Operations Management Software for the K computer

● Kouichi Hirai ● Yuji Iguchi ● Atsuya Uno ● Motoyoshi Kurokawa

Supercomputer systems have been increasing steadily in scale (number of CPU cores and number of nodes) in response to an ever increasing demand for computing power. Operations management software is the key to operating such ultra-large-scale systems in a stable manner and providing users with a high-performance computing environment having a high utilization rate. Fujitsu previously developed software called "Parallelnavi" to provide uniform operations management for its 3000-node supercomputer systems, but to uniformly manage an ultra-large-scale system like the K computer on a scale of more than 80 000 nodes, it expanded its development of operations management technologies. This paper introduces operations management software developed for the K computer, focusing on functions for achieving stable operation of an ultra-large-scale system and a job scheduler for providing a high-performance computing environment.

1. Introduction

The following goals were established to provide stable system operation and a high-performance computing environment through the operations management of an ultra-large-scale system such as the K computer.[note 1)]

- Reduce the overhead of operations management software
- Provide a user-friendly display summarizing huge amounts of information
- Prevent drops in execution performance due to interference between batch jobs

To meet these goals and to effectively extract the high computational performance provided by hardware, we expanded our development efforts in operations management software technologies.

This paper describes our approach to meeting the above goals in operations management software for the K computer and describes the features of this software.

2. Configuration of operations management software

Operations management software for the K computer is broadly divided into system management functions and a job scheduler, as summarized below.

1) System management functions

These functions provide the system administrator with an operations management view in the form of a single system. They enable the administrator to manage system configuration information and to monitor the operating state of each node making up the system while providing high system availability in the face of abnormal occurrences. They also enable the administrator to install new software and maintain existing software through the application of patches and other means.

2) Job scheduler

This function provides an execution

note 1) "K computer" is the English name that RIKEN has been using for the supercomputer of this project since July 2010. "K" comes from the Japanese word "Kei," which means ten peta or 10 to the 16th power.

environment for batch jobs submitted by multiple users to achieve shared use of the supercomputer system.

The following sections introduce the features of these system management functions and the job scheduler.

3. System management functions

3.1 Dealing with ultra-large-scale system

System management functions continuously monitor the operating state of the system for the occurrence of faults. As the scale of a system expands, however, network loads caused by regular monitoring and monitoring-related processing on compute nodes become too large to ignore. For this reason, system management functions incorporate the following measures to keep up with the ongoing expansion of a supercomputer system.

- Load distribution of monitoring processing through a hierarchical structure
- System noise reduction using the Tofu interconnect

These and other measures such as a display for summarizing the operating state of system nodes have been devised to cope with an ultra-large-scale system.

3.2 Load distribution using hierarchical structure

To provide the system administrator with an operation control environment having a single system view, the load must not be allowed to concentrate on the administrator's personal computer (management PC) or associated server. If each node in a more than 80 000-node environment were to simply send monitoring data over the network to the management PC, the processing capacity of the intermediate network paths or of the management PC itself could easily be exceeded. The number of system nodes can also be expected to increase over time, so some form of load distribution that can be scaled up as needed becomes necessary.

Operations management software for the K computer uses the hierarchical software architecture described below to achieve load distribution and parallel processing in operations management.

As shown in **Figure 1**, this hierarchical structure divides the system into node groups and allocates a job management sub-node to each group to manage the nodes belonging to that group. Having job management sub-nodes govern the operations-management processing within their respective node groups distributes the operations-management processing load across the system. Such load distribution can easily be achieved even if the number of nodes increases after initial system deployment by simply adding more node groups, which makes for a highly extensible system.

In addition to enabling the monitoring and control of nodes during regular operations, this hierarchical structure also acts as a platform for maintenance work such as system installations and patch applications and as a platform for any kind of control operations between nodes such as the initiation and termination of a parallel program. Using multiple job management sub-nodes in this way to distribute and parallelize operations-management processing enables a parallel program that is to be run on 3000 nodes to be initiated within one second compared to several seconds in past systems.

This hierarchical structure can therefore benefit not only operations management but also other processes such as the startup performance of ultra-large-scale parallel programs since it can help shorten various operation-processing times that cannot be ignored in an ultra-large-scale system.

3.3 Consolidation of node operation information using Tofu interconnect

The monitoring of node status and the execution of various types of controls by

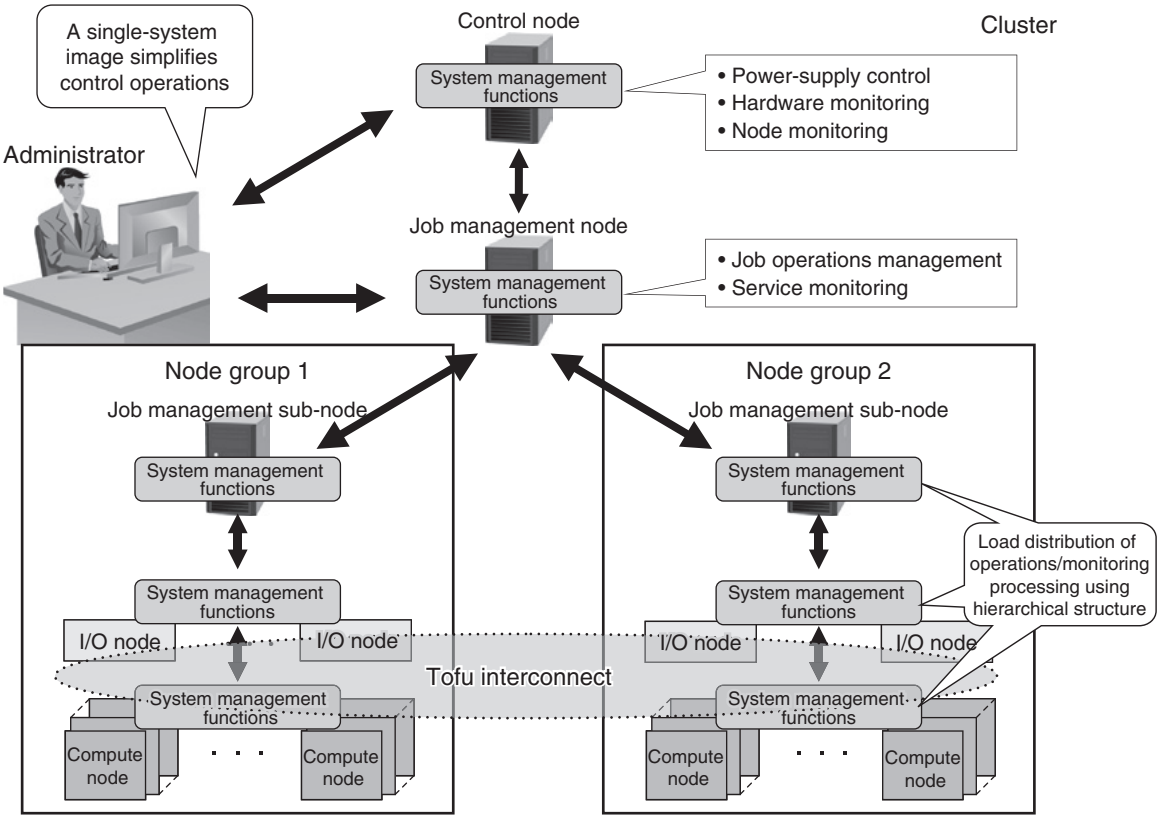

Figure 1
Distribution of load by using hierarchical structure.

operations management software is generally carried out using the following scheme: a daemon program running on a node receives an interrupt from the network such as a Gigabit Ethernet, gathers information on the node in question, and transfers that information to the management PC over the same network. However, operating such a scheme on an ultra-large-scale system can result in system noise caused by network interrupts and daemon processing, and this noise can significantly degrade the execution performance of a parallel program. Thus, instead of collecting and returning information on the occasion of a network interrupt, the operations management software for the K computer uses a scheme in which each node independently and periodically stores in memory its own operational status to be obtained from the outside using remote direct memory access (RDMA)[note 2] communications of the Tofu interconnect. This monitoring and storing of node operation status on each node is performed on the kernel layer of node software as much as possible to significantly reduce the generation of system noise. This and other countermeasures to system noise included in the operations management software have succeeded in reducing system noise to about 1/100 that of a PC cluster, which generates system noise about once every millisecond.

3.4 Other functions

In addition to the above operations-management overhead reduction measures for an ultra-large-scale system, the following functions are also provided as operations management functions for achieving stable system operation.

note 2) A communications function of the Tofu interconnect.

1) High-availability system

This function enables jobs currently in progress to continue execution even if certain nodes in the operations-management system should fail. It therefore guarantees the continuous operation of jobs even during the occurrence of system faults. Even with this function, however, it would not be possible to accept new jobs when certain management nodes fail, and for this reason, a redundant configuration for management nodes is also supported to guarantee continuous operation throughout the system.

2) User-friendly display summarizing huge amounts of information

If the states of more than 80 000 nodes under management were to be simply displayed on a one-node-per-line basis from a terminal command line, the result would obviously be a massive display of more than 80 000 lines. As an alternative, this function displays a summary of system-wide information, such as only the number of nodes currently in different states. Various options are also provided if more detailed information is needed. For example, the user can filter the display of information by specifying a certain node group or a single node.

4. Job scheduler

4.1 Development background

Job schedulers previously developed by Fujitsu for supercomputer systems have the following features.

- Prevention of mutual interference between jobs by advanced reservation of computing resources
- Adoption of the "parallel-job" concept in which a parallel program running on multiple nodes is treated as one job
- Support of diverse requirements of shared-use supercomputer centers

In developing the job scheduler for the K computer, it was decided to incorporate these features and to add two important development themes: load distribution to support an ultra-large-scale system and improvement of system utilization rate.

4.2 Supporting ultra-large-scale system

The number of jobs handled by a job scheduler increases dramatically as system scale expands. The K computer is expected to handle more than 1 000 000 jobs simultaneously.

The job scheduler for the K computer executes the processing steps for controlling a job from job acceptance to job completion (i.e., acceptance processing, execution-prioritizing processing, allocation of computing resources, job-execution processing, and termination processing). These steps are performed in parallel to shorten overall processing time. The job scheduler also performs processing to determine in what order, at what time, and on what nodes jobs are to be executed (a process called "scheduling"). The computational complexity for determining an optimal solution in this regard increases in accordance with system scale, making for high processing cost. The job scheduler for the K computer shortens this computing time by performing the process of selecting execution nodes in parallel. As a result of the above measures, this job scheduler achieves a job-input performance on the level of 3 milliseconds compared to the 0.4 seconds required to submit one job using a job-scheduler product from another company (PBSpro) used for PC clusters (as investigated by Fujitsu).

Additionally, as in the case of system management functions, the job scheduler for the K computer incorporates functions for dealing with the large amount of information related to job processing. For example, it can display information in summarized form to prevent the simultaneous output of a massive amount of data, and, as shown in **Figure 2**, it enables the user to simplify the display of information by selecting what information to display in detail, specifying sort keys, etc.

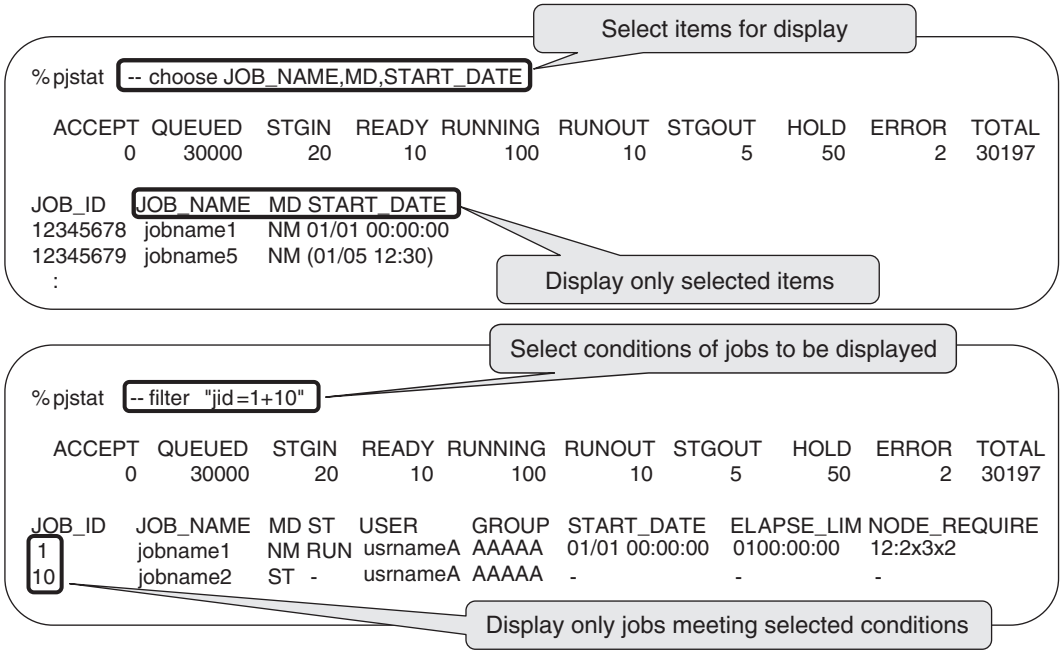

Figure 2
Examples of selecting and displaying job information.

4.3 Improving system utilization rate

In a shared-use center, jobs of diverse scale—from simple jobs needing only one node to ultra-large-scale parallel jobs requiring tens of thousands of nodes—are running at the same time. In such an environment, simply executing jobs in the order in which they are submitted means that if the scale of the job to be executed next exceeds the number of free nodes (nodes on which jobs are not currently being executed), that job will have to wait, thereby degrading the utilization rate of the entire system.

To solve this problem, the job scheduler for the K computer supports backfill scheduling, which raises the utilization rate by having small-scale jobs execute before a large-scale job during the period that computing resources are free prior to the time that the large-scale job is scheduled to commence.

In addition, the following measures are taken when pre-allocating compute nodes to waiting jobs with the aim of raising the utilization rate by taking the characteristics of the Tofu interconnect into account.

- To simplify the pre-allocation of a group of neighboring nodes, compute nodes yet to be allocated (free nodes) are controlled so as to be as adjacent (contiguous) as possible.
- When performing this kind of control, candidate nodes (for executing jobs) are searched for among a variety of three-dimensional patterns using the Tofu interconnect so that contiguous nodes become the most plentiful among free nodes. By uncovering many free nodes in this way, subsequent node allocation is easier, which effectively raises the utilization rate.

4.4 Dealing with I/O contention (file staging)

As a system becomes “ultra-large” in scale, the possibility increases that job execution performance will drop due to I/O contention among concurrently running jobs. For this reason, the K computer supports a “file staging function” that transfers the files needed for job execution to a local file system before commencing job execution and that collects

the files containing execution results after job execution completes.

A general file staging function performs file transfer as part of the job being executed (synchronous staging), but with this technique, the computing resources needed for job execution remain reserved even during file transfer, resulting in a waste of valuable resources. The staging function on the K computer, in contrast, separates file transfer from job execution, as shown in **Figure 3**, thereby preventing such a waste of computing resources. In other words, it exploits the existence of I/O nodes independent of compute nodes (a feature of this system's configuration) and performs file transfers on those I/O nodes asynchronously with job execution (asynchronous staging). Completing file transfer before commencing job execution in this way enables file-transfer time to be deducted from apparent job execution time, which makes for high system utilization.

4.5 Improving system operability

The operation policies of shared-use supercomputer centers in Japan differ widely from center to center. There is therefore a need for a job scheduler that can provide flexible support for a variety of operation policies. Operation policies that need to be customized in a shared-use center can be broadly categorized into two types.

- Individual policies applied to each job to define the limitations and restrictions applicable at job execution time (amount of memory and number of CPUs that may be used, maximum execution time, etc.)
- Overall job-operation policy specifying, for example, which jobs have execution priority

The job scheduler of the K computer defines the various limitations and restrictions for individual policies by using a function called "job ACL " and defines the overall job-operation policy by using a function called "scheduling policy." The development of these functions simplifies the application of operation policies.

5. Conclusion

We developed operations management soft-

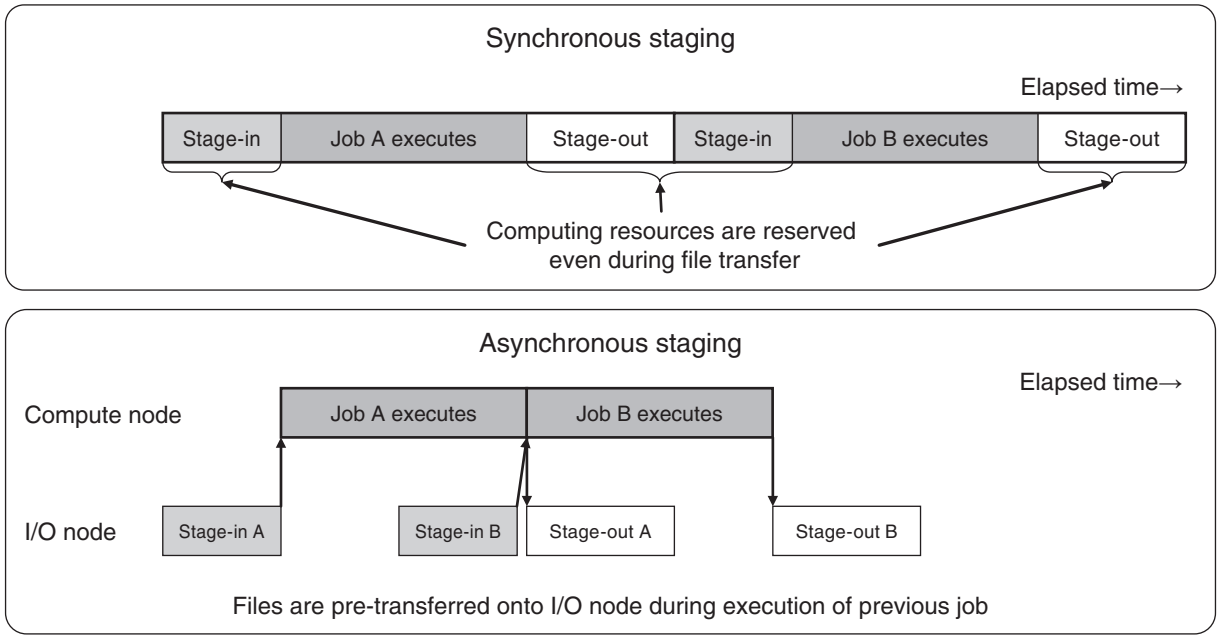

Figure 3
Use of asynchronous staging to improve utilization rate.

ware for the K computer to respond to issues unique to an ultra-large-scale system and to support the further expansion of system scale in the years to come.

This paper focused on the efforts that we have so far taken to support the operation of an ultra-large-scale system. Additional new approaches will be needed to further improve system operability. For example, there is a need for a tool that can simulate system behavior when changing system parameters so that a system administrator can perform system tuning and capacity planning. Combining such a tool with operations management software should facilitate the achievement of stable and efficient system operation.

Kouichi Hirai
Fujitsu Ltd.
Mr. Hirai is engaged in the development of operations management software for supercomputers.

Atsuya Uno
RIKEN
Dr. Uno is engaged in coordinating development of system software.

Yuji Iguchi
Fujitsu Ltd.
Mr. Iguchi is engaged in the development of operations management software for supercomputers.

Motoyoshi Kurokawa
RIKEN
Dr. Kurokawa is engaged in development of system as a whole.