Development and launch of parallel C-DVMH and Fortran-DVMH programs on supercomputer MVS-100K

DVM System

December 17, 2016

To compile and launch parallel DVMH-programs on supercomputer MVS-100K it is necessary:

1. To initiate DVM system version installed on supercomputer MVS-100K. To perform the command:

   ```
   /nethome/admdvm/MVS-100K/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] <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 model with the name `jac3d_perf`.

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

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

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

   ```
   ./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. The option `-noH` sets the mode of compiler operation. When specifying option `-noH` additional threads for job distribution inside the cluster node won't be created.
   ii. DVMH-program can have standard extension (for example, `*.f90` or `*.c`).
   iii. 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) Maximum runtime of the program. `maxtime` variable:
   `export maxtime=10`
   Default value is 30 minutes. For more long calculations (for example, 1 hour) it is necessary to set:
   `export maxtime=60`

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

```
export DVMH_PPN='2'
```

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

```
export DVMH_NUM_THREADS='4,4'
```

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

For example, for the configuration:

```
export DVMH_PPN='2'
export DVMH_NUM_THREADS='4,4'
```

on each compute node of a supercomputer MVS-100K two MPI processes will be launched, each of them will use 4 threads for job distribution.

When specifying a configuration:

```
export DVMH_PPN='1'
export DVMH_NUM_THREADS='8'
```

one MPI process using 8 threads for job distribution will be launched on each node of the cluster.

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='4,4'
```

two compute nodes will be selected for the task execution. Two MPI processes will be launched on each of them. In total 16 cores will be used for computations.

If there are not enough free resources and the program was queued (Task "<task name>.<task number>" queued successfully), it is possible to delete it from queue (mqdel <task name>.<task number), to get information about a number of free processors (mfree) 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
```

The detailed information about DVM system is on DVM.
Questions and notes should be sent to the address: dvm@keldysh.ru.
The instruction how to work on supercomputer MVS-100K is available on the site.