Green Destiny and its Evolving Parts

Wu-chun Feng and Chung-hsing Hsu
P.O. Box 1663, M.S. D451; Los Alamos National Laboratory; Los Alamos, NM 87545
E-mail: feng@lanl.gov, Phone: +1-505-665-2730, Fax: +1-505-665-4934
Keywords: dynamic voltage-scaling, power-aware supercomputing, NAS, SPEC.

Abstract

Although the performance of supercomputers on our n-body cosmology code has improved by a factor of nearly 2000 since 1991, the performance per watt has only improved 300-fold and the performance per square foot only 65-fold. Clearly, we are building less and less efficient supercomputers, thus resulting in the construction of new machines rooms[1] and even entirely new buildings. Furthermore, as these supercomputers continue to follow "Moore's Law for Power Consumption," the reliability of these supercomputers continues to plummet, relative to Arrenhius' equation for microelectronics.

To address these problems, we built a super-efficient supercomputer dubbed *Green Destiny*, a 240-processor supercomputer that fits in a telephone booth (i.e., a footprint of five square feet) and sips less than 5.2 kW of power at full load [FWW02, WWF02, Feng03]. This "Supercomputer for the Rest of Us" – a 2003 R&D 100 award-winning machine – provided affordable, general-purpose supercomputing to our application scientists while sitting in an 85-90˚ F (29-32˚ C) dusty warehouse at 7,400 feet (2256 meters) above sea level. Furthermore, it delivered reliable computing cycles without any special facilities, i.e., no air conditioning, no humidification control, no air filtration, and no ventilation, and *without any unscheduled downtime*.

Impact of Green Destiny

This supercomputer resulted in an avalanche of news coverage in over a hundred media outlets, including *BBC News*, *CNN*, *Communications of the ACM*, *GRIDtoday*, *HPCwire*, *PCWorld*, *The New York Times*, and *The Register*.

However, although Green Destiny demonstrated a total price-performance ratio (ToPPeR) that was 50% better than a traditional Beowulf cluster or supercomputer, power efficiency (i.e., performance-power ratio) that was up to eight times better, and space efficiency (i.e., performance-space ratio) that was up to thirty times better, both the raw performance and price/performance lagged a traditional Beowulf cluster or supercomputer by a factor of two. Thus, many would argue that Green Destiny sacrificed too much performance in achieving power and space efficiency (and thus, better reliability and total cost of ownership).

Therefore, we propose to evolve Green Destiny with a hybrid software-hardware solution, one that uses commodity processors from AMD (i.e., Athlon XP-M, Athlon 64, and Opteron) to achieve better performance, coupled with AMD's "Cool-N-Quiet" technology (formerly PowerNow!) and our novel dynamic voltage-scaling (DVS) technique to reduce power consumption by as much as 40% while impacting performance by less than 7%.

Motivation

In 1991, a Cray C90 vector supercomputer occupied about 600 square feet (sf) and required 500 kilowatts (kW) of power. The ASCI Q supercomputer at Los Alamos National Laboratory will ultimately occupy over 21,000 sf and require 3000 kW of power. Though the performance between these two systems has increased by nearly a factor of 2000, the performance per watt has only increased by 300-fold, and the performance per square foot has only increased by a paltry factor of 65. This latter number implies that supercomputers are making less efficient use of the space that they occupy, which oftentimes results in the design and construction of new machine rooms, and in some cases, requires the construction of entirely new buildings. The main reason for this less efficient use of space is the exponentially increasing power requirements of compute nodes, i.e., "Moore's Law for Power Consumption" (Figure 1). When nodes consume and dissipate more power, they must be spaced out and aggressively cooled.

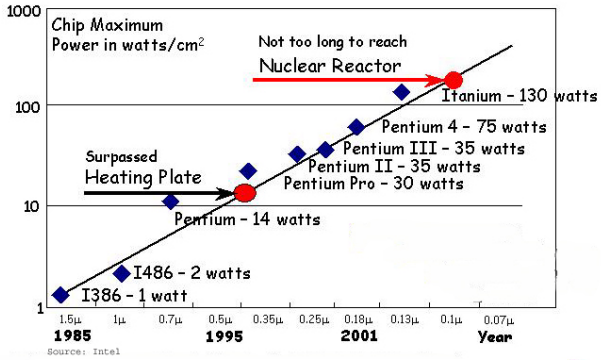

Figure 1. Moore's Law for Power Consumption

[1] For example, Celera Genomics paid as much for their DEC Alpha-based supercomputer as they did for their machine room – $6M for 6000 square feet. (See the GenomeWeb article – "Craig Venter Goes Shopping for Bioinformatics to Fill His New Sequencing Center" – which discusses his intent to look into our "green machine": http://cmbi.bjmu.edu.cn/news/0210/97.htm.)

Without exotic cooling facilities, traditional (inefficient) supercomputers would be so unreliable (due to overheating) that they would always be unavailable for use by the application scientist. In fact, our own empirical data as well as unpublished empirical data from a leading vendor corroborates that the failure rate of a compute node *doubles* with every 10˚C (18˚ F) increase in temperature, as per Arrenhius' equation when applied to microelectronics, and temperature is proportional to power consumption. Unfortunately, building exotic cooling facilities can cost as much as the supercomputer itself, e.g., the building for the ASCI Q computer cost nearly $100M. Operating and maintaining such facilities costs even more and is still no guarantee that the supercomputer will not suffer failures. In sum, all this adds up to an astronomical total cost of ownership (TCO).

The Origin of Green Destiny

To address the above problems, we constructed our super-efficient *Green Destiny* (see Figure 2), a 240-processor supercomputer that fits in a telephone booth (five square feet) and sips less than 5.2 kW of power at full load [FWW02, WWF02, Feng03]. This 2003 R&D 100 award-winning machine provides affordable, general-purpose computing to our application scientists while sitting in an 85-90˚ F (29-32˚ C) dusty warehouse at 7,400 feet (2256 meters) above sea level. More importantly, it provides reliable computing cycles without any special facilities, i.e., no air conditioning, no humidification control, no air filtration, and no ventilation, and without any downtime.[2]

Green Destiny takes a novel approach to supercomputing, one that ultimately re-defines performance to encompass metrics that are of more relevance to end users – efficiency, reliability, and availability – by designing an architecture around which to appropriately stitch together the modified building blocks of Green Destiny. The modified building blocks include a 1-GHz Transmeta-based RLX ServerBlade as the compute node and World Wide Packets' Lightning Edge network switches configured in a one-level tree topology for efficient communication. By selecting a Transmeta processor, Green Destiny takes a predominantly hardware-based approach to power-aware supercomputing. A Transmeta processor eliminates about 75% of the transistors used in a traditional RISC architecture and implements the lost (but inefficient) hardware functionality in its code morphing software (CMS), a software layer that sits directly on the Transmeta hardware. However, while the Transmeta processor may be significantly more reliable because it runs so much cooler than a conventional mobile processor, its Achilles' heel is its floating-point performance. Consequently, we modified the CMS to create a "high-performance CMS" that improved floating-point performance by 42% and ultimately matched the performance of a conventional mobile processor on a clock cycle-by-clock cycle basis (e.g., 1.2-GHz Intel Pentium III-M), but it still lagged the performance of the fastest processor at the time by a factor of two.

(In March 2003, an upgraded 480-processor version of *Green Destiny* achieved 200 Gflops on Linpack – a feat that would have placed the machine at #372 of the Top500 Supercomputer List in November 2002 and just outside the Top500 List in June 2003.)

Figure 2. Green Destiny

The Evolution of Green Destiny

Though we demonstrated in [FWW02] that the total price-performance ratio (ToPPeR) of Green Destiny was approximately 1.5 times better than a traditional Beowulf cluster or supercomputer, the power efficiency (i.e., performance-power ratio) was 7 to 8 times better, and the space efficiency (i.e., performance-space ratio) was 20 to 30 times better, both the raw performance and price/performance lagged a traditional Beowulf cluster by a factor of two. Thus, with Green Destiny, many could argue that we went to an opposite extreme of supercomputing, i.e., sacrificed too much performance to achieve power efficiency and space efficiency (and thus better reliability and reduced total cost of ownership). Another criticism of Green Destiny was that it was only based on "pseudo-commodity" parts, specifically

[2] In contrast, a more traditional, high-end 240-processor supercomputer such as a Beowulf cluster generally requires a specially-cooled machine room to operate reliably as such a supercomputer easily consumes as much as 36.0 kW at load, roughly seven times more than Green Destiny. (Remember that in this scenario, one not only needs electricity to power the supercomputer, but it also needs electricity to power the cooling and ventilation systems.)

the Transmeta processor on an RLX ServerBlade. Nowhere else could one find a Transmeta processor in a server other than RLX, and Transmeta's target market is the laptop and embedded systems market, not the server market.

To address the above criticisms of our architecturally-based, power-aware Green Destiny, we propose a hybrid hardware-software solution, one that uses commodity CPUs from AMD (i.e., XP-M, Athlon 64, and Opteron) to achieve better performance and its associated "Cool-n-Quiet" technology to reduce power consumption by nearly 40% while impacting performance by less than 7%. We achieve this feat through the judicious use of a well-known mechanism called dynamic voltage (and frequency) scaling or DVS. In general, a DVS algorithm needs to determine *when* to adjust the current frequency-voltage setting (i.e., *scaling point*) and *what* to set the new frequency-voltage setting to (i.e., *scaling value*).[3]

Today's DVS algorithms solve the following minimization problem: Given a task of workload W and deadline D, find a schedule $\{t_f\}$ such that when the task is executed for t_f seconds at frequency f, the total energy usage E is minimized, the deadline D is met, and the required work W is performed. This minimization problem assumes that W is known a priori [LS01] and that all applications are CPU-bound applications. Unfortunately, as shown in [XMM03, SAM03], W is not always constant across frequencies; thus, knowing W a priori is virtually impossible. Furthermore, not all applications are CPU-bound. Therefore, we propose a new formulation that eliminates the hard-to-predict W and includes performance modeling for all types of applications by replacing the aforementioned W constraint (i.e., $\Sigma_f\ f \cdot t_f = W$) with the following constraint $\Sigma_f\ t_f / T(f) = 1$. The optimal solution for this new formulation, i.e., $\{t_f^*\}$, can then be characterized by the following theorem:

Theorem: Deadline-Constrained Scheduling for Energy Minimization.
For $D < \max_f T(f)$, if the ratios $\gamma_i = [E_{f_i} - E_{f_{i+1}}] / [T(f_i) - T(f_{i+1})]$ are negative and non-increasing, the optimal solution $\{ t_f^* \}$ is

$$t_f^* = \begin{cases} [D - T(f_{j+1})] / [T(f_j) - T(f_{j+1})] \cdot T(f_j) & f = f_j \\ D - t_{f_j}^* & f = f_{j+1} \\ 0 & \text{otherwise} \end{cases}$$

where $T(f_{j+1}) < D \le T(f_j)$.

This theorem says that if E_f is convex and non-increasing on $T(f)$, then running at the ideal single frequency f^*, where $T(f^*) = D$, will minimize the total energy usage. If the frequency f^* is not directly supported by the system, the two neighboring frequencies f_j and f_{j+1}, $T(f_{j+1}) < D \le T(f_j)$, can emulate it and results in minimum energy consumption.

To qualitatively understand the above theorem, let us consider a 600-1600 MHz Intel Pentium-M processor with the performance levels as specified by its Intel datasheet. Figure 3 depicts the convexity of $E(T(f))$ in terms of a CPU cycle. All the performance levels except 1.4 GHz satisfy the condition of the theorem, thus 1.4 GHz should never be used in any DVS algorithm because its speed can be emulated by other levels with lower energy consumption, e.g., 1.4-GHz performance can be emulated by running half of the time at 1.2 GHz and half of the time at 1.6 GHz, resulting in 13.9 nJ/cycle (versus the 14.3 nJ/cycle that running at 1.4 GHz would produce).

Based on the above theorem, we propose a new interval-based DVS algorithm that abstracts the performance model T(f) as a single parameter β and then uses β to direct the DVS algorithm on-the-fly.

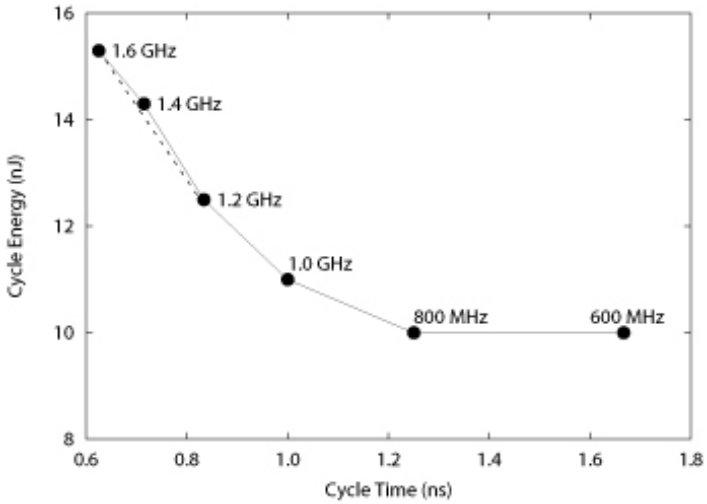

Figure 3. Cycle Energy vs. Cycle Time

[3] In the meantime, researchers at the University of Tsukuba in Japan continue to pursue the low-power architectural approach with a *"Green Destiny II"* cluster, which is twice as dense and uses Gigabit Ethernet rather than fast Ethernet as its network interconnect.

Our performance model consists of a single parameter β called the *performance-scalability factor* such that $\beta \in [0,1]$ and is defined as follows: $T(f) / T(f_{max}) = \beta \cdot f_{max} / f + (1 - \beta)$. The parameter β represents the sensitivity of the application performance to the change in CPU speed. Conceptually, it is similar to the scalability of performance in the field of parallel processing, but the number of processors is replaced here by the various CPU frequencies. If $\beta = 1$, it means that the execution time will be cut in half when the CPU speed is twice as fast. If $\beta = 0$, the execution time will remain constant even when running at the slowest frequency. That is, CPU-bound applications will have $\beta \approx 1$ whereas memory- and I/O-bound applications will have $\beta \approx 0$.

The above theorem describes an optimal schedule for DVS in terms of a deadline D and performance model $T(f)$. However, there is no consensus on how to assign a deadline to a general-purpose application, and the parameter β only captures $T(f)$ relative to the different performance levels. Because we need absolute values for $T(f_j)$ in order to apply the theorem, we propose the following as part of our β-driven DVS algorithm: (1) The algorithm provides an additional parameter δ defined in terms of the relative execution time, i.e., $D = (1 + \delta) \cdot T(f_{max})$. (2) The algorithm uses the optimal schedule in every fixed-length time interval instead of the entire execution period. The resulting β-driven DVS algorithm is shown in Figure 4.

While β can be provided with the application program, its value may depend on the input data and even the underlying computer architecture. As a result, we need a way to estimate β reasonably accurately "on the fly." At a high level, we calculate β on-the-fly by keeping track of the average MIPS rate for each frequency and compute a new β value using a least-squares fit.

1. Compute the ideal single frequency f^*.

$$f^* = \begin{cases} f_{min} & \text{if } \beta \le \delta \\ f_{max} / (1 + \delta / \beta) & \text{otherwise} \end{cases}$$

2. Figure out f_j and f_{j+1} such that $f_j \le f^* < f_{j+1}$.
3. Compute the ratios, r_{fj} and r_{fj+1}.
   $r_{fj} = [\, (1 + \delta / \beta) / f_{max} - 1 / f_{j+1} \,] / (1 / f_j - 1 / f_{j+1})$
   $r_{fj+1} = 1 - r_{fj}$
4. For every second, perform the following:
   a. Run r_{fj} seconds at frequency f_j.
   b. Run r_{fj+1} seconds at frequency f_{j+1}.

Figure 4. β-driven DVS Algorithm.

Benchmarking a Commodity Compute Node with a β-driven DVS Algorithm

To test our β-driven DVS algorithm, we ran SPEC benchmarks on an AMD XP-M processor, one of three potential successors to our Transmeta processor in Green Destiny[4] and one of the processors of choice in Sun's Sun Fire B1600 Blade System. For each SPEC benchmark, we derived a β value by profiling $T(f)$ for all the performance levels on the AMD XP-M processor and using a least-squares fit; these values are shown in Table 1. Given that application developers would rather *not* profile their codes a priori, we have also developed an algorithm to calculate β on the fly, one which estimates the β values in Table 1 with a surprising degree of accuracy; the derivation of the algorithm for the "on-the-fly" β estimator is currently outside the scope of this paper. Future work will apply this β-driven DVS algorithm when running parallel codes such as the entire NAS Parallel Benchmark suite and a sanitized ASCI parallel benchmark that was used to benchmark a prototype of ASCI Purple. Our initial results with NAS are quite promising – on average, we see a 25% reduction on power consumption with only a 2% impact on peak performance.

Table 2 presents the performance of our β-driven and β-on-the-fly DVS algorithm versus three well-known DVS algorithms: (1) a frequency-based algorithm *(freq)* that is more suited for CPU-intensive applications only, (2) a "retired instructions" algorithm *(mips)*, and (3) Intel's *SpeedStep*, noted to be the best interval-based DVS algorithm by Grunwald et al. [GLF00] back in 2000.

When *SpeedStep* is applied to the SPEC benchmarks, we see that it has little to no effect on performance and energy usage. This is because *SpeedStep* is intended for interactive applications such as Microsoft Office, i.e., when the processor can detect long periods of idle time or when the processor is running on a battery, rather than non-interactive applications such as the SPEC benchmarks. The benchmarks all run in virtually the same amount of time with the same amount of energy usage with *SpeedStep*. Because *freq* depends upon the number of CPU cycles as the metric for specifying the CPU work requirement and the number of CPU cycles *varies* significantly across frequency-voltage performance settings, the *freq* DVS algorithm does not perform particularly well either – energy reductions of 3-5% at the expense of 2-3% performance degradation. The *mips* algorithm performs better than the previous two DVS algorithms because the number of retired instructions is a better metric for specifying the CPU work requirement since the number of instructions tends to remain constant across all performance levels. Our β-driven DVS algorithm clearly performs the best of all the DVS algorithms with energy savings as high as 39% with performance degradation only around 5% on average.

[4] The other processors being considered are the AMD Athlon 64 and AMD Opteron with Cool-n-Quiet Technology enabled. Our early results, running SPEC benchmarks, NAS Parallel Benchmarks, and a sanitized ASCI parallel benchmark, on this platform are quite promising.

SPEC Program	β
swim	0.02
tomcatv	0.24
su2cor	0.27
compress	0.37
mgrid	0.51
vortex	0.65
turb3d	0.79
go	1.00

Table 1. β Values for SPEC Benchmarks

SPEC Program	*freq*	*mips*	*SpeedStep*	β-derived	β on-the-fly
swim	1.00 / 0.96	1.00 / 1.00	1.00 / 1.00	1.07 / 0.61	1.07 / 0.63
tomcatv	1.00 / 0.97	1.03 / 0.83	1.00 / 1.00	1.04 / 0.79	1.02 / 0.86
su2cor	1.00 / 0.95	1.01 / 0.96	0.99 / 0.99	1.04 / 0.81	1.03 / 0.85
compress	1.02 / 0.97	1.05 / 0.92	1.02 / 1.02	1.06 / 0.86	1.04 / 0.89
mgrid	1.01 / 0.97	1.00 / 1.00	1.00 / 1.00	1.03 / 0.88	1.03 / 0.89
vortex	1.01 / 0.97	1.07 / 0.94	1.01 / 1.00	1.04 / 0.92	1.06 / 0.90
turb3d	1.03 / 0.97	1.01 / 1.00	1.00 / 1.00	1.05 / 0.95	1.05 / 0.95
go	1.02 / 0.99	0.99 / 0.99	1.00 / 1.00	1.04 / 0.96	1.06 / 0.96

Table 2. Performance of DVS Algorithms. (Each table entry is in the format of "relative time / relative energy" with respect to total execution time and system energy usage running the benchmark at full speed. Thus, 1.00 / 1.00 means that a DVS algorithm took the same amount of time and consumed the same amount of energy.)

References

[Feng03] W. Feng, "Making a Case for Efficient Supercomputing," *ACM Queue*, October 2003.

[FWW02] W. Feng, M. Warren, and E. Weigle, "The Bladed Beowulf: A Cost-Effective Alternative to Traditional Beowulfs," *Proc. of IEEE Cluster 2002*, Sept. 2002.

[GLF00] D. Grunwald, P. Levis, K. Farkas, C. Morrey, and M. Neufeld, "Policies for Dynamic Clock Scheduling," *Proc. of the 4th Symp. On Operating System Design and Implementation (OSDI)*, October 2000.

[LS01] J. Lorch and A. Smith, "Improving Dynamic Voltage Algorithms with PACE," *Proc. of ACM SIGMETRICS*, Jun. 2001.

[SAM03] K. Seth, A. Anantaraman, F. Mueller, and E. Rotenberg, "FAST: Frequency-Aware Static Timing Analysis," *Proc. of the IEEE Int'l Real-Time Systems Symp. (RTSS)*, Dec. 2003.

[WWF02] M. Warren, E. Weigle, and W. Feng, "High-Density Computing: A 240-Processor Beowulf in One Cubic Meter, *Proc. of Supercomputing 2002 (SC'02)*, Nov. 2002.

[XMM03] F. Xie, M. Martonosi, and S. Malik, "Compile-time Dynamic Voltage Scaling Settings: Opportunities and Limits," *Proc. of ACM SIGPLAN Conf. on Programming Languages Design and Implementation (PLDI)*, Jun. 2003.