Snakemake

Johannes Köster
Genome Informatics, Institute of Human Genetics,
Faculty of Medicine, University Duisburg-Essen

April 10, 2014

Structure

1. Motivation
2. Basic Idea
3. Advanced Features

Outline

Motivation

What we liked about GNU Make:

- text based
- rule paradigm
- lightweight

And what not:

- cryptic syntax
- limited scripting
- multiple output files
- scalability

Snakemake

- hook into python interpreter
- pythonic syntax for rule definition
- full python scripting
- scalability
- workflow specific functionality beyond Make basics
- stable community:

Outline

Syntax

```
SAMPLES = "500 501 502 503".split()

# require a bam for each sample
rule all:
    input:
        expand("{sample}.bam", sample=SAMPLES)

# map reads
rule map:
    input:
        "reference.bwt",
        "{sample}.fastq"
    output:
        "{sample}.bam"
    threads: 8
    shell:
        "bwa mem -t {threads} {input} | "   # refer to threads and input files
        "samtools view -Sbh - > {output}"   # refer to output files

# create an index
rule index:
    input:
        "reference.fasta"
    output:
        "reference.bwt"
    shell:
        "bwa index {input}"
```

Basic Usage

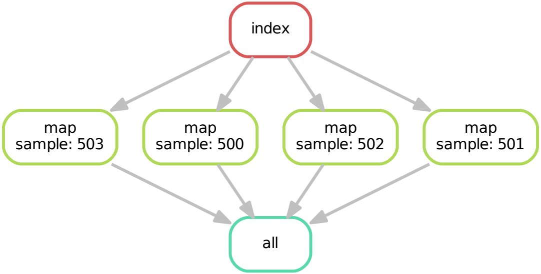
```
# perform a dry-run
$ snakemake -n

# execute the workflow using 8 cores
$ snakemake -j 8

# execute the workflow on a cluster (with up to 20 jobs)
$ snakemake -j 20 --cluster "qsub -pe threaded {threads}"
```

Visualization

```
# visualize the DAG of jobs
$ snakemake --dag | dot | display
```



Outline

Advanced Syntax

```
SAMPLES = "500 501 502 503".split()

rule all:
    input:
        expand("{sample}.bam", sample=SAMPLES)

# map reads with peanut
rule map:
    input:
        "reference.hdf5",
        "{sample}.fastq"
    output:
        "{sample}.bam"
    threads: 8
    resources: gpu=1  # define an additional resource
    version: shell("peanut --version")
    shell:
        "peanut map -t {threads} {input} | "
        "samtools view -Sbh - > {output}"

# create an index with peanut
rule index:
    input:
        "reference.fasta"
    output:
        "reference.hdf5"
    shell:
        "peanut index {input} {output}"
```

Scheduling

Maximize the number of running jobs with respect to

- priority
- number of descendants
- input size

while not exceeding

- provided cores
- provided resources

A multi-dimensional knapsack problem.

Sub-Workflows

```
SAMPLES = "500 501 502 503".split()

# define subworkflow
subworkflow:
    workdir: "../mapping"

rule all:
    input:
        expand("{sample}/results.xprs", sample=SAMPLES)

# estimate transcript expressions
rule express:
    input:
        REF,
        mapping("{sample}.bam")   # refer to output of subworkflow
    output:
        "{sample}/results.xprs"
    shell:
        "express {input} -o {wildcards.sample}"
```

HTML5 Reports

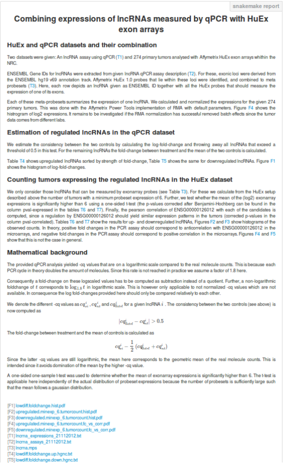
```
from snakemake.utils import report

rule report:
    input:
        T1="results.csv",
        F1="plot.pdf"
    output:
        html="report.html"
    run:
        report("""
        ==========
        Some Title
        ==========

        See table T1_, display
        some math

        .. math::
            |cq_0 - cq_1| > {MDIFF}

        """, output.html, **input)
```



snakemake report

Combining expressions of lncRNAs measured by qPCR with HuEx exon arrays

HuEx and qPCR datasets and their combination

Two datasets were given: An lncRNA assay using qPCR (T1) and 274 primary tumors analysed with Affymetrix HuEx exon arrays whithin the NRC.

ENSEMBL Gene IDs for lncRNAs were extracted from given lncRNA qPCR assay description (T2). For these, exonic loci were derived from the ENSEMBL hg19 v69 annotation track. Affymetrix HuEx 1.0 probes that lie within these loci were identified, and combined to meta probesets (T3). Here, each row depicts an lncRNA given as ENSEMBL ID together with all the HuEx probes that should measure the expression of one of its exons.

Each of these meta-probesets summarizes the expression of one lncRNA. We calculated and normalized the expressions for the given 274 primary tumors. This was done with the Affymetrix Power Tools implementation of RMA with default parameters. Figure F4 shows the histrogram of log2 expressions. It remains to be investigated if the RMA normalization has successful removed batch effects since the tumor data comes from different labs.

Estimation of regulated lncRNAs in the qPCR dataset

We estimate the consistency between the two controls by calculating the log-fold-change and throwing away all lncRNAs that exceed a threshold of 0.5 in this test. For the remaining lncRNAs the fold-change between treatment and the mean of the two controls is calculated.

Table T4 shows upregulated lncRNAs sorted by strength of fold-change, Table T5 shows the same for downregulated lncRNAs. Figure F1 shows the histogram of log-fold-changes.

Counting tumors expressing the regulated lncRNAs in the HuEx dataset

We only consider those lncRNAs that can be measured by exonarray probes (see Table T3). For these we calculate from the HuEx setup described above the number of tumors with a minimum probeset expression of 6. Further, we test whether the mean of the (log2) exonarray expressions is significantly higher than 6 using a one-sided t-test (the p-values corrected after Benjamini-Hochberg can be found in the column pval-expressed in the tables T6 and T7). Finally, the pearson correlation of ENSG00000126012 with each of the candidates is computed, since a regulation by ENSG00000126012 should yield similar expression patterns in the tumors (corrected p-values in the column pval-correlated). Tables T6 and T7 show the results for up- and downregulated lncRNAs, Figures F2 and F3 show histograms of the observed counts. In theory, positive fold changes in the PCR assay should correspond to anticorrelation with ENSG00000126012 in the microarrays, and negative fold changes in the PCR assay should correspond to positive correlation in the microarrays. Figures F4 and F5 show that this is not the case in general.

Mathematical background

The provided qPCR analysis yielded -cq values that are on a logarithmic scale compared to the real molecule counts. This is because each PCR cycle in theory doubles the amount of molecules. Since this rate is not reached in practice we assume a factor of 1.8 here.

Consequently a fold-change on these logscaled values has to be computed as subtraction instead of a quotient. Further, a non-logarithmic foldchange of t corresponds to $\log_{1.8} t$ in logarithmic scale. This is however only applicable to not normalized -cq values which are not available. In consequence the log fold-changes provided here should only be compared relatively to each other.

We denote the different -cq values as cq^i_{xi}, cq^i_{nt} and cq^i_{0ctrl} for a given lncRNA i . The consistency between the two controls (see above) is now computed as

$$|cq^i_{0ctrl} - cq^i_{nt}| > 0.5$$

The fold-change between treatment and the mean of controls is calculated as

$$cq^i_{xi} - \frac{1}{2}\left(cq^i_{0ctrl} + cq^i_{nt}\right)$$

Since the latter -cq values are still logarithmic, the mean here corresponds to the geometric mean of the real molecule counts. This is intended since it avoids domination of the mean by the higher -cq value.

A one-sided one-sample t-test was used to determine whether the mean of exonarray expressions is significantly higher than 6. The t-test is applicable here independently of the actual distribution of probeset expressions because the number of probesets is sufficiently large such that the mean follows a gaussian distribution.

[F1] lowdiff.foldchange.hist.pdf
[F2] upregulated.minexp_6.tumorcount.hist.pdf
[F3] downregulated.minexp_6.tumorcount.hist.pdf
[F4] upregulated.minexp_6.tumorcount.fc_vs_corr.pdf
[F5] downregulated.minexp_6.tumorcount.fc_vs_corr.pdf
[T1] lncrna_expressions_21112012.txt
[T2] lncrna_assays_21112012.txt
[T3] lncrna.mps
[T4] lowdiff.foldchange.up.hgnc.txt
[T5] lowdiff.foldchange.down.hgnc.txt

Data Provenance

Summarize output file status

```
$ snakemake --summary
```

```
file      date                        rule  version  status                   plan
500.bam   Thu Apr 10 10:55:17 2014    map   1.0      ok                       no update
501.bam   Thu Apr 10 10:55:17 2014    map   1.0      ok                       no update
502.bam   Thu Apr 10 10:55:17 2014    map   1.0      updated input files      update pending
503.bam   Thu Apr 10 10:55:17 2014    map   0.9      version changed to 1.0   no update
```

Trigger updates:

```
# update files with changed versions
$ snakemake -R `snakemake --list-version-changes`

# update files with changed code
$ snakemake -R `snakemake --list-code-changes`
```

Conclusion

Snakemake is a Make-like workflow system providing

- a readable syntax
- sophisticated scripting with python
- scalability from single-core to cluster
- support for hybrid computing
- data provenance
- modularization capabilities

Roadmap:

- DRMAA support
- a workflow or rule library

`http://bitbucket.org/johanneskoester/snakemake`

Köster, J., Rahmann, S., Snakemake – a scalable bioinformatics workflow engine. Bioinformatics 2012.