CHAPTER 5

APPLICATION-SPECIFIC EVALUATION

This chapter presents an evaluation of the algorithm on a real application scheduler. The focus is on feasibility with practicality of the algorithm.

The proposed algorithm can be useful for applications which have long computation time, such as drug-screening application. The objective of this application is to find the best or just top-rank results first. This algorithm is implemented on the top of general schedulers. However, general schedulers can not efficiently control this kind of application. Therefore, the High-level Workflow Scheduling is developed in order to efficiently handle with such application.

This chapter can be divided into 4 parts which are High-level Workflow Scheduling Implementation, drug-screening application, Result and discussion. The details of these parts are described as follow.

5.1 High-level Workflow Scheduling Implementation

Torque the PBS scheduler is selected as a job scheduler in the system. Such a scheduler has a high capability to control jobs but it is inconvenient to control workflows. To improve the ability of workflow controlling on a job scheduler, the high-level workflow scheduling and management is implemented on top of the Torque.

5.1.1 A design of the High-level workflow scheduling

Instead of building a new workflow scheduler from scratch, a higher-level software layer is build on top of available local batch schedulers.

The architecture of the high-level workflow scheduling is illustrated in Figure 5-1, which consists of four major components.

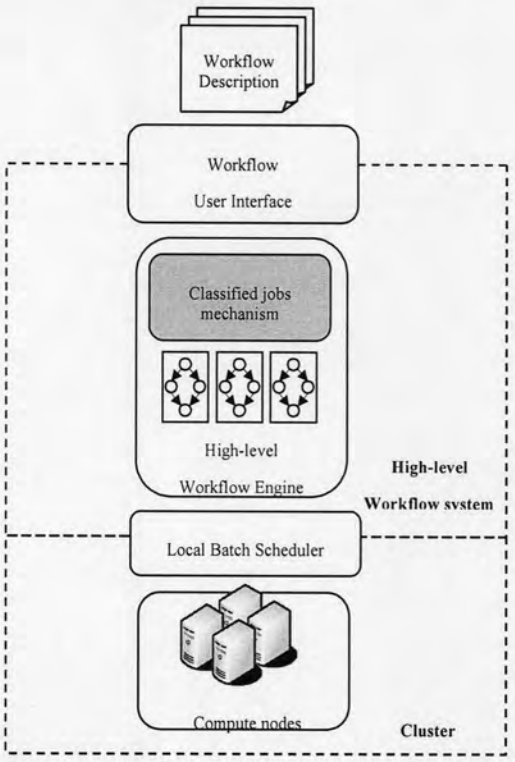

Figure 5-1 The framework of high-level workflow scheduling and management

- *Workflow description* is a method to identify workflow specification as a DAG and define workflow priority for particular workflows or applying the default workflow priority.
- *Workflow user interface* is a tool for handling workflows that are waiting in the queue or running
- *High-level workflow engine* is the control center of workflow execution. Workflow instances are organized and managed by interpreting the workflow description.
- *Local batch scheduler* is a scheduler to dispatch individual jobs into cluster nodes

The detail of each component is described in the following subsections.

5.1.1.1 Workflow description

In the workflow description, workflow instances are defined with the data directory, job dependencies, workflow priority and job priority, as shown in Figure 5-2. The structure of workflow description is listed as follow.

- *Workflow Definition* specifies workflow directory, workflow name, and default workflow priority. WF_DIR specifies the top-level directory that contains subdirectories for each workflow in the batch. Each workflow subdirectory contains input data, temporary data and output data. The three-column list specifies workflow name, subdirectory, and priority of each workflow, respectively. The priority can be absolute or relative to the default priority.
- *Job Definition* specifies job directory, job name and default job priority. The job directory contains job scripts. Each job name is assigned with a job script and priority.
- *Dependency Definition* specifies dependency among job by using PARENT-CHILD relationships. This is similar to the syntax of DAGMan. In addition, this can define the evaluation of intermediate result in the Best-Intermediate-Result-First heuristic by adding a special rule which is called EVALUATE.

```
# Workflow Description
# ==================

# Workflow definition
# ------------------------------
WF_DIR=/home/chem/drug_discovery/data/wf/
DEFAULT_WF_PRIORITY=0
WF_19_opt       19_opt  (10)
WF_20_opt       20_opt
WF_23_opt       23_opt  (+5)
WF_27_opt       27_opt  (20)

# Job definition
# ------------------------------
JOB_DIR=/home/chem/drug_discovery/description/user_data/
Job_A=jobA.sh
Job_B=jobB.sh
Job_C=jobC.sh
Job_D=jobD.sh
Job_E=jobE.sh
Job_F=jobF.sh
Eval_1=eval_1.sh
```

Figure 5-2 Workflow description

5.1.1.2 High-Level Workflow User Interface

The system provides many operations for workflow management including:

- *wf_submit* submitting a workflow instance to the workflow engine
- *wf_stat* displaying the status of a workflow instance, including state, priority and computation time
- *wf_del* removing a workflow instance from workflow scheduler queued
- *wf_setpri* changing the priority of a workflow instance

5.1.1.3 High-level workflow engine

In torque, there are many utilities including capabilities to define the individual job dependency and the priority of the job. However, Torque can perform only at the job level and can not treat a workflow as a single entity. The proposed

workflow engine works at the workflow level so that workflow operations, such as submitting workflow instances, monitoring status, removing workflow from queue and changing workflow priority, can affect all jobs within the workflow.

The engine is started after obtaining a workflow description. It interprets the description and creates workflow instances and the jobs within the workflows. It determines workflow jobs that are ready to execute by examining whether their dependencies has been resolved. Ready jobs are then submitted to the workflow scheduler queue. When the engine selects a workflow instance from the queue, individual jobs are extracted from the workflow and submitted to local batch scheduler. After that, the individual job is executed when it is selected from local batch scheduler queue. Finally, after computation is finish, the engine retrieves the final result from compute node and transfers it to the data directory.

5.1.2 Management of the High-level workflow scheduling

An overview of the system is shown in Figure 5-3. This system comprises of two events, namely creating and controlling a workflow. Creating a workflow composes of two sub-events, namely creating a workflow and dispatching workflow instances from workflow scheduler. This event involves defining a workflow with a workflow description language and submitting the workflow description to a high-level workflow scheduler using a workflow submission command (*wf_submit*). When the scheduler selects a workflow from the workflow scheduler queue, it analyses the dependency among jobs and submits them into the local batch scheduler queue. Subsequently, jobs are dispatched from the local batch scheduler to execute at the compute nodes in the order specified by the DAG.

The controlling workflow event is used in workflow management. This includes many operations such as *wf_status* that shows workflow process status, *wf_del* that removes workflow from workflow scheduler queue, and *wf_setpri* that changes workflow priority.

This high-level workflow framework gives a great deal of simplicity and efficiently supports scheduling and control of workflow operations.

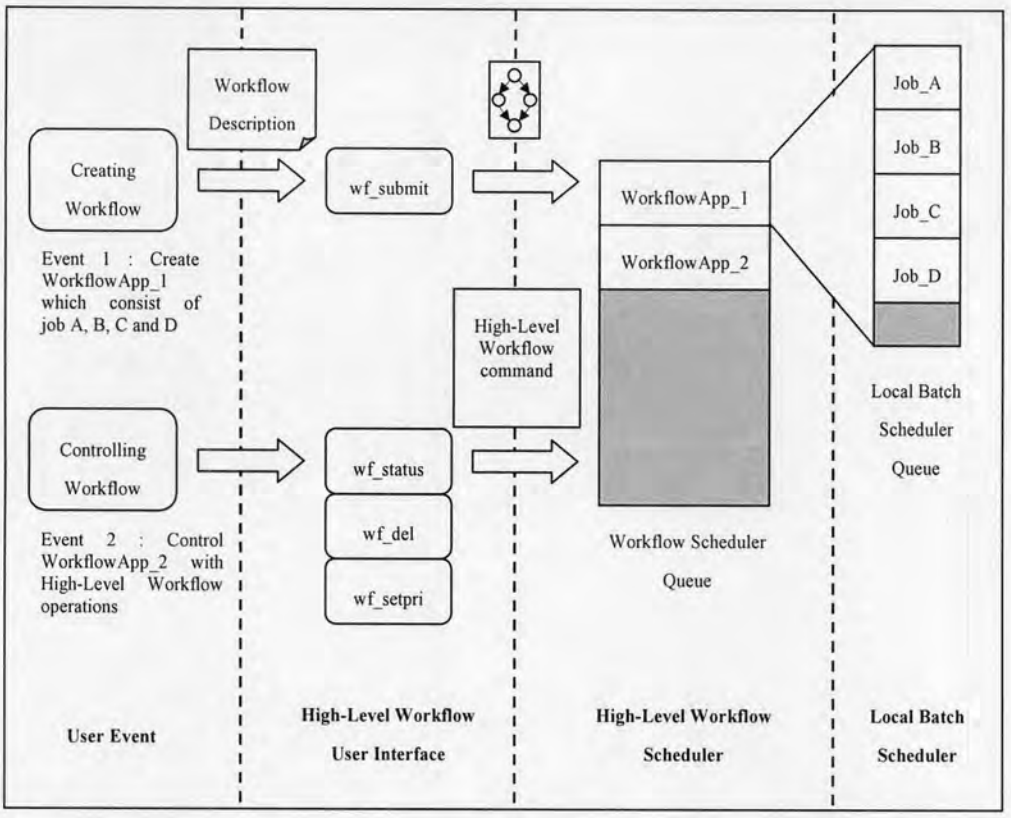

Figure 5-3 The event of workflow scheduling and management

5.2 Drug Discovery Application

The proposed technique has been applied to highthroughput drug screening. It is the process of screening a database of ligands (drugs or small molecules) to find the most compatible drug for a particular protein. Generally, drug screening consists of two major steps, namely molecular docking and scoring [2]. The docking process predicts ligand conformation within a targeted binding site. The scoring process evaluates the predicted conformations. In our particular drug screening application, two major programs, AutoDock [3] and Gaussian [4], are used in complement. AutoDock is a docking program to prepare ligand-protein complexes. Its scoring scheme is based on molecular mechanics calculation of free energy and binding energy. Gaussian uses quantum mechanics calculation to provide another scoring scheme for selection of the most suitable binding modes provided by AutoDock.

Figure 5-4 shows the workflow of the drug screening process. There are a few small programs involving data preparation for AutoDock, namely mkgpf, mkdpf and AutoGrid. Another program is needed for extracting the scoring result from AutoDock output file.

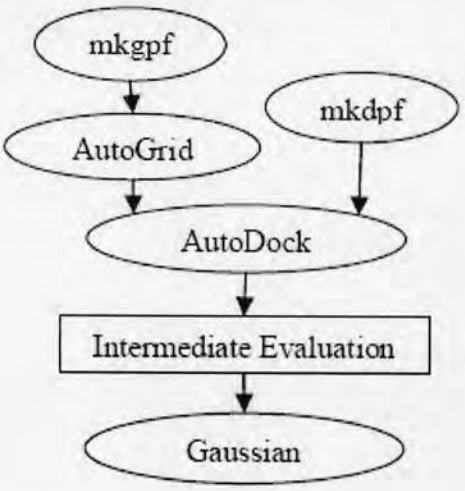

Figure 5-4 Drug screening workflow

5.3 Experiment Setting

The preliminary evaluation was performed with a small set of samples consisting of 24 ligands and one protein. The data of the ligands and targeted protein were real data obtained from the Computational Chemistry Lab. Twenty-four instances of the workflow were created by varying the ligands. Initially, each workflow instance was run on a PC with a 2.8 GHz Pentium IV processor in order to obtain the result and timing data. AutoDock and Gaussian programs are computing-intensive and produce small text files (a few hundreds kilobytes). The results occupy a specifically labeled section consisting of a few lines that can be quickly extracted with a simple Perl script. To quickly start a pilot study, we use the workflow scheduler that we have implemented and a small cluster with 4 compute nodes as a simulated environment. Each compute node is assigned with only one job at a time. Application components in the workflows were replaced with sleep commands. The sleeping times are set according to run-times of component programs obtained from the real execution of each particular sample. The collected run-times were scaled down 30 times in the simulated run. The proposed technique was compared against the conventional FIFO scheduling scheme. In that scheme, the workflow with the earliest arrival time was selected from the waiting queue. Once it started, it would run until finished. Intermediate evaluation was done but not used for scheduling.

Each workflow instance was selected by random and submitted to the simulator subsequently without delay. This step was repeated 50 times. In other words, 50 random sequences of the 24 workflows were generated for the evaluation.

Each sequence was evaluated with the proposed technique and the conventional technique.

5.4 Results and Discussion

The practicality of the proposed algorithm depends on the correlation between the intermediate and final results. Each compound is given two ranking numbers between 1 and 24 according to the results from AutoDock and Gaussian respectively.

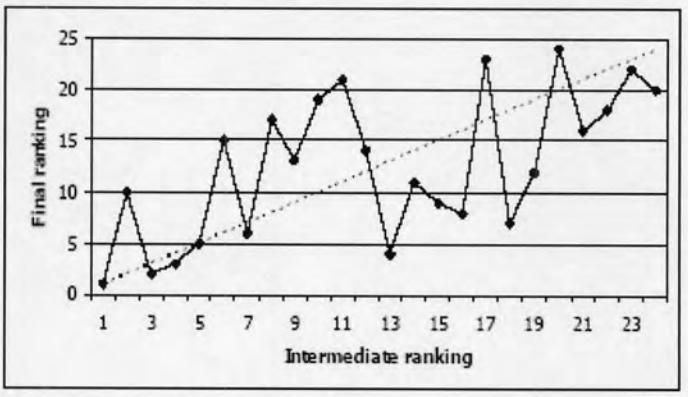

Figure 5-5 Intermediate and final ranking

The relationship between the two ranking numbers is shown in Figure 5-5. The dashed line represents the ideal case. The difference between the two ranking numbers was 5.08 on average. Their correlation efficient was 0.61. Therefore, there was indeed a correlation between them, even though it was not high. In addition, the average before-evaluation time and after-evaluation time is 17.25% and 82.75% of the completion time respectively.

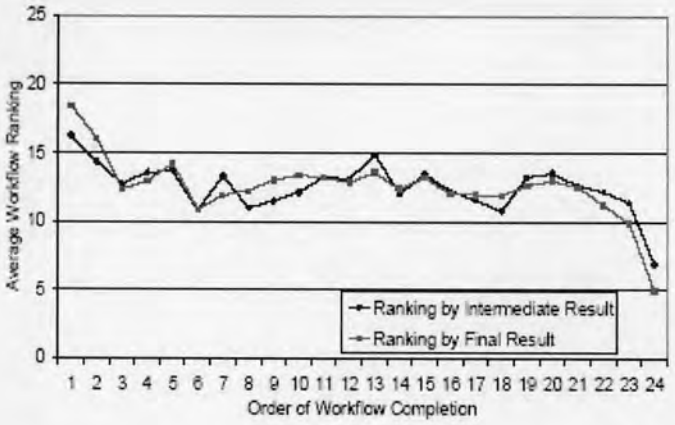

Figure 5-6 Average workflow ranking of completed workflows (with conventional scheduling algorithm)

Figure 5-6 shows the average rank of each workflow that completed in order on the conventional system. Both ranks by the intermediate results and by the final results are shown. Since the order of completion was not affected by the results, the ranks of the workflows with any order of completion were close to 12.5 (the average value of numbers from 1 to 24). The anomaly at both ends of the graph was due to the particular data set being used. A few workflows with the shortest completion time gave very low ranked results and also a few workflows that took the longest time happened to be the top ranked ones.

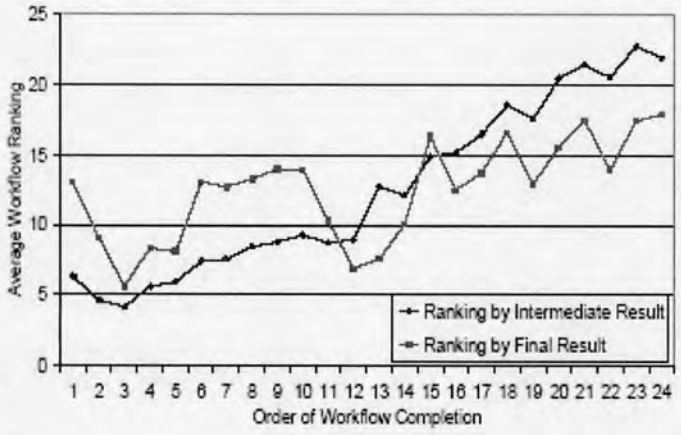

Figure 5-7 Improved average workflow ranking

Figure 5-7 shows the relationship between the order of completion and the average rank when our technique was applied. The correlation coefficient between the order of completion and the ranking by the intermediate results was 0.97. It indicated that the algorithm performed very well. The correlation coefficient between the order of completion and the ranking by the final results was 0.66. That was due to the

differences between the intermediate ranking and the final ranking as mentioned above. In an ideal situation that the final results are in agreement with the intermediate results, our algorithm will perform better.

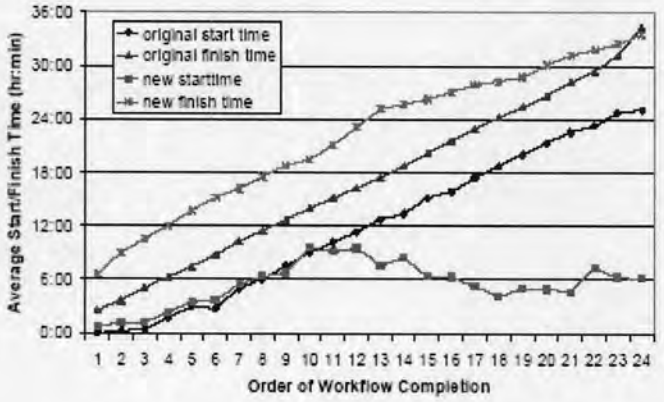

Figure 5-8 Average start and finish times

Figure 5-8 shows the average start time and average finish time for each order of completion. The parallel lines in the middle are start and finish times of workflows on the conventional system. The bottom and the top lines are start and finish times on the new system. Apparently the new system produced the output later than the conventional one. In about six hours the conventional system finished three workflows while the new system just almost finished the first.

However, by looking more closely and in complement to the previous graphs, we can see the true benefit of the new system. With the proposed technique, workflows started to execute earlier than the conventional technique. At the beginning there were more workflows running in the before-evaluation phase than those running in the after-evaluation phase. Therefore, there was higher possibility to run the before evaluation workflows. The graphs show that most workflows started within less than 10 hours. By that time, most workflows had passed the evaluation point. Therefore, the first results of the new system came out later than in the conventional system.

Also, since the workflows with lower intermediate ranks might be preempted by the higher ranks, their execution in the second phase was delayed and the completion time is extended. In about 18 hours, half way of the total time, the conventional system finished 13 workflows while the new system finished only 8. However, the average ranking of the 13 workflows was 13.1 and 13.4 for intermediate ranking and final ranking respectively. The average ranking of the 8 workflow was

6.2 and 10.4 for intermediate ranking and final ranking respectively. That means the new technique could make most of good results come out before bad results.

Within 24 hours the new system finished 12 workflows. All of them have the average intermediate ranking less than 9 and six of them have the average final ranking less than 12 or in the better half of all workflows. At the same time the conventional system finished 17 workflows with the average ranking of 12.9 and 13.2 for intermediate ranking and final ranking respectively.

After 24 hours, the new system finished workflows more frequently than the conventional system. The last workflows finished at about the same time. That means the makespans of both systems were the same. In the conventional system, the completion times were about the same for all completion orders. For the new system, the workflows that complete earlier had less completion times. The completion times were much greater after 10 workflows completed.