CgiHunter

CATCGTGGACGAT

Whole Genome Annotations

This tutorial is focused on how whole genome annotations of CpG islands can be generated with the CgiHunter software.

1. Installation

The standard installation is described in “CgiHunter User Manuel.pdf”. If a customized installation is preferred, the necessary steps can easily be performed manually. It is assumed that Python 2.4 is installed on your computer system.
Obtain the source distribution of CgiHunter from the project website at http://cgihunter.bioinf.mpi-inf.mpg.de . Unpack the archive into one directory in your file system. We will refer to this directory as <CgiHunter-Home>. You can now start the program as commandline tool (CgiHunter.py) or via a Graphical User Interface (CgiHunterGUI.py).
If you want to import CgiHunter into your own custom scripts, you have to ensure that your Python interpreter can find these files on demand. Therefore, you have to add <CgiHunter-Home> to the PYTHONPATH environment variable. This can be done in different ways, depending on the platform you are using (for more information see appendix). To verify that the PYTHONPATH is set correctly, open a python shell and enter:

import sys
print sys.path

The installation directory should be included in the displayed list of directories. If this is not the case you can add it manually by typing:

sys.path.append(‘<CgiHunter-Home>’)

This will make the CgiHunter modules available within this python shell until it is closed. Finally, you can verify that the software is running, by entering:

import CgiHunter

If no error message is displayed the installation was successful.
To start the Graphical user interface (GUI) from the shell type:

import CgiHunterGUI

or use a command line shell by changing to the <CgiHunter-Home> directory and starting the CgiHunterGUI.py script directly:

python CgiHunterGUI.py

on Unix systems or:

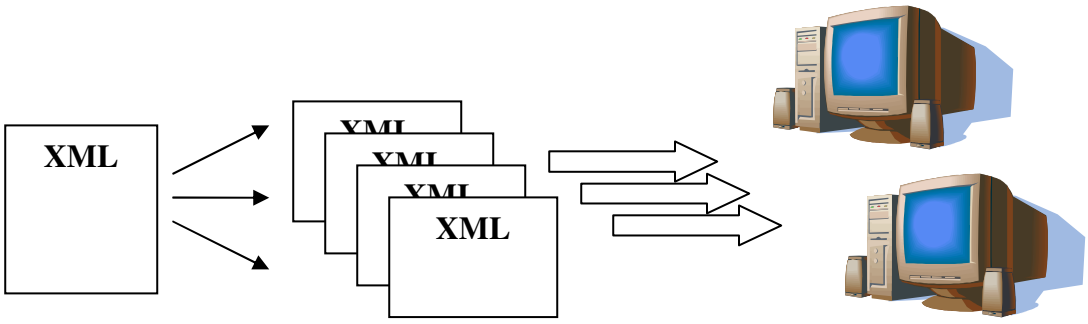
CgiHunterGUI.py

on windows machines.

Finally, on windows systems you can start the script also by clicking on the CgiHunterGUI icon in the <CgiHunter-Home> folder.

2. Procedure

Whole genome annotations, as all types of CgiHunter annotations, are defined by specific XML files. These files can be generated automatically by using the Graphical User Interface of the CgiHunter Tool (see "CgiHunter User Manuel.pdf").
To accelerate the annotation procedure, an individual XML file can be distribute into several subtasks that can be processed independently from each other.



This step is performed twice during the whole genome annotation. The first time during the filter procedure, when all involved DNA sequence files are analyzed for the regions that can be excluded from a more detailed scan. In consequence a list of the remaining regions is generated (cluster map). Each of the cluster in the cluster map can be processed independently from the other cluster, and hence, for the detail scan the xml files are distributed a second time.

3. Step by Step

1) Prepare your DNA sequence data. The program accepts only sequences in the FASTA format.
2) Generate the XML file that specifies your analysis. You can use the CgiHunterGUI for this step or adapted an existing template. It is highly recommended to use the GUI, as the XMLs it produces are 100% compatible with the main program.
3) In this step you chose a working directory (we will refer to it as <*wd*>), which will be the root directory for all further data connected to the annotation. Further, you chose a name for the analysis <*run name*> and set all other parameters of the analysis.
4) Distribute the tasks for the filter step. In this step the original XML file is used to generate a subtask for each referenced FASTA file. These subtasks are placed in a new directory *FilterTask* that is generated in your working directory <*wd*>. Further, more, a list of all subtasks is generated in <*wd*>. To assist you in processing all the subtasks CgiHunter can generate three types of shell scripts.
5) The **array script** is designed to be used by an automatic scheduling system. It basically specifies that the command *python CgiHunter.py –x <taskname>.xml* should be called once for each subtask. The scheduling system then can distribute these calls on its computational resources. More information on the array script can be found in the appendix.
6) The **batch script** simply lists all individual subtask calls to CgiHunter. This approach is appropriate if your infrastructure can distribute these calls automatically on your computational resources.
7) The **separated scripts** option generates can be used if you have a fixed number of computer nodes available. It generates one script for each node that then can be started manually.
8) This last option makes use of CgiHunters powerful comandline options, especially the –X option of CgiHunter.py can also be used for custom scripts. It takes a task list as argument an executes by default every task in it. To distribute the tasks on different machines you can set in addition the –e and –o options. For example, *python CgiHunter.py –X <tasklist> -e 3 –o 10* executes every third out of ten tasks in the list. By adjusting the –e option from 1 to 10, the jobs then can be easily distributed among ten computers.
9) You can check in the GUI, if all processes have finished correctly. Therefore, click on the buttons “Genome scale” -> “Prepare detail scan” and chose the <*run_name*>*_Filter.xml*. Here you can revisit the Parameters you have chosen and under the option “Final” monitor the progress of the analysis. The recent status of each job is listed here. If the computation of the array script has been interrupted, or is finished and some subtasks have not terminated correctly, you can simply restart the process and only the incomplete jobs will be recomputed.
10) If the filter step is finished you can proceed with the detail scan. Here the subtasks are stored in the subdirectory *DetailTasks* and the array script is named

<*run_name*>*_Detail.xml*. In this step you can additionally choose the degree of parallelization by choosing the number of clusters per subtask.

11) Proceed like in steps 4 to 8.
12) When you have verified that all jobs have finished correctly in the GUI under buttons “Genome scale” -> “Generate Annotation”, you can generate the final annotation, which will be placed as BED file under the name of the used genome assembly in <*wd*>*/Results*.

Appendix:

Setting the PYTHONPATH environment variable:
On Unix systems modify the configuration file of your shell. For example for *bash* by adding to ~/.bashrc the line:

export PYTHONPATH= <CgiHunter-Home>

On Windows to set the PYTHONPATH variable:

1. Go to Control Panel -> System-> Advanced.
2. Click *'Environment Variables'* button.
3. In the 'System Variables' panel that appears, *click* New and enter PYTHONPATH in the 'Variable Name' field. In the 'Variable Value' field enter path to <CgiHunter-Home>.

The array script

The script has six lines of code. It is used as a template for the scheduling system.
The first line specifies that this script should be processed by the program *sh* that is assumed to by installed in the directory */bin*. The second line specifies that the scheduling system should use the program *bash* in the directory for the further processing */bin* and that this template should be executed several time. In each run, the variable $SGE_TASK_ID is changed with values ranging from x to y. This is specified by –t x-y.
The third line of code specifies the path of the CgiHunter program.
The fourth line of code defines the xml file of the recent instance of the template. It is determined by fetching the line number <$SGE_TASK_ID> from the file that lists the xml files of the subtasks.
Finally the last line invokes the interpreter of Python 2.4 and passes all necessary arguments to it.

Depending on your system you may have to adapt the path names of sh and bash or change the name of the python interpreter. If you do not use a schedule system that supports this kind of array scripts, you can build on the structure of this script to generate a custom script that fits your system architecture. If you use CgiHunter more often, it is also possible to adjust the *generateArraySkript* function in *CgiHunterTools.py*. It is 18 lines long and can be modified easily. Please document all changes in the header the module.