

Table of Contents

Table of Contents	1
Job Manager for remote execution of ATK scripts	2
A single remote machine	3
Settings	4
Environment	6
Resources	7
Notifications	8
Diagnostics	9
Save and test the new machine	10
Custom job settings	14
Debugging	15
Adding several remote machines	16
Rename and export	16
A machine for threaded jobs	18



[Docs](#) » [Tutorials](#) » [VNL tasks and workflows](#) » Job Manager for remote execution of ATK scripts

Job Manager for remote execution of ATK scripts

Version: 2016.0

Downloads & Links

- [PDF version](#)
- [mpi_check.py](#)
- [test_mpi.py](#)
- [silicon.py](#)

In this tutorial you will learn how to use the Job Manager for execution of ATK jobs on remote computing clusters. In particular, we will learn how to:

- add a remote machine to the Machine Manager;
- use custom Machine Settings for individual jobs;
- add several different machines and import/export machine settings.

Note

You will set up a remote machine for running jobs in parallel using MPI, as well as with threading. We strongly recommend you go through the tutorial [Job Manager for local execution of ATK scripts](#) before continuing with this one.

Important

There are two essential requirements when using the Job Manager for executing and managing ATK jobs on a remote cluster. You need:

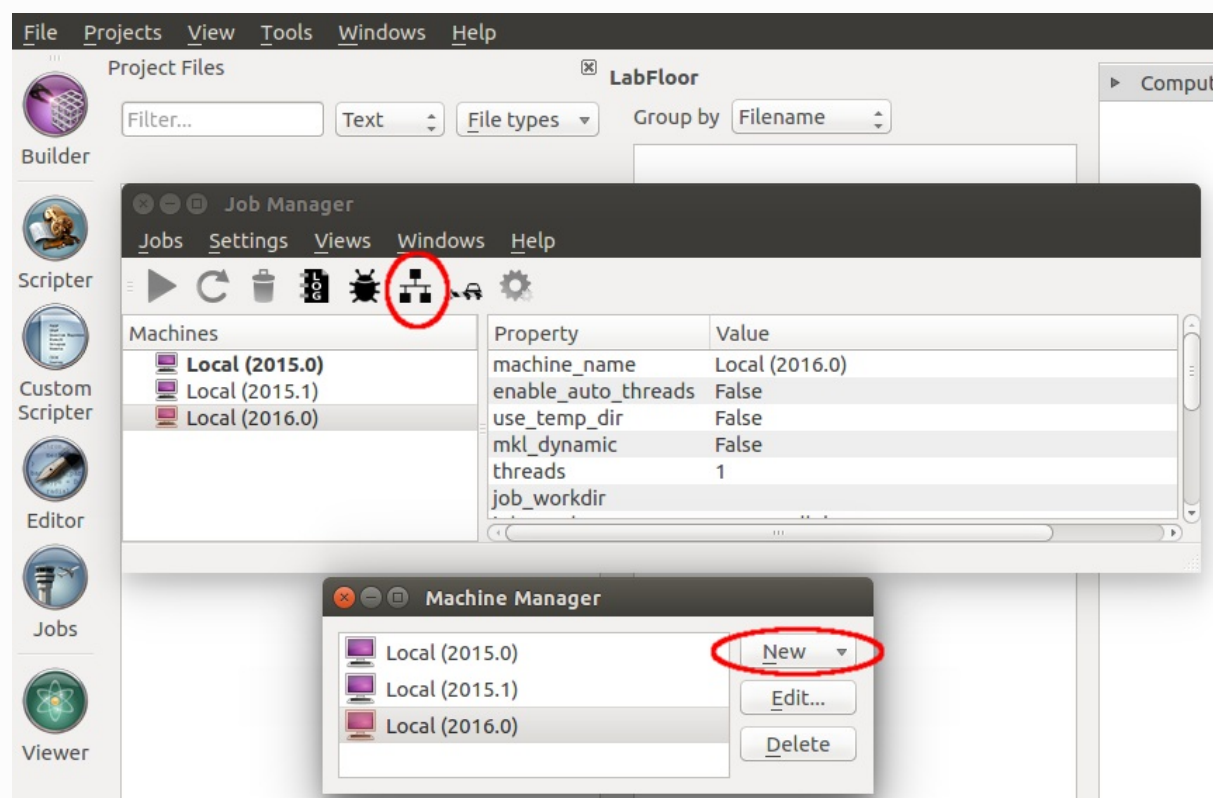
1. ATK installed on the local machine and on the cluster;
2. password-less SSH connection from your local machine to the cluster.

Please refer to the tutorial [SSH keys](#) if you need help setting up the SSH connection.



A single remote machine

Open the  **Machine Manager**, and click *New* in order to add a new machine.



Note

Please note that the Machine Manager options may differ between versions of VNL.

The menu that appears has three options, *Local*, *Remote PBS*, and *Remote Direct*.

Click *Remote PBS* to start setting up the connections to a remote Linux cluster with a PBS job scheduling system. The option *Remote Direct* is appropriate for clusters that have no queue system.

The **Machine Settings** widget pops up. It has five main tabs:

- Settings (*remote connection*),
- Environment (*software on the remote cluster*),
- Resources (*allocated computing resources and time*),
- Notifications (*job progress updates*),
- Diagnostics (*check the current setting*).

Machine Settings

SettingsEnvironmentResourcesNotificationsDiagnostics

Hint

Most options have default values (which must be checked). Red fields are mandatory, but have no default.

Settings

Connection to the remote cluster.

Machine Settings

SettingsEnvironmentResourcesNotificationsDiagnostics

Machine name*

Sabalcore

Connection

Hostname*

sciclust^{er}.com

Port

22

Username*

user01

Private key directory

~/.vnl

...

Node type names

orange

Queue names

Path to PBS binaries*

/usr/local/torque-4.2.8/bin

* Required fields

Export...

Import...

Cancel

OK

You need to specify the following fields:

Machine name
Hostname

Username

The following options vary from cluster to cluster.

Private key directory

The directory containing your private SSH key.

Node type names

Optional, but in this example we have access to node type “orange”.

Queue names

Optional, could for example be “long”.

Path to PBS binaries

The directory containing the **qsub** and other PBS executables must be specified. Log on to the cluster and use the `which` command to locate it (here `/usr/local/torque-4.2.8/bin`):

```
$ which qsub
/usr/local/torque-4.2.8/bin/qsub
```

Once all settings are added, you can check if they are correct by navigating to the Diagnostics tag and clicking the “Run Diagnostics” button.

Machine Settings

SettingsEnvironmentResourcesNotificationsDiagnostics

Private key check	✓
Connection check	✓
Directory check	✓
ATK check	✓
Sourcing check	✓
Module loading check	✓
PBS binary check	✓
Queue check	✓

Run Diagnostics

* Required fields

Export...Import...CancelOK

Tip

The Diagnostics tab checks if the options in the **Settings** and **Environment** tabs allow your local computer to connect to the remote cluster and execute the commands needed for job submission and management.

If some field is not specified, the diagnostics will check the default setting. If the connection to the cluster works well with that default, or is at least not disrupted, it will be marked as OK.

You therefore need to run an actual test job to make absolutely sure that all settings are indeed OK.

Environment

Computing environment on the cluster.

Machine Settings

Settings
Environment
Resources
Notifications

Working directory*
\$HOME/calculations

ATK executable path
\$HOME/QuantumWise/VNL-ATK-2016.0/bin/atkpython

mpiexec executable
mpiexec

Scripts to source
~/.bashrc

Export statements
OMP_NUM_THREADS=1
QUANTUM_LICENSE_PATH=\$HOME/QuantumWise/atk_license.lic

Modules to load
mpich3

☒ Export all environment variables from the submitting shell

* Required fields

Export...
Import...
Cancel
OK

This tab concerns the environment (directories, executables, modules, etc.) on the *remote cluster*, not on your local computer. `$HOME` is therefore your home directory on the cluster, and VNL-ATK must of course be installed on the cluster.

Any scripts that should be sourced in order to get the environment working must be listed. The same goes for required export statements (may be needed to correctly set the `QUANTUM_LICENSE_PATH`) and cluster modules that should be loaded.

Note

You should almost always export `OMP_NUM_THREADS=1` for MPI parallelized jobs. This completely eliminates threading.

Resources

Computing resources requested at job submission.

×

Machine Settings

Settings

Environment

Resources

Notifications

Diagnostics

Nodes

Number of nodes

2

Number of cores per node

8

Node type name

orange

☐ Enable MKL_DYNAMIC

Queue name

Number of MPI processes

16

Number of MPI processes per node

8

Maximum amount of physical memory (MB)

0

Maximum wall-clock time

0 H

0 M

0 S

* Required fields

Export...

Import...

Cancel

OK

This tab specifies the default computing resources (nodes, cores, queue, time, etc.) requested at job submission. Note that `MKL_DYNAMIC` is disabled by default, which means that MKL is not allowed to dynamically decrease the number of threads at runtime.

Tip

For a job with only MPI parallelization (no threading, `OMP_NUM_THREADS=1`), the number of nodes times the number of cores per node should equal the (total) number of MPI processes. In the example above, we have $2 \times 8 = 16$ cores, so we ask for 16 MPI processes and specify that each node should have 8 of those processes (only one MPI process per core).

Notifications

Job progress reports.

Machine Settings

SettingsEnvironmentResourcesNotificationsDiagnostics

Server update

Log file should be updated every30seconds

Running job should be checked every60seconds

E-mail notifications

E-mail addressmy_email_address@email.com

Send e-mail when job☐ begins☒ aborts☒ ends

* Required fields

Export...Import...CancelOK

The Job Manager will regularly check the job progress on the remote cluster and report it in the log file. You can also receive e-mail notifications from the PBS scheduler when the job starts, finishes, or aborts.

Diagnostics

Use this tab to test the machine settings. Green check-mark indicates that all settings appear to be OK. Red circle indicates some problem that should be fixed in one of the tabs.

Machine Settings

Settings

Environment

Resources

Notifications

Diagnostics

Private key check	✓
Connection check	✓
Directory check	✓
ATK check	✓
Sourcing check	✓
Module loading check	✓
PBS binary check	✓
Queue check	✓

Run Diagnostics

* Required fields

Export...

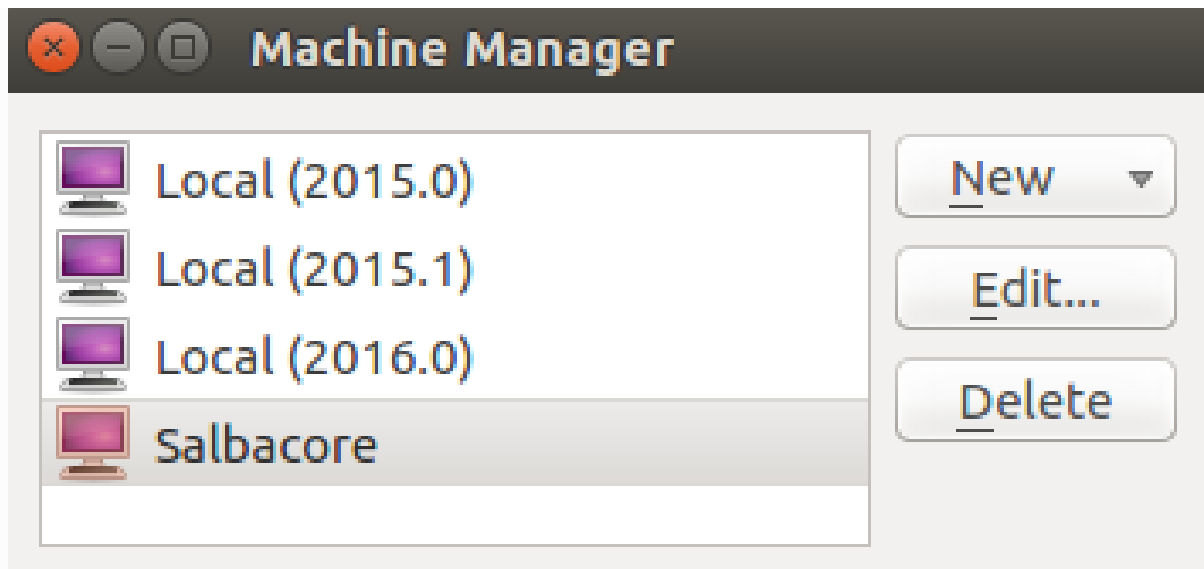
Import...


Cancel

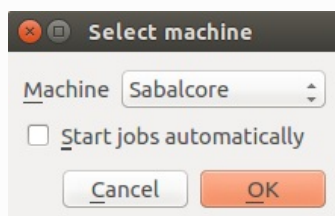
OK


Save and test the new machine

Click *OK* to add the machine to the **Machine Manager**.



Next, you should run the test scripts [📄 mpi_check.py](#) and [📄 test_mpi.py](#) to test the machine settings. In the VNL main window, drag and drop a script onto the  **Job Manager**. Choose the newly added machine (in this example "Salbacore"), and click *OK*.



Click  to submit the job and watch how the **Task State** changes from *Pending* to *Finished*. For a longer job, you can click *Download log* to regularly retrieve the log file during the remote job execution.


Job Manager

_Jobs _Settings _Views _Windows _Help

Machines

- Local (2015.0)
- Local (2015.1)
- Local (2016.0)
- ▼ Sabalcore
 - test_mpi.py
 - mpi_check.py

Task	State
Prepare working directory	Pending
Copy files to remote	Pending
Create PBS script	Pending
Submit on remote	Pending
Queued on remote	Pending
Running on remote	Pending
Download results	Pending
Download log	Ready
Stop execution of job	Ready
Download results	Ready

Once a job has finished you should check the log file to inspect the job output to see if the expected number of processes were used. Click the  icon to open the log. In the two examples below, 16 cores were used in total on two different nodes.

Logs

test_mpi.log (Sabalcore)

```

-----+
|
| Atomistix ToolKit 2016.dev [Build b5fc11d]
|
|-----+
Slave node: n583024
Slave node:Slave node: n583024
Slave node: n583025
Slave node: n583024
Slave node: n583025
Master node: n583024
Slave node: n583025
Slave node: n583024
Slave node: n583025
Slave node: n583024
Slave node: n583025
Slave node:n583025 n583025
  
```

Logs

mpi_check.log (Sabalcore)

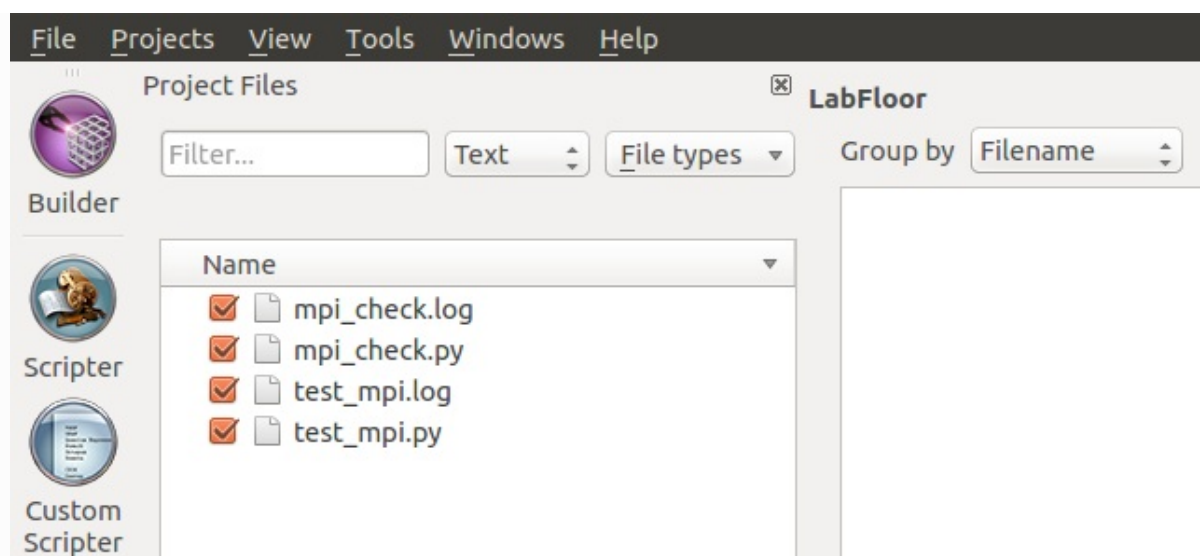
```
+-----+
|
| Atomistix ToolKit 2016.dev [Build b5fc11d]
|
+-----+
MPI process 3 of 16 reporting.MPI process 1 of 16 reporting.
MPI process 4 of 16 reporting.
MPI process 8 of 16 reporting.MPI process 14 of 16 reporting.
MPI process 11 of 16 reporting.
MPI process 12 of 16 reporting.
MPI process 5 of 16 reporting.
MPI process 13 of 16 reporting.
MPI process 6 of 16 reporting.
MPI process 7 of 16 reporting.

MPI process 15 of 16 reporting.


MPI process 0 of 16 reporting.
MPI process 2 of 16 reporting.
MPI process 10 of 16 reporting.
MPI process 9 of 16 reporting.

Timing:                                Total      Per Step      %
-----
Loading Modules + MPI    :      6.36 s      6.36 s      95.02% |
=====|
-----
Total                    :      6.70 s
```

The job outputs have of course also appeared on the **LabFloor**:



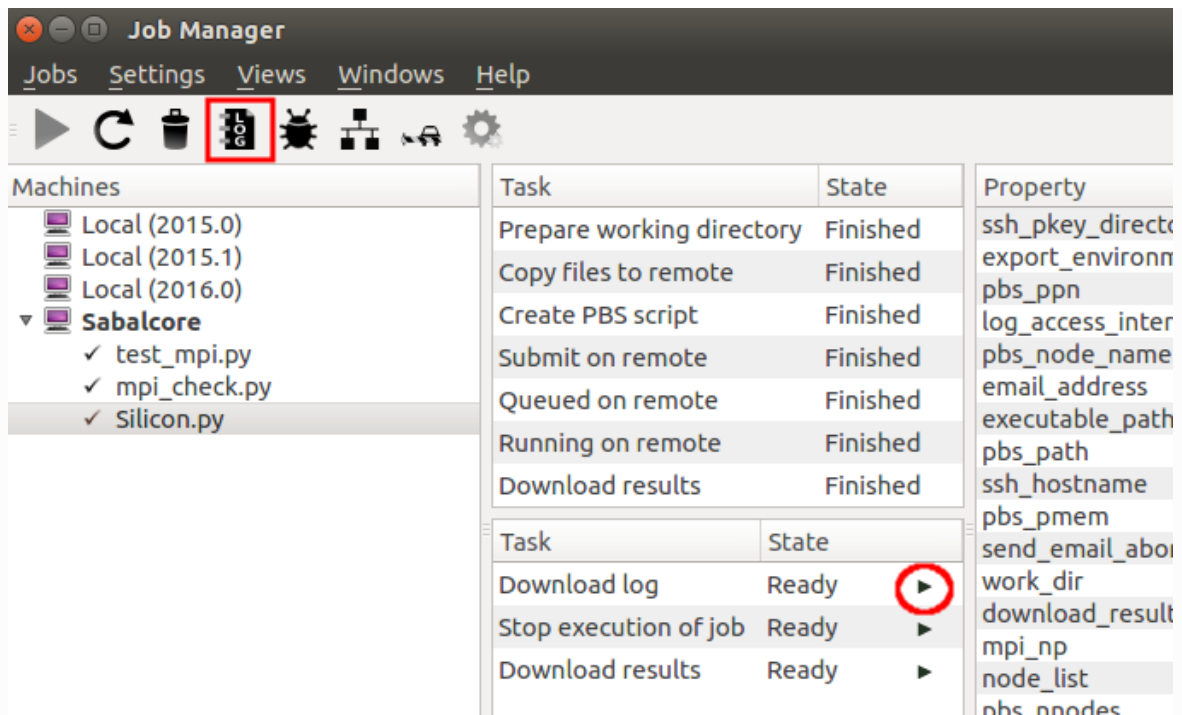
Custom job settings

As explained in the tutorial [Job Manager for local execution of ATK scripts](#), you can customize many of the job settings before submitting a job. Use the script [silicon.py](#) as an example ATK script. Download the script, send it to the **Job Manager**, and choose the newly created remote machine. Then click the  **Job Settings** plug-in.

You can now customize the *Resources* and *Notifications* tabs, and thereby submit the job with settings different from the default ones you specified above. For example, you can change the number of requested cores and cluster queue and/or maximum wall-clock time. You can also change the notifications settings.

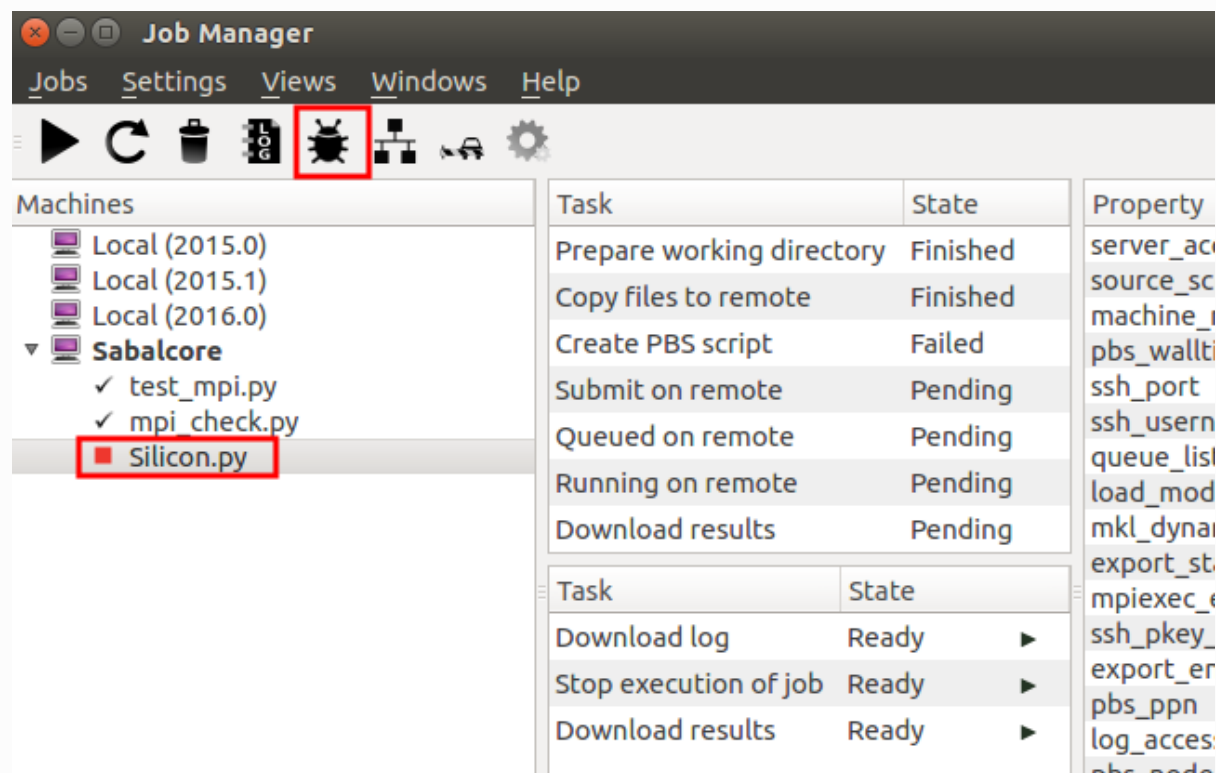
Set up the job settings as you like, e.g. 4 cores on a single node and 4 MPI processes. Then click **OK** and submit the calculation to the remote cluster.

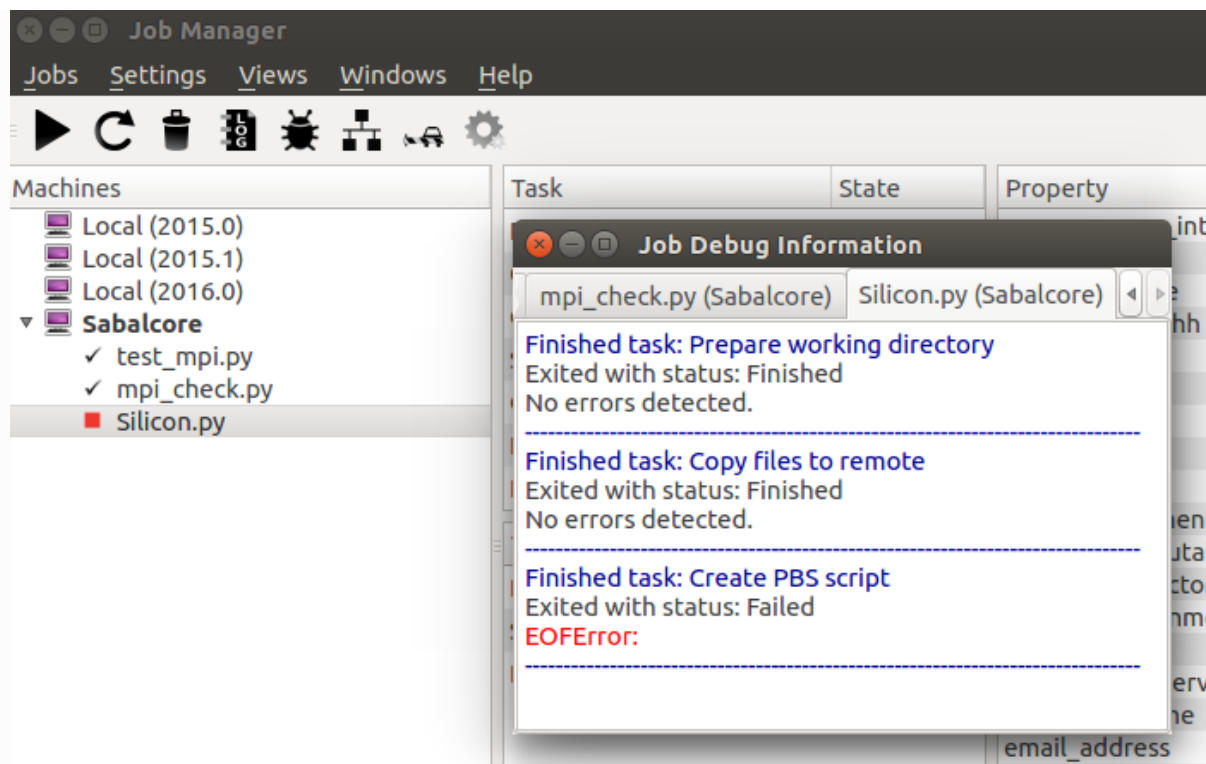
The job log is automatically retrieved when the job finishes. For long jobs, however, you need to click "Download log" if you want to see the log while the job is running.



Debugging

If an error occurs during the execution of the job, this will be indicated by a red square in the queue, as shown below. You can then click the **Debug logs** icon to open the job debugs information window, which will show you details about the error.





Adding several remote machines

Several remote machines can be added to the **Machine Manager**. You can of course add new machines from scratch, but you can also export/import the settings of an existing machine and use those as a template for new machines.

Tip

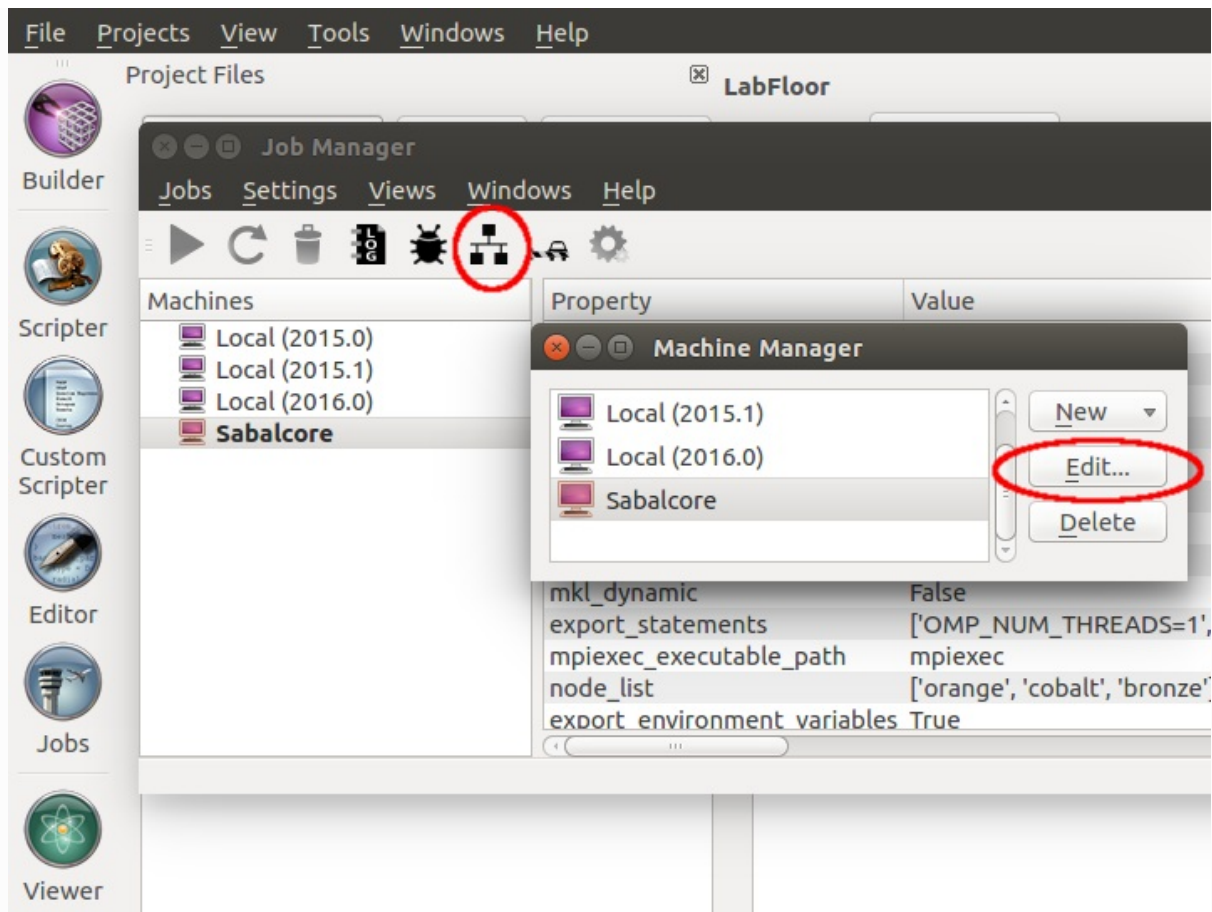
The import/export functionality is very convenient for sharing machine settings within a group of researchers.

In the following, you will rename and export the settings of the newly created machine, and then add one more remote machine suitable for threaded calculations.

Rename and export

In the **Job Manager**, remove the jobs that are in the queue of the newly added machine (named "Sabalcore" in our example).

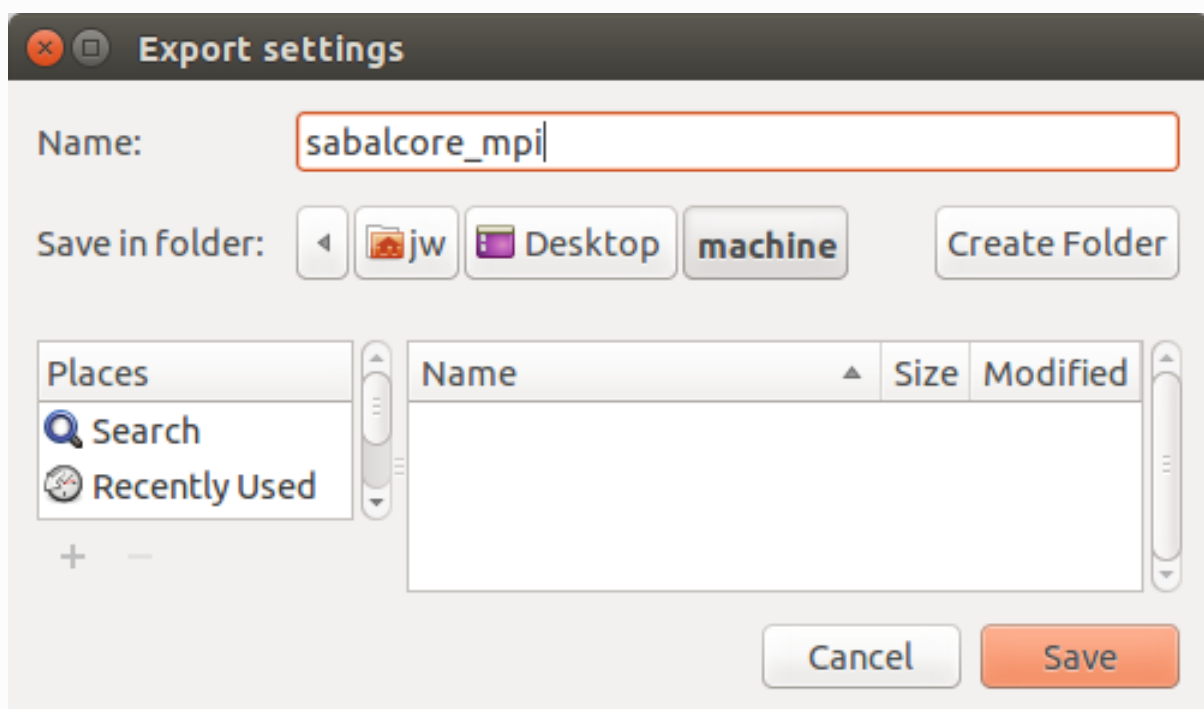
Click  **Machine Settings** and choose to *Edit* the machine.



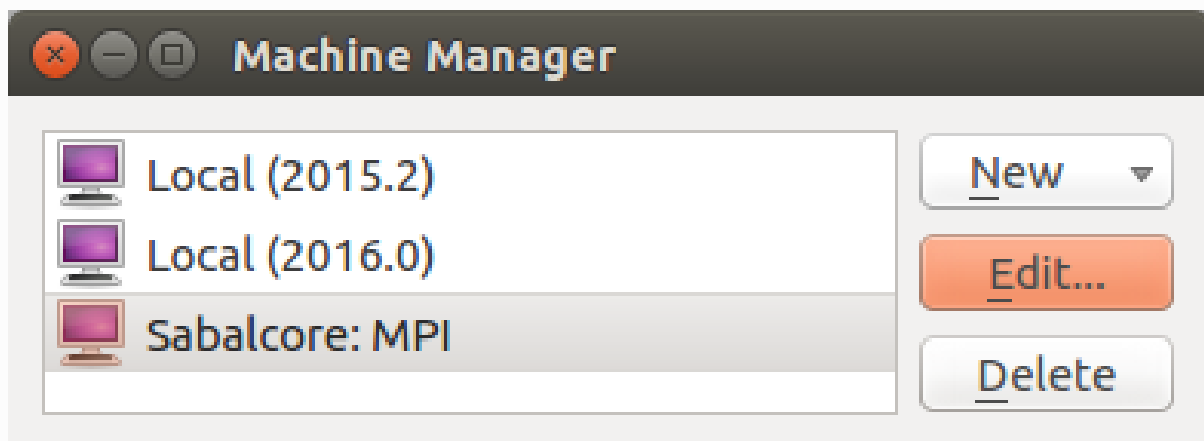
Note

You always need to empty the queue of a machine before you can edit its settings.

This machine is already set up for MPI parallelization. Therefore, append ": MPI" to the machine name, click *Export*, and save the settings in a file.



Click *OK* to accept the renaming and return to the Machine Manager window.



A machine for threaded jobs

Next, add a new remote machine with default settings for "Remote PBS". Then click *Import* and import the settings file you just saved.

You can now modify the settings to create a machine with default settings that are suitable for a threaded calculation:

- In the Environment tab, remove the `OMP_NUM_THREADS=1` export statement.
- In the Resources tab, use only a single MPI process but enable dynamic scheduling of MKL threads.

Machine Settings

Settings

Environment

Resources

Notifications

Diagnostics

Working directory* \$HOME/calculations

ATK executable path \$HOME/QuantumWise/VNL-ATK-2016.0/bin/atkpython

mpiexec executable mpiexec

Scripts to source ~/.bashrc

Export statements QUANTUM_LICENSE_PATH=\$HOME/QuantumWise/atk_license.lic

Modules to load mpich3

☒ Export all environment variables from the submitting shell

* Required fields

Export...

Import...

Cancel

OK

Machine Settings

SettingsEnvironmentResourcesNotificationsDiagnostics

Nodes

Number of nodes1

Number of cores per node8

Node type nameorange

☒ Enable MKL DYNAMIC

Queue name

Number of MPI processes1

Number of MPI processes per node1

Maximum amount of physical memory (MB)0

Maximum wall-clock time

0 H

0 M

0 S

* Required fields

Export...

Import...

Cancel

OK

Give the new machine a reasonable name, e.g. “Salbacore: Threading”, and click *OK* to add the machine to the Machine Manager. This machine should be convenient for submitting ATK-Classical calculations.

Machine Manager

Local (2015.2)

Local (2016.0)

Sabalcore: MPI

Sabalcore: Threading

New

Edit...

Delete

[← Previous](#)

[Next →](#)

© Copyright 2017, QuantumWise.