

The background features a complex network of blue lines and arrows. Some lines are solid and straight, while others are dashed and curved, suggesting a path or a network. The lines intersect and branch out, creating a sense of movement and connectivity. The overall aesthetic is technical and modern.

CCR ON DEMAND: UB'S ONE-STOP SHOP FOR HIGH PERFORMANCE COMPUTING

L. Shawn Matott, Ph.D.

Computational Scientist
UB Center for Computational Research

OVERVIEW

Workshop Objectives

- To learn some basic concepts of high performance computing (HPC)
- To learn about the UB Center for Computational Research (CCR)
- To learn how to use the CCR On Demand portal



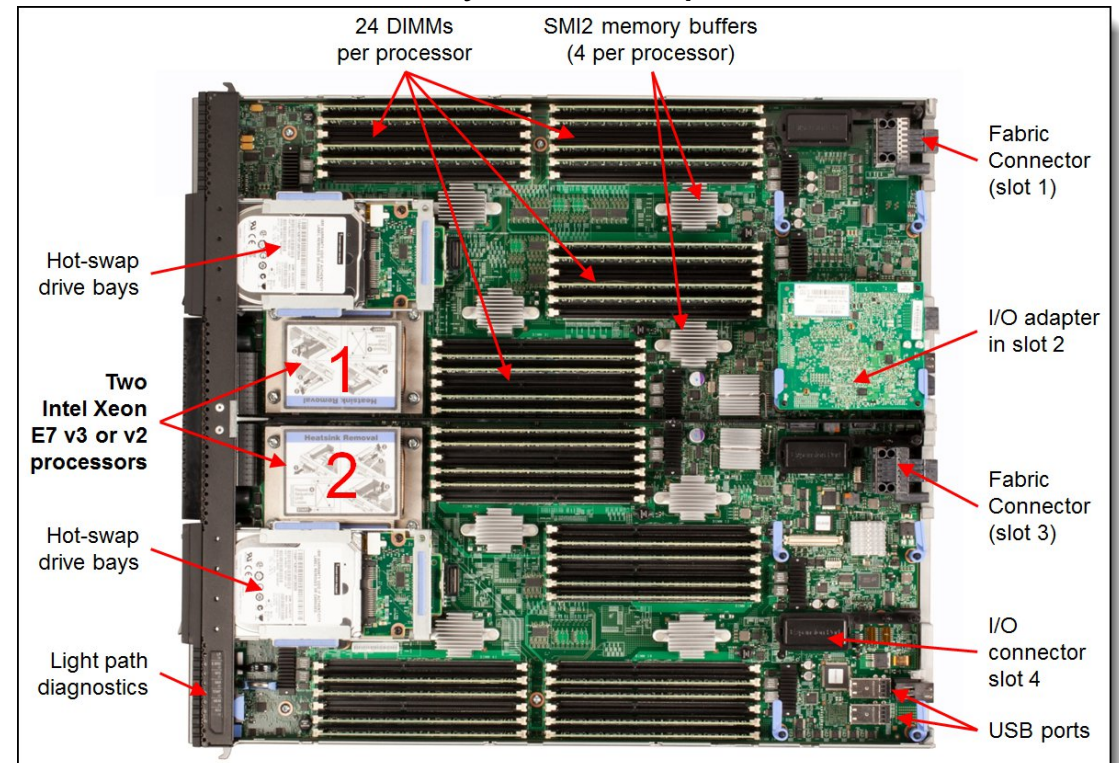
source: UB Photo Database

BASIC HPC CONCEPTS

Some Basic HPC Terminology

- **Compute Node** – a physical self-contained compute unit consisting of an operating system (OS), processors, memory, storage and networking interfaces. Also known as a “blade”. Compute nodes can contain multiple processors.
- **Processor** – a computing chip that can contain multiple processing cores.
- **Core** – a single processing unit within a processor. These are what ultimately perform computations.
 - Cores on the same compute node must share the node resources (e.g. memory, storage, and networking).
 - Within software, cores are often referred to as cpus or processors

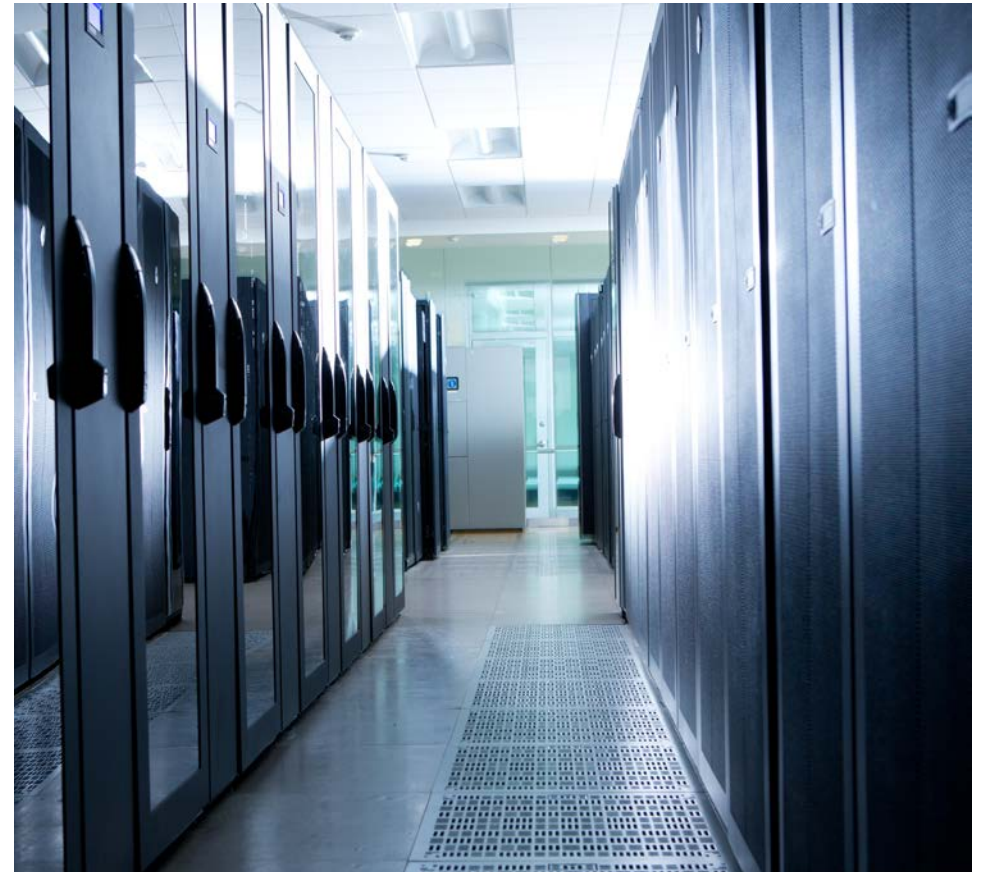
Anatomy of a compute node



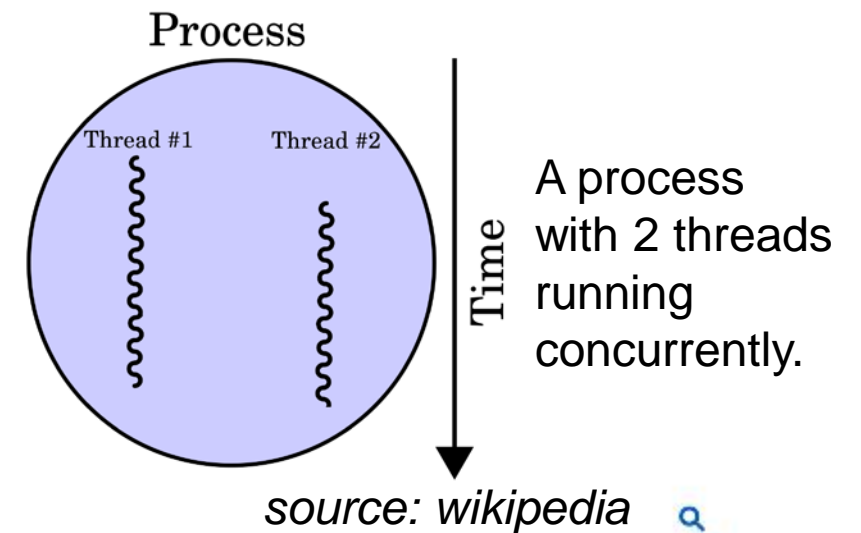
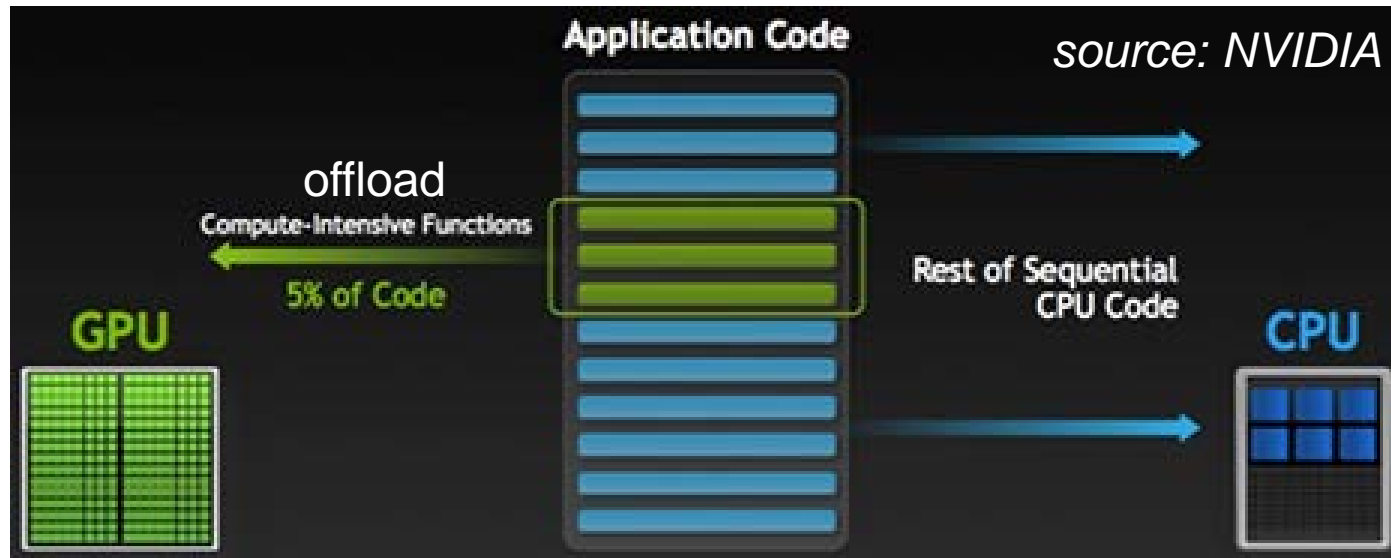
Source: lenovopress.com

- *Node Rack* – a cabinet that stores a stack of interconnected nodes and other compute equipment
- *Compute Cluster* – a set of interconnected racks.
- *Shared Memory Computing* – an HPC application where each core (all residing on the same node) uses a shared block of memory.
- *Distributed Memory Computing* – an HPC application where each core (possibly residing on different nodes) uses separate blocks of memory and work is coordinated using message passing.

Two Rows of Node Racks at UB CCR



- *Thread* – short for “thread of execution”, a sequence of instructions that is run concurrently with other threads of the same process. Threads within a process can share memory.
- *Accelerator* – a compute device that a connected processor can use to offload certain computations.



UB CCR: HISTORY AND MISSION

What is UB CCR?

- CCR provides UB researchers and affiliated partners, including industry, with access to advanced computing resources.
 - Academic, Industrial, and Faculty Compute Nodes
 - High Performance Storage and Networking
 - Visualization and Cloud Computing Resources

History

- 1999 – UB CCR founded, 60 GFLOPS peak performance
- 2006 – UB CCR relocated from North Campus to downtown Medical Campus
- 2007 – Major Upgrades, 13 TFLOPS, 30 TB Storage
- 2010 – Major Upgrades, 170 TFLOPS, 500 TB Storage, GPU Nodes
- 2014 – Add Industry Cluster, 250 TFLOPS
- 2015 – Expand Storage to 4.2 PB
- 2016 to 2018 – Expand Faculty Clusters, 550 TFLOPS
- 2018 – Expand Industry and Academic Clusters, 1.3 PFLOPS, new GPU Nodes



Mission

- Enable research and scholarship among UB faculty
- Provide hi-tech workforce training
- Foster economic development and job creation among area industries



Researchers



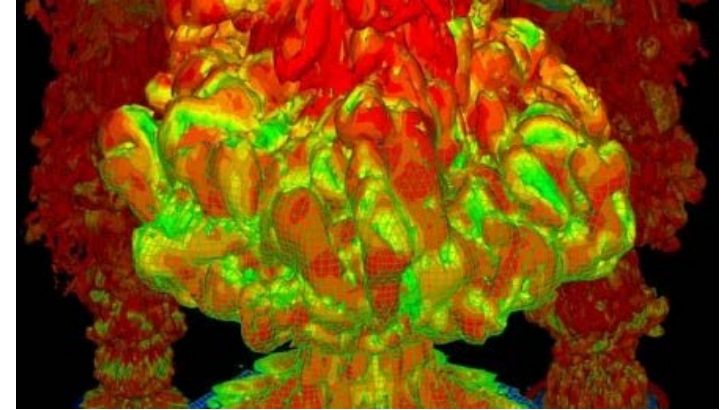
Students



Business Partners

Research

- UB CCR serves all decanal units at UB
 - 565 total users in 2017
 - 1200 publications since 2016
- No cost for faculty groups to use CCR compute resources
- Online account requests:
buffalo.edu/ccr/support/ccr-help/accounts.html
- Research topics (*not comprehensive!*):
 - Precision Agriculture
 - Bioinformatics
 - Fluid Dynamics
 - Molecular Modeling
 - Stormwater Management
 - Computer Vision
 - Computational Chemistry
 - Crystallography
 - Volcanology
 - Many other areas!



Training

- UB CCR provides free training to faculty and students on the use of its compute resources
 - By request or drop-in at our Bell Hall satellite office
- Helpdesk
 - Online ticket system for requesting technical support
 - Also provides searchable self-help knowledgebase
- Other Outreach Activities
 - Tours of the facility
 - VIDIA – Virtual Infrastructure for Data Intensive Analysis
 - Summer Workshops for High School Students
 - STEAM activities at local K-12 schools
 - ICDS/CDSE – “short course” modules and workshops
 - Guest lectures for numerous UB courses
 - Hackathon support (e.g. BlockChain, TopCoder, etc.)



Solutions / CCR Services / High Performance Cluster Computing

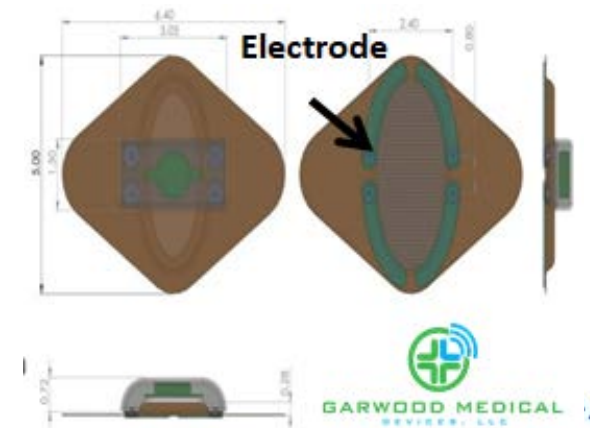
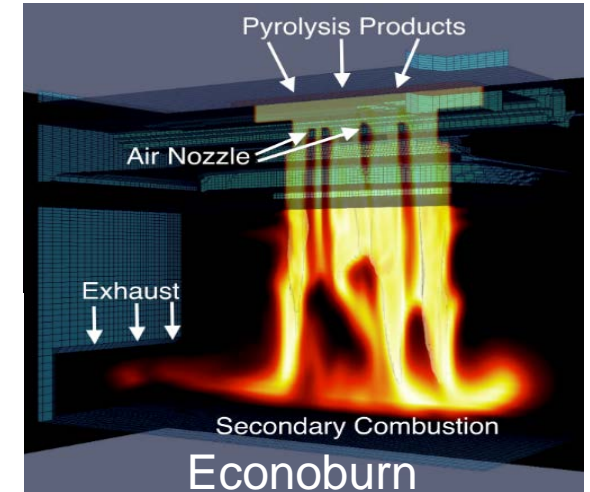


Economic Development

- UB CCR provides local industry with access to a 72 TFLOP (3,456 processors) cluster
- In process of expanding to include additional processors and a batch of NVIDIA Volta V100 GPU nodes
 - Will allow industry to leverage machine learning and deep neural networks
- Since 2013, CCR has assisted over 24 companies
 - 68 jobs created, 27 retained, \$52 million in economic impacts (savings, investment, revenue, etc.)



Sentient Science



UB CCR: COMPUTING RESOURCES

Compute Resources (as of April 2018)

- Total Compute Capability:
 - 1,373 Compute Nodes
 - 20,044 Processing Cores
- Academic Compute Capability:
 - 684 Compute Nodes
 - 7,828 Processing Cores
- Faculty* Compute Capability:
 - 471 Compute Nodes
 - 8,760 Processing Cores
- Industry* Compute Capability:
 - 216 Compute Nodes
 - 3,456 Processing Cores

Configuration of UB CCR's Academic Computing Cluster (ub-hpc)

Type of Node	Approximate # of Nodes	# Cores per Node	Clock Rate	RAM	Network*	SLURM TAGS	Local /scratch
Compute	32	16	2.20GHz	128GB	Infiniband (M)	IB CPU-E5-2660	773GB
Compute	372	12	2.40GHz	48GB	Infiniband (QL)	IB CPU-E5645	884GB
Compute (Dell)	128	8	2.13GHz	24GB	Infiniband (M)	IB CPU-L5630	268GB
Compute (IBM)	128	8	2.27GHz	24GB	Infiniband (M)	IB CPU-L5520	268GB
High Memory Compute (INTEL CPUs)	8	32	2.13GHz	256GB	Infiniband (QL)	IB CPU-E7-4830	3.1TB
High Memory Compute (AMD CPUs)	8	32	2.20GHz	256GB	Infiniband (QL)	IB CPU-6132HE	3.1TB
High Memory Compute (INTEL CPUs)	2	32	2.13GHz	512GB	Infiniband (QL)	IB CPU-E7-4830	3.1TB
GPU Compute	26**	12	2.67GHz	48GB	Infiniband (M)	IB CPU-X5650	0.9TB

* NETWORK: Infiniband (M) = Mellanox and Infiniband (QL) = Q-Logic

* - Unused Faculty and Industry cycles can be scavenged via pre-emptive scheduling



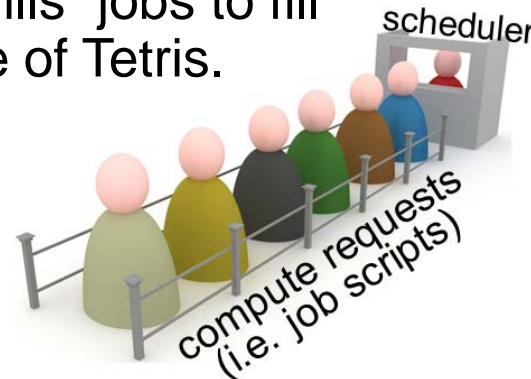
Data Storage

- Unix/Linux File Permissions
 - All nodes run CentOS Linux
- 5 GB of user space
 - backed up daily
- up to 1 TB of projects space
 - backed up daily
- ‘unlimited’ scratch space
 - deleted after 21 days
 - NOT backed up

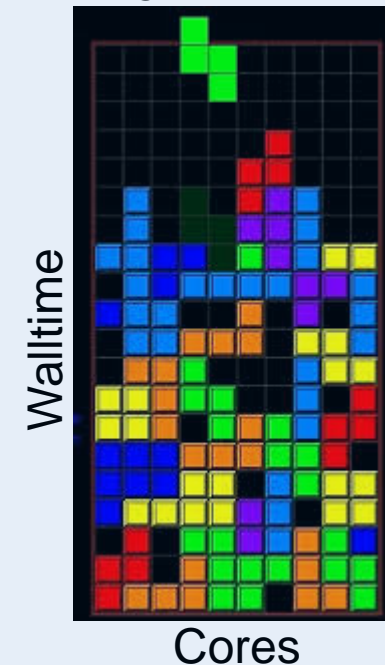


Batch Computing

- UB CCR's compute resources are organized as a batch computing system
 - Users submit compute requests (jobs) to the scheduler software (SLURM)
 - The scheduler examines requested resources (cores, memory, walltime, etc.) and determines when jobs should run and on what nodes
 - When jobs finish early, "holes" form in the schedule. The scheduler "backfills" jobs to fill these holes, kind of like a game of Tetris.

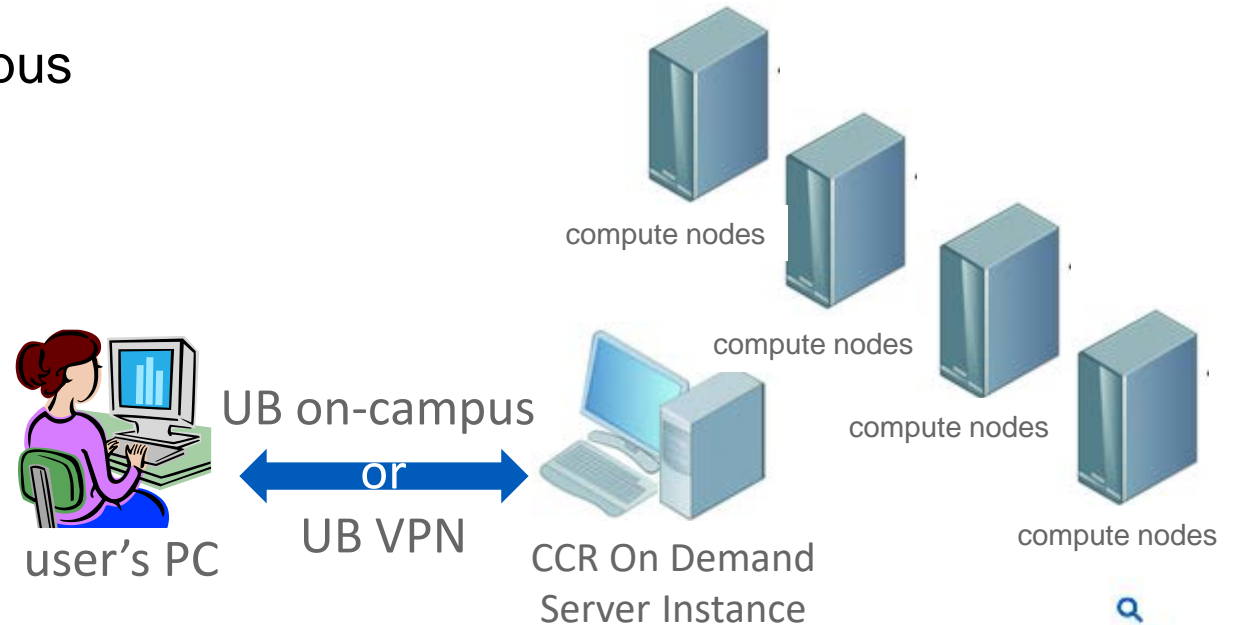


Scheduling compute jobs can be likened to a game of Tetris



Cluster Access

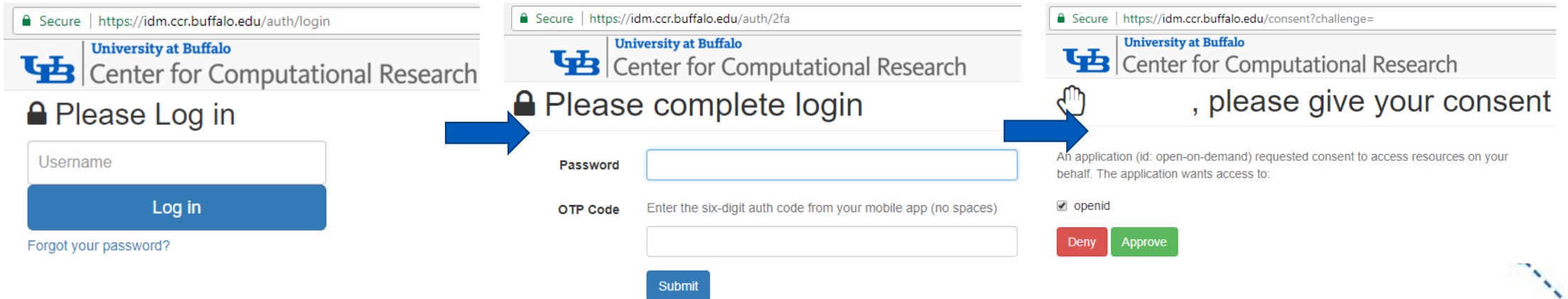
- UB CCR provides secure remote access to its resources
 - Hardware and software firewalls
 - Virtual Private Network for Off-Campus
 - Secure Shell (ssh) login node
- Connect to “CCR On Demand” to:
 - Request compute nodes
 - Load software
 - Launch batch or interactive jobs
 - Monitor job status
 - Transfer and edit files



UB CCR ON DEMAND

Login to CCR On Demand

- If off-campus, must first connect to VPN
 - UB CIT provides “AnyConnect” software (see ubit.buffalo.edu)
- Point a web-browser to ondemand.ccr.buffalo.edu
 - Login using your UB CCR username and password
 - Answer a security question or provide a two-factor code
 - Give consent for resource access



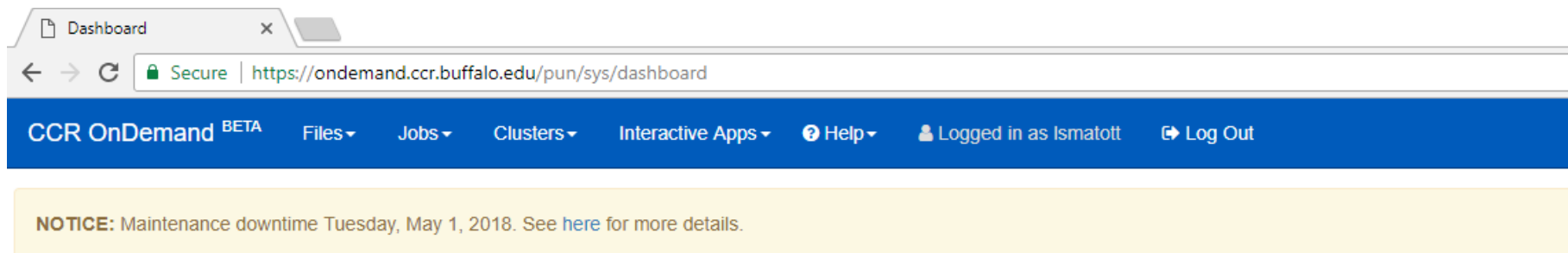
The image displays three sequential screenshots of the CCR On Demand login process, connected by blue arrows indicating the flow.

Screenshot 1: The browser address bar shows `https://idm.ccr.buffalo.edu/auth/login`. The page header includes the University at Buffalo logo and "Center for Computational Research". The main heading is "Please Log in". There is a text input field for "Username" and a blue "Log in" button. A link for "Forgot your password?" is visible at the bottom.

Screenshot 2: The browser address bar shows `https://idm.ccr.buffalo.edu/auth/2fa`. The page header is the same. The main heading is "Please complete login". There are two input fields: "Password" and "OTP Code" (with the instruction "Enter the six-digit auth code from your mobile app (no spaces)"). A blue "Submit" button is at the bottom.

Screenshot 3: The browser address bar shows `https://idm.ccr.buffalo.edu/consent?challenge=`. The page header is the same. The main heading is ", please give your consent". Below the heading, it says "An application (id: open-on-demand) requested consent to access resources on your behalf. The application wants access to:". There is a checked checkbox for "openid" and two buttons: a red "Deny" button and a green "Approve" button.

On Demand Dashboard



OnDemand provides an integrated, single access point for CCR's HPC resources

Users can transfer files, access a shell environment on the cluster front-end login server, launch interactive and remote visualization jobs, and monitor jobs all without installing any client software or web plug-ins. Access these features using the menus at the top of this page. Note that many of the apps will launch in a new tab or new browser window but the dashboard will remain open in the original window.

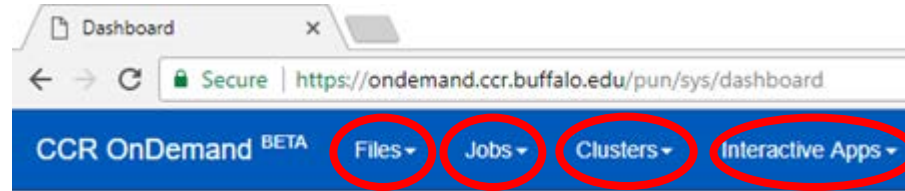
For documentation on the CCR OnDemand portal, please read these articles:

- [How to use CCR OnDemand](#)
- [Remote Visualization in OnDemand](#)

Additional CCR web resources

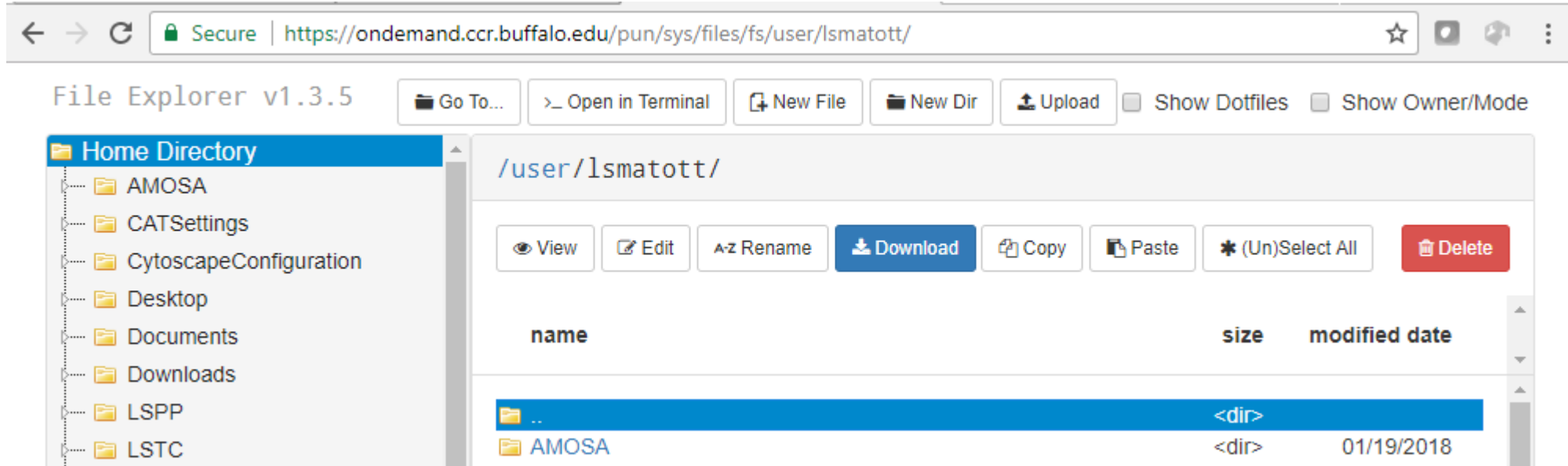
- [CCR website](#)
- [Knowledgebase and help portal](#)
- [Identity management portal - password, one-time tokens and SSH key management](#)
- [Coldfront - subscription management & account request portal for PIs](#)
- [XDMoD - job statistics and usage information](#)
- [WebMO - Chemistry application portal](#)

On Demand Dashboard



- Files
 - Launches “File Explorer”
 - Navigate storage
 - Transfer files to/from PC
 - Edit, Delete, Copy, & Rename files
- Jobs
 - Check status of active jobs
 - Compose and submit new job scripts (under development)
- Clusters
 - Command line shell terminal
- Interactive Apps
 - Remote Desktop Sessions (for GUI applications)

File Explorer



File Explorer v1.3.5

Go To... >_ Open in Terminal New File New Dir Upload Show Dotfiles Show Owner/Mode

Home Directory

- AMOSA
- CATSettings
- CytoscapeConfiguration
- Desktop
- Documents
- Downloads
- LSPP
- LSTC

/user/lsmatott/

View Edit A-z Rename Download Copy Paste * (Un)Select All Delete

name	size	modified date
..	<dir>	
AMOSA	<dir>	01/19/2018

- “Go To ...” – Navigate to project or scratch areas
- “Open in Terminal” – Open a command line shell in the currently viewed directory
- “Upload” – Transfer files from PC to CCR storage
- “Download” – Transfer files from CCR storage to PC

Active Jobs

[←](#) [→](#) [↻](#) Secure | https://ondemand.ccr.buffalo.edu/pun/sys/activejobs?jobcluster=all&jobfilter=all ☆ 🔍 🏠

CCR OnDemand / Active Jobs

All Jobs ▾ All Clusters ▾

Active Jobs

Show entries

Filter:

	ID	Name	User	Account	Time Used	Queue	Status	Cluster
>	8652018	ene_1196	sushreet	wjzheng	00:07:53	general-co...	Running	UB HPC
>	8651643	gpu3-cifar10	ruoyuyan	rahulrai		gpu	Queued	UB HPC
>	8652049	bash	arpanmuk	rahulrai		gpu	Queued	UB HPC
>	8649925	apo_mdg1_c21	sushreet	wjzheng		gpu	Queued	UB HPC
>	8649938	apo_mdg1_e1	sushreet	wjzheng		gpu	Queued	UB HPC
>	8649873	apo_mdg1_c51	sushreet	wjzheng		gpu	Queued	UB HPC
>	8651633	xdmod.benchmark.mpi.imb.job	xdtas	ccrstaff		general-co...	Queued	UB HPC



Cluster Shell

```
Secure | https://ondemand.ccr.buffalo.edu/pun/sys/shell/ssh/rush.cbls.ccr.buffalo.edu
Last login: Tue Apr 10 09:48:01 2018 from cheaptrick.ccr.buffalo.edu
#####

Next maintenance downtime: Tuesday, May 1, 2018
Time: 7am-5pm

Resources affected by downtime:

UB-HPC cluster (general-compute, debug, viz, largemem, and gpu partitions)
Industry cluster (compute, scavenger partitions)

See downtime schedule: https://ubccr.freshdesk.com/discussions/topics/13000016412

#####

NEED TO TRANSFER DATA?
Please use transfer.ccr.buffalo.edu
Faster connection speeds, less contention with other users
More details: https://ubccr.freshdesk.com/solution/articles/13000014882

#####

[lsmatott@rush:~]$
```



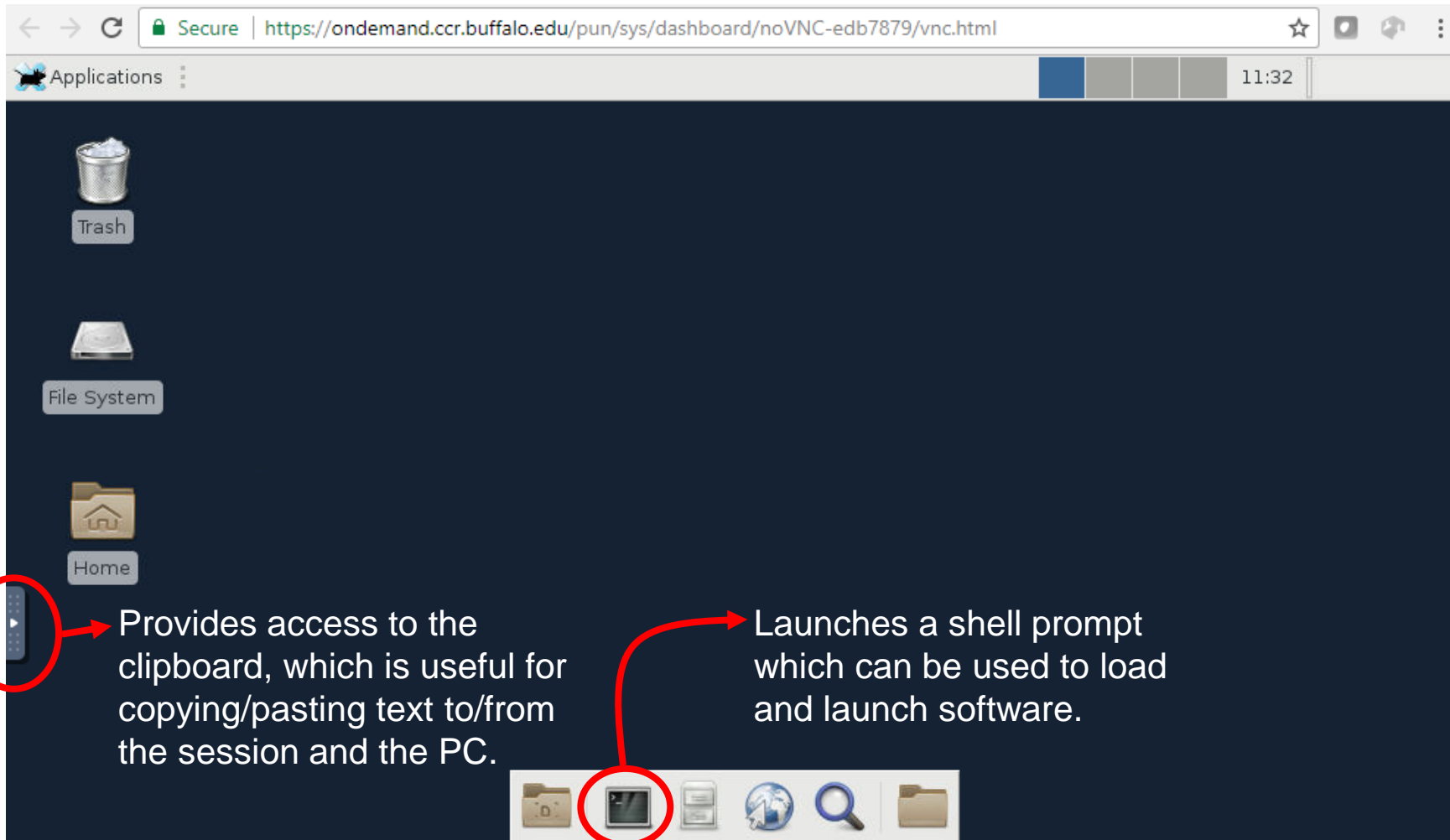
Remote Dekstop

The screenshot shows the CCR OnDemand web interface. At the top, there is a navigation bar with 'CCR OnDemand BETA' and several menu items: 'Files', 'Jobs', 'Clusters', and 'Interactive Apps'. Below this is a yellow notice box stating: 'NOTICE: Maintenance downtime Tuesday, May 1, 2018. See here for more details.' The breadcrumb trail reads 'Home / Interactive Sessions / OpenGL Desktop'. On the left, a sidebar titled 'Interactive Apps' lists several desktop options: 'UB-HPC Desktop (academic users only)', 'OpenGL Desktop' (highlighted in blue), 'StarCCM Desktop', and 'CUDA Desktop'. A red circle with the number '1' is drawn around this sidebar. The main content area contains a warning message: 'WARNING: You may experience long wait times for an OpenGL desktop. This app will launch an OpenGL enabled session on the Remote Visualization nodes. If you do not require hardware accelerated graphics, the UB-HPC desktop session will likely start much faster than this. The longest allowable wall time for these nodes is 24 hours. However, you can select less if desired and this will potentially decrease your wait time.' Below the warning is a form with a 'Number of hours' input field containing '24', which is circled in red with the number '2'. There is also a checkbox for 'I would like to receive an email when the session starts' and an 'Email Address (Must be entered to receive email)' input field. A blue 'Launch' button is circled in red with the number '3'. At the bottom, a small asterisk note reads: '* All OpenGL Desktop session data is generated and stored under the user's home directory in the corresponding data root directory.'

1. Choose the type of desktop.
2. Choose the duration of session.
3. Click Launch.
4. Wait for Session to Start.
5. When running, click “Launch noVNC in New Tab”.

The screenshot shows the session management interface for an 'OpenGL Desktop (8652514)'. The status bar indicates '1 node | 5 cores | Running'. A red 'Delete' button is visible. A blue button labeled 'Launch noVNC in New Tab' is circled in red with the number '5'. Below this is a 'View Only (Share-able Link)' button. In the background, a browser tab titled 'Dashboard' is circled in red with the number '4'.

Remote Dekstop



Provides access to the clipboard, which is useful for copying/pasting text to/from the session and the PC.

Launches a shell prompt which can be used to load and launch software.

UB CCR: SOFTWARE MODULES

Software Modules

- CCR supports 300+ software packages
- Installed as “modules”
- Important module commands:

`module avail` – what’s installed?

`module load` – load module

- Example (try from On Demand):

```
$ rstudio
```

```
-bash: rstudio: command not found
```

```
$ module load R
```

```
$ rstudio
```

```
(successfully launches rstudio)
```



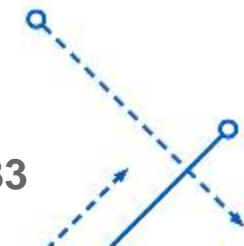
Engineering	Machine Learning	Hadoop/MapReduce	Math/Physics	Image Processing	Utilities
ABAQUS	Caffe	Hadoop	BandUP	EMAN2	7zip
ANSYS	Caffe2	HBase	CrystFEL	FFMPEG	Ant
AVL FIRE	Keras	Hive	FEniCS	OpenCV	CMAKE
CFX	MikeNet	Pig	GAP	OpenEXR	DMTCP
COMSOL	TensorFlow	Spark	Gnuid	OpenSlide	gnu-parallel
CPLEX	Theano	ZooKeeper	Gurobi	Environmental	HDF/HDF5
DAKOTA	Torch	GPU Programming	kmos	Cantera	KchmViewer
iso2mesh	Data Analytics	CUDA	ROOT	GRASS	lxml
LSDYNA	edgar	Intel-OpenCL	VMTK	HYSPLIT	Mono
NETGEN	Jags	OpenCL	Visualization	LANDIS-II	netCDF, NCO, CDO
OpenFOAM	MCL	PyCUDA	de.caff	MODFLOW	pbzip2
OpenSees	ms	Editors	NCL	NaSt3DGP	pigz
pyFormex	R	Emacs	ParaView	puffin	TauBench
QUCS	SAS	Idle	TecPlot	TELEMAC	tmux
SALOME	Stan	NetBeans	VTK	TITAN	xclip
StarCCM	STATA	Vim	xmgrace	WRF-WPS	xfig



Molecular Dynamics	Quantum Chemistry	Libraries (cont.)	Programming	Programming (cont.)
Amber , AmberTools	ABINIT	Elemental	Cilk	Julia
AutoDock	Dalton	GLPK	DDT/MAP	Mathematica
CHARMM	Elk	GMP	DMD	MATLAB
CP2k	GAMESS	Intel: MKL, MPI, TBB	gcc , gdb	Maven
GROMACS	GPAW	libgd	golang	Mercurial
LAMMPS	NWChem	libgpuarray	google-api	mpiP
MDAnalysis	OpenBabel	libgtextutils	IDL	Octave
MMTSB	Orca	libmatheval	Intel-Advisor	PAPI
MODELLER	Q-Chem	libmesh	Intel-CLCK	Perl
NAMD	QUANTUM ESPRESSO	mvapich2	Intel-Compiler	PGI
OpenMM	Siesta	OpenBLAS	Intel-DAAL	PP (Parallel Python)
PLUMED	Libraries	OpenMPI	Intel-Inspector	Python
pyRosetta , Rosetta	Armadillo	PETSc , SLEPc	Intel-IPP	Rdesktop
Schrodinger	ARPACK	scikit-tensor	Intel-ITAC	Scala
supercell	BOOST	TetGen	Intel-vTune	SCons
VMD	CGAL	Tioga	Java	Valgrind



Bioinformatics / Genomics (130 modules)									
AdmixTools	BedTools	CNVnator	fusioncatcher	Kallisto	miRDeep2	PARE	qctool	seqtk	TopHat
Admixture	bio3d	Cromwell	gatk	Kraken	MixMapper	PATSER	QIIME	SnEff	Tracer
AlignGraph	bioconda	Cufflinks	GCTA	lobSTR	mothur	pBlat	Randfold	SnSift	TreeMix
AML	bioperl	cutadapt	genipe	lumpy-sv	MrBayes	picard	RAxML	Snptest	TrimGalore
ANGSD	BLASR	DADI	GMAP	MACS	MSMC	Pindel	RELION	SomaticSniper	Trinity
ANNOVAR	BLAST	deepTools	GTOOL	MACS2	MuMmer	PLINK	RSem	speedseq	VarDict
BadiRate	Blat	deFuse	HLAreporter	mapDamage	NCBI	PLINK/SEQ	RSeQC	Squid	VCFtools
Bambino	Bowtie	dice	HOMER	MEME	NgsAdmix	Polysolver	Sailfish	SRA Toolkit	VEGAS2 / VEGAS2v02
bam-readcount	Bowtie2	EIGENSOFT	HTSeq	MERLIN	ngsPopGen	Preplot	Salmon	Strelka	Velvet
BamTools	BreakDancer	EMBOSS	HTSlib	METAL	NgsRelate	Preseq	SAMtools	StringTie	VEP
bcl2fastq	BWA	EPACTS	IgBLAST	MetAMOS	nucleo	ProDy	schmutzi	SurvivalGWAS_SV	ViennaRNA
Beagle	CDK	FastQC	Jellyfish	Minimac3	OpenGATE	psmc	SeqAn	svtyper	WDLtool
BEAST2	CGHub	FASTX	JointSNVMix	MIRA	PAML	PyBedTools	SEQLinkage	tabix	XWAS



UB CCR: SUBMITTING BATCH JOBS

About the Scheduler

- SLURM
 - Simple Linux Utility for Resource Management
- Useful scheduler commands (issue these from an On Demand “Clusters” shell prompt):
 - **sbatch** – submit a job script
 - **squeue/sqstat** – check the status of a job
 - **scancel** – delete a job
 - **snodes** – show node info and status
 - **sranks** – show job priorities
 - **stimes** – show when jobs are expected to start



Decomposing An Example Job Script

```
#!/bin/sh
#SBATCH --partition=debug
#SBATCH --time=00:15:00
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=12
##SBATCH --mem=48000
#SBATCH --constraint=IB&CPU-E5645
#SBATCH --job-name="hello_test"
#SBATCH --output=test-mpi-debug-%j.out
#SBATCH --error=test-mpi-debug-%j.err
#SBATCH --mail-user=cdc@buffalo.edu
#SBATCH --mail-type=ALL
```

Job scripts begin with a list of SBATCH directives. These directives inform the scheduler about the resources needed for the job along with who to contact when the job completes and what to do with output that would normally be printed on the screen.

```
echo "SLURM_JOB_ID"=$SLURM_JOB_ID
echo "SLURM_JOB_NODELIST"=$SLURM_JOB_NODELIST
echo "SLURM_NNODES"=$SLURM_NNODES
echo "SLURMTMPDIR"=$SLURMTMPDIR
echo "working directory = "$SLURM_SUBMIT_DIR

echo "*****"
module load intel/17.0
module load intel-mpi/2017.0.1
module list
ulimit -s unlimited
```

The next step in the job script is to configure the desired computing environment by loading required modules and setting relevant variables.

We typically also print (echo) various useful SLURM variables, like the job id, the nodelist, and the working directory. These can come in handy if there is a need to troubleshoot a completed job.

Decomposing An Example Job Script

```
#export I_MPI_DEBUG=4
export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi.so
srun ./helloworld

#
echo "All Done!"
```

The final step in the job script is to launch the desired application. How an application is launched depends on the application. **srun** is the preferred launcher for multi-node applications. For single node applications, just launch the application directly (e.g. **python ./helloworld**). Most applications also require various command line arguments.

CCR provides example job scripts for various applications. These are located in the **/util/academic/slurm-scripts** folder. You can copy the example script of your choice and edit to suit your needs.

An Example Python Multiprocessing Job Script

```
#!/bin/bash
#SBATCH --job-name=pi_mp
#SBATCH --output=pi_mp.out
#SBATCH --error=pi_mp.err
#SBATCH --mail-user=your_user_name@buffalo.edu
#SBATCH --mail-type=END
#SBATCH --time=00:10:00
#SBATCH --nodes=1
#SBATCH --cpus-per-task=12
#SBATCH --exclusive
#SBATCH --partition=general-compute
#SBATCH --constraint=CPU-E5645
#SBATCH --mem=48000
#SBATCH --tasks-per-node=12

module load python/anaconda
ulimit -s unlimited

python pi_mp.py
```

This script requests:

1 node, 12 processors, and 48 GB RAM

It loads the python/anaconda module and then launches a python script. The pi_mp.py script contains some Python code that takes advantage of the multiprocessing module to use the 12 requested processors.

An Example MPI for Python Job Script

```
#!/bin/bash
#SBATCH --job-name=mpi4py
#SBATCH --output=mpi4py.out
#SBATCH --error=mpi4py.err
#SBATCH --mail-user=your_user_name@buffalo.edu
#SBATCH --mail-type=END
#SBATCH --time=00:10:00
#SBATCH --nodes=2
#SBATCH --cpus-per-task=1
#SBATCH --exclusive
#SBATCH --partition=general-compute
#SBATCH --constraint=IB&CPU-E5645
#SBATCH --mem=48000
#SBATCH --tasks-per-node=12

# load modules
module load python/anaconda
module load intel-mpi
ulimit -s unlimited

# enable mpi4py module
export PYTHONPATH=/util/academic/python/mpi4py/v2.0.0/lib/python2.7/site-packages:$PYTHONPATH

# launch app
export I_MPI_FABRICS_LIST=tcp
export I_MPI_DEBUG=4
export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi.so
srun -n $SLURM_NPROCS python mpi4py_pi.py
```

This script requests:

2 nodes, 24 processors, and 48 GB RAM per node

It loads the python/anaconda and intel-mpi modules and then adds the location of the mpi4py installation to the Python path. Finally, it sets up the intel-mpi environment and launches a python script using the “**srun**” launcher. **srun** is the preferred MPI launcher (instead of **mpirun** or **mpiexec**) on the CCR system.

CLOSING REMARKS

What have we learned?

- Some basic HPC terms
- About UB CCR – its history and mission
- UB CCR's current resources
- CCR's On Demand access portal
- Working with modules and SLURM

