Quick Introduction to MD code

Ed Smith

April 5, 2011

1 Overview

The following document serves as a quick introduction and overview to the Molecular Dynamics (MD) Fortran 90 code intended for coupling. The implementation is largely based on the introductory book by Rapaport (2004) extended to run in parallel using the Message Passing Interface (MPI). An outline of the code's structure is presented in the next section followed by instructions on how to compile and run a simple case.

2 Code Structure

The code is structured in a modular way. The files are grouped into four types: setup files (`setup*`), simulation files(`simulation*`), finish files (`finish*`) and general library files (e.g. `modules.f90, linklist.f90, messenger.f90`).

- Setup - Routines to initialises a new simulation or read in an old restart file; parameters are set by the `input` file; arrays are allocated, the messenger MPI is initialised and output files are opened.
- Simulation - Routines which advances the code by calculating molecular interactive forces and advancing the molecules. A spatial decomposition is used for parallelism and message passing is used to exchange halos between the processes. Outputs are written at intervals.
- Finish - Save final state files, finish and close outputs and deallocate all arrays
- General library files - Routines used throughout the setup, simulation and final parts of the code. All MPI routines are called through a messenger file which has a parallel and serial version. Linklists manipulation as other general library routines are included in this category

Each subroutine includes comments which should help to explain it's purpose. The most important components of the code are the main calling program `md.f90`; the force calculation `simulation_compute_forces.f90` and the movement of molecules `simulation_move_particles.f90`. After extracting the files from the tarball, look at the file `md.f90` to see an overview of how the routines are called by the code.

Other important files for running a simulation include;
`input` - set the parameter which define the simulation case
`param.inc` - Specify the number of processors to sub-divide the domain into. This must agree with the number specified at run time.

3 Building and Running

3.1 The code

The code is contained in the tarball '110405_MD_dCSE.tar'. It should include the following files
modules.f90 functions_lib.f90 linklist.f90 molecular_properties.f90 external_forces.f90 messenger.serial.f90 parallel_io.serial.f90 messenger.MPI.f90 parallel_io.MPI2.f90 setup_inputs.f90 setup_set_parameters.f90 setup_initialise_microstate.f90 setup_initial_record.f90 simulation_compute_forces.f90 simulation_move_particles.f90 simulation_checkrebuild.f90 simulation_record.f90 finish_final_record.f90 finish_clear_all.f90 md.f90 md_main.f90 build_path param.inc input makefile run_script.sh

3.2 Compiling

Compilation is performed using a makefile. This used the value of `mpif90` as the default compiler, this can be changed if necessary. First compile a serial executable, type

```
make s
```

which should generate the executable `md.exe`. This can be run as explained in the next section. To build a parallel executable- first adjust `param.inc` to the required number of processors and then type

```
make p
```

To see the list of other options including various debug and optimised modes

```
make help
```

The `buildpath` file also contains the commands to build the code using the intel fortran compiler. It should be possible to build the executable from the command line using these instead.

There is no dependence on libraries or files outside of the tarball, other than fortan MPI which should be included in mpif90 command.

3.3 Running

The executable `md.exe` can be run locally from the command line by typing `./md.exe`. A `results` folder **must** be created in the directory that the executable is run in to store the outputs. The default simulation is 1000 steps of a periodic system with 2048 molecules. Various outputs should be printed to the screen. The input file can be adjusted to different numbers of molecules and various other parameters can be recorded.

To run in parallel on CX1/2, the queueing system must be used. An example `run_script` for 4 processors is:

```
 #!/bin/bash
#PBS -o /work/USERNAME/110405_MD_dCSE/results/mdresult.out
#PBS -S /bin/bash
#PBS -l select=1:ncpus=4
#PBS -l walltime=00:30:00
#PBS -N MD
module load intel-suite
module load mpi
cd /work/USERNAME/110405_MD_dCSE/
date
mpiexec ./md.exe >> history
date
```

which should be submitted as `qsub run_script.sh`

3.4 TroubleShooting

If there are any problems or questions email me at edward.smith@imperial.ac.uk or call 07792677912. Below are a quick list of some common errors and possible solutions

3.4.1 compiling

The code is generally compiled using the Intel fortran compiler. It has been tested with gfortran as well as a few other compiler previously but not for several revisions of the code. If would like to run it using another compiler, let me know and I will get it working.
Error message of the form

```
[USERNAME 110405_MD_dCSE]$ make s
make[1]: Entering directory ‘/home/USERNAME/110405_MD_dCSE’
mkdir obj
mpif90 -O3 -funroll-loops  -c modules.f90 -o obj/modules.o
...
```

```
mpif90 -O3 -funroll-loops  -c external_forces.f90 -o obj/external_forces.o
md.o: In function ‘simulation_md_’:
md.f90:(.text+0x15c): undefined reference to ‘simulation_move_particles_’
mv: cannot stat ‘md.exe’: No such file or directory
make[1]: *** [md.exe] Error 1
make[1]: Leaving directory ‘/home/es205/codes/MD_continuum/110405_MD_dCSE’
make: *** [s] Error 2
```

Solution Error in linking stage due to missing subroutine - check all files included in the correct directory (see list above). grep ’simulation_move_particles’ *.f90 to see if routine is included and if it is, try compiling from the command line.

3.4.2 running

Error message of the form

```
[USERNAME 110405_MD_dCSE]$ ./md.exe
forrtl: No such file or directory
forrtl: severe (29): file not found, unit 4, file ......
```

Solution Create a results directory in the folder containing the executable - `mkdir results`

Error message of the form (built with `make debug_s`)

```
[USERNAME 110405_MD_dCSE]$ ./md.exe
...
forrtl: severe (408): fort: (2): Subscript #3 of the array CELLNP has
value 7992 which is greater than the upper bound of 11
```

Solution This is the most common error for an unstable simulation. The value of cellnp indicates if a molecules is in the domain and a value outside the domain indicates a molecule has escaped. This can be caused by input condition which cause molecules to move too fast, turning off periodic boundaries and not replacing them with walls or a number of other problems. Check input conditions.

References

Rapaport, D. C. 2004 *The Art of Molecular Dynamics Simulation*, 2nd edn. Cambridge University Press.