Variant Normalization

Overview

> *A genomic variant is represented by a locus* (chromosome + position)*, the reference allele and list of alternate alleles. Genotypes are represented by the two alleles in the sample at the locus.*

Different variant calling tools may use subtly different representations for the same biological sequence variant. If variants called from a sample are to be annotated or those from multiple samples are to be merged it is important that variant calls are normalised to ensure consistent representation; see this vt article or GiaB article for info. In some cases normalisation may also be useful to identify and remove spurious duplicates called within a call set from a single sample.

OpenCGA performs variant normalisation by default when genotypes are loaded into the database. The procedures implemented by OpenCGA v2.0 are described in this document. The approach is similar but not identical to other tools that perform variant normalisation such as bcftools, vt, GATK and vcflib. This means that the representation of variants normalised by OpenCGA may differ from those from other tools.

Table of Contents:

Normalisation Procedure in OpenCGA v2.0

The normalisation procedure implemented by OpenCGA has been designed to resolve ambiguous representations commonly found in VCF data. The OpenCGA variant data model is not constrained by the VCF specification. This allows OpenCGA to represent some genotypes that are difficult for VCF to represent. Normalisation assumes correct VCF input according to the VCF specification, e.g. variant positions are 1-based.

The primary aim of OpenCGA normalisation is to standardise variant representation for storage and annotation within the OpenCGA database. A side effect of the ability to export VCF from OpenCGA is that the database of can be used as a VCF normalisation and merging tool. If used in this way users must be mindful of limitations of VCF in the correct representation of some variants.

Regardless of normalisation the original call as specified in the input VCF is stored by OpenCGA allowing the original record to be recapitulated if required.

Each step of the OpenCGA normalisation procedure as described below is performed sequentially on each record of the input VCF file.

1. Rename chromosomes

Due to the lack of standard for the chromosome naming it is common to see different labels for the same chromosome depending on the variant calling workflow. OpenCGA strips chromosome prefixes (*chrom*, *chrm*, *chr* and *ch*). For example, *chr1* and *chromX* are normalised to *1* and *X* respectively. Also, for mitochondria, the label *M* is normalised to *MT.*

2. Encode genotypes

VCF allows two different ways of representing the genotype alleles; with or without explicit allele sequence. OpenCGA normalises to the latter, i.e. an allele code is used instead of the allele itself: A 0 value represents the reference allele, and any other value is a 1-based index into the alternate alleles. A pseudo-VCF example of mapping from explicit to coded genotype alleles is shown in the following table:

	Input	Result
Encoding 1	`#CHR POS REF ALT S1 S2 S3 S4 S5` `1 100 A T A/A T/A A/T T|A T/.`	`#CHR POS REF ALT S1 S2 S3 S4 S5` `1 100 A T 0/0 0/1 0/1 1|0 ./1`

3. Split Multi-allelic records

Multi-allelic VCF records are produced in two main scenarios:

1. **Single-sample**: one sample (or individual) is multi-allelic for one specific position, ie. both chromosomes are mutated at the same position with a different allele.
2. **Multi-sample**: as a consequence of merging VCF from different samples, ie. different samples with different alleles come together in the same VCF record

Consider this multi-sample VCF input record at chromosome 1 position 100. It lists four samples with their genotypes being; homozygous reference [AA/AA], heterozygous SNP [AA/AT], heterozygous insertion [AT/AAC] and heterozygous deletion [AA/A]:

```
#CHROM POS    REF    ALT       FORMAT  SAMPLE1       SAMPLE2
SAMPLE3        SAMPLE4
1      100    AA     AT,AAC,A  GT:AD   0/0:40,1,0,0  0/1:19,20,1,0  2/1:0:
20,22,0  0/3:19,0,0,20
```

OpenCGA splits such multi-allelic record to create one output record for each alternate allele. Note that the multi-allelic nature of each record is maintained and allele-based fields are reordered. This is shown in the pseudo-VCF below;

```
#CHROM POS    REF    ALT       FORMAT  SAMPLE1       SAMPLE2
SAMPLE3        SAMPLE4
1      100    AA     AT,AAC,A  GT:AD   0/0:40,1,0,0  0/1:19,20,1,0  2/1:
0,20,22,0  0/3:19,0,0,20
1      100    AA     AAC,AT,A  GT:AD   0/0:40,0,1,0  0/2:19,1,20,0  1/2:
0,22,20,0  0/3:19,0,0,20
1      100    AA     A,AT,AAC  GT:AD   0/0:40,0,1,0  0/1:19,0,20,1  2/1:
0,0,20,22  0/3:19,20,0,0
```

Each output record from the split is then subjected *independently* to the remaining steps of the normalisation procedure. Normalisation in OpenCGA can therefore be considered "per allele", not "per position".

4. Left align alleles vs. reference

In the left alignment step the start position of the variant is shifted as far to the left as possible with respect to the reference sequence, "left aligned", as possible. The following example is adapted from the Centre for Statistical Genetics. Consider the following deletion record; its alignment against the reference shows that it falls within a short tandem repeat:

#CHROM POS	REF	ALT	POS: 12345678901234 REF: GGGCACACACAGGG ALT: A--
1 9	ACA	A	

Several other records could be proposed that would result in the exact same sequence change, for example;

#CHROM POS	REF	ALT	POS: 12345678901234
1 7	ACA	A	ALT: A--
1 5	ACA	A	ALT: A--
1 3	GCA	A	ALT: G--

OpenCGA normalises to the leftmost representation, i.e. the one with the smallest POS value (in this case POS=3).

For optimal left alignment especially in the case of insertions and deletions the flanking sequence of the reference genome is required. The reference genome can be specified with the OpenCGA parameter referenceGenome. Basic normalisation will still be performed if the parameter is omitted but it may be suboptimal.

5. Allele Trimming

Allele trimming consists on removing the leading (left trimming) and trailing (right trimming) bases that are identical in both reference and alternate alleles. Left trimming requires the variant position to be updated, for right trimming the variant position is unchanged.

Simple trimming

The following table shows a basic example of left and right trimming in pseudo-VCF notation.

	Input				Result			
Left trim	#CHROM 1	POS 100	REF AA	ALT AC	#CHROM 1	POS 101	REF A	ALT C
Right trim	#CHROM 1	POS 100	REF AA	ALT CA	#CHROM 1	POS 100	REF A	ALT C

Trimming InDels

Unlike VCF, variants in OpenCB do not require any "context base". Trimming can therefore result in empty strings for the reference or alternate alleles. The following table shows two valid representations of a deletion of 'T' at position 101 and the insertion of 'T' between positions 100 and 101. The table also shows how OpenCGA normalisation results in a unique variant for both deletion and insertion.

	Input				Result			
Deletion	#CHROM 1 1	POS 100 101	REF AT TC	ALT A C	#CHROM 1	POS 101	REF T	ALT -
Insertion	#CHROM 1 1	POS 100 101	REF A T	ALT AT TT	#CHROM 1	POS 101	REF -	ALT T

Trim rightmost first

For deletion or insertion in a region of repeated nucleotides the trimming operation can be done in multiple ways. For this input there are four possible ways to normalise the variant. OpenCGA ensures leftmost alignment by performing first the right trimming first

Input				Possible normalisations				OpenCGA result			
#CHR 1	POS 100	REF CTCTCA	ALT CTCA	#CHR 1 1 1 1	POS 100 101 102 103	REF CT TC CT TC	ALT - - - -	#CHR 1	POS 100	REF CT	ALT -

Example

OpenCGA represents variants internally as JSON objects, not pseudo-VCF records! Example JSON representation of the four variants resulting from normalisation of the single VCF record in the second table is shown on the Variant Normalization Example page. This example uses several of the procedures described above.

Decomposition of alleles

The OpenCGA normalisation process does not, other than for left alignment and trimming, edit individual alleles. For instance, decomposition of multi-nucleotide alleles into single-nucleotide primitives is not currently performed.

Identification of duplicate variants

A result of normalisation can be the identification of duplicated records (i.e. alleles) in a single file /sample. OpenCGA supports two deduplication policies: "Discard all" or "Max qual". The former discards b oth duplicates whilst the latter retains the duplicate with the highest quality score. In both cases a warning is logged.

Skip normalization

In certain scenarios the normalisation process could be undesired. This process can be skipped in OpenCGA with the option normalizationSkip. Use of this option is strongly discouraged.