The epic of Gilgamesh: CREWES' new cluster computer

Kayla Bonham, Kevin Hall, Robert J. Ferguson

ABSTRACT

Gilgamesh is the new parallel Linux cluster computer recently purchased, installed, and put into service by the CREWES project at the University of Calgary. Gilgamesh consists of a vertical rack and power supply, containing multiple 1U servers manufactured by Super Micro of Santa Clara, California. The architecture of the system is of a 6U master node with 14 TB of disk capacity, acting as gateway and file server to 18 identical 1U worker nodes with smaller local disks.

Each node has two quad-core 2.66 GHz Intel XEON 5400 CPU chips running at 2.66 GHz, a two-level cache, 16 GB RAM, and 300 GB local disk. This gives the cluster a total of 152 CPUs, 300 GB RAM and 20 TB disk storage. The estimated combined raw CPU performance (local memory access, minimal I/O) is approximately 1 TFLOPS.

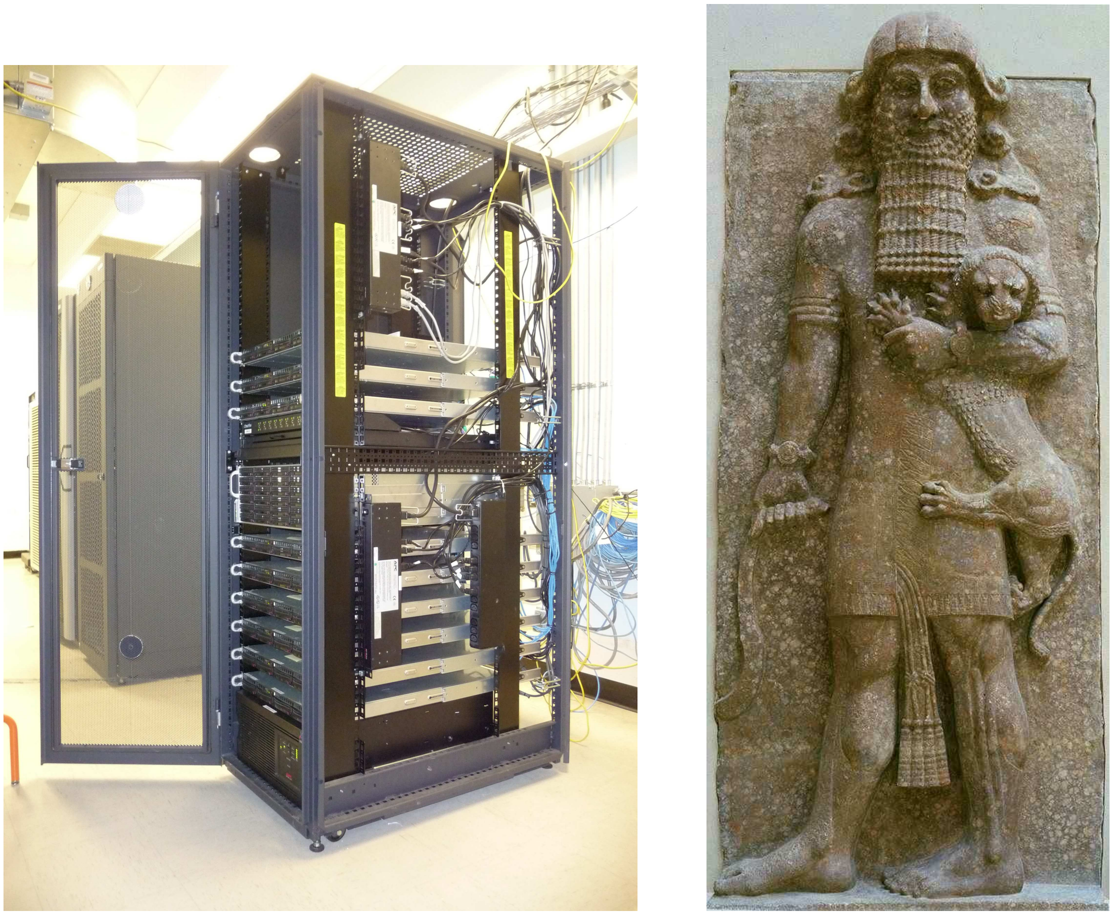
Gigabit Ethernet is the communications backbone. CentOS 5 Linux is the operating system on all nodes. The rack has space for future expansion and upgrades. Power and cooling capacity within the server room are issues of concern regarding upgrades.

Gilgamesh is providing CREWES faculty, staff and student researchers with high throughput data processing using commercial and locally-developed software packages and provides a test platform for exploring and developing parallel algorithms for 3D geophysical modelling applications.

INTRODUCTION

At the initiative of Rob Ferguson and Gary Margrave, funding for a highly parallel computer was assembled in 2007-2008. Proposals from several hardware vendors were solicited, aiming for the best balance of the number of high-speed processors, amount of shared memory among multiple processors, disk storage capacity and high-throughput interconnection. The ability to run familiar software packages already in use by CREWES was also considered essential. It became obvious that the solution offered by Super Micro Computer of Santa Clara, California would give an excellent combination of numbers of high performance processors, memory, disk storage and communications capacity, and the ability to run a commonly available software platform (Linux), at an excellent price/performance ratio.

The system was purchased, configured and delivered by PowerByte Systems in the spring of 2008. We have been working to acquire, install and put into service a variety of software systems, language environments, and packages that will make the parallel processing capabilities of the system conveniently available to CREWES researchers. Figure 1 shows an oblique view of the system in its 19 inch 42U vertical rack, beside its namesake, the mighty *King Gilgamesh of Uruk.*



(a) Gilgamesh in a rack.

(b) Gilgamesh in Iraq.

FIG. 1. Gilgamesh: modern vs. mythic.

MOTIVATION

Prior to the acquisition of the Gilgamesh cluster, CREWES project members had been using a 1U rack-mounted Pentium-based Linux cluster known as *"Impala"* (Maier et al., 2003). Impala had a master node with one 3 GHz Pentium 4 CPU, 2 GB RAM, and 300 GB disk. There were ten worker nodes with one 3 GHz CPU each, and 2 GB RAM. Although occupying more rack space than Gilgamesh, Impala's performance in total processing power, disk capacity, memory, I/O and communication can be matched by just one worker node in the new Gilgamesh cluster. However, it did serve its purpose well in its time.

CREWES members also have a variety of single- and dual-CPU Unix and Windows-based file and compute servers available, plus many desktop and laptop PCs, on which to do computational research.

Certain considerations led to the decision to acquire a high-performance cluster computer exclusive to CREWES. Reasons for CREWES to have a high performance cluster computer of its own include:

- A CREWES-controlled facility can be dedicated to running software pertinent to geophysical research and the cluster can be a testbed for software architectures, libraries and tools suited to geophysical computing.
- Security: CREWES research often involves processing of sensitive proprietary data sets provided by sponsors. Physically having control over access will help protect against disclosure of sensitive data to unauthorized persons
- Large data sets typically reside on physical media such as flash ROMs and DVDs (not to mention 9-track and 8 mm tape drives for archival purposes!) which CREWES staff may load or archive as required without going outside our local network.
- Concentration of expertise: faculty, staff and students can familiarize themselves with a single architecture and computing environment for mutual assistance, education and collaboration, rather than scatter expertise among different environments and facilities.
- With access under control of CREWES, processing can take place without interference from unpredictable loading by non-CREWES users.

NAME

Gilgamesh is the hero of the ancient Sumerian "Epic of Gilgamesh", a work composed c. 2600 B.C., written on clay tablets by later Babylonian/Akkadian cultures, unearthed by archaeologists c. 1840 and successfully translated in 1872.

Certain story elements are thought to have influenced or been directly incorporated into the Hebrew Bible, as well as Homer's Iliad and Odyssey, and other heroic myths. It is renowned as humanity's oldest surviving work of literature (George, 1999).

(a) Tablet 4 of the Epic (partial)

(b) Slaying the Bull of Heaven.

FIG. 2. Assyrian tablet and bas-relief.

In this story, Gilgamesh is the greatest king ever to have lived, the semi-divine ruler of the city-state of Uruk, located in what is now the country of Iraq. He and his trusted friend Enkidu undertake various adventures, but in their prideful blunders they offend the gods. They endeavour to make peace with the gods and safeguard the kingdom. Enkidu dies, and, after first attempting to bring his friend back from the realm of dead souls under the earth, Gilgamesh goes on a quest for the secret of immortality. He doesn't achieve that, but on his eventual return to his kingdom, has understanding of how to be a good and just king and live a full life.

Gilgamesh seems an appropriate name for a heroic, all-conquering problem solver, interpreting layers of meaning inscribed in the clay, searching for sources of knowledge and wisdom in the sands of time, below the earth and the sea. Besides, the Greek, Saxon and Viking epics have all been done!

Figure 2(a) shows a fragment of the story, found in the ruins of Ashurbanipal's palace at Nineveh. Figure 2(b) shows Gilgamesh preparing to hurl the dismembered haunch of the Bull of Heaven at the goddess Ishtar (no wonder the gods were insulted).

ARCHITECTURE

The original *Beowulf* [*] clusters were based on desktop PCs connected by an ordinary off-the-shelf local network (10 Mbps Ethernet) each with its own memory, disk storage, and independent operating system (Brown, 2004).

The technology of high speed CPUs, multi-level memory cache, multi-core processors, rack-mounted multi-server systems and fast data communications, has greatly increased

[*] named for the monster-slaying Norse hero, the *Gilgamesh* of his day!

the overall computing capacity of so-called "commodity computer" -based clusters. Performance has increased in every sector, however, modern cluster computers now have internal architectural heterogeneity which may be tricky to exploit optimally.

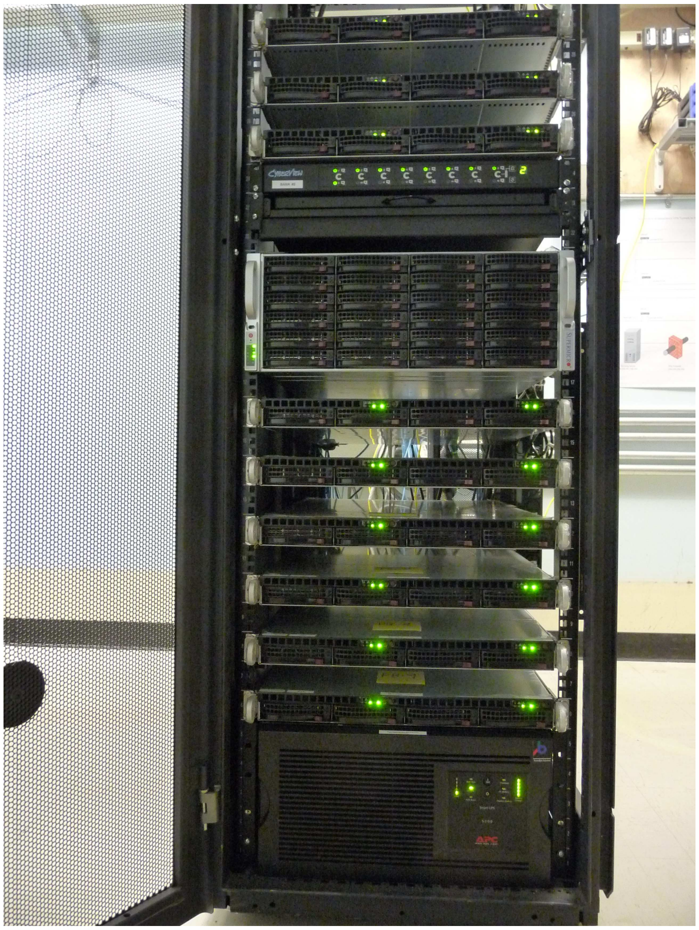


FIG. 3. Gilgamesh, front view.

Gilgamesh (figures 3 and 4) currently comprises 19 nodes (one master and 18 worker nodes) interconnected by a 1 Gigabit/s asynchronous network backbone (Gigabit Ethernet, or gigE). The master node is a SuperServer 6015TW-T system, manufactured by SuperMicro Computer, Inc. The worker nodes are identical model Supero X7DVL-E units also from SuperMicro. (SuperMicro, 2007a), (SuperMicro, 2007b)

Each 1U slot in the rack holds two worker nodes side-by-side, each with its own independent motherboard. Each motherboard contains two quad-core 64-bit Intel XEON 5400 chips running at 2.66 GHz, with a 1.33 GHz front-side bus. Each core has 32 kB of level 1 cache and shares 4 MB of level 2 cache with the other 3 cores on the chip. A node has 16 GB of RAM shared among all 8 processors on that board. A worker node has 300 GB mass storage on two internal hard disks. The local file system is intended to be used for temporary storage, to avoid loading the gigE backbone with ordinary file I/O traffic.

Figure 5 is a schematic drawing showing the architecture of CPUs, cache and memory on a node.

The master (primary) node has a single full-width motherboard and a 4U vertical profile (occupying four full-width 1U slots vertically in the rack). The master node's architecture is similar to that of the worker nodes (i.e. two quad-core XEON 5400 chips, 16 GB RAM, 2-level cache etc.). As the primary file server for the cluster, it has a total capacity of 14 TeraBytes on twenty-four 750 GB disk drives in a RAID 6 configuration (RAID = Redundant Array of Independent Disks). The master's file system is shared with workers, using NFS(Networked File System).

This makes a total of 152 CPUs, 300 GB RAM and 20 TB disk storage for the entire cluster. Based on the clock speed and test results from other installations, the overall system should be able to achieve up to 1 TeraFLOPS (10^{12} FLoating point Operations Per Second) in raw CPU performance, assuming negligible interference from disk and communication channels.

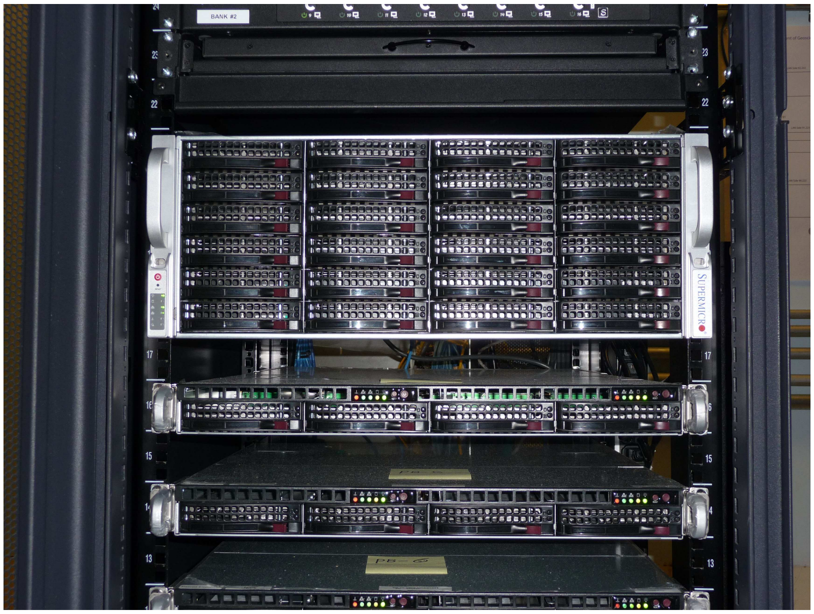


FIG. 4. Closeup of the master node and two side-by-side worker pairs

If communication throughput becomes a limiting factor, the servers have the capability to use a 20 Gb/s InfiniBand interconnection (also known as "switched fabric"), which is the communication backplane of many parallel cluster computers. This would require additional hardware, which the project has no current plans to acquire.

A simpler way to upgrade the interconnection throughput would be to add a second gigE local network, for which built-in interfaces already exist, and use software 'striping' to combine them both into a single channel of double the speed.

PARALLEL APPLICATION STRUCTURE

Considering that each node within Gilgamesh is an 8-CPU shared memory parallel computer in its own right, just one node of the *Gilgamesh* cluster is almost as powerful as our entire previous *Impala* cluster. Linux can assign 8 processes or tasks within a process, to its own CPU. Inter-process communication software can utilize the shared memory for very high speed local communication. Communicating among more than eight processes requires using the gigE local network, and is slower than shared memory, but faster than communicating with machines outside the rack.

Embarrassingly parallel program distribution

On a system like Gilgamesh, one could reasonably utilize “embarrassingly parallel processing” by taking an ordinary sequential (non-parallel) program or algorithm that one usually runs on a single standalone computer, replicating 19 independent copies of it, each with different parameters or input files: one copy per node, and get a perfect 19-times speedup relative to running them one at a time serially, as long as they don’t compete for gigE bandwidth, or access to the primary file system disks on the master node.

In fact, one would probably obtain almost as good (linear) speedup by treating *all* of the CPUs on all of the nodes as independent computers, and run 152 copies of our serial program, for a 152-times speedup, again as long as they don’t compete *too much* over gigE bandwidth, access to the master node’s file system disks or access to the local worker node’s disks. Many problems in geophysics are parallelizable in this fashion, eg. frequency-by-frequency depth migration, trace-by-trace or shot-by-shot seismic analysis, parameter sensitivity testing.

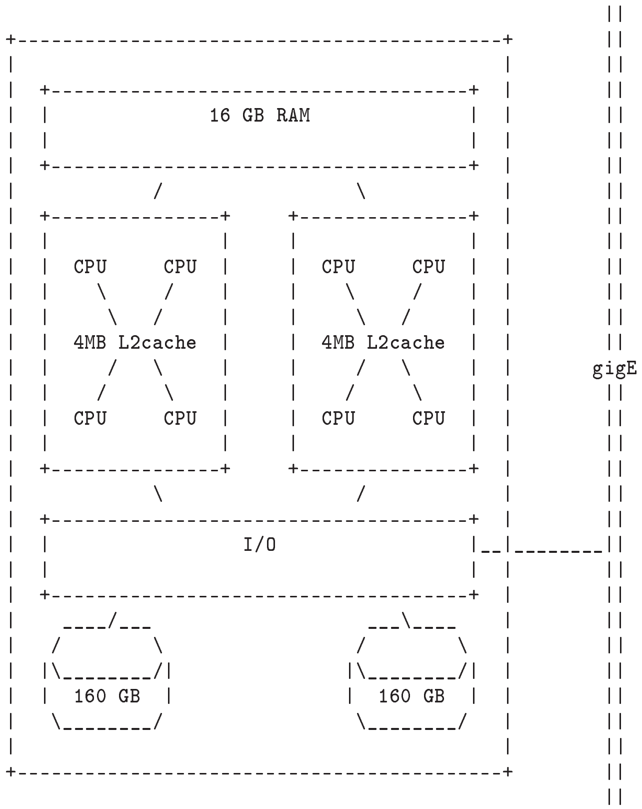


FIG. 5. Node architecture schematic

Communicating multi-task architecture

More sophisticated usage of parallel computing involves software algorithms that are split into multiple cooperating tasks, that communicate with each other and work in parallel, using MPI (Message Passing Interface), PVM (Parallel Virtual Machine) or other software infrastructure. Tasks asynchronously forward their results to a designated collection point for output, display, or further processing.

Communications overhead is a consideration when making sure all the separate tasks receive data, interact without deadlock, and assemble results correctly upon completion. Certain algorithms are easily expressed in this type of explicitly parallel organization, but they can be much more difficult to write, debug, and describe, than programs with only a single, serial control path.

CREWES intends to use parallel multitasking constructs in MATLAB, C/C++, Fortran, (e.g. using the "`parfor`" construct in both MATLAB and Fortran 90, explicit send-receive-reply or remote procedure calls for exchanging messages between processes in any language) built on top of MPI, PVM, and other process communication support systems when we need to code parallel algorithms for high performance.

Parallel libraries called from sequential programs

Intermediate between the two forms is the case where the user's program is expressed as a sequential process (suitable for execution on a single processor), but is designed to use one or more 'canned' software libraries, that have parallel processing support built in, and detect and use parallel computation where available. Many highly-parallelizable functions, such as matrix operations and sparse array manipulation libraries, are already implemented like this in MATLAB, as are certain numerical packages callable from Fortran, C, Perl, etc.

The bottom line is still to apply the highest number of available CPU cycles to solve a problem within as short a time as possible, or conversely, to solve as big a problem as practical in the time one is willing to wait for an answer.

Vendors and providers of numerical analysis software packages invest much effort getting their systems to use parallel algorithms on high performance computing systems where available. With this approach we are able to benefit from their hard work without having to develop a lot of parallel code ourselves.

EFFICIENCY CONSIDERATIONS:

Heterogeneity at both cluster and node level favours breaking down problems to take advantage of shorter, more efficient data paths where possible. For example, it makes sense that if certain tasks of a program communicate more closely than others, one should assign those tasks to CPUs within the same node, so that they can use shared memory communication, rather than on separate nodes which would require use of the gigE network. If <*N*> tasks use the same executable program, assigning them to CPUs within one node could take advantage of loading the executable file over the network once instead of <*N*> times. This opens the possibly of all <*N*> sharing a common copy in the program cache, for faster loading and swapping. On the other hand, if various tasks in a parallel program all use the local disk for temporary files, then it may make sense to explicitly send them to separate nodes, so that the file I/O can occur without interference.

The expectation is that cluster users will have a home directory on the cluster server (master node), containing the data they wish to work on. When users launch a distributed software run, their data should be copied to temporary storage area (`/tmp`) on the worker nodes that will carry out the computation, and the computation will use the local copy of the data as much as possible. `/tmp` is persistent, so data for subsequent runs of the same program may already be resident on the worker node.

SOFTWARE

The operating system on Gilgamesh is CentOS Linux, one independent instance per 8-CPU node. Eight user processes can run simultaneously on a node, and a "load average" of 8 would mean the system is running at 100% capacity. Of course, it is harmless(but slower) to run more than 8 processes. Then normal Unix time-sharing will occur, allocating each active process its share of the 8 available processors.

The Gilgamesh cluster behaves as a cooperating network of 19 independent CentOS Linux systems. Computations will distribute themselves over an appropriate number of nodes in a virtual cluster using MPI (Message Passing Interface), PVM (Parallel Virtual Machine) or other software infrastructure via `ssh`, (the secure shell), either manually by the system manager or user, automatically with a script, or with the aid of a time reservation or job-level scheduling system.

The master node's processors are available to the application, but relying on them is discouraged in general, as everyone's interface to the system is through the master node, and heavy computational processes will only slow down the users' interactive editing/viewing sessions.

Parallel programming environment and application packages:

- parallel MATLAB (comprising core MATLAB extended for parallel computation using the parallel computing toolkit PCT, and distributed computing service MDCS)
- MPI (Message Passing Interface) – communication package
- PVM (Parallel Virtual Machine) – parallel computation package
- Fortran `gfortran, f77, f90, f2003` – the ever-favourite scientific programming language
- ProMax – seismic data processing package
- Tiger (SINTEF) – a 3D finite-difference modelling package
- C / C++ (`gcc, g++`), also Java – system programming languages)

- Perl, Python, Ruby ... – data manipulation script languages
- HTML, CGI, XML ... – web scripting/GUI description languages

PHYSICAL LAYOUT

Gilgamesh currently embodies 19 nodes (one master and 18 worker nodes) mounted in a single 19 inch 42U vertical rack. The master server with its swappable disks is mounted at mid-level in a standard 19-inch rack, occupying 4U of vertical space. Worker nodes are apportioned in the 3 slots above and 6 slots below the server, at 1U spacing, two per case, thus occupying 9 full-width 1U slots. An ingenious sliding LCD keyboard and monitor, switchable among the various nodes, acts as a versatile system console (figure 6).

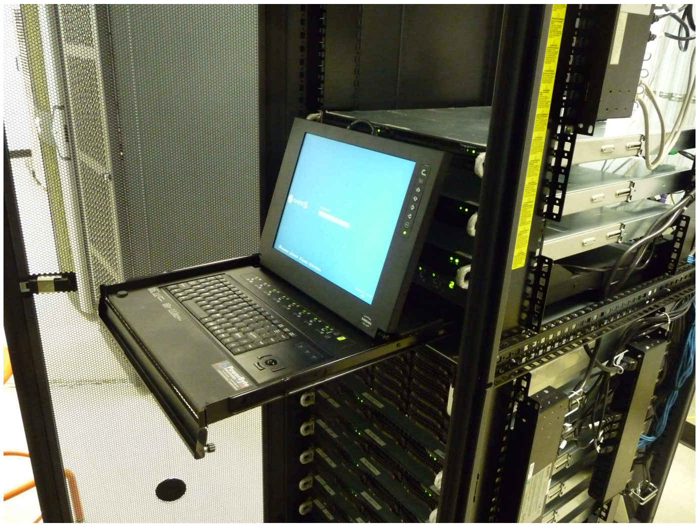


FIG. 6. Sliding 2U flip-up KVM system console.

The rack could potentially accommodate another twenty 1U servers (40 worker nodes) though the electrical and cooling capacity in the server room would certainly need upgrading first. We are already stressing the cooling capacity of our chilled water room coolers with the present collection of compute servers. When all 152 CPUs are operating at full capacity, one can watch the main Gilgamesh node's on-board temperature sensor, as well as the server room temperature sensor, begin climbing precipitously.

GATEWAY NODE SPECIFICATIONS

- package: SuperMicro SuperServer 6015TW-T
- mother board: X7DWT-INF server board,
- processors: dual 2.66 GHz intel 64-bit xeon LGA 771 quad-core processors
- front side bus speed 1333 MHz
- memory: eight 240-pin DIMMs for up to 64 GB DDR2 800/667/533 memory
- on-board ATI 16 MB ES1000 graphics controller (2D graphics accelerator)

- 4 x 1 Gbit LAN ports (by "ESB 2 south bridge")

WORKER NODE SPECIFICATIONS

- mother board: SuperMicro Supero X7DVL-E
- processors: dual 2.66 GHz Intel 64-bit xeon LGA 771 quad-core processors
- front side bus speed 1333 MHz
- memory: six 240-pin DIMMS for up to 16 GB DDR2 667/533
- memory on-board ATI 16 MB ES1000 graphics controller (2D graphics accelerator)
- 2 x 1 Gbit LAN ports (by "ESB 2 south bridge")
- serial ATA (SATA) disk interface 3 Gbps

PERFORMANCE

Reviewers have measured a dual quad-core Intel 5400 Harpertown 3.0 GHz system, similar to our Super Micro boards, at more than 60 GFLOPS on the standard LINPACK benchmark, and “could easily achieve 70-80 with fine tuning” (“techno23”, 2008). Extrapolating from that, a full 19-node system could perform up to 1.1 TeraFLOPS on a purely CPU-intensive problem with perfect scaling and minimal communications and I/O interference. Our CPUs are clocked somewhat slower, so we would likely need some of that fine tuning to achieve 1 TFLOPS on this test, using the entire cluster. Though it will be interesting to see how gilgamesh performs on standard benchmarking tests, our real interest is making geophysical applications run faster, and on ever bigger problems.

SUMMARY

CREWES has acquired what is an enormous leap forward in processing capacity for a very reasonable financial outlay. It is instructive to recall the computing facilities CREWES has used in past years (Maier et al. (2003), (Hall and Maier, 2004)). All funding for Gilgamesh came from pooling the resources of grant holders participating in the CREWES, FRP and POTSI projects. We look forward to developing and presenting many improvements in 2D and 3D processing and modelling methods, with the aid of this tool, over the coming years.

From the moment a computer is built it starts to become obsolete. In the original *Epic*, the mighty Gilgamesh learns to accept his eventual mortality. It may be interesting to see over how many years the CREWES Gilgamesh will remain worthy of its name as a high performance computer, before being succeeded by one “whose merest operational parameters we are too puny to calculate,” to paraphrase the fictional *Deep Thought* (Adams, 1979). But for now, we at CREWES are very happy to have this powerful tool at our command.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the members and directors of CREWES, FRP and POTSI projects for their enthusiasm and financial support, enabling the *Gilgamesh* acquisition to happen, We also thank Arnold Woelfle of PowerByte Systems for his untiring efforts configuring and refiguring system components to make it technically feasible.

REFERENCES

Adams, D., 1979, The Hitchhiker's Guide to the Galaxy: Pan Books.

Brown, R. G., 2004, Engineering a beowulf-style compute cluster : http://www.phy.duke.edu/∼rgb/Beowulf/beowulf_book/beowulf_book/index.html.

George, A. R., 1999, The Epic of Gilgamesh: Penguin Books.

Hall, K. W., and Maier, R., 2004, Crewes computer systems: an update: CREWES Research Report, **16**, 10.1–10.2.

Maier, R., Hall, K. W., and Bland, H. C., 2003, Crewes computer systems: CREWES Research Report, **15**, 10.1–10.6.

SuperMicro, 2007a, Super X7DVL-E User's Manual, rev. 1.1a: Super Micro Computer, Inc.

SuperMicro, 2007b, SuperServer 6015TW-T/INF User's Manual, rev. 1.0: Super Micro Computer, Inc.

"techno23", 2008, Intel harpertown quad core CPU: http://www.techfuels.com/cpu-components/64-intel-harpertown-quad-core-cpu.html.