COMPLECS: COMPrehensive Learning for end-users to Effectively utilize CyberinfraStructure

Robert S. Sinkovits
sinkovit@sdsc.edu
University of California, San Diego
La Jolla, California, USA

Nicole Wolter
nickel@sdsc.edu
University of California, San Diego
La Jolla, California, USA

Martin C. Kandes
mkandes@sdsc.edu
University of California, San Diego
La Jolla, California, USA

Mary P. Thomas
mpthomas@ucsd.edu
University of California, San Diego
La Jolla, California, USA

Monica Sweet
msweet@ucsd.edu
University of California, San Diego
La Jolla, California, USA

ABSTRACT

The needs of cyberinfrastructure (CI) Users are different from those of CI Contributors. Typically, much of the training in advanced CI addresses developer topics such as MPI, OpenMP, CUDA and application profiling, leaving a gap in training for these users. To remedy this situation, we developed a new program: COMPrehensive Learning for end-users to Effectively utilize CyberinfraStructure (COMPLECS). COMPLECS focuses exclusively on helping CI Users acquire the skills and knowledge they need to efficiently accomplish their compute- and data-intensive research, covering topics such as parallel computing concepts, data management, batch computing, cybersecurity, HPC hardware overview, and high throughput computing.

CCS CONCEPTS

• **Social and professional topics** → **Computing education**.

KEYWORDS

Computing education, workforce development, advanced cyberinfrastructure

ACM Reference Format:
Robert S. Sinkovits, Nicole Wolter, Martin C. Kandes, Mary P. Thomas, and Monica Sweet. 2024. COMPLECS: COMPrehensive Learning for end-users to Effectively utilize CyberinfraStructure. In *Practice and Experience in Advanced Research Computing (PEARC '24), July 21–25, 2024, Providence, RI, USA*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3626203.3670553

1 INTRODUCTION

The explosive growth in data collections spanning all domains of science has expanded the use of cyberinfrastructure (CI) beyond the traditional base of HPC practitioners engaged in numerically intensive simulations. In addition, the development of new programming languages and usage modalities has made CI much more accessible. Equally important, the widespread availability of robust 3rd-party software allows a much broader community of researchers to benefit from CI without having to write their own code. For most researchers their intellectual contributions come from choosing the appropriate problems, algorithms and methods, prioritizing computations that yield the most insight and analyzing results rather than developing new software or methods.

Although some domain scientists still write their own software, we are seeing a growing divergence between CI contributors and CI Users (see [3] for NSF definitions). This is a welcome trend since it allows the two groups to specialize, resulting in more robust software and deeper scientific inquiry. The needs of CI Users are clearly different from those of CI Contributors. They may be required to write basic scripts to post-process data or make minor changes to Jupyter notebooks, but their efforts do not rise to the level of what would be considered software development. Nonetheless, CI Users still need a wide range of other skills to accomplish their research. Figure 1 shows how appropriately crafted training for these users can provide substantial benefits to themselves and others to reduce time to solution and minimize frustrations.

Unfortunately, while the community of CI Users has grown significantly in both size and breadth, much of the training still emphasize software development topics rather than the skills other than programming that are needed to effectively use advanced CI. Whereas CI Contributors often pick up these complementary skills along the way, either in formalized settings or from other developers, the CI Users can be left behind. To remedy this situation, we developed a new training program, COMPLECS, which focuses exclusively on helping CI Users acquire the transferable skills and knowledge they need to efficiently accomplish their compute- and data-intensive research. A key feature of our program is that the topics form a coherent package, broken down into manageable chunks spanning the foundational to specialized skills, that enable domain scientists to independently execute complete HPC and data-intensive workflows.

While all are welcome to participate, the target audience for COMPLECS consists of two CI User groups: (1) students and educators at minority serving (MSI) and primarily undergraduate (PUI) institutions; (2) and researchers from domains that haven't traditionally used advanced CI. These groups are less likely to have exposure to advanced CI or access to expertise on their campuses.

PEARC '24, July 21–25, 2024, Providence, RI, USA

ACM ISBN 979-8-4007-0419-2/24/07
https://doi.org/10.1145/3626203.3670553

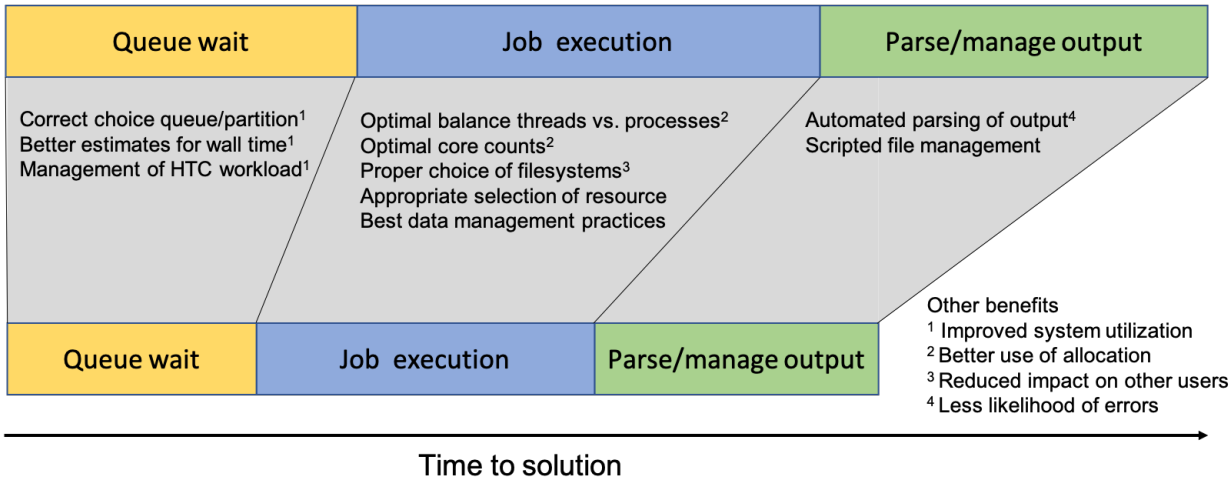

Figure 1: Simple workflow where user submits jobs, waits until jobs complete and parses/manages the output before proceeding to interpretation of results. Deploying skills and knowledge taught in COMPLECS, users can reduce time to solution in addition to realizing additional benefits for themselves and other users of the resource. Note that rectangle widths are for illustrative purposes only and speedups for each portion of workflow will depend on system utilization, queue policies and other factors.

Working directly with students will help them to advance their careers and, in the case of undergraduates, prepare them for graduate work. A focus on educators has a multiplier effect since they can integrate what they learn into their coursework and mentoring programs.

Participants have three learning options. First, we will host intensive multi-day workshops twice per year. Since these events have caps on attendance, priority is given to the groups mentioned above. Second, the topics are presented in a series of biweekly webinars or tutorials of 1-2 hours in duration. These are open to all, but we will still make a concerted effort to reach our target audiences. Finally, all training materials, including hands-on exercises [5], PDFs of slide decks [4] and recordings of webinars with curated transcripts, are made available for self-study.

Table 1: Organization of COMPLECS training into three tiers. Numbers in parentheses indicate section where topic is described.

Foundational knowledge	Parallel computing concepts (2.1) Intermediate Linux & scripting (2.2)
Essential skills	Hardware (2.3) Batch computing (2.4) Data management (2.5) Security (2.6) Interactive computing (2.7) Getting help (2.8)
Optional skills	Code migration (2.9) HTC (2.10) Linux text tools (2.11)

We try to minimize dependencies when possible so that participants who do not wish to follow the entire series can pick and choose just those topics that are of interest. The training begins with parallel computing concepts and intermediate Linux and shell scripting, providing the foundational knowledge necessary to master additional material. We move onto core topics that are important for everyone who works with advanced CI: data management, hardware essentials, batch job submission, cybersecurity, alternatives to the Linux command line and how to get help. We conclude with topics that may be of interest to only a subset of the participants, such as code migration and installation, high-throughput computing and Linux tools for file processing (Fig. 1). In all instances, the emphasis will be on skills that can be used on a broad range of resources and in a variety of situations.

2 TRAINING TOPICS

2.1 Parallel computing concepts

Parallel computing concepts are usually taught as part of the computer science curriculum or in specialized courses for application scientists who expect to develop their own parallel codes. As a consequence, CI Users generally miss out on the opportunity to learn about the distinctions between threads and processes or the inherent limits on the scalability of parallel applications. This is unfortunate since an understanding of these concepts is vital to the efficient use of advanced CI. Many 3rd-party applications are parallelized using a hybrid approach and getting the best performance depends on finding the optimal balance between threads and processes. Learning about Amdahl's law (strong scaling) [7] and Gustafson's law (weak scaling) [8] makes users aware of performance limitations and guides them to choose appropriate core, node, or GPU counts rather than simply throwing more hardware at the problem.

2.2 Intermediate Linux and shell scripting

Knowledge of Linux is indispensable for using advanced CI. While GUIs are becoming more prevalent, being able to work at the command line interface (CLI) provides the greatest power and flexibility. A task that would be tedious using the GUI, such as deleting or moving a subset of files with names matching a certain pattern, could be accomplished with a single Linux command. Beyond a familiarity with common utilities (ls, rm, cd, pwd, cat) and competence with at least one standard Linux editor (vi/vim, emacs, nano), CI Users should go deeper and have an understanding of symbolic links, aliases, environment variables, the Linux/UNIX Filesystem Hierarchy Standard, configuration files and basic scripting. These skills enable users to automate repetitive tasks and provide a foundation for the topics addressed in subsequent training. Since the definition of what constitutes *intermediate* Linux varies widely, we will ensure that our participants have a clear understanding of the prerequisites for our training.

2.3 Hardware essentials

Knowing the specifications of the hardware being used is helpful when trying to determine which factors affect application performance. With all else being equal, do faster clock speeds, advanced instruction sets, larger caches or different memory configurations translate to better performance? This can help guide resource selection, whether choosing among nationally allocated systems, purchasing dedicated hardware, or selecting cloud resources. Understanding hardware also opens up the use of monitoring tools that display CPU/GPU utilization and memory usage, enabling identification of common issues that affect performance such as multiple compute processes being mapped to the same core, underutilization of resources or memory leaks.

2.4 Batch computing

As computational and data requirements grow, researchers may need to transition from personal or lab resources to campus clusters or nationally allocated systems. Jobs on these shared resources are typically managed by batch submission system such as Slurm, PBS, LSF or SGE, where the user needs to specify the job duration, account information, hardware requirements and partition or queue. Our training covers the fundamentals of batch computing, such as fair share scheduling [10] and backfilling [6], before diving into the details of any particular workload manager, which will make it easier for users to adapt to new resources.

2.5 Data management

Proper data management is essential for the effective use of advanced CI. At minimum, users need to know how to efficiently move data and choose an appropriate file system. They should also understand the basics of file compression, archives, data sharing, file permissions, working with large numbers of small files, measuring data usage, identifying redundant files [2] and FAIR data principles [12]. We will address scenarios such as writing large volumes of data to home directories or many thousands of small files to Lustre file systems that can adversely impact other users. As a consequence, our training can have positive externalities that benefit everyone.

2.6 Best practices in securing data and research

Maintaining a secure CI environment is essential for ensuring the integrity of sensitive data sets and research results. Unfortunately, security practices are often sidelined or even circumvented by researchers in order to simplify or expedite a research workflow. Users sometimes avoid implementing safeguards, often based on the naive assumption that their research data/resources are not at risk. This module focuses on identifying risks, understanding the impacts of security breaches and learning the best practices for securing accounts, data collections and resources so as to avoid data loss, data corruption and unauthorized access. Note that most users have experienced a data loss incident for various reasons, many of which are not due to a malicious attack. Data protection skills learned here have applicability beyond the context of cybersecurity.

2.7 Interactive Computing

Interactive computing refers to working with software that accepts input from the user as it runs, either through a command line interface (CLI) or application GUI (e.g., Jupyter Notebooks, Matlab, RStudio). In an HPC environment, this often involves code development, real-time data exploration, advanced visualization or exploration of large data sets. Although interactive computing can hide some of the complexity of using advanced CI, users still need to be aware of what's going on under the hood. We will cover allocation of nodes for interactive jobs via the batch scheduler, monitoring hardware utilization, timing the execution of code blocks or notebook cells and understanding the implications of using web-based services versus X11 applications. We will also give a brief overview of Open OnDemand [9] given it's centrality to the ACCESS.

2.8 Getting help

CI Users inevitably encounter roadblocks that impede progress on their educational and research goals. Reducing the time and effort needed to address problems related to application performance, batch job submission or data management can minimize frustration and enable the users to become more productive. This module addresses two essential and related sets of skills that are frequently overlooked: (1) how to solve problems on your own and (2) knowing how to best work with the help desk or user support. One of our goals will be to educate CI Users on the best resources for solving their own problems. CI Users should also know how to collect the information that will be useful to the help desk, such as run times, data transfer rates, time and date when events occurred, batch job identifiers and hardware that was used. Finally, users should learn how to properly formulate a request for assistance since it will reduce the number of back-and-forth exchanges and minimize time to resolution.

2.9 Code migration, porting, and installation

The CI landscape is changing rapidly as new resources and applications become available and older ones are retired. As a result, CI Users often need to install or reinstall applications, sometimes reverting to earlier versions to maintain compatibility with previously generated results. Given changes in libraries, architecture instruction sets and software dependencies, building applications to

run with optimal performance in an HPC environment can be challenging. In this module, we address the topics that CI Users should know when doing their own installations. These range from understanding how to compile code, to working with package managers and containers.

2.10 High-throughput computing (HTC)

Many users have high-throughput computing (HTC) workloads characterized by large numbers of small jobs, frequently involving parameter sweeps where the same type of calculation is done repeatedly with different input values or data processing pipelines where an identical set of operations is applied to many files. Although these workloads can be handled using standard batch job submission, there are some pitfalls. Schedulers can be overwhelmed by the simultaneous submission of thousands of jobs, impacting all users. Operating policies that limit the number of running jobs, combined with heavy demand for resource, can lead to large delays relative to execution time. In addition, resource managers often impose a minimum charge of one "unit", typically a core-hour, and large numbers of very short jobs can quickly deplete an allocation. For users with these HTC workloads, there are alternatives that range from simply bundling together multiple calculations into a single job to using pilot job solutions or HTC frameworks [1] such as the Open Science Grid (OSG) [11]. In this module, we cover the basics of HTC, with pointers to more detailed materials.

2.11 Linux tools for text processing

Many computational and data processing workloads require preprocessing of input files to get the data into a format that is compatible with the user's application and/or post-processing of output files to extract key results. While these tasks could be done by hand, the process can be time-consuming, tedious and, worst of all, error prone. This is especially true if they need to be done many times, such as extracting a value from thousands of files generated during a set of parameter-sweep calculations. In this module we show how the utilities awk, sed, grep, sort, head, tail, cut, paste, cat and split can be used alone or in combination to easily automate repetitive tasks. Users who need additional details will be directed to primary sources.

3 DISCUSSION

COMPLECS is still in its early stages, with the first webinar delivered in January 2024. The emphasis to date has been on the development of new training materials and preparations for the upcoming SDSC HPC and Data Science Summer Institute, where COMPLECS topics constitute the first two days. Initial impressions are that COMPLECS has been successful, with 425 attendees at 7 webinars and 73 applications (so far) to the Summer Institute. While response rates to the post-webinar surveys have been relatively low, feedback so far has been positive. We are discussing strategies to improve the response rates, such as offering incentives using non-federal funds, minimizing survey length to reduce survey fatigue, and providing multiple opportunities and reminders to respond.

After each webinar topic has been presented at least once, we will use the results of the surveys together with the attendance numbers to make course corrections. Popular topics may be repeated more often or expanded into multiple webinars that go into greater depth, while less popular topics might be condensed or only made available online for self-directed training. Regardless of the final decision on each topic, we will use the feedback we receive to improve the quality of all our materials. Longer-term outcomes will be measured at the end of each grant year. Open-ended, longer-term outcome-related questions that we wish to answer include:

- Have you used what you learned from our training events to facilitate your research?
- Have you shared what you learned with colleagues or students at or beyond your institution, either informally (conversations, meetings, lab presentations) or formally (classes taught, seminars, workshops, direct tutoring).
- How is the information you learned and/or skills you've gained being used in practice in your academic life?
- If you changed your career status in the past year, did our training play a role in helping you secure your new position? If not, please rate and define your views of the perceived future utility of this program
- How do you think participation in COMPLECS will impact your future academic self and surroundings?

ACKNOWLEDGMENTS

This work was supported by NSF Award #2320934. The authors thank Susan Rathbun, Cindy Wong and Chante Powell for their excellent support of COMPLECS.

REFERENCES

[1] [n. d.]. HTCondor. https://research.cs.wisc.edu/htcondor
[2] 2021. The 5 Best Tools to Find and Remove Duplicate Files in Linux. https://www.makeuseof.com/best-tools-find-and-remove-duplicate-files-linux/
[3] 2022. NSF 23-520: Training-based Workforce Development for Advanced Cyberinfrastructure (CyberTraining). https://new.nsf.gov/funding/opportunities/training-based-workforce-development-advanced/nsf23-520/solicitation
[4] 2023. COMPLECS presentation PDFs. https://bit.ly/COMPLECS
[5] 2023. SDSC-COMPLECS GitHub. https://github.com/orgs/sdsc-complecs/repositories
[6] 2023. Slurm Scheduling Configuration Guide. https://slurm.schedmd.com/sched_config.html
[7] Gene M Amdahl. 1967. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*. 483–485.
[8] John L Gustafson. 1988. Reevaluating Amdahl's law. *Commun. ACM* 31, 5 (1988), 532–533.
[9] Dave Hudak, Doug Johnson, Alan Chalker, Jeremy Nicklas, Eric Franz, Trey Dockendorf, and Brian L McMichael. 2018. Open OnDemand: a web-based client portal for HPC centers. *Journal of Open Source Software* 3, 25 (2018), 622.
[10] Judy Kay and Piers Lauder. 1988. A fair share scheduler. *Commun. ACM* 31, 1 (1988), 44–55.
[11] Ruth Pordes, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, and Frank Würthwein. 2007. The open science grid. In *Journal of Physics: Conference Series*, Vol. 78. IOP Publishing.
[12] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, and Philip E Bourne. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data* 3, 1 (2016), 1–9.