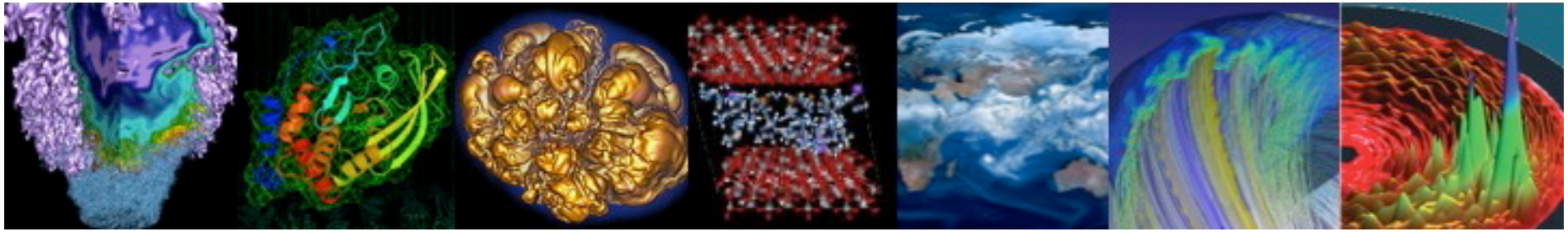


Introduction to Materials Science and Chemistry Applications at NERSC

Zhengji Zhao
NERSC User Services
ZZhao@lbl.gov
June 26, 2012

U.S. DEPARTMENT OF ENERGY | Office of Science

NERSC National Energy Research Scientific Computing Center

BERKELEY LAB Lawrence Berkeley National Laboratory

Outline

- Getting started with precompiled materials science and chemistry applications available at NERSC
- Out of memory error and parallel scaling
  - G09
  - VASP
- Running G09 on Hopper under Cluster Compatibility Mode (CCM)
- Summary

- Getting started with the precompiled materials science and chemistry applications available at NERSC

Computing resources at NERSC

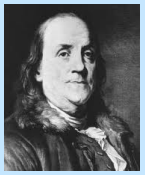
Large-Scale Computing Systems

Hopper (NERSC-6): Cray XE6

- 6,384 compute nodes, 153,216 cores
- 144 Tflop/s on applications; 1.3 Pflop/s peak

Franklin (NERSC-5): Cray XT4

- 9,572 compute nodes, 38,288 cores
- 25 Tflop/s on applications; 0.35 Pflop/s peak



Midrange

140 Tflops total

Carver

- IBM iDataplex cluster
- 9884 cores; 106TF

PDSF (HEP/NP)

- ~1K core cluster

GenePool (JGI)

- ~5K core cluster
- 2.1 PB Isilon File System

NERSC Global Filesystem (NGF)

Uses IBM's GPFS

- 8.5 PB capacity
- 15GB/s of bandwidth

HPSS Archival Storage

- 240 PB capacity
- 5 Tape libraries
- 200 TB disk cache

Analytics & Testbeds

Euclid

(512 GB shared memory)

Dirac 48 Fermi GPU nodes

Magellan Hadoop

Materials sciences and chemistry applications run on Hopper and Carver

- **Hopper** compute nodes have 24 cores per node
  - 6008 32 GB memory nodes, 1.33GB per core
  - 384 large memory nodes, 64GB per node, 2.67GB per core
- **Carver** compute nodes have 8 cores per node
  - 320 24GB memory nodes, 3Gb per core
  - Carver has 80 large memory nodes, 48 GB memory per node, 6GB per core
  - Memory limit: soft 2.5GB and 5.5GB; hard 20Gb and 44Gb, respectively

Precompiled materials science and chemistry codes at NERSC

Codes	Hopper	Carver
ABINIT	✔	✔
CP2K	✔	✔
CPMD	✔	
Quantum Espresso	✔	✔
LAMMPS	✔	✔
SIESTA	✔	✔
VASP	✔	✔
WIEN2k	✔	✔
BerkeleyGW		

Codes	Hopper	Carver
AMBER	✔	✔
G09	✔	✔
GAMESS	✔	✔
GROMACS	✔	✔
MOLPRO	✔	✔
NAMD	✔	✔
NWChem	✔	✔
Q-Chem		✔

To find the full list of avaialble applications:

1. http://www.nersc.gov/users/software/applications/
2. Type “module avail” on each machine

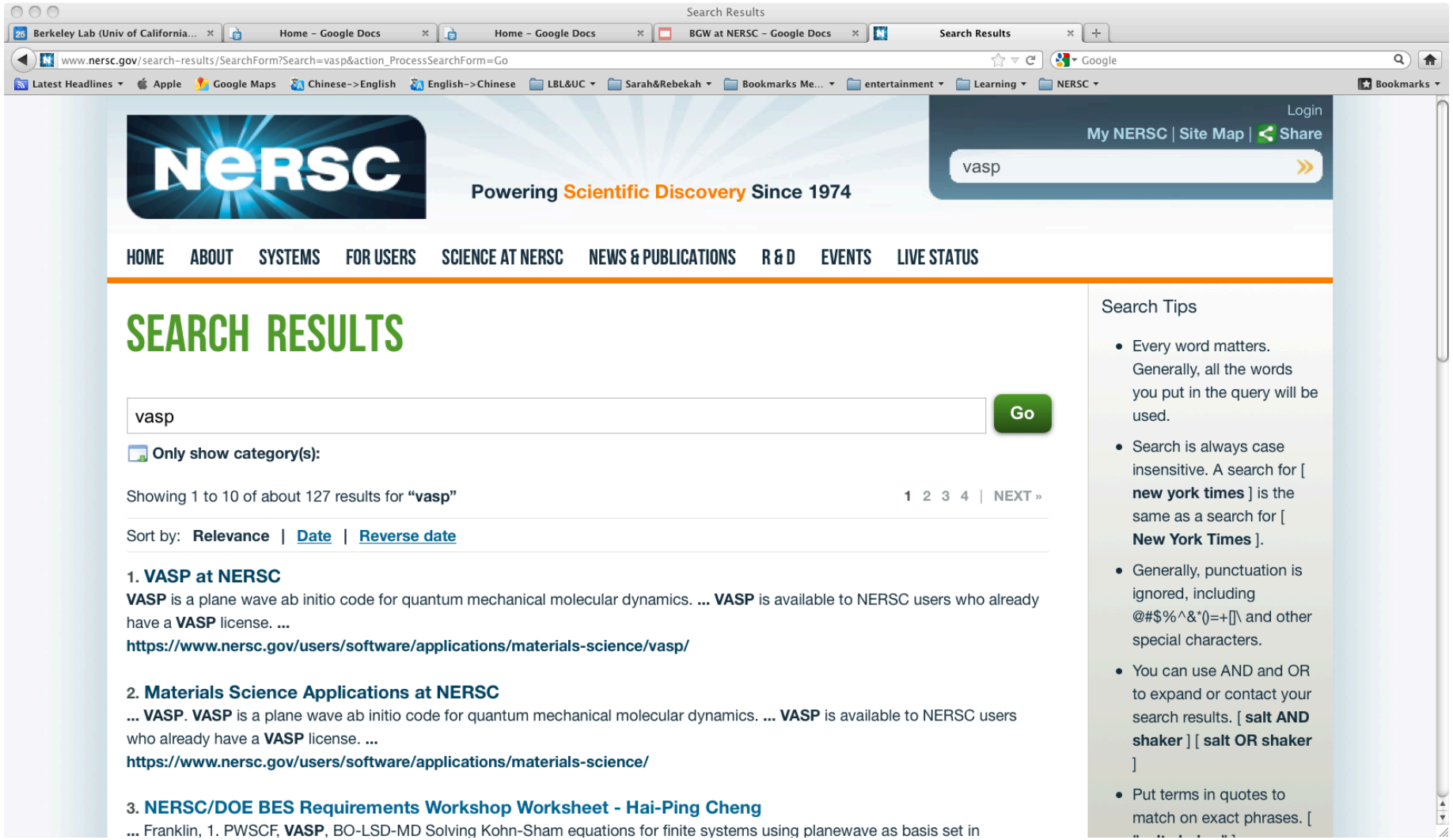
How to access - module approach

- Modules
  - An approach that manage user environment for different versions of software
  - Simply use “load” and “unload” to control user environment
  - Commonly used module commands (man module):
    - Module avail - to see available modules
    - Module load, module unload
    - Module list - to see loaded modules list
    - Module show- show what envs defined in the module
- Modules just define some environment variables in your shell environment if loaded

Some applications have access restrictions

- Access restrictions:
  - G09 – just need to agree the license statement
  - VASP – available to the users who own VASP licenses by themselves
- Some module commands display
  - module show vasp
    - ls -l /usr/common/usg/vasp/5.2.11/bin
  - module load vasp
    - which vasp
  - module show g09
    - ls -l /usr/common/usg/g09/b11/g09/*.exel
    - ls –l /usr/common/usg/g09/b11/g09/tests/com

Application specific website at NERSC



How to run jobs on Carver

Running interactively

```
% qsub –I –V –l nodes=2:ppn=8 –q
    interactive

    % cd $PBS_O_WORKDIR
    % module load vasp
    % mpirun –np 16 vasp
```

Running through batch jobs

```
% cat test.pbs
        #PBS -N test_vasp
        #PBS -q regular
        #PBS –l nodes=4:ppn=8
        #PBS –l walltime=12:00:00
        #PBS -j oe
        #PBS -V
        cd $PBS_O_WORKDIR
        module load vasp
        mpirun -n 32 vasp
% qsub test.pbs
```

Note:

Be aware of the parallel job launching scripts in chemistry codes, eg., qchem, molpro, gamess,…, the aprun or mpirun is called inside the launching script.

How to run jobs on Carver

G09 sample job script

```
#!/bin/bash -l
#PBS -N t1
#PBS -q regular
#PBS -l nodes=2:ppn=8,walltime=06:00:00
#PBS -j oe
#PBS -V

mkdir -p $SCRATCH/g09/$PBS_JOBID
cd $SCRATCH/g09/$PBS_JOBID
module load g09
ulimit –Sv unlimited
g09l < $HOME/g_tests/T/t1.gif > $HOME/
g_tests/T/t1.out
ls -l
```

Memory limit on Carver compute nodes:

2.5GB and 20GB for soft and hard memory limit on small memory nodes;
5.5GB and 44GB for large memory nodes

To raise the limit:

For bash/ksh:
ulimit –Sv unlimited
For csh/tcsh:
limit vmemoryuse unlimited

Standard out/error redirection:

avoid file name conflict

```
#PBS –o t1.out
#PBS –e t1.err
```

How to run jobs on Carver

G09 sample job script

```
#!/bin/bash -l
#PBS -N t1
#PBS -q regular
#PBS -l nodes=2:ppn=8,walltime=06:00:00
#PBS -j oe
#PBS -V

#Or if running from a scratch directory

cd $PBS_O_WORKDIR
module load g09
ulimit –Sv unlimited

g09l <t1.gif > t1.out
```

Memory limit on Carver compute nodes:
2.5GB and 20GB for soft and hard memory limit on small memory nodes;
5.5GB and 44GB for large memory nodes

To raise the limit:
For bash/ksh:
ulimit –Sv unlimited
For csh/tcsh:
limit vmemoryuse unlimited

Standard out/error redirection:
avoid file name conflict

```
#PBS –o t1.out
#PBS –e t1.err
```

Running on scratch file system

- MP2 in g09 can easily fill up your global home quota (40GB)
  - Nbasis=694, %Nproclinda=4

```
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-14106.scr-00003
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-17399.scr-00002
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-10198.scr-00001
-rw-r--r-- 1 zz217 zz217 557272006656 Mar 30 23:57 Gau-9483.rwf
```

- Always run jobs on scratch file system to access larger quota (20TB) and better IO performance.
- cd $SCRATCH or cd $GSCRATCH to access your scratch directory

Note: Scratch file system is subject to purge, save important results to HPSS archive system.

http://www.nersc.gov/nusers/systems/hpss/

How to run jobs on Hopper

Running interactively

```
% qsub –I –V –l mppwidth=48 –q
    interactive

    %cd $PBS_O_WORKDIR
    % module load vasp
    % aprun –n 48 vasp
```

Running as batch jobs

```
% cat test.pbs
        #PBS -N test_vasp
        #PBS -q regular
        #PBS –l mppwidth=128
        #PBS –l walltime=12:00:00
        #PBS -j oe
        #PBS -V
        cd $PBS_O_WORKDIR
        module load vasp
        aprun –n 128 vasp
% qsub test.pbs
```

Tips:

Keep jobids in the batch standard output and error file names, $PBS_JOBNAME.[o,e]$PBS_JOBID for debug (eg., to query job logs)

Running on unpacked nodes

Running on 12 cores per node

```
% cat test.pbs
      #PBS -N test_vasp
      #PBS -q regular
      #PBS –l mppwidth=768
      #PBS –l walltime=12:00:00
      #PBS -j oe
      #PBS -V

      cd $PBS_O_WORKDIR
      module load vasp
      aprun –n 384 –N12 –S3 vasp
% qsub test.pbs
```

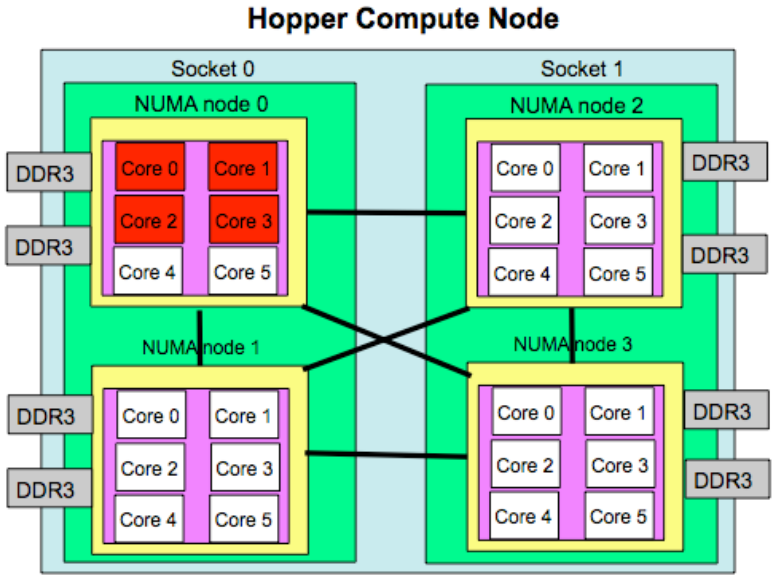
-N12: 12 tasks per node,
-S3: 3 tasks per numa node/socket
man aprun for aprun options

Note:

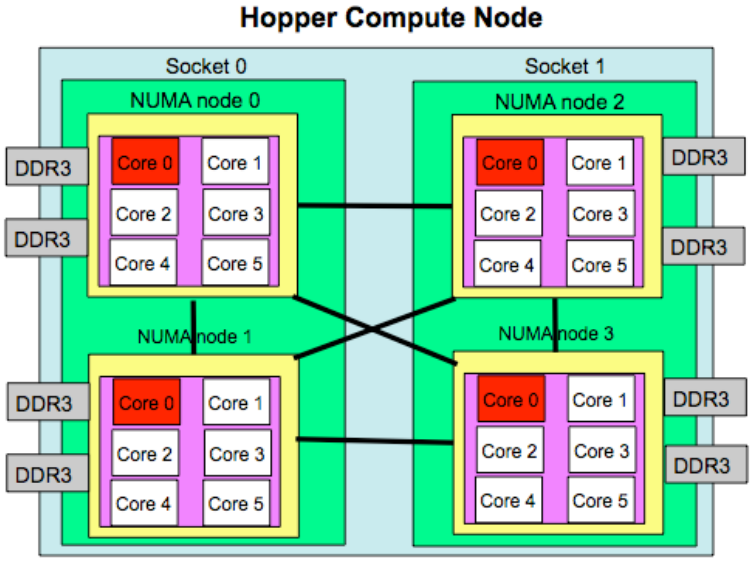
The -S option is important when running on fewer than 24 cores on the node. On a vasp job with 660 atom system, ~2.5 times performance difference has been observed.

Optimal MPI task placement on Hopper nodes

aprun –n128 –N4



aprun –n128 –N4 –S1



Bundle up jobs on Hopper

```
#!/bin/bash -l
#PBS -q regular
#PBS -l mppwidth=144
#PBS -l walltime=12:00:00
#PBS -N my_job
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module load vasp
for d in 1 2 3
do
cd dir$d
aprun -n 72 vasp &
cd ../
done
wait
```

This is useful when these jobs have similar run time

Note:

A similar job script like this would not work on Carver, because the parallel job launcher mpirun on Carver always starts from the first node allocated regardless if other jobs have used the node or not. For Carver job script, see

http://www.nersc.gov/users/computational-systems/carver/running-jobs/batch-jobs/#toc-anchor-10/

Useful commands

- qsub, qstat, qalter, qmove
  - qstat -Qf
  - qalter –l walltime=15:00 jobid
  - qmove debug jobid
- Showq, checkjob
  - Showq - approximate priority in the queue
  - Checkjob –v jobid, check job status, can catch some obvious errors
- man qsub

Good practice

- Test job script before submitting a long job using interactive/debug queue
- Request the shortest safe wall clock time if possible for a better queue turnaround
- Check point your jobs if available
- Keep job ids for your jobs for debug purpose in case your jobs run into errors

- Out of memory error and parallel scaling

Two types of memory errors

- Memory requirement depends on job types, and code implementations
- Some codes work within the memory requested (or default memory), G09, NWChem, Molpro, …, etc.
  - Gracefully exit when memory is not sufficient
- Others use all memory available on the node, VASP, Quantum Espresso, LAMMPS, NAMD,…
  - Killed by the operating system (OOM killer)

Parallel scaling issues

- Running at too high concurrency
  - Not necessarily reduce the time to solution
  - Code behavior often is not predictable outside of the scaling region
  - Waste resources
- Running at too low concurrency
  - Lose productivity unnecessarily
  - Easily run into memory issues

Example 1: G09

- Request memory in the input file:
  - Default 32mw=256mb
  - %mem=18gb for SMP+Linda parallel execution
  - %mem=2gb for Linda only parallel execution
- Parallel execution of G09 has to be requested in the g09 input file
  - %NprocShared=8
  - %NprocLinda=2
  - If not, jobs run in serial/SMP, only 1 core in use, the rest idle
  - Slowdown productivity, wasting computing resources

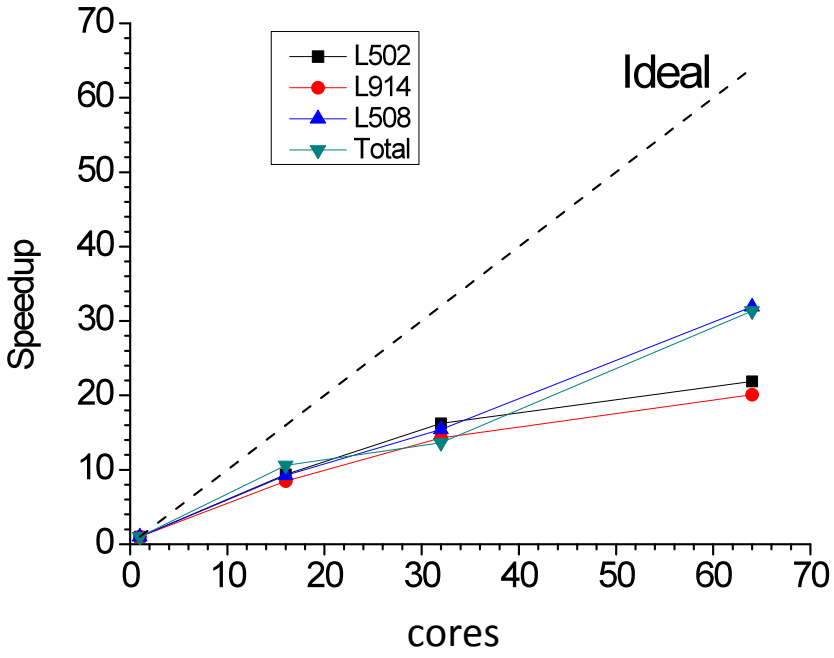
Memory usage of G09

- G09 provides ways to estimate the memory requirement for various jobs
  - $M + 2(N_B)^2$ (in 8-byte words), where M is the default 32mw, N_B is the number of basis functions
  - freqmem – determines the memory needed for frequency jobs
- Link 0 Recommendations:
  - Run in SMP+Linda parallel
  - %mem=18gb
  - %NprocShared=8
  - %NprocLinda=2
- G09 reduces threads until fit into memory

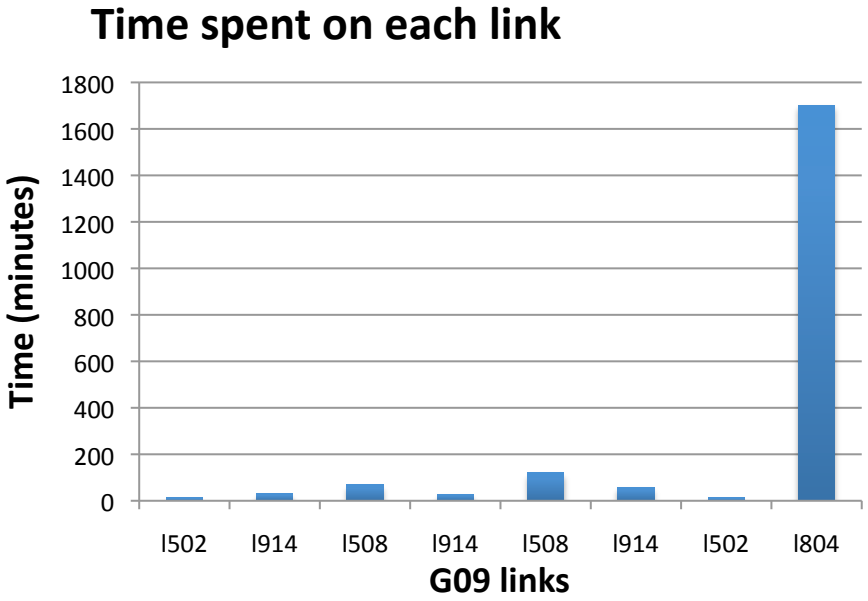
Parallel scaling of G09

- G09 runs with a sequence of executables (links), but not all of them can run on multiple nodes
  - Some of them are Linda parallel (could run on multiple nodes); and some of them are SMP parallel only; and the rest are serial only. 17 out of 79 links are Linda parallelized.
  - Use %Kjob l301 to find out if the sequence of executables (links) that need to run in advance, so to determine if the main components of your calculation can be run on multiple nodes or not.

Parallel scaling of some g09 links



UHF calculation for a system with 61 atoms, NBasis=919



UHF calculation for a system with 61 atoms, NBasis=919

Followed by UCIS calculation

NprocLinda=4

Nprocshared=8

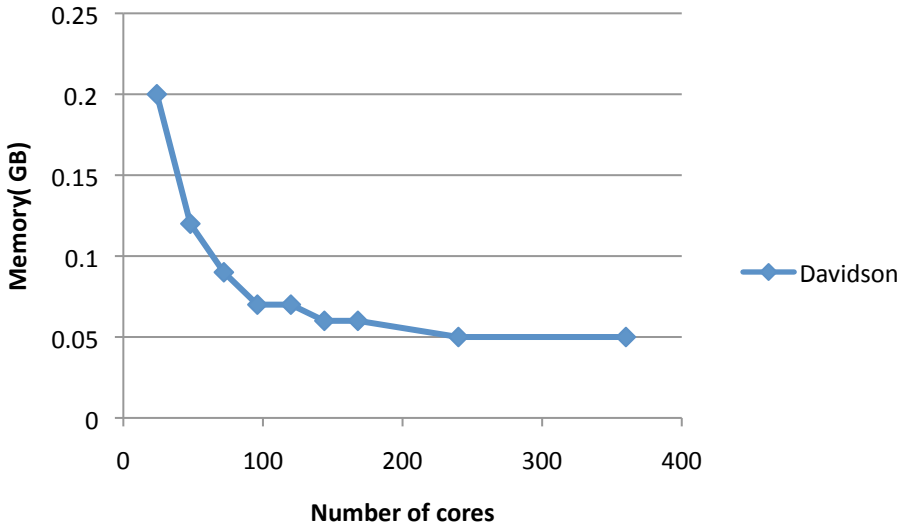
Note: Link 804 runs on only one node, other 3 nodes idle!

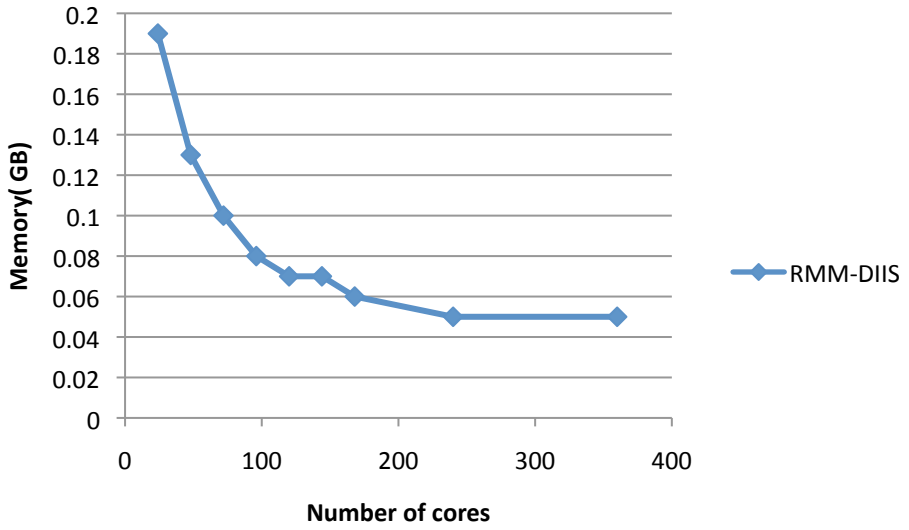
Parallel scaling of G09

- When using multiple nodes to run g09 jobs, need to consider if the serial/SMP only components are the most time consuming part.
- If yes, then submit separate jobs (dependent jobs) instead of a job with multiple job steps
- Using many nodes to run g09 is not a very good idea, use with caution

Example 2: VASP

- Memory requirement of VASP
  - Accurate estimation is difficult for a parallel run
  - NKDIM*NBANDS*NRPLWV*16 –wave function
  - 4*(NGXF/2+1)*NGYF*NGZF*16 – work arrays
  - http://cms.mpi.univie.ac.at/vasp/guide/node90.html





Test case: 150 atoms system for 1 k-point, measured max memory during the first 4 electronic steps on Hopper

**** VASP performance data provided by NERSC summer student, Matthew Farrell.**

If your job run out of memory (OOM)

- Use smaller NPAR if possible. The default NPAR=the number of cores used (upto 256 cores). But some VASP jobs don't run with reduced NPAR value, eg., hybrid.
  - http://cms.mpi.univie.ac.at/vasp/guide/node139.html
- Use more cores
  - so each core needs to store less distributable data
- Running on reduced number of cores per node
  - more memory available for each task, especially helpful if the memory to store local data was not sufficient
- Use larger memory nodes
  - Trade-off is slow queue turnaround

Running on a reduced number of cores per node on Carver

```
#PBS -q regular
#PBS -l nodes=4:ppn=2
#PBS -l pvmem=10GB
#PBS -l walltime=00:10:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module lod vasp
mpirun -np 8 vasp
```

Process Memory Limits

Type of Node	Soft Limit	Hard Limit
Login Node	2GB	2GB
24GB Compute Node	2.5GB	20GB
48GB Compute Node	5.5GB	44GB

ppn	24GB Node	48GB Node
1	pvmem=20GB	pvmem=44GB
2	pvmem=10GB	pvmem=22GB
4	pvmem=5GB	pvmem=11GB

Running on large memory nodes

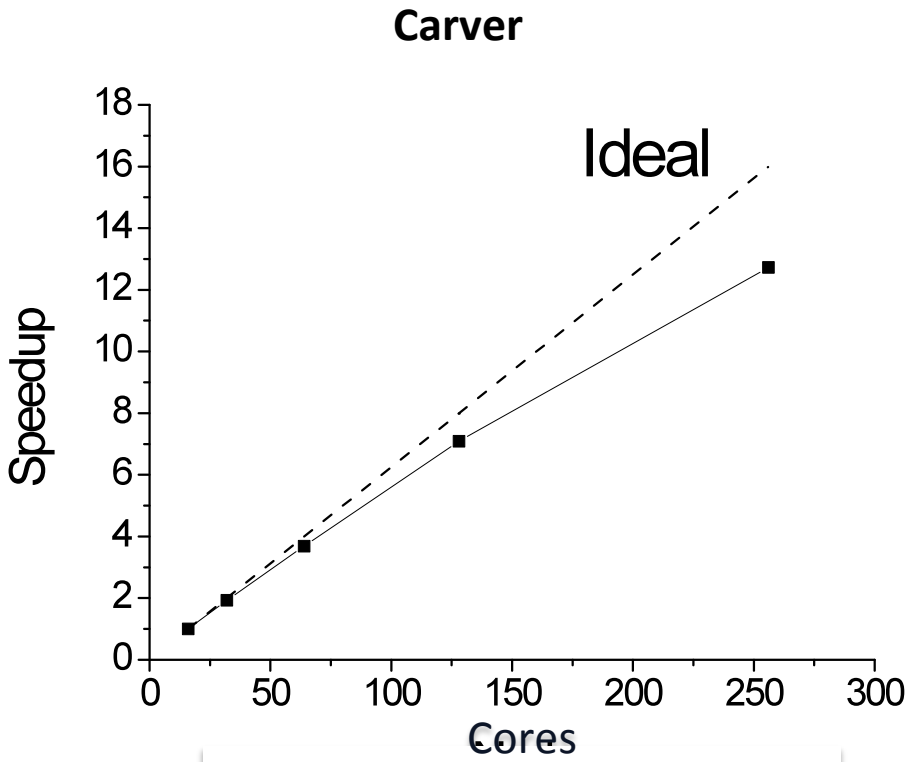
Carver

```
#PBS -q regular
#PBS -l nodes=4:ppn=8:bigmem
#PBS -l walltime=00:10:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module lod vasp
mpirun -np 8 vasp
```

Hopper

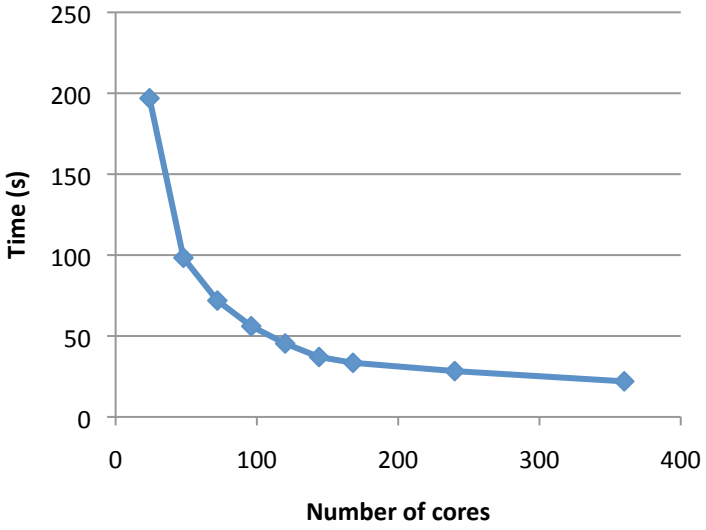
```
#PBS -q regular
#PBS –l mppwidth=768
#PBS -l mpplabels=bigmem
#PBS –l walltime=00:30:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module load vasp
aprun –n 768 vasp
```

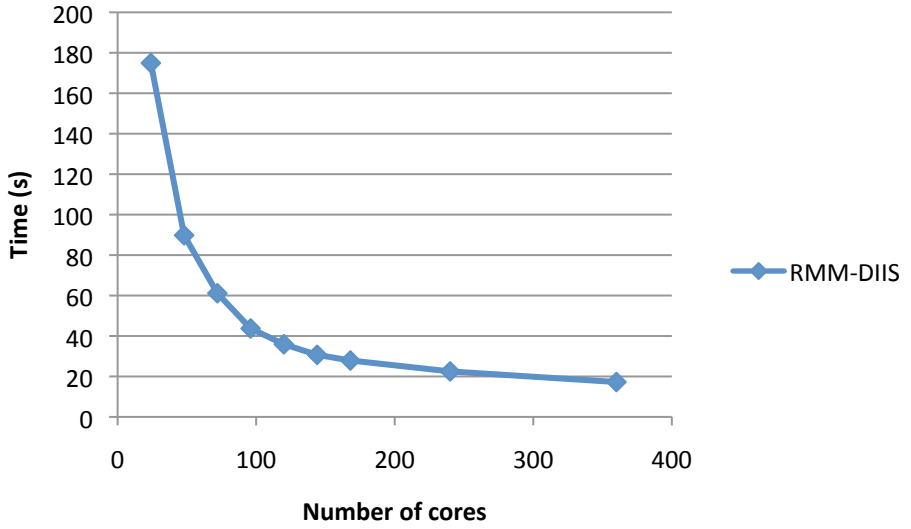
Parallel scaling of VASP

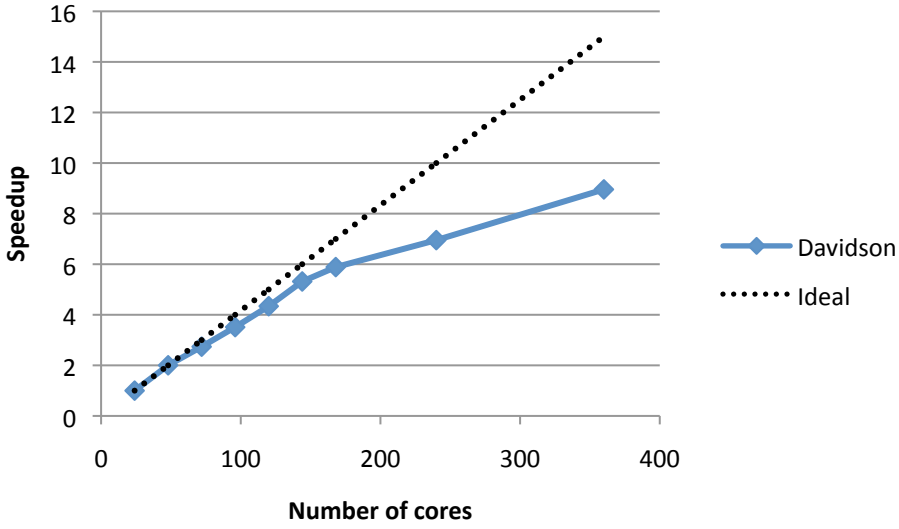


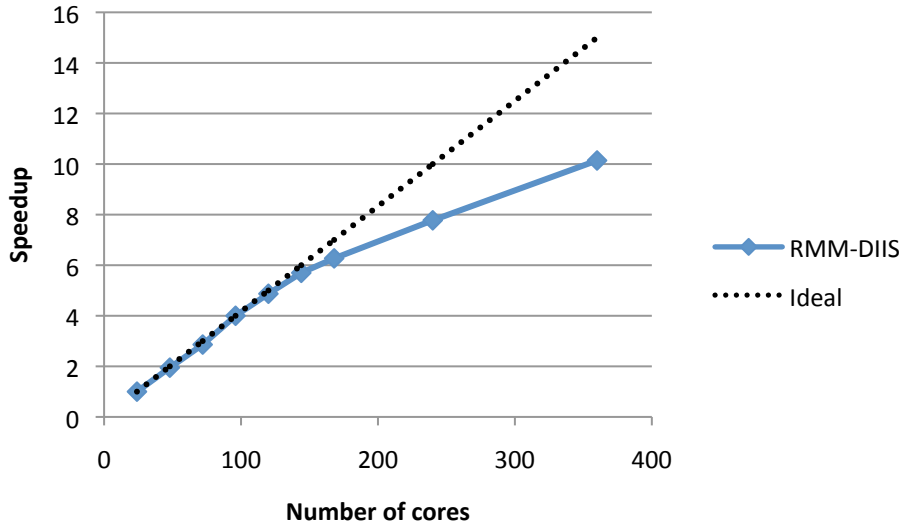
Strong scaling
System with 154 atoms,8-kpoints

Parallel scaling of VASP (Test case: 154 atom, vasp, 1-kpoint)



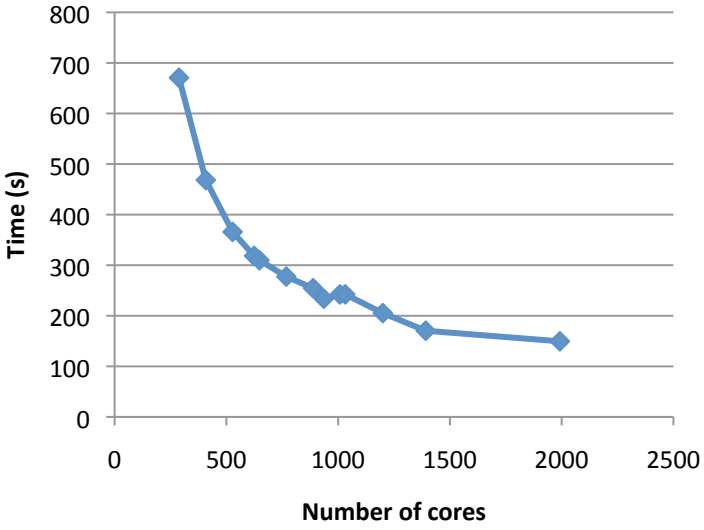


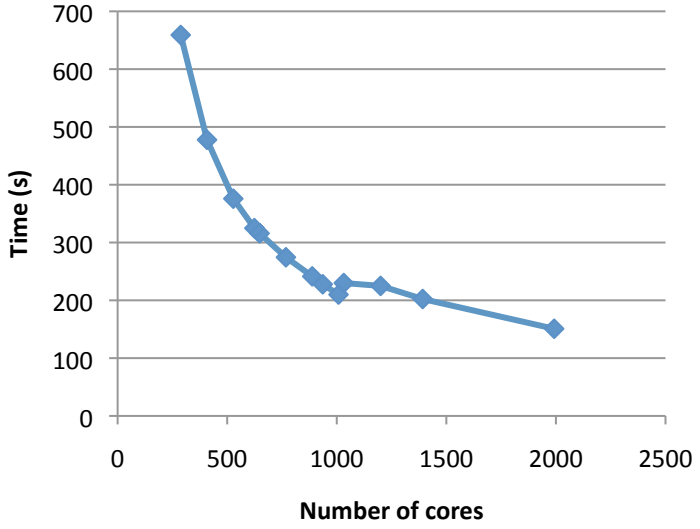


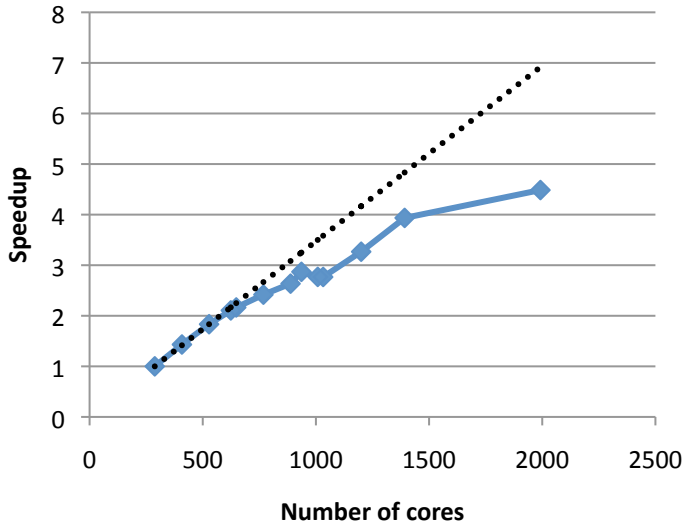


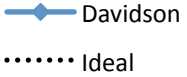
Parallel scaling of VASP

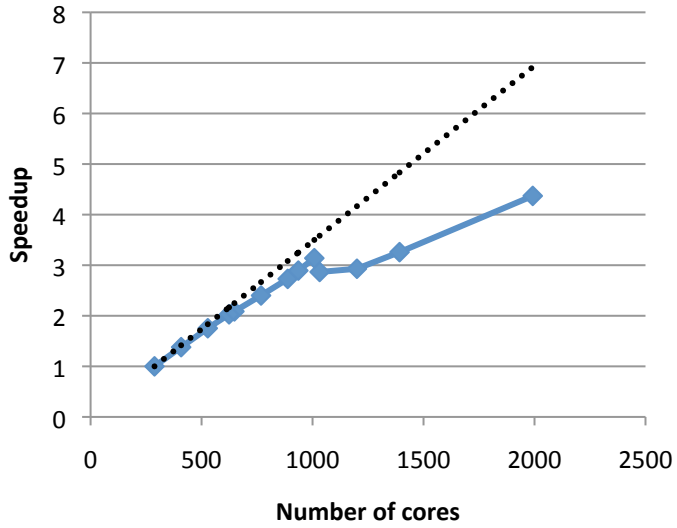
(Test case: 660 atom, gvasp)

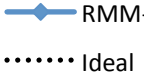












Gamma point only VASP

- Comparison between gamma-only version and the general k-point version

	Memory (GB)	Execution time(s)	WAVECAR size
General kpoint version	0.61	209*	21328479040
Gamma point only version	0.49	204*	10664239520

*Time to execute the second SC step for RMM-DIIS scheme for 660 atom system

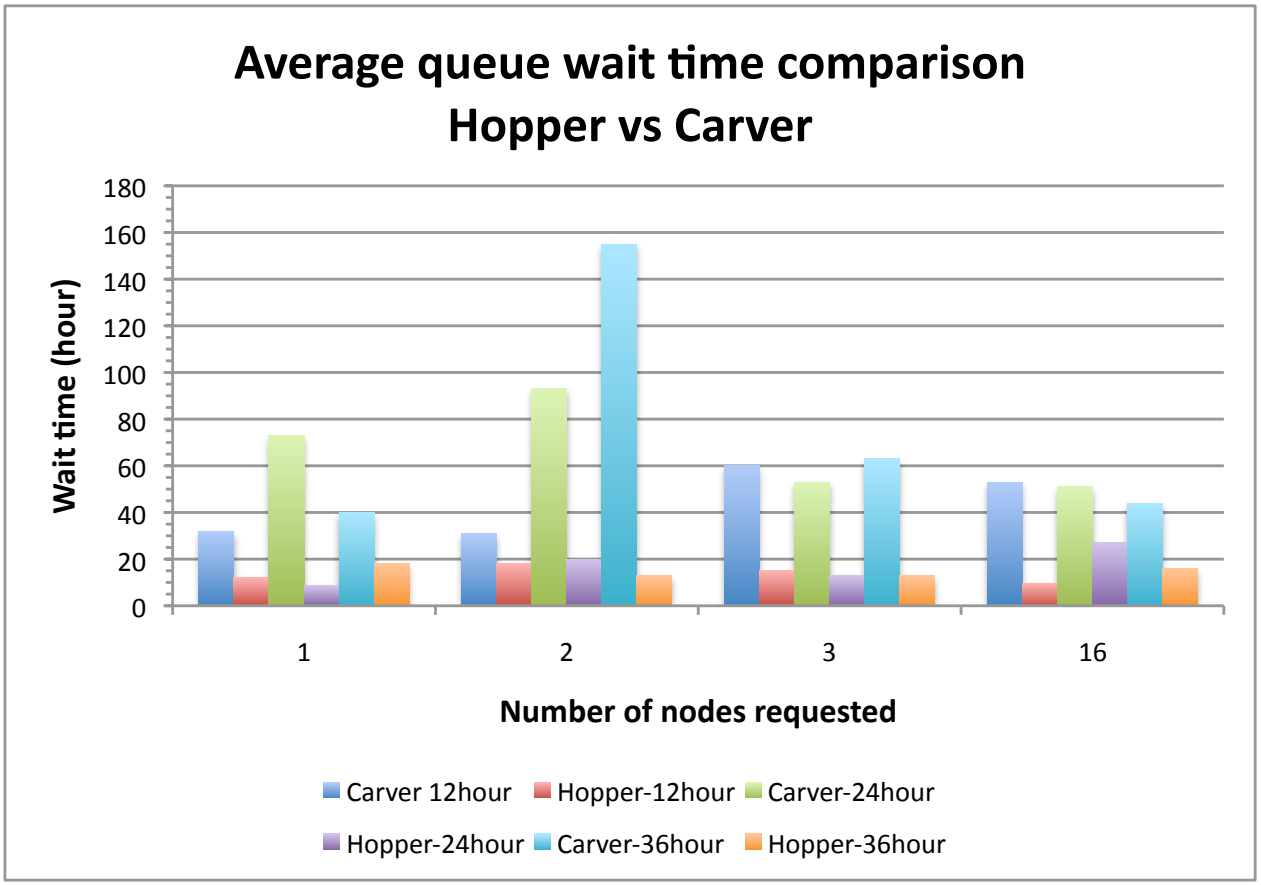
- It is recommended to use the gamma point only version if the system contains only gamma point

Parallel scaling of VASP

- VASP (5.2.11) scales well up to near 1 core/atom level both on Hopper and Carver
- When choosing how many cores to use for your job, 1/2 ~ 1 core/per atom would be a good number to start with.
- The scaling of VASP could be affected by many other parameters.

- Running G09 jobs under Cluster Compatibility Mode (CCM) on Hopper

Queue wait time for the last 3 months on Carver and Hopper



Hopper - Cray XE6,
Two 12-core AMD 'MagnyCours' 2.1 GHz processors
Peak flops: 1.28 Petaflops
6384 nodes, **153,216 cores**
24 cores/node
32 GB memory per node
Gemini Interconnect

Carver – IBM iDataPlex
Intel Nehalem 2.67 GHz processors, 8 cores/node
Peak flops: 106.5 Tflops
1202 nodes, **9984 cores**
24GB memory per node
4X QDR InfiniBand

2 node jobs requesting 36 hours waited 4 times longer on Carver than on Hopper!

Data covers the period of 7/1/2011-10/1/2011 and obtained from
https://www.nersc.gov/users/job-information/usage-reports/jobs-summary-statistics/

Why these users prefer to stay on Carver?

- Some applications don't run on Hopper due to the lack of TCP/IP support on compute nodes
  - Gaussian code, NAMD replica simulations
  - Wien2k
- Friendly environment for serial workload
- Faster processor
- Larger memory per core
- More queue and memory options
  - Huge memory node (1TB)
  - 3 weeks long queue
  - Serial queue
- Data analysis tools not available on Hopper
  - Matlab

Why these users prefer to stay on Carver?

- Some applications don't run on Hopper due to the lack of TCP/IP support on compute nodes
  - Gaussian code, NAMD replica simulations
  - Wien2k
- Friendly environment for serial workload
- Faster processor
- Larger memory per core
- More queue and memory options
  - Huge memory node (1TB)
  - 3 weeks long queue
  - Serial queue
- Data analysis tools not available on Hopper
  - Matlab

What is Cluster Compatibility Mode (CCM)?

- CCM is a Cray software solution that provides services needed to run most cluster-based independent software vendor (ISV) applications on the Cray XE6. CCM supports
  - TCP/IP - MPI runs over it
  - Standard services: ssh, rsh, nscd, ldap
  - Complete root file system
- CCM is implemented as a queue, ccm-queue
  - Dynamically allocates and configures compute nodes when a CCM job starts
  - ccmrun - launch jobs onto the head node
  - ccmlogin – interactively login to compute nodes
  - Nodes are released to compute nodes pool at job exit
- In CCM, applications run in a generic Linux cluster environment

More info about CCM:
http://www.nersc.gov/users/computational-systems/hopper/cluster-compatibility-mode/

CCM enables G09 jobs on Hopper

- G09 parallel implementation:
  - Master/slave mode
  - Intra node: OpenMP
  - Inter node: **ssh**
- A g09 job consists of Links - component executables
- Test case (user case):
  - UHF calculation for a system with 61 atoms, NBasis=919
  - Figure (next slide) shows the 3 most time consuming components of the job, Link 502, Link 914 and Link 508.
- G09 were run under CCM on Hopper test machine, Grace

Gaussian 09 job script

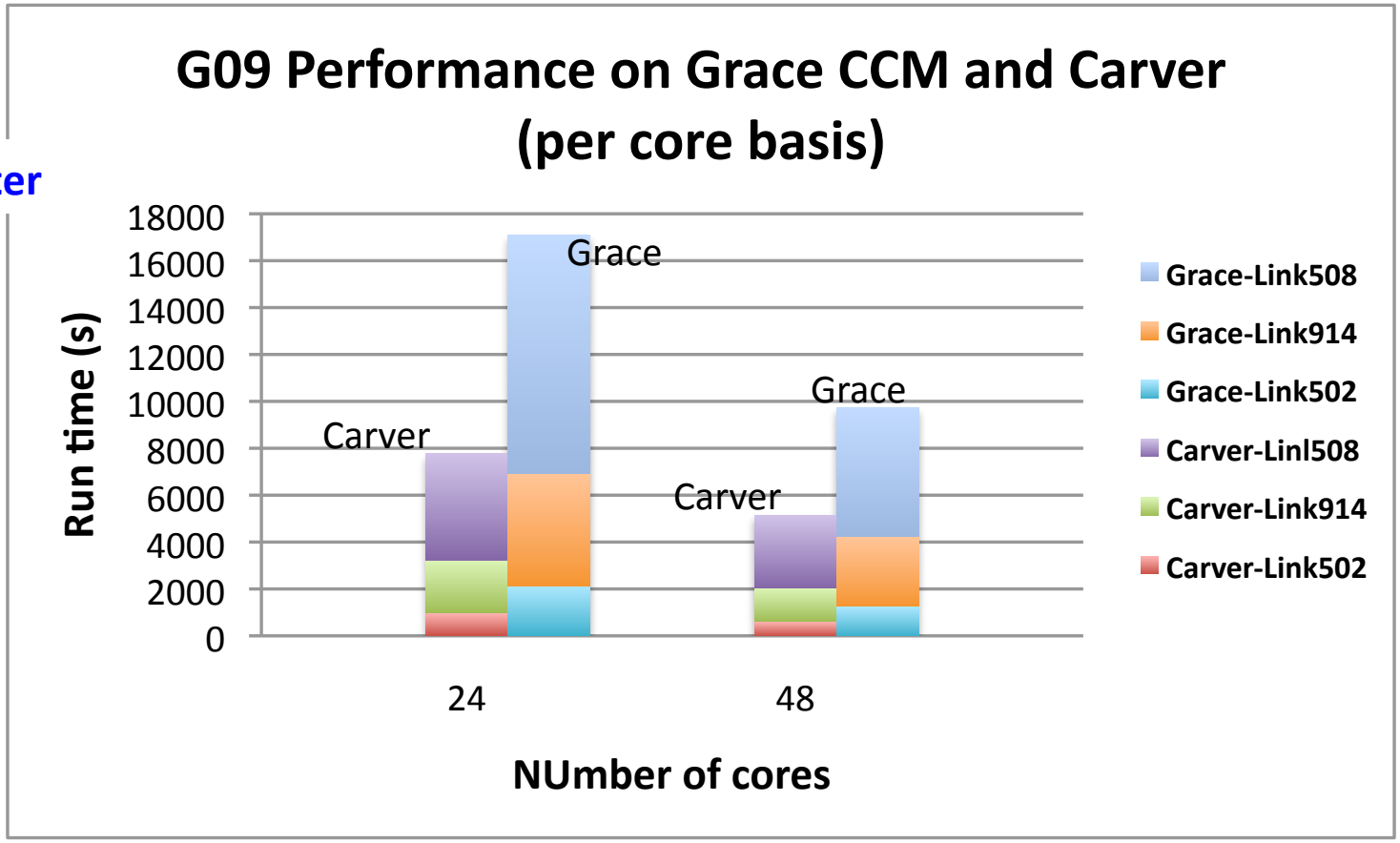
```
#PBS -S /bin/csh
#PBS -N ccm_g09
#PBS -q ccm_queue
#PBS -l mppwidth=48,walltime=24:00:00
#PBS -j oe

module load ccm
setenv CRAY_ROOTFS DSL

module load g09
set input=input.L2S24
set output=output.L2S24.$PBS_JOBID

mkdir -p $SCRATCH/g09/$PBS_JOBID
cd $SCRATCH/g09/$PBS_JOBID
ccmrun g09l < $PBS_O_WORKDIR/$input > $PBS_O_WORKDIR/$output
```

In ~/.cshrc.ext file, add
module load g09

```
%mem=24gb
%Nproclinda=2
%Nprocshared=24
```

CCM enables G09 jobs on Hopper

Lower is better



1. G09 runs around 2 times (runtime doubled) slower than on Carver when running on the same number of cores.
2. Most Carver g09 jobs will fit to run on 1-2 nodes on Hopper.

A few notes about G09 on Hopper

- All g09 jobs ran with 24 threads with a single task on the node.
- There is no good way to control the process/memory affinity through ssh, so the OpenMP threads don't run at the optimal task/thread ratio and placement on Hopper nodes.
- Running G09 over multiple nodes doesn't shorten time to solution due to a CCM bug. So running on single node only is recommended until the bug is fixed.

Summary

- Getting started with the precompiled materials and Chemistry applications at NERSC
  - Visit our application website to get NERSC specific knowledge about using applications provided by NERSC
  - Run jobs on $SCRATCH file system to access larger disk quota and better IO performance.
  - Request the shortest safe wall clock time if possible for a better queue turnaround, and checkpoint your jobs if available
  - Bundle up multiple jobs where applicable

Summary --continued

- Taking G09 and VASP as examples, addressed two major issues (memory and parallel scaling) users run into when running jobs at NERSC
  - For G09 our recommendation is to request the maximum available memory for g09 jobs and not to use a lot of nodes unless you know what you are doing. Be aware of the serial components of your g09 jobs.
  - For VASP jobs that run into out of memory error, in addition to trying NPAR=1 in the VASP input file where applicable, they could be run on more cores and/or on a fewer number of cores per node. Also large memory nodes can be used.

Summary --continued

  - VASP can scale up to ~ 1 core/atom, although the parallel scaling of VASP code depends on the problem sizes and various internal control parameters of VASP. Using ½ core/atom – 1 core/node is a good core count to start with. And use medium NPAR value, eg., ~sqrt(# of cores) where applicable
  - Use the Gamma-point only VASP (gvasp) for Gamma point only calculations
- Users are encouraged to run g09 on Hopper under CCM to get a better queue turnaround.
- We didn't address other codes, but way manage VASP out of memory error can be applied to other codes as well.

- Recommended readings:
  - NERSC website, especially
    - http://www.nersc.gov/nusers/systems/carver/running_jobs/index.php
    - https://newweb.nersc.gov/users/computational-systems/hopper/running-jobs/
  - man pages:
    - mpirun
    - aprun
    - qsub, runtime environment variables

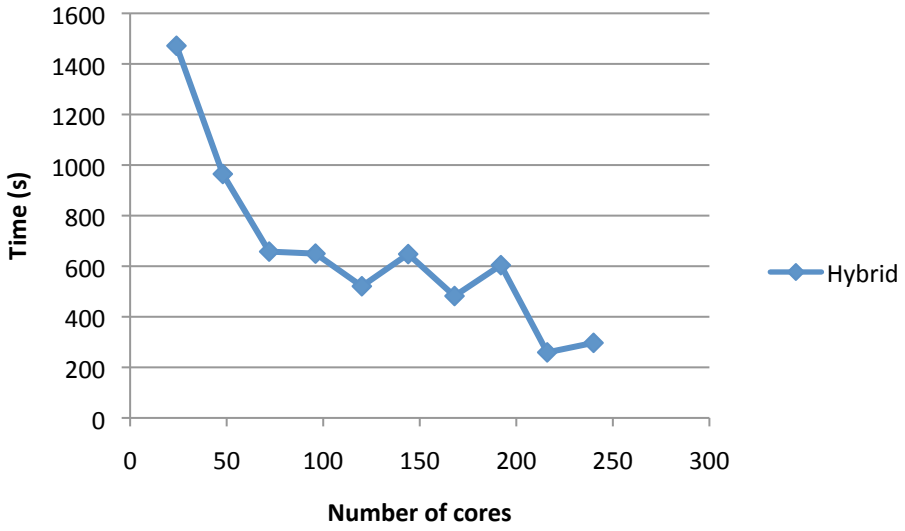
- Ask NERSC consultants questions
  - Email: consult@nersc.gov
  - Phone: 1-800-666-3772 (or 1-510-486-8600), menu option 3
  - We work with users on a variety of issues
  - Some issues can be solved immediately, others require collaborations for weeks or months

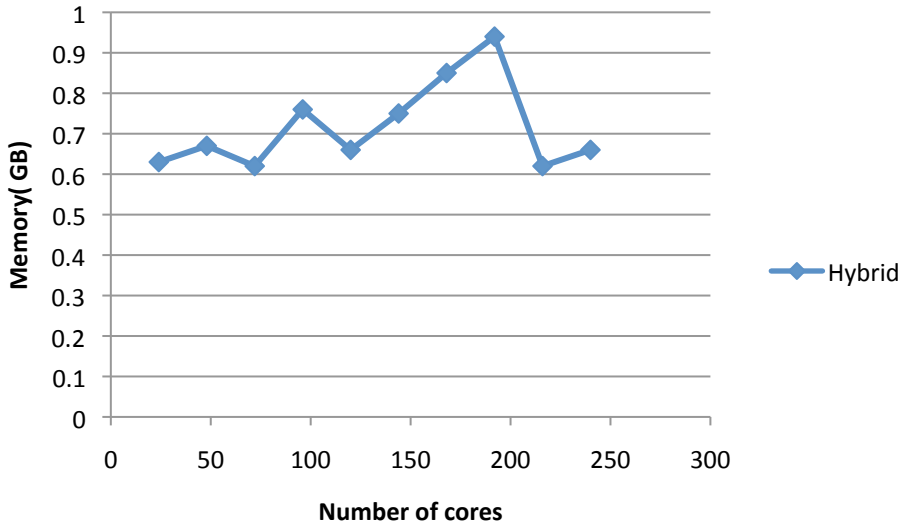
Acknowlegement

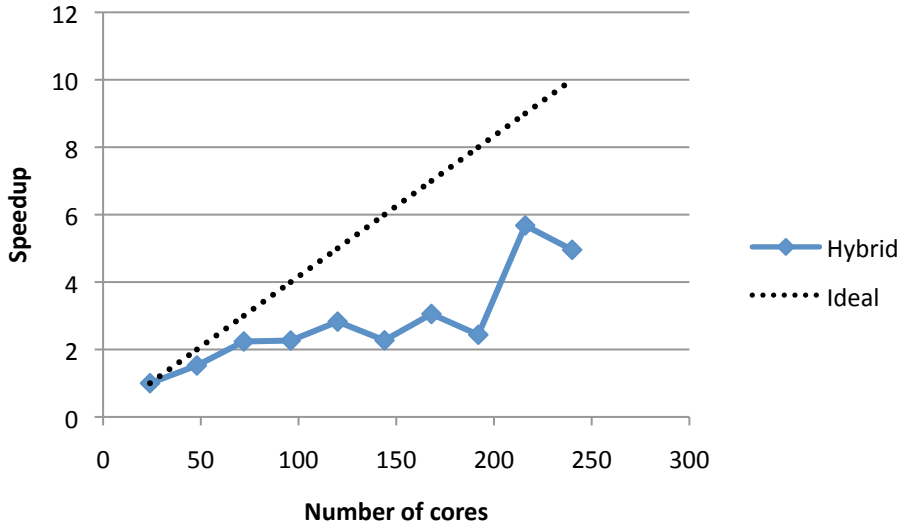
- Matthew Farrell, NERSC summer student, for providing VASP scaling tests data
- NERSC users who provided the test cases for VASP and G09
  - dag, ching, maxhutch,
  - gpzhang

Backup slides

VASP parallel scaling and memory usage on Hopper (Hybrid job)







Test case: B.hR105
105 atoms, 218 bands, hybrid calculation

VASP memory usage (test case: 660 atom system)