Academic Code and High Performance Computing

Benjamin Kiessling

October 16, 2023

Introduction

Today's lecture is divided into two parts.

The first half will be on academic software and how to get it running.

The second half will be on High Performance and Cloud Computing.

Academic Software

Academic Software

Academic software is any code that is primarily intended to support scientific work, most often produce results for a publication.

This makes these software packages highly specific, going so far as using hard coded datasets.

The data is more important than the actual code.

Its target audience are usually not end users but other academics or even just the author(s) themselves.

Academic Software vs Scientific Computing

Academic software is not scientific computing software!

Common scientific computing libraries such as numpy, scipy, ...are stable, well-documented, and have a large community of users and developers willing to help.

More fringe software can share some features with academic code but they are still continuously developed packages.

Academic Software

Because of the limited audience of experts and a singular purpose the quality of these software packages is in almost all cases astonishingly low.

Documentation is lacking, non-existant, or out of date.

The software is often unmaintained and has arcane dependencies. Even at the time it was written it was probably barely running in the first place.

There is no support as there are no other users and the author(s) are probably working on their next publication.

Academic Software

Why would one use academic software?

- Access to state of the art methods that aren't available in more stable libraries
- For comparison purposes for own publications
- As a base for an own reimplementation of the method

Academic Software

When should one avoid academic software?

- when a stable library already implements the desired method!
- when a stable method that can be distributed to less experienced users is required.
- when the state of the art provides only marginal gains to more established software
- when you expect to need help in your own implementation

Finding academic software - literature search

It has become best common practice in Computer Vision to publish the source code to the results and methods of an article or conference presentation.

In these cases there is a link provided to the web page with the code in the article itself.

The code is usually maintained in a version control system, most commonly git and hosted on a public platform like github.

Finding academic software - literature search

This assumes we already have an article we are interested in and can judge the efficacy of the method.

Just picking the best method is inadvisable:

- as methods with the highest score tends to just use the largest networks, making adaptation without a lot of ressources infeasible.
- as results are not always reproducible, i.e. people were fudging the numbers, trained until they found a network that works particularly well on the dataset, ...
- as small increases in metrics on the standard datasets will often not result in commensurate improvements in your data.

Finding academic software - literature search

Many Computer Vision tasks have existed for quite a few years.

There are most likely keystone publications that have advanced the state of the art substantially and are cited by hundreds of other authors.

These often change the way the problem is approached fundamentally, can be easier to understand and have more straightforward implementations.

Without a good reason to choose something more esoteric the most recent keystone publication is probably a good place to start.

Finding academic software

For popular papers, especially without available source code, people will often (re-)implement the method.

Also fairly popular are reimplementations using different libraries such as a pytorch reimplementation of a package utilizing tensorflow originally.

These are also maintained in a git repository hosted on a public platform.

Finding academic software

The straightforward way to find reimplementations or original implementations that were not mentioned in the publication is through a basic google search `method name + implementation`, e.g. `UNet implementation`.

There will be a lot of noise such as incomplete or incorrect implementations.

Finding academic software - reversal

In almost all cases we are to solve a problem and are not really interested in the particular method.

An alternative approach is to find implementations of methods for a particular problem.

People curate lists of articles, datasets, and implementations under the name scheme `awesome-`, such as `awesome-semantic-segmentation`, `awesome-optical-character-recognition`, ….

There is also a web site `paperswithcode.com` that lists articles with implementations by task.

Installing academic software

We have found an academic software package we want to use for our problem. We now have to install it.

As it is a computer vision or machine learning software it is most likely written in python and uses a common machine learning library like tensorflow or pytorch.

Installing academic software - Preparations

The software is probably ugly and can easily break your system if you just install it globally.

It is therefore necessary to separate it from everything else on your system, at least in a rudimentary manner.

Anaconda environments are designed to achieve just that. They create a virtual (not only) python environment where you can install the arcane and broken dependencies without affecting anything else on the system.

Installing academic software - Preparations

The first step is therefore to create a new environment:

```
$ conda create -n $name
$ conda activate $name
```

This will set up the environment with the default python version of anaconda.

Installing academic software - Preparations

If a specific python version is required this can also be specified:

```
$ conda create -n $name python=3.8
```

Unneeded environments can be removed:

```
$ conda remove -n $name
```

Installing academic software - Preparations

Because the anaconda environment is separate from our main system it has to be manually activated before we can use the software therein:

```
$ conda activate $name
```

To get back to the main environment it suffices to deactivate an entered environment:

```
$ conda deactivate $name
```

Installing academic software

Even the worst software has usually some installation instructions.

After downloading the repository there should be a file named README or INSTALL. These should list the requirements and/or describe which commands to run to make the software operational.

Apart from manually installing the dependencies, there are two main ways to install python software with dependencies: pip and setuptools.

Installing academic software

Requirements

- pytorch >= 0.2.0
- torchvision >= 0.1.8
- fcn >= 6.1.5
- Pillow
- scipy
- tqdm

Installation

```
git clone https://github.com/wkentaro/pytorch-fcn.git
cd pytorch-fcn
pip install .

# or

pip install torchfcn
```

Installing academic software - setuptools

Setuptools is an older python packaging system. Its main drawback is that it is not able to define versions of dependencies.

For older software (>1 year) the install process might succeed but running the installed programs will not as the interface for the dependencies has changed!

The general installation routine for setuptools-based applications is:

```
$ python setup.py install
```

Installing academic software - pip

Pip solves some of the problems with setuptools. It is generally nicer to use and can install software from a public repository on the internet, similar to anaconda.

The general installation routine for pip-based applications is:

```
$ pip install .
```

in the directory of the source code.

Installing academic software - pip

There is also a special way to install code with pip, an editable install. In this mode you can change the code in the repository and the changes will be automatically reflected in the installed software without having to reinstall it.

```
$ pip install --editable .
```

Installing academic software - anaconda

Both pip and setuptools automatically download source code from the internet and, if necessary, build it before installation.

Dependencies of academic code can tricky to compile with specific requirements to the build system that can be hard to debug.

Installing academic software - anaconda

Luckily anaconda contains binary packages for many common libraries. It is almost always easier to just install these requirements manually through anaconda when getting weird errors in a pip or setuptools installation process.

```
$ pip install --editable .
Collecting numpy
Building wheels for collected packages: numpy
Building wheel for numpy (setup.py) ... error
$ conda install numpy
$ pip install --editable .
```

Installing academic software - anaconda

Very rarely a whole anaconda environment definition is offered. Here it is not necessary to create a dedicated environment beforehand but it suffices to instantiate the environment yourself:

Install the latest 1.0 release through conda:

```
$ wget https://raw.githubusercontent.com/mittagessen/kraken/master/envir
$ conda env create -f environment.yml
```

Installing academic software - no installation

Sometimes there are no facilities provided to install the software at all. If there are no mentions in the documentation and there exist no `setup.py`, `environment.yml`, `requirements.txt` files the software can't be installed at all.

In such cases we have to run the scripts directly from their directory and copy modules into our own software manually.

Running academic software

The standard way to expose functionality is through command line drivers whose syntax is sometimes documented in the README file.

For machine learning software there are usually two applications, one for training, and one for inference.

Running academic software - Training

Training

```
> python train.py -h
usage: train.py [-h] [-e E] [-b [B]] [-l [LR]] [-f LOAD] [-s SCALE] [-v '

Train the UNet on images and target masks

optional arguments:
  -h, --help            show this help message and exit
  -e E, --epochs E      Number of epochs (default: 5)
  -b [B], --batch-size [B]
                        Batch size (default: 1)
  -l [LR], --learning-rate [LR]
                        Learning rate (default: 0.1)
  -f LOAD, --load LOAD  Load model from a .pth file (default: False)
  -s SCALE, --scale SCALE
                        Downscaling factor of the images (default: 0.5)
  -v VAL, --validation VAL
                        Percent of the data that is used as validation (
                        (default: 15.0)
```

Running academic software - Training

For reasons of convenience, training datasets are expected to be in the format of the standard dataset(s) for the specific task, i.e. MS-COCO for object detection, VOC2017 for semantic segmentation, ...

It is therefore necessary to convert your own data to fit the common datasets.

Running academic software - Training

A common pitfall is that the code has been written to only run on GPUs.

It will crash with an error (`Device not found` or `... not compiled with CUDA support`) when ran on a machine without a properly configured Nvidia graphics card.

Here it is necessary to convert all calls referencing a graphics card (cuda) to CPU (cpu).

Running academic software - Inference

Inference is the process of running our trained model on unknown data.

Often there is a separate script for inference that might or might not be documented.

It can also be called prediction in the documentation.

Running academic software - Inference

Prediction

After training your model and saving it to MODEL.pth, you can easily test the output masks on your images via the CLI.

To predict a single image and save it:

```
python predict.py -i image.jpg -o output.jpg
```

To predict a multiple images and show them without saving them:

```
python predict.py -i image1.jpg image2.jpg --viz --no-save
```

Running academic software - Inference

Sometimes there is no inference script as it isn't necessary to produce the metrics for publications.

Either find another implementation or look into the evaluation code and extract the forward pass code manually to create your own script.

Running academic software - Integration

Academic software packages are decidedly not libraries!

There is nothing to import and use right away.

Substantial amount of logic is likely found in command line drivers.

Lots of unused code laying around.

Running academic software - Integration

If at all possible use the command line drivers for integration.

In all other cases try to extract the core of the inference functionality and move them over into a new module that only performs inference. Use this module only and throw the rest of the implementation away.

Cluster Computing

Cluster Computing

A cluster is a parallel machine built of commodity components and running commodity software.

Clusters consist of nodes with one or more processors, memory that is shared, additional peripherals such as hard drives. They are connected to other nodes by a network.

Cluster Computing - Why?

Same reasons as supercomputers.

Increased computational power to solve larger, more complex problems:

- large run times
- large memory usage
- high I/O usage
- fault tolerance

Cluster Computing - When?

Research problems that outgrow smaller, less dedicated hardware like a laptop or workstation.

They are usually employed when:

- computations need more memory, runtime, ...than available on single machines
- programs that can be sped up by running them in parallel
- programs that can be sped up by utilizing special hardware such as GPUs.

Cluster Computing - Where?

Originally clusters where located at large national laboratories (INRIA, CNRS, Max-Planck-Institues, ...) and universities.

Used for all kinds of computational science not just maths/CS.

They have become more widespread with smaller systems installed on a per department or even research group basis.

Cluster Computing - Where?

Affiliation in a research group usually comes with access to a university or national cluster.

Unfortunately, not at the University of Tours.

Access rights are determined by the funding provided by your member institution. Just asking for it can never hurt though.

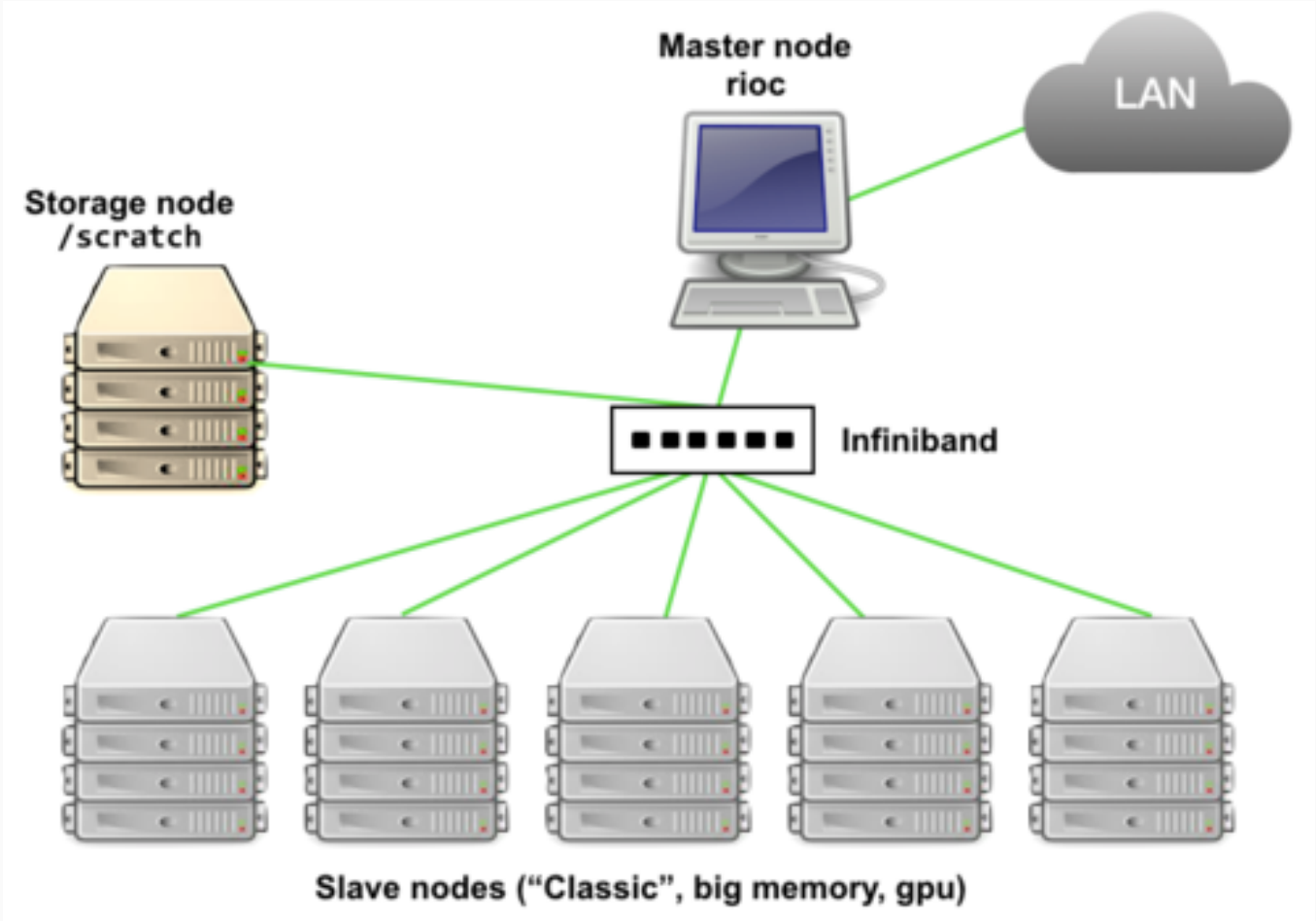
Cluster Computing - Architecture

An HPC cluster is a collection of many separate servers (nodes), which are connected via a fast interconnect.

There may be different types of nodes for different types of tasks (distributed memory machines, fat, ...) but the one of interest for machine learning purposes are nodes equipped with graphics cards.

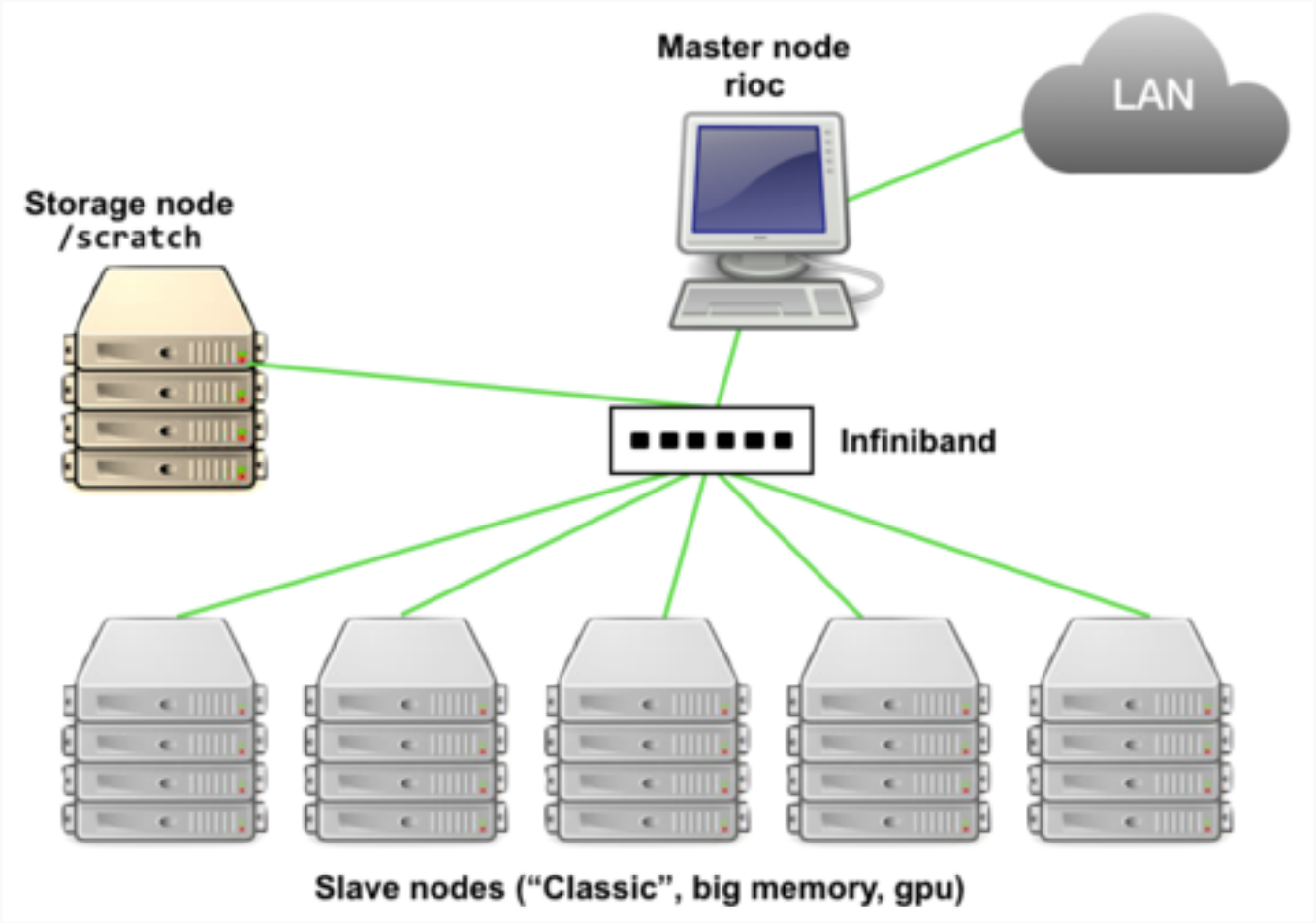
Cluster Computing - Architecture

The normal configuration of an HPC cluster consist not only of compute nodes but also shared storage, login nodes (or jump hosts), and sometimes data transfer nodes.



Cluster Computing - Architecture

Users do not run programs directly on the compute nodes but first login to the jump host and prepare their operation.



Cluster Computing - Architecture

Compute nodes in a cluster run an operating system like other computers. These are usually some kind of Unixoid, such as Linux.

They are not equipped with desktop environments and therefore have to be accessed using the command line, using an ssh client.

Above the operating system there is usually some kind of management software that allocates cluster access. Most common are slurm and OAR.

Cluster Computing - Management software

Clusters are usually shared by many users at the same time. They are also quite expensive.

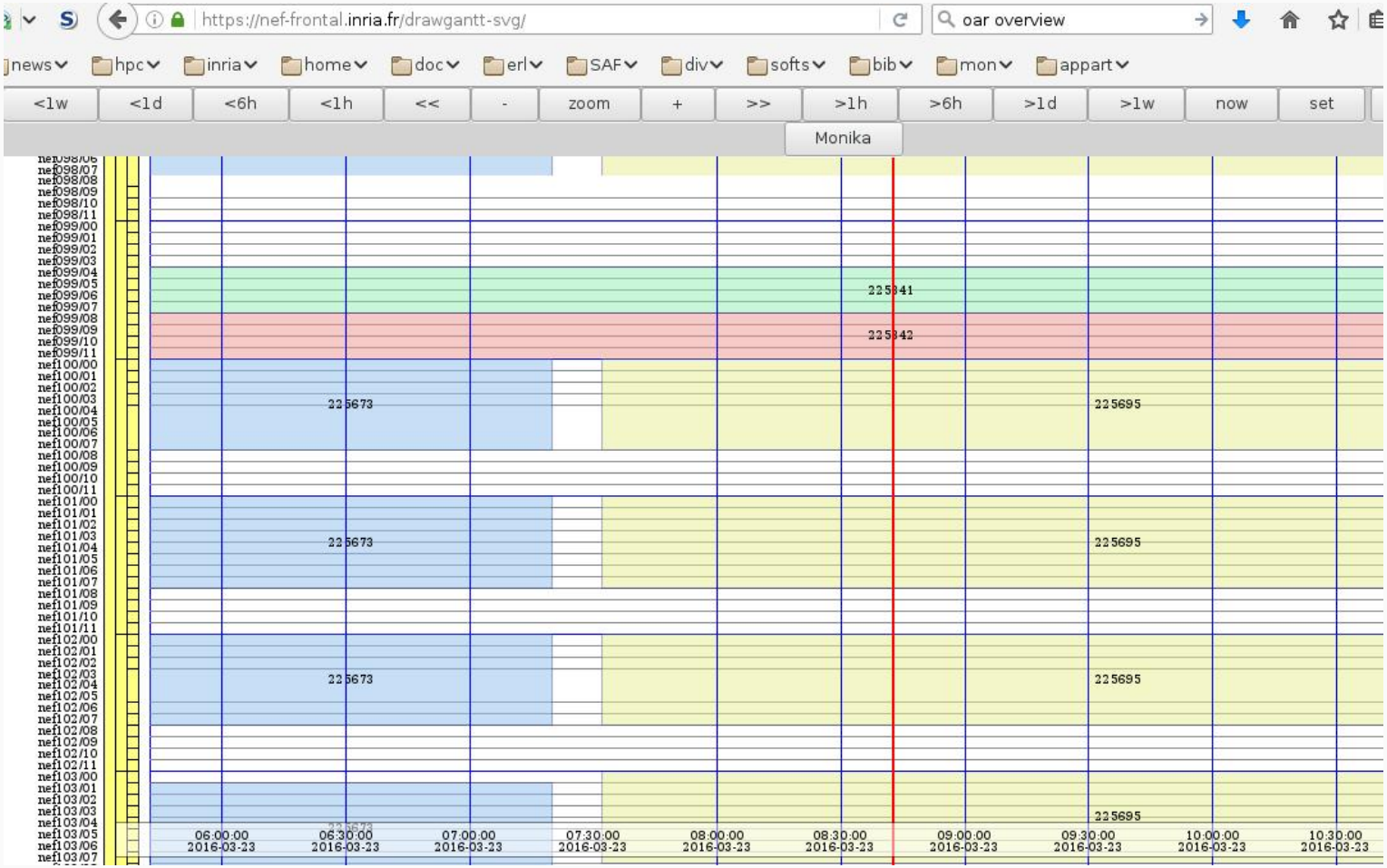
To make the most of the available hardware, cluster access is not real-time. One submits a batch which consists of all the code to run, the necessary data, special hardware requests, and the duration of the computation.

The management software enqueues the batch and decides when to run it. It also performs accounting so users can be billed accurately.

Cluster Computing - Management software

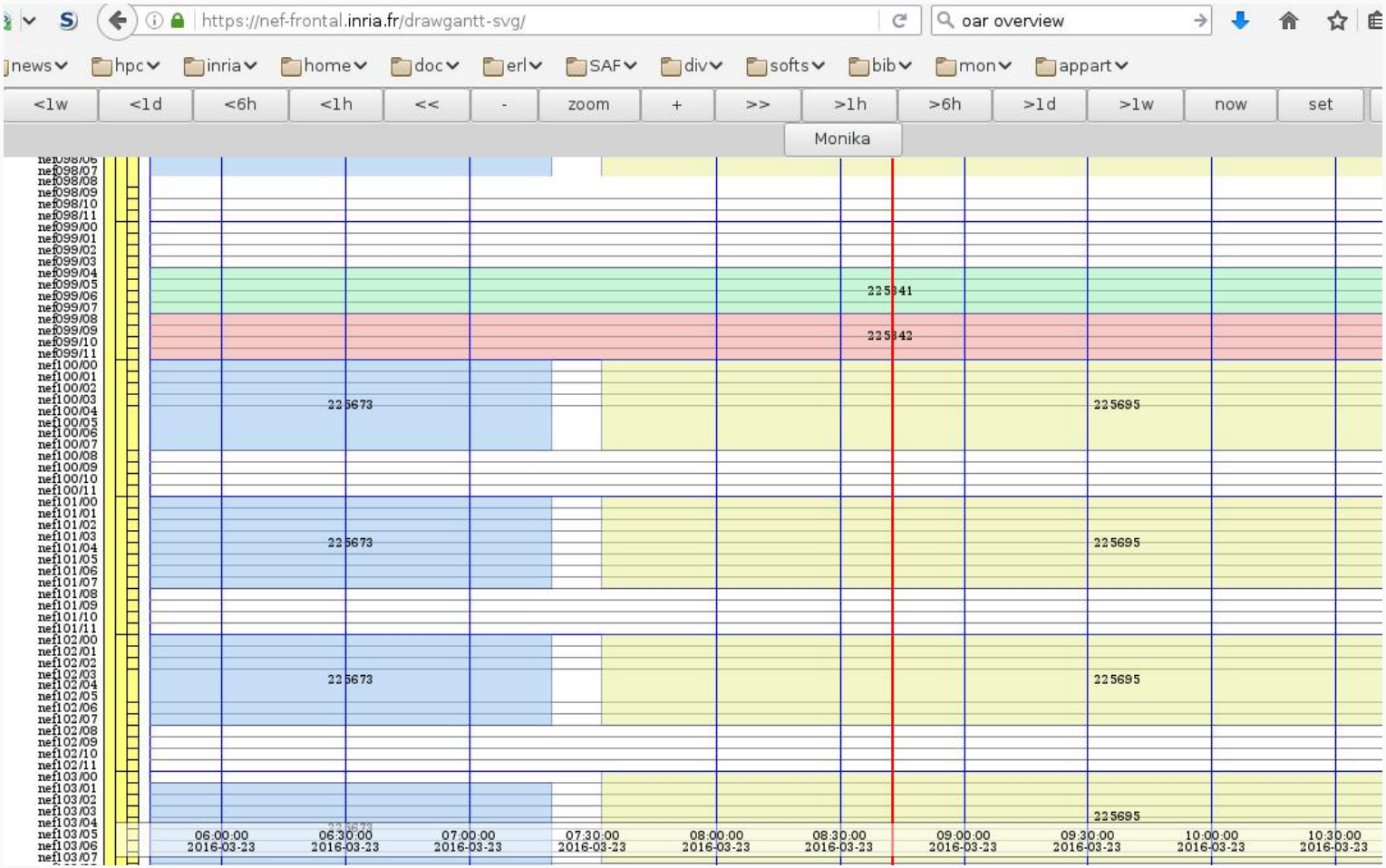
Many clusters have multiple queues for different needs.

Common configurations run shorter tasks first or prioritize people who pay more (usually the physicists and life sciences).



Cluster Computing - Management software

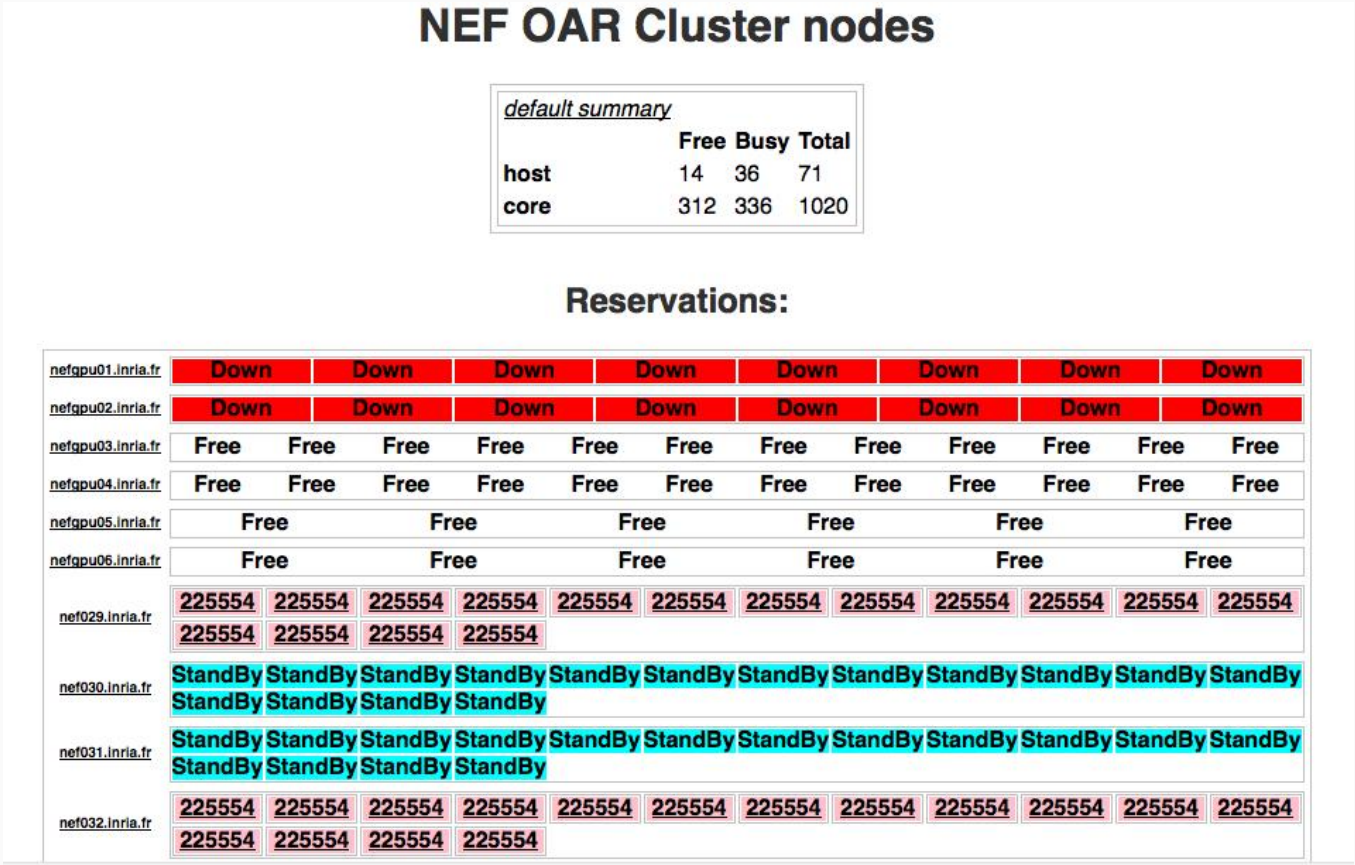
The management software is also able to terminate batches at will. Once the requested quota has been used up or when you request it the batch will be killed.



Cluster Computing - Management software

The management software also includes ways to monitor existing tasks, inspect the queue, and see which other batches are running.

Apart from the integral tools, there are also external tools and interfaces provided by the cluster operator.



Cluster Computing - Storage

Lots of scientific computing requires very large datasets. Keeping it all available on the compute nodes with low latency is undesirable, therefore there is a hierarchy of storage with larger one being farther away than smaller ones.

First is the local memory of the individual cluster nodes. This is the fastest but will be purged everytime a batch is terminated. It therefore has to be populated from slower storage during batch initialization.

Local memory is mostly used for data that has to be loaded multiple times and/or non-consecutively. It is in almost all cases smaller than 10Gb.

Cluster Computing - Storage

Next are the local hard drives of the individual cluster nodes.

These contain the operating system, applications, and scratch space available to the user.

This scratch is regularly scrubbed and not available to processes running on other nodes.

Like local memory, scratch space has to be populated during batch setup. It is not available from the login node.

Cluster Computing - Storage

Most of the data on a cluster is kept on a separate file server connected to all nodes (including the login nodes).

Depending on the hardware configuration of the cluster access to data on the file server can be fairly slow. The advantage is that it is usually plentiful and data can be kept mid-term.

Even this storage is usually not backed up. It is recommended to treat data on a cluster as ephemeral and saveguard it on other systems for archival.

Cluster Computing - Storage

Most of the data on a cluster is kept on a separate file server connected to all nodes (including the login nodes).

Depending on the hardware configuration of the cluster access to data on the file server can be fairly slow. The advantage is that it is usually plentiful and data can be kept mid-term.

Even this storage is usually not backed up. It is recommended to treat data on a cluster as ephemeral and saveguard it on other systems for archival.

Cluster Computing - Accessing storage

Data is transferred to clusters through SSH.

As login nodes are not well-equipped, sometimes separate data transfer nodes are provided that should be used for this purpose.

Data on the file server is accessible in the same location on both the login node and the compute nodes. To provision data for a batch it suffices to leave it at a known location on the file server from the login node.

Cluster Computing - Accessing storage

When the batch is running, processes can access files (on the file server) changed by other nodes immediately, although there is often considerable synchronization delay involved.

Likewise, it is necessary to ensure that the processes of a batch do not interfere with each other by writing the same files concurrently.

There are low-level libraries offering synchronization primitives in a cluster setting. They require a good understanding of parallelization to be used effectively and are usually not necessary for running basic machine learning tasks.

Cluster Computing - Software

The cluster nodes run a normal, albeit usually out-of-date, operating system.

Users are not able to install software permanently on the cluster but scientists need their scientific computing libraries, quite often a specific version of it.

The most common requirements are packaged into modules that can be loaded inside the batch to provide certain functionality.

Cluster Computing - Software

The modules available are rarely sufficient for modern machine learning frameworks.

Luckily, anaconda works fine on most clusters.

It can be installed locally and packages are built in a way that allows them to run even on very old operating systems.

Cluster Computing - Software

The modules available are rarely for sufficient for modern machine learning frameworks.

Luckily, anaconda works fine on most clusters.

It can be installed locally and packages are built in a way that allows them to run even on very old operating systems. This holds even for non-python software.

Cluster Computing - Usage

Tying everything together the procedure for a cluster batch is:

1. Upload data and applications using login/data transfer node
2. Create conda environment on the login node
3. Write a setup script that configures the environment on the compute node and runs the application.
4. Submit batch.
5. Wait for results (or an error).

Cloud Computing

Cloud Computing

Not everyone has institutional access to ressources like a cluster.

An alternative, although usually paid, is cloud computing.

Cloud computing also offers some additional advantages that HPC clusters can rarely provide.

Cloud Computing - Definition

There are multiple definitions.

Not all are helpful.

Good ones are extensive (and catch non-cloud applications as well).

Cloud Computing - Definition

B
y using virtualized computing and storage resources and modern web technologies, Cloud Computing provides scalable, network-centric, abstracted IT infrastructures, platforms, andapplications as on-demand services. These services are billed on a usage basis.

Fundamental technologies:

- Virtualization
- Web Services

Services:

- IaaS, PaaS, SaaS
- scalable
- network-centric
- on-demand

Cloud Computing - Definition

Not all of these characteristics are important for scientific use.

Web services, abstracted, on-demand execution can be dispensed with in most scientific settings.

Not all cloud service models are suitable for scientific computing.

Cloud Computing - Virtualization

Allows an abstract, logical perspective of physical resources such as servers, storage, network, specialized hardware (GPUs!).

Isolates (ideally) the physicial hardware, making it possible to:

- use the physical hardware in a shared and transparent way
- combine disparate hardware into a homogeneous resource pool

The major advantage to a classic HPC cluster is that virtualization allows fine-grained access to resources!

Cloud Computing - Web Services

HPC clusters are interacted with command line Unix tools.

Cloud computing systems are interfaced through RESTful web services.

These are stateless HTTP requests for provisioning resources, retrieving results, monitoring tasks, ...

For popular services there are higher level abstraction layers available.

Cloud Computing - Services

Cloud computing is an umbrella term for different services.

They are usually distinguished according to their functionality in addition to their organization (public, private, hybrid cloud).

Cloud Computing - Functional distinction of services

Software as a Service (SaaS) The provider runs a web application and the customer only needs a browser. Mostly uninteresting from a scientific computing perspective but this model includes academic research infrastructure like Clarin, escriptorium, archetype, ...

Platform as a Service (PaaS) The provider runs a scalable runtime environment. Customers run their own web applications in the infrastructure of the service provider. For machine learning these often offer a specialized ML application such as NLP tasks.

Infrastructure as a Service (IaaS) The provider run physical servers and customers run VMs with any operating systems and unmodified applications. Some HPC clusters move into this direction.

Cloud Computing - Humans as a Service

Crowdsourcing and citizen science run on platforms that resemble cloud providers.

Human creativity is offered on demand for low cost or donated from volunteers.

Interesting for training data creation (Transcribing Bentham), semantic annotation, ...

Platforms like Zooniverse are free to use and require only community management!

Cloud Computing - SaaS/PaaS

Depending on the use case, some SaaS/PaaS systems can be of use in digital humanities research.

Google Cloud Vision is able to perform object detection and OCR/HTR.

Their use in research is problematic for multiple reasons:

Domain mismatch Historical and non-Latin data is rarely modelled well as there is little commercial interest.

Reproducibility and explainability The fundamental software is proprietary and subject to constant changes. Results for a publication are neither explainable nor garuanteed to be reproducible in the future.

Data ownership By linking data produced with public funding to proprietary platforms some of the results of public research are effectively privatized.

Cloud Computing - IaaS

There are several large IaaS providers, such as MS Azure, Amazon AWS, Google Cloud.

They all provide a collection of services which one can pick and choose from:

Elastic Compute Infrastructure for virtual servers

Simple Storage Service Storage service for web objects

Elastic Block Store Storage service for virtual storage volumes

Elastic Load Balancing service for virtual load balancers

Cloud Computing - IaaS

As we are only interested in running computational tasks we don't need most of these.

In addition, a traditional HPC cluster is more suitable for many applications, namely highly parallelized problems.

What IaaS is good at is giving us a lot of GPUs fairly quickly (and cheaply-ish).

Cloud Computing - Amazon EC2

Users can create, use and control virtual server instances in Amazons data centers.

Lots of different operating systems supported.

Virtual servers are created from Amazon Machine Images. Amazon provides pre-built images. You can also create your own.

There are up-to-date images with all the libraries to start training machine learning models.

Cloud Computing - Amazon EC2

There are free accounts for students!

There are also free trial accounts for GPU servers up to 750h/month for 12 months!

It is easy to rack up charged though.

Cloud Computing - Amazon EC2

The process for spooling up your own compute server for a task:

1. Create an SSH key pair (just like for a cluster).
2. Set up the firewall rules for the server. Usually everything can remain closed as it is only used for computation.
3. Pick an operating system image and the correct instance type (with or without GPU, size, ...).
4. Pick the region and availability zone.
5. The instance is created and assigned a random public IP address.

One can now connect to the server and transfer data, run the computation, ...

Cloud Computing - Google Colab

Colab is a google research tool for data science. It is open to the general public and it is free (as in beer).

It functions as a public Python notebook server.

The most interesting functionality for students is that it provides a powerful GPU for free up to 12h at a time.

Cloud Computing - Google Colab

To get started you need a GMail account.

Afterwards you can go to `https://colab.research.google.com` and start creating your own python notebooks as shown last week.

Many machine learning frameworks (pytorch, tensorflow, ...) already come preinstalled.

Cloud Computing - Google Colab

As only a notebook is provided there are some tricks one has to employ to install additional non-provided software and data.

Shell commands can be executed by beginning a line with an `!`:

```
! cmd_1 arg arg arg
```

Cloud Computing - Google Colab

As only a notebook is provided there are some tricks one has to employ to install additional non-provided software and data.

To download data from somewhere on the internet:

```
! wget http://my.very/fancy/data
```

Cloud Computing - Google Colab

As only a notebook is provided there are some tricks one has to employ to install additional non-provided software and data.

To install a random python package:

```
! pip install my_fancy_package
```

Cloud Computing - Google Colab

It is also possible to install anaconda but the process is a bit more involved and should be avoided as it rewires a lot of the internals of the python kernel.

The virtual environments functionality won't be available though and everything needs to be installed in one global environment.

Google Colab - Data transfer

The notebook is not running in a defined environment so there needs to be a dynamic way to import and export data.

There are multiple ways to import data, some more cumbersome than others:

1. Calling `wget` manually.
2. Creating an upload button.
3. Mounting your Google Drive.

Google Colab - Data transfer

Creating an upload button is practical if you want to run the same methods on different datasets and don't mind reuploading the data every time the notebook is reset.

```
[ ]  from google.colab import files
     uploaded = files.upload()
```

Google Colab - Data transfer

The uploaded file(s) are in a python dictionary as strings. To parse them back into a binary file we have to use the python StringIO class:

```
import io
from PIL import Image

fp = io.StringIO(uploaded['mydata.png'].decode('utf-8'))
im = Image.open(fp)
```

Google Colab - Data transfer

The same module allows to download single files through the `download` function:

```
from google.colab import files
files.download('my_results.txt')
```

It is likewise possible to download files manually through the notebook user interface.

Google Colab - Data transfer

Especially when designing your pipeline you will often experiment on the same data and change the parameters of the methods.

In these cases uploading the same data multiple times can be tedious.

Uploading it once into a Google Drive and mounting it directly in the notebook is a lot more efficient.

Google Colab - Data transfer

To mount the drive at a specific location:

```
[ ]  from google.colab import drive
     drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth
Enter your authorization code:
··········
Mounted at /content/drive
```

Google Colab - Data transfer

You can now read and write to the drive as wth local files:

```
with open('/content/drive/my_fancy_data.txt', 'r') as fp:
    data = fp.read()

with open('/content/drive/my_fancy_results.txt', 'w') as fp:
    fp.write(results)
```