Spring 4-15-2016

ChIPathlon: A competitive assessment for gene regulation tools.

Avi Knecht
University of Nebraska-Lincoln, avi@kurtknecht.com

Adam Caprez
University of Nebraska-Lincoln, acaprez2@unl.edu

Istvan Ladunga
University of Nebraska-Lincoln, sladunga@unl.edu

ChIPathlon: a competitive assessment for gene regulation tools
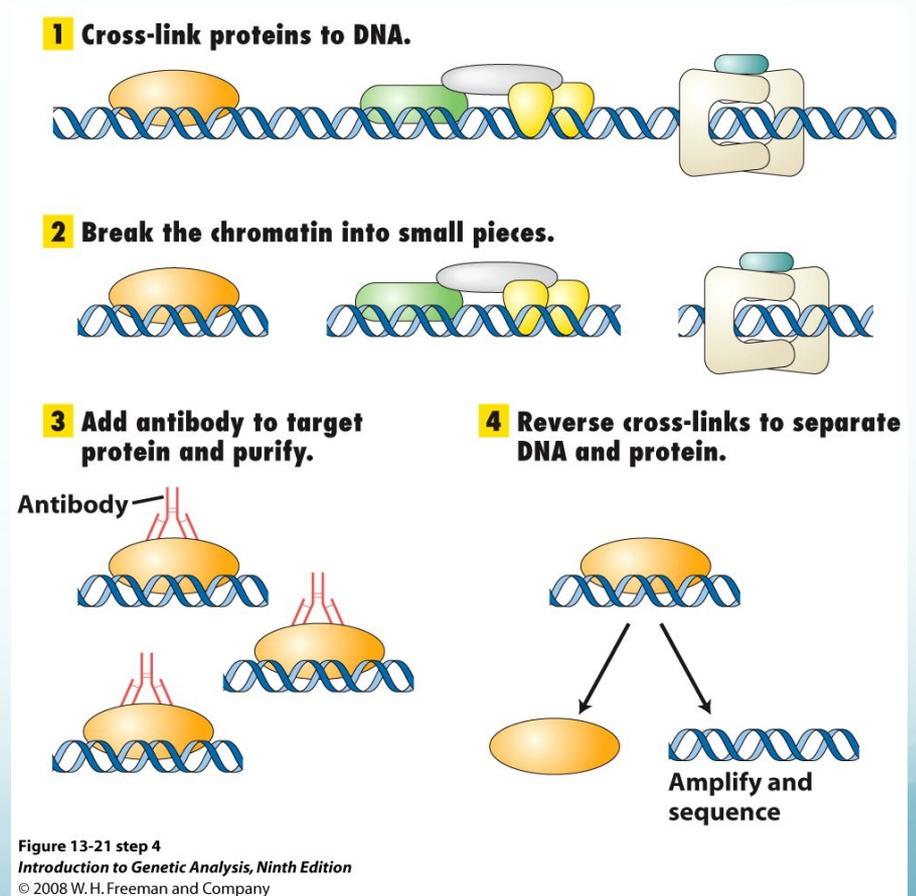
Avi Knecht, Adam Caprez, Istvan Ladunga

Gene regulation: why do we care?

- When gene regulation of the cell cycle malfunctions, it frequently causes cancer.

- Adult, differentiated cells can be reprogrammed to induced pluripotent stem cells
  - Which can then be reprogrammed to heart muscle, skin, etc, to repair damaged tissue (to limited extent in clinical practice)

Mapping transcription factors & histone modifications to genome

- Genes, are regulated by transcription factors and proteins, that bind to specific sequences on the DNA.

- Transcription factors are mapped to the DNA by chromatin immunoprecipitation followed by next-generation sequencing.

1 Cross-link proteins to DNA.

2 Break the chromatin into small pieces.

3 Add antibody to target protein and purify.

Antibody

4 Reverse cross-links to separate DNA and protein.

Amplify and sequence

Figure 13-21 step 4
Introduction to Genetic Analysis, Ninth Edition
© 2008 W. H. Freeman and Company

Critical steps in data analysis
Peak calling
Using strand dependent bimodality in peak calling
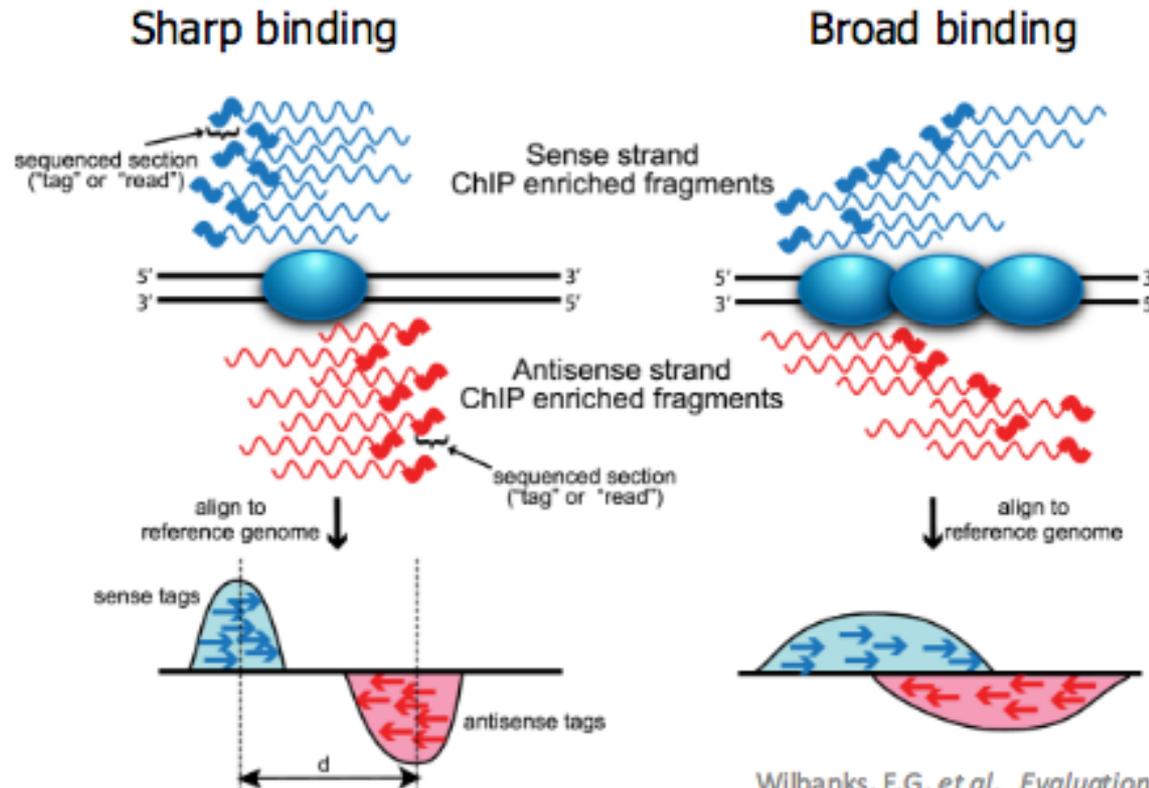
Wilbanks, E.G. *et al.* *Evaluation of Algorithm Performance in ChIP-Seq Peak Detection . PLoS ONE July (2010)*

Challenges in mapping transcription factors to the genome

- Background correction is an open problem.

- DNA fragments can be much larger than the binding site.

- Sequencing read location does not follow any statistical distribution.

- Over 50 different methods are used for mapping, which produce different results.

ChIPathlon

Evaluate the performance of all transcription factor mapping (peak calling) methods.
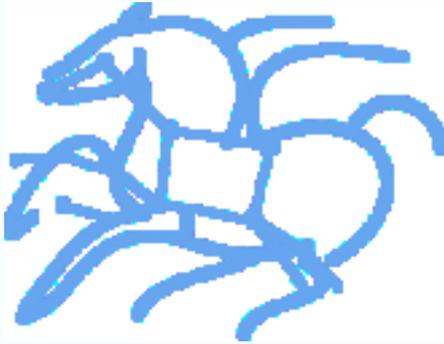
To this end, we will develop a scalable and easy to use super computing pipeline to stage data, compare many different peak calling and differential binding site tools, and store all results into a single database.

MongoDB

- Works well with large data sets.
- Can handle incomplete data.

Pegasus

- Too much data for manual processing, need to create workflows.
- Built on condor, which is used by a variety of super computing centers.

Python

- Many bioinformatics packages already managed in python under bioconda.
- Has interfaces for both MongoDB & Pegasus.

YAML to Workflows I

Each individual job is defined in a plain text YAML file.

```
1   zcat_awk_sort_peaks:
2     inputs:
3       - bed:
4           type: file
5     additional_inputs: null
6     outputs:
7       - bed:
8           type: file
9     command: zcat
10    arguments:
11      - "$inputs.0":
12          changeable: false
13          required: true
14          has_value: false
15      - "$outputs.0":
16          changeable: false
17          required: true
18          has_value: false
19    walltime: 2000
20    memory: 2000
21    cores: 1
22
```

YAML to Workflows II

Jobs are chained together by using module YAML files.

```yaml
 1  peak_call:
 2      - spp[tool]:
 3          - r_spp_nodups:
 4              inputs:
 5                  - exp.bed:
 6                      type: file
 7                  - control.bed:
 8                      type: file
 9              additional_inputs: null
10              outputs:
11                  - results.narrowPeak:
12                      type: file
13                  - results.pdf:
14                      type: file
15                  - results.ccscore:
16                      type: file
17          - zcat_awk_sort_peaks:
18              inputs:
19                  - results.narrowPeak:
20                      type: file
21              additional_inputs: null
22              outputs:
23                  - results_sorted.narrowPeak:
24                      type: file
```

YAML to Workflows III

Users need to input a file selecting ENCODE experiment id's for files to process, tools they want to use, and a path to a genome.

```yaml
 1  v  runs:
 2  v    - experiment: "ENCSR605MFS"
 3         align: bwa
 4         peak: spp
 5  v    - experiment: "ENCSR605MFS"
 6         align: bowtie2
 7         peak: spp
 8  v    - experiment: "ENCSR000ERE"
 9         align: bwa
10         peak: spp
11  v    - experiment: "ENCSR000EGZ"
12         align: bowtie2
13         peak: macs2
14  v  genomes:
15  v    bwa:
16         grch38p6: "/path/to/genome/base"
17  v    bowtie2:
18         grch38p6: "/path/to/genome/base"
```

Conclusion

- The current pipeline handles all downloads, alignment of single or paired end reads, and peak calling.
- The modularity of the underlying architecture makes it very easy to add additional tools or processing steps without changing the workflow generation code.
- Workflows can be generated for any ENCODE experiment, making this a very versatile pipeline for comparing bioinformatics tools.
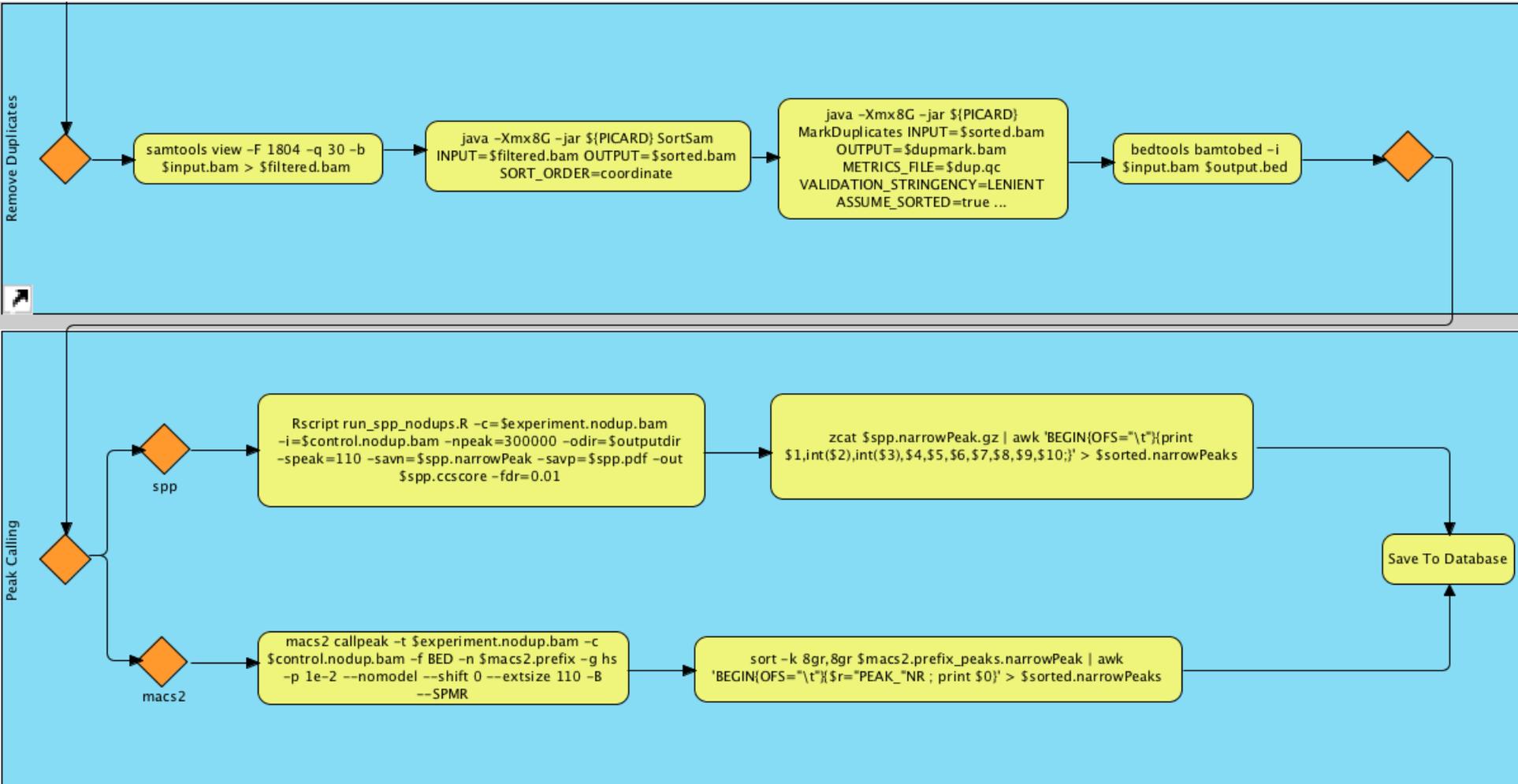
Database Architecture
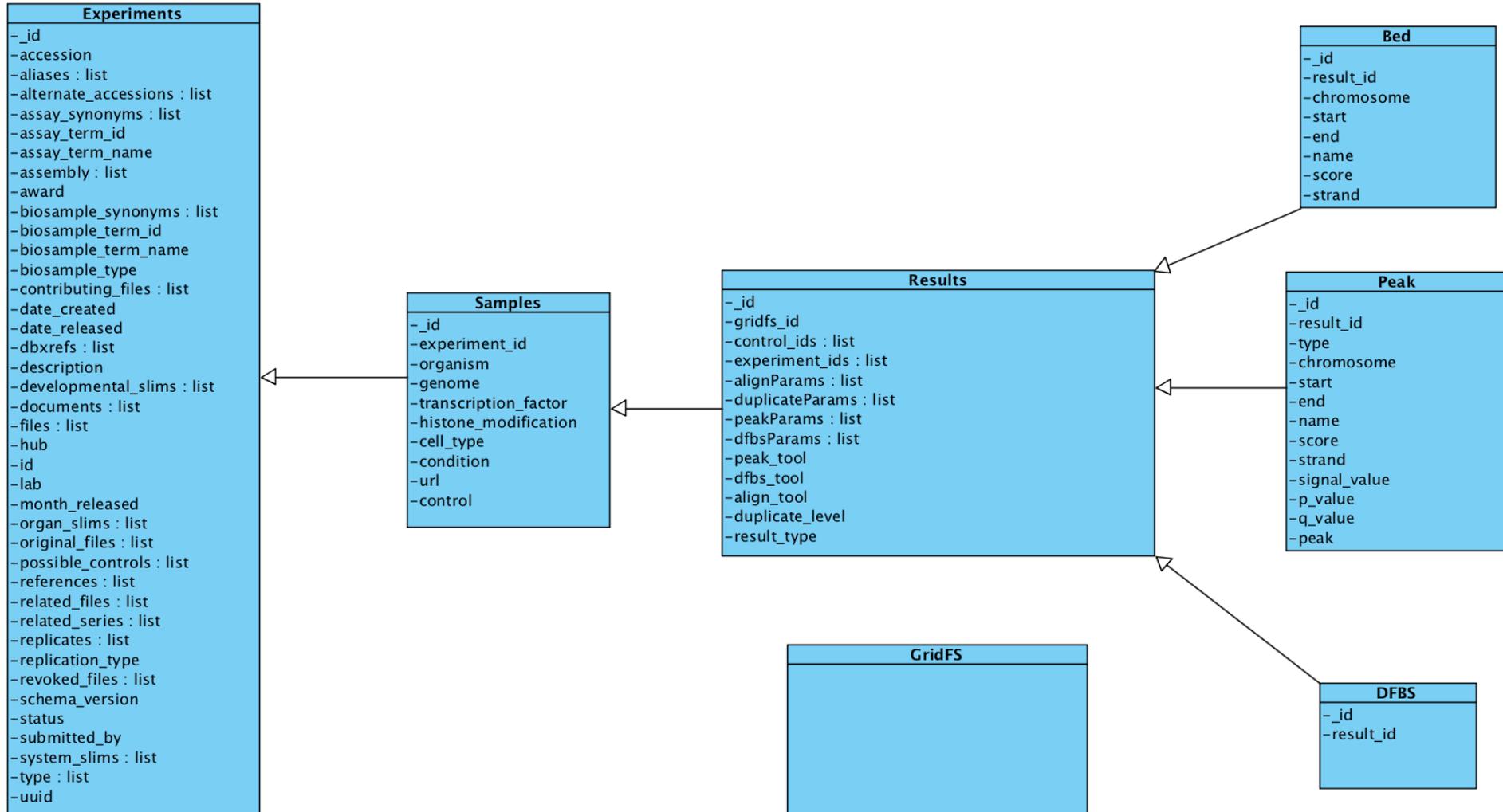
Workflow I
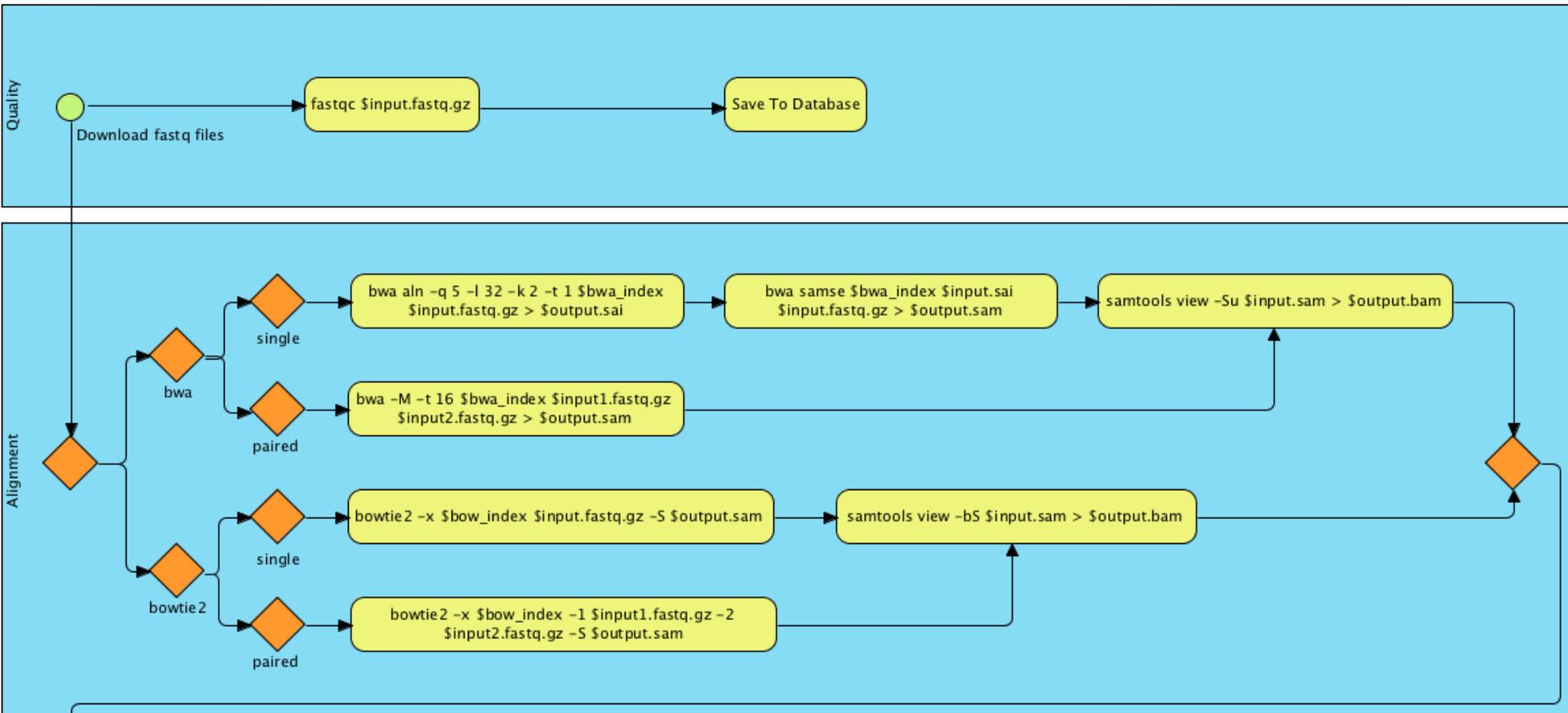
Workflow II

Regulation of cRPGs (outer circle) and mRPGs (circle next from the outside). Yellow circles: RPGs green diamonds: regulators Icon size is proportional to the number of regulatory relationships.

Note the extreme density of regulatory relationships.