The SAM Format Specification (v1.3-r837)

The SAM Format Specification Working Group

November 18, 2010

1 The SAM Format Specification

SAM stands for Sequence Alignment/Map format. It is a TAB-delimited text format consisting of a header section, which is optional, and an alignment section. If present, the header must be prior to the alignments. Header lines start with '`@`', while alignment lines do not. Each alignment line has 11 mandatory fields for essential alignment information such as mapping position, and variable number of optional fields for flexible or aligner specific information.

1.1 An example

Suppose we have the following alignment with bases in lower cases clipped from the alignment. Read `r001/1` and `r001/2` constitute a read pair; `r003` is a chimeric read; `r004` represents a split alignment.

```
Coor     12345678901234  5678901234567890123456789012345
ref      AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT

+r001/1        TTAGATAAAGGATA*CTG
+r002         aaaAGATAA*GGATA
+r003       gcctaAGCTAA
+r004                     ATAGCT..............TCAGC
-r003                            ttagctTAGGC
-r001/2                                        CAGCGCCAT
```

The corresponding SAM format is:

```
@HD VN:1.3 SO:coordinate
@SQ SN:ref LN:45
r001 163 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002   0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAGGATA    *
r003   0 ref  9 30 5H6M       *  0   0 AGCTAA   *   NM:i:1
r004   0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003  16 ref 29 30 6H5M       *  0   0 TAGGC    *   NM:i:0
r001  83 ref 37 30 9M         =  7 -39 CAGCGCCAT         *
```

1.2 Terminologies and Concepts

Template A DNA/RNA sequence part of which is sequenced on a sequencing machine or assembled from raw sequences.

Fragment A contiguous (sub)sequence on a template which is sequenced or assembled. For sequencing data, fragments are indexed by the order in which they are sequenced. For fragments of an assembled sequence, they are indexed by the order of the leftmost coordinate on the assembled sequence.

Read A raw sequence that comes off a sequencing machine. A read may consist of multiple fragments.

1-based coordinate system A coordinate system where the first base of a sequence is one. In this coordinate system, a region is specified by a closed interval. For example, the region between the 3rd and the 7th bases inclusive is $[3, 7]$. The SAM and GFF formats are using the 1-based coordinate system.

0-based coordinate system A coordinate system where the first base of a sequence is zero. In this coordinate system, a region is specified by a half-closed-half-open interval. For example, the region between the 3rd and the 7th bases inclusive is $[2, 7)$. The BAM, BED, Wiggle and PSL formats are using the 0-based coordinate system.

Phred scale Given a probability $0 < p \leq 1$, the phred scale of p equals $-10 \log_{10} p$, rounded to the closest integer.

1.3 The header section

Each header line begins with character '`@`' followed by a two-letter record type code. In the header, each line is TAB-delimited and each data field follows a format '`TAG:VALUE`' where `TAG` is a two-letter string that defines the content and the format of `VALUE`. Each header line should match: `/^@[A-Za-z][A-Za-z](\t[A-Za-z][A-Za-z]:[ -~])+$/`.

The following table give the defined record types and tags. Tags with '*' are required when the record type is present.

Tag		Description
`@HD`		The header line. The first line if present.
	`VN`*	Format version. *Accepted format*: `/^[0-9]+\.[0-9]+$/`.
	`SO`	Sorting order of alignments. *Valid values*: `unknown` (default), `unsorted`, `queryname` and `coordinate`. For coordinate sort, the major sort key is the RNAME field, with order defined by the order of @SQ lines in the header. The minor sort key is the POS field. For alignments with equal RNAME and POS, order is arbitrary. All alignments with * in RNAME field follow alignments with some other value but otherwise are in arbitrary order.
`@SQ`		Reference sequence dictionary. The order of `@SQ` lines defines the alignment sorting order.
	`SN`*	Reference sequence name. Each `@SQ` line must have a unique `SN` tag. The value of this field is used in the alignment records in RNAME and PNEXT fields. Regular expression: `[!-)+-<>-~][!-~]*`
	`LN`*	Reference sequence length. *Range*: `[1,2^29-1]`
	`AS`	Genome assembly identifier.
	`M5`	MD5 checksum of the sequence in the uppercase, with gaps and spaces removed.
	`SP`	Species.
	`UR`	URI of the sequence. This value may start with one of the standard protocols, e.g http: or ftp:. If it does not start with one of these protocols, it is assumed to be a file-system path.

`@RG`		Read group. Unordered multiple lines are allowed.
	`ID`*	Read group identifier. Each `@RG` line must have a unique `ID`. The value of `ID` is used in the RG tags of alignment records. Must be unique among all read groups in header section. Read group IDs may be modified when merging SAM files in order to handle collisions.
	`CN`	Name of sequencing center producing the read.
	`DS`	Description.
	`DT`	Date the run was produced (ISO8601 date or date/time).
	`LB`	Library.
	`PI`	Predicted median insert size.
	`PL`	Platform/technology used to produce the read. *Valid values*: `ILLUMINA`, `SOLID`, `LS454`, `HELICOS` and `PACBIO`.
	`PU`	Platform unit (e.g. flowcell-barcode.lane for Illumina or slide for SOLiD). Unique identifier.
	`SM`	Sample. Use pool name where a pool is being sequenced.
`@PG`		Program.
	`ID`*	Program record identifier. Each `@PG` line must have a unique `ID`. The value of `ID` is used in the alignment `PG` tag and `PP` tags of other `@PG` lines. PG IDs may be modified when merging SAM files in order to handle collisions.
	`PN`	Program name
	`CL`	Command line
	`PP`	Previous `@PG-ID`. Must match another `@PG` header's `ID` tag. `@PG` records may be chained using `PP` tag, with the last record in the chain having no `PP` tag. This chain defines the order of programs that have been applied to the alignment. PP values may be modified when merging SAM files in order to handle collisions of PG IDs. The first PG record in a chain (i.e. the one referred to by the PG tag in a SAM record) describes the most recent program that operated on the SAM record. The next PG record in the chain describes the next most recent program that operated on the SAM record.
	`VN`	Program version
`@CO`		One-line text comment. Unordered multiple lines are allowed.

1.4 The alignment section: mandatory fields

Each alignment line has 11 mandatory fields. These fields always appear in the same order and must be present, but their values can be '0' or '*' (depending on the field) if the corresponding information is unavailable. The following table gives an overview of the mandatory fields in the SAM format:

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
2	FLAG	Int	$[0, 2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0, 2^{29}-1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0, 2^{8}-1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next fragment
8	PNEXT	Int	$[0, 2^{29}-1]$	Position of the mate/next fragment
9	TLEN	Int	$[-2^{29}+1, 2^{29}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	fragment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

1. QNAME: Query template NAME. Each template has a unique name.
2. FLAG: bitwise FLAG. Each bit is explained in the following table ('*' means no bits are set):

Bit	Description
0x1	template having multiple fragments in sequencing
0x2	each fragment properly aligned according to the aligner
0x4	fragment unmapped
0x8	next fragment in the template unmapped
0x10	SEQ being reverse complemented
0x20	SEQ of the next fragment in the template being reversed
0x40	the first fragment in the template
0x80	the last fragment in the template
0x100	secondary alignment
0x200	not passing quality controls
0x400	PCR or optical duplicate

- Bit 0x4 is the only reliable place to tell whether the fragment is unmapped. If 0x4 is set, no assumptions can be made about RNAME, POS, bits 0x2, 0x10 and 0x100 and the bit 0x20 of the next fragment in the template.
- If 0x40 and 0x80 are both set, the fragment is part of a linear template, but it is neither the first nor the last fragment. If both 0x40 and 0x80 are unset, the index of the fragment in the template is unknown. This may happen for a non-linear template or the index is lost in data processing.
- Bit 0x100 marks the alignment not to be used in certain analyses when the tools in use are aware of this bit.
- If 0x1 is unset, no assumptions can be made about 0x2, 0x8, 0x20, 0x40 and 0x80.

3. RNAME: Reference sequence NAME of the alignment. If `@SQ` header lines are present, RNAME (if not '*') must be present in one of the `SQ-SN` tag. An unmapped fragment without coordinate has a '*' at this field. However, an unmapped fragment may also have an ordinary coordinate such that it can be placed at a desired position after sorting. If RNAME is '*', no assumptions can be made about POS and CIGAR.

4. POS: 1-based leftmost mapping POSition of the first matching base. The first base in a reference sequence has coordinate 1. POS is set as 0 for an unmapped read without coordinate. If POS is 0, no assumptions can be made about RNAME and CIGAR.

5. MAPQ: MAPping Quality. It equals $-10\log_{10}\Pr\{\text{mapping position is wrong}\}$, rounded to the nearest integer. A value 255 indicates that the mapping quality is not available.

6. CIGAR: CIGAR string. The CIGAR operations are given in the following table (set '*' if unavailable):

Op	BAM	Description
`M`	0	alignment match (can be a sequence match or mismatch)
`I`	1	insertion to the reference
`D`	2	deletion from the reference
`N`	3	skipped region from the reference
`S`	4	soft clipping (clipped sequences present in SEQ)
`H`	5	hard clipping (clipped sequences NOT present in SEQ)
`P`	6	padding (silent deletion from padded reference)
`=`	7	sequence match
`X`	8	sequence mismatch

- S/H can only be the first or the last operation.

7. RNEXT: Reference sequence name of the NEXT fragment in the template. For the last fragment, the next fragment is the first fragment in the template. If `@SQ` header lines are present, RNEXT (if not '*' or '=') must be present in one of the `SQ-SN` tag. This field is set as '*' when the information is unavailable, and set as '=' if RNEXT is identical RNAME. If not '=' and the next fragment in the template has one primary mapping (see also bit 0x100 in FLAG), this field is

identical to RNAME of the next fragment. If the next fragment has multiple primary mappings, no assumptions can be made about RNEXT and PNEXT. If RNEXT is '*', no assumptions can be made on PNEXT and bit 0x20.

8. PNEXT: Position of the NEXT fragment in the template. Set as 0 when the information is unavailable. This field equals POS of the next fragment. If PNEXT is 0, no assumptions can be made on RNEXT and bit 0x20.

9. TLEN: signed observed Template LENgth. If all fragments are mapped to the same reference, the unsigned observed template length equals the number of bases from the leftmost mapped base to the rightmost mapped base. The leftmost fragment has a plus sign and the rightmost has a minus sign. The sign of fragments in the middle is undefined. It is set as 0 for single-fragment template or when the information is unavailable.

10. SEQ: fragment SEQuence. This field can be a '*' when the sequence is not stored. If not a '*', the length of the sequence must equal the sum of lengths of `M/I/S/=/X` operations in CIGAR. An '=' denotes the base is identical to the reference base. No assumptions can be made on the letter cases. Anything other than `A/C/G/T/=` is regarded as ambiguous base `N`.

11. QUAL: ASCII of base QUALity plus 33 (same as the quality string in the Sanger FASTQ format). A base quality is the phred-scaled base error probability which equals $-10\log_{10}\Pr\{\text{base is wrong}\}$. This field can be a '*' when quality is not stored. If not a '*', SEQ is not a '*' and the length of the quality string must equal the length of SEQ.

1.5 The alignment section: optional fields

All optional fields are presented in the `TAG:TYPE:VALUE` format where `TAG` is a two-character string that matches `/[A-Za-z][A-Za-z0-9]/`, `TYPE` is a casesensitive single letter which defines the format of `VALUE`:

Type	Regexp matching `VALUE`	Descrption
A	`[!-~]`	Printable character
i	`[-+]?[0-9]+`	Singed 32-bit integer
f	`[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?`	Single-precision floating number
Z	`[ !-~]+`	Printable string, including space
H	`[0-9A-F]+`	Hex string, high nybble first

Each `TAG` can only appear once in one alignment line. A `TAG` containing lowercase letters are reserved for end users.

Predefined tags are shown in the following table. You can freely add new tags, and if a new tag may be of general interest, you can email `samtools-help@lists.sourceforge.net` to add the new tag to the specification. Note that tags started with '`X`', '`Y`' and '`Z`' are reserved for local use and will not be formally defined in any future version of this specification.

Tag	Type	Description
X?	?	Reserved fields for end users (together with Y? and Z?)
AM	i	The smallest template-independent mapping quality of fragments in the rest
AS	i	Alignment score generated by aligner
BQ	Z	Offset to base alignment quality (BAQ), of the same length as the read sequence. At the i-th read base, $\mathrm{BAQ}_i = Q_i - (\mathrm{BQ}_i - 64)$ where Q_i is the i-th base quality.
CM	i	Edit distance between the color sequence and the color reference (see also NM)
CQ	Z	Color read quality on the original strand of the read. Same encoding as QUAL.
CS	Z	Color read sequence on the original strand of the read. Of the same length as CQ.
E2	Z	The 2nd most likely base calls. Of the same length as SEQ.
FI	i	The index of fragment in the template.
FS	Z	Fragment suffix.
LB	Z	Library. Value to be consistent with the header RG-LB tag if @RG is present.
H0	i	Number of perfect hits
H1	i	Number of 1-difference hits (see also NM)
H2	i	Number of 2-difference hits
HI	i	Query hit index, indicating the alignment record is the i-th one stored in SAM
IH	i	Number of stored alignments in SAM that contains the query in the current record
MD	Z	String for mismatching positions [TODO: add descriptions]
MQ	i	Mapping quality of the mate/next fragment
NH	i	Number of reported alignments that contains the query in the current record
NM	i	Edit distance to the reference, including ambiguous bases but excluding clipping
OQ	Z	Original base quality (usually before recalibration). Same encoding as QUAL.
OP	i	Original mapping position (usually before realignment)
OC	Z	Original CIGAR (usually before realignment)
PG	Z	Program. Value matches the header PG-ID tag if @PG is present.
PQ	i	Phred likelihood of the template, conditional on both the mapping being correct
PU	Z	Platform unit. Value to be consistent with the header RG-PU tag if @RG is present.
Q2	Z	Phred quality of the mate/next fragment. Same encoding as QUAL.
R2	Z	Sequence of the mate/next fragment in the template.
RG	Z	Read group. Value matches the header RG-ID tag if @RG is present in the header.
SM	i	Template-independent mapping quality
TC	i	The number of fragments in the template.
U2	Z	Phred probility of the 2nd call being wrong conditional on the best being wrong. The same encoding as QUAL.
UQ	i	Phred likelihood of the fragment, conditional on the mapping being correct

The GS, GC, GQ, CC, CP, MF, S2 and SQ are reserved for backward compatibility.

2 Recommended Practice for the SAM Format

This section describes the best practice for representating data in the SAM format. They are not required in general, but may be required by a specific software package for it to function properly.

1. The header section
   - 1.1 The `@HD` line should be present with the `SO` tag specified.
   - 1.2 The `@SQ` lines should be present if reads have been mapped.
   - 1.3 When a `RG` tag appears anywhere in the alignment section, there should be a single corresponding `@RG` line with matching `ID` tag in the header.
   - 1.4 When a `PG` tag appears anywhere in the alignment section, there should be a single corresponding `@PG` line with matching `ID` tag in the header.
2. Adjacent CIGAR operations should be different.
3. No alignments should be assigned mapping quality 255.
4. Unmapped reads
   - 4.1 For a unmapped paired-end or mate-pair read whose mate is mapped, the unmapped read should have RNAME and POS identical to its mate.
   - 4.2 If all fragments in a template are unmapped, their RNAME should be set as '*' and POS as 0.
   - 4.3 If POS plus the sum of lengths of `M/=/X/D/N` operations in CIGAR exceeds the length specified in the `LN` field of the `@SQ` header line (if exists) with an SN equal to RNAME, the alignment should be unmapped.
5. Multiple mapping
   - 5.1 When one fragment is present in multiple records, only one record should have the primary alignment flag bit (0x100) set. RNEXT and PNEXT point to the primary alignment of the next fragment.
   - 5.2 SEQ and QUAL of secondary alignments should be set to '*' to reduce the file size.
6. There should be no overlap between fragments of a read[1].
7. Optional tags:
   - 7.1 If the template has more than 2 fragments, the `TC` tag should be present.
   - 7.2 The `NM` tag should be present.

[1] Few/no existing aligners follow this practice.