CHAPTER 3

THE BEST-INTERMEDIATE-RESULT-FIRST ALGORITHM

After receiving the best intermediate result, the high priority is set to the corresponding workflow.

3.1 Assumptions

There are some assumptions or preconditions that need to be satisfied before using the proposed algorithm. First of all, the algorithm is not designed to be used as a general purpose scheduler. Applications that can benefit from this technique must have a particular characteristic. Its workflow must have at lease one stage that produce intermediate results that can be used for evaluation of its likeliness to give good final result. The users have to define the way to compare the intermediate results. We also assume that computation time that is required for the evaluation and comparison of intermediate results is negligible.

Note that many scheduling algorithms also need some kind of preconditions. For example, the Shortest-Job-First algorithm requires that the execution time can be accurately estimated.

3.2 Algorithm

A new heuristic for scheduling parameter-sweep workflows called Best-Intermediate-Result-First (BIRF) is proposed to achieve the objective. The ideal outcome is to have the workflows complete in the same order that their results are sorted. However, the achievable goal is to minimize the turnaround time of the workflows that give the best results. To achieve that, the workflows that give better results are given higher priority than the ones that give worse results.

The scheduling algorithm is based on dynamically adjusted priority according to intermediate data obtained at some stage in the workflow. When a workflow reaches the point where intermediate result is created, the intermediate result is evaluated and compared to the intermediate result of other workflows in order to assign priority to the remaining tasks of the workflow.

For simplicity of the algorithm, this research first consider the case when a workflow has only one evaluation point. Therefore, a workflow can be in two phases, namely before-evaluation and after-evaluation.

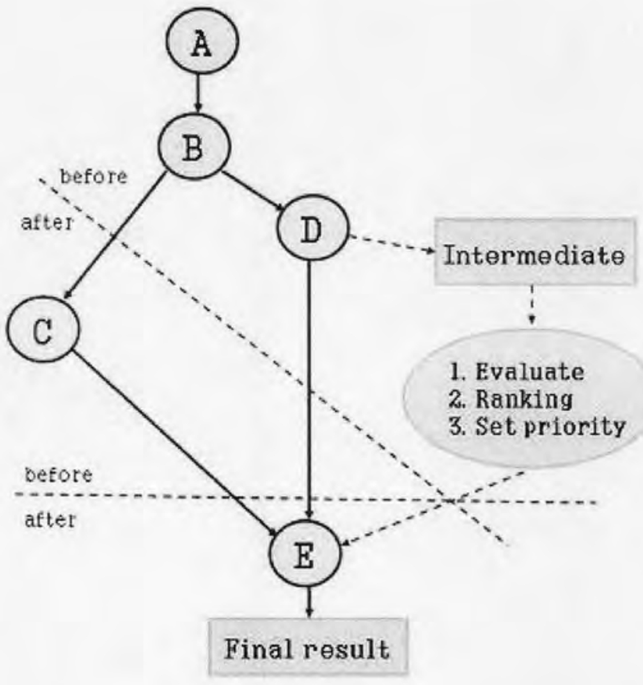

Figure 3-1 Workflow and intermediate evaluation

Figure 3-1 depicts a sample workflow with five computing stages, A, B, C, D and E. Stage D produces the intermediate result. When D is completed, the workflow reaches the evaluation point. The available result is passed to some application-specific program that can extract data value needed for evaluation. This value may be directly extracted from an output file or indirectly calculated, depending on the nature of the application. The value is passed to the evaluator program. The evaluator collects the intermediate results from workflows and sorts them and assigns priority accordingly. When a machine becomes available, a ready-to-run job of the top ranked workflow will be assigned to that machine.

All stages that start execution after the evaluation point are given the new priority level. The stages that have already started are not affected. For example, the stage E always runs in the after-evaluation phase because it runs after stage D. However, the stage C can run in the before-evaluation or after-evaluation phase depending on whether it starts before or after the intermediate evaluation is done.

This algorithm is on-line. That means it does not require that all workflows must be evaluated before ranking. Rather, any workflow that passes the evaluation point is immediately pushed into a priority queue. The intermediate ranking process involves only the workflows that wait in the priority queue.

There arises a new problem. The scheduler ought to select the top ranked workflow to run. However, there are workflows that have not been evaluated and better results may come from them. Therefore, the scheduling algorithm needs to balance between speeding up the best evaluated workflows and looking for better results in new-coming workflows.

The solution is based on the use of two queues for workflows in the two phases, namely before-evaluation queue and after-evaluation queue. Figure 3-2 shows a diagram representing the structure of the scheduling system. When a workflow is submitted it is put into the before-evaluation queue first. After the workflow is executed and the evaluation point is passed, it will be moved to the after-evaluation queue. The before-evaluation queue is a FCFS queue, while the after evaluation queue is a priority queue.

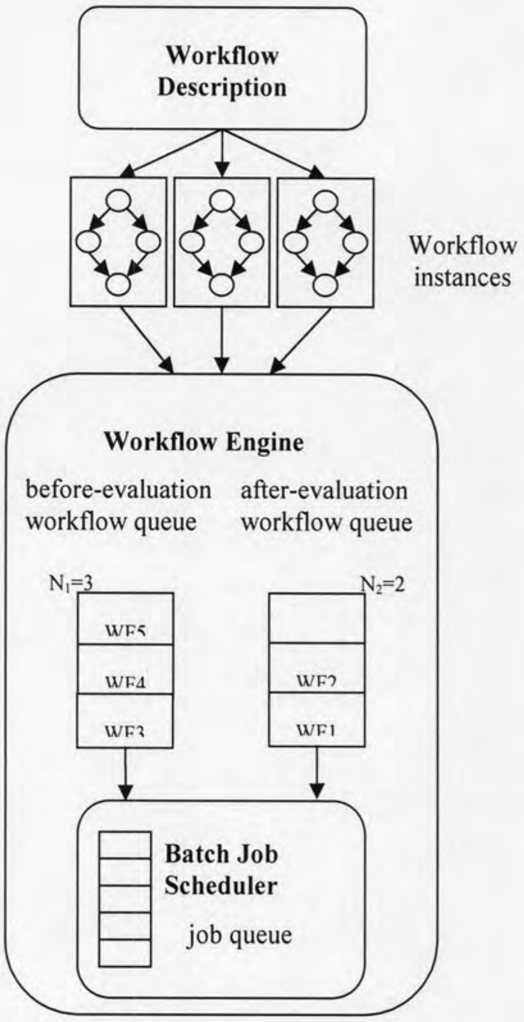

Figure 3-2 Workflow scheduling system

When a machine is available, the scheduler first selects between the before-evaluation queue and the after-evaluation queue based on an adjustable probability condition.

Let P_1 and P_2 be the probability that the before-evaluation queue and the after-evaluation queue will be selected, respectively. If the objective is to explore more samples looking for better candidate solutions, P_1 should be greater than P_2. If instead the objective is to get the results from the evaluated workflows quickly, P_2 should be greater.

It is interesting that by varying the value of P_1 and P_2, we have spectrum of scheduling algorithms. If P_1 is 0 and P_2 is 1, the result is the FIFO algorithm. If P_1 is 1 and P_2 is 0, all workflows are evaluated and ranked before the next phase can begin.

With more workflows being compared, ranking is more accurate but the results will be further delayed. We call this algorithm as Before-Evaluation-Queue-First (BEQF)

At the middle of the spectrum is where both P_1 and P_2 are 0.5. This can be achieved by randomly selecting between the two queues. Therefore, we call this algorithm as Random Algorithm.

Another variation is to select the queue that has more tasks. This algorithm is thus called Weight Queue. The result is that when the Before-Evaluation Queue is longer, the algorithm behaves like the BEQF algorithm, otherwise it behaves like the FIFO algorithm.

The last is the BIRF algorithm. Its basic idea is to do the best for all situations. When there are more unevaluated workflows than the evaluated ones, especially at the beginning, more chances are given to evaluating the new comers. When there are more evaluated candidates, more chances are given to the best candidate to complete the job.

Scheduling type	P_1	P_2
First-In-First-Out	$0 \; if \; N_2 > 0$ $1 \; if \; N_2 = 0$	$1 \; if \; N_2 > 0$ $0 \; if \; N_2 = 0$
Before-Evaluation-Queue-First	$1 \; if \; N_1 > 0$ $0 \; if \; N_1 = 0$	$0 \; if \; N_1 > 0$ $1 \; if \; N_1 = 0$
Weight queue	$1 \; if \; N_1 \geq N_2$ $0 \; if \; N_1 < N_2$	$0 \; if \; N_1 \geq N_2$ $1 \; if \; N_1 < N_2$
Random	0.5	0.5
Best-Intermediate-Result-First	$\frac{N_1}{(N_1 + N_2)}$	$\frac{N_2}{(N_1 + N_2)}$

Table 3-1 Summarizes the algorithms and their probability value

Let N_1 and N_2 be the number of workflows in the before-evaluation queue and the after-evaluation queue respectively. The scheduler gets a random numbers, R, between 1 and N_1+N_2. If R is less than N_1, then the before-evaluation queue is selected. Otherwise, the after-evaluation queue is selected. Therefore, P_1 is

$\frac{N_1}{(N_1+N_2)}$ and P_2 is $\frac{N_2}{(N_1+N_2)}$. Subsequently, the scheduler selects a workflow from the selected queue. As a result, the BIRF algorithm is an adaptive algorithm that combines all the above algorithms.