ESIF-HPC-2 Benchmark Instructions

Contents

1 Objective of ESIF-HPC-2 Benchmarking

The purpose of the ESIF-HPC-2 benchmarking effort is to enable evaluation of the performance of NREL HPC applications on the high performance computing and file systems proposed by the Offeror in response to the ESIF-HPC-2 RFP.

A summary of the benchmarks is provided in the benchmarking section of Attachment 1: Technical Specifications of the ESIF-HPC-2 RFP. This document gives specific instructions to Offerors on how the benchmarks are to be run, what information should be returned to NREL, and other requirements.

Offerors who respond only to the DSS portion of the system or only to the HPC portion of the system only need to run the benchmarks associated with the respective system as described in Section 7 of Attachment 1: Technical Specifications.

2 How Benchmarks are Supplied

The benchmarks used for the ESIF-HPC-2 procurement are available from the NREL HPC Benchmarking website: https://hpc.nrel.gov/benchmarks. This site provides compressed archive files for each of the individual benchmarks. Each tar file includes

- specific instructions on building and running the benchmark
- reference output
- example run scripts used to generate the reference output
- a validation script (where relevant)
- either source code or information on how to acquire the necessary materials, including a version requirement.

The Offeror is responsible for acquiring licensed applications from the application vendor.

3 Ownership and Dissemination of Results

Results, including performance, run configurations, output files and system logging data specific and relevant to benchmark runs, reporting documents, and metrics derived from any of these will be considered sole property of NREL. Results should not be released publicly without prior written consent from NREL's Legal office.

4 Right to Request Benchmark Rerun Before Award

NREL reserves the right to require that an Offeror repeat any or all of the ESIF-HPC-2 benchmark runs prior to final award if questions arise from the results provided in the Offeror's response to this RFP.

5 Benchmark Rules

5.1 Core and Memory Subscription

Unless otherwise specified, for benchmarks with MPI or thread parallelism, cores within a node should be fully subscribed. If memory limitations prevent a benchmark from running in a fully subscribed configuration, the benchmark should be run using the minimum number of nodes required to allow the benchmark to run. In this situation the cores used may be spread evenly across participating nodes. Reasons for and details of under-subscription should be explained in the Offeror's benchmark report.

Any explicit control of rank and/or thread mapping by the Offeror must be noted in the Offeror's benchmark report.

5.2 Benchmark System(s)

The system used by the Offeror to run the ESIF-HPC-2 benchmarks should be as architecturally close as possible to the offered system.

Information about the benchmark system(s) must be provided in the "Benchmark Systems" reporting worksheet and a description of the principal features of the system should be provided in the Offeror's benchmark report.

All benchmarks should be run on the same system where possible. If a benchmark is run on a different system, a clear explanation of the reason must be provided.

5.3 Benchmark Results Projection

In the event the Offeror's benchmark system differs from the Offered system, all timings from the benchmark system will need to be projected to the Offered system. The benchmark report must contain a clear description of the differences between the systems, and a description of the performance model and projection method. The performance model must be described in sufficient detail for evaluation.

For trivial scaling algorithms (*e.g.*, simple scalar multiplication), the benchmark reporting spreadsheet should include the scaling model as embedded equations. The derivation of all parameters (*e.g.*, scalar constants) must be explained in the benchmark report. For more complex transformations, calculations of all scaling factors should be provided in the benchmark report.

Benchmarks will be run at Acceptance, and success will depend on reproduction of the projected performance and throughput.

5.4 As-Is and Optimized Benchmark Results

The Offeror shall supply benchmark results from "as-is" and, optionally, "optimized" code.

"As is" results are intended to be a measure of an Offeror's system's performance without optimization work, and must include only the minimal set of modifications to the code, provided either in the benchmark suite or by a vendor, needed to produce correct results on the benchmarked system.

"Optimized" results are intended to showcase the Offered system's capabilities that could be achieved by skilled developers. The optimized results may be generated from source code with modifications of the types described in section 5.5.

5.5 Benchmark Code Changes and Optimization

In addition to compiler flags and run-time settings, the optimized results may be generated from source code with modifications with the following restrictions:

- The Offeror may not change the floating-point precision used.
- The Offeror may not use assembly-level recoding.
- Source code modification should be done either in the original source language or via addition of compiler directives or pragmas.
- Modified source code must pass the individual benchmark's validation criteria.

All modified source files must be provided to NREL as part of the File response with clear identification (*e.g.*, in a directory distinct from runtime results). Modifications shall be isolated and shall be enabled or disabled via conditional compilation using pre-processor directives. For example:

```
#ifdef ESIF_Optimized
   <Offeror-specific code>
#else
   <Original code>
#endif
```

The benchmark report should include a summary of modifications made, as well as the reasons for making them.

5.6 Compilers and Libraries Used for Benchmarking

Any software tool or package used to generate reported results must be offered and installed on the delivered system prior to acceptance testing.

5.7 Use of Optimized Third-party Binaries

For "as-is" reporting, the Offeror may not run a third-party binary. For kernel benchmarks only (HPL, STREAM, SHOC, IOR, mdtest, and Bonnie++), a third-party binary may be used. Results should be reported in the "Optimized" sections of the reporting spreadsheet, and be clearly delineated from results arising from source code optimization reported in the File response. Any results from a third-party binary will be considered a commitment to be reproduced during acceptance testing.

6 Reporting of Benchmark Results

6.1 Reporting Worksheet Structure

The workbook contains one worksheet per benchmark application, as well as a sheet for reporting details of each system used to generate benchmark data and a sheet for the sustained throughput metric. Reference data where available from NREL's Peregrine system is contained in the individual benchmark sheets. The sheets are divided into 4 sections by black row/column delimiters. Test and Offered system sections are on the left and right side of the vertical delimiter, respectively; "As-is" and "Optimized" code reporting sections are above and below the horizontal delimiter, respectively.

Performance tests. Additional instructions for identifying the performance data that is to be entered into the reporting spreadsheet are supplied as part of the benchmark distribution.

Throughput tests. To enable calculation of the sustained throughput metric, which is a proxy for NREL's workload, two benchmarks require additional runs beyond those used to assess performance: HPGMG-FV and Nalu. The details of these tests are contained in the individual benchmark descriptions below, and in the benchmark-specific instructions found in the README.md file of the distributions. Reporting comprises documentation of the run configurations used (each unique configuration on a separate row, with number of instances of each configurations run entered as "# Instances"), and a single aggregate number. These numbers, together with time from the single-node VASP job, will be used to create a final throughput metric for the Offered system, which will be used to assess the multiplier of increased throughput over NREL's current system.

6.2 Terms

The following terms apply.

a) "Test system" is assumed to be the system upon which Offeror benchmark data has been generated. This system could be full-scale with respect to the Offering, a smaller representative of the Offered node types, or another system from which performance projected for the Offered system may be derived.

b) "Offered system" is the system being offered to NREL. Performance figures reported in sections for the Offered system may be directly measured or projected from the Test system. Regardless, these results are taken as targets that the final installed system must meet. If these results are projected, the algorithm used for projection must be explained in the benchmark report or coded into the spreadsheet.

c) "As is," to contain performance data obtained using unmodified code or ISV applications, or code minimally modified for porting, per section (f) below.

d) "Optimized," to contain data measured using code that has been optimized to achieve optimal performance on the offered system, as described in section 5.4 of this document.

e) "Projected," to contain data for the Offered system, which Offeror commits to achieve during acceptance testing.

f) “Third-party binary,” a complete binary executable, or such an executable built from an object file containing the main logic, of a kernel benchmark for which (a) the executable itself or the object code is provided by an entity other than the Offeror, and (b) no source code can be provided that could be compiled by NREL to the same object file or executable.

6.3 Mandatory elements of response

Results for the Offered system with “as-is” code must be returned. It is assumed that if the benchmark system is not the same as the system that would be delivered (i.e., different processor SKUs, different memory configurations or capacities, etc.), that the Offered system results will comprise a projection from benchmark results; in this case, results for the Test system that form the basis for the projection must be included as well.

The benchmarking response comprises three key components for each benchmark.

6.3.1 File response

The file response includes batch submission scripts, and all scripts used for test automation; environment dumps; standard output and error streams from builds and runs; all log files; and, any artifacts specific to a benchmark as named in that benchmark’s README.md file.

This shall be returned as a compressed tarfile, with the file structure clearly delineating which results are associated with which benchmark (*e.g.*, separate directories for each benchmark, separate sub-directories for “as-is” and “optimized” runs, and separate sub-sub-directories for each run configuration).

The compressed tarfile(s) shall be returned to NREL on ISO 9660 compliant DVD-ROM, USB flash drive or USB hard drive if larger than 20 megabytes in size. Alternatively, compressed tar files that are smaller than 20 megabytes may be submitted to the NREL Contracts Administrator via email attachment rather than on physical media.

6.3.2 Spreadsheet response

The Microsoft Excel workbook provided by NREL should contain all the requested timing and performance results, as detailed in the instructions for each benchmark. The workbook with Offeror’s results shall be returned as an Appendix to the Technical Volume of the Offeror’s response.

6.3.3 Text response (Benchmark report)

This should be returned as an Appendix to the Technical Volume of the Offeror’s response.

This component includes all descriptions required by the general benchmark rules and descriptive details described in each benchmark’s instructions. The response should include an end-to-end description of the environment in which each benchmark was run, including:

- client and server system configurations, including
  - node and processor counts,
  - processor models
  - memory size and speed for both volatile (DRAM) and non-volatile storage,
  - OS (names and versions);

    - client and server sysctl settings,
    - driver options,
    - network interface options, and
- filesystem configuration options (e.g., striping, caching);
- storage used for filesystems and storage configuration;
- network fabric used to connect servers, clients, and storage, including network configuration settings;
- makefiles, compiler name and version, compiler options, and libraries used to build benchmarks;
- process and thread pinning information; and,
- command lines used for each benchmark run, if not evident in the File response.

If performance projections have been made, the benchmark report should describe the projection models and methodology, in sufficient detail for evaluation.

Additional content might include considerations for a benchmark that the Offeror feels would benefit NREL in interpreting the returned data, or tabular information outside of the Spreadsheet response structure correlating to benchmark runs.

Guidance on specific information and files to return can be found below for each benchmark, and in the associated README.md file in the benchmark distribution.

7 NREL Benchmark Suite

7.1 Application benchmarks

7.1.1 VASP

Minimum # run configurations: 3 each for bench1 and bench2
Node types: Standard
Reported metric: runtime in seconds

The VASP benchmark measures strong scaling behavior on two common workloads at NREL.

The VASP benchmark consists of two cases. Bench1 represents high-accuracy band structure calculations for semiconductors and has 3 parts: a GGA calculation to be run on the following core counts: 128, 256 and 320, an HSE calculation to be run on 128, 256 and 320 cores and a GW calculation to be run on 16, 32 and 64 cores. Bench2 represents large unit cell calculations for surface catalysis. It should be run on 32, 64 and 128 cores.

Bench2 requires an additional run on a single *node*. The timing from this test will be used to calculate a throughput metric representing NREL’s high-throughput VASP workload. Parallel configuration within the node may be optimized by the Offeror to minimize time-to-solution.

Enter runtimes from the “time” utility in the reporting spreadsheet.

7.1.2 Nalu

Minimum # run configurations: 10 + throughput

Node types: Standard
Reported metric: runtime from log data (scaling), timesteps/s (throughput)

Nalu is a modern computational fluid dynamics code designed for massive parallelism on unstructured meshes. We are interested in both single job scalability and throughput on the standard compute nodes only.

For strong scaling, the Nalu benchmark uses two input mesh sizes, designated here simply as “256” and “512”. Runs on 2, 128, 256, 512 and 1024 nodes are required for the “256” mesh. Results from runs on 64, 128, 256, 512 and 1024 nodes are required for the “512” mesh. Enter wallclock times as reported under the “avg” column in logfiles in the reporting spreadsheet.

The throughput test uses the 512 mesh case only. All standard nodes must be subscribed, the jobs must start at the same time, and a single filesystem must be used for all I/O.

Throughput to be reported is calculated as the **sum** of timesteps/s achieved over **up to** 100 simultaneous instances of Nalu. For each instance, timesteps/s is simply the number of timesteps Nalu runs (a constant 2500, as configured in the distributed files) divided by wallclock time as reported under the “avg” column in logfiles. Run configurations should be chosen so that the chosen number of instances fill the complete set of offered standard compute nodes. If the configurations (# nodes, # ranks, # threads, etc.) of the 100 instances aren’t identical, the details of each distinct configuration should be entered into the reporting spreadsheet and lines should be added if more than 3 distinct configurations are used.

7.1.3 HPGMG-FV

Minimum # run configurations: 8 + throughput
Node types: Standard
Reported metric: degrees of freedom/s

The Finite Volume implementation of the HPGMG benchmark reflects workflow common to combustion applications, and more generally, hybrid parallel codes with balanced requirements for system resources.

For scaling, benchmark results should be reported for 64, 128, 256, 512 and 1024 MPI ranks as well as for all, ½, and ¼ of the standard nodes in the Offered system. The Spreadsheet response should contain the average of first-line Degrees-of-Freedom per second (DOF/s) reported by the application from 3 separate runs.

For throughput, the Standard node offering should be filled with 8-12 concurrent HPGMG runs. The specific number may be chosen to best fit the offered system. The HPGMG-FV benchmark runs for a fixed time, and reports performance as degrees of freedom per second. The **sum** of the DOF/s values from these concurrent jobs should be reported in the Spreadsheet response. If the configurations (# nodes, # ranks, # threads, etc.) of the 8-12 instances aren’t identical, the details of each configuration should be entered into the reporting spreadsheet and lines should be added if more than 3 distinct configurations are used.

7.1.4 LAMMPS

Minimum # run configurations: 5 Standard, 4 Big Memory, 2 validation
Node types: Standard, Big Memory
Reported metric: log data: # steps, loop time (s), simulated timesteps/s

LAMMPS represents classical atomistic molecular dynamics work at NREL. The LAMMPS benchmark consists of three cases. The benchmark includes a single template unit cell, from which three actual unit cells of increasing size are constructed automatically for testing weak scaling.

For standard nodes, the small benchmark case should be run on 1, 4, 16 and 64 nodes. The large and medium cases should be run on 4, 16, 64 and 128 standard nodes. For big memory nodes, all 3 cases should be run on 1, 2, 4 and 8 nodes.

Validation must be done on two separate runs of 16 and 64 *MPI ranks*, as described in README.md.

The simulation rate (timesteps/sec) reported in the LAMMPS logfile output is the performance metric to be entered into the reporting sheet. In addition, the number of simulated timesteps taken and the "Loop time," should be entered.

7.1.5 HiBench

Minimum # run configurations: 10
Node types: Big Memory
Reported metric: throughput, throughput/node

HiBench is a big data benchmark for Hadoop and Spark workloads. Results from a 5-node cluster are required, but Offeror's may optionally provide additional results for larger clusters.

Performance results for the "bigdata" data profile for the Wordcount, Sort, Bayes, K-Means and Enhanced Hadoop filesystem I/O (DFSIOE) tests are required.

7.1.6 Gaussian

Minimum # run configurations per node type: 2
Node types: Standard, Big Memory
Reported metric: runtime (s)

The Gaussian16 benchmark consists of two cases: a single-node, multi-threaded job and a two-node job that uses both distributed and shared memory parallelism. In both cases, each node hosts one process and all cores should be used by threads.

The Offeror may make changes to input files only to affect how Gaussian uses memory, local storage and parallelism.

Enter runtimes from the "time" utility in the reporting spreadsheet.

7.2 Micro-benchmarks

7.2.1 Intel MPI Benchmarks (IMB)

Minimum # run configurations per node type: 2 (1 rank/node and packed ranks)
Node types: Standard, Big Memory
Reported metric: reported times (microseconds) or bandwidths (MB/s)

IMB is used to measure application-level latency and bandwidth, particularly over a high-speed interconnect, associated with a wide variety of MPI communication patterns with respect to message size. This benchmark includes only a selection of possible tests, namely: PingPong, Sendrecv, Exchange, Barrier, Uniband, Biband, Allgather, Allreduce, and Alltoall.

Each test except PingPong and Barrier includes running with a single MPI rank per node, and with a packed node (i.e., 1 rank per physical core), using 4 message sizes (0, 65536, 524288, and 4194304 bytes). Where node memory limits prohibit these message sizes (e.g., collectives), the Offeror should include as many of the above message sizes as possible, and augment data with the largest message size that will fit into the Offered memory.

The PingPong test requires only 0-byte and 0.5 MB message sizes, placed either on two adjacent nodes in the network topology, or on two nodes as far apart as possible in that topology. The PingPong test involves only 2 nodes, but all other tests must be run on 2, 64, 128, 256, or 512 nodes.

Reported data need only be transcribed directly from the IMB output to the benchmark reporting sheet.

7.2.2 STREAM

Minimum # run configurations per node type: 6 (3 subscriptions, 2 capacities)
Node types: Standard, Big Memory
Reported metric: Triad bandwidth (GB/s)

The STREAM Triad benchmark is used to measure sustained memory bandwidth. Results should be reported for standard nodes, big memory nodes and DAV nodes using (a) a single thread, (b) one thread on each physical core, and (c) the minimum number of threads needed to achieve maximum bandwidth.

Enter the STREAM Triad rate (GB/s) from the stdout file in the benchmark reporting spreadsheet.

7.2.3 SHOC

Minimum # run configurations: 2
Node types: DAV
Reported metric: bandwidths (GB/s)

The SHOC benchmark is used to measure bus transfer rates and memory bandwidth on GPUs. The Offeror is required to report the BusSpeedDownload and BusSpeedReadback from the level 0 benchmark and Triad from level 1 benchmark using “Run size” 4, which is intended for HPC-focused GPUs.

The benchmark should be run for a single GPU and for both GPUs in a single DAV node.

7.2.4 HPL

Minimum # run configurations: 12
Node types: Standard, Big Memory
Reported metric: performance (Gflops) and time

For Standard nodes, benchmark results should be provided for 1, 32, 128, 256, 512, 1024, N/2 and N *nodes*, where N is the number of standard nodes Offered, with every core hosting a thread of execution. On Big Memory nodes, benchmark results should be provided for 1, 4, 16 and 32 nodes.

The Time and Gflops data from HPL stdout should be transcribed directly into the reporting spreadsheet.

7.3 I/O Benchmarks

The I/O benchmarks should be run for the proposed file systems from the standard, big memory and DAV nodes.

Modifications to the I/O benchmarks are only permissible to enable correct execution on the target platform. Any modifications must be fully documented. Changes related to optimization and tuning must be practical for production utilization of the proposed file system. Tuning hints that can be controlled by users are allowed but no optimizations that require super-user privilege and which are not part of the proposed configuration of the file system for production use are allowed.

In addition to the output files for each run, the Offeror must provide an end-to-end description of the environment in which the benchmark was run. This should include

- Client and server sysctl settings
- Driver options
- Network interface options
- File system configuration options
- Compiler name and version, compiler options and libraries used to build benchmarks.

7.3.1 IOR

Node types: Standard, Big Memory, DAV
Reported metric: bandwidths

The intent of this benchmark is to measure streaming and random 4k I/O performance of the proposed file systems and storage.

The ESIF-HPC-2 benchmark includes tests of both random and streaming performance of each node type to each globally accessible file system (PFS and HFS) as well as the performance of local storage on the big memory and DAV nodes. Streaming performance must be reported using 4MB transfers but Offerors may additionally provide streaming performance for the optimal transfer size for the proposed system.

In addition to single node performance measurements, we include a test that should use the number of standard nodes that yields the peak performance of each globally accessible file system.

Offerors are not permitted to use optimizations related to client side caching.

7.3.2 mdtest

Minimum # run configurations per node type: 10
Node types: Standard, Big Memory, DAV
Reported metric: creation, removal, and stat rates (files/s)

The intent of this benchmark is to measure the performance of file metadata operations on each proposed globally accessible file system. For each node type, each test should be run

- with (a) a single MPI process on a single node, (b) the optimal number of MPI processes on a single compute node, (c) the minimal number of MPI processes on multiple nodes that achieves the peak performance on the Offered system, and (d) the optimal number of MPI ranks over all nodes; and,
- with 1 file in 1 directory, 1048576 files in 1 directory, and 1048576 files in multiple directories.

The file creation, stat and removal rates from standard output should be reported in the benchmark reporting spreadsheet.

7.3.3 Bonnie++

Minimum # run configurations per node type: 3 (service), 6 (login), 9 (other)
Node types: Standard, Big Memory, DAV, login, service
Reported metric: sequential read, sequential write, and sequential re-write rates (kB/s)

The intent of this benchmark is to measure the performance of local and network file systems. It should be run for each network file system on each node type (except HFS from service nodes), as well as for the local file system on the Standard, Big Memory, and DAV nodes.

Each node type requires 3 configurations: a single core active, half the cores active, and all cores active. In this case, a “core” implies a physical core.

The sequential write (create), sequential rewrite, and sequential read rates should be reported in the benchmark reporting spreadsheet.