CaRCC Data-Facing Call, 2025-03-04

Introduction to High-Performance Computing (HPC) clusters

Speakers: Brian Haymore and Martin Cuma - University of Utah, Center for High Performance Computing

Abstract

Join us for an introduction to High-Performance Computing (HPC) clusters. This month's session will provide an overview of HPC systems, their components, and their applications. We'll also discuss practical considerations like access, use case, and common challenges. If you are already experienced with HPC, please come and contribute your perspectives and insights as well.

Table of Contents

Please Note:

- ***We will record this call and post shortly thereafter on CaRCC's YouTube channel.***
- ***We expect all persons on the call to adhere to Our Community Policies.***

Announcements

- Welcome to the Data-Facing Track of the CaRCC People Network!
  - Join the People Network email lists. Join the CaRCC Slack workspace.
  - Track descriptions
- The CaRCC People Network is your community for research computing and data professionals. Please see this brief overview and https://carcc.org/ for more details.
- If you have questions about CaRCC or are interested in becoming more involved, please contact:
  - df-coordinators@carcc.org for D-F-related activities, or
  - getstarted@carcc.org or getinvolved@carcc.org for other CaRCC work
- We expect all persons on the call to adhere to CaRCC's Code of Conduct.

Sign-In (Name / Affiliation / Email)

Note: To help with our outreach efforts, please sign in with the suggested format (we will not sell/share your information per our Privacy Policy). Switch your editing mode to "Viewing" after sign-in, if you're worried about making accidental edits.

1. Brian Haymore / University of Utah / brian.haymore@utah.edu
2. Bobby Roybal / University of Nevada, Las Vegas / bobby.roybal@unlv.edu

3. Amy Nurnberger / Massachusetts Institute of Technology / nurnberg@mit.edu
4. Amy Koshoffer / University of Cincinnati / koshofae@ucmail.uc.edu
5. Moira Downey / NC State University / mcdowney@ncsu.edu
6. Jan Day / Amazon Web Services / janday@amazon.com
7. Gabriel Smith / University of British Columbia / gabriel.smith@ubc.ca
8. Martin Cuma / University of Utah/ m.cuma@utah.edu
9. Cameron Cook / UW-Madison / cameron.cook@wisc.edu
10. Nabeel Alzahrani /CSU San Bernardino/ alzahran@csusb.edu
11. Doug Hall / Amherst College / dhall@amherst.edu
12. David Warden / SUNY Geneseo / warden@geneseo.edu
13. Ian Kaufman / UC San Diego / ikaufman@ucsd.edu
14. Mirakyl Drake / University of Nevada, Las Vegas / mirakyl.drake@unlv.edu
15. Jaime Cleek / Dartmouth College / jaime.cleek@dartmouth.edu
16. Manasvita Joshi / Harvard / manasvitajoshi@g.harvard.edu
17. Elizabeth Summers / Northwestern University / elizabeth.summers2@northwestern.edu
18. David Merten-Jones / The Claremont Colleges Library / david.merten-jones@claremont.edu
19. Mara Sedlins / Colorado State University / mara.sedlins@colostate.edu
20. Xinlei Qiu / Stanford / xinlei@stanford.edu
21. Matt Gregas,/Boston College/gregas@bc.edu
22. Rodney Nelson/University of Michigan-Michigan Medicine rodtfo@med.umich.edu
23. Exequiel Punzalan / Rutgers University / ep523@oarc.rutgers.edu
24. Ken Taylor / University of Illinois / ktaylo@illinois.edu
25. Alison Peterson / San Diego State University / apeterson5@sdsu.edu
26. Pedram Esfahani / The University of Chicago – RCC / esfahani@uchicago.edu

Max attendee count: 36

Call Notes

Slides: Intro to HPC Clusters

Notes

- What are HPC clusters?
  - interconnected computers designed to solve complex computational problems
  - large complex parallel work: tasks that would be impossible or impractical for single computers
  - large numbers of independent tasks: can perform these tasks in parallel
- Common components (all of which play a role in whether and how to use a cluster):
  - CPU/GPU nodes
  - Login nodes (surface point where users interact with the cluster)
  - Storage systems
  - Batch & scheduling system (e.g., Slurm)

  - Network (high performance/low latency)
- Common applications & tools
  - Base operating system, IAM, scheduler, monitoring & security (these are largely available only to sysadmins)
  - Programming tools (compilers, debuggers, profilers)
  - Application stack
    - Libraries (which includes communication – message passing interface (MPI)), some may be optimized for specific nodes
    - Provided or user-installed applications
    - Deployment tools (Spack, Easybuild, Conda, container runtimes) – addresses increasing complexity of these computing environments, build in dependencies and libraries automatically
    - Environment management tools (e.g. Lmod) – allows you to load and unload computing environments to run specific tools
      - Also facilitates deployment of specific versions of software for various users
- Computing paradigms
  - High performance computing (tightly coupled parallelism, requires a fast local network) – information exchange among closely collocated resources
  - High throughput computing (independent parallelism, can be geographically distributed) – good for tasks like parameter sweeps
    - Open Science Grid opportunistically leverages computing resources from around the country
  - On premises computing (local resources) – good when you can keep the resources used/busy all of the time
  - Cloud computing (external resources) – good for sporadic or burst-y usage; good to keep in mind that management can be complex and billing can be complicated, but cloud can offer many different hardware choices
- Use cases for HPC clusters?
  - Any workload that is parallelizable
  - Use cases that are iffier:
    - non-parallelizable, long-running code (could work to approximate to parallelize)
    - code that needs large shared memory (distributed shared memory can serve as a workaround)
    - interactive applications (e.g. Jupyter notebooks) – however there can be specifically dedicated queues for these kinds of applications
    - I/O intensive applications
  - Things that aren't a good fit for HPC:
    - Tasks that require elevated permissions (working with a multi-tenant environment)

        - Service oriented applications (e.g., web servers, databases) – these are better suited for virtual machine-based solutions
        - Windows-based applications (although you can emulate windows on linux with mixed results)
- Execution models
    - Non-interactive: "fire and forget"; jobs are submitted and scheduled to run as resources are available
    - Interactive: needs immediate resources availability – useful in a classroom or training setting
- Open on Demand – GUI mode of interacting with the cluster, which helps to soften the learning curve experienced with the command line interface
    - Another option, FastX
- User training and support resources – in addition to documentation, asynchronous training and videos, live trainings, there are community resources that can help you navigate HPC: ACCESS, CaRCC, various local and regional groups, and various interest groups like US Research Software Engineers (US-RSE), Open Science Grid (OSG), the Regulated Research Community of Practice (RRCOP)
- Common support challenges:
    - Differences from local computing (things like data location, compute environment, and scheduling)
    - Understanding parallelism (speedup doesn't happen for free, need to be aware of how many parallel tasks can be run efficiently, etc)
    - Command line proficiency (alleviated some through Open OnDemand)
    - Understanding the share multi-user nature of HPC resources
- For Python programmers who want to parallelize their code, the multiprocessing and Dask modules are a good place to start: https://docs.python.org/3/library/multiprocessing.html
- https://examples.dask.org/

Questions

1. What is the difference between High Throughput Computing OSG (open science grid) vs High Performance Computing in terms of performance. When do you use one versus the other?

- Difference depends on what hardware is available in your local system.

2. So when you have different clusters available and the use case can fit a distributed model?
3. One quick question..if HPC has powerful I/O, why wouldn't database applications be a fit?
    - Clustered SQL databases exist … but there are caveats and difficulties

- Various NoSQL databases handle this better, but your approach to using databases changes fundamentally.

Chat Comments

- For Python programmers who want to parallelize their code, the multiprocessing and Dask modules are a good place to start: https://docs.python.org/3/library/multiprocessing.html
    - https://examples.dask.org/
- It would be a good discussion about serverless databases vs. running a server 24/7 for a database

Connection Details

https://utah.zoom.us/my/carcc?pwd=TjFuR3VVM2d5eE5zWnEvWWxDTFBCUT09

Meeting ID: 824 051 8198
Password: 31415926

One tap mobile
+13462487799,,8240518198#,,#,31415926# US (Houston)
+16699006833,,8240518198#,,#,31415926# US (San Jose)
Dial by your location
+1 346 248 7799 US (Houston)
+1 669 900 6833 US (San Jose)
Join by Skype for Business: https://utah.zoom.us/skype/8240518198

Cursor Garden ⤵ *“Don't leave your cursor in a seedy alleyway”*

🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳
🌳 🌹🌱🌱🍄🌷🌱🌳🌱🌱🌱🌱🌲🌼🌾🌱🌱🌲🌱🌱🌱🌻 🌱🌲🌱💐🌱☘️🌳
🌳 🌱🌲🌱🌱🌱🌲🌱🌹🌱🌱🌿🌱💐🌱🌱🌳🌾🌱🌱🌱🌱🌱🌺🌱🌱🌼🌱🌳
🌳 🌹🌱🌼🌱🌻 🌱🍀🌿🌱🌱🌱🌻 🌱🌲🌱💐🌱☘️🌹🌱🌱🍄🌷🌱🌳🌱🌱🌳
🌳 🌱🌱🌲🌼🌾🌱🌱🌲🌱🌱🌱🌱🌱🌺🌱🌱🌼🌱🌱🌲🌱🌱🌱🌲🌱🌹🌱🌳
🌳 🌱🌱🌱🌱🌱🍄🌱🌷🌱🌹🌱🌱🍄🌷🌱🌳🌱🌱🌹🌱🌼🌱🌻 🌱🍀🌿🌱🌳
🌳 🌱🌿🌱💐🌱🌱🌳🌾🌱🌱🌲🌱🌱🌱🌲🌱🌹🌱🌱🌱🌲🌼🌾🌱🌱🌲🌱🌳
🌳 🌱🌱🌻 🌱🌲🌱💐🌱☘️🌹🌱🌼🌱🌻 🌱🍀🌿🌱🌱🌱🌱🌱🌱🍄🌱🌷🌱🌳
🌳 🌱🌱🌱🌱🌺🌱🌱🌼🌱🌱🌱🌲🌼🌾🌱🌱🌲🌱🌱🌿🌱💐🌱🌱🌳🌾🌱🌳
🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳🌳