Projecting genes and variants mapped to previous assembly

When several assemblies and annotation layers are produced over the years (or by different teams) it is often necessary to map features such as genes or variants, or more generally, anything that can be encoded in a BED file, across them. This way it should be possible to trace which gene in version A corresponds to which gene(s) in version B. If we move beyond a particular feature you might be interested on, this touches on a long-standing issue of traceability among versions, which was also highlighted at Ensembl 4 Breeders (Feb 2019). I wonder whether genome version control software, inspired by git, could be produced for these tasks in the long-term. This might be related to genome patches, which I mentioned at Modelling pan-genomes and structural variation (PAV).

The tasks described below were performed originally with *solanum_lycopersicum_core_41_94_250* (assembly SL.50, in mirror db server) and *solanum_lycopersicum_core_42_95_3* (SL3.0) but should be applicable to other species and divisions.

Make sure you read Mapping variation data to GRCh38 as well, as this touches similar issues and actually benchmarks projecting vs remapping (which requires having BAM files at hand).

Another useful doc is *NOTES.projection.txt*

- Assembly Mapping (ATAC) with latest assembly in mysql-eg-prod-2 and previous taken from mirror

```
bsub -q production-rh7 -R "rusage[mem=16000]" -M 16000 -Is bash

# set virtual env for this python software (using Dan Bolser's settings)
source /nfs/production/panda/ensemblgenomes/development/dbolser/ATAC/virtualenv-15.1.0/atacVE/bin
/activate
eval "$(pyenv init -)"
pyenv shell atac_assembly_mapping

# move to wherever you have your github repos for development
cd ~/devel/

cd eg-assemblyconverter/
python atac_ensembl.py show-dbs solanum_lycopersicum
#['solanum_lycopersicum_core_41_94_250'] -> previous assembly in mirror db

# NOTE this impacts 4 tables: assembly (main), seq_region, coord_system, meta (lift-over)
# KEY: coord_system should show contigs from at least two different assembly versions after this
# KEY: default params mean that on average you in most cases alignments with identity >= 96% (we will
only use gapless segments)
python atac_ensembl.py map-current eg_prod3 solanum_lycopersicum_core_42_95_3 --fasta_dir $ensmapftp --
atac_dir $ensmapatac

lynx $ensmapatac/solanum_lycopersicum_core_41_94_250#SL2.50-TO-solanum_lycopersicum_core_42_95_3#SL3.0.
atac.uids.html
#FASTA  total sequence length % covered by mapping
#   SL2.50 823,630,941           89.25
#   SL3.0  827,747,456           88.80

# move to the current release ensembl repo
cd ../ensembl-96

# create chain files from new modified tables in db
# files placed at $ensmapatac/solanum_lycopersicum
# NOTE: output refers to seq_region names
perl ensembl/misc-scripts/assembly/ensembl_assembly_to_chain.pl \
        --host=mysql-eg-prod-2 --port=4239 --user=ensro --species=solanum_lycopersicum --dir=$ensmapatac
```

- **Heath checking**

  The relevant check is org.ensembl.healthcheck.testcase.generic.AssemblyMultipleOverlap, which is part of the Integrity HCs.

  This is concerned mostly with the assembly table (https://www.ensembl.org/info/docs/api/core/core_schema.html#assembly).

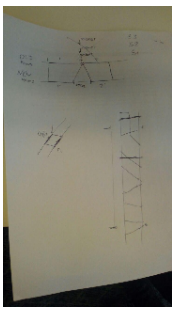
```
/* The assembly table allows to artificially chunk chromosome sequence into smaller parts.
Each row represents a component (ie contig), at least part of which is present in the golden path.
The part of the component that is in the path is delimited by fields cmp_start and cmp_end (start <
end);
the absolute position within the golden path chromosome (asm_seq_region_id) is given by asm_start and
asm_end. */

-- get all multiply-assembled components
 SELECT sr1.name AS cmp_sr_name, cs1.name AS cmp_cs, sr2.name AS asm_sr_name,
        cs2.name AS asm_cs, a.asm_seq_region_id, a.cmp_seq_region_id, COUNT(*) AS count
    FROM
        assembly a, seq_region sr1, seq_region sr2, coord_system cs1, coord_system cs2
    WHERE
        a.cmp_seq_region_id = sr1.seq_region_id AND
        a.asm_seq_region_id = sr2.seq_region_id AND
        sr1.coord_system_id = cs1.coord_system_id AND
        sr2.coord_system_id = cs2.coord_system_id
        GROUP BY asm_seq_region_id, cmp_seq_region_id, asm_start, cmp_start, ori HAVING count > 1;

-- list sorted golden path ranges, this will allow to detect segments that overlap, which cause the HC
failure
SELECT asm_start, asm_end, ori FROM assembly WHERE asm_seq_region_id=? AND cmp_seq_region_id=? ORDER BY
asm_start"
```

- **Mapping of gene stable_id** by quering mysql-eg-staging-2 (see https://www.ensembl.org/info/docs/api/core/core_schema.html#stable_id_event)

```
# extract gene coords from SL2.50
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_250 -e "SELECT s.name, g.seq_region_start, g.seq_region_end,
g.stable_id, g.seq_region_strand FROM gene g  INNER JOIN seq_region s ON g.seq_region_id = s.seq_region_id" -N
| perl -ne 's/\t-1$/\t-/; s/\t1$/\t+/; print' > solanum_lycopersicum_core_42_95_250.genes.bed

# project gene coords to current assembly
$LINUXBREW_HOME/bin/CrossMap.py bed $ensmapatac/solanum_lycopersicum/SL2.50_to_SL3.0.chain \
        solanum_lycopersicum_core_42_95_250.genes.bed solanum_lycopersicum_core_42_95_250.genes.mapped.bed

# how many mapped? 38254
wc -l solanum_lycopersicum_core_42_95_250.genes.mapped.bed

# how many left unmapped? 75
wc -l solanum_lycopersicum_core_42_95_250.genes.mapped.bed.unmap

# extract gene coords from SL3.0
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_3 -e "SELECT s.name, g.seq_region_start, g.seq_region_end, g.
stable_id, g.seq_region_strand FROM gene g  INNER JOIN seq_region s ON g.seq_region_id = s.seq_region_id ORDER
BY s.name, g.seq_region_start" -N | perl -ne 's/\t-1$/\t-/; s/\t1$/\t+/; print' >
solanum_lycopersicum_core_42_95_3.genes.bed

# check whether any mapping sessions have been recorded for this genome
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_3 -e "SELECT * FROM mapping_session"

# add new mapping session
export MAPSESSIONID=1
mysql-eg-staging-2-ensrw solanum_lycopersicum_core_42_95_3 -e "INSERT INTO mapping_session VALUES
($MAPSESSIONID,'solanum_lycopersicum_core_42_95_250','solanum_lycopersicum_core_42_95_3','94','95','SL2.
50','SL3.0',NOW());"
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_3 -e "SELECT * FROM mapping_session"

# intersect genes from SL2.50 to SL3.0
# produce 7-column file matching table stable_id_event: old_stable_id, old_version, new_stable_id, new_version,
mapping_session_id, type, score
# NOTE: version is set to NULL as that only makes sense to genes produced by genebuild
bedtools intersect -wo -a solanum_lycopersicum_core_42_95_250.genes.mapped.bed -b
solanum_lycopersicum_core_42_95_3.genes.bed | perl -lane '$map{$F[3]."\tNULL\t".$F[8]."\tNULL"} += $F[10]; END{
foreach (keys(%map)){ print "$_\t$ENV{MAPSESSIONID}\tgene\t$map{$_}" } }' | sort >
solanum_lycopersicum_42_95_250_to_42_95_3.stable_id.tsv

# check stable_id_events table
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_3 -e "SELECT * FROM stable_id_event"

# load new mappings into stable_id_events
mysql-eg-staging-2-ensrw solanum_lycopersicum_core_42_95_3 -e "LOAD DATA LOCAL INFILE
'solanum_lycopersicum_42_95_250_to_42_95_3.stable_id.tsv' INTO TABLE stable_id_event;"
mysql-eg-staging-2 solanum_lycopersicum_core_42_95_3 -e "SELECT * FROM stable_id_event LIMIT 5"
```

- **Projecting variations**

Here I will concentrate on SNPs and small indels, as transfering large structural variants (SV) across assemblies might not be safe.



NOTE: you must run org.ensembl.healthcheck.testgroup.EGVariationHandover after mapping variations.

```
bshell25

# extract variation coords and name from SL2.50
mysql-eg-staging-2 solanum_lycopersicum_variation_42_95_250 -e "SELECT s.name, g.seq_region_start, g.
seq_region_end, g.variation_name, g.seq_region_strand FROM variation_feature g INNER JOIN seq_region s ON g.
seq_region_id = s.seq_region_id" -N | perl -ne 's/\t-1$/\t-/; s/\t1$/\t+/; print' >
solanum_lycopersicum_core_42_95_250.variations.bed

# project variation coords to current assembly
time $LINUXBREW_HOME/bin/CrossMap.py bed $ensmapatac/solanum_lycopersicum/SL2.50_to_SL3.0.chain \
        solanum_lycopersicum_core_42_95_250.variations.bed solanum_lycopersicum_core_42_95_250.variations.
mapped.bed
#real    19m2.206s
#user    18m46.018s
#sys     0m6.171s

# how many mapped? 71099950
wc -l solanum_lycopersicum_core_42_95_250.variations.mapped.bed

# how many left unmapped? 58336
wc -l solanum_lycopersicum_core_42_95_250.variations.mapped.bed.unmap


# TO BE DONE
# make sure seg_region names from the old assembly are converted to the corresponding names of the newer one
```