Building a J90 Cluster

Nicholas P. Cardo, Sterling Software, Inc., Numerical Aerodynamic Simulation Facility, NASA Ames Research Center, M/S 258-6, Moffett Field, CA 94035-1000 USA

ABSTRACT: *J90 systems have the potential to provide cost-effective computing. By harnessing the processing power of several J90 systems with a parallel application, a potential for greater processing power can be achieved with one application. The Numerical Aerodynamic Simulation (NAS) Facility at NASA Ames Research Center has installed four J90 systems in order to construct a J90 cluster. The Portable Batch System (PBS), has been installed to harness the processing potential of all four systems in the cluster. This paper discusses the hardware and software configurations of the cluster as well as its operational aspects and capabilities.*

Parallel processing has demonstrated its potential for achieving high performance for certain applications. However, not all applications can be parallelized and an efficient and cost-effective means of processing conventional vector/scalar applications along with parallel applications is needed. Configuring multiple cost-effective systems in a cluster can potentially provide a mechanism for processing both conventional and parallel applications.

1 The Newton Project

The Newton Project was conceived to provide a production-quality resource to NAS users that supports a transitional path from current High Speed Processing methodology into an increasingly parallel scientific computing paradigm. The Newton Project was initiated to provide a platform for the development of parallel algorithms and methods that will have an impact on critical NAS/NASA projects.

In the first phase of the Newton Project, proposals were requested that would help delineate the transitional pathways to parallel processing from conventional High Speed Processing. Application development projects accepted for Newton are required to have real world applicability and relevance to industry and/or the scientific community.

The overall project goal is to successfully transition workhorse C90 sequential Computational Fluid Dynamics application codes to operate in a parallel environment at a price performance level better than that available in the workstation market.

2 Cluster Hardware Configuration

The NAS J90 cluster consists of four J90 systems. Table 1 shows the configuration of each of the 4 nodes within the Cluster.

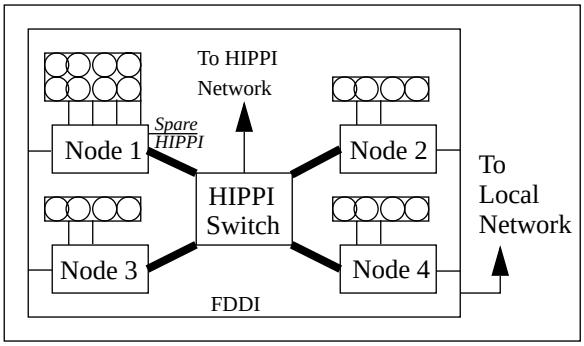
Table 1. Node configuration

	Node 1	**Node 2**	**Node 3**	**Node 4**
Disk	72 GB	36 GB	36 GB	36 GB
CPU's	8	4	4	4
Memory	128 MW	128 MW	128 MW	128 MW

Node 1 has been given more processors and disk space in order to serve as the "master" node in the cluster. As the "master" node, it will have the responsibility of interactive use as well as remotely serving the filesystems to the batch-only nodes. To keep Node 1 from becoming overloaded, only four processors are made available for batch processing. The remaining four processors are set aside for compiling, interactive usage, and performing network access to the disks. Nodes 2 through 4 are designated for batch processing only. Figure 1 shows the layout of the clusters connectivity. Each node is connected through a HIPPI switch as well as over FDDI.

The primary node of the cluster has 72 GB of SCSI disk connected across four data paths. Nodes 3 through 4 each have 36 GB of SCSI disk connected across two channels.

Figure 1: Cluster Layout



3 Cluster Software Configuration

One goal of a cluster is to control costs. The fact is, software is expensive but necessary. Having the same software on all nodes within a cluster increases the licensing costs. However, every software package is not required on all nodes. Compilers can be consolidated on a single node saving the licensing costs on the batch processing nodes. However, extra steps are required to still permit compiling but to control it so that it happens only on a single node. Since node 1 contained 4 additional processors and was to be used for interactive work, it made sense to place all the compilers on node 1 and provide a batch queue to allow compiling on node 1 only.

Not all packages can be configured this way. An example of this is the Message Passing Toolkit (MPT). As a cluster, jobs can be initiated on any node and can execute across multiple nodes. Because of this, the components of MPT are required on all nodes within the cluster.

4 Batch System

The Portable Batch System (PBS) was chosen for batch processing on the cluster. Not only did this save the cost of purchasing the Network Queueing Environment and its source license, it provided a flexibility easily adapted to a cluster. PBS is designed to operate in a distributed environment suitable for cluster computing. One of the appealing features of PBS is the ability for each site to develop their own batch scheduler. Included with PBS are application frameworks and application programming interfaces for TCL/TK and C. The flexibility afforded in constructing a batch scheduler simplified the ability to fully utilize the cluster without over-allocating its resources.

4.1 Batch Configuration

Four production queues were required for the cluster. Table 2 shows the memory and wallclock limits on the queues.

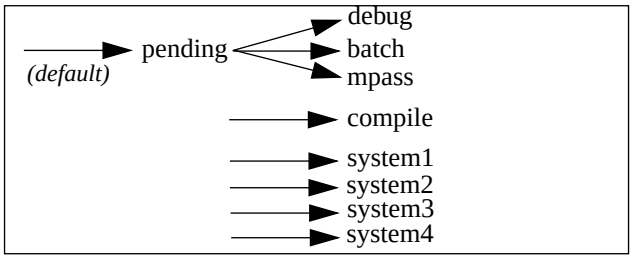
In addition to the four production queues, there exists one system queue per node of the cluster for the purpose of running system maintenance jobs.

Each queue within the cluster's configuration is designed to process specific types of batch jobs. The `debug` queue is designed to allow the customers to perform debugging operations on their jobs without having to wait for the backlog of normal running jobs to complete. Steps have been take to prevent resources from being allocated to more than one batch job. Therefore, if the resources are not available, the debug job will wait.

Table 1. Queue Limits

Queue	Memory	Walltime
compile	20 MW	30 minutes
debug	400 MW	15 minutes
mpass	400 MW	2 hours
batch	100 MW	2 hours

Figure 2: Queue Configuration



The `batch` queue is designed to run single-tasked jobs. Jobs in this queue are run only when there are no jobs queued or running in the `debug` or `mpass` queues. If a job is running in the `batch` queue and other jobs appear in the `debug` or `mpass` queues, the job will be checkpointed and will again wait until no other jobs can be run. Essentially, jobs in this queue are run only when the cluster is idle.

Parallel jobs are run out of the `mpass` queue. Jobs in this queue have access to 16 CPUs across 4 nodes in the cluster. Parallel work is performed with MPI, PVM, HPF, and multi-tasking.

Since the cluster's configuration has compilers only on the first node, batch compiles must also take place on the first node. The `compile` queue is designed to allow batch compiles to be performed only on the first node in the cluster.

4.2 Allocating Processors for Batch Jobs

While MPI and HPF allow for the specification of individual processors, PVM allows for the specification of hosts to use. The requirements of the job for processors or whole nodes need to be specified as resources for the job at the time of submission. To accomplish this, two job attributes are available. The `neednodes` attributes allows for the specification of entire nodes, whereas the `cpus` attribute is used to indicate individual processors. Customers can then inform the batch system exactly how much of the cluster they will require, allowing for the processors to be distributed without over-allocation.

5 Parallel Processing

When constructing a cluster to perform parallel processing, the types of parallel jobs to run must be considered. The NAS J90 cluster has the requirement to run four types of parallel jobs:

- Message Passing Interface (MPI)
- High Performance Fortran (HPF)
- Parallel Virtual Machine (PVM)
- Multitasking

5.1 Message Passing Interface

MPI allocates processes for each parallel task. Therefore, it is possible to run a 16-processor parallel job across the four nodes within the cluster. In the cluster's configuration, there are 16 processors for batch processing, 4 per node. Therefore, only 4 tasks would be allocated per node for a maximum of 16 tasks.

For a batch job, the customer would specify how many CPUs are required. This is accomplished by `-lcpus=##` where ## is the number of CPUs requested. This can also be placed within the batch script as a directive: `#PBS-lcpus=##`.

MPI uses a "machines" file to indicate where processes are to be started. Each line of the machines file is an MPI process. Therefore, if more than one MPI process were to be started on a single host, the hostname must appear once for each MPI process. The batch scheduler will automatically generate a "machines" file for the batch job. The number of MPI processes is set to the number of entries in the "machines" file.

This will change in an upcoming release of MPT. It will be possible to use MPI with shared memory libraries which will increase the performance of the job within a node. Additionally, the evolution of this will allow for shared memory within a node and traditional communications between nodes. This will mean that MPI jobs will need to specify how many tasks within a node, along with how many tasks overall.

5.2 High Performance Fortran

HPF works similarly to the traditional MPI. Jobs using HPF allocate tasks on the processor level. Therefore a customer would specify how many processors are required for the batch job.

For a batch job, the customer would specify how many CPUs are required. This is accomplished by `-lcpus=##` where ## is the number of CPUs requested. The batch scheduler will generate a "machines" file for the batch job for the processors to be used.

5.3 Parallel Virtual Machine

PVM's unit of allocation is on the host level. Therefore, PVM jobs must indicate the number of hosts on which the job will run.

For a batch job, the customer would specify how many nodes they require. This is accomplished by `-lneednodes=##` where ## is the number of nodes requested.

PVM uses a "hosts" file which contains all the hostnames for systems on which to start the `pvmd3` daemon. The file contains one entry per host on which it can run. The batch scheduler will automatically generate a "hosts" file for the batch job with the nodes to be used.

5.4 Multitasking

Multitasking is a form of parallelization. Although restricted to a single node, the job can parallelize across all processors within the node.

Multitasking jobs need to be given a single node. This is accomplished by `-lneednodes=1` to indicate that the job requires a single node in the cluster. However, a multitasking job may only run efficiently if fanned out across a subset of the total processors. A new mechanism is being investigated that will allow for the specification of the number of CPUs to multitask across. The use of this new mechanism implies the entire job must run on a single node. Any CPUs not allocated to the multitasking job can be made available for other processing.

6 Networked File System

A consistent filesystem structure is needed across the cluster. Common filesystems, such as home directories, need to be available on each node in the cluster. This can be accomplished with NFS 2, NFS 3, or possibly DFS. DCE/DFS is discussed in detail in a paper entitled, "DCE/DFS on a Cluster of J90s," by Alan Powers.

7 Issues

Several problems arose while configuring the cluster. These problems ranged from administrative to operational.

7.1 Incorrect Process/Session Limits

MPI, HPF, and PVM use `remsh` to spawn the necessary tasks. The problem with this is that when processes are started with `remsh`, they use the user's interactive udb limits. This causes several problems in a cluster configuration.

The first problem is that the limits set in the batch system are not carried forward to remote tasks. The result is that all spawned tasks will have the limits set by the udb rather than by the batch system. Process and session limits such as CPU time, memory, and accounting information are not set properly.

MPI, HPF, and PVM all use `remsh` to spawn tasks which is hard coded into the packages. One idea to resolve the limits problem is to replace `remsh` with a customized version which would pass additional information and set appropriate limits and fields for remote tasks. SPR 104572 requests the ability to substitute `remsh` with a replacement program.

7.2 Batch Limits

Parallel jobs can run across multiple nodes within a cluster. However, no mechanism exists to pass session resource usage information back to the host where the job initiated. Consequently, the batch memory and CPU time limits for the job cannot be enforced across nodes of the cluster.

7.3 Accounting

Remote tasks do not belong to the same session on each node and are not tracked from the initiating session. Providing accu-

rate accounting information for a single parallel batch job is difficult, therefore, a global session identifier is needed to be allocated across all nodes of the cluster. Accounting records can then be synchronized to provide a true summary of the job.

7.4 Monitoring

The NAS J90 cluster is configured with a single master node and three slave nodes. Although there exists a concentrated effort of monitoring on the master node of the cluster, the slave nodes don't receive the needed attention. Automated tools are required to detect problems.

7.5 Checkpointing

Since HPF, MPI, and PVM use `remsh` to spawn tasks, this leaves pipes open. Checkpointing these jobs will fail with the error EPIPE. A mechanism for checkpointing parallel jobs is needed.

7.6 Inter-Node Latencies

Since parallel jobs will be distributed across nodes in the cluster, the inter-node communication speeds are important. Currently the system is using FDDI for inter-node message passing. Tests have shown that performing message passing over the HIPPI network will improve the performance.

7.7 Disk I/O Performance

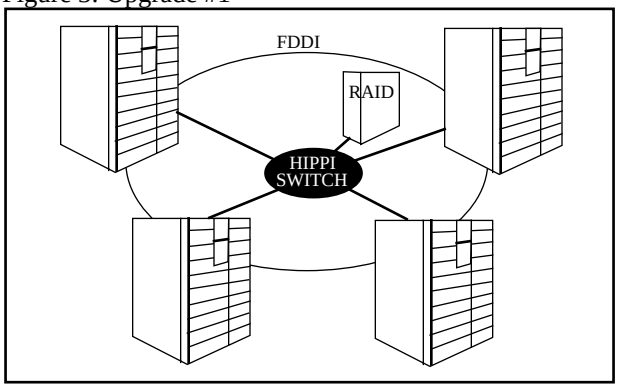
The cluster currently uses a cost-effective SCSI disk and is limited to the performance rates of SCSI disks. Periods of heavy swapping will cause noticeable system delays. Additionally, since there is no SSD to be used for caching, programs which do large amounts of I/O perform poorly. Although memory could be used as a cached device, this would reduce the amount available for normal processing.

8 Future Plans

Several upgrades are planned for the J90 cluster to improve parallel job performance.

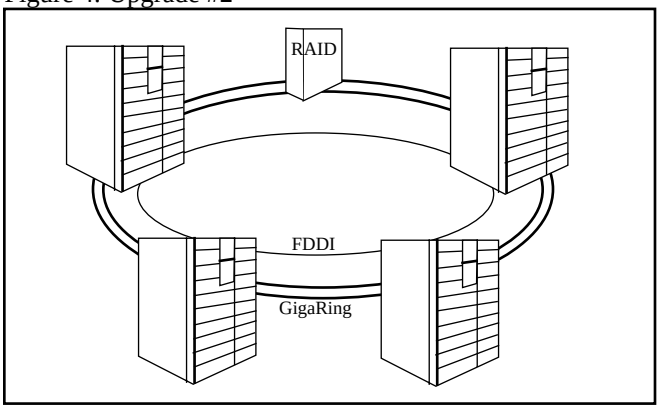
The first planned upgrade is to add a HIPPI RAID device to a HIPPI switch. This device would then be shared, possibly with the Shared File System (SFS), with all nodes of the cluster.

Figure 3: Upgrade #1



The second planned upgrade is to convert the nodes to J90se's with GigaRing. While the J90se's will provide improved scalar performance, the GigaRing should drastically improve the system's I/O. An additional benefit to converting to the GigaRing, is that cost-effective Fiber Channel disks become available.

Figure 4: Upgrade #2



9 Summary

While the basic functionality of cluster computing exists, the software to fully and effectively utilize a cluster is lacking. Additionally, delays in hardware delivery affect the ability to provide cost-effective computing. Rather than invest in cost-effective peripherals, a much higher investment must be made to utilize the cluster.

Although the hardware supports several communication interconnects such as FDDI and HIPPI, the most flexible interconnect, the GigaRing, has been delayed. The GigaRing would have offered improved inter-node communication performance as well as the ability to obtain cost-effective peripherals.

Disk I/O performance is slow. However, the GigaRing would have made high performance fiber channel disks available at an affordable price.

Message passing libraries are evolving and improvements can be seen with each new release. However, it seems that the corporate focus has been on the T3E systems, causing advances in cluster computing to be put on hold.

Even though the cluster environment is not perfect, parallel work is being accomplished. Chart 1 shows the aggregate MFLOPS per CPU for the cluster.

This graph represents the performance of user codes in terms of MFLOPS. The data was obtained from HPM data and does not reflect actual overall system performance. Using these performance numbers as a baseline, it is possible to compute the performance of the cluster in terms of GigaFlops. Using the MFLOPS/CPU and extrapolating them to 20 CPUs, a measurement of the cluster capability can be derived. Chart 2 shows the GFLOP rating of user codes when extrapolated to 20 CPUs.

Timings from an MPI code running across increasing numbers of processors shows that turnaround time is increased when multiple nodes are required. Chart 3 shows the timings obtained on the cluster.

Chart 1: MFLOPS/CPU

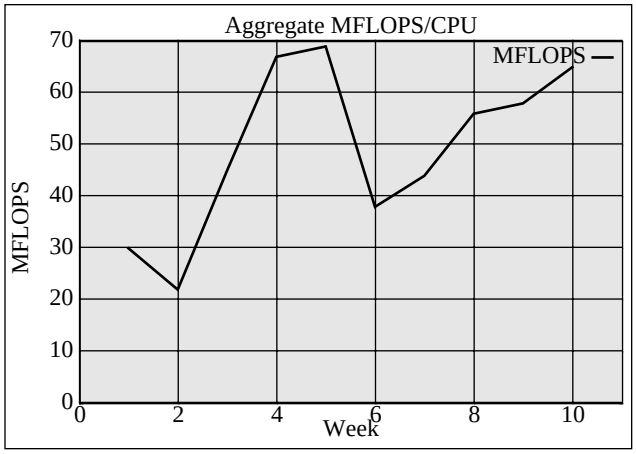
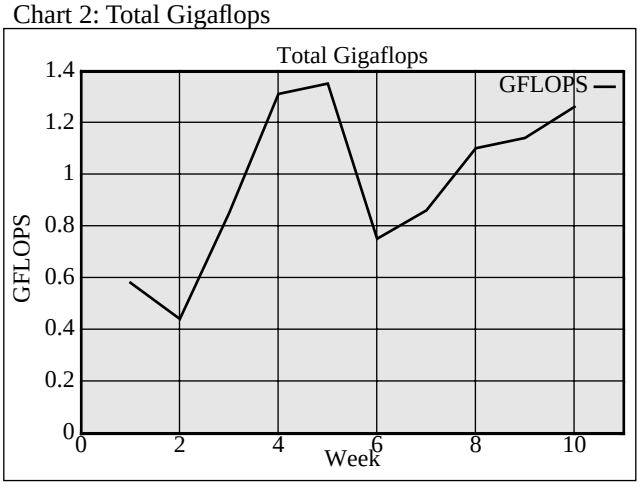
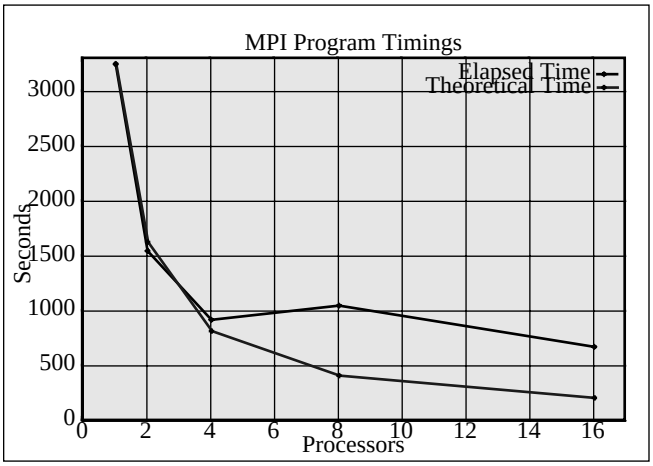


Chart 2: Total Gigaflops



The chart contains two data lines. The first data line shows the actual timings of the code. The second data line represents the theoretical timings that could be obtained by doubling the number of processors. For example, the theoretical number for 4 processors would be half the time for 2 processors. When going from 4 to 8 processors, the turnaround time increases. This shows the boundary of going from 1 to 2 nodes. While running within a single node, the actual numbers follow closely to the theoretical numbers.

Chart 3: Timings



The Numerical Aerodynamic Simulation Facility (NAS) at NASA Ames Research Center is actively pursuing the futures of parallel computing. United States sites are welcome to visit NAS on the World Wide Web at `http://www.nas.nasa.gov`. The parallel efforts at NAS can be found at `http://lovelace.nas.nasa.gov/Parallel/home.html`. The Newton Project's home page can be found at `http://lovelace.nas.nasa.gov/Parallel/Newton/index.html`. The author can be reached at `cardo@nas.nasa.gov`.

10 Acknowledgments

Original text for Newton Project overview was written by George Myers. The Charts for MFLOPS/CPU and Total Gigaflops were originally produced by George Myers.

The Chart representing timings was produced from data derived from the efforts of Thomas Faulkner.

Building a J90 Cluster

Nicholas P. Cardo
Sterling Software, Inc
NASA Ames Research Center
Numerical Aerodynamic Simulation
Moffett Field, CA
cardo@nas.nasa.gov
http://www.nas.nasa.gov

Cluster Hardware

	Node 1	Node 2	Node 3	Node 4
Disk	72gb	36gb	36gb	36gb
CPU's	8*	4	4	4
Memory	128mw	128mw	128mw	128mw

** Only 4 available to the batch system*

The Newton Project

The Newton Project was conceived to provide a production quality resource to NAS users that supports a transitional path from current High Speed Processing methodology into an increasingly parallel scientific computing paradigm. The Newton Project was initiated to provide a platform for the development of parallel algorithms and methods that will have an impact on critical NAS/NASA projects.

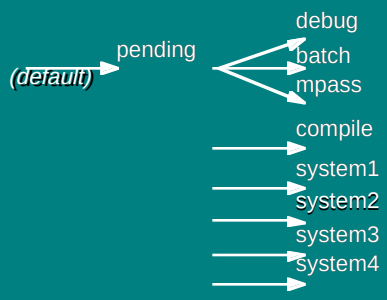
Software Configuration

- Portable Batch System
- Message Passing Toolkit
- Fortran Compiler (Node 1 only)
- Standard C Compiler (Node 1 only)

The Newton Project Goal

The overall project goal is to successfully transition workhorse C90 sequential Computational Fluid Dynamics (CFD) application codes to operate in a parallel environment at a price performance level better than that available in the workstation market.

Batch Queue Configuration



Batch Queue Limits

Queue	Memory	Walltime
compile	20mw	30 min
debug	400mw	15 min
mpass	400mw	2 hours
batch	100mw	2 hours

HPF

- Specify # cpus *(- `lcpus=##`)*
- Maximum of 16 cpus
- "machines" file generated by batch scheduler

Parallel Processing

- Message Passing Interface (MPI)
- High Performance Fortran (HPF)
- Parallel Virtual Machine (PVM)
- Multitasking

PVM

- Specify # of nodes *(- `lneednodes=##`)*
- "hosts" file a generated by batch scheduler

MPI

- Specify # cpus *(- `lcpus=##`)*
- Maximum of 16 cpus
- "machines" file generated by batch scheduler
- Looking forward to shared memory within node and tcp between nodes

Multitasking

- Specify 1 node *(- `lneednodes=1`)*
- Future: `-lncpus=#`

Future Plans

- Add HIPPI attached RAID
- Upgrade to J90se with GigaRing

Issues

- Incorrect process/session limits
- Batch limits not enforcable
- Cluster wide accounting
- System monitoring
- Parallel job checkpointing
- Inter-Node latencies
- Disk I/O performance
- ??? GigaRing ???

For More Information

NAS is on the web...

```
http://www.nas.nasa.gov
http://lovelace.nas.nasa.gov/Parallel/home.html
http://lovelace.nas.nasa.gov/Parallel/Newton/index.html
```