IEEE Cluster 2002 (CLUSTER 2002), Chicago, Illinois, September 2002.

The Bladed Beowulf: A Cost-Effective Alternative to Traditional Beowulfs*

W. Feng†, M. Warren‡, and E. Weigle†
{feng, msw, ehw}@lanl.gov

† Advanced Computing Laboratory
Los Alamos National Laboratory
Los Alamos, NM 87545

‡ Theoretical Astrophysics Group
Los Alamos National Laboratory
Los Alamos, NM 87545

Abstract

We present a new twist to the Beowulf cluster — the Bladed Beowulf. In contrast to traditional Beowulfs which typically use Intel or AMD processors, our Bladed Beowulf uses Transmeta processors in order to keep thermal power dissipation low and reliability and density high while still achieving comparable performance to Intel- and AMD-based clusters.

Given the ever increasing complexity of traditional supercomputers and Beowulf clusters; the issues of size, reliability, power consumption, and ease of administration and use will be "the" issues of this decade for high-performance computing. Bigger and faster machines are simply not good enough anymore. To illustrate, we present the results of performance benchmarks on our Bladed Beowulf and introduce two performance metrics that contribute to the total cost of ownership (TCO) of a computing system — performance/power and performance/space.

Keywords: Beowulf, cluster, blade server, RLX, Transmeta, code morphing, VLIW, price-performance ratio, performance-power ratio, performance-space ratio, ToPPeR.

1 Introduction

Because Beowulf clusters [14] have mobilized a community around a standard set of software tools, Bell & Gray [3] note that the economics and sociology of Beowulf are "poised to kill off the other archtitectural lines," e.g., specialized, distributed, shared-memory multiprocessors, and "will affect traditional supercomputer centers as well."

The advantages of Beowulf are clear; the hardware and software result in a two- to five-fold savings in cost as traditional supercomputer centers *explicitly* incur additional costs in space, maintenance, administration, operation, and consulting. The latter four costs are particularly labor-intensive. In contrast, Bell & Gray [3] note that these costs are *implicit* for homegrown Beowulf clusters, where such clusters ride "free" on an institution's overhead [3]. However, not all these implicit costs are necessarily "free." Researchers in high-performance computing and communications (HPCC) must carve out labor costs from their funding in order to work on the labor-intensive aspects of building and maintaining a Beowulf. For example, our 128-CPU Linux cluster took days (and arguably, weeks) to install, integrate, and configure properly and initially required *daily* intervention by the technical staff. After the system stabilized and eventually moved to an ambient-temperature office space (i.e., approximately 80°F), the cluster required weekly to monthly maintenance due to the lack of reliability (or robustness) of the commodity off-the-shelf (COTS) hardware. Thus, while the price/performance ratio of Beowulf is quite good when compared to traditional supercomputers and when price is defined as the cost of acquisition; the advantages of Beowulf, particularly large-scale Beowulf, are not as compelling when price is defined as the the *total cost of ownership (TCO)* [5].

In short, "performance at any cost" is no longer good enough; the biggest issues of this decade will be size, reliability (and indirectly, power consumption), and ease of administration and use. We find support for this argument from "New Challenges for the PostPC Era" [11], an invited talk by David Patterson at IPDPS 2001, and "The Future of the Microprocessor Business" by Bass & Christensen [2]. Over the past two decades, the goals in PC computing (and cluster computing) have been to improve performance and price/performance; and likewise, these are the same two metrics featured for the Gordon Bell Award at SC (formerly Supercomputing) every year. Furthermore, the assumptions in PC computing have been that (1) humans are "perfect" and do not make mistakes during installation, wiring, upgrading, maintenance, and repairing; (2) software will eventually be bug free; and (3) hardware mean time between failure (MTBF) is already very large and will continue to in-

*This work was supported by the U.S. Dept. of Energy's *Los Alamos Computer Science Institute* and *Information Architecture - Linux* Programs through Los Alamos National Laboratory contract W-7405-ENG-36. An extended version of this paper is available as Los Alamos Unclassified Report 02-2582, April 2002.

crease. After two decades of improving performance and price/performance and relying on the above assumptions, the belief is that the key metrics of the post-PC era will be *reliability* and *availability* [2, 11].

With this discussion in mind, we present a novel cluster architecture called the *Bladed Beowulf*. Designed by RLX Technologies and integrated and configured at Los Alamos National Laboratory, our Bladed Beowulf cluster consists of compute nodes made from COTS parts mounted on motherboard blades called RLX ServerBlades$^{\text{TM}}$ (see Figure 1), measuring $14.7'' \times 4.7'' \times 0.58''$. Each motherboard blade (node) contains a 633-MHz Transmeta TM5600$^{\text{TM}}$ CPU [9], 256-MB memory, 10-GB hard disk, and three 100-Mb/s Fast Ethernet network interfaces. Twenty-four such ServerBlades mount into a chassis, shown in Figure 2, to form a "Bladed Beowulf" called the RLX System 324 that fits in a rack-mountable 3U space, i.e., $19''$ in width and $5.25''$ in height.[1]

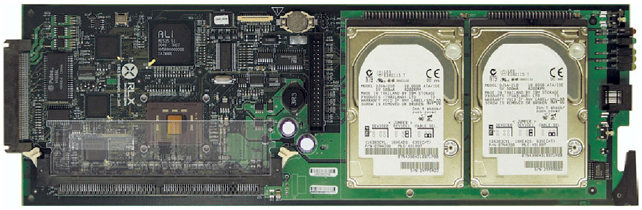

Figure 1. The RLX ServerBlade

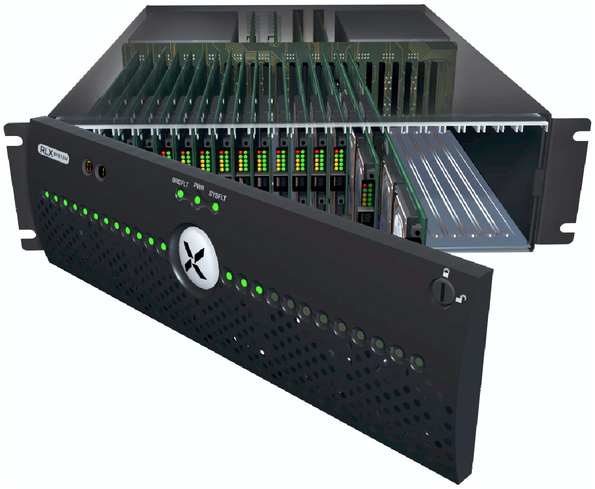

Figure 2. The RLX System 324

The rest of the paper is organized as follows. Section 2 presents the architecture and technology behind our Bladed Beowulf. Then, we present the parallel n-body simulation and treecode library that are used to benchmark our Bladed Beowulf in Section 3. Section 4 presents a performance evaluation of our Bladed Beowulf via a series of benchmarks. With these performance numbers in hand, we then introduce three metrics that relate to the efficiency, reliability, and availability of a system: [1] <u>To</u>tal <u>P</u>rice-<u>Pe</u>rformance <u>R</u>atio (ToPPeR), where Total Price encompasses the total cost of ownership (TCO), [2] performance/space, and [3] performance/power.

2 Architecture of a Bladed Beowulf

Each blade of our Bladed Beowulf consists of a Transmeta Crusoe TM5600 processor, implemented by a 128-bit "very long instruction word" (VLIW) hardware engine and code-morphing$^{\text{TM}}$software (CMS), version 4.2.6. Aggregating twenty-four such blades, referred to as RLX ServerBlades, and interconnecting them with 100-Mb/s Fast Ethernet results in our Bladed Beowulf cluster called *MetaBlade*.

2.1 The Transmeta Crusoe TM5600

In contrast to the traditional transitor-laden, and hence, power-hungry CPUs from AMD and Intel, the Transmeta Crusoe TM5600 CPU is fundamentally software-based with a small hardware core. This TM5600 CPU consists of a VLIW hardware engine surrounded by a software layer called *code morphing*, which presents an x86 interface to the BIOS, operating system (OS), and applications.

2.1.1 VLIW Engine

Having the code-morphing software (CMS) handle x86 compatibility frees hardware designers to create a very simple, high-performance VLIW engine with two integer units, a floating-point unit, a memory (load/store) unit, and a branch unit. Each of the integer units is a 7-stage pipeline, and the floating-point unit is a 10-stage pipeline.

The VLIW engine includes a 128-KB on-chip L1 cache and an integrated 512-KB on-chip L2 write-back cache; an integrated NorthBridge with two memory controllers; and an integrated PCI controller.[2] Each VLIW can be 64 or 128 bits long and can contain up to four RISC-like instructions, which are executed in parallel. The format of the VLIW directly determines how the RISC-like instructions get routed to functional units, thus greatly simplifying the decode and dispatch

[1] While the blade-to-chassis interface is RLX proprietary, the remainder of the cluster is COTS. Note, however, that a recent announcement (Feb. 5, 2002) by HP provides for an open enhancement of the CompactPCI (cPCI) specification to standardize blade servers across manufacturers.

[2] The TM5600 has already been replaced by the Transmeta Crusoe TM5800, and the 4.2.6 version of CMS has been replaced by version 4.3.2, which improves floating-point performance by 25%. The next-generation Transmeta Crusoe increases the VLIW to 256 bits and simultaneously promises less power and a two- to three-fold increase in performance based on simulation results. This performance increase will allow Transmeta to easily surpass the performance of similarly-clocked AMD and Intel microprocessors but at a fraction of the power dissipation.

hardware. And unlike superscalar architectures, the instructions are expected in order, eliminating the need for complex out-of-order hardware which accounts for roughly 20% of the transistor count in a superscalar architecture.

Further, the Transmeta architecture eliminates about 75% of the transistors used in traditional RISC and implements the lost (but inefficient) hardware functionality in its CMS instead. The reasoning here is that implementing a superscalar architecture with out-of-order execution, speculative execution, and predicated execution results in *complex* RISC chips that are highly inefficient, e.g., at any given time, as many as 100 instructions are being executed, of which only a few are the correct ones to execute.[3]

Because modern CPUs are more complex, have more transistors, perform more functions than their early RISC predecessors, and are clocked at much higher speeds; the hardware requires *lots* of power, and the more power a CPU draws, the hotter it gets. The hotter that a CPU gets, the more likely it will fail (or clock down), and likely, cause other components to fail. More concretely, Arrenhius' equation (when applied to microelectronics) predicts that the failure rate of a given system *doubles* with every 10°C (18°F) increase in temperature. And in fact, unpublished empirical data from two leading vendors indicates that the failure rate of a compute node does indeed *double* with every 10°C increase.

At idle, a Transmeta TM5600 CPU by itself generates less than a watt of power while a typical Pentium 4 found in a traditional Beowulf cluster generates as high as 75 watts. At load, the Transmeta TM5600 and Pentium 4 generate approximately 6 and 75 watts, respectively, while an Intel IA-64 generates over 130 watts![4] If the traditional mantra of "performance at any cost" continues, and hence, Moore's law continues, the microprocessor of 2010 will have over *one billion* transistors and will dissipate over *one kilowatt* of thermal energy (see Figure 3); this is considerably more thermal energy per square centimeter than a nuclear reactor!

Because of the substantial difference in power dissipation, the TM5600 requires no active cooling whereas a Pentium 4 (and most definitely, an Itantium or IA-64) processor can heat to the point of failure if it is not aggressively cooled. Consequently, as in our *MetaBlade* Bladed Beowulf (24 CPUs in a 3U), Transmetas can be packed closely together with no active cooling, thus resulting in a tremendous savings in the total cost of ownership with respect to reliability, electrical usage, cooling requirements, and space usage.[5]

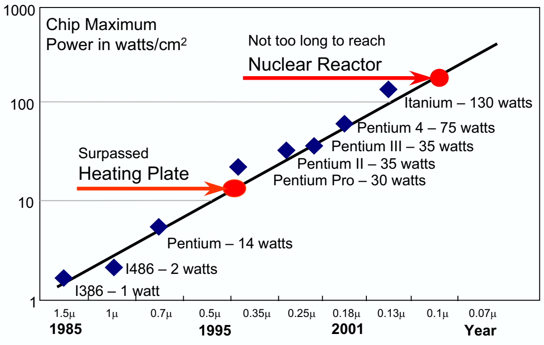

Figure 3. Power Dissipation Per Square Centimeter for Commodity Microprocessors (Source: Fred Pollack, Intel. New Microprocessor Challenges in the Coming Generations of CMOS Technologies, MICRO32)

2.1.2 Code Morphing

The Code Morphing software (CMS) allows Transmeta to decouple the x86 instruction set architecture (ISA) from the underlying processor hardware, thus allowing the hardware to be completely different from the conventional x86 implementation. Furthermore, the underlying hardware engine can be radically changed and not affect legacy x86 code. That is, each new CPU design only requires a new version of CMS to translate x86 instructions to the new CPU's native instruction set. And because CMS typically resides in standard flash ROMs, improved versions can be downloaded into already-deployed processors. This provides two huge advantages over traditional microprocessor fabrication. First, optimizing and fixing bugs amounts to replacing CMS in the Transmeta world whereas it may result in a costly hardware redesign and re-fabricration in the Intel and AMD world. Second, changing to a different instruction set, e.g., from x86 to SPARC, simply involves a change in CMS rather than a complete change from one hardware microprocessor to another.

CMS contains two main modules that work in tandem to create the illusion of running on an x86 processor: interpreter and translator. The interpreter module interprets x86 instructions one at a time, filters infrequently executed code from being needlessly optimized, and collects run-time statistical information about the x86 instruction stream to decide if optimizations are necessary. When CMS detects critical and frequently used x86 instruction sequences, CMS invokes the translator module to re-compile the x86 instructions into optimized VLIW instructions called *translations*. These native translations reduce the number of instructions executed by packing instructions into a VLIW, thus resulting in better performance.

[3] The MIPS R10000 and HP PA-8000 RISC processors are arguably *more* complex than today's standard CISC architecture, the Pentium II.

[4] At the end of 2001, the current-generation Crusoe CPU (i.e., TM5800) *at load* dissipated only 1 watt (on average) @ 366 MHz and 2.5 watts (on average) @ 800 MHz. Even more revealing is that the "TM5800 CPU + Northbridge + Southbridge + Graphics" combination has a thermal design power of only 3.6 watts when playing a DVD [4].

[5] However, due to our adverse 80°F computing environment, we use RLX's small internal fans as a necessary precaution.

Caching the translations in a *translation cache* allows for re-use. When a previously translated x86 instruction sequence is encountered, the CMS skips the translation process and executes the cached translation directly out of the translation cache. Thus, caching and re-using translations exploits the locality of instruction streams such that the initial cost of the translation is amortized over *repeated* executions.

2.2 The RLX System 324: Bladed Beowulf

The RLX System 324 comes in three sets of easy-to-integrate pieces: the 3U system chassis, 24 ServerBlades, and bundled cables for communication and power.

The system chassis fits in an industry-standard 19-inch rack cabinet and measures $5.25''$ high, $17.25''$ wide, and $25.2''$ deep. It features two hot-pluggable 450-watt power supplies that provide power load-balancing and auto-sensing capability for added reliability. Its system midplane passively integrates the system power, management, and network signals across all RLX ServerBlades. The ServerBlade connectors on the midplane eliminate the need for internal system cables and enable efficient hot-pluggable ServerBlade support.

The chassis also includes two sets of cards: the Management Hub card and the Network Connect cards. The former provides connectivity from the management network interface of each RLX ServerBlade to the external world. Consolidating 24 ServerBlade management networks in the hub card to one "RJ45 out" enables system management of the entire chassis through a single standard Ethernet cable. The Network Connect cards provide connectivity to the public and private network interfaces (i.e., data network interfaces) of each RLX ServerBlade. And instead of using 24 RJ45 connectors, i.e., one for each ServerBlade's network interface, RLX bundles the 24 network links into a pair of RJ21 connectors, as shown by the gray cables in the middle of our newly-constructed, 240-node Bladed Beowulf (dubbed "Green Destiny") in Figure 4.

Both our *MetaBlade* Bladed Beowulf and our recently completed *Green Destiny* Bladed Beowulf reside in a dusty, unventilated, and $80°$F environment. As we are still running system diagnostics and going through the initial "burn-in" period for *Green Destiny*, the focus of this paper will be on *MetaBlade*, our original 24-node Bladed Beowulf with 633-MHz Transmeta TM5600 CPUs and CMS 4.2.6, and *MetaBlade2*, a 24-node Bladed Beowulf with 800-MHz Transmeta TM5800 CPUs and CMS 4.3.1 that was "on-loan" from RLX Technologies during SC 2001 in November 2001.

3 Parallel N-Body Simulation & Treecode Library

We use a parallel N-body code to evaluate the performance of our Bladed Beowulf. The current version of the code isolates the elements of data management from the parallel computation. The former is implemented as a generic, hashed oct-tree library [15–17], which can then be used to support a large class of particle-based simulations. For instance, implementing the gravitational N-body simulation, used for benchmarking in this paper, requires only 2000 lines of codes external to the library. The vortex particle method only needs 2500 lines interfaced to exactly the same library. And smoothed particle hydrodynamics takes only 3000 lines of code to implement.

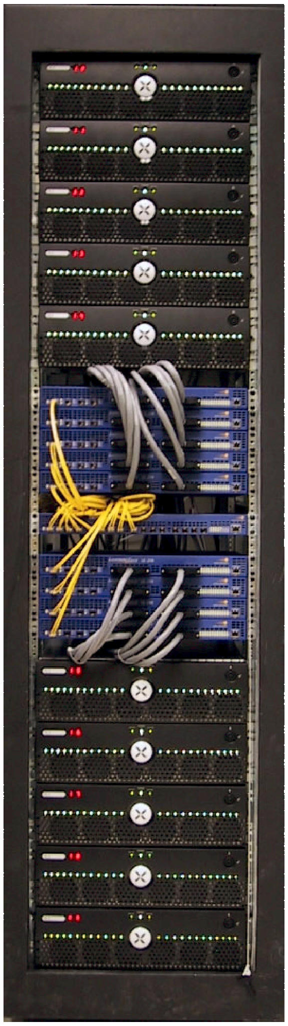

Figure 4. Green Destiny: A 240-Node Bladed Beowulf

Though the parallel N-body application lends itself to parallelism better than some applications, the communication between particles requires an extremely low-latency, high-bandwidth communication substrate if the application is written in the traditional synchronous "compute & communicate" cycle. Our code, however, implements an efficient mechanism for latency hiding during the traversal of the hashed oct-tree. To avoid stalls during non-local data access, we effectively do “explicit context switching” rather than block on pending communication. In order to manage the complexities of the required asynchronous message traffic, we have developed a paradigm called “asynchronous batched messages” (ABM) built from primitive `send/recv` functions whose interface is modeled after that of active messages.[6]

The most time-consuming part of an N-body simulation is computing components of the accelerations of particles [8]. For example, the x-component of the acceleration for particle j under the gravitational influence of particle k is given by

$$\frac{Gm_k(x_j - x_k)}{r^3}$$

where G is the gravitational constant, m_k is the mass of particle k, and r is the separation between the particles, i.e.,

$$r = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2 + (z_j - z_k)^2}$$

Evaluating $r^{-3/2}$ is the slowest part of computing the acceleration, particularly when the square root must be performed in software. Consequently, because of the importance of this calculation in particle-based simulations, we will use it as the basis of our gravitational microkernel benchmark to evaluate the uniprocessor performance of instruction-level parallelism over a range of COTS processors.

4 Experimental Study

Our *MetaBlade* Bladed Beowulf consists of 24 compute nodes with each node containing a 633-MHz Transmeta TM5600 CPU (100% x86 compatible) running CMS 4.2.6, 256-MB SDRAM, 10-GB hard disk, and 100-Mb/s network interface. We connect each compute node to a 100-Mb/s Fast Ethernet switch, resulting in a cluster with a star topology.

We first use a gravitational microkernel benchmark based on an N-body simulation to evaluate the uniprocessor performance of instruction-level parallelism over a wide range of COTS processors. Second, we run a full-scale N-body simulation with the treecode library to obtain a Gflop rating for our Bladed Beowulf and compare it to previously benchmarked clusters and supercomputers.

4.1 Gravitational Microkernel Benchmark

As discussed in Section 3, the most time-consuming part of an N-body simulation is computing components of the accelerations of particles [8], in particular, evaluating $r^{-3/2}$ where r is the separation between particles. Because of the importance of this calculation to our n-body codes at Los Alamos National Laboratory, we evaluate the instruction-level parallelism of the Transmetas using two different implementations of a reciprocal square root function. The first implementation uses the $sqrt$ function from a math library while the second implementation uses Karp’s algorithm [8]: table lookup, Chebychev polynomial interpolation, and Newton-Raphson iteration. To simulate Eqn. (1) in the context of an N-body simulation (and coincidentally, enhance the confidence interval of our floating-point evaluation), our microkernel benchmark loops 100 times over the reciprocal square root calculation.

Table 1 shows the Mflops ratings for six commodity processors over the two different implementations of the gravitational microkernel benchmark. (Note: All the Karp $sqrt$ numbers with the exception of the 633-MHz Transmeta TM5600 were hand-optimized to their respective architectures.) Considering that the Transmeta CPUs are software-hardware hybrids and the other CPUs are all-hardware designs, the Transmetas perform remarkably well. Specifically, the Transmetas perform comparably to similarly-clocked Intels, 1/2 to 2/3 as well as a 1.2-GHz AMD Athlon MP, and roughly 1/3 as well as the fastest Intel and AMD processors.

Processor	Math $sqrt$	Karp $sqrt$
500-MHz Intel Pentium III	87.6	137.5
533-MHz Compaq Alpha EV56	76.2	178.5
633-MHz Transmeta TM5600	115.0	144.6
800-MHz Transmeta TM5800	174.1	296.6
375-MHz IBM Power3	298.5	379.1
1200-MHz AMD Athlon MP	350.7	452.5

Table 1. Mflop Ratings on a Gravitational Microkernel Benchmark

The Transmetas hold their own with respect to traditional CPUs based on the behavior of its CMS. The first several times a specific x86 code sequence executes, the CMS interprets the code by decoding the instructions one at a time and then dispatching execution to corresponding VLIW instruction units. Once the x86 code executes a few times (i.e., iterates), the CMS translates the x86 instructions into highly optimized and extremely fast VLIW native instructions, executes the translated code, and caches the native-instruction translations for future use. If the same sequence of x86 instructions executes again, the high-performance cached translations execute immediately, and no re-translation is required. [7]

[6] Independently, one of the co-authors proposed a similar scheme called *buffered co-scheduling* [12], which effectively implements ABM in a distributed operating system, thus freeing the application programmer from the tedious bookkeeping of asynchronous communication and allowing the programmer to focus on the application itself.

[7] At the time of this writing, we are implementing a timer with cycle-counting granularity so we can quantitatively state the improvement of the Mflop rating on a per-iteration basis.

4.2 Treecode Benchmark

In November 2001, we ran a simulation with $9,753,824$ particles for about 1000 timesteps on our *MetaBlade* Bladed Beowulf consisting of 24 nodes where each node had a 633-MHz Transmeta TM5600, 256-MB SDRAM, 10-GB hard disk, and 100-Mb/s Fast Ethernet. Figure 5 shows an image of this simulation. The simulation was of a spherical region of space 100 Mpc (Megaparsec) in diameter, a region large enough to contain a few hundred thousand typical galaxies. The region inside a sphere of diameter 100 Mpc was calculated at high mass resolution, while a buffer region of 50 Mpc with a particle mass 8 times higher was used around the outside to provide boundary conditions. The initial conditions were extracted from a 134-million point initial dataset, calculated using a a 512^3 point 3-D FFT, from a Cold Dark Matter power spectrum of density fluctuations. Overall, the simulation completed about 1.3×10^{15} floating-point operations sustaining a rate of 2.1 Gflops. With a peak rating of 15.2 Gflops, this real application code running on our *MetaBlade* Bladed Beowulf achieves 2.1 / 15.2 = 14% of peak. We also ran the simulation on a loaned Bladed Beowulf (for SC 2001) from RLX Technologies, Inc., which we call *MetaBlade2*. *MetaBlade2* consisted of 24 nodes of 800-MHz Transmeta TM5800s, 256-MB SDRAM, 10-GB hard disk, and 100-Mb/s Fast Ethernet and produced 3.3 Gflops. With a peak rating of 19.2 Gflops, this real application code running on *MetaBlade2* achieves 3.3 / 19.2 = 17% of peak.

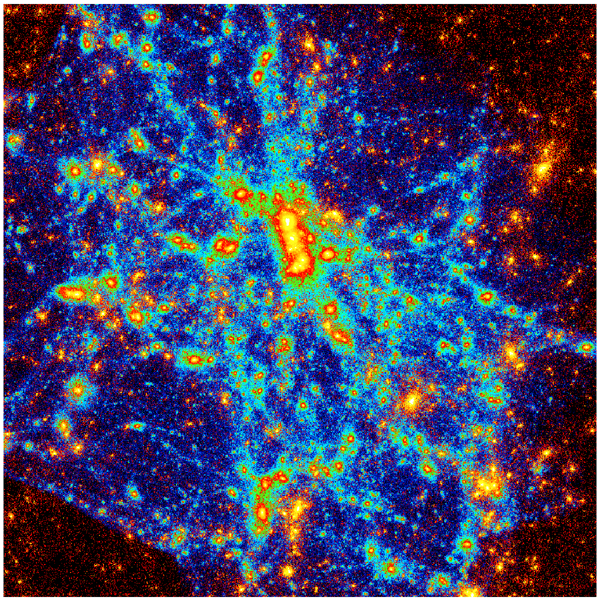

Figure 5. Intermediate Stage of a Gravitational N-body Simulation with 9.7 Million Particles. The region shown is about 150 million light years across.

Table 2 shows the relative placing of our *MetaBlade* (MB) Beowulf and the loaned *MetaBlade2* (MB2) Beowulf as well as 212 nodes of our newly-installed, 240-node *Green Destiny* (GD) Bladed Beowulf with respect to Mflops/processor. All three systems perform well on a performance-per-processor basis. So, although the RLX System 324 was designed for web hosting, we have demonstrated that it shows promise as a high-performance computing platform as well.

Per processor for this benchmark, the performance of the Transmeta Crusoe TM5800 with CMS 4.3.1 is about twice that of the Intel Pentium Pro 200 which was used in the Loki Beowulf cluster that won the Gordon Bell Price/Performance Award in 1997 and performs about the same as the 533-MHz Compaq Alpha processors used in the Avalon cluster. The Transmeta Crusoe TM5800, while architecturally identical to the TM5600, uses a newer version of CMS (i.e., 4.3.1) that performs approximately 25% better than CMS 4.2.6.

5 Performance Metrics

Despite Hennessy and Patterson [7] having shown the pitfalls of using processor clock speed, instructions per second (ips), and floating-point operations per second (flops) as performance metrics, scientists still tend to evaluate the performance of computing platforms based on floating-point operations per second (and even worse, some scientists compare processor clock speeds across different families of processors) despite the introduction of benchmark suites such as NAS [1] and SPEC [10]. However, the most prominent benchmarking list in the high-performance computing community has been the Top500 list at http://www.top500.org. This list is based on the "flop" rating of a single benchmark, i.e., Linpack, which solves a dense system of linear equations.

5.1 ToPPeR: Total Price-Performance Ratio

At SC, the world's premier supercomputing conference, the Gordon Bell Awards are based on performance (where performance is measured in "flops") and price-performance ratio (where price is the cost of acquisition and performance is in "flops"). However, we propose a new (but related) performance metric: total price-performance ratio (ToPPeR) where total price is the total cost of ownership.

Total cost of ownership (TCO) refers to all the expenses related to buying, owning, and maintaining a computer system within an organization. We break TCO into two components: acquisition cost (AC) and operating cost (OC), i.e., $TCO = AC + OC$. (To ensure a simple but still valid model, we assume that a given cost includes both the direct and indirect costs to an institution.)

The AC simply consists of hardware costs (HWC) and software costs (SWC), i.e., $AC = HWC + SWC$. This cost is generally a *fixed, one-time* cost at the time of purchase. The

Site	Machine	Processor	Procs	Gflops	Mflops/proc
LLNL	ASCI White	IBM Power3	8192	2500.00	305.2
LANL	SGI Origin 2000	MIPS R10000	64	13.10	205.0
SC ’01	MetaBlade2	Transmeta TM5800	24	3.30	138.0
LANL	Avalon	DEC Alpha 21164A	128	16.16	126.0
LANL	Green Destiny	Transmeta TM5600	212	21.40	100.9
LANL	MetaBlade	Transmeta TM5600	24	2.10	87.5
LANL	Loki	Intel Pentium Pro	16	1.28	80.0
NAS	IBM SP-2(66/W)	IBM SP-2	128	9.52	74.4
Sandia	ASCI Red	Intel Pentium Pro	6800	464.9	68.4
SC ’96	Loki+Hyglac	Intel Pentium Pro	32	2.19	68.4
Caltech	Naegling	Intel Pentium Pro	96	5.67	59.1
NRL	TMC CM-5E	Sun SuperSPARC	256	11.57	45.2
Sandia	ASCI Red	Intel Pentium Pro	4096	164.3	40.1
JPL	Cray T3D	Cray	256	7.94	31.0
LANL	TMC CM-5	Sun SPARC 2	512	14.06	27.5
Caltech	Intel Paragon	Intel iPSC/860	512	13.70	26.8
Caltech	Intel Delta	Intel i860	512	10.02	19.6

Table 2. Historical Performance of Treecode on Clusters and Supercomputers

OC, however, is much more difficult to quantify as it tends to be highly variable and recurring; this cost includes, but is not necessarily limited to, system-administration costs (SAC) such as installation, configuration, maintenance, upgrading, and support, power-consumption costs (PCC), space-consumption costs (SCC), and downtime costs (DTC).[8] The system-administration costs (SAC) of a Beowulf cluster can be particularly onerous as they involve the recurring costs of labor and materials.

In sum, using the notation defined above, we propose the following sets of straightforward equations as steps towards defining the total cost of ownership in high-performance computing.

$$TCO = AC + OC$$

where

$$\begin{aligned} AC &= HWC + SWC \\ OC &= SAC + PCC + SCC + DTC \end{aligned}$$

Table 3 presents a summary of the total cost of ownership (TCO) on five comparably-equipped, 24-node clusters based on AMD Athlons, Compaq/DEC Alphas, Intel Pentium IIIs (PIIIs) and Pentium 4s (P4s), and Transmeta Crusoe TM5600s, respectively. Each cluster runs a basic software set-up (e.g., no sophisticated job-scheduling software) and operates in a relatively *hot* environment, i.e., $80\,^{\circ}$F, rather than in a much cooler machine room.

For the purposes of our TCO calculation, we assume that the operational lifetime of each cluster to be four years. The system administration cost (SAC) assumes a burdened labor rate of $100/hour — burdened as in what an institution ultimately pays for an hour of work, not what the employee receives for an hour of work. Based on our own empirical data from our four traditional and one Bladed Beowulf clusters as well as lab-wide empirical data, the system administration cost on a traditional Beowulf runs $15K/year or $60K over four years.[9] In contrast, working from one disk-imaged blade, our *MetaBlade* Bladed Beowulf took 2.5 hours to install and configure and has been highly reliable with zero hardware failures and zero software failures in nine months; at $100/hour, that amounts to $250/year or $1000 over four years. Although there have been no failures thus far, we will assume that one major failure will occur per year and that the cost of the replacement hardware and the labor to install it amounts to $1200/year. Thus, over a four-year period, SAC runs $5050.

We estimate the power-drawing costs of the clusters based on the power dissipation of each node. For example, a complete Intel P4 node (with memory, disk, and network interface) generates about 85 watts under load, which translates to 2.04 kW for 24 nodes. Assuming a typical utility rate of

[8] Other OC components that may be seen more in an enterprise environment rather than a high-performance computing (HPC) environment include centralization, standardization, evaluation for re-investment, training, and auditing. In our calculation for TCO, we only use the OC components relevant to HPC but note that the calculation can be extended for other environments.

[9] These costs would be significantly less if the clusters were housed in a cooler, machine-room environment. With an 80°F environment, hardware failures are the norm. In fact, our internal 18-node Bladed Beowulf based on Intel processors produces incorrect answers after only ten minutes of operation and causes over half of the nodes to become inaccessible.

Cost Parameter	Alpha	Athlon	PIII	P4	TM5600
Acquisition	$17K	$15K	$16K	$17K	$26K
System Admin	$60K	$60K	$60K	$60K	$5K
Power	$8K	$4K	$4K	$8K	$2K
Space	$8K	$8K	$8K	$8K	$2K
Downtime	$12K	$12K	$12K	$12K	$0K
TCO	$105K	$99K	$100K	$105K	$35K

Table 3. Total Cost of Ownership for a 24-node Cluster Over a Four-Year Period

$0.10 kWh, 8760 hours per year, or 35,040 hours over four years, the total cost runs $7,148. (Note: The network interconnect, which would be the same for all the above clusters, is not accounted for in the above calculation.)

Space costs are rarely considered in the TCO of a computer system. Given that Pittsburgh Supercomputing Center leased space from Westinghouse Electric Company and spent $750,000 to renovate the facilities in order to house its new 6-TFLOP Terascale Computing System [13], these costs ought to be included as part of the total cost of ownership. In our space-cost calculation, we make the *significantly* more conservative assumption that space is being leased at a cost of $100 per square foot per year. For example, a 24-node Alpha cluster takes up 20 square feet, which translates to a four-year space cost of $8000. In contrast, a 24-node Bladed Beowulf takes up 6 square feet for a four-year cost of $2400. [10]

Based on how supercomputing centers charge for time on their clusters and supercomputers, we can estimate the cost of downtime based on the amount of lost revenue. We assume a conservative $5.00 charged per CPU hour (although [11] notes that the downtime cost per hour for a NYC stockbroker is $6,500,000). In the case of a 24-node cluster, these costs are relatively small even when we assume that a single failure causes the entire cluster to go down. Specifically, we experience a failure and subsequent 4-hour outage (on average) every two months on one of our more reliable, traditional Beowulf clusters. Thus, the cost of the downtime is 96 hours over four years for the cluster; with 24 nodes, the total CPU downtime is 96 hours × 24 = 2304 hours. The total downtime cost is then $11,520. In contrast, our Bladed Beowulf has yet to fail after nine months of operation; so, the downtime cost is $0.

Thus, for the five comparably-equipped and comparably-performing, 24-node CPUs, the TCO on our Bladed Beowulf is three times better than the TCO on a traditional Beowulf. In a large-scale supercomputing environment, the results are even more dramatic. However, the biggest problem with this metric is identifying the hidden costs in the operational costs; furthermore, the magnitude of most of these operational costs is institution-specific. To address this issue more concretely, we propose two related metrics — performance/space and performance/power — in the next section.

In summary, the TCO of our 24-node Bladed Beowulf is roughly three times better than a traditional Beowulf and performs comparably to similarly-clocked traditional Beowulfs. Thus, the ToPPeR value for our Bladed Beowulf is about 1/3 that of a traditional cluster, i.e., three times better than a traditional Beowulf.

5.2 Performance/Power

The performance of a microprocessor on a program is proportional to its speed (i.e., clock frequency, f) multiplied by the number of instructions it executes per clock cycle (i.e., instructions per clock, ipc) divided by the number of instructions, n, in the program [7]. Thus, chip designers improve performance by increasing the clock frequency, by increasing the number of instructions that execute each clock cycle, and by adding more powerful instructions (i.e., migrate back toward CISC).

By taking advantage of improvements in semiconductor fabrication, chip designers load the micrprocessor design with more transistors and increase clock frequency. The combination of these design choices enables the doubling of microprocessor performance every 18 months. However, it also quickly drives up the power dissipation of microprocessors as $power\ \alpha\ C \times V^2 \times f$, where C is capacitance, V is voltage, and f is frequency. For the time being, chip designers address this issue, particularly for mobile processors, by decreasing the voltage (and aggressive cooling). Lowering the voltage by half drops the power dissipation to one-quarter of its former value; however, lowering the operating voltage also lowers the maximum operating frequency, thus decreasing performance. This vicious cycle will continue throughout this decade and become more and more difficult to deal with.

The need to minimize the exponentially-growing power dissipation of microprocessors exists because of the following general relationship between power and temperature:

$$T_{surface} = T_{ambient} + P_{total} \times R_{thermal}$$

where $T_{surface}$ is the temperature of the surface of the die,

[10] Note: If we scaled up our Bladed Boewulf to 240 nodes, i.e., cluster in a rack, the cost per square foot over four years would *remain* at $2400 while the traditional Beowulfs' cost would increase ten-fold to $80,000!

$T_{ambient}$ is the ambient temperature, P_{total} is the total thermal power dissipated, and $R_{thermal}$ is the thermal resistance.[11] $T_{ambient}$ stays roughly constant while $R_{thermal}$ is a constant. Thus, $T_{surface}$ depends *directly* on P_{total}. And as stated previously, because unpublished (but reliable) empirical data from two leading vendors indicates that the failure rate of a component *doubles* for every $10\,^{\circ}$C increase in temperature, $T_{surface}$ must be minimized, and therefore, so does the total thermal power dissipation, i.e., P_{total}, in order to enhance the reliability of the system.[12]

So, what is the solution to this dilemma? Quit using the "increasing clock frequency = increasing performance" marketing ploy. We believe that the ideal clocking frequency will be in the 500-MHz to 1-GHz range and that to improve performance, the microprocessor must do more work on a per-cycle basis, e.g., see the results for the IBM Power3 in Table 1. By keeping both the voltage and frequency low, power dissipation is also kept low. A detailed discussion about this controversial statement is beyond the scope of this paper.

In any case, the "frequency" war will continue. In the meantime, we use this opportunity to propose a new metric aimed at keeping Intel and AMD "honest" — *performance/power ratio*. Table 4 shows the performance/power ratio with our Bladed Beowulfs (240-node RLX ServerBlade 667s, i.e., *Green Destiny* or GD, and 24-node RLX ServerBlade 800s, i.e., *MetaBlade2* or MB2), a traditional Beowulf (Avalon), and two of the most powerful supercomputers in the world (ASCI Red and ASCI White). In this table, we see that the Transmeta-based Bladed Beowulfs clearly outperform the traditional Beowulf and ASCI Red and White. In fact, when comparing the *MetaBlade2* to ASCI Red, the former performs over *ten* times better per watt of thermal energy dissipated.

5.3 Performance/Space

Although performance has increased by a factor of 2000 since the Cray C90, performance per square foot has only grown by a factor of 65. This implies that supercomputers are making less efficient use of the space that they occupy, and space is money. Supercomputers have gotten so big that institutions must lease space (e.g., Pittsburgh Supercomputing Center's Terascale Computing System) and renovate facilities (e.g., $750,000 for a better cooling system) or even construct a new building to house the supercomputer. These costs are rarely, if ever, included as part of the cost for purchasing a supercomputer. Consequently, we propose a new efficiency metric called *performance/space ratio*.

As in the previous section, Table 5 presents the performance/space ratio of our Bladed Beowulfs (*Green Destiny* and *MetaBlade2*), a traditional Beowulf (Avalon), and two of the most powerful supercomputers in the world (ASCI Red and ASCI White). With respect to performance/space, the Transmeta-based Bladed Beowulfs again outperform the traditional Beowulf and ASCI Red and White (despite the fact that *MetaBlade2* is not even a full rack of processors but simply a 3U box, i.e., 19" wide, 25.2" deep, and 5.25" high). If we were to extrapolate the single 3U box into a rack of 3U boxes with associated network switches, the performance/space ratio would be 5500 or an order of magnitude better than the traditional Beowulf and DOE ASCI supercomputers. In fact, using 667-MHz Transmeta TM5600s, we could build a 16-Tflop (peak) supercomputer in only 600 sq. ft., i.e., $20' \times 30'$; and with the latest 933-MHz Transmeta TM5800s, a 16-Tflop (peak) supercomputer could be built in only 420 sq. ft., i.e., $20' \times 21'$.

Acknowledgements

We want to thank Gordon Bell and Chris Hipp for their encouragement on this endeavor. Chris Hipp, founder of RLX Technologies and designer of the RLX System 324, was particularly instrumental in expediting the delivery of the necessary resources for the construction of *Green Destiny*.

Finally, we would like to thank the anonymous reviewers who provided insightful comments and constructive criticism, thus enabling us to improve the quality of the paper.

6 Conclusion

In this paper, we presented a novel twist on the traditional Beowulf cluster — the Bladed Beowulf. Although the acquisition cost of this cluster is roughly twice as much as a comparably-equipped but traditional Beowulf cluster, our experiences and calculations predict that the total cost of ownership (TCO) of a Transmeta-based Bladed Beowulf will be three times cheaper than a traditional Beowulf cluster.

We also introduced three new (but related) performance metrics: (1) performance/power, (2) performance/space, and (3) ToPPeR: Total Price-Performance Ratio, where total price encompasses TCO. While the latter metric can be quite difficult to quantify and to normalize across institutions, the former two metrics are easily quantified and contribute to the TCO.

Currently, our biggest concern is the continued pursuit of Moore's law and its effect on system reliability. The continued tracking of Moore's law will result in the microprocessor of 2010 having over *one billion* transistors and dissipating over *one kilowatt* of thermal energy; this is considerably more energy per square centimeter than even a nuclear reactor. As Intel pushes forward with its even more voracious Itanium

[11] This general equation becomes marginally more complicated when a cooling fin or a heat sink is added to the die.

[12] For instance, without any active cooling, a conventional mobile processor runs at a minimum of 105°C (221°F) while a Transmeta Crusoe burns at a minimum of 48°C (118°F). Thus, without any active cooling, the failure rate of a conventional mobile processor is roughly 64 times worse than with the Transmeta Crusoe.

Machine	RLX TM5600	RLX TM5800	Avalon	ASCI Red	ASCI White
Performance (Gflops)	21.4	3.3	17.6	600	2500
Power (kilowatts)	5.2	0.52	18.0	1200	2000
Perf/Power (Mflops/watt)	4.12	6.35	0.978	0.5	1.25

Table 4. Performance-Power Ratio for Five Parallel-Computing Systems

Machine	RLX TM5600	RLX TM5800	Avalon	ASCI Red	ASCI White
Performance (Gflops)	21.4	3.3	17.6	600	2500
Area (feet^2)	6	6	120	1600	9920
Perf/Power (Mflops/feet^2)	3500	550	150	375	252

Table 5. Performance-Space Ratio for Five Parallel-Computing Systems

processor, which will dissipate 130 watts; Transmeta continues its push in the other direction, i.e., even lower power consumption but with expected performance being competitive (or even better) than the Itanium. In short, we hope this paper will disrupt the "performance at any cost" mantra of the HPC community and motivate HPC researchers to consider alternative cluster and supercomputer architectures such as the RLX System 324 cluster and IBM Blue Gene supercomputer, respectively.

Finally, although we compared our Bladed Beowulf directly to a traditional Beowulf and two of the most powerful supercomputers in the world, we note that the Bladed Beowulf is *not* meant to be a replacement for these other types of architectures (at least, not yet). It is meant as a cost-effective alternative, an alternative that would excel as a departmental cluster or a developmental cluster for clustering software. When it comes down to raw performance, the Bladed Beowulf simply cannot compete with ASCI-style supercomputers due to their massive compute and communication capabilities. However, this research can perhaps be viewed as the foundation for the supercomputer of 2010.

References

[1] D. Bailey, T. Harris, W. Saphir, R. van der Wijngaart, A. Woo, and M. Yarrow. The NAS Parallel Benchmarks 2.0. *The International Journal of Supercomputer Applications*, December 1995.

[2] M. Bass and C. Christensen. The Future of the Microprocessor Business. *IEEE Spectrum*, April 2002.

[3] G. Bell and J. Gray. What's Next in High-Performance Computing? *Communications of the ACM*, February 2002.

[4] D. Ditzel. The TM6000 Crusoe: 1-GHz x86 System on a Chip. 2001 Microprocessor Forum, October 2001.

[5] W. Feng, M. Warren, and E. Weigle. Honey, I Shrunk the Beowulf! In *Proc. of the International Conference on Parallel Processing (ICPP)*, August 2002.

[6] W. Feng, M. Warren, and E. Weigle. Honey, I Shrunk the Beowulf! Technical Report LA-UR 02-1210, Los Alamos National Laboratory, March 2002.

[7] J. Hennessy and D. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers, 2 edition, 2000.

[8] A. Karp. Speeding Up N-body Calculations on Machines Lacking a Hardware Square Root. *Scientific Programming*, 1(2), 1992.

[9] A. Klaiber. The Technology Behind Crusoe Processors. *White Paper*, January 2000.

[10] SPEC Newsletter. Spec benchmark suite release, 1990.

[11] D. A. Patterson. New Challenges for the PostPC Era. Invited Talk at the IEEE International Parallel and Distributed Processing Symposium, April 2001.

[12] F. Petrini and W. Feng. Buffered Co-Scheduling: A New Methodology for Multitasking Parallel Jobs on Distributed Systems. In *Proc. of the Int'l Parallel & Distributed Processing Symposium (IPDPS 2001)*, May 2000.

[13] B. Spice. Wiring, Air-Cooling Systems Go In As Assembly of Terascale Approaches: Setting the Stage for the Supercomputer. *The Pittsburgh Post-Gazette*, April 2001.

[14] T. Sterling, D. Becker, D. Savarese, J. Dorband, U. Ranawake, and C. Packer. Beowulf: A Parallel Workstation for Scientific Computation. In *Proc. of the Int'l Conf. on Parallel Processing (ICPP)*, August 1995.

[15] M. Warren and J. Salmon. A Parallel Hashed Oct-Tree N-body Algorithm. In *Proc. of Supercomputing '93*, November 1993.

[16] M. Warren and J. Salmon. A Parallel, Portable and Versatile Treecode. In *Proc. of SIAM Conference on Parallel Processing for Scientific Computing*, February 1995.

[17] G. Winckelmans, J. Salmon, M. Warren, and A. Leonard. The Fast Solution of Three-Dimensional Fluid Dynamical N-Body Problems Using Parallel Tree Codes: Vortex Element Method and Boundary Element Method. In *Proc. of SIAM Conference on Parallel Processing for Scientific Computing*, February 1995.