Chapter 7

The Program Execution Environment

The program execution environment on a CM-5 partition supports both interactive program executive and batch execution. In either case, the program executes on the partition manager and accesses the associated set of processing nodes, plus I/O devices and other devices (such as graphics workstations) as needed.

Two job control systems are available. The Distributed Job Manager (DJM) provides job control and load balancing for both interactive and batch execution. Alternatively, the Network Queueing System (NQS) can be used to execute batch jobs.

Access to the interactive environment, therefore, can be achieved through remote login or remote shell commands, or through DJM's **`jrun`** command. Access to the batch environment can be achieved through either DJM's **`jsub`** command or NQS's **`qsub`** command.

The interactive environment is, by default, a timeshared environment. DJM, however, can be used to create a dedicated (single-user) interactive environment at chosen times. In addition, an administrator may limit access to any given partition by setting UNIX permissions to grant access only to certain users or projects. The administrator can similarly tailor batch queues to the needs of particular groups of users or types of jobs, and can define times for dedicated batch access. The system administrator thus has power, not only to partition the system optimally for the site's users, but also to choose the type of environment available on each partition at any given time: timeshared or dedicated, interactive or batch or both.

To further enhance the program execution environment, the CM-5 offers the Prism programming environment (discussed in the previous chapter), with its suite of tools for debugging and for performance analysis of both data-parallel and message-passing programs. Additional tools, such as the CM timers and the

Copyright © 1993 Thinking Machines Corporation

checkpointing facility are also provided, and are discussed at the end of this chapter.

7.1 Batch Utilities

The CM-5 offers two batch subsystems:

- NQS (Network Queueing System), batch system for UNIX.
- DJM (Distributed Job Manager), which provides both job management and load balancing for Connection Machines.

7.1.1 DJM

The Distributed Job Manager (DJM) was initially developed at the Minnesota Supercomputer Center. It is designed to

- manage the flow of jobs through a Connection Machine system
- avoid resource conflicts
- provide a load-balancing capability among the various control processors of a CM-5 system

In order to balance system load, DJM handles all application processes running on the CM-5, interactive jobs as well as batch jobs. It allows jobs for which there are sufficient resources to execute immediately; it queues all other jobs for later execution.

DJM ensures that it handles all jobs by trapping jobs submitted directly, rather than via its own job-submission commands. It has the ability to impose limits upon such jobs, or even to kill them when appropriate.

DJM allows users and administrators alike a great deal of flexibility. Users can request either dedicated time or multi-user access. They can specify many parameters for their job, or only a few — or, they can accept DJM's default parameters. They can ask that DJM send them mail when a job begins or ends execution. They can move a job from interactive execution ("the foreground") to batch execution ("the background"), or vice versa.

November 1993
Copyright © 1993 Thinking Machines Corporation

Administrators can choose how many queues to set up and what restrictions to put on various queues (which users can use the queue, what resource limits are placed on jobs using this queue, etc.). They can choose whether to allow dedicated access to the system, and when to allow it. They can determine how "foreign" jobs (jobs not submitted via DJM commands) are to be handled.

DJM does not necessarily perform strict FIFO queueing (although it can do so, if so configured). Instead, it sets job priorities by calculating a "score" for each running job and for each queued job. Assuming available resources, jobs with higher scores execute before jobs with lower scores.

Scores are recalculated at regular intervals. This allows, for instance, "amount of time spent waiting in queue" to raise a job's score, and "job has exceeded its expected CPU usage" to lower a job's score.

Users submitting jobs via DJM must specify at least three facts about their jobs:

- the number of processors required
- the CPU time the job is expected to require
- the amount of memory the job is expected to require

DJM treats this information as "soft limits" to help it allocate resources and schedule jobs. Jobs that overrun their estimates become vulnerable and may be replaced by jobs with higher priority, ifany such jobs are queued and waiting to run. The administrator sets both hard and soft limits for queues, and also provides values for the parameters used to construct scores for queues and running jobs. Thus, the administrator has great control over DJM's handling of jobs.

The handling of dedicated access is also flexible. The administrator can choose when to put a partition into dedicated mode; whether timeshared jobs executing at the changeover time can continue to execute or not; whether the changeover will happen automatically, or whether it requires the presence of a job in the "dedicated queue" to trigger it; and so on. All these elements of flexibility make DJM particularly useful for sites where the CM-5 gets heavy and/or varied usage.

7.1.2 NQS

The Connection Machine supports the Network Queueing System (NQS) batch system. This batch system supports two types of queues: batch queues, which are directed to a specific PM, and which run on the partition that is controlled by that PM at the time the job is submitted; and pipe queues, which feed jobs (via batch

Copyright © 1993 Thinking Machines Corporation

queues) to any suitable batch queue that is available to run them. The pipe queue can be directed to any batch queue, or only to batch queues that meet specified minimal resources. NQS queries current partitions to find one suitable for running jobs from these queues.

NQS allows the administrator to control the number and characteristics of queues at a site and to define the hours during which each queue will accept and execute jobs. Note that the two sets of hours are not necessarily identical: a queue might accept jobs from 8 am till midnight, but execute jobs between 8 pm and 8 am. (A queue that accepts jobs is said to be enabled; one that executes jobs is said to be started.)

Creating and Configuring Queues

An NQS manager decides how many queues to create and what characteristics each queue will have, thus tailoring the batch system to the needs of the particular site. The administrator uses the **`qmgr`** utility to create each queue, naming and describing the queue and defining

- the hours during which the queue operates (queues with restricted hours start and stop automatically at designated times)
- the priority of this queue in relation to other queues
- the users or groups of users who can submit jobs to the queue
- time and size limitations for jobs executing from the queue
- the CM system resources available to jobs executing from the queue

Submitting Batch Requests

Frequently, the NQS manager defines a number of queues with different characteristics. Users can then choose the queue most suitable for each program. In addition, users can further define the execution environment for a program by using options to the job submittal command that

- request that execution be delayed until a particular time
- request the use of a specified shell
- request that all environment variables be exported with the job

November 1993
Copyright © 1993 Thinking Machines Corporation

- direct the method by which output is to be handled
- set various per-process limits
- assign a priority to the job

Users can also ask for notification by electronic mail of a job's progress, and can query the system for information on the characteristics and availability of queues and on the status of queued requests.

Controlling Batch Queues

NQS operators can start and stop queues, enable and disable queues, and shut down NQS. When necessary, they can also remove waiting and executing jobs from queues.

7.2 Timers

The CM-5 offers two sets of timers: the global CM timers (which time parallel actions across an entire partition) and the single-node CMMD timers. Both sets can calculate, with microsecond precision, both the total elapsed time for a program or routine and the amount of time during which the nodes are active.

Calls to CM timers or CMMD timers can be inserted anywhere in a program. A program can use (and nest) up to 64 timers for simultaneous coarse-grain and fine-grain timing.

7.3 Timesharing

The Connection Machine system uses the UNIX timesharing mechanisms, with all the administrative flexibility they provide. Each partition manager controls timesharing on its partition, switching processes in and out as necessary. (Because a data parallel process running on the PM plus the nodes is a single process, it is switching as a single entity.)

Copyright © 1993 Thinking Machines Corporation

7.4 Checkpointing

Many applications that run on the Connection Machine system require extended execution time. Users may need to be able to interrupt and later restart such a program for any number of reasons: to allow it to run only when the system is not needed for other use, to allow for scheduled machine downtime, to protect against unscheduled halts, or simply to allow for restarting the program from some intermediate state during debugging. The Connection Machine system supports this need with a checkpointing facility.

Checkpointing a program lets the user save (and later restart) an executable copy of a program's state. This includes the program's state on the partition manager (PM) and nodes, a list of the files that the program has open at the time of the checkpoint, and a stored copy of the checkpointed program.

The CM checkpointing facility offers three basic methods of checkpointing:

- inserting checkpoints at particular points in a program
- having checkpoints occur periodically
- having a checkpoint occur when a program is sent a particular signal, such as the signal sent during a planned shutdown of the system

Checkpointing can be used from within batch jobs and interactive jobs, including those running under **`cmdbx`** and Prism. It can be used on programs that execute on the PM only, as well as those that use both the PM and the nodes.

November 1993
Copyright © 1993 Thinking Machines Corporation