Center for
HIGH PERFORMANCE COMPUTING
THE UNIVERSITY OF UTAH

Newsletter

Summer 2019

Modernizing the Field of Fracture Mechanics through High-Fidelity Modeling, Data-Driven Science, and Machine Learning

Research Highlight · Ashley Spear, Department of Mechanical Engineering

Cracks are mysterious and have haunted materials scientists and design engineers for at least a century. At the smallest length scales, cracks are born when atomic bonds break to create new surface area that previously did not exist. Of course, we generally do not notice cracks until they are much, much larger than the atomic scale. Often, cracks are regarded as being detrimental to the integrity of structures (e.g., aircrafts, biomedical implants, civil infrastructure); on the other hand, there are many scenarios where the ultimate goal is to cause cracking (e.g., opening household items like soda cans, wood splitting, hydraulic fracturing). For critical, high-stakes applications, being able to predict when and where cracks will form and how quickly they will grow is necessary to estimating the safety, reliability, and cost-effectiveness of both new and existing technologies. Making these kinds of predictions is central to the field of *fracture mechanics*.

Like most areas of science and technology, the field of fracture mechanics is currently undergoing a transition to something that Dr. Tony Hey, former VP of Microsoft Research, referred to as the *fourth paradigm* [1]. The so-called fourth paradigm in fields of science and technology is characterized by data-intensive scientific discovery. The first three paradigms that historically precede the fourth paradigm are, in chronological order:

(1) empirical science,
(2) theoretical-model science, and
(3) computational science.

The field of fracture mechanics—which began around the 1920s with foundational work by Griffith—has been dominated primarily by the first two paradigms, through which empirical and theoretical fracture models have enjoyed success in predicting behavior of long, idealized cracks. However, for small, microscopic cracks, techniques in the first two paradigms tend to break down, and we must, therefore, turn to computational models and data-driven science to help in understanding and predicting crack behavior at that scale. Ironically, the formation and growth of very small (microscopic) cracks tend to consume the greatest portion of total "life" of most structures; yet they are also the most difficult to predict due to complex governing physics acting at microscopic scales and to inherent variability in the material microstructure.

With access to computational resources through the CHPC, the Multiscale Mechanics & Materials (MMM) Lab (*https://mmm.mech.utah.edu/*) is working to modernize the field of fracture mechanics to move beyond the first two paradigms described above. Through the development and use of sophisticated, high-fidelity numerical modeling, data-driven science, and machine learning (ML), we have been working to address some of the most fundamental questions concerning the origins and propagation of small cracks in engineering materials, some of which are highlighted below.

How can we predict crack interaction with material microstructure and with other cracks?

One of the challenges of predicting the life of a structural component is being able to model and predict how microscopic cracks might interact with one another and with their local microstructural environment. It is widely known in the fracture community that at small scales in heterogeneous materials, cracks tend to change their behavior as they approach the boundary of a new material region. For example, in polycrystalline metals, many researchers have observed phenomena like crack deflection or deceleration near grain boundaries. Despite such observations, being able to model this kind of crack behavior is extremely difficult for a number of reasons, including challenges associated with computationally representing a 3D discontinuity evolving throughout a geometrically complex multi-material domain. To address this challenge, the MMM Lab has implemented a robust meshing framework [2] capable of modeling the complex crack-growth phenomena that researchers have long observed at the microstructural length scale and in heterogeneous materials. The framework provides a powerful tool that could enable accurate predictions of structural life, materials-design optimization based on microstructural influence on crack path, and discovery of the fundamental mechanisms driving the growth of small cracks.

Financial support provided by the U.S. Air Force Office of Scientific Research (FA9550–15–1–0172) and by the AFRL Summer Faculty Fellowship Program.

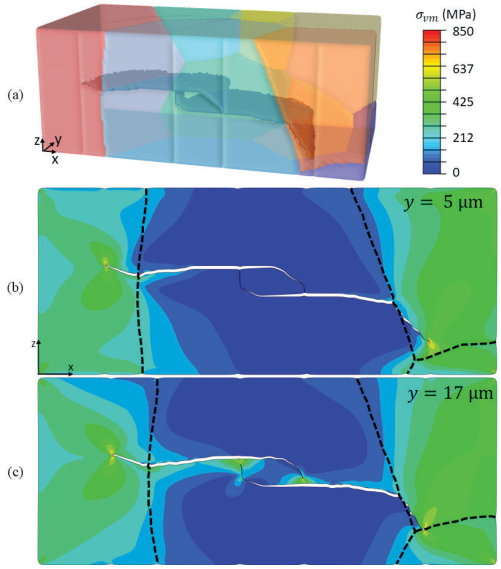


Figure 1. *High-fidelity simulation performed on notch-peak to predict cracks interacting with grain boundaries and coalescing [2].*

How much volume is required to capture the important "neighborhood" of a crack?

As illustrated above, it is important in making accurate predictions of structural life to capture the microstructural features—experimentally or computationally—that are responsible for influencing the evolution of small cracks. This leads to the important yet unanswered question: *How much volume is required to capture the dependence of a crack's behavior on the surrounding microstructure?* Including too much volume could result in unnecessarily large (and potentially intractable) experimental or numerical expense, and using too small of a volume could lead to inaccurate analysis of crack behavior. Thus, the objective of this research is to extend the concept of the representative volume element (RVE) by proposing RVE$_{MSC}$, the smallest heterogeneous volume containing a microstructurally small crack (MSC) such that the fields along the crack front are insensitive to volume size and boundary effects. To determine RVE$_{MSC}$, the MMM Lab has implemented a finite-element-based framework that accounts for the effect of microstructural variation and boundary conditions on the size of the required volume for accurate analysis of cracks [3]. So far, the team has run over 7,000 numerical simulations to establish a closed-form solution that conservatively approximates the size of volume required to guarantee convergence of crack-front parameters for a given crack size in a heterogeneous, linear-elastic material. The team subsequently conducted data-driven correlation analyses to shed light on the characteristics of the microstructural neighborhood of a crack that tend to increase the size of the required volume. Currently, we are leveraging the numerical data set to train an ML algorithm to predict efficiently the size of RVE$_{MSC}$ for more complex, elasto-plastic materials.

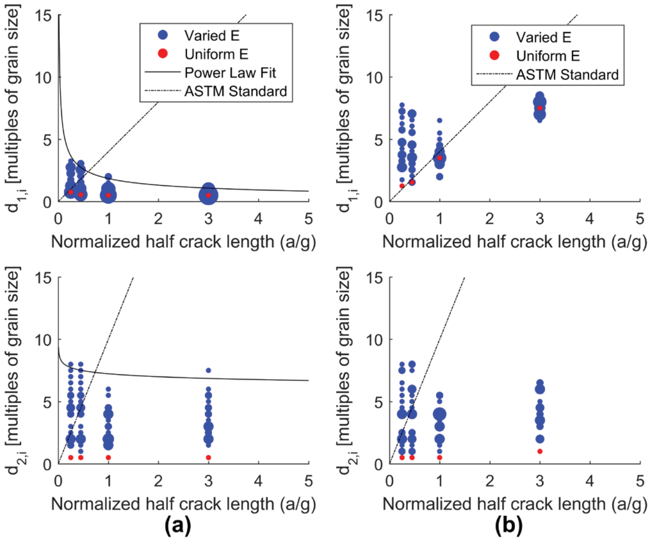


Figure 2. *Two parameters (d1 and d2) representing the volume requirements for microstructure-sensitive cracks based on over 7,000 numerical simulations performed on kingspeak [3].*

Financial support provided by the U.S. Air Force Office of Scientific Research (FA9550–15–1–0172) and by the National Defense Science and Engineering Graduate (NDSEG) Fellowship awarded to Karen DeMille.

Is it possible to predict the path of a crack in 3D using only basic information about its microstructural neighborhood?

The MMM Lab recently implemented a deep-learning algorithm to test the hypothesis that information from an uncracked microstructure encodes the path that a crack will eventually take in 3D. To test the hypothesis, the team leveraged a sophisticated data set that combines synchrotron measurements and high-fidelity simulation data. Specifically, the team used 3D measurements of crystal structure taken for a polycrystalline sample of Al-Mg-Si containing a measured crack surface [4]. The measurements were then digitally reconstructed [5] to create a high-fidelity representation of the volume *without* a crack. Crystal-plasticity simulations were performed to predict how stresses and strains evolved within the experimentally measured microstructure. Finally, data from both the experimental measurements and the simulations were integrated to create a massive training data set by discretizing the microstructural volume into millions of individual data points. The data set was used to train a convolutional neural network (CNN) to predict the 3D crack path, resulting in a complete 3D crack surface over the entire prediction domain [6,7]. The performance of the CNN was compared with other types of ML models and for different types of training data. In general, the ML models are able to predict the crack surface more accurately than simply taking a naïve approach to extending the crack forward. The most significant—and rather surprising—finding was that the CNN model performs well when given information about only the crystal orientations throughout the polycrystal. This result could have important implications for making rapid predictions of crack paths in different 3D microstructural configurations using just a basic representation of the material, akin to using just RGB values in an image. The project provides an early example of the role of data science and machine learning in advancing the field of fracture mechanics.

Financial support provided by the U.S. Air Force Office of Scientific Research (FA9550–15–1–0172).

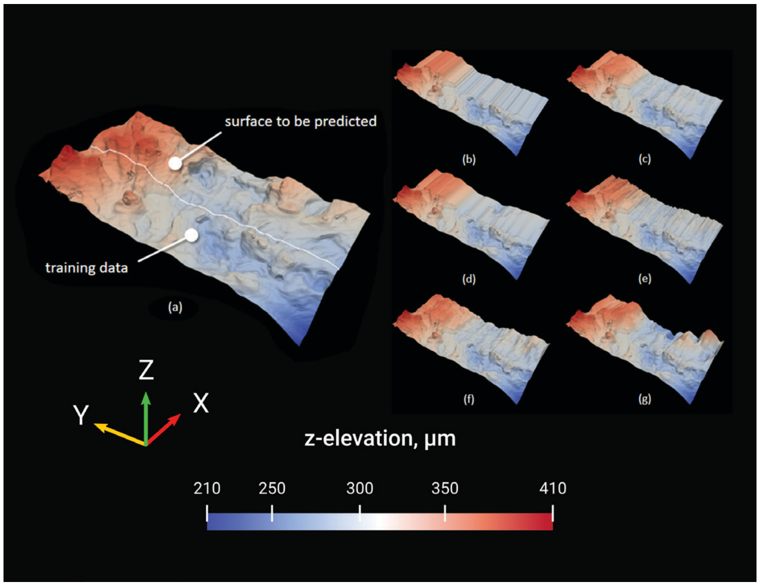


Figure 3. *Crack-surface predictions made by different ML models, compared to an actual crack surface [7]. Simulations were performed on kingspeak.*

References

[1] T. Hey, S. Tansley, and K.M. Tolle. *The Fourth Paradigm: Data-Intensive Scientific Discovery*, vol. 1 (Redmond: Microsoft Research, 2009).

[2] B.R. Phung and A.D. Spear. *Engineering Fracture Mechanics*. 2019. 209(2019):404-422.

[3] K.J. DeMille and A.D. Spear. Submitted (in review).

[4] A.D. Spear, S.F. Li, J.F. Lind, R.M. Suter, and A.R. Ingraffea. *Acta Materialia*. 2014. 76(2014):413-424.

[5] A.D. Spear, J.D. Hochhalter, A.R. Cerrone, S.F. Li, J.F. Lind, R.M. Suter, and A.R. Ingraffea. *Fatigue & Fracture of Engineering Materials & Structures*. 2016. 39(6):737-751.

[6] K.D. Pierson, J.D. Hochhalter, and A.D. Spear. *JOM*. 2018. 70(7):1159-1167.

[7] K.D. Pierson, A. Rahman, and A.D. Spear. *JOM*. 2019. In press.

New Staff at CHPC

CHPC hired two new staff members this summer. Please join us in welcoming them!

Paul Fischer

Paul Fischer was initially hired by CHPC as a student employee in early 2015 and worked part-time on a variety of system provisioning, storage, and usage monitoring projects while pursuing a mathematics degree through the Honors College. This spring, he graduated and accepted a full-time system administrator position at CHPC. His primary focus right now is deploying the Open XDMoD usage and performance analytics package, which will allow CHPC staff and users to explore and visualize a huge amount of data relating to their CHPC jobs through a web portal. He also assists users with technical questions, particularly those relating to getting Python, shell, C, and other codes running on CHPC resources.

David Nordello

David Nordello was previously an IT Support Engineer at RIVA Solutions, Inc. (a contractor for the National Weather Service) responsible for providing IT support for the Fire Weather Program. David brings with him over 25 years of experience in Linux, Windows, and computer security. In his new role at CHPC, he will provide support for the protected environment.

Open OnDemand Web Portal Update

Technology · Martin Cuma

CHPC has recently updated the Open OnDemand (OOD) web portal in an effort to enhance its usability. Open OnDemand is an NSF-funded open-source high performance computing (HPC) portal, which provides features such as file management, terminal shell access, and cluster job management and monitoring. However, from our standpoint, the most unique feature is that users can run graphical desktops and applications as an interactive cluster job, all accessible from a web browser.

We started offering OnDemand two years ago, but due to the high load of our HPC cluster queues, there was often not a node available to run an interactive job. This problem led us to recently introduce a new partition on the notchpeak cluster, named ***notchpeak-shared-short***, with limits that better allow for interactive use. This partition consists of two nodes, each with 64 physical cores and 512 GB of memory. The limits for jobs running in this partition are a maximum wall time of 8 hours, maximum task count of 32, and maximum memory of 128 GB per job, with an additional limit of 32 total maximum consecutive tasks per user. It should be noted that this new ***notchpeak-shared-short*** partition is not dedicated for use by jobs submitted via Open OnDemand; it can also be used for debugging and testing purposes through the standard cluster command line access, as well as for short jobs that fit within the limits. However, the utility of this partition really shines in connection with Open OnDemand, as it allows for interactive, workday- long access to a dedicated compute resource. This is even more important with the interactive node load monitoring that we have started in early April, which limits the usage of the login and frisco nodes.

Interactive Jobs with Open OnDemand

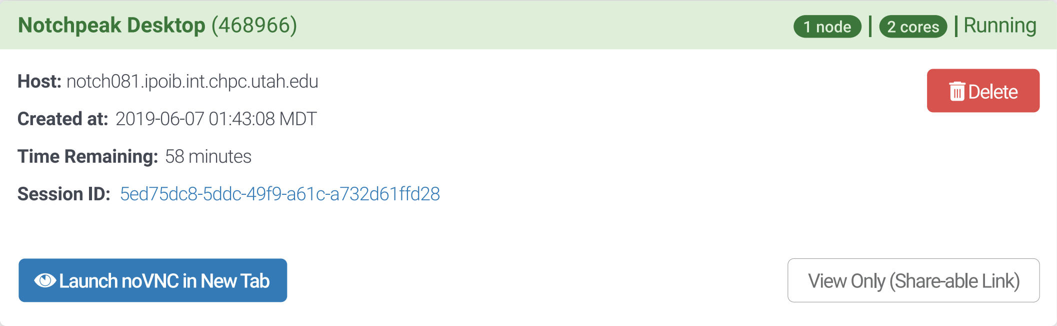
To start an interactive job using the OnDemand web interface, first open *https://ondemand.chpc.utah.edu* in your web browser and authenticate using CHPC credentials. Then navigate the menu *Interactive Apps – Notchpeak Desktop*. Choose the required task, node count, wall time, account, and partition. Then push the *Launch* button to submit the job. A page appears notifying that the job has been submitted, followed by another page when the job starts (Figure 1). Pushing *Launch noVNC in a New Tab* will start a remote desktop session in a new browser tab. The session runs on the compute node assigned to the job. Note that you can also share the web link to this desktop session with others in the read-only mode, available by pushing the *View Only (Share-able Link)* button.

Apart from the interactive desktop sessions, OnDemand also allows admins and users to create their own *Apps*, which are custom ways to launch applications inside of the interactive job. So far we have pre-installed four different applications: *ANSYS Workbench*, *MATLAB*, *Jupyter notebooks*, and *RStudio server*. All four of these applications can take advantage of the dedicated multi-core compute node-based resource that the ***notchpeak-shared-short*** partition provides. Using OnDemand and the compute nodes for these types of interactive workload is a better approach than running them on the cluster interactive nodes or the frisco nodes, where the job is limited by the policy constraints enforced by the Arbiter script. Note that for jobs that do not fit into the constraints of the ***notchpeak-shared-short*** partition, users can still use OnDemand to submit to the regular partitions on any of the clusters.

OnDemand application development, documented at *https://osc.github.io/ood-documentation/master/app-development.html*, can be done either by admins or by users. If you would like us to either add support for a new application or assist you in doing this yourself, please contact us at *helpdesk@chpc.utah.edu*.

Jupyter Notebook on OnDemand

As mentioned above, OnDemand gives one a dedicated interactive compute resource, which is not available on other CHPC-provided Jupyter options. The Jupyter Notebook that CHPC's OnDemand installation uses runs the `python/3.5.2` module and also includes `R/3.4.4` and `matlab/R2018a`. Other plugins can be installed by request to *helpdesk@chpc.utah.edu*. To start a new Jupyter Notebook job using OnDemand, navigate the menu to *Interactive Apps – Jupyter Notebook on Notchpeak*, select


Notchpeak Desktop (468966)

1 node | 2 cores | Running

Host: notch081.ipoib.int.chpc.utah.edu

Created at: 2019-06-07 01:43:08 MDT

Time Remaining: 58 minutes

Session ID: 5ed75dc8-5ddc-49f9-a61c-a732d61ffd28

Figure 1. *Interactive desktop job window. The window provides information about the job—the number of nodes and cores— as well as buttons to view and share the session.*

the appropriate job parameters, such as the ***notchpeak-shared-short*** partition, and push the *Launch* button. The job queue window appears again, followed by the running job window, which indicates that the job has started. Push *Connect to Jupyter* to open a Jupyter instance in a new browser tab. One can run multiple notebooks within one job, each of which will open in a new browser tab.

To use Python packages that are not installed in the CHPC's `python/3.5.2` module, install the packages in your home directory, and then set `PYTHONPATH` environment variable in *~/.custom.csh* for tcsh shell or *~/custom.sh* for bash shell (and **not** in *~/.tcshrc* or *~/.bashrc*, which are maintained by CHPC and should not be edited) pointing to this directory.

OnDemand Future Outlook

We hope that the improved OnDemand job interactivity will attract new users to this promising new way to access HPC resources. More details on the OOD usage are at CHPC's documentation page, *https://www.chpc.utah.edu/documentation/software/ondemand.php*. As a project that is being actively developed, OOD has an extensive roadmap with plans for new features such as job summary and statistics through XDMoD (*https://open.xdmod.org*), interactive jobs and apps on HPC resources other than Linux clusters (interactive nodes and cloud resources, for example), science gateways, and improvements to the interface. We also appreciate any user experiences and suggestions that we can pass on to the developers of OnDemand.

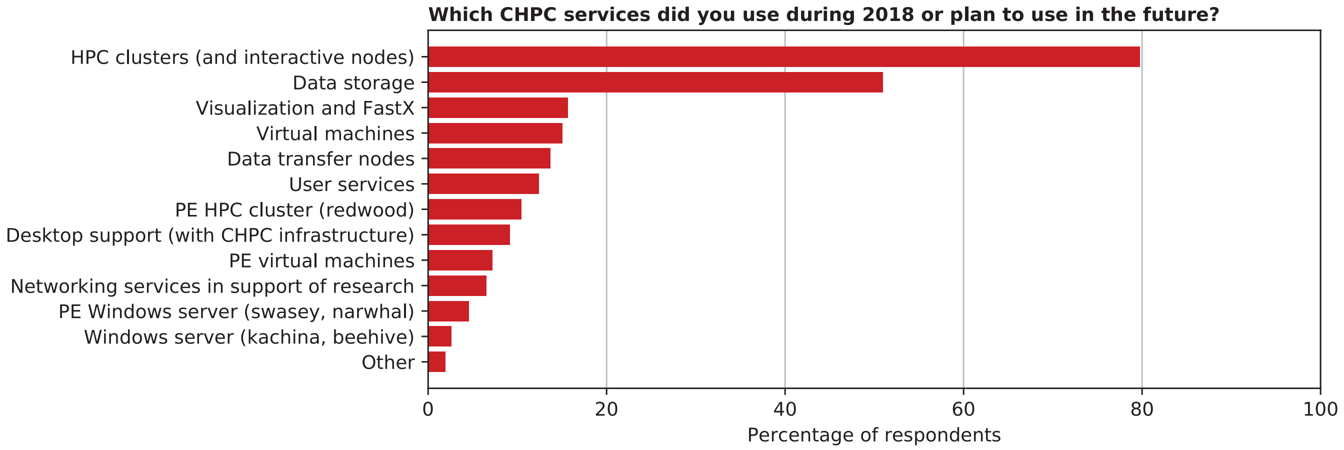
Annual PI Survey Results

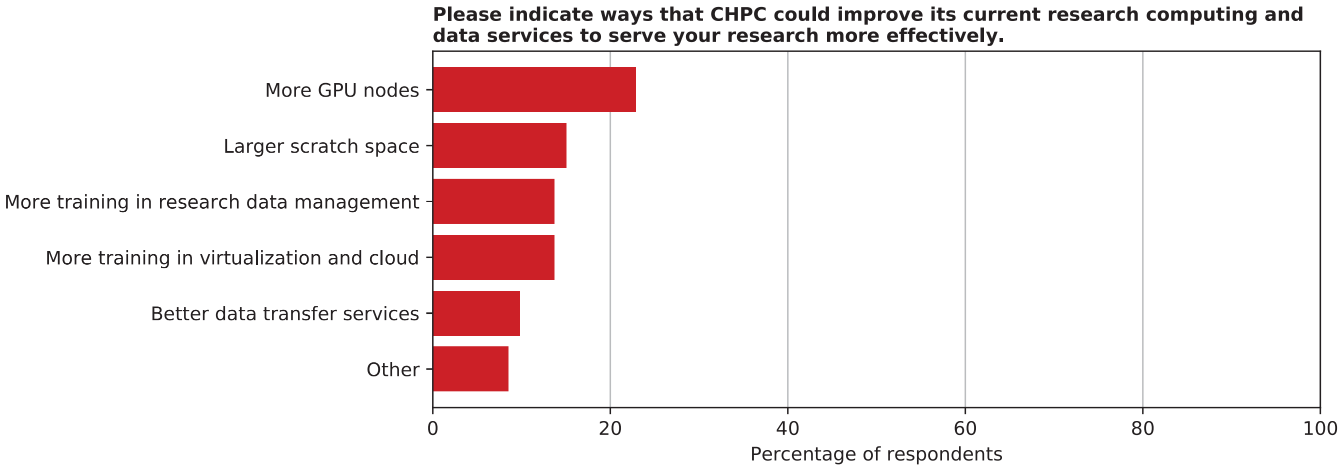
News and Updates · Anita Orendt

For the last four years, CHPC has sent a survey about the use of CHPC resources to all of our PIs at the start of the year. Each year, the response rate has increased, with 153 (26%) of the PIs completing the survey in 2019. The survey covers a number of topics; two are highlighted below.

In response to the question on ways CHPC can improve, we have recently added additional GPU nodes to notchpeak and provisioned a new scratch space. We provide more details about these in the Resource Updates sections. We are also working on expanding and improving training videos as well as the CHPC Presentation Series offerings.

If you have any additional suggestions about other ways CHPC might better serve your research computing needs, please reach out to us at *helpdesk@chpc.utah.edu* to set up a meeting to discuss!





Node Sharing Enabled on CHPC General Resources

News and Updates · Anita Orendt

Effective July 1, the start of the Summer allocation quarter, node sharing has been enabled on all CHPC compute nodes. The motivation for adding the node sharing option is that the newer nodes have more cores – the newest notchpeak nodes have 40 physical compute cores per node – and there are some computational workloads that do not efficiently make use of this number of cores. Node sharing allows for multiple jobs to use a single node, with each being submitted as a separate batch submission, each using only a portion of the node. It should be noted that we have had node sharing in place on owner nodes as well as on the GPU nodes for a couple of years.

With this addition, users will see that for the general nodes on a given cluster there will be four defined partitions instead of the current two: ***cluster-shared*** and ***cluster-shared-freecycle*** will be added to the existing ***cluster**** and ***cluster-freecycle***. In a similar fashion, along with ***notchpeak-guest***, there will be a ***notchpeak-shared-guest***. For example, looking at notchpeak, you will see the following:

PARTITION	NODES	NODELIST
notchpeak*	28	notch[005-018,035-045,068,096-097]
notchpeak-shared	28	notch[005-018,035-045,068,096-097]
notchpeak-freecycle	35	notch[001-018,035-045,068,086-088,096-097]
notchpeak-shared-freecycle	35	notch[001-018,035-045,068,086-088,096-097]
notchpeak-guest	54	notch[019-034,046-054,056-059,061-067,069-080,090-095]
notchpeak-shared-guest	54	notch[019-034,046-054,056-059,061-067,069-080,090-095]

When submitting a job that will use the entire node to the general nodes, you do not need to make any changes to the way you submit a job now: use your account and set the partition to the ***cluster*** or ***cluster-freecycle***, depending on whether or not your group has allocation. Also, for using the entire nodes in the guest mode, nothing changes. Existing Slurm scripts should not be affected by this change.

When you want to use only a portion of the node, however, you will change the partition to ***cluster-shared***, ***cluster-shared-freecycle***, or ***cluster-shared-guest*** and you **must** also specify the number of cores and the amount of memory that the job requires in the batch script:

```
#SBATCH --partition=kingspeak-shared
#SBATCH --ntasks=2
#SBATCH --mem=32G
```

Node sharing automatically enables CPU core affinity, allowing the job to run only on as many cores as there are requested tasks.

Note that if no memory directive is specified, the default setting of 2 GB per core will be used. Also note that for Gaussian, ***cpus-per-task*** must be used instead of ***ntasks***, as using ntasks will launch that number of copies of the same Gaussian calculation within the job.

More information about node sharing can be found on the CHPC documentation at *https://www.chpc.utah.edu/documentation/software/node-sharing.php*.

Potential Performance Implications

Despite the task affinity, the performance of any job run in a shared manner can be impacted by other jobs run on the same node due to shared infrastructure being used, in particular the I/O to storage and the shared communication paths between the memory and CPUs. Therefore, if you are doing benchmarking using only a portion of a node, you should not use node sharing but should instead request the entire node.

Implication on Amount of Allocation Used

The allocation usage of a shared job is based on the percentage of the cores and the memory used—whichever is higher. The highest fractional utilization of the two will be used for accounting calculations. This can lead to situations where the total allocation used by two separate jobs can exceed the expected maximum core hour usage.

As an example, consider a node with 24 cores and 128 GB. Without node sharing, during any one-hour period, the maximum usage would be 24 core hours. However, with node sharing, one could envision that a serial job which requires a lot of memory may request 1 core and 96 GB of memory. For an hour-long run, this would use 18 core hours, since 96 GB is three-fourths of the memory. This leaves 23 cores and 32 GB of memory that can be used by other jobs. If a job uses 20 cores and 32 GB of memory, it will be charged 20 core hours. No additional jobs can be added, as these two are using all of the memory on the node. If these two jobs run simultaneously for one hour, the total allocation used would be 38 core hours.

Alternatives to Node Sharing to Run Multiple Jobs on a Single Node

There are times when a user may need to run a large number of similar independent calculations. Due to logistics in setting up these calculations, it is often easier to run these as a single job instead of running each single calculation as a separate job. A number of strategies that can accomplish this are discussed in detail at *https://www.chpc.utah.edu/documentation/software/serial-jobs.php*.

If you have any questions about the best approaches for your specific use case, please contact us at *helpdesk@chpc.utah.edu* and we would be happy to work with you.

Additional Note: Resource Limits

With this change, we have also renamed the current ***notchpeak-shared*** partition to ***notchpeak-shared-short*** (which is mentioned in the Open OnDemand article in this newsletter). This partition contains two nodes, notch[081-082], each with 64 physical cores and 512 GB of memory. These nodes can only be used in the shared mode. This new name reflects that these two nodes can be used for short jobs, both in the amount of resources (memory, CPU) and the wall time.

The current limits are:

- A wall time of 8 hours
- A maximum of 10 jobs in the queue per user
- A maximum of 2 running jobs per user
- A maximum of 32 cores per user
- A maximum memory usage of 128 GB per user

What's New: Resource Updates

News and Updates · Anita Orendt and Brett Milash

New Scratch Space

In case you missed the earlier announcement, CHPC has added a new scratch space, */scratch/general/nfs1*. With this addition, there are three scratch spaces that are available on all of the CHPC clusters, including all interactive nodes, compute nodes, and frisco nodes. The scratch spaces are:

- */scratch/general/lustre*: a 700 TB Lustre system
- */scratch/kingspeak/serial*: a 175 TB NFS system
- */scratch/general/nfs1*: a 595 TB NFS system

Please remember that these scratch spaces are a shared resource open to all users. Any files not accessed in 60 days will be automatically removed.

New General Notchpeak Compute Nodes

CHPC has recently added three additional general GPU nodes to the notchpeak cluster. They are notch[086-088]. Each of these nodes has 40 physical CPU cores, 192 GB of memory, and 4 RTX 2080 Ti GPUs. To get more information about these new nodes, along with other GPU nodes, see the documentation at *https://www.chpc.utah.edu/documentation/guides/gpus-accelerators.php*.

Along with these GPU nodes, we also recently added two new general CPU nodes, with 40 physical cores and 192 GB of memory each. These nodes are notch096 and notch097. There are also an additional five general notchpeak nodes with these same specifications on order that will be added to the cluster later this summer.

Windows Server in General Environment

CHPC has recently replaced the Windows server hardware in the general environment. The *kachina* server has been replaced with *beehive*, which has 48 cores, 512 GB of memory, and about 2 TB of disk space (and can mount user and group spaces if needed). See *https://www.chpc.utah.edu/documentation/guides/beehive.php* for details.

VM Policy

CHPC has recently refreshed the hardware of the Virtual Machine (VM) Farm in the general environment. This follows the refresh of the VM hardware in the protected environment last year. Along with the hardware refresh, CHPC has implemented two changes:

- VM size specifications have been standardized
- VMs are no longer available free of charge

All users or groups with existing VMs have been informed of this new policy.

Information on the block sizing as well as the price of VMs can be found on the CHPC documentation at *https://www.chpc.utah.edu/resources/virtualmachines.php*.

BLAST Databases

CHPC maintains a number of sequence databases for searching with NCBI's BLAST software. We are moving these databases into several locations in the */scratch* file system to improve performance and reduce the impact of the use of BLAST on the */uufs* file system. The `blast/2.3.0` and `blast/2.7.1` modules will be modified to direct your BLAST processes to one of these locations, depending on which cluster you are using. The change of location should be transparent to any scripts using BLAST as long as you rely on the `BLASTDB` environment variable as set by the `blast` modules.

The `blast` modules will be updated to implement these changes on Monday, July 1. Access to the old database location, */uufs/chpc.utah.edu/sys/installdir/blastdb*, will be discontinued on Wednesday, July 3.

If you manually set `BLASTDB` to access these databases, please update your scripts by July 3. We suggest the following settings:

CLUSTER	DATABASE FILE LOCATION
redwood	/scratch/mammoth/serial/app-repo/blastdb/DbFiles
kingspeak	/scratch/general/lustre/app-repo/blastdb/DbFiles
notchpeak	/scratch/general/lustre/app-repo/blastdb/DbFiles
ember	/scratch/general/nfs1/app-repo/blastdb/DbFiles
lonepeak	/scratch/general/nfs1/app-repo/blastdb/DbFiles

We will continue to update these databases on a monthly basis, typically on the 5th of the month.

If you have any questions or concerns about the recent changes, please contact us at *helpdesk@chpc.utah.edu*.

Center for
HIGH PERFORMANCE COMPUTING
THE UNIVERSITY OF UTAH

The University of Utah
University Information Technology
Center for High Performance Computing
155 South 1452 East, Room 405
SALT LAKE CITY, UT 84112–0190

Thank you for using CHPC resources!

Welcome to CHPC News!

If you would like to be added to our mailing list, please provide the following information via the contact methods described below.

Name:
Phone:
Email:

Department
or Affiliation:

Address:
(campus
or U.S. mail)

Please acknowledge the use of CHPC resources!

If you use CHPC computer time or staff resources, we request that you acknowledge this in technical reports, publications, and dissertations. An example of what we ask you to include in your acknowledgments is:

"A grant of computer time from the Center for High Performance Computing is gratefully acknowledged."

If you make use of the CHPC Protected Environment, please also acknowledge the NIH shared instrumentation grant:

"The computational resources used were partially funded by the NIH Shared Instrumentation Grant 1S10OD021644-01A1."

Please submit copies or citations of dissertations, reports, pre-prints, and reprints in which CHPC is acknowledged in one of the following ways:

Electronic responses

By email:	*helpdesk@chpc.utah.edu*
By fax:	(801) 585–5366

Paper responses

By U.S. mail:	155 South 1452 East, Rm 405 Salt Lake City, UT 84112–0190
By campus mail:	INSCC 405