

Readme for Cactus

1. INTRODUCTION

This is a suite of tools for creating and manipulating 'Cactus' databases, used for genome comparison. Much of this suite of programs is experimental and under development, so naturally not all of it is yet documented.

This guide will help you build a genome MSA of a set genomes. It does not yet cover building phylogenetic trees, creating ancestral genomes or deriving a novel reference genome from an alignment.

2. INSTALLATION

To install Cactus please follow the instructions in INSTALL.txt

3. GETTING STARTED

3.1 CREATING AN EXPERIMENT

The basic inputs to Cactus are a set of sequences, organised into sets that comprise 'genomes', and a tree that relates these genomes.

As the number of parameters can be large for building a genome MSA each run is controlled with an XML parameter file. The following shows a simple, complete example:

```
<cactus_workflow_experiment
  sequences="HUMAN CHIMP BABOON MOUSE RAT DOG CAT PIG COW"
  species_tree="(((H:0.0067,C:0.0098):0.025,B:0.045):0.11,(M:0.073,R:
0.081):0.26):0.023,((D:0.07,C:0.07):0.087,(P:0.06,C:0.06):0.10):0.04);"
  config="default"
>
  <cactus_disk>
    <st_kv_database_conf type="tokyo_cabinet">
      <tokyo_cabinet database_dir="F00" />
    </st_kv_database_conf>
  </cactus_disk>
</cactus_workflow_experiment>
```

The top level tag in the file is called 'cactus_workflow_experiment', so called because of the script used to run it (and control everything), called cactus_workflow.py.

It has a number of simple variables. Three are 'required'.

1. The 'sequences' attribute, this string gives the location of the genomes that will be aligned. In this case there are nine paths specified (HUMAN, CHIMP, BABOON, MOUSE, RAT, DOG, CAT, PIG, COW). Each may be either:
 - A. A single (multi) fasta file containing sequence entries.

- B. A directory, which is searched non-recursively, and which may contain a number of files, each of which must itself be a (multi) fasta file. The collection of sequences associated with each path is conceptually a genome (ED: we should change the name of the attribute to 'genomes').
- 2. The 'species_tree' attribute (ED: we should change the name of this attribute to simply 'tree'). This is any valid newick tree string, such that (1) the number of leaves is the same as the number of genomes in the sequences tag, (2) and every branch except the root branch has a length, each of which is an estimate of the number of substitutions per neutrally evolving site that occurred along that branch. The tree can have multi-furcating nodes (more than two children, two children, or just one child). The genomes are associated with the leaves of the tree by the left-to-right ordering of the nodes in the tree. Thus, the first leaf node in the tree string ('H') is the node for the HUMAN genome, the second leaf node in the tree string ('C') is the node for the CHIMP genome, etc.
- 3. The 'config' attribute. This is a path to the location of a parameter file. As this file is frequently shared by several alignments its contents are not embedded within the experiment file. The keyword 'default' can be used in place of a file path to specify that the default parameters should be used.

Nested within the 'cactus_workflow_experiment' tag is a 'cactus_disk' tag. This tag specifies the database to be used. Currently it just (redundantly) contains a nested 'st_kv_database_conf' tag. This tag is where the type and location of the database is specified. We now describe this tag/

3.2 DATABASE CONFIGURATION

A database tag of the form:

```
<st_kv_database_conf type="foo">
  <foo />
</st_kv_database_conf>
```

Is used to link an experiment to a database. "foo" may currently be either "tokyo_cabinet", "kyoto_tycoon", "mysql" or "postgresql". In the case of tokyo_cabinet the nested tag is of the form:

```
<tokyo_cabinet database_dir="F00" />
```

Where the attribute database_dir locates the new embedded database to connect to. Details about tokyo cabinet can be found at: <http://fallabs.com/tokyocabinet/>

In the case of mysql the nested tag is of the form:

```
<mysql host="F00" port="INT" user="F00" password="F00" database="F00"
table="F00"/>
```

Each of these parameters is therefore required for specifying a database and table on a machine. The postgresql and kyoto tycoon interfaces are under developed and will be detailed shortly.

3.3 RUNNING THE EXPERIMENT

Once an experiment file is created the `cactus_workflow.py` executable script must be run. To create a simple alignment for a given experiment file the command is run as follows:

```
-> cactus_workflow.py --experiment experimentFile.xml --  
setupAndBuildAlignments
```

The required named argument 'experimentFile.xml' gives the location of the experiment file, whose contexts are described above. The option '--setupAndBuildAlignments' instructs the script to setup the database and build the alignment. The script runs using the jobTree, and can accept all the standard arguments described for jobTree (see <https://github.com/benedictpaten/jobTree>).

3.4 EXTRACTING MAF ALIGNMENTS

A cactus database is accessible via a substantial C API. However, we are writing utilities to extract information from the database in common file formats. To generate the 'out.maf' MAF file from a given database the following command is run:

```
-> cactus_MAFGenerator --outputFile out.maf --cactusDisk  
'<st_kv_database_conf .... </st_kv_database_conf>'
```

Where the embedded `st_kv_database_conf` tag is as described in Section 3.2, and instructs the program how to access the stored data.