SRA File Formats Guide

Version 1.1 10 Mar 2010

National Center for Biotechnology Information – National Library of Medicine
EMBL European Bioinformatics Institute
DNA Databank of Japan

1 Contents

2 Overview

This document reviews the file formats currently supported for deposit by the Sequence Read Archives (SRA) at NCBI, EBI, and DDBJ, and gives guidance to submitters about current and future file formats and policies regarding SRA submissions.

The SRA is one of the International Nucleotide Sequence Databases and this Collaboration (INSDC) sets policies and goals for the partner databases. This document is intended to be compatible with INSDC policies.

2.1 Goals

This document guides submitters of sequencing data in order to:

- Specify which data formats are currently supported by SRA.
- Enable submitters to validate and convert data prior submission to avoid unnecessary data transfers.
- Improve the speed of submission processing.
- Reduce the probability of failed submissions.
- Improve other services provided by SRA by freeing up time previously spend to correct and transform data.

This document guides depositors to the Archives so that they may:

- Understand how to prepare data for submission to one of the Archives.
- Know what formats are supported by Archives facilities or toolkits, and which ones may have to be developed by the user
- Understand why certain technical issues prevent the usage of certain file formats

2.2 Related Documents

- Illumina, Inc: Sequencing Analysis Software User Guide For Pipeline Version 1.4 and CASAVA Version 1.0, 2009. Documents the qseq format.
- Standard Flowgram Format (SFF): http://www.ncbi.nlm.nih.gov/Traces/trace.cgi?cmd=show&f=formats&m=doc&s=formats#sff
- SRA format description: http://www.ncbi.nlm.nih.gov/Traces/sra/static/SRAToolkit.pdf
- Standard Flowgram Format (SFF) documentation: http://www.ncbi.nlm.nih.gov/Traces/trace.cgi?cmd=show&f=formats&m=doc&s=formats#sff
- Sequence Read Format (SRF) homepage: http://srf.sourceforge.net

2.3 Revision History

- Reviewed by NCBI 2009-10-01
- Reviewed by EBI 2009-10-07
- Reviewed by DDBJ 2009-11-27

3 Input Formats

3.1 General Requirements

The SRA prefers to transact in container files. By "container", it is meant an unambiguous binary file. These are objects that contain both the data and a description or specification of the data. Containers have the following advantages:

- All data from a run is contained in one file, so *tar* is not needed.
- Data are indexed for random access
- Data are compressed so *gzip* and other compression utilities are discouraged.
- Data are streamable (can be read from one input handle)
- Data are self-identifying (file type can be interrogated with *file*)
- Data come with run-time configuration and execution parameters, including run date, instrument name, flowcell name, processing program and version, etc.

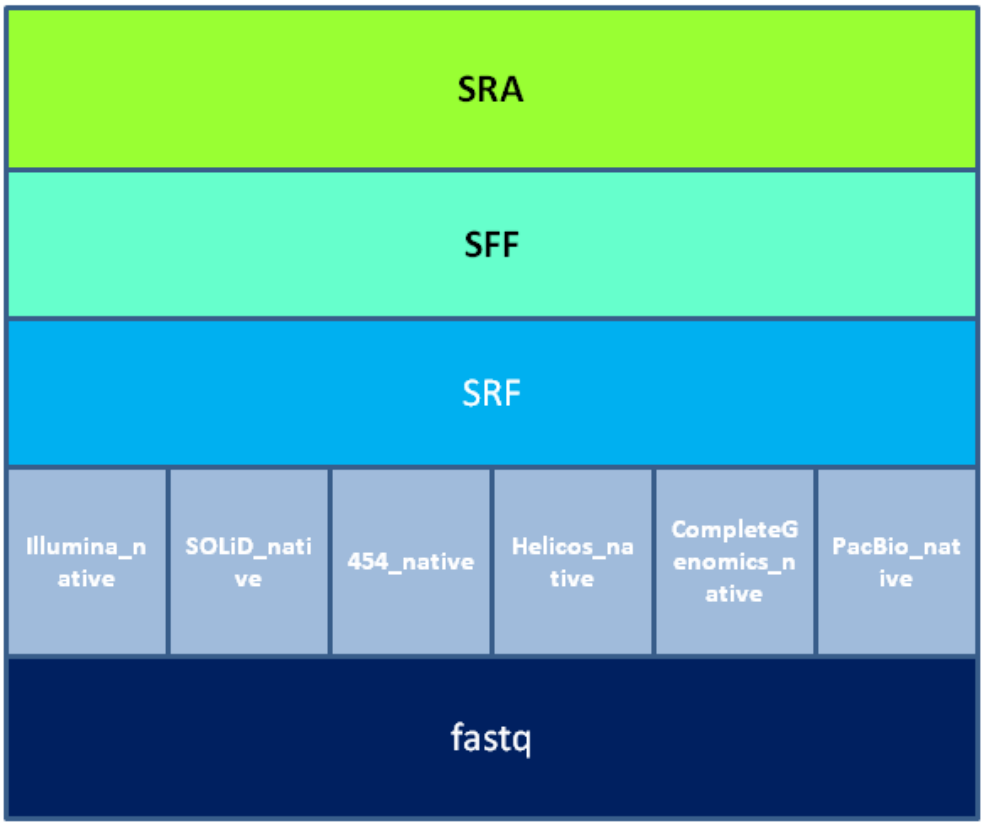

Figure 1 – Input file types supported by the SRA

Figure 1 shows the hierarchy of file types supported by the SRA on input. Table 1 shows what properties desirable in input data file formats are supported.

File model	Archive ready?	Stream-able on load?	Auxiliary data?	Run meta data?	Com-pressed?	Indexed?	Read names parseable ?	Read names indexable?
SRA	y	y	Y	Y	Y	Y	Y	Y
SFF	N	y	Y	Y	Y	Y	Y	Y
SRF	N	Y	Y	Y	Y	Y	Y	Y
native formats	N	N	Y	N	N	N	Y	Y
fastq	n	Y	N	N	N	N	N	N

Table 1 – Input file types and their general properties

3.2 Sequence Read Archive format (SRA)

SRA is the native archive format for the INSDC SRAs. Naturally, submitting data in this form gives the most complete rendering of the data, is fastest to load, and will suffer the fewest load failures and rework. Tools for preparing SRA submissions are being developed as well as the capability to accept them as part of a deposit.

3.3 Standard Flowgram Format (SFF)

SFF was developed by 454 and NCBI to encode 454 flowgrams for deposit in the Trace Archive. They have also worked well as an input file format for the SRA.

3.4 Sequence Read Format (SRF)

SRF is a community standard developed by James Bonfield and Asim Siddiqui. It has been used to contain large amounts of Illumina and SOLiD data for deposit and have served as a backing storage format. Several implementations exist. Io_lib based implementations maintained as part of the Staden package, as well as the AB implementation for SOLiD, have been qualified and successfully used for submission. An additional implementation based on Illumina changes to io_lib exists but has not yet been qualified. A new implementation from Rob Davies at Sanger Institute has been introduced but not yet qualified. The SRA Archives are working to qualify the various implementations of SRF.

3.5 Native Vendor Formats

Native formats have the advantage that they can be read directly and do not need software to be installed in order to read the file or extract its contents. However, while container formats

present many technical advantages, text formats suffer from a mirror set of disadvantages when considered in the context of Archive input and output.

Native formats have the following limitations :

- Data from a run may be situated in multiple files, so *tar* is needed.
- Data are not for random access
- Data are not compressed so *gzip* and other compression utilities are needed.
- Data are not generally streamable (can be read from one input handle)
- Data are not self-identifying (file type must be discerned by a full file scan). The file's suffix is NOT taken as a statement of its filetype.
- Data lack run-time configuration and execution parameters, including run date, instrument name, flowcell name, processing program and version, etc., so these must be delivered as metadata along with the file(s).

Vendor native formats are generally supported for a short time for new market entrants. Eventually support for these formats are retired as the technology matures and the vendor adopts archive-ready file formats suitable for submission by their customers.

Submitters may submit native vendor formats as described only without alternations.

The following general restrictions also apply:

- The number of base calls and quality scores must be equal.
- There cannot be two entries with the same read name.
- The file must contain at least one complete record.
- Quality zero is interpreted to mean no base call (N or .) and the base call is changed to A in such cases. Non-'N' with zero quality is changed to quality 1. 'N' with non-zero quality is changed to quality 0.
- Quality scores above 63 are saturated at 63.
- Negative values (-2, -1, -0) for coordinates are rounded to 0.

3.5.1 Illumina_native

Submitters may submit native data from the primary analysis output of the Illumina GA. The filetype is Illumina_native and constituent files for a run should be tarred together into a single tar file.
Illumina GA readname can be defined as follows:

<flowcell> ::= [a-zA-Z0-9_-]{2}+
<lane> ::= 1..8
<tile> ::= 1..1024
<X> ::= 0..4096
<Y> ::= 0..4096
<sep> ::= [_:\t]

READNAME ::= [<flowcell><sep> | s_]<lane><sep><tile><sep><x>sep><y>

The interpretation of the separator (<sep>) is right associative. Within a run DATA_BLOCK, records are must be grouped by tile. Runs are not recommended to span lane boundaries.

Records are fixed length.

BASES ::= [AaCcTtGgNn\.]+
QUALITIES ::= \!\"\#\$\%\&\'\(\)*\+,\-\.\/0-9:;<=>\?\@A-I]+
or
\@A-Z\[\\\]\^_`a-h]+

where the number of quality scores and base calls is the same in each read.

3.5.1.1 qseq

In Illumina pipeline versions 1.3 and later, the basecalling program Bustard emits a *_qseq.txt* file for the lane (two files for mate pairs). By default, this is the only sequencing data output by the primary analysis pipeline, and should be sufficient for most applications. Paired end data are presented in the orientation in which they will be aligned to a reference (5'-3'-3'-5'), which is the same orientation they were sequenced in.

Each record is one line with tab separator in the following format:

- Machine name: unique identifier of the sequencer.
- Run number: unique number to identify the run on the sequencer.
- Lane number: positive integer (currently 1-8).
- Tile number: positive integer.
- X: x coordinate of the spot. Integer (can be negative).
- Y: y coordinate of the spot. Integer (can be negative).
- Index: positive integer. No indexing should have a value of 1.
- Read Number: 1 for single reads; 1 or 2 for paired ends.
- Sequence (BASES)
- Quality: the calibrated quality string. (QUALITIES)
- Filter: Did the read pass filtering? 0 - No, 1 - Yes.

3.5.1.2 Illumina fastq

This format is emitted from Gerald, the secondary analysis pipeline. Illumina fastq contains the READNAME, index, read number parameters with quality basis character @.. Paired end data are presented in the orientation in which they will be aligned to a reference (5'-3'-3'-5'), which is the same orientation they were sequenced in.

The index and read number labels are defined as:

- Index: string. Currently 0, should be the index of the multiplexed sample in barcoded experiments, for example @EAS51_105_FC20G7EAAXX_R1:1:1:471:409#ATCACG/2
- Read Number: 1 for single reads; 1 or 2 for paired ends.

Formally,

@<READNAME>[#<index>]/<read_number>
<BASES>
+<READNAME>[#<index>]/<read_number>
<QUALITIES>

3.5.1.3 seq, prb, int

The _seq.txt, _prb.txt, and _int.txt files are emitted by Bustard, the primary analysis program. In Illumina pipeline versions 1.3 and earlier produced tab files in the following formats first defined in the 1.1 version of the GA pipeline:

The sequence text files (_seq.txt) have this format:
<READNAME>\t<BASES>

The qualities text files have four scores per base call (_prb.txt) in this format:
<READNAME>\t{%d %d %d %d}+ with value range [-40,40]

The intensity text files have four scores per base call (_int.txt) in this format:
<READNAME>\t{%5.1f %5.1f %5.1f %5.1f}+ with value range [-16384.0,16383.0]

Each of these files were either presented tile by tile, or in one file per lane. The number of records (lines) must be equal between the input files for a lane.Illumina pipeline versions 1.4 and later could only produce these files by running Bustard under non-default conditions.

3.5.1.4 Illumina scarf

Another text file output by Gerald analysis stage is a single colon separated file with one record per line containing read name, sequence, and quality.

3.5.2 SOLiD_native

SOLiD users may submit *csfasta* and *qual* as native SOLiD data, idenfitied as filetypes SOLiD_native_csfasta and SOLiD_native_qual, respectively. Primary analysis output of the SOLiD system is in color space. Paired end data are presented in the orientation in which they

will be aligned to a reference (5'-3'-5'-3'), which is the same orientation they were sequenced in.

Specific bindings for the AB SOLiD System are:

<flowcell> ::= [a-zA-Z0-9_-:]{2}+
<slide>::= 0..1
<panel> ::= 1..4096
<X> ::= 1..4096
<Y> ::= 1..4096
BASES ::= [TtGg][0123\.]+
QUALITIES ::= [0-9]+ | <quality>\s[0-9]+
<sep> ::= [_]
READNAME ::= <flowcell><sep><slide><sep><panel><sep><x>sep><y>
TAGNAME ::= <panel><sep><x><sep><y><sep><tag>

The interpretation of the separator (<sep>) is right associative. Reads are sorted in panel order within a given DATA_BLOCK. Runs should not span slide boundaries.

Records are fixed length.

The files have a header. This can be defined as:

HEADER ::=
<date> <path> [--flag]* --tag <tag> --minlength=<length> --prefix=<prefix> <path>
Cwd: <path>
Title: <flowcell>

The grammar for the csfasta file is:

HEADER
>TAGNAME
BASES

The grammar for the qual file is:

HEADER
>TAGNAME
QUALITIES

3.5.3 454_native

Specific bindings for the Roche/454 Genome Sequencer are:

The 454 read name is a 14 character alphanumeric string that encodes the plate, region and raster address of the read. The plate name is an encoding of a timestamp plus one character hash value that is virtually unique. The region is a two place decimal indicating the gasket division (there is always at least one gasket). The raster coordinate indicates the x and y coordinates on the plate modulus 4096 in base 36 encoding. Paired end data are presented in the orientation in which they will be aligned to a reference (5'-3'-5'-3'), which is the same orientation they were sequenced in.

<plate> ::= [A-Z0-9]{7}
<region> ::= [0-9]{2}
<xy> ::= [A..Z0..9]{5}
BASES ::= [ACGTN]+
QUALITIES ::= [0-9]+ | <quality>\s[0-9]+
READNAME ::= <plate><region><xy>

Reads are sorted in order of readname. Records are variable length.

The grammar for the 454 sequence file, identified as filetype 454_native_seq, is:

>READNAME
BASES

The grammar for the 454 qualities file, identified as filetype 454_native_qual, is:

>READNAME
QUALITIES

3.5.4 Helicos_native

Bindings for Helicos are:
<flowcell> ::= VHE-[0-9]+
<channel> ::= 1-25
<field> ::= 1-1100
<camera> :: [1234]
<position> :: 1-100000
<sep> ::= [-]
READNAME::= <flowcell><sep><channel><sep><field><sep><camera><sep><position>
QUALITIES ::= [!-I]+

A single record grammar is:

@READNAME
BASES
+
QUALITIES

For example,

@VHE-232481681003-9-1100-3-7971
ATCATTAAACATAAGTTTCAATCAACACTAATCATCAAC
+
//////////////////////////////////////

Records are variable length.

3.5.5 Other native formats

Other native formats may be developed as needed to support new marketplace entrants.

3.6 fastq

Because of the many benefits of container formats, INSDC SRAs intend to cease support for native and fastq forms by 2011.

Fastq is actually a style similar to "fasta" of presenting sequencing base calls and per-base quality scores in text form in one or more files. There are many variations so fastq cannot be called a "format".

A related form, "fasta", consists only of readname and base calls (no qualities). It too has many variants and dialects, principally in the way they treat the "defline" (line containing the readname).

3.6.1 Fastq acceptable to SRA

fastq has been narrowly defined with the goal of minimizing submission failures. However, this narrow definition may imply interpretation and conversion by the submitter.

The following terms are defined in general:

READNAME = Text string terminated by white space.
BASES ::= [ACGTNactgn.]+
QUALITIES ::= [0-9]+ | <quality>\s[0-9]+
or
[\!\"\#\$\%\&\'\(\)*\+,\-\.\/0-9:;<=>\?\@A-I]+
or
[\@A-Z\[\\\]\^_`a-h]+

The permissible fastq format is simply:

@READNAME
BASES
+[READNAME]
QUALITIES

where each instance of READNAMESPEC, CALLS, QUALITIES are separated newline.

The QUALITIES string can be defined numerically or using the 64 characters starting with ascii 33 (!). Formally,

<quality> ::= [0-9]+ | <quality>\s[0-9]+

or

<quality> ::= [!-I]+

3.7 fasta

The SRA does NOT accept fasta files only for submission as raw sequencing data. The base calls in fasta must be paired with a quality file that gives one quality score for each base/color call.

4 Understanding Variants

There exists a wide variety of text input files. These variants arise from simple syntactic changes, use of an alternate quality scoring system, and submission of data gathered from secondary rather than primary analysis phases.

4.1 Syntactic variants

Syntax variants can be classified as to whether one or more files are used to represent each data segment (subsequence of the raw spot sequence), and whether one or more files are used to represent distinct data series.

Singe Text – One text file represents all the data and all the records. For example,

```
@HWI-20090926_2:1:1:42:419/1
AAAAAAAACC
+HWI-20090926_2:1:1:42:419/1
IIIIIIIIII
@HWI-20090926_2:1:1:42:419/2
AAAAAAAACC
+HWI-20090926_2:1:1:42:419/2
IIIIIIIIII
```

or

```
@HWI-20090926_2:1:1:42:419#0/1
AAAAAAAACC
+HWI-20090926_2:1:1:42:419#0/1
IIIIIIIIII
@HWI-20090926_2:1:1:42:419#0/2
AAAAAAAACC
+HWI-20090926_2:1:1:42:419#0/2
IIIIIIIIII
```

Multi-text – One text file represents the sequence and qualities for a certain partition of the raw sequence, for example one file for the "forward" read, and another file for the "reverse" read or mate pair. The records in each file are arranged in the same readname sort order.

Poly-text – One text file represents each data series (sequence, qualities, signal) of the raw spot sequence. Poly-text representation can also be multi-text, where there are further divisions of these files into "forward" and "reverse" mate pairs.

4.2 Quality Scoring Variants

There are two scoring systems in use. The classic log probability of error (used by the *phred* base caller for capillary reads) is expressed as

$$-10 \log 1/p$$

where 1/p is the probability of error for that base call (or color call). An alternate formulation of quality measures the log ratio of frequency of event to frequency of non-event (log odds). This is expressed as :

$$-10 \log (p/(1-p))$$

The latter function can give negative values. These two functions are asymptotic for Q20 (one error expected per hundred base calls) or better, as can be seen from Figure 1.

4.3 Pipeline Variants

Input file variants may arise from capturing data output from different phases of analysis. The data may have been treated according to whether it aligned or assembled, or survived a filtration process specific to the application or problem that the sequencing was trying to address.

The SRA is intended as a repository of raw sequencing data. Submitters should follow these guidelines when selecting from data emitted by different analysis phases.

- You may submit quality scores that have been calibrated against a training set, but you should NOT submit sequencing data that has been filtered and scored by alignment to a particular reference sequence.
- You may screen (drop from the dataset) low quality reads but you should not omit reads that are of high quality that are sequenced from the same library preparation. An exception can be made for screening of human contamination where the disclosure of such sequence may violate privacy or usage restrictions. If the sequencing run was determined to be irrelevant to the experiment (not enough good reads of value), the entire run should be dropped from submission, not the portion that is "irrelevant."
- Never trim sequencing data of low quality bases.
- Never remove technical sequence such as bar codes, adapters, linkers. If these are sequenced along with the biological target, they are intended to remain in the raw spot sequence and annotated with start and stop indexes.
- Do not submit calibration lanes or plates (unless they are part of a methods paper).

4.4 Defline strings

Some files arrive with multiple strings on the defline separated by white space, as is commonly done with *fasta* files. Such text files cannot be processed by SRA, but it is straightforward to modify them for input by filtering the text file to retain only the first string encountered. For example:

```
cat my.fastq | awk '{ print $1 }' > my.good.fastq
```