Development and launch of parallel C-DVMH and Fortran-DVMH programs on hybrid supercomputer NKS-30T

DVM System

December 17, 2016

To compile and launch parallel DVMH-programs on supercomputer NKS-30T it is necessary:

1. 1. To initiate DVM system version installed on supercomputer NKS-30T. To perform the command:

   ```
   /ifs/apps/DVM/bin/dvminit
   ```

   As a result of given command performing, `dvm` and `usr.par` files necessary for DVMH-program compilation and execution will be copied to the current directory of a user, and also the symbolic reference ~`dvm_current` to the installed DVM system version will be created. Demonstrational DVMH-programs are available in the directory ~`dvm_current/dvm_sys/demo` (files with `*.cdv` and `*.fdv` extensions).

2. Compilation of DVMH-programs.

   (a) Compilation of C-DVMH program is performed by the command:

   ```
   ./dvm c [-noH] [-noCuda] <program name>.<cdvm-ext>,
   ```

   where `<cdvm-ext> = cdv|c|h`
   For example,

   ```
   ./dvm c jac3d_perf.cdv
   ```

   The result of the command is ready-for-run parallel program in MPI/OpenMP/CUDA model with the name `jac3d_perf`.

   ```
   ./dvm c -noCuda jac3d_perf.cdv
   ```

   The result of the command is ready-for-run parallel program in MPI/OpenMP model with the name `jac3d_perf`.

   ```
   ./dvm c -noH jac3d_perf.cdv
   ```

   The result of the command is ready-for-run parallel program in MPI model with the name `jac3d_perf`.

   (b) Compilation of Fortran-DVMH program is performed by the command:

   ```
   ./dvm c [-noH] [-noCuda] <program name>.<fdvm-ext>,
   ```

   where `<fdvm-ext> = fdv|f|ftn|for|f90|f95|f03`
   For example,

   ```
   ./dvm f jac3d_perf.fdv
   ```

   The result of the command is ready-for-run parallel program in MPI/OpenMP/CUDA model with the name `jac3d_perf`.

   ```
   ./dvm f -noCuda jac3d_perf.fdv
   ```

   The result of the command is ready-for-run parallel program in MPI/OpenMP model with the name `jac3d_perf`.

   ```
   ./dvm f -noH jac3d_perf.fdv
   ```

   The result of the command is ready-for-run parallel program in MPI model with the name `jac3d_perf`.
   Notes.

i. DVMH-program can have standard extension (for example, `*.f90` or `*.c`).

ii. For Fortran-programs the record format (free or fixed) is determined automatically by the program extension. For the programs having extensions `*.f90`, `*.f95`, `*.f03`, the free format is used, and for all remaining – the fixed one. It is possible to specify a format using compiler options: `-f90` (free) and `-FI` (fixed). For example,

```
./dvm f -f90 jac.fdv
```

3. Launching DVMH-program.

Before DVMH program start it is necessary to set the following environment variables in `dvm` script (details see in documentation):

(a) The compute nodes of NKS-30T supercomputer where DVMH-program will be executed. `queuemode` variable:

`export queuemode='<queue name>'`

Default value is `'SL_q'`.

Brief information about the nodes of NKS-30T supercomputer is available by the command `pbsinfo`.

To launch a parallel program in MPI or MPI/OpenMP model it is recommended to use `g6_q` or `G7_q` queues:

`export queuemode='g6_q'`

or

`export queuemode='G7_q'`

To launch a parallel program in MPI/OpenMP/CUDA model it is recommended to use `SL_q` queue:

`export queuemode='SL_q'`

Full information about compute nodes of NKS-30T supercomputer is available on the site.

(b) Maximum runtime of the program. `maxtime` variable:

`export maxtime=00:30:00`

Default value is 30 minutes. For more long calculations (for example, 1 hour) it is necessary to set:

`export maxtime=01:00:00`

(c) A number of MPI processes launched on a node. `DVMH_PPN` variable:

`export DVMH_PPN='2'`

(d) A number of the threads used by each MPI process. `DVMH_NUM_THREADS` variable:

`export DVMH_NUM_THREADS='4'`

Note. Value of this variable is ignored for the programs compiled with the option `-noH`.

(e) For each MPI process it is necessary to specify a number of used graphic accelerators. `DVMH_NUM_CUDAS` variable:

`export DVMH_NUM_CUDAS='3'`

Note. Value of this variable is ignored for the programs compiled with option `-noH` or `-noCuda`.

Recommended configurations:

When specifying the configuration:

```
export queuemode='SL_q'
export DVMH_PPN='3'
export DVMH_NUM_THREADS='0'
export DVMH_NUM_CUDAS='1'
```

three MPI processes will be launched on each compute node of NKS-30T hybrid cluster in `SL_q` queue and each of them will use its own graphic accelerator for computations.

When specifying the configuration:

```
export queuemode='SL_q'
export DVMH_PPN='2'
export DVMH_NUM_THREADS='6'
export DVMH_NUM_CUDAS='0'
```

two MPI processes with 6 threads per process will be launched on each compute node of NKS-30T hybrid cluster in `SL_q` queue. Graphic accelerators won't be used.

When specifying the configuration:

```
export queuemode='SL_q'
export DVMH_PPN='12'
export DVMH_NUM_THREADS='1'
export DVMH_NUM_CUDAS ='0'
```

12 MPI processes will be launched on each compute node of NKS-30T hybrid cluster in `SL_q` queue. There won't be any job distribution between threads and graphic accelerators.

When specifying the configuration:

```
export queuemode='SL_q'
export DVMH_PPN='1'
export DVMH_NUM_THREADS='12'
export DVMH_NUM_CUDAS ='3'
```

all computing resources of a node in `SL_q` queue will be used. One MPI process will be launched on each node. This process will use 3 graphic accelerators and 12 threads.

When specifying the configuration:

```
export queuemode='SL_q'
export DVMH_PPN='2'
export DVMH_NUM_THREADS='6'
export DVMH_NUM_CUDAS ='1'
```

two MPI processes will be launched on each compute node of NKS-30T cluster in `SL_q` queue and each of them will use 6 threads and one graphic accelerator for computations.

Note.

When a program is executed simultaneously both on CPU, and on GPU it may be required to set environment variables `DVMH_CPU_PERF` and `DVMH_CUDAS_PERF` which specify the relative productivity of CPU and GPU and control the distribution of computations inside a cluster node Valu (in more detail see in documentation).

After setting of environment variables listed above, the DVMH-program can be launched by the command:

```
./dvm run <processor matrix> <program name> [<task parameters>]
```

The parameter `<processor matrix>` sets rank of a processor grid on which the task will be run (`<n1> <n2>... <nK>`). During the task execution `<n1>*<n2>*...*<nK>` MPI processes will be created. For example,

```
./dvm run 2 1 2 jac3d_perf
```

As a result of this command the task which uses four MPI processes will be created. For the configuration:

```
export DVMH_PPN='2'
export DVMH_NUM_THREADS='0'
export DVMH_NUM_CUDAS ='1'
```

two compute nodes will be provided for the task execution. Two MPI processes will be launched on each of them. Each from the processes will use one graphic accelerator. In total 4 graphic accelerators will be used for computations.

After the task is successfully queued as a result of execution of `./dvm run` command, batch system returns the task identifier (task identifier) of the form `<number>.<queue name>`. This identifier is necessary for the task status verification, modification or deleting of the task. For example, if there are not enough free resources and the program was queued, it is possible to delete it from queue (`qdel <number>.<queue name>`), to query a number of free processors (`pbsinfo`) and to restart the task on smaller number of processors.

Outputs of the task are redirected to the files `<task name>.<task number>.1/output` and `<task name>.<task number>.1/error`.

Note. `<task name>` is `<program name>.<number of processors>.<task number>`. The field `<task number>` is a number of the program starts on given number of processes in the current directory.

4. Obtaining efficiency characteristics of DVMH-program execution. Command:

```
./dvm pa <task name>.sts.gz+ <name of output file with characteristics>
```

5. The help information about other DVM commands is available as:

```
./dvm help
```

he detailed information about DVM system is on DVM site.
Questions and notes should be sent to the address: dvm@keldysh.ru.
The instruction how to work on NKS-30T hybrid supercomputer is available on the site.