ConnectBook / Home

Autodock-Vina workflow

Creado por Balamurugan (Bala) Desinghu , modificado por última vez en abr 30, 2015

Overview

In this tutorial, we will explore how pegasus simplifies the docking of several ligands for a given protein. Once a pegasus workflow is established for a sample case, we could re-use the workflow for docking calculations of different ligand libraries and proteins. In the long run, the pegasus workflow approach saves a lot of time in adding ligands to library or set up calculations for set of receptors and ligands.

Docking with library of ligands

We want to screen a library of ligands using Autodock-Vina and pegasus workflow manager. The necessary files to run the example are available to you via *tutorial* command.

```
$ tutorial ### Without an argument, it shows the list of available tutorials.
$ tutorial pegasus-vina  ### The files to run the pegasus-vina tutorial are created under the director
$ cd tutorial-pegasus-vina  ### Change to this directory
```

Let us take a look at the files inside the directory "tutorial-pegasus-vina". The following are vina input files.

```
receptor_config.txt   ### Configuration file (input)
receptor.pdbqt        ### Receptor pdbqt file that contains coordinates and charges of atoms(input)
```

The ligands are listed inside "input_ligands" directory.

```
$ ls input_ligands   ### There are 40 ligands inside this directory.
```

The following files are related to pegasus workflow management

```
pegasusrc ### The configuration file for pegasus
dax-generator-singleJob.py ### Python script generates dax.xml file
sites-generator.bash   ###   Bash script generates sites.xml file
submit.bash   ### Bash script submits the pegasus workflow.
```

The file "pegasusrc" contains the pegasus configuration information. We can simply keep this file in the current working directory without worrying much about the details (If you would like to know the details, please visit the pegasus home page). The files - dax.xml and sites.xml contain the information about the work flow and data management.

Submit script

Let us pay attention to few parts of the "submit" script to understand about submitting the workflow. Open the file "submit" and take a look

```
...
line 9 ./dax-generator-vina.py ### Execution of "dax-generator-vina.py" script. Generates dax.xml.
...
line 14 ./sites-generator.bash   ### Execution of "sites-generator.bash" script. Generates sites.xml.
...
line 17 pegasus-plan \               ###  Executes the pegasus-plan with the following arguments
line 18    --conf pegasusrc \             ###   pegasus configuration file
line 19    --sites condorpool \           ###   jobs are executed in condorpool
line 20    --dir $PWD/workflows \         ###   The path of the workflow directory
line 21    --output-site local \          ###   Outputs are directed to the local site.
line 22    --cluster horizontal \         ###   Cluster the jobs horizontally
line 23    --dax dax.xml \                ###   Name of the dax file
line 24    --submit                       ###   Type of action is submit
```

sites-generator

The purpose of sites-generator script is to generate the sites.xml file. There are several lines declared in the sites-generator script. We need to understand the lines defining the scratch and output directories.

```
...
line 4 cat >sites.xml <<EOF  ### creates the file "sites.xml" and appends the following lines.
...
line 11 <file-server protocol="file" url="file://" mount-point="$PWD/scratch"/>   ###  Define the path
line 12 <internal-mount-point mount-point="$PWD/scratch"/>                        ###  Define the path
...
line 17 <file-server protocol="file" url="file://" mount-point="$PWD/outputs"/>   ### Define the path
line 18 <internal-mount-point mount-point="$PWD/outputs"/>                        ### Define the path
...
line 32 EOF   ### End of sites.xml file
```

The files "submit.bash" and "sites-generator.bash" will not change very much for a new workflow. We need to edit these two files, when we change the name of the dax-generator and/or the path of outputs, scratch and workflows.

DAX generator

The file - dax.xml contains the workflow information, including the description about the jobs and required input files. We could manually write the dax.xml file but it is not very pleasant for the human eye to deal with the xml format. Here, dax.xml is generated via the python script "dax-generator-singleJob.py". Take a look at the python script, it is self explanatory with lots of comments. If you have difficulty to understand the script, please feel free to send us an email. Here is the brief description about dax-generator python script.

```
...
line 8 dax = ADAG("vina-ligand-receptor") ###   Name of dax. You may change any interesting name you l
...
line 10 base_dir = os.getcwd()   ### The path of the reference base directory defined as the current w
line 11
...     ###  Adds "vina_wrapper.bash" file path to the dax.xml file
line 17
...
line 18
...     ###  Adds "receptor_config.txt" file path to the dax.xml file
line 23
...
line 24
...     ###  Adds "receptor.pdbqt" file path to the dax.xml file
line 28
...
line 31 input_ligands_dir = base_dir + "/input_ligands"   ### Defines the path of the input ligands di
...
line 35 ### Loops over the ligand files inside the "input_ligands" directory.
...
line 36
...     ###  Adds each ligand file path to dax.xml
line 38
...
line 40
...     ### Name and information of output files
line 43
...
line 45
...    ### Define a job with the description about input and output files.
line 53
```

Job submission and status

To submit the job

```
###   To submit the job
$ ./submit
```

To check the status of the submitted job

```
$ pegasus_status
###  or you can also check with the condor_q command
$ condor_q username   ###   username is your login ID
```

Pegasus creates the following directories

```
scratch/   ### Contains all the files (including input, parameter and execution) required to run the j
workflows/   ###  Contains the workflow files including DAGMan, data transfer scripts and condor job f
outputs/   ###  Where the NAMD output files are stored at the end of each job.
```

The path of the scratch, workflows and outputs directories are declared in the "submit" scripts at lines 19, 20, 25,26 and 47.

Key points

- Pegasus requires dax.xml, sites.xml and pegasusrc files. These files contain the information about executable, input and output files and the relation between them while executing the jobs.
- It is convenient to generate the xml files via scripts. In our example, dax.xml is generated via python script and sites.xml is generated via bash script.
- To implement a new workflow, edit the existing dax-generator, sites-generator and submit scripts. In the above examples, we modified the workflow for the single NAMD job to implement the workflows of N-sequential and M-parallel, N-sequential jobs.

References

1. Pegasus documentation
2. OSG Connect QuickStart

For further assistance or questions, please email connect-support@opensciencegrid.org.