'_eval.gatkreport --eval
'.$resdir.'/'.$pname.'_raw.snps.indels.g.vcf.gz'."\n";
        close ORD;
        $order = 'sbatch --dependency=afterok:'.$ujobid.' '.$orderfile;
        $jobid = `$order`;
        $jobid = ( split ' ', $jobid )[ -1 ];
        push @jobids, $jobid;
}
my $orderfile = $outdir.'/wgs_end.sh';
open ORD, ">$orderfile";
print ORD '#!/bin/bash'."\n";
print ORD '#SBATCH -J wgs_end'."\n";
print ORD '#SBATCH --mail-type=END'."\n"; #email cuando termine o falle
print ORD '#SBATCH --mail-user='."$ENV{'USER'}\n";
print ORD '#SBATCH -o '.$outdir.'/wgs_end-%j'."\n";
unless ($debug){
        print ORD "rm -rf $w_dir/*/tmp\n";
}else{
        print ORD ":\n";
}
close ORD;
my $sjobs = join(',',@jobids);
my $order = 'sbatch --depend=afterok:'.$sjobs.' '.$orderfile;
print "$order\n";
exec($order);
```

Ejecucion

Troubleshooting