Open Science Grid

Intro to the Virtual School Pilot, HTC, and OSG

Monday, July 13

Tim Cartwright, Lauren Michael

Welcome to the

OSG Virtual School Pilot 2020!

Why You Are Here

- You need large-scale, HTC-style computing –
  or support researchers who do

- **Do not let computing block your research!**
  - Computing is cheap and plentiful
  - Push the limits of what you can do
  - If you run out of science to do, transcend the boundaries of your science
  - When computing becomes a barrier, push us to fix the problems

- Help & encourage others:
  In your lab, in your department, in your field, friends, etc.

Goals

- Help you learn the basics of HTC quickly
- **Reach your own objectives!**

Format

- **Lectures:** Tue, Wed, Thu + Tue – 11 am · 3 pm CT
- **Showcase** (2nd Wed): science transformed by HTC
- **Bonus topics** (2nd Wed): optional, mentor can help pick
- **Work times**: Most days – *strongly* encouraged, if you can
- **Consultations:** Meet one-on-one with your mentor
- **Lightning talks** (2nd Fri): optional, chance to show work

Mentor

- Your primary point of contact
  But you will work with others, too!
- Will help set goals, monitor progress, find resources/solutions, etc.
- When in doubt, contact your mentor
  (or Tim or Lauren or any staff member)

Communication Tools

- **Blackboard Collaborate**
  Lectures, Work times when needed, Showcase
- **Slack**
  All the time! #general, topics, DMs
- **Zoom** (or other)
  Scheduled for one-on-one meetings or small groups
- **Website**
  Slides, exercises, lectures (eventually), schedule, etc.

Feedback Is Vital

- Tell us what we could do better
- Any time, any staff member, any reason, …
- Planning on 2 evaluation surveys

We are here for you!!!

Intro to HTC and OSG

Overview

- What is *high throughput computing (HTC)* ?
- What is the Open Science Grid (OSG)?
- How do you get the most out of the above?

HTC: An Analogy

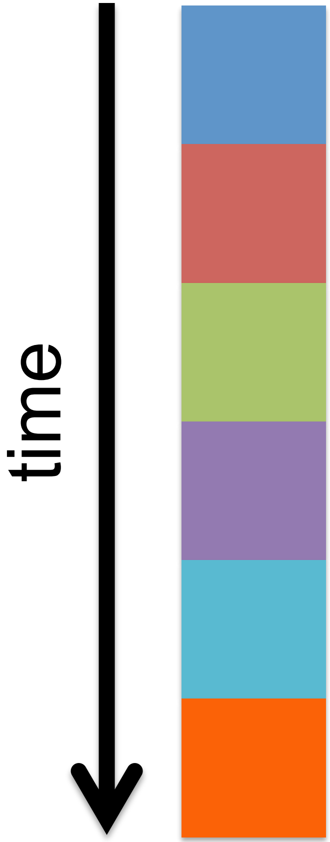
HTC: An Analogy

Open Science Grid

Serial Computing

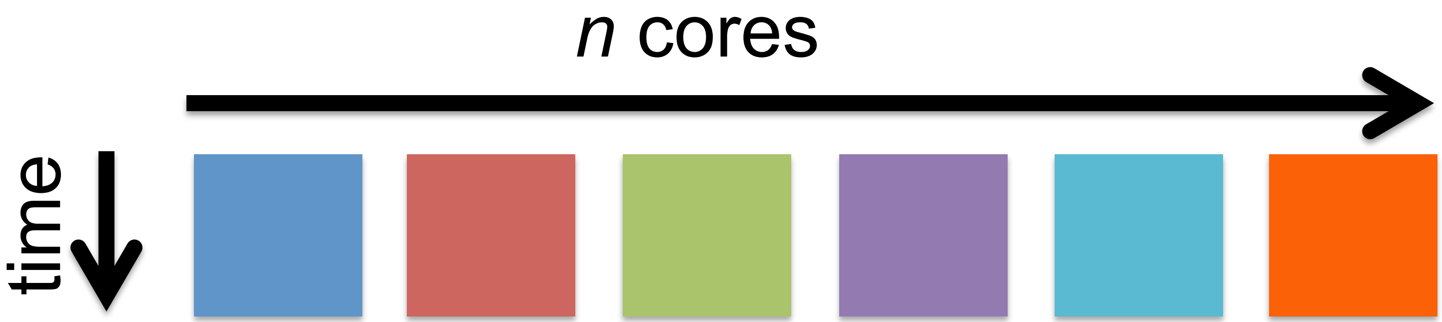
What many programs look like:

- *Serial execution*, running one task at a time
- Overall compute time grows significantly as individual tasks get more complicated (long) or if the number of tasks increases
- ***How can you speed things up?***



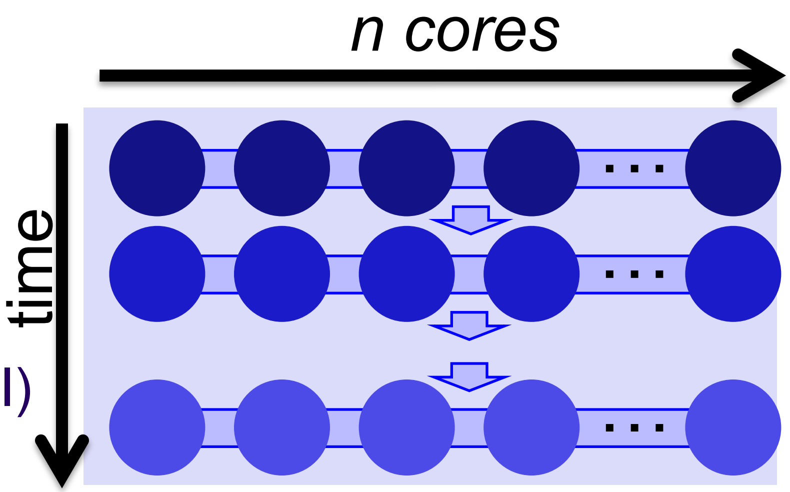
High Throughput Computing (HTC)

- Parallelize!
- Independent tasks run on different cores



High Performance Computing (HPC)

- Benefits greatly from:
  - CPU speed + homogeneity
  - shared filesystems
  - fast, expensive networking (e.g. Infiniband) and servers co-located
- Requires special programming (MP/MPI)
- Scheduling: **Must wait until all processors are available**, *at the same time* and *for the full duration*
- ***What happens if one core or server fails or runs slower than the others?***

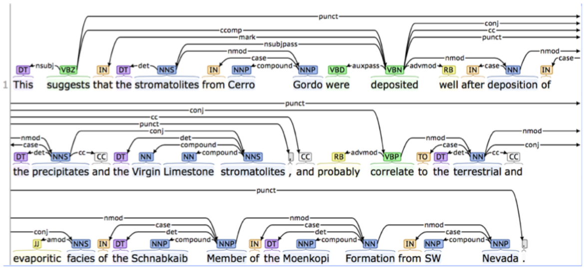


High Throughput Computing (HTC)

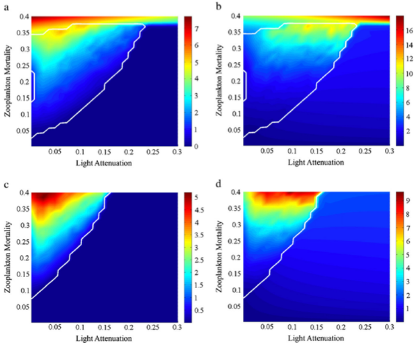
- Scheduling: only need **1 CPU core for each** (shorter wait)
- Easier recovery from failure
- No special programming required
- Number of concurrently running jobs is *more* important
- CPU speed and homogeneity are *less* important

Open Science Grid

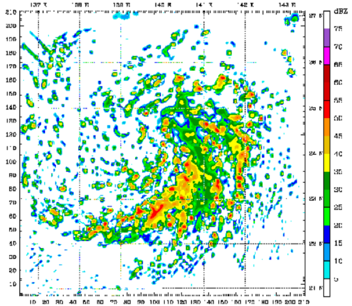
HTC Examples



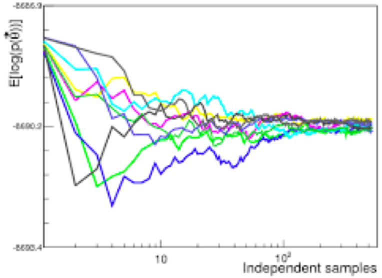
text analysis (most genomics …)



parameter sweeps



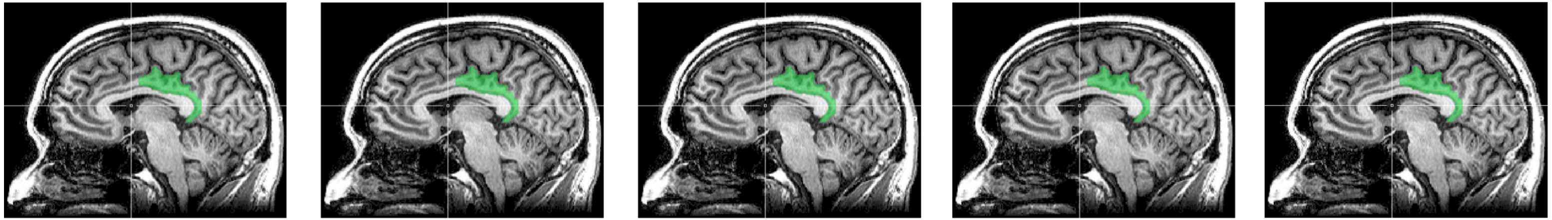
multi-start simulations



statistical model optimization (MCMC, numerical methods, etc.)

multi-image and multi-sample analysis

Example Challenge



You need to process 48 brain images for each of 168 patients. **Each image takes ~1 hour of compute time.**

168 patients x 48 images = ~8000 tasks = ~8000 hrs

Conference is next week.

Distributed Computing

- Use many computers, each running one instance of our program
- Example:
  - **1 laptop (1 core) => 8,000 hours = ~1 year**
  - 1 server (~40 cores) => 500 hours = ~1.5 weeks
  - 1 large job (400 cores) => 20 hours = ~1 day
  - **A whole cluster (8,000 cores) = ~1 hour**

Signs of HTC-able work

- Any mention of **numerous** samples, images, models, parameters, etc.
- Nearly anything **written by the primary user** (e.g. c/fortran, Python, R)
  - Break out of loops!
  - Common internal parallelism could really be HTC (e.g. Matlab's 'parfor', 'distributed server', etc.)
- Some community softwares that use **multi-threading or multiprocessing** (e.g. OpenMP)
  - many are simply looping over data portions or independent tasks
  - HTC-able: break up input (or 'parameter' space), turn off multi-threading, combine results
- **Long-running** jobs (especially if non-MPI); see above explanations

What computing resources are available?

- A single computer?
- A local cluster?
  - Consider: What *kind* of cluster is it? Typical clusters tuned for HPC (large MPI) jobs may not be best for HTC workflows! Do you need even more than that?
- Open Science Grid (OSG)
- Other
  - European Grid Infrastructure
  - Other national and regional grids
  - Commercial cloud systems (e.g. HTCondor on Amazon)

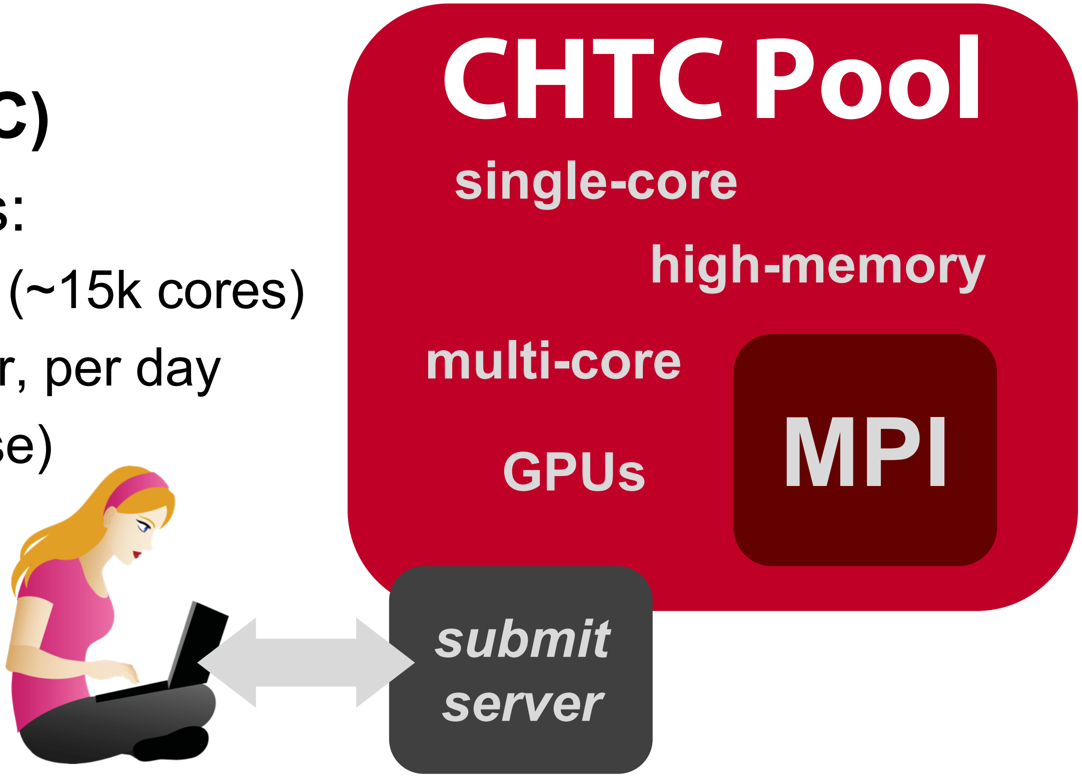
Example Local Cluster

- UW-Madison's **Center for High Throughput Computing (CHTC)**
- Recent CPU hours:

  ~120 million hrs/year (~15k cores)

  Up to 15,000 per user, per day

  (~600 cores in use)



Open Science Grid

- (d)**HTC for Everyone**
  - ~100 contributors
  - **Past year:**
    - >1.6 billion CPU hours
    - >200 petabytes transferred

- Can submit jobs locally, they backfill across the country
  - interrupted at any time (but not too frequent)
- https://www.opensciencegrid.org/

Is it *OSG-able*?

Per-Job **Resources**	**Ideal Jobs!** (up to 10,000 cores, per user!)	**Still Very Advantageous!**	**Probably not…**
cores (GPUs)	**1** (1; non-specific)	**<8** (1; specific GPU type)	**>8 (or MPI)** (multiple)
Walltime *(per job)*	**<10 hrs*** *or checkpointable	**<20 hrs*** *or checkpointable	**>20 hrs**
RAM *(per job)*	**<few GB**	**<10 GB**	**>10 GB**
Input *(per job)*	**<500 MB**	**<10 GB**	**>10 GB**
Output *(per job)*	**<1 GB**	**<10 GB**	**>10 GB**
Software	*'portable' (pre-compiled binaries, transferable, containerizable, etc.)*	*most other than* →→→	*licensed software; non-Linux*

For Researchers and Campuses

Proactive, personalized facilitation and support for:

- Individual researchers via **OSG Connect**
- Institutions and large collaborations
  - **Share local resources** via OSG
  - Locally-supported **submit points**
    - data and identity federation
    - integration of cloud capacity
  - Local HTC Capacity
    - Learn from OSG’s **Research Computing Facilitators**

- **Presentations/Training** in OSG compute execution, HTC Facilitation, and local HTC systems administration