An overview of batch processing

02-Jun-2022
One-on-one

Your computer

Your program
One way to speed things up within ROOT: multi-threaded code (RDataFrame, PROOF)

Pro tip: See Part Five of the ROOT tutorial to explore this option, or my "fast way to analyze" talk if I give it.
Out of ROOT, you can have multiple programs on a single computer (UNIX command “at”)

You have to remember not to submit too many jobs or you’ll overload the computer
A batch system managing multiple programs on a single computer (UNIX command “batch”)
A batch system managing multiple programs on multiple computers

- Your computer
  - Batch manager
    - Batch node
    - Batch node
    - Batch node
      - Batch node
        - Your program
        - Your program
        - Your program
        - Your program
        - Your program
        - Your program
- Your program
- Your program
- Your program
- Your program
- Your program
- Your program
The standard software for managing batch systems in scientific computing is HTCondor (or just Condor).

Main web page
http://research.cs.wisc.edu/htcondor/

Quick start

Full manual
https://htcondor.readthedocs.io/en/latest/

- Stick to the “vanilla” universe; the “standard” universe won’t work for ROOT or any other particle-physics software (so you don’t need condor_compile).
Condor will halt a queue in favor of an interactive program

Submit machine

Condor master

Batch node

Batch node

Batch node

Batch node

Batch node

Someone logged in!

Condor pool

on hold

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program

Your program
Condor managing multiple programs on multiple computers with multiple configurations
Condor uses “ClassAds” to match your requirements with what each node offers.

Condor pool

Your requirements (job ClassAd)

What a node offers (machine ClassAd)

on hold
Resource Planning

- Condor can’t do *everything* for you.
- Think about input files (including programs) and output files and how they’ll be accessed.
- Think about disk space. “df –h” and “du –shx *” can help.
- Fun fact: The particle-physics Condor pools *can’t* see your home directory!
- Moral: Let condor transfer your files... when possible.

**Execution environment**

- Some anonymous directory on some batch node somewhere
- Executable = `<my_program>` (your program)
- Input = `<input_file>` (an input file)
- Should transfer files = YES
- Transfer input files = `<file1>`,...
- Initial_dir = `<directory>`
  (where to look for input files)
  (where output files will be sent)
- Transfer output files = `<file1>`,...
  (files copied back to initial_dir)
- Output = `<file>.out`
- Error = `<file>.err`
- Log = `<file>.log`
Resource Planning

• Condor can’t do *everything* for you.
• Think about input files (including programs) and output files and how they’ll be accessed.
• Think about disk space. “df -h” and “du -sh x *” can help.
• Fun fact: The particle-physics Condor pools *can’t* see your home directory!
• Moral: Let condor transfer your files... when possible.

When you can’t let condor transfer your files, here are disk-sharing methods outside of condor:

• NFS – used at Nevis
• CVMFS – Fermilab and CERN
• Grid, BlueArc – only used at Fermilab
Resource Planning

- Condor can’t do *everything* for you.
- Think about input files (including programs) and output files and how they’ll be accessed.
- Think about disk space. “df –h” and “du –shx *” can help.
- Fun fact: The particle-physics Condor pools *can’t* see your home directory!
- Moral: Let condor transfer your files… when possible.

What we don’t do

/home

Your server
Resource Planning

- Condor can’t do *everything* for you.
- Think about input files (including programs) and output files and how they’ll be accessed.
- Think about disk space. “df -h” and “du -sh x *” can help.
- Fun fact: The particle-physics Condor pools *can’t* see your home directory!
- Moral: Let condor transfer your files... when possible.

What we do

Your server

File server

/home

/share

/data
Particle-Physics Computer Systems
Linux Cluster

Administrative servers
- hecate: VM server, library
- artemis: VM server
- selene: VM server
- notebook: Jupyter

Virtual machines
- franklin: Mail
- ada: web server
- sullivan: mailing-list server
- hypatia: NIS, LDAP
- twiki: wiki server
- shelley: backup server
- hermes: DNS

Workgroup/Login servers
- kolya: ATLAS
- tehanu: VERITAS/CTA
- houston: Neutrino
- shang: Heavy-ion

File servers
- milne: student files

Workstations
- xenia
- xenia2
- serret
- ged
- vetch
- amsterdam
- westside
- bleecker
- riverside

Virtual Machines
- milne: student files

File servers
- xenia
- xenia2
- serret
- ged
- vetch
- amsterdam
- westside
- bleecker
- riverside

Workstations
- batch nodes

https://twiki.nevis.columbia.edu/twiki/bin/view/Main/LinuxCluster
https://twiki.nevis.columbia.edu/twiki/bin/view/Main/ListOfMachines
Bringing the job to the data

Some wrapper script

requirements = (machine = node04.nevis.columbia.edu)

Submit machine

Condor master

node01
bigfile1.root

node02
bigfile2.root

node03
bigfile3.root

node04
bigfile4.root

node05
bigfile5.root

node06
bigfile6.root
Final tips

• Split up your task so each condor job takes 20-60 minutes

• If your job must be preempted, it will have to run from the beginning on the same machine that cancelled the job

• Test your job with one process before submitting it for 10,000 processes!
Resources

Main web page
http://research.cs.wisc.edu/htcondor/

Quick start

Full manual
https://htcondor.readthedocs.io/en/latest/

Nevis particle-physics condor guide
https://twiki.nevis.columbia.edu/twiki/bin/view/Nevis/Condor

Basic Condor@Nevis tutorial
http://www.nevis.columbia.edu/~seligman/root-class/