VitisOmics

(see contributors below)

July 8, 2016

Contents

1 Overview

This document describes the "VitisOmics" project. This project aims at handling "omics" data in the genus Vitis (e.g. grapevine) in an open and reproducible way. Nevertheless, it requires some basic knowledge and skills about bioinformatics on GNU/Linux computers.

A large amount of such "omics" data are already available, and several committes from the IGGP (International Grape Genome Program) strive at improving interoperability. However, several issues

remain, among which:

- partially-overlapping data present at multiple locations (URGI, CRIBI, NCBI, EBI, etc);
- data downloadable as files in various formats not always easy to interchange (fasta, genbank, gff3, gtf, etc);
- data often available without meta-data inside the file;
- large files not always available in compressed form;
- main efforts dedicated to wet-labs grapevine biologists.

These have consequences in terms of ambiguity, inefficiency, potential mistakes, etc. Therefore, I hope that my attempt, via the usage of git and GitHub, could prove for the community to be a useful addition to the IGGP efforts.

The repository can be easily cloned from GitHub:

```
git clone https://github.com/timflutre/VitisOmics.git
```

The project directory is organized as advised by Noble (PLoS Computational Biology 2009). On any Unix-like system, it can be easily compressed and transferred (ignoring large data files):

```
cd ..; tar -czvf VitisOmics.tar.gz \
--exclude=VitisOmics/data --exclude=VitisOmics/results \
--exclude="*~" --exclude=".*" VitisOmics
```

In order to concretely promote collaborative editing in a distributed manner, the content of the "VitisOmics" repository should be based on plain text files. As a consequence, this document is written in the org format, and can thus be automatically exported, best by emacs, but also by pandoc, into the pdf and html formats for easy reading. A choice is also made to use as much as possible softwares widely available on any GNU/Linux computer, such as bash, awk, etc, but of course it may sometimes be much easier to use R (with Bioconductor), Python (with Biopython), etc.

Last but not least, feel free to contribute, by reporting issues or forking the repository!

1.1 Contributors

The person roles comply with R's guidelines (The R Journal Vol. 4/1, June 2012).

- Timothée Flutre (cre,aut)
- Gautier Sarah (ctb)
- ...

1.2 References

The NCBI has web pages about genome assembly:

- Assembly information;
- NCBI Genome Assembly Model;
- AGP specification.

The original article for grapevine (PN40024) is a must read:

- Jaillon, et al. The grapevine genome sequence suggests ancestral hexaploidization in major angiosperm phyla. Nature 449, 463-467 (2007).

About genome annotations:

- Grimplet et al. Comparative analysis of grapevine whole-genome gene predictions, functional annotation, categorization and integration of the predicted gene sequences. BMC Research Notes 5, 213+ (2012).
- Grimplet et al. The grapevine gene nomenclature system. BMC Genomics 15, 1077+ (2014).
- Vitulo et al. A deep survey of alternative splicing in grape reveals changes in the splicing machinery related to tissue, stress condition and genotype. BMC Plant Biology 14, 99+ (2014).

2 Data

```
mkdir -p data; cd data/
```

TODO: retrieve whole genome data from other cultivars than PN40024, e.g. Sultanina, Cabernet-Sauvignon, etc

2.1 URGI

- `https://urgi.versailles.inra.fr/Species/Vitis`

```
mkdir -p urgi; cd urgi/
../../src/download_urgi.bash
```

Remarks:

- when needed, the script decompresses zip files and compress them again but with gzip instead;
- the AGP file for the PN40024 8x assembly is not available at URGI's web site.

At the bottom of this page, one can find the following assemblies summary.

12x.2 assembly:

- contigs: 14642
- scaffolds (> 2kb): 2059
- mapped scaffolds: 366 (coverage of the genome: 95%)

12X.0 assembly:

- contigs: 14642
- scaffolds (> 2kb): 2059
- mapped scaffolds: 211 (coverage of the genome: 91.2%)

8X assembly:

- contigs: 19577
- scaffolds: 3514
- mapped ultracontigs: 191 (coverage of the genome: 68.9%)

N. Choisne from URGI (personal communication, 13/10/2015):

- "mapped scaffolds": scaffolds anchored on the linkage groups (i.e. chromosomes) using the markers from the reference genetic map;
- unmapped scaffolds hence are unanchored, and gathered into chrUn;
- "supercontig" is a synonym of "scaffold";
- "ultracontig": one level above supercontigs; localized and orientated according to data from BAC libraries.

N. Choisne from URGI on the 18K SNP chip (personal communication, 09/02/2016): let's take an example

- identifier `chr1_27655_C_T`;
- probe sequence `TTGTCTACGAAGTTTGACAATTTCTATTTTTCATAAGTTTACACAAATTA[T/C]TGAACAGTGAGTTAGTGAC`
- SNPs on the 18K Illumina Infinium microarray are named so that the first allele (here, `C`) corresponds to the reference (PN40024 12x.0) and the second allele (here, `T`) to the variant (almost sure for the "species" SNPs and should also be the case for the "vinifera" SNPs);

- in the sequence, the fact that the `T` comes before the `C` only reflects the fact that the chip is an Infinium II, with 2 beads, only allowing loci `A/G`, `A/C`, `T/G`, `T/C`, whereas the Infinium I only allows `A/T` et `C/G`.

2.2 NCBI

- `http://www.ncbi.nlm.nih.gov/genome/401`
- `ftp://ftp.ncbi.nlm.nih.gov/genomes/Vitis_vinifera/`

```
mkdir -p ncbi; cd ncbi/
../../src/download_ncbi.bash
```

Remarks:

- the important file `scaffold_names` provides the correspondence between original scaffold names (i.e. from the sequencing center) and various NCBI identifiers (RefSeq, GenBank, etc);
- in `ARCHIVE/`, `BUILD.1.1/` corresponds to the 8x genome sequences of PN40024.

2.3 EBI

```
mkdir -p ebi; cd ebi/
../../src/download_ebi.bash
```

Remarks:

- a genome soft-masked by RepeatMasker is available.

2.4 CRIBI

- `http://genomes.cribi.unipd.it/grape/`

```
mkdir -p cribi; cd cribi/
../../src/download_cribi.bash
```

2.5 Genoscope

- `http://www.genoscope.cns.fr/spip/Vitis-vinifera-whole-genome.html`

- http://www.genoscope.cns.fr/externe/Download/Projets/Projet_ML/data/

```
mkdir -p genoscope; cd genoscope/
../../src/download_genoscope.bash
```

3 Results

```
mkdir -p results; cd results/
```

TODO: compress fasta files with `bgzip` instead of `gzip`

3.1 Comparisons of original "assembly" files

3.1.1 URGI vs NCBI

Files from URGI:

```
cd urgi/
zcat VV_8X_embl_98_WGS_contigs.fsa.gz | grep -c ">" # 19577
zcat VV_8X_embl_98_Scaffolds.fsa.gz | grep -c ">" # 3514
zcat VV_chr8x.fsa.gz | grep -c ">" # 35
zcat VV_12X_embl_102_WGS_contigs.fsa.gz | grep -c ">" # 14642
zcat VV_12X_embl_102_Scaffolds.fsa.gz | grep -c ">" # 2059
zcat VV_chr12x.fsa.gz | grep -c ">" # 33
cat 12x0_chr.agp | wc -l # 390
cat 12x0_scaffolds.lg | wc -l # 2059
cat 12x0_chr.lg | wc -l # 33
zcat 12Xv2_grapevine_genome_assembly.fa.gz | grep -c ">" # 20
```

Files from NCBI:

```
cd ncbi/
ls ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr*.fa.gz | grep -v "Pltd" | while read f; do
    zcat $f; done | grep -c ">" # 3514
ls ARCHIVE/BUILD.1.1/Assembled_chromosomes/vvi_ref_chr*.fa.gz | while read f; do
    zcat $f; done | grep -c ">" # 19
zcat ARCHIVE/BUILD.1.1/allcontig.agp.gz | grep -v "#" | cut -f 5 | sort | uniq -c #
    F=1 N=16063 W=19577
cat ARCHIVE/BUILD.1.1/scaffold_names | sed 1d | wc -l # 3514
ls CHRS/vvi_ref_12X_chr*.fa.gz | grep -v -E "Pltd|MT" | while read f; do zcat $f;
    done | grep -c ">" # 2059
```

```
ls Assembled_chromosomes/vvi_ref_12X_chr*.fa.gz | grep -v -E "Pltd|MT" | while read
    f; do zcat $f; done | grep -c ">" # 19
cat scaffold_names | sed 1d | wc -l # 2061
```

See also the script `src/vitisomics.R` using R and Bioconductor. It confirms that the 12x scaffolds have the exact same sequence, whether they come from the URGI or the NCBI. Note however that the file from the NCBI allows to know easily on which chromosome a placed sequences is.

Remarks concerning PN40024 at URGI:

- the file `12x0_chr.agp.info` doesn't correspond to `12x0_chr.agp` (it doesn't even correspond to the description of a proper AGP file, as specified here);
- no mitochondrial nor chloroplastic data are available.

Remarks concerning PN40024 at NCBI:

- contig `NC_007957.1` in `ARCHIVE/BUILD.1.1/allcontig.agp.gz` (with `fragment_type=F` for "finished") corresponds to the chloroplast;
- `scaffold_names` contains all 2059 scaffolds of nuclear DNA as well as the assembled genome of the mitochondria and the chloroplast.

For its build 1.1 (corresponding to the 8x sequences of the PN40024 variety), the NCBI has one file per assembled chromosome. However, all unlocalized and unplaced scaffolds are gathered in a single file `chrUn`. This is not the case at URGI which has unlocalized scaffolds in files as `chr3_random` and a `chrUn_random` file with all unplaced scaffolds (and only them). Unfortunately, the NCBI has the annotation of the 8x (in the GenBank format), but the URGI hasn't.

3.1.2 URGI vs CRIBI

File from CRIBI:

```
tar -tzvf Genome12X.tar.gz | wc -l # 33
```

See also the script `src/vitisomics.R` using R and Bioconductor. It confirms that each of the 12x.0 chromosomes have the exact same sequence at URGI and CRIBI, the only differences being the headers.

3.2 Comparisons of annotations

3.2.1 By Genoscope and from Genoscope (8x and 12x.0)

The Genoscope annotated the 8x and 12x.v0 assemblies with the Gaze software:

```
cd data/genoscope/
zcat 8X/annotation/Vitis_vinifera_annotation_v1.gff.gz | md5sum # 8
    f6c98c2d3ac58fddea61d1073ad3b81
zcat 12X/annotation/Vitis_vinifera_annotation.gff.gz | md5sum # 2
    d568ed155422060dd2ca42eaf14bb3b
zcat 8X/annotation/Vitis_vinifera_annotation_v1.gff.gz | cut -f2 | sort | uniq -c #
    234890 Gaze_filter
zcat 12X/annotation/Vitis_vinifera_annotation.gff.gz | cut -f2 | sort | uniq -c #
    245272 Gaze
```

Both GFF files contain several types of annotations:

```
zcat 8X/annotation/Vitis_vinifera_annotation_v1.gff.gz | cut -f3 | sort | uniq -c
zcat 12X/annotation/Vitis_vinifera_annotation.gff.gz | cut -f3 | sort | uniq -c
```

assembly	gene	mRNA	UTR	CDS
8x	30434	30434	24671	149351
12x.0	26346	26346	35815	156765

For both 8x and 12x.0 assemblies, gene identifiers all start with `GSVIVG`, and by `GSVIVT` for mRNA:

```
zcat 8X/annotation/Vitis_vinifera_annotation_v1.gff.gz | awk -F"\t" '{if($3=="gene")
    {split($9,a," "); print substr(a[2],1,6)}}' | sort | uniq -c
zcat 8X/annotation/Vitis_vinifera_annotation_v1.gff.gz | awk -F"\t" '{if($3=="mRNA")
    {split($9,a," "); print substr(a[2],1,6)}}' | sort | uniq -c
zcat 12X/annotation/Vitis_vinifera_annotation.gff.gz | awk -F"\t" '{if($3=="gene"){
    split($9,a," "); print substr(a[2],1,6)}}' | sort | uniq -c
zcat 12X/annotation/Vitis_vinifera_annotation.gff.gz | awk -F"\t" '{if($3=="mRNA"){
    split($9,a," "); print substr(a[2],1,6)}}' | sort | uniq -c
```

3.2.2 By Genoscope but from URGI (8x and 12x.0)

The annotations made by Genoscope for the 12x.0 are also available for download at URGI (exact same file):

```
cd data/urgi/12x_annotation_Genoscope_V0/
zcat Vitis_vinifera_annotation.gff.gz | wc -l # 245272
zcat Vitis_vinifera_annotation.gff.gz | md5sum # 2d568ed155422060dd2ca42eaf14bb3b
```

The URGI also provides more specific files, e.g. mRNA-only, peptide-only, as well as repeat annotations by RepeatMasker and TRF.

3.2.3 By CRIBI but from URGI (12x.0)

TODO

3.2.4 By CRIBI and from CRIBI

The CRIBI makes available many versions of its annotations (0, 1, 2, 2.1) and several README files give some details about the differences.

```
tar -xzOf data/cribi/GFF/V0.tar.gz V0/all.GAZE | wc -l # 245276
tar -xzOf ../../data/cribi/GFF/V1.tar.gz | wc -l # 244596
zcat data/cribi/GFF/V1_phase.gff3.gz | wc -l # 392377
zcat data/cribi/V2/V2/V2.gff3.gz | wc -l # 821809
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | wc -l # 820944
```

Look at the latest (2.1), which seems to be a correct GFF3 file, even though it has no meta-data:

```
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | md5sum # fd5bd711563892ab42c50d77a27458dc
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | grep -c "##" # 0 meta-data
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | cut -f1 | sort | uniq -c | wc -l # 33 seqid:
   chr1 ...
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | cut -f2 | sort | uniq -c | wc -l # 4 sources:
    . EVM JIGSAWGAZE vitis_repeat_maskedN.fa
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | cut -f3 | sort | uniq -c | wc -l # 6 types
zcat data/cribi/V2/V2.1/V2.1.gff3.gz | awk -F "\t" '{split($9,a,";"); for(i in a){
   split(a[i],b,"="); print b[1]}}' | sort | uniq -c # ID Name Parent
```

type	count
gene	31845
mRNA	55564
exon	321050
five_prime_utr	58389
CDS	297312
three_prime_utr	56784

3.2.5 By NCBI (8x and 12x.0)

The NCBI annotated the 8x and 12x.0 assemblies with the GNOMON software.

TODO: convert NCBI annotation files from `gbs` into `gff3` (see work in progress below)

3.3 Manipulations of files from URGI

```
mkdir -p urgi; cd urgi/
```

3.3.1 Reformat sequence headers for VITVI_PN40024_8x_scaffolds_EMBL_r98

Launch script:

```
ln -s ../../data/urgi/VV_8X_embl_98_Scaffolds.fsa.gz .
echo "../../src/reformat_VV_8X_embl_98_Scaffolds.bash" \
  | qsub -cwd -j y -V -N reformat_VV_8X_embl_98_Scaffolds -q normal.q
```

Check:

```
zcat VV_8X_embl_98_Scaffolds.fsa.gz | wc -l # 8127179
zcat VV_8X_embl_98_Scaffolds.fsa.gz | grep -c ">" # 3514
zcat VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz | wc -l # 8127179
zcat VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz | grep -c ">" # 3514
diff <(zcat VV_8X_embl_98_Scaffolds.fsa.gz) <(zcat
   VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz | md5sum # 621197
   f19fee4a34e2f106ab5e6a485a
```

Length of each sequence:

```
zcat VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz \
  | awk 'BEGIN{RS=">"} {split($0,a,"\n");
if(length(a)==0)next;
sum=0; for(i=2;i<=length(a);++i){sum+=length(a[i])};
print a[1]": "sum; sumTot+=sum} END{print sumTot}'
```

3.3.2 Reformat sequence headers for VITVI_PN40024_8x_chroms_URGI

Launch script:

```
ln -s ../../data/urgi/VV_chr8x.fsa.gz .
echo "../../src/reformat_VV_chr8x.bash" \
  | qsub -cwd -j y -V -N reformat_VV_chr8x -q normal.q
```

Check:

```
zcat VV_chr8x.fsa.gz | wc -l # 8291865
zcat VV_chr8x.fsa.gz | grep -c ">" # 35
zcat VITVI_PN40024_8x_chroms_URGI.fa.gz | wc -l # 8291865
zcat VITVI_PN40024_8x_chroms_URGI.fa.gz | grep -c ">" # 35
diff <(zcat VV_chr8x.fsa.gz) <(zcat VITVI_PN40024_8x_chroms_URGI.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_8x_chroms_URGI.fa.gz | md5sum # 4b6ea1cb4ff189ac587fa269077885b5
```

Length of each sequence:

```
zcat VITVI_PN40024_8x_chroms_URGI.fa.gz \
  | awk 'BEGIN{RS=">"} {split($0,a,"\n");
if(length(a)==0)next; split(a,b," ");
sum=0; for(i=2;i<=length(a);++i){sum+=length(a[i])};
print b[1]": "sum; sumTot+=sum} END{print sumTot}'
```

header	length (bp)
chr1	15630816
chr10	9647040
$chr10_{random}$	2206354
chr11	13936303
$chr11_{random}$	1958407
chr12	18540817
$chr12_{random}$	2826407
chr13	15191948
$chr13_{random}$	1580403
chr14	19480434
$chr14_{random}$	5432426
chr15	7693613
$chr15_{random}$	4297576
chr16	8158851
$chr16_{random}$	4524411
chr17	13059092
$chr17_{random}$	1763011
chr18	19691255
$chr18_{random}$	5949186
chr19	14071813
$chr19_{random}$	1912523
$chr1_{random}$	5496190
chr2	17603400
$chr2_{random}$	60809
chr3	10186927
$chr3_{random}$	1343266
chr4	19293076
chr5	23428299
chr6	24148918
chr7	15233747
$chr7_{random}$	176143
chr8	21557227
$chr8_{random}$	12125
chr9	16532244
$chrUn_{random}$	154883714
total	497508771

3.3.3 Reformat sequence headers for `VITVI_PN40024_12x_v0_scaffolds_EMBL_r102`

Launch script:

```
ln -s ../../data/urgi/VV_12X_embl_102_Scaffolds.fsa.gz .
echo "../../src/reformat_VV_12X_embl_102_Scaffolds.bash" \
  | qsub -cwd -j y -V -N reformat_VV_12X_embl_102_Scaffolds -q normal.q
```

Check:

```
zcat VV_12X_embl_102_Scaffolds.fsa.gz | wc -l # 8091565
zcat VV_12X_embl_102_Scaffolds.fsa.gz | grep -c ">" # 2059
zcat VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz | wc -l # 8091565
zcat VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz | grep -c ">" # 2059
diff <(zcat VV_12X_embl_102_Scaffolds.fsa.gz) <(zcat
   VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz | md5sum # 4
   fa2432d7a66c019c7cb41ee4d0cb7bc
zcat VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz | grep -v ">" | md5sum #
   df5cdb0c6f73cb133261905374cdf2f2
```

3.3.4 Reformat sequence headers for `VITVI_PN40024_12x_v0_chroms_URGI`

Launch script:

```
ln -s ../../data/urgi/VV_chr12x.fsa.gz .
echo "../../src/reformat_VV_chr12x.bash" \
  | qsub -cwd -j y -V -N reformat_VV_chr12x -q normal.q
```

Check:

```
zcat VV_chr12x.fsa.gz | wc -l # 8240706
zcat VV_chr12x.fsa.gz | grep -c ">" # 33
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz | wc -l # 8240706
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz | grep -c ">" # 33
diff <(zcat VV_chr12x.fsa.gz) <(zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz | md5sum #
    eff315994fafe35333462b9595e10ce5
```

Length of each sequence:

```
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz \
  | awk 'BEGIN{RS=">"} {split($0,a,"\n");
if(length(a)==0)next; split(a[1],b," ");
sum=0; for(i=2;i<=length(a);++i){sum+=length(a[i])};
print b[1]": "sum; sumTot+=sum} END{print sumTot}'
```

header	length (bp)
chr1	23037639
$chr1_{random}$	568933
chr2	18779844
chr3	19341862
$chr3_{random}$	1220746
chr4	23867706
$chr4_{random}$	76237
chr5	25021643
$chr5_{random}$	421237
chr6	21508407
chr7	21026613
$chr7_{random}$	1447032
chr8	22385789
chr9	23006712
$chr9_{random}$	487831
chr10	18140952
$chr10_{random}$	789605
chr11	19818926
$chr11_{random}$	282498
chr12	22702307
$chr12_{random}$	1566225
chr13	24396255
$chr13_{random}$	3268264
chr14	30274277
chr15	20304914
chr16	22053297
$chr16_{random}$	740079
chr17	17126926
$chr17_{random}$	829735
chr18	29360087
$chr18_{random}$	5170003
chr19	24021853
chrUn	43154196
total	486198630

3.3.5 Reformat sequence headers for `VITVI_PN40024_12x_v2_chroms_URGI`

Launch script:

```
ln -s ../../data/urgi/12Xv2_grapevine_genome_assembly.fa.gz .
echo "../../src/reformat_12Xv2_grapevine_genome_assembly.bash" \
  | qsub -cwd -j y -V -N reformat_12Xv2_grapevine_genome_assembly -q normal.q
```

Check:

```
zcat 12Xv2_grapevine_genome_assembly.fa.gz | wc -l # 8103449
zcat 12Xv2_grapevine_genome_assembly.fa.gz | grep -c ">" # 20
zcat VITVI_PN40024_12x_v2_chroms_URGI.fa.gz | wc -l # 8103449
zcat VITVI_PN40024_12x_v2_chroms_URGI.fa.gz | grep -c ">" # 20
diff <(zcat 12Xv2_grapevine_genome_assembly.fa.gz) <(zcat
    VITVI_PN40024_12x_v2_chroms_URGI.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_12x_v2_chroms_URGI.fa.gz | md5sum # 4
    e487c28eaf19ef59b0b6128b73935af
```

Length of each sequence:

```
zcat VITVI_PN40024_12x_v2_chroms_URGI.fa.gz \
  | awk 'BEGIN{RS=">"} {split($0,a,"\n");
if(length(a)==0)next; split(a,b," ");
sum=0; for(i=2;i<=length(a);++i){sum+=length(a[i])};
print b[1]": "sum; sumTot+=sum} END{print sumTot}'
```

header	length (bp)
chr1	24233538
chr2	18891843
chr3	20695524
chr4	24711646
chr5	25650743
chr6	22645733
chr7	27355740
chr8	22550362
chr9	23006712
chr10	23503040
chr11	20118820
chr12	24269032
chr13	29075116
chr14	30274277
chr15	20304914
chr16	23572818
chr17	18691847
chr18	34568450
chr19	24695667
chrUkn	27389308
total	486205130

3.3.6 Format `VITVI_PN40024_8x_chroms_URGI` for BLAST

Launch:

```
echo "../../src/blast_format.bash VITVI_PN40024_8x_chroms_URGI.fa.gz" \
  | qsub -cwd -j y -V -q normal.q -N blast_format_VITVI_PN40024_8x_v0_chroms_URGI
```

3.3.7 Format `VITVI_PN40024_12x_v0_chroms_URGI` for BLAST

Launch:

```
echo "../../src/blast_format.bash VITVI_PN40024_12x_v0_chroms_URGI.fa.gz" \
  | qsub -cwd -j y -V -q normal.q -N blast_format_VITVI_PN40024_12x_v0_chroms_URGI
```

3.3.8 Format VITVI_PN40024_12x_v2_chroms_URGI for BLAST

Launch:

```
echo "../../src/blast_format.bash VITVI_PN40024_12x_v2_chroms_URGI.fa.gz" \
  | qsub -cwd -j y -V -q normal.q -N blast_format_VITVI_PN40024_12x_v2_chroms_URGI
```

3.3.9 Index VITVI_PN40024_12x_v0_chroms_URGI for BWA

Launch:

```
echo "../../src/bwa_index_VITVI_PN40024_12x_v0_chroms_URGI.bash" \
  | qsub -cwd -j y -V -N bwa_index_VITVI_PN40024_12x_v0_chroms_URGI -q normal.q
```

3.3.10 Index VITVI_PN40024_12x_v2_chroms_URGI for BWA

Launch:

```
echo "../../src/bwa_index_VITVI_PN40024_12x_v2_chroms_URGI.bash" \
  | qsub -cwd -j y -V -N bwa_index_VITVI_PN40024_12x_v2_chroms_URGI -q normal.q
```

3.3.11 Prepare VITVI_PN40024_12x_v2_chroms_URGI for SAMtools and Picard

Make an index as well as a SAM header.

Launch:

```
echo "../../src/samtools-picard_prep_VITVI_PN40024_12x_v2_chroms_URGI.bash" \
  | qsub -cwd -j y -V -N samtools-picard_prep_VITVI_PN40024_12x_v2_chroms_URGI -q
     normal.q
```

3.3.12 Index VITVI_PN40024_12x_v0_chroms_URGI for Bowtie2

Launch:

```
echo "../../src/bowtie2_index_VITVI_PN40024_12x_v0_chroms_URGI.bash" \
  | qsub -cwd -j y -V -N bowtie2_build_VITVI_PN40024_12x_v0_chroms_URGI -q normal.q
```

3.3.13 Index `VITVI_PN40024_12x_v2_chroms_URGI` for Bowtie2

Launch:

```
echo "../../src/bowtie2_index_VITVI_PN40024_12x_v2_chroms_URGI.bash" \
  | qsub -cwd -j y -V -N bowtie2_build_VITVI_PN40024_12x_v2_chroms_URGI -q normal.q
```

3.3.14 Index `VITVI_PN40024_12x_v2_chroms_URGI` for Bowtie2 compatible with Tassel

Tassel requires numbers as chromosome identifiers.

Launch:

```
echo "../../src/bowtie2_index_VITVI_PN40024_12x_v2_chroms_URGI_for_Tassel.bash" \
  | qsub -cwd -j y -V -N bowtie2_build_VITVI_PN40024_12x_v2_chroms_URGI_for_Tassel -
     q normal.q
```

3.3.15 Translate CRIBI annotations from 12x.0 to 12x.2

Requirement: use or write a script taking as input the 12x.0 GFF3 file as well as the 12.0-12.2 AGP file, and returns as output the 12x.2 GFF3 file

The URGI provides the following AGP file: `golden_path_V2_111113_allChr.csv`. Unfortunately, after looking at the official specification of the AGP format, the URGI file doesn't seem to be valid, neither for version 1.1, nor 2.2. After contacting URGI, they told me they were working on it (October 2015).

TODO: look at the annotations from CRIBI on 12x.0 transposed to 12x.2 by URGI

Another script was developped by G. Sarah, but it suffers from several issues.

TODO: test CrossMap

3.3.16 Convert SNP data of the 18K Illumina array from `xls` to `txt.gz`

On the command-line, working with tabulated files is much easier, and they should be compressed (e.g. with `gzip`). The 18071 probe sequences also need to be saved in two fasta files, one for the 13562 "vinifera" SNPs and one for the 4509 "species" SNPs.

See the corresponding task in the script `src/vitisomics.R`.

Note that, as of January 2016, even though at least two articles were published which used this genotyping array, the data are not (yet?) part of dbSNP nor EVA, unfortunately. Therefore, no unambiguous SNP identifiers exist which can be used across studies and genome assemblies.

3.3.17 Align Illumina probes on PN40024 assemblies

Requires the `VITVI_PN40024_12x_v0_chroms_URGI.fa.gz` bank to be formatted for BLAST (see above).

```
echo "zcat GrapeReSeq_Illumina_18K_SNP_vinifera_probes.fa.gz | blastn -query - -task
    megablast -db VITVI_PN40024_12x_v0_chroms_URGI -out /dev/stdout -outfmt 6 |
   gzip > Ill18Kprobes-vinifera_12x0-chroms_megablast.txt.gz" | qsub -cwd -j y -V -
   q normal.q -N blastn-megablast_Ill18Kprobes-vinifera_12x0-chroms
```

The alignments are analyzed in the corresponding task in the script `src/vitisomics.R`.

Among the 13562 "vinifera" probes, 33 are not aligned on the 12x0 assembly of the PN40024 genome, 24 are aligned on different chromosomes than indicated (mostly plastid genomes), and all the others look fine.

3.4 Manipulations of files from NCBI

```
mkdir -p ncbi; cd ncbi/
```

3.4.1 Reformat sequence headers for `VITVI_PN40024_8x_scaffolds_NCBI`

Launch script:

```
ls ../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr*.fa.gz | grep -v -E "Pltd" |
   while read f; do ln -s $f .; done
echo "../../src/reformat_scaffs_NCBI-8x.bash" \
  | qsub -cwd -j y -V -N reformat_scaffs_NCBI-8x -q normal.q
```

Check:

```
\ls vvi_ref_chr* | while read f; do zcat $f; done | wc -l # 6963886
\ls vvi_ref_chr* | while read f; do zcat $f; done | grep -c ">" # 3514
zcat VITVI_PN40024_8x_scaffolds_NCBI.fa.gz | wc -l # 6963886
zcat VITVI_PN40024_8x_scaffolds_NCBI.fa.gz | grep -c ">" # 3514
diff <(\ls -v vvi_ref_chr* | while read f; do zcat $f; done) <(zcat
   VITVI_PN40024_8x_scaffolds_NCBI.fa.gz)
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_8x_scaffolds_NCBI.fa.gz | md5sum #
    a66f86ab2d89eb582935454ae3b7a49d
```

3.4.2 Reformat sequence headers for VITVI_PN40024_12x_v0_scaffolds_NCBI

Launch script:

```
ls ../../data/ncbi/CHRS/vvi_ref_12X_chr*.fa.gz | grep -v -E "Pltd|MT" | while read f
    ; do ln -s $f .; done
echo "../../src/reformat_scaffs_NCBI-12x.bash" \
  | qsub -cwd -j y -V -N reformat_scaffs_NCBI-12x -q normal.q
```

Check:

```
\ls vvi_ref_12X_chr* | while read f; do zcat $f; done | wc -l # 6934292
\ls vvi_ref_12X_chr* | while read f; do zcat $f; done | grep -c ">" # 2059
zcat VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz | wc -l # 6934292
zcat VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz | grep -c ">" # 2059
diff <(\ls -v vvi_ref_12X_chr* | while read f; do zcat $f; done) <(zcat
    VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz) | less
```

Only the headers differ, not the sequences, so everything is fine.

Basic stats:

```
zcat VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz | md5sum # 20
    fa822ed5679519a20fe768c422a701
zcat VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz | grep -v ">" | md5sum # 9
    ddbb5761fe0e4356c7ef73410011ccb
```

3.4.3 Format VITVI_PN40024_8x_scaffolds_NCBI for BLAST

Launch:

```
echo "../../src/blast_format.bash VITVI_PN40024_8x_scaffolds_NCBI.fa.gz" \
  | qsub -cwd -j y -V -N blast_format_VITVI_PN40024_8x_scaffolds_NCBI -q normal.q
```

3.4.4 Format VITVI_PN40024_12x_v0_scaffolds_NCBI for BLAST

Launch:

```
echo "../../src/blast_format.bash VITVI_PN40024_12x_v0_scaffolds_NCBI.fa.gz" \
  | qsub -cwd -j y -V -N blast_format_VITVI_PN40024_12x_v0_scaffolds_NCBI -q normal.
     q
```

3.4.5 Convert gbs files to GFF3

Check that there is one LOCUS entry per scaffold:

```
ls ../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr*.gbs.gz | grep -v "Pltd" |
    while read f; do zcat $f; done | grep -c "LOCUS" # 3514
```

Use the $\text{bp}_{\text{genbank2gff3.pl}}$ script from BioPerl:

```
zcat ../../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr1.gbs.gz | bp_genbank2gff3
    .pl -in stdin -out stdout | gzip > vvi_ref_chr1.gff3.gz
# Error::throw
# Bio::Root::Root::throw /usr/local/share/perl5/Bio/Root/Root.pm:449
# Bio::SeqFeature::Tools::Unflattener::unflatten_seq /usr/local/share/perl5/Bio/
    SeqFeature/Tools/Unflattener.pm:1636
# main::unflatten_seq /usr/local/bin/bp_genbank2gff3.pl:1030
# /usr/local/bin/bp_genbank2gff3.pl:504
```

Use the $\text{convert}_{\text{genbanktogff3.py}}$ script from biocode:

```
zcat ../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr1.gbs.gz > vvi_ref_8x_chr1.
    gbs
convert_genbank_to_gff3.py -i vvi_ref_8x_chr1.gbs -o vvi_ref_8x_chr1.gff3 --no_fasta
# File "convert_genbank_to_gff3.py", line 196, in <module> main()
# File "convert_genbank_to_gff3.py", line 95, in main
# locus_tag = feat.qualifiers['locus_tag'][0]
# KeyError: 'locus_tag'
```

Additional remarks:

- it is written in Python;
- it uses Biopython, but also custom libraries;
- it is on GitHub;

- it doesn't handle gzipped file as input;
- it skips features not from type gene, mRNA, tRNA, rRNA and CDS.

Use the `GFF` library from BCBio (not yet integrated into Biopython) as explained here:

```
zcat ../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr1.gbs.gz > vvi_ref_8x_chr1.
    gbs
genbank_to_gff.py vvi_ref_8x_chr1.gbs
```

Remarks:

- the `sequence-region` are interspersed in the output file;
- what does the first data line correspond to, with source `annotation`?
- the source is present in the output as a feature;
- why is =feature = added in the 2nd field?
- why is it written `db_xref` instead of `Dbxref` (from official specification)?
- same for `note` instead of `Note`?
- exons seem to have 2nd field as `feature mRNA`

TODO: Use gffutils (doc, code)

TODO: Use a custom script based on Biopython only:

```
genbank2gff3.py -i ../../data/ncbi/ARCHIVE/BUILD.1.1/CHRS/vvi_ref_chr1.gbs.gz -o
    vvi_ref_8x_chr1.gff.gz -t 29760 -g "NCBI 1.1" -s Genbank
```

TODO: check for "pseudo" but empty

3.5 Creation of R/Bioconductor resources

- `http://www.bioconductor.org/`
- Huber, W. et al. Orchestrating high-throughput genomic analysis with bioconductor. Nature Methods 12, 115-121 (2015). URL `http://dx.doi.org/10.1038/nmeth.3252`.
- `http://bioconductor.org/packages/release/BiocViews.html#___Vitis_vinifera`

Reference genome sequences are made available via BSgenome packages, whereas annotations are made available via the AnnotationHub (as GRanges and TxDb objects).

3.5.1 BSgenome IGGP12Xv2 package

Retrieve the sequence data from URGI:

```
cd results/
mkdir -p make_BSgenome_IGGP12Xv2
cd make_BSgenome_IGGP12Xv2/
ln -s ../../data/urgi/12Xv2_grapevine_genome_assembly.fa.gz .
```

Split into one chromosome per file (in the headers, discard everything after the first space):

```
zcat 12Xv2_grapevine_genome_assembly.fa.gz | awk 'BEGIN{RS=">"} {if(NF==0)next;
    split($0,a,"\n"); split(a[1],b," "); print b[1]; print ">"b[1] > b[1]".fa"; for(
    i=2;i<length(a);++i){print a[i] >> b[1]".fa"}}'
gzip chr*.fa
```

Using the latest version of Bioconductor and its BSgenome package, prepare the seed file (`IGGP12Xv2_seed.txt`) by hand as indicated in the vignette as well as in the official R manual "Writing R extensions". Following this article, I chose the CC0 license (present in the R list of licenses in `share/licenses/license.db`). Following suggestions from Hervé Pagès (Bioconductor staff):

- the `common_name` field can be `Grape`;
- the `organism_biocview` field has to be `Vitis_vinifera` (see this link).

Forge the target package from the seed file:

```
echo "date; echo \"library(BSgenome); forgeBSgenomeDataPkg(\\\"IGGP12Xv2_seed.txt\\\
    "); sessionInfo()\" | R --vanilla; date" | qsub -cwd -j y -V -N forge_BSgenome -
    q normal.q
```

Build the package and check it:

```
echo "date; R CMD build BSgenome.Vvinifera.URGI.IGGP12Xv2; date" | qsub -cwd -j y -V
     -N build_BSgenome -q normal.q
echo "date; R CMD check BSgenome.Vvinifera.URGI.IGGP12Xv2_0.1.tar.gz; date" | qsub -
    cwd -j y -V -N check_BSgenome -q normal.q
```

The target package is now ready to be installed:

```
R CMD INSTALL BSgenome.Vvinifera.URGI.IGGP12Xv2_0.1.tar.gz
```

A.-F. Adam-Blondon (INRA, member of IGGP) and other colleagues also from INRA gave positive feedback. I hence sent the package to the Bioconductor team (Hervé Pagès, maintainer of the BSgenome generic package). The 12Xv2 package is now available here, and it also appears in this list.

3.5.2 BSgenome IGGP12Xv0 package

Similarly as for the 12Xv2 package, retrieve the sequence data from URGI:

```
cd results/
mkdir -p make_BSgenome_IGGP12Xv0
cd make_BSgenome_IGGP12Xv0/
ln -s ../../data/urgi/VV_chr12x.fsa.gz .
```

Split into one chromosome per file (headers as `chr1`, `chr1_random`, etc):

```
zcat VV_chr12x.fsa.gz | awk 'BEGIN{RS=">"} {if(NF==0)next; split($0,a,"\n"); split(a
    [1],b," "); print b[length(b)]; print ">"b[length(b)] > b[length(b)]".fa"; for(i
    =2;i<length(a);++i){print a[i] >> b[length(b)]".fa"}}'
gzip chr*.fa
```

Replace `chrUn` by `chrUkn` to be compatible with the 12Xv2:

```
zcat chrUn.fa.gz | sed 's/chrUn/chrUkn/' | gzip > chrUkn.fa.gz
diff <(zcat chrUn.fa.gz) <(zcat chrUkn.fa.gz) # check
rm chrUn.fa.gz
```

Prepare the seed file (`IGGP12Xv0_seed.txt`) using the one for `IGGP12Xv2` as a template.

Forge the target package from the seed file:

```
echo "date; echo \"library(BSgenome); forgeBSgenomeDataPkg(\\\"IGGP12Xv0_seed.txt\\\
    "); sessionInfo()\" | R --vanilla; date" | qsub -cwd -j y -V -N forge_BSgenome -
    q normal.q
```

Build the package and check it:

```
echo "date; R CMD build BSgenome.Vvinifera.URGI.IGGP12Xv0; date" | qsub -cwd -j y -V
     -N build_BSgenome -q normal.q
echo "date; R CMD check BSgenome.Vvinifera.URGI.IGGP12Xv0_0.1.tar.gz; date" | qsub -
    cwd -j y -V -N check_BSgenome -q normal.q
```

The target package is now ready to be installed:

```
R CMD INSTALL BSgenome.Vvinifera.URGI.IGGP12Xv0_0.1.tar.gz
```

The 12Xv0 package is now available here.

3.5.3 BSgenome IGGP8x package

Similarly as for the 12Xv2 and 12Xv0 packages, retrieve the sequence data from URGI:

```
cd results/
mkdir -p make_BSgenome_IGGP8X
cd make_BSgenome_IGGP8X/
ln -s ../../data/urgi/VV_chr8x.fsa.gz .
```

Split into one chromosome per file (headers as `chr1`, `chr1_random`, etc):

```
zcat VV_chr8x.fsa.gz | awk 'BEGIN{RS=">"} {if(NF==0)next; split($0,a,"\n"); split(a
    [1],b," "); print b[length(b)]; print ">"b[length(b)] > b[length(b)]".fa"; for(i
    =2;i<length(a);++i){print a[i] >> b[length(b)]".fa"}}'
gzip chr*.fa
```

Prepare the seed file (`IGGP8X_seed.txt`) using the one for `IGGP12Xv0` as a template:

```
cp ../make_BSgenome_IGGP12Xv0/IGGP12Xv0_seed.txt IGGP8X_seed.txt
```

Forge the target package from the seed file:

```
echo "date; echo \"library(BSgenome); forgeBSgenomeDataPkg(\\\"IGGP8X_seed.txt\\\");
     sessionInfo()\" | R --vanilla; date" | qsub -cwd -j y -V -N forge_BSgenome -q
    normal.q
```

Build the package and check it:

```
echo "date; R CMD build BSgenome.Vvinifera.URGI.IGGP8X; date" | qsub -cwd -j y -V -N
     build_BSgenome -q normal.q
echo "date; R CMD check BSgenome.Vvinifera.URGI.IGGP8X_0.1.tar.gz; date" | qsub -cwd
     -j y -V -N check_BSgenome -q normal.q
```

The target package is now ready to be installed:

```
R CMD INSTALL BSgenome.Vvinifera.URGI.IGGP8X_0.1.tar.gz
```

The 8X package is now available here (Bioc 3.3).

3.5.4 TxDb on IGGP12Xv0 from CRIBI (V2.1)

Set up the directory:

```
cd results/
mkdir -p make_TxDb_IGGP12Xv0_CRIBIv2-1
cd make_TxDb_IGGP12Xv0_CRIBIv2-1/
```

Make official GFF3 header:

```
echo -e "##gff-version 3" > V2.1_updated.gff3
tar -xzOf ../../data/cribi/Genome12X.tar.gz | awk 'BEGIN{RS=">"} {split($0,a,"\n");
    if(length(a)==0) next; seqlen=0; for(i=2;i<=length(a);++i){seqlen += length(a[i
    ])}; printf "##sequence-region "a[1]" 1 "seqlen"\n"}' | sort -k2,2V >> V2.1
    _updated.gff3
```

Concatenate the annotations below, using `chrUkn` as was done for the BSgenome packages:

```
zcat ../../data/cribi/V2/V2.1/V2.1.gff3.gz >> V2.1_updated.gff3
sed 's/chrUn/chrUkn/g' V2.1_updated.gff3 | gzip > V2.1_updated.gff3.gz
rm V2.1_updated.gff3
```

Convert the GFF3 to GRanges (R/Bioconductor objects): see `src/vitisomics.R`.

The resource is sent to Valerie Obenchain from Bioconductor (`valerie.obenchain@roswellpark.org`, as advised by Hervé Pagès).

The resource is available via the AnnotationHub as of May 26, 2016.

3.5.5 TxDb on IGGP12Xv0 from Genoscope

Set up the directory:

```
cd results/
mkdir -p make_TxDb_IGGP12Xv0_Genoscope
cd make_TxDb_IGGP12Xv0_Genoscope/
```

Convert the input GFF2 file into GFF3 (ignore UTRs):

```
ln -s ../../data/urgi/12x_annotation_Genoscope_V0/Vitis_vinifera_annotation.gff.gz .
zcat Vitis_vinifera_annotation.gff.gz | grep -v "##" | wc -l # 245272
ln -s ../urgi/VITVI_PN40024_12x_v0_chroms_URGI.fa.gz .
echo "../../src/genoscope_gff2_to_gff3.py --gff2 Vitis_vinifera_annotation.gff.gz --
    fa VITVI_PN40024_12x_v0_chroms_URGI.fa.gz --gff3 Vitis_vinifera_annotation.gff3.
    gz" \
  | qsub -cwd -j y -V -N stdout_genoscope_gff2_to_gff3 -q normal.q
zcat Vitis_vinifera_annotation.gff3.gz | grep -v "##" | wc -l # 209457
```

Use `chrUkn` as was done for the BSgenome packages:

```
zcat Vitis_vinifera_annotation.gff3.gz \
  | sed 's/chrUn/chrUkn/g' | gzip > Vitis_vinifera_annotation_updated.gff3.gz
```

Convert the GFF3 to GRanges (R/Bioconductor objects): see `src/vitisomics.R`.

The resource is sent to Valerie Obenchain from Bioconductor (`valerie.obenchain@roswellpark.org`, as advised by Hervé Pagès).

The resource is available via the AnnotationHub as of June 6, 2016.

3.5.6 TxDb on IGGP8X from Genoscope

Set up the directory:

```
cd results/
mkdir -p make_TxDb_IGGP8X_Genoscope
cd make_TxDb_IGGP8X_Genoscope/
```

Convert the input GFF2 file into GFF3 (ignore UTRs):

```
ln -s ../../data/genoscope/8X/annotation/Vitis_vinifera_annotation_v1.gff.gz .
zcat Vitis_vinifera_annotation_v1.gff.gz | grep -v "##" | wc -l # 234890
ln -s ../urgi/VITVI_PN40024_8x_chroms_URGI.fa.gz .
echo "../../src/genoscope_gff2_to_gff3.py --gff2 Vitis_vinifera_annotation_v1.gff.gz
    --fa VITVI_PN40024_8x_chroms_URGI.fa.gz --gff3 Vitis_vinifera_annotation_v1.
   gff3.gz" \
  | qsub -cwd -j y -V -N stdout_genoscope_gff2_to_gff3 -q normal.q
zcat Vitis_vinifera_annotation_v1.gff3.gz | grep -v "##" | wc -l # 210219
```

Use `chrUkn` as was done for the BSgenome packages:

```
zcat Vitis_vinifera_annotation_v1.gff3.gz \
  | sed 's/chrUn/chrUkn/g' | gzip > Vitis_vinifera_annotation_v1_updated.gff3.gz
```

Convert the GFF3 to GRanges (R/Bioconductor objects): see `src/vitisomics.R`.

The resource is sent to Valerie Obenchain from Bioconductor (`valerie.obenchain@roswellpark.org`, as advised by Hervé Pagès).

The resource is available via the AnnotationHub as of June 6, 2016.

3.5.7 TxDb on IGGP12Xv0 from NCBI

TODO: see the "Making New Organism Packages" vignette from the R/Bioconductor package "AnnotationForge"

See also the new genbank package from Bioc `http://www.bioconductor.org/packages/release/bioc/html/genbankr.html`

3.6 Whole genome alignments

- MUMmer is an appropriate software for this kind of analyzes
- as a recent example, see Schatz et al (2014) who compared the genome of 3 rice varieties

3.6.1 12x scaffolds on 12x0 chrs

Run chr by chr to avoid memory shortage:

```
cd results/mummer_ref-12x-chrs_qry-12x-scaffs
ln -s ../urgi/VITVI_PN40024_12x_v0_chroms_URGI.fa.gz .
zcat ../urgi/VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa.gz >
    VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa
chr=$(zcat VITVI_PN40024_12x_v0_chroms_URGI.fa | grep ">" | awk '{sub(">","",$1);
    print $1}' | head -1)
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz | awk 'BEGIN{RS=">"} /^'${chr}' /{print
    ">" $0}' > VITVI_PN40024_12x_v0_${chr}_URGI.fa
cmdNm="nucmer -maxmatch -p out-nucmer_${chr} VITVI_PN40024_12x_v0_${chr}_URGI.fa
    VITVI_PN40024_12x_v0_scaffolds_EMBL_r102.fa"
cmdDf="delta-filter -l 10000 -q out-nucmer_${chr}.delta > out-nucmer_${chr}_filter.
    delta"
cmdPl="mummerplot -postscript --filter -p out-nucmer_${chr}_filter -title ref-12x0-$
    {chr}_qry-12x-scaff out-nucmer_${chr}_filter.delta"
cmdSc="show-coords -c -l -L 10000 -r -T out-nucmer_${chr}_filter.delta | gzip > out-
    nucmer_${chr}_filter_coords.txt.gz"
echo ${cmdNm}"; "${cmdDf}"; "${cmdPl}"; "${cmdSc} | qsub -cwd -j y -V -N
    stdout_mummer_${chr} -q normal.q
```

3.6.2 8x scaffolds on 12x0 chrs

Run chr by chr to avoid memory shortage:

```
cd results/mummer_ref-12x-chrs_qry-8x-scaffs
ln -s ../urgi/VITVI_PN40024_12x_v0_chroms_URGI.fa.gz .
zcat ../urgi/VITVI_PN40024_8x_scaffolds_EMBL_r98.fa.gz >
   VITVI_PN40024_8x_scaffolds_EMBL_r98.fa
chr=$(zcat VITVI_PN40024_12x_v0_chroms_URGI.fa | grep ">" | awk '{sub(">","",$1);
   print $1}' | head -1)
zcat VITVI_PN40024_12x_v0_chroms_URGI.fa.gz | awk 'BEGIN{RS=">"} /^'${chr}' /{print
   ">" $0}' > VITVI_PN40024_12x_v0_${chr}_URGI.fa
cmdNm="nucmer -maxmatch -p out-nucmer_${chr} VITVI_PN40024_12x_v0_${chr}_URGI.fa
   VITVI_PN40024_8x_scaffolds_EMBL_r98.fa"
cmdDf="delta-filter -l 10000 -q out-nucmer_${chr}.delta > out-nucmer_${chr}_filter.
   delta"
cmdPl="mummerplot -postscript --filter -p out-nucmer_${chr}_filter -title ref-12x0-$
   {chr}_qry-8x-scaff out-nucmer_${chr}_filter.delta"
cmdSc="show-coords -c -l -L 10000 -r -T out-nucmer_${chr}_filter.delta | gzip > out-
   nucmer_${chr}_filter_coords.txt.gz"
echo ${cmdNm}"; "${cmdDf}"; "${cmdPl}"; "${cmdSc} | qsub -cwd -j y -V -N
   stdout_mummer_${chr} -q normal.q
```

3.6.3 PN40024-12x2 vs CabSauv-PacBio

TODO