Filtering Alignment Files

Many HTS data alignment procedures record information about the short reads which are aligned to a reference genome. Short reads with undesirable characteristics can be filtered before downstream analyses, such as SNV calling.

We will focus this discussion on the Sequence Alignment/Map (SAM) format ([1],[2]) since it is the emerging file format standard for storing nucleotide sequence alignments. The BAM file format is a compressed, binary version of the SAM file format which can be indexed for computational efficiency. SAMtools [1] is a popular software suite that provides a toolkit for manipulating SAM format alignments.

1 SAM/BAM Bitwise Flags

Among the bitwise flags of the SAM/BAM alignment section, the final 3 flags may indicate that the read is undesirable and should be filtered before SNV calling. Please see table 1 for a summary of these flags. The specific definitions of these flags may vary depending on the sequencing and alignment software that is used. Since the following is general information about these flags, *we encourage readers to consult the documentation of the software that they use.*

The bit 0x100 flag indicates a non-primary read alignment and read alignment ambiguity. The bit 0x200 flag indicates that certain read quality metrics were not passed. The assessment of these reads quality metrics or controls is typically done during pre-alignment sequencing. The bit 0x400 flag indicates that the read is most likely either a polymerase chain reaction (PCR) duplicate or an optical duplicate. PCR duplicates arise when the same parent DNA molecule is repeatedly sequenced over the course of many PCR cycles, thus these PCR duplicates do not offer unique information. Optical duplicates can occur when, during the image analysis of sequencing, the sequences of one cluster are falsely identified to belong to another real or illusory cluster.

2 Minimally Recommended Practices

Many users choose to filter reads with the above flags to prevent the bias that these reads can introduce in SNV calling. For example, retaining PCR or optical duplicate reads can distort the true allele frequency distribution and lead to false positive or false negative SNV calls. Likewise, retaining reads with established quality control failures or ambiguous alignments can also lead to false SNV calls.

Bit (Hex)	Hex-to-Decimal	Decimal	Description
0x100	$1 \times (16^2)$	256	Non-primary alignment
0x200	$2 \times (16^2)$	512	Read fails quality controls
0x400	$4 \times (16^2)$	1024	Read is PCR or optical duplicate

Table 1: Selected bitwise flags from the FLAG field of the SAM/BAM alignment section [2]. The columns provide 1) the bitwise flags written in hexadecimal (hex) with the Unix-like/C "0x" notational prefix, 2) an expression to obtain the flag decimal value from the hexadecimal value, 3) the flag decimal value and 4) a short flag description.

A popular procedure used to filter these reads is `samtools view` with the `-F` option. This option takes as its argument the integer sum of the decimal representations of the bitwise flags that we wish to filter. To demonstrate, suppose we wish to filter out any read that has at least one of the following bitwise flags: 0x100, 0x200 and 0x400. To obtain the correct `samtools view -F` argument, we simply sum the decimal representations of these bits: $256 + 512 + 1024 = 1792$. Then we run `samtools view -F 1792` on our BAM file. Likewise, if we wish to filter out only those reads that have either the non-primary alignment flag (0x100) or the PCR/optical duplicate flag (0x400), then we should run `samtools view -F 1280` since $256 + 1024 = 1280$.

The following are **minimally** recommended command-lines for filtering a SAM/BAM alignment file and converting the filtered BAM file to a pileup file, which can then be processed by the GeMS SNV caller. Please see the SAMtools documentation for more information on these procedures and the recommended usage of their options.

```
# 0. If starting with SAM, convert SAM to BAM
samtools view -bT ref.fasta aln.sam -o aln.bam
# 1. Sort the BAM and output to srt.bam
samtools sort aln.bam srt
# 2. Index the BAM which creates srt.bam.bai
samtools index srt.bam
# 3. Output mapped(3rd col)/unmapped(4th col) read counts
samtools idxstats srt.bam
# 4. Filter out reads with flags 0x100, 0x200 and 0x400
samtools view -bF 1792 srt.bam -o flt.bam
# 5. Index the new filtered BAM file
samtools index flt.bam
# 6. Read counts should be less than or equal to before
samtools idxstats flt.bam
# 7. Create pileup file with mapping(q)/base(Q) quality filters
samtools mpileup -q 20 -Q 17 -sf ref.fasta flt.bam > flt.pileup
```

3 Contact

Xinping Cui
xinping.cui@ucr.edu

https://sites.google.com/a/bioinformatics.ucr.edu/xinping-cui/
https://github.com/cui-lab

References

[1] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, August 2009.

[2] The SAM/BAM Format Specification Working Group. Sequence alignment/map format specification. https://samtools.github.io/hts-specs/SAMv1.pdf.