High Performance Computing for Genomics

Part III: Advanced

Advanced Topics

- **TMP directories**
- Own module installation
- How to do the management for a group

TMP directories

What?

> A tmp or temporary directory or folder is a directory used to hold temporary files. These files are used by some software, and are typically cleaned (removed) after an interval or at completion of the software.

Why?

> To store intermediate files (files used in calculations, but not needed as an end result).

TMP directories

Example: The creation of a bam file typically has a lot of intermediate files:

		Fastq
Mapping	BWA, Bowtie2,...	SAM (reads in order of fastq file)
SAM to BAM conversion	samtools view	BAM (reads in order of fastq file)
Sort according position	samtools sort	BAM (reads in order of position)
Add sample information	Picard AddOrReplaceReadGroups	Regular BAM file (reads in order of position + sample information)

TMP directories

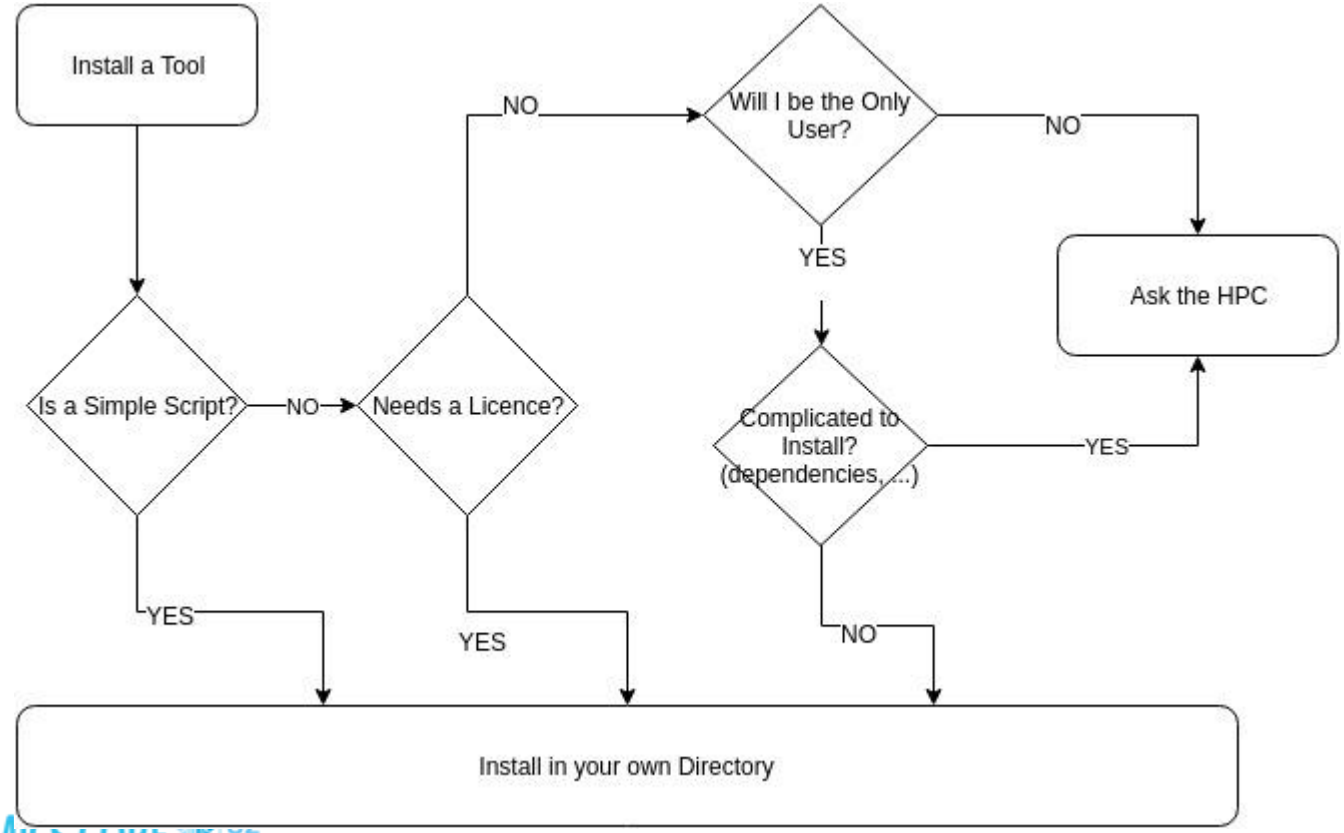
Create temporary files or directories:

```
TMPDIR=$VSC_SCRATCH_NODE;
TMP_DIR=`mktemp -d -t tmp.XXXXXXXX`;
cd $TMP_DIR;
```

Advanced Topics

- TMP directories
- **Own module installation**
- How to do the management for a group

Installing own modules



Installing own modules

- Create a software directory
- Create a directory with the tool name
- Create a directory with the tool version
  (and additional information like Java, Python, GCC, ... version)
- Install the tool in the directory (or in a lib directory)
- Create a bin directory
- Create links to the executables inside the bin directory (these links will be the names of the tools)
  OR
  Create scripts that launches the tool correctly
- Run the generate_modules script

Installing own modules: Example GBSX

- Goto software folder
- mkdir -p GBSX/v1.3-Java1.8.0_77/lib
- cd GBSX/v1.3-Java1.8.0_77/lib
- wget https://github.com/GenomicsCoreLeuven/GBSX/releases/download/v1.3/GBSX_v1.3.jar
- cd ..
- mkdir bin
- cd bin
- create GBSX file

Installing own modules: Example GBSX: bin/GBSX file

```
#!/bin/bash
JAVA_TMP_DIR="${VSC_SCRATCH}/tmp";
mkdir -p "${JAVA_TMP_DIR}";
java -Djava.io.tmpdir="${JAVA_TMP_DIR}" -Xmx32G -jar
/staging/leuven/stg_00019/software/gbsx/v1.3-Java1.8.0_77/lib/GBSX.jar "${@}";
```

Installing own modules: Example GBSX

- Goto software folder: cd software_folder
- Change permissions:
  chmod -R 666 GBSX
- Generate module:
  generate_modulefile -n GBSX/v1.3-Java1.8.0_77 -s GBSX -v v1.3-Java1.8.0_77 -d “GBSX demultiplexer” -a “source switch_to_2015a” -a “module load Java1.8.0_77”
- Confirm the writing, if the file looks ok
- To use the modules:
- module use software_folder
- module load GBSX/v1.3-Java1.8.0_77

Advanced Topics

- TMP directories
- Own module installation
- **How to do the management for a group**

How to do management for a group

- Group management
- Data organization and access
- Software management
- Resources management

Group management

- Make sure your group has a staging. This is the only type of storage that can be share by multiple users!
- Choose a moderator (the person who will be in charge)
  Tasks of a moderator are:
  - Management of the groups (access to staging, projects, …)
  - Taking care of the credit management of the group
  - Managing the file structure
  - Optional scheduling backups to archive
  - Installing/updating software

Management of groups

You need at least one group: for the general access to the staging.
A group can have different categories (when create l is added in front of the group name):

- Data group
  Groups to manage data access.
- Credit group
  Groups to manage the credits. (when created, must start with p_)

Advise:

- Use the full name of your PI/Collective as general access group. (lp_PIname)
- Use the initials _ project name for individual credit/data groups. (lp_PI_projectName)

Management of groups

Groups can be managed through: https://account.vscentrum.be/django/

- Login with your institution account
- The tab “View Groups” lists all your groups (as member)
- In the tab “New/Join Group” you can create your own group, or join an existing group
- In the tab “Edit Group”, select one of your groups (as moderator) and select edit. Here you can:
  - Remove members
  - Invite new members
  - Change moderators (a moderator is always a member)

Data organization and access

The data can be organized in multiple ways. It is advised to make clear agreements. On top level (staging entry) the following rules can apply, on deeper levels (per project) this can change from project to project, depending the projects members.

Top level arrangement examples:

- When multiple PIs are sharing a single staging:
  /staging/leuven/stg_000XX/PI_name/Project_name
- When the staging belongs to a single group:
  /staging/leuven/stg_000XX/Project_name

Data organization and access

Data access is managed by the use of groups.

- You can change the permissions of files and directories by using the chmod command (you can only change permissions if you have rwx)
- You can change the group of files and directories by using the chgrp command (you can only change to groups if you are a member)
- You can change the owner of files and directories by using the chown command (you can only change the owner from your own files)

Software management

A big topic at the start of these slides shows how to install your own software.

You have 2 options on software management:

- You ask for all tools to be installed by the VSC in the global toolchain.
  Is the easy way to install, but simple scripts are discarded, and you will have to wait for some time
- You can install it yourself.
  Harder to do, but you will have full version control

Software management

If you install software yourself:

- Create a toolchain directory in your staging, follow the instructions above to install a tool.
- Make this toolchain directory accessible to all members of your group

Hint: install Anaconda and Bioconda. This is a very easy thing to do. This software allows you to create environments where you can install software and all dependencies without having to search them in the main toolchain of VSC.
A lot of Bioinf tools are available and easy to install in bioconda.

Software management

Bioconda software install:

- Load your anaconda module (you will need to install like before)
- Check the bioconda website if the tool is available
- Create an environment with the toolname_version (you can install multiple versions next to each other)
- Create the directory structure like before
- In the bin folder, create a script with the same name as the tool command, which activates the environment, calls the tool, and deactivates
- Follow further instructions like the installs before

Resource Management

A lot of resources are the same over projects, and would take a lot of space if they are copied multiple times. Examples:

- Genomes
- Genome indexes for mappers, blast databases, …
- Annotation files

Create a resources directory. Inside create a directories with the species_name/genome_version. This way multiple species can be installed, with multiple genome versions. Here the fasta file of the genome can be placed.

Resource Management

Per genome build/version, multiple tool indexes can be created, multiple annotation and other files can be found.

- For tool indexes:
  - Create a directory with the tool_name/tool_version (tools can have multiple versions)
  - In this directory create a link to the genome with ln
  - Create your tool index
- For annotation and other files:
  - Create a directory with the source_name/version (for example ncbi/build37)
  - In this directory create a link to the genome
  - Install the databases (gff, bed, …)
  - In this directory tool indexes (as above) can be installed, which uses these databases.

Now you should be ready
for the HPC adventure