Geophysical Research Abstracts
Vol. 13, EGU2011-10767, 2011
EGU General Assembly 2011

Workflows for simulating meteorological phenomena using the WRF model

Viera Sipkova and Ladislav Hluchy
Institute of Informatics, Slovak Academy of Sciences, Bratislava, Slovakia (viera.sipkova@savba.sk)

This research work is focused on the performance investigation of the WRF (Weather Research and Forecasting) model. Its main goal was to find out an optimal configuration of the model considering the given input data and target hardware architecture: a multi-core cluster and grid.

The whole WRF simulation process consists of the accomplishment of many components of different kind and complexity, where each of them may require a various number of processors for execution. The *WRF workflow* is defined as a sequence of three consecutive jobs, where the input and execution of the second/third job are dependent on the output of the first/second job. WRF workflow can be represented as a simple *Directed Acyclic Graph*. Individual jobs of the WRF workflow are defined as follows:

1. *WPS pre-processing* – a normal job comprising of a set of serial programs.
2. *WRF modeling* – a parallel job comprising of two parallel programs: an initialization program and a numerical integration program.
3. *WPP post-processing* – a parallel job comprising of one parallel program running in a cycle over output data for all hours of the predicted time period. As there is no dependency between processing data of individual hours, the computation cycle may be split and the job can be realized as a parametric study.

Each of the jobs is performed through the invocation of the corresponding *job-submission* script which provides for execution of all necessary operations, including parameters checking, environment variables settings, data management, and calls of executables. In order to automate the process of WRF workflow runs with a minimum of human intervention, we developed *workflow-manager* scripts that realize the whole simulation, on both the local cluster and EGEE Grid. A workflow-manager script is responsible for the completion of the next actions:

- It accepts the input arguments specifying the time period to be predicted and the number of sub-jobs the WPP post-processing should be divided into. Arguments are passed to the subsequent operations.
- It produces job submission scripts which serve as an input either to the *Portable Batch System*, in case of submitting the job to a local cluster, or to the *gLite middleware* for submitting the job to the EGEE Grid.
- It provides for the execution of the WRF workflow using the generated job-submission scripts to start individual jobs. Within grid versions also monitoring the status of submitted jobs, and retrieving output of finished jobs is performed.

Achieved results

To analyze and evaluate WRF workflows we ran a number of experiments. We concentrated mainly on the job of WRF modeling, as this represents the dominant, most time-consuming part. Within the grid-aware simulation we measured the grid overhead for different workflow structures. We compiled and examined two parallel scenarios: a *pure MPI* and a *hybrid MPI+OpenMP* variant. Experiments were performed on the cluster with 16 compute nodes, each comprising of one dual-core processor.

Running on the local cluster: The comparison of results obtained from several variants of the WRF workflow shows, that for given input data the only visible performance improvement could be achieved using the hybrid parallel configuration of the model, where the total execution time of the simulation was decreased by about 30%.

Running on the EGEE Grid: Because the hybrid parallel MPI+OpenMP jobs have not been supported yet on the EGEE Grid, only the pure MPI scenario was examined. We measured the grid overhead for three different WRF workflow exemplars. The comparison between these cases reveals clearly that the grid overhead grows proportional with the complexity of the workflow structure and with the increasing number of data transmissions between computing and storage resources.