1. Technical Summary

iForge is NCSA’s premiere HPC resource for Private Sector Partners. iForge features four distinct hardware platforms, each configured for different computational needs. Below is a technical summary:

<table>
<thead>
<tr>
<th>CPU</th>
<th>Intel &quot;Haswell&quot; Xeon E5 2680 v3</th>
<th>Intel &quot;Ivy Bridge&quot; Xeon E5 2680 v2</th>
<th>AMD “Abu Dhabi” Opteron 6380</th>
<th>Intel “Haswell-EX” Xeon E7 8890 v3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12-core CPU 2.9 GHz</td>
<td>10-core CPU 3.1 GHz</td>
<td>8-core CPU 3.4 GHz</td>
<td>18-core CPU 2.9 GHz</td>
</tr>
<tr>
<td></td>
<td>30 MB L3 Cache</td>
<td>25 MB L3 Cache</td>
<td>16 MB L3 Cache</td>
<td>45 MB L3 Cache</td>
</tr>
<tr>
<td>Cores/Node</td>
<td>12</td>
<td>10</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>Total Nodes</td>
<td>80</td>
<td>56</td>
<td>16</td>
<td>72</td>
</tr>
<tr>
<td>Total Cores</td>
<td>1,920</td>
<td>1,120</td>
<td>512</td>
<td>144</td>
</tr>
<tr>
<td>Memory/Node</td>
<td>64 GB, 2133 MHz</td>
<td>256 GB, 1866 MHz</td>
<td>256 GB, 1600 MHz</td>
<td>512 GB, 1600 MHz</td>
</tr>
<tr>
<td>Queue</td>
<td>“normal”</td>
<td>“big_mem”</td>
<td>“amd”</td>
<td>“super_mem”</td>
</tr>
<tr>
<td>Storage</td>
<td>700 TB on network filesystem (IBM GPFS)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interconnect</td>
<td>QDR InfiniBand, 40 Gb/sec bandwidth, 100 ns latency</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td>Red Hat Enterprise Linux 6.7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Frequency based on all-cores utilization for non-AVX compiled applications. Actual CPU frequencies depend on CPU utilization and amount of vectorization.

2. Access & Support

Connecting to iForge is easy. Simply point your SSH client to iforge.ncsa.illinois.edu and enter your NCSA Kerberos username and password.

To create new accounts, an industry partner’s Principal Investigator (or “PI”) should fill out the form at: http://www.ncsa.illinois.edu/UserInfo/Allocations/AddUserForm.html

To modify or remove accounts, a PI should email: allocations@ncsa.illinois.edu

For general support, NCSA maintains a 24/7/365 help desk. You can contact the help desk by email at help@ncsa.illinois.edu or by phone at 1+(217) 244-0709.

For application support, please contact:

koric@illinois.edu / qiyuelu1@illinois.edu  
taha@illinois.edu / vcmadhu@illinois.edu  
lmainzer@illinois.edu  

FEA-related  
CFD-related  
Bioinformatics-related
3. Directories

By default, all users are allocated a /home directory with a 50 GB disk quota. In addition, NCSA partners are allocated 50 TB of /project space. Additional directory information includes:

Project Space: /projects/$NAME [GPFS-based, 50 terabytes of disk space, no purging]

Scratch: /scratch/users/$LOGIN [GPFS-based, no disk quota, purged periodically]

Note: iForge compute nodes are “stateless,” meaning they have no local disks and no SWAP space. Instead, all iForge compute nodes share access to the large GPFS-based filesystem.

4. Compilers & MPI

By default, iForge loads Intel Compiler 12 (intel/12.0.4) and MVAPICH2 (mvapich2-1.7rc1-intel-12.0.4) for MPI. For highest performance on MPI-based codes, however, we recommend using the latest Intel Compiler (intel/15.0) and MPI (intel/impi-5.0.1) suite.

To check the current modules in your profile, run: module list
To check available modules on iForge, run: module avail
To check the information on this module, run (e.g.): module show intel/15.0
To switch between modules, run (e.g.):
module unload intel/12.0.4
module load intel/15.0

These commands switch from Intel Compiler 12 to Intel Compiler 15. Or, you can also use
module swap intel/12.0.4 intel/15.0
to switch between them. To make sure it works, run:
module list
to check again after switching and incorporate these commands into job submission scripts as well.

Same procedure can be used on other module load/unload pairs as well.
To check available sub-commands and arguments, run:
module -H

Here is an example of switching compiler and MPI modules:

After logging in, in your profile, the default modules are used. With the command
module list
, you can find that
intel/12.0.4 and mvapich2-1.7rc1-intel-12.0.4 are currently active compiler and MPI modules.
Assuming there is a need to test users’ code with newer Intel compiler/MPI suite, e.g., \texttt{intel/15.0} and \texttt{intel/impi-5.0.1}, we have several steps to do before new modules being active:

1) Switch the compiler module (MKL will be switched automatically at this step)
   \begin{verbatim}
   module unload intel/12.0.4
   module load intel/15.0
   \end{verbatim}

2) Switch the MPI module
   \begin{verbatim}
   module unload mvapich2-1.7rc1-intel-12.0.4
   module load intel/impi-5.0.1
   \end{verbatim}

3) Check current modules with and built the code.
   \begin{verbatim}
   module list
   \end{verbatim}

4) Put these four lines in your batch script before the execution line.
   \begin{verbatim}
   module unload intel/12.0.4
   module load intel/15.0
   module unload mvapich2-1.7rc1-intel-12.0.4
   module load intel/impi-5.0.1
   \end{verbatim}

In general, if users should use MPI modules with an Intel compiler other than the default version, they need to switch the compiler module first and then the appropriate MPI module.

Additional information about compilers and MPI versions on iForge includes:

**Serial Code:**

Intel FORTRAN: \texttt{ifort foo.f}

Intel C: \texttt{icc foo.c}

Intel C++: \texttt{icpc foo.cpp}

**MPI: Intel-MPI**

\texttt{Switch:}
\begin{verbatim}
module unload intel/12.0.4
module load intel/15.0
module unload mvapich2-1.7rc1-intel-12.0.4
module load intel/impi-5.0.1
\end{verbatim}

\texttt{Wrappers:}
FORTAN 77/90: \texttt{mpiifort foo.f}

C: \texttt{mpiicc foo.c}

C++: \texttt{mpiicpc foo.cpp}

The mpirun launcher line should be:
\begin{verbatim}
mpirun --rsh=ssh -np \{NP\} -hostfile \{PBS_NODEFILE\} ./your.exe
\end{verbatim}

**MPI: Mvapich2**

\texttt{[loaded by default]}

\texttt{Wrappers:}
FORTRAN 77: `mpif77 foo.f`
FORTRAN 90: `mpif90 foo.f`
C: `mpicc foo.c`
C++: `mpicxx foo.cpp`
The mpirun launcher line should be:
```
mpirun_rsh -ssh -np ${NP} -hostfile ${PBS_NODEFILE} ./your.exe
```

**MPI: OpenMPI**

Make the following switch if you are using inter/15.0 compiler:
```
module unload mvapich2-1.7rc1-intel-12.0.4
module load openmpi-1.8.4-intel-15.0
```
Wrappers (versions may vary):

FORTRAN 77/90: `mpif77/mpif90 foo.f` (for older versions)
FORTRAN 77/90: `mpifort foo.f` (for v1.7 and above)
C: `mpicc foo.c`
C++: `mpicxx foo.cpp`
The mpirun launcher line should be:
```
mpirun -np ${NP} -hostfile ${PBS_NODEFILE} --bind-to-core ./your.exe
```
It is observed that the option `--bind-to-core` enforces proper CPU affinity on iForge.

**OpenMP:**

Fortan 77/90: `ifort -openmp foo.f`
C: `icc -openmp foo.c`
C++: `icpc -openmp foo.cpp`

*Note: For other compiler and linker options, please contact: koric@illinois.edu / qiyuelul@illinois.edu*
5. Libraries

By default, iForge uses MKL for accelerated math functions, which are loaded automatically when the Intel compiler is loaded. To check the absolute path of the current MKL, run:

```bash
echo $MKL_HOME
```
for intel/12.0.4.

Run

```bash
echo $MKLROOT
```
for intel/15.0

Additional information about libraries on iForge includes:

**Intel MKL**

To choose the libraries, Intel MKL link line advisor is recommended:


Figure 1 below is an example of choosing libraries by Intel MKL link line advisor. This is a FORTRAN code, dealing with integers less than 32-bits, without using OpenMP, linking libraries dynamically, using ScalAPACK library, and with a BLAS FORTRAN 95 interface.

**Note:** `-LLIB_PATH` should be specified before these libraries. Suggested libraries in the figure will appear in the link line as follows:

```bash
-L{MKLROOT}/lib/intel64 -lmkl_blas95_lp64
-llmkl_scalapack_lp64
-llmkl_blacs_intelmpi_lp64 -lpthread -lm
```

And the compiler option:

```bash
-I$(MKLROOT)/include/intel64/1p64 -mkl=sequential
```

The 'ldd' command can help to check whether the library is linked dynamically.

```bash
ldd code_name | grep lib_name
```

If static linking is chosen, grouping symbols `-Wl,--start-group` and `-Wl,--end-group` might not be recognized if the code has its own configuration tools. In general, the advisor tool provides the name of libraries needed but how to link them in the MAKEFILE might vary slightly among different codes.

**Note:** Other math libraries are available in `/usr/apps/math` and accessible by modules. And HDF (Hierarchical Data Format) is also available on iForge.
Hierarchical Data Format (HDF) [accessible by the following modules]

- module load hdf/hdf4-2.6
- module load hdf/hdf4-2.6-noncdfs
- module load hdf/hdf5-1.8.7
- module load hdf/hdf5-1.8.7-shared
- module load hdf/phdf5-1.8.7

You can also use Intel’s “Link Line Advisor” to see what libraries are recommended for a particular use case. For more, visit: [https://software.intel.com/en-us/articles/intel-mkl-link-line-advisor](https://software.intel.com/en-us/articles/intel-mkl-link-line-advisor)

![Figure 1: An example of choosing libraries by Intel MKL link line advisor](image)
6. Queues & Batch Runs

iForge uses PBS Torque with Moab Workload Manager. All jobs have a 30-day wall-clock limit.

iForge features several queues that you can send your jobs to:

```
** Welcome to iForge **

NCSA's Premiere HPC Resource for Private Sector Partners

For support, contact help@ncsa.illinois.edu or +1 (217)244-0709

Unauthorized access or use is prohibited

Available Queues:

*normal* - 24 Intel "Haswell" cores, 64 GB of RAM per node (1920 cores total)
*bigrmem* - 20 Intel "Ivy Bridge" cores, 256 GB of RAM per node (1120 cores total)
*super_mem* - 72 Intel "Haswell" cores, up to 5 TB of RAM per node (144 cores total)
*amd* - 32 AMD "Abu Dhabi" cores, 256 GB of RAM per node (512 cores total)
*debug* - For software debugging, 30 minute wall-clock limit
```

Sample batch scripts for MVAPICH2, Intel-MPI, and OpenMP are in `/usr/apps/csm/SCRIPTS`. You can run interactive batch jobs by prefixing the resource line of a PBS script with `qsub -I`.

* e.g. requesting 2 nodes, each with 64 GB of RAM, for 30 minutes (i.e. “normal” queue):
  
  ```
  qsub -I -V -l walltime=00:30:00,nodes=2:ppn=24
  ```

* e.g. requesting 8 nodes, each with 256 GB of RAM, for 45 minutes (i.e. “big_mem” queue):
  
  ```
  qsub -I -V -l walltime=00:45:00,nodes=8:ppn=20 -q big_mem
  ```