GTF2.2: A Gene Annotation Format

(Revised Ensembl GTF)

Contents

Introduction

GTF stands for Gene transfer format. It borrows from GFF, but has additional structure that warrants a separate definition and format name.

Structure is as GFF, so the fields are:
<seqname> <source> <feature> <start> <end> <score> <strand> <frame> [attributes] [comments]

Here is a simple example with 3 translated exons. Order of rows is not important.

```
381 Twinscan  CDS          380   401   .   +   0  gene_id "001"; transcript_id "001.1";
381 Twinscan  CDS          501   650   .   +   2  gene_id "001"; transcript_id "001.1";
381 Twinscan  CDS          700   707   .   +   2  gene_id "001"; transcript_id "001.1";
381 Twinscan  start_codon  380   382   .   +   0  gene_id "001"; transcript_id "001.1";
381 Twinscan  stop_codon   708   710   .   +   0  gene_id "001"; transcript_id "001.1";
```

The whitespace in this example is provided only for readability. In GTF, fields must be separated by a single TAB and no white space.

Top

GTF Field Definitions

<seqname>
The name of the sequence. Commonly, this is the chromosome ID or contig ID. Note that the coordinates used must be unique within each sequence name in all GTFs for an annotation set.

<source>
The source column should be a unique label indicating where the annotations came from --- typically the name of either a prediction program or a public database.

<feature>
The following feature types are required: "CDS", "start_codon", "stop_codon". The features "5UTR", "3UTR", "inter", "inter_CNS", "intron_CNS" and "exon" are optional. All other features will be ignored. The types must have the correct capitalization shown here.

CDS represents the coding sequence starting with the first translated codon and proceeding to the last translated codon. Unlike Genbank annotation, the stop codon is not included in the CDS for the terminal exon. The optional feature "5UTR" represents regions from the transcription start site or beginning of the known 5' UTR to the base before the start codon of the transcript. If this region is interrupted by introns then each exon or partial exon is annotated as a separate 5UTR feature. Similarly, "3UTR" represents regions after the stop codon and before the polyadenylation site or end of the known 3' untranslated region. Note that the UTR features can only be used to annotate portions of mRNA genes, not non-coding RNA genes.

The feature "exon" more generically describes any transcribed exon. Therefore, exon boundaries will be the transcription start site, splice donor, splice acceptor and poly-adenylation site. The start or stop codon will not necessarily lie on an exon boundary.

The "start_codon" feature is up to 3bp long in total and is included in the coordinates for the "CDS" features. The "stop_codon" feature similarly is up to 3bp long and is excluded from the coordinates for the "3UTR" features, if used.

The "start_codon" and "stop_codon" features are not required to be atomic; they may be interrupted by valid splice sites. A split start or stop codon appears as two distinct features. All "start_codon" and "stop_codon" features must have a 0,1,2 in the <frame> field indicating which part of the codon is represented by this feature. Contiguous start and stop codons will always have frame 0.

The "inter" feature describes an intergenic region, one which is by almost all accounts not transcribed. The "inter_CNS" feature describes an intergenic conserved noncoding sequence region. All of these should have an empty transcript_id attribute, since they are not transcribed and do not belong to any transcript. The "intron_CNS" feature describes a conserved noncoding sequence region within an intron of a transcript, and should have a transcript_id associated with it.

<start> <end>
Integer start and end coordinates of the feature relative to the beginning of the sequence named in <seqname>. <start> must be less than or equal to <end>. Sequence numbering starts at 1. Values of <start> and <end> that extend outside the reference sequence are technically acceptable, but they are discouraged.

<score>
The score field indicates a degree of confidence in the feature's existence and coordinates. The value of this field has no global scale but may have relative significance when the <source> field indicates the prediction program used to create this annotation. It may be a floating point number or integer, and not necessary and may be replaced with a dot.

<frame>
0 indicates that the feature begins with a whole codon at the 5' most base. 1 means that there is one extra base (the third base of a codon) before the first whole codon and 2 means that there are two extra bases (the second and third bases of the codon) before the first codon. Note that for reverse strand features, the 5' most base is the <end> coordinate.

Here are the details excised from the GFF spec. **Important: Note comment on reverse strand.**

> '0' indicates that the specified region is in frame, i.e. that its first base corresponds to the first base of a codon. '1' indicates that there is one extra base, i.e. that the second base of the region corresponds to the first base of a codon, and '2' means that the third base of the region is the first base of a codon. **If the strand is '-', then the first base of the region is value of <end>**, because the corresponding coding region will run from <end> to <start> on the reverse strand.

Frame is calculated as (3 - ((length-frame) mod 3)) mod 3.

> (length-frame) is the length of the previous feature starting at the first whole codon (and thus the frame subtracted out).
> (length-frame) mod 3 is the number of bases on the 3' end beyond the last whole codon of the previous feature.
> 3-((length-frame) mod 3) is the number of bases left in the codon after removing those that are represented at the 3' end of the feature.
> (3-((length-frame) mod 3)) mod 3 changes a 3 to a 0, since three bases makes a whole codon, and 1 and 2 are left unchanged.

[attributes]
All nine features have the same two mandatory attributes at the end of the record:

> *gene_id value;* A globally unique identifier for the genomic locus of the transcript. If empty, no gene is associated with this feature.
> *transcript_id value;* A globally unique identifier for the predicted transcript. If empty, no transcript is associated with this feature.

These attributes are designed for handling multiple transcripts from the same genomic region. Any other attributes or comments must appear after these two and will be ignored.

Attributes must end in a semicolon which must then be separated from the start of any subsequent attribute by exactly one space character (NOT a tab character).

Textual attributes should be surrounded by doublequotes.

These attributes are required even for non-mRNA transcribed regions such as "inter" and "inter_CNS" features.

[comments]
Comments begin with a hash ('#') and continue to the end of the line. Nothing beyond a hash will be parsed. These may occur anywhere in the file, including at the end of a feature line.

Top

Examples

Here is an example of a gene on the negative strand including UTR regions. Larger coordinates are 5' of smaller coordinates. Thus, the start codon is 3 bp with largest coordinates among all those bp that fall within the CDS regions. Note that the stop codon lies between the 3UTR and the CDS

```
140    Twinscan    inter         5141     8522     .    -    .    gene_id ""; transcript_id "";
140    Twinscan    inter_CNS     8523     9711     .    -    .    gene_id ""; transcript_id "";
140    Twinscan    inter         9712     13182    .    -    .    gene_id ""; transcript_id "";
140    Twinscan    3UTR          65149    65487    .    -    .    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    3UTR          66823    66992    .    -    .    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    stop_codon    66993    66995    .    -    0    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    CDS           66996    66999    .    -    1    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    intron_CNS    70103    70151    .    -    .    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    CDS           70207    70294    .    -    2    gene_id "140.000"; transcript_id "140.000.1";
```

```
140    Twinscan    CDS           71696    71807    .    -    0    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    start_codon   71805    71806    .    -    0    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    start_codon   73222    73222    .    -    2    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    CDS           73222    73222    .    -    0    gene_id "140.000"; transcript_id "140.000.1";
140    Twinscan    5UTR          73223    73504    .    -    .    gene_id "140.000"; transcript_id "140.000.1";
```

Note the frames of the coding exons. For example:

> The first CDS (from 71807 to 71696) always has frame zero.
> Frame of the 1st CDS =0, length =112. (3-((length - frame) mod 3)) mod 3 = 2, the frame of the 2nd CDS.
> Frame of the 2nd CDS=2, length=88. (3-((length - frame) mod 3)) mod 3 = 1, the frame of the terminal CDS.
> Alternatively, the frame of terminal CDS can be calculated without the rest of the gene. Length of the terminal CDS=4. length mod 3 =1, the frame of the terminal CDS.

Note the split start codon. The second start codon region has a frame of 2, since it is the second base, and has an accompanying CDS feature, since CDS always includes the start codon.

Here is an example in which the "exon" feature is used. It is a 5 exon gene with 3 translated exons.

```
381 Twinscan  exon          150   200   .   +   .  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  exon          300   401   .   +   .  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  CDS           380   401   .   +   0  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  exon          501   650   .   +   .  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  CDS           501   650   .   +   2  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  exon          700   800   .   +   .  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  CDS           700   707   .   +   2  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  exon          900  1000   .   +   .  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  start_codon   380   382   .   +   0  gene_id "381.000"; transcript_id "381.000.1";
381 Twinscan  stop_codon    708   710   .   +   0  gene_id "381.000"; transcript_id "381.000.1";
```

Top

Scripts and Resources

Several Perl scripts have been written for checking, parsing, correcting, and comparing GTF-formatted annotations. Most of the important ones are included the Eval package, which comes equipped with a GTF parsing Perl package GTF.pm.

> Eval Software
> Eval Documentation

The Eval documentation contains a complete code-level documentation of GTF.pm, suitable for able Perl programmers to create and parse GTF files.

The script `validate_gtf.pl` included in the Eval package is particularly useful for checking that your GTF annotation is consistent and well-formed. This can also be done over the web:

> Web-based GTF Validator

Here are some more useful links:

> GFF Specification at Sanger
> Eval Homepage
> Brent Lab Homepage

Top