Appears in the 3^{rd} Workshop on High-Performance, Power-Aware Computing (HPPAC 2007)

Power, Performance, and Thermal Management for High-Performance Systems

Heather Hanson[1], Stephen W. Keckler[2], Karthick Rajamani[3], Soraya Ghiasi[3],
Freeman Rawson[3], and Juan Rubio[3]

[1,2]The University of Texas at Austin
[1]Dept. of Electrical & Computer Engineering
[2]Dept. of Computer Sciences
Austin, Texas, USA
hhanson@mail.utexas.edu, skeckler@cs.utexas.edu

[3]IBM Corporation
Austin Research Laboratory
Austin, Texas, USA
{karthick,sghiasi,frawson,rubioj}@us.ibm.com

Abstract

In future high-performance systems it will be essential to balance often-conflicting objectives of performance, power, energy, and temperature under variable workload and environmental conditions. In this work, we describe a goal-driven approach that conveys multiple expectations to managers that dynamically tune operating states to best meet those demands. We show the benefit of a concise goal specification for complex objectives and the feasibility of managing multiple constraints while maintaining high performance and safe operation. We evaluate key features of our approach with a prototype implementation on a Pentium M platform with Red Hat Enterprise 4 that controls voltage and frequency scaling to achieve the desired performance, power and temperature goals.

1 Introduction

In this work, we describe a run-time manager for multiple constraints, named *PET* for performance, power, energy, and temperature management. A run-time manager should respond quickly to workload and environmental conditions, and maintain safe operation while pushing the system to its limits for best performance. PET accomplishes this with *g*oal-driven control that adapts to a wide range of operating conditions and resource usage. PET continuously monitors the system, chooses appropriate power states (p-states, such as voltage and frequency levels), and takes action to reach desired goal targets.

Our concise goal-specification method enables the user to easily specify multi-dimensional constraints and directives while our underlying implementation transparently translates resulting objectives to metrics appropriate for the system under current conditions. PET separates constraints into hard limits and soft limits to enforce safety-related boundaries such as maximum temperatures or power draw, while allowing flexibility for additional objectives. To explore the key features of the PET approach, we built a prototype software implementation that controls a single-core Intel Pentium M processor via DVFS, dynamic voltage and frequency scaling. The prototype platform offers a controlled environment for observing software and hardware behavior with a low-power processor architecture that has been incorporated into high-density server systems such as IBM's Integrated xSeries servers [7], HP's ProLiant BL10e G2 [5], and Fujitsu Siemens' PRIMERGY Blade BX300 [4].

In this paper, we demonstrate the prototype PET manager's abilities in two situations. First, we show how PET maximizes performance within power and thermal budgets by adapting DVFS settings to dynamic workload behavior with the SPEC CPU2000 suite, with fixed goal conditions. In the second case, we create a scenario where the goal also changes during execution, moving along a spectrum between lower operating cost and higher performance. In both cases, the manager chooses appropriate frequencies to achieve the best possible performance with minimal constraint violations.

Section 2 discusses related work. Section 3 outlines the PET model. Section 4 describes the experimental prototype. Section 5 presents data collected from our experiments and Section 6 concludes the paper.

2 Related Work

The cluster of power problems—power, energy, heat and heat density—has been studied extensively. The primary distinctions of our work are the multi-dimensional and goal-

This research was supported by an IBM University Partnership Award and DARPA Contract F33615-03-C-4106.

oriented aspects. We apply DVFS to an Intel Pentium M system to meet power, thermal, and performance targets. DVFS is a popular technique for power and energy management and is widely available in commercial systems, including implementations in AMD's PowerNow [1] and Intel's Enhanced SpeedStep [9] and dual-core Dynamic Power Coordination [11].

Juang, *et al.* propose closed-loop control to coordinate DVFS levels for individual cores within a chip multiprocessor to match frequencies to the amount of work in parallel segments between software synchronization points [10]. Ranganathan *et al.* employ a centralized, policy-driven controller to dictate power levels on individual blades primarily to meet total power budget constraints, with policies to minimize performance degradation [15]. Weissel, *et al.* implemented a hierarchy of *energy containers* within an operating system to track energy usage throughout a multi-core system, estimate temperature, and dynamically assign work to meet individual components' thermal budgets. Cameron, *et al.* developed PowerPack, a framework to measure and manage power in large-scale high-performance systems by lowering CPU frequency at opportune times to save power and energy (and therefore, operating costs) with minimal performance impact [2].

One solution designed for explicitly managing both power and temperature simultaneously is Intel's Foxton Technology [12], which employs a feedback control system to maximize processor frequency, and therefore performance, within a power-thermal envelope. As the processor power consumption and temperature vary with workload and environment, on-die sensors provide information every 8 μsec to the embedded Foxton micro-controller, which first enforces the thermal limits. If the processor is within a safe thermal zone, the controller raises or lowers frequency in small increments until it reaches a programmable power limit. Our work also provides the capability to maximize performance within power and thermal limits. PET also allows the user to express preferences within constraints and translates objectives on the fly to current conditions, to finely tune the system response.

3 PET

The PET management approach provides a level of indirection between **macro objectives**—such as throughput, operating cost, reliability—and **micro directives**—clock frequency, sleep modes, etc.—to allow true preferences expressed at a high level while providing direction to tight lower-level control. One obstacle to effective management is translating the desired outcome to multiple measured and controlled quantities. Our solution is a multi-dimensional space that allows the PET manager to evaluate trade-offs among each dimension simultaneously. The manager maps actuator settings to their expected outcomes within the space; each point in the space represents the effect of a p-state setting. A user (whether a human operator or management software) provides a goal that defines the desired outcome within the space.

Formal methods exist for reasoning about goals specified in the form of qualitative conditional preferences. Domshlak, *et al.* [3], introduce methods of approximating qualitative preferences such as "I prefer to save power when utilization is low." These preferences can be represented via Conditional Preference nets (CP-nets), but identifying optimal solutions on the resulting partially-ordered graphs is NP-hard. Domshlak introduces a mechanism for approximating CP-nets as soft-constraint satisfaction problems, taking into account both constraints and qualitative preferences simultaneously. In PET, the goal specification consists of three components, designed to capture constraints and conditional preferences in an interface suitable for controlling hardware systems:

1. **hard limits** for each dimension to enforce strict requirements for safety

2. **soft limits** for each dimension, defining a preferred region

3. **objective function** to identify the best candidate within the set of acceptable solutions.

The PET manager first finds the set of points that meet all hard constraints. If no such points exist, the manager retreats to a pre-determined 'safe mode' p-state and reports an error. In the typical case, multiple p-states would meet hard constraints, and the manager proceeds to find the set of points within the preference region bounded by soft constraints. The PET manager then evaluates that set within soft constraints and chooses the point with the maximum benefit as defined by the objective function.

Figure 1 shows a snapshot for one sample period of PET prototype operation. The manager collected power, performance, and temperature samples while the benchmark `gzip` executed at a 1200 MHz frequency p-state. Measured data for this point and expected outcomes for all other p-states, 8 total, are projected into a performance-power-thermal multi-dimensional space; details of the projection follow in Section 4. The plot illustrates the inter-related effects of performance, power, and temperature; higher-performance p-states yield higher power and temperatures. For illustration, consider a case where the highest point in the graph (x) would violate hard limits and the remaining points (⋄) are acceptable by safety standards, though the soft limits indicate the lowest 3 points are preferable. An objective function of "maximum performance" within the hard and soft constraints selects one p-state (*). Note

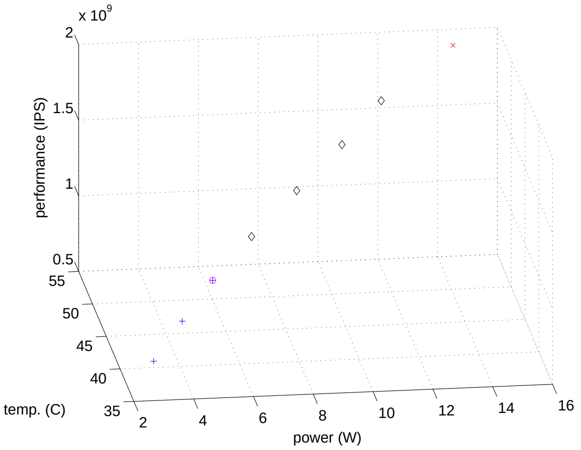

Figure 1. PET multi-dimensional goal space

that the goal syntax allows a performance-oriented objective within a preference region of lower power, allowing the user to tailor the goals to accurately reflect the objectives. The figure shows the multi-dimensional space for one snapshot in time; point locations within the space will change over time due to workload activity, ambient temperature, etc. The manager continuously loops through the sequence: **monitor** the system, **map** the multi-dimensional space, and **choose** an appropriate p-state to achieve the goal.

4 PET Prototype

4.1 Hardware

The prototype hardware consists of an Intel Pentium M 755 desktop processor system: a single-core “Dothan” series 90-nm processor supported by a Foxconn heat-sink and fan-assembly, an Intel 855GME chipset, 512 MB of DDR SDRAM memory and a Radisys uni-processor motherboard [13]. The motherboard resides in a conventional PC enclosure, with the top panel removed to allow access for probe cables. The photo in Figure 2 shows the Pentium M (left) with the probe cables for voltage and current measurements running to the data acquisition modules (right).

Intel’s Pentium M employs two power-management mechanisms: dynamic voltage and frequency scaling and clock throttling. DVFS supports 8 frequency-voltage pairs listed in Table 1, with 200 MHz steps from 600 MHz to 2.0 GHz, and voltages corresponding to the most conservative settings, VID#A in the processor datasheet [8]. Changing the DVFS setting incurs a stall up to 10μsec, effectively instantaneous at millisecond sampling time scales. The processor is also equipped with clock throttling, which effectively stops the clock for a specified percentage of the time. We studied clock throttling alone and in combination with DVFS and found that for this system, DVFS provided superior power-performance behavior. The rare situation that would warrant clock throttling would be to reduce power below the 2-3 watts which the 600 MHz p-state could provide; clock throttling was not used in this study.

Figure 2. Pentium M system (left) and measurement PC (right).

Frequency	Voltage
2000	1.340
1800	1.292
1600	1.244
1400	1.196
1200	1.148
1000	1.100
800	1.052
600	0.988

Table 1. Pentium M p-states: frequency and voltage pairs

We tapped the high-precision resistors between each of voltage regulator modules and the processor with a voltage probe, providing voltages to a National Instruments data acquisition system that monitors processor supply voltage and also calculates supply current from the voltage drop across the sense resistors. The monitor on the right in Figure 2 displays our custom virtual oscilloscope program in NI LabView software that displays voltage and current information and sends UDP packets of measured data to the Pentium M.

4.2 PET Prototype Software

We created prototype software for the Pentium M with customized drivers to monitor performance, power, and temperature and control DVFS settings. During benchmark execution, the software reads Pentium M performance counter values for Instructions Retired and Data-Cache Unit Miss Outstanding events with the RDPMC instruction at each sampling interval and generates an output file of timestamps and counter values at the conclusion of benchmark execution. At a nominal sampling rate of 100 samples per second, the sample interval length varies slightly, with most samples within 10-15 ms and a mean sample length of 13 ms. Power samples arrive in UDP packets at a rate of 100 samples per second. Temperature is sampled less often, at a user-controlled rate of 1 temperature sample per N performance samples (50 temperature samples per second in these experiments), via a custom driver to the LM85 fan controller chip.

The manager software initializes event counters and software timers, establishes a connection via UDP to the measurement PC, then spawns a benchmark as a child process with the highest user-level priority. The manager continously samples performance, power, and temperature measurements while the benchmark executes. During each sampling period, the manager uses data collected at the current p-state to estimate performance, power, and temperature for all p-states to populate the multi-dimensional space with expected outcome for present run-time conditions. The manager uses a performance metric of instructions per second, IPS, gleaned from hardware event counters and applies a non-linear performance estimator from [14] to predict performance at other frequencies, considering the present compute-bound or memory-bound application behavior. We explored several power models, with a range of complexity and accuracy; for this work, we use a simple power model:

$$P_{predicted} = \alpha P_{measured} + \beta \tag{1}$$

where $P_{measured}$ is the most-recent measured power value at a known p-state. Coefficients α and β are empirically derived through regression analysis of SPEC CPU2000 benchmarks with the ‘TRAIN’ input set, with separate coefficients for each p-state. We constructed a linear model for temperature estimation from regression analysis of empirical ambient temperature and power for steady-state microbenchmarks measured at each p-state. The model captures the effects of both environmental conditions and power consumption on the CPU temperature:

$$T_{cpu} = (\tau * P_{predicted}) + T_{ambient} \tag{2}$$

where τ varies slighly with CPU activity due to spatial distribution of chip hotspots relative to the single temperature sensor in the processor package. We applied $\tau = 1.25$ for all benchmarks in experiments presented in this paper. In reality, CPU and ambient temperatures affect each other; a rising CPU temperature causes a rise in ambient, even with the maximum cooling capacity. This model neglects the effect of the CPU temperature on the ambient due to the slow rate of ambient temperature change; the ambient temperature is approximately steady during the next sample, regardless of CPU temperature changes.

The PET manager software compiles estimates for three dimensions (performance, power, and temperature) for each p-state, then proceeds to evaluate each p-state’s outcome relative to the goal. Initial goal parameters are specified on the manager command line, and can be adjusted via software signals `SIGUSR1` and `SIGUSR2` during execution. The manager finds the set of p-states that satisfy both hard and soft constraints, calculates the benefit of each point according to the objective function, and chooses the p-state with the maximum benefit. Linux drivers change the CPU voltage and frequency to the selected p-state setting. The process of monitoring system status, evaluating projected outcomes, and directing p-state changes repeats each sampling period.

5 Experimental Results

In this section, we use our single-processor prototype to illustrate how the PET approach guides the system to a desired outcome by adapting p-state choices in response to variable run-time environment and workload behavior. The capability to sense present conditions on the fly and take appropriate action at the processor level demonstrated here applies to large-scale systems, as well, where a centralized controller can provide global goals and low-level decisions can be distributed throughout the system to local managers. In the first experiment, the manager chooses appropriate p-states to provide the best performance within fixed power and thermal limits. In the second experiment, we illustrate a scenario where the goal changes as a user moves the desired operating point along a spectrum of performance to cost trade-offs.

5.1 Experiment 1: Fixed Goal, Variable Workload

In the first experiment, the manager’s goal is the best performance within a power and thermal budget. Hard limits delineate strict bounds of 17 Watts and 70 C. Soft limits impose additional preferences for a 10-Watt power budget and 43 C thermal limit and the objective function is “maximize performance.” The workload is the `ref` input set of the SPEC CPU2000 suite, consisting of 11 integer and 15 floating-point applications. For reference, the SPEC CPU2000 suite completes in approximately 70 minutes using the maximum frequency of 2 GHz on our Pentium M system, with an average power consumption of 13 Watts. Graphs in Figure 3 show how the manager directs frequency changes in response to varying workload demands to meet the fixed goal specification. The CPU frequency hovers between 1200 and 1400 MHz with excursions up to the maximum 2000 MHz and down to 1000 MHz, tracking the workload intensity.

The next graph illustrates the power profile; most measurements are recorded between 5-10 Watts, with occasional lower- and higher-power points. The unpredictable timing of the power feedback by UDP in this experimental system causes outdated information from periods of previous low activity to adversely affect frequency choices, causing higher power than the budget would allow for the *current* activity. Even with unreliable power information, the manager rarely violates hard power constraints in these experiments, overshooting the hard constraints for a total of 91 milliseconds out of 99.1 minutes of execution, approximately 0.0015% of total execution time. The manager respects soft power limits for most decisions, violating the preference bounds for 0.52% of the execution time with an average of 1 Watt, a 10% overshoot. The temperature plot shows that measured temperatures were well within a generous 70-C hard limit (not shown on this scale). The CPU temperature occasionally breached the 43 C soft-limit, 0.5% of the time with a mean of 1.1 C overshoot.

While the PET manager intends to respect all constraints, run-time system behavior does not always match the manager's expectations. Using previously measured data to predict future response poses a fundamental problem for controlling variable system behavior. In addition, the prototype PET manager relies on low-overhead estimates of power, performance, temperature, etc. based on sensor data recorded for one p-state and extrapolated to all other p-states. Estimation error in each prediction increases the the risk of constraint violations. However, permitting occasional mis-steps—with the capability to correct overshoots quickly—allows the system to operate near constraint boundaries without being hampered by more conservative margins.

Overall, the manager chooses appropriate p-states on the fly to meet the desired goal with variable workload activity in most cases. Mild constraint violations such as those observed this experiment may be acceptable in most situations. Systems that do require tighter control could add additional safety measures, including conservative guardbands on the constraints, more detailed estimation models, or upgrades in monitoring hardware (such as integrated power monitoring features available on blade servers).

5.2 Experiment 2: Variable goal and workload.

The next challenge is to demonstrate that the manager is capable of tracking variable workload behavior while the goal also varies. Conceptually, experiment 2 emulates an operator moving a sliding bar between two extremes, high performance and lower operating cost, to indicate the current desired outcome as priorities shift due to task urgency, quality-of-service contracts, system load, etc. In this prototype system without a graphical interface, user actions are conveyed by a bash shell script that sends signals to the manager's process to modify the soft limits, nudging the soft-limit boundary one notch either toward better performance with SIGUSR1 or toward lower cost with SIGUSR2 every 4 seconds. The arbitrary pattern of SIGUSR1 and SIGUSR2 signals mimics the change in objectives over time by an external operator, independent of application activity. The objective function is set to 'maximize performance' and hard limits are fixed at 80 C and 17 Watts, and 50% performance. Performance requirements specify the minimum acceptable instructions per second (IPS), as a percentage of IPS expected at the maximum frequency. Soft limits are initially set to 75 C, 14 Watts, and 80% of max IPS.

Throughout benchmark execution, the software signals representing external inputs intermittently modify the goal. SIGUSR1 increases soft limits and SIGUSR2 decreases soft limits; each signal received adjusts the limits by +/-10% performance, +/-5 degrees C, and +/-1 Watt.

Graphs in Figure 4 show how soft constraints change in each dimension. Note that as the soft limits shift in response to the user preference, they may exceed hard limits, in which case the manager overrides user preferences and clips the soft region boundaries to the hard limit values.

The first plot shows performance as IPS. The hard limit is 50% of max IPS. The soft limit slides higher or lower every few seconds; the measured performance should be equal or greater to this minimum performance preference. The power plot illustrates the soft power limit adjusting in concert with the performance limit. As the user indicates higher performance, the power budget is relaxed; for lower operating costs, the power budget becomes tighter. The thermal plot also shows the hard and soft limits, although in this experiment, temperature was not a limiting factor, as evidenced by the slack between the limits and measured values. The frequency plot illustrates the fluctuating frequency levels chosen to meet the moving target and workload behavior. The frequency ranges from 1000 MHz at the more restrictive power budgets to 2000 MHz for regions of higher performance expectations and/or workload behavior that could meet the power budgets at higher frequencies.

The manager meets the performance hard requirement and maintains performance near the soft limits, violating the soft limits 0.3% of execution time with a slight undershoot of 0.06% on average. The manager violates power hard constraints in 0.01% of execution time and power soft limits for 12.5% of execution time. The manager does not violate hard or soft temperature constraints in this experiment.

The greater number of violations in this experiment is due primarily to exposing regions of lower fidelity in the estimation models as the system moves through a wider range of operation as the goal shifts from higher performance

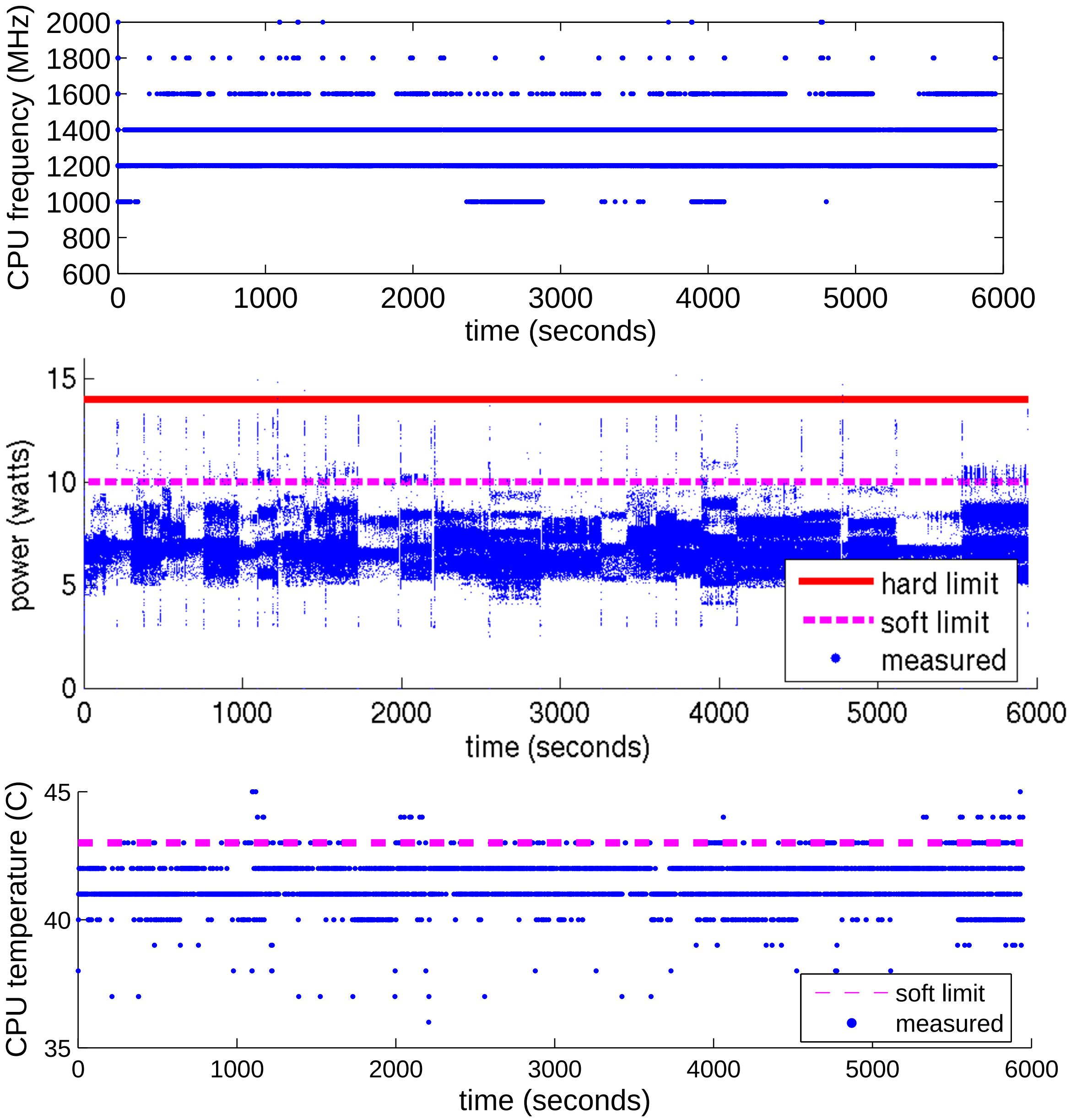

Figure 3. Experiment 1: Fixed goal, variable workload. SPEC CPU2000 suite executes with a goal of maximum performance given hard and soft power and thermal limits.

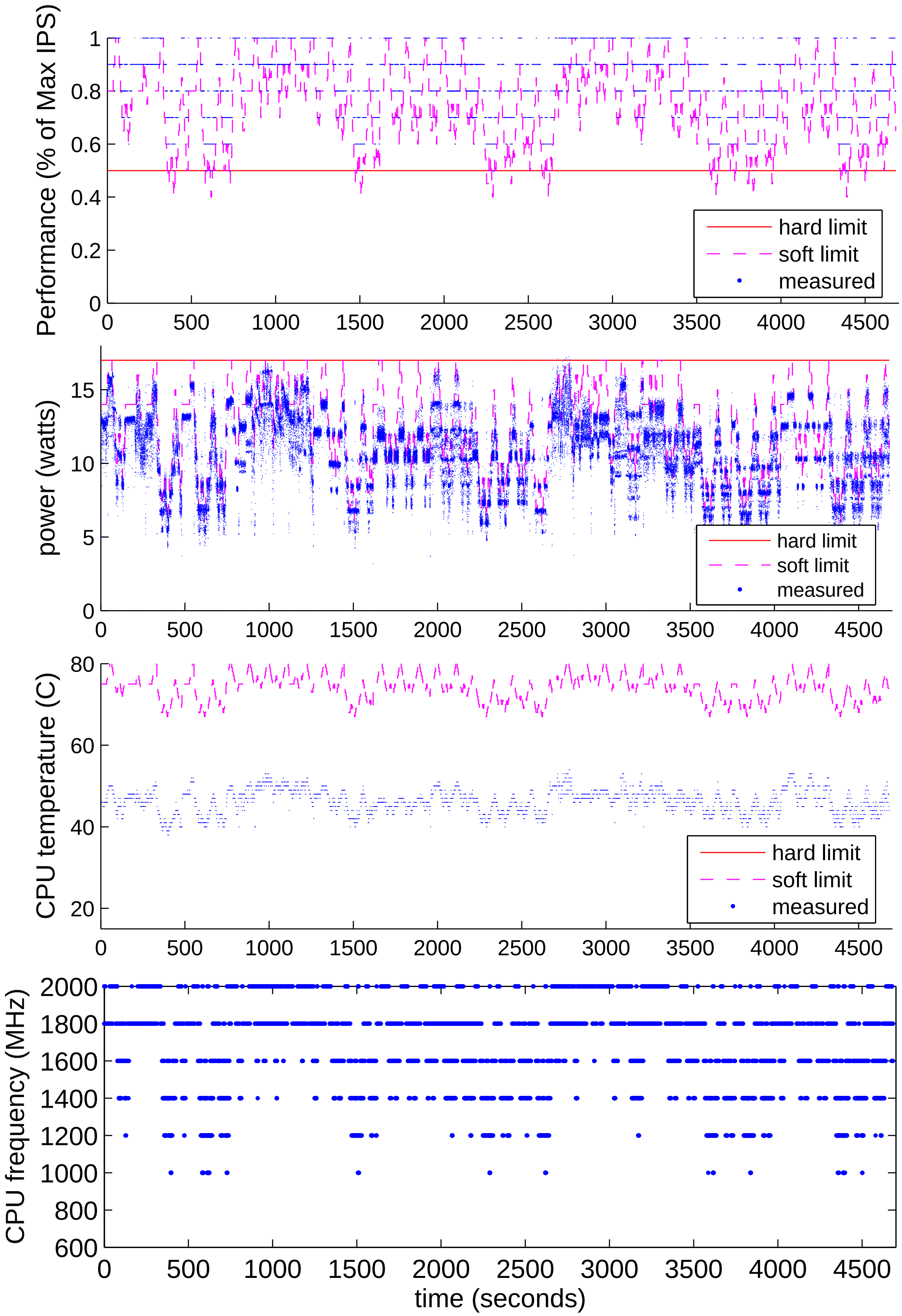

Figure 4. Experiment 2: Variable goal and workload. SPEC CPU2000 suite executes while the goal shifts through a spectrum between better performance and lower operating costs.

through lower-cost targets. It is important to note that despite the limited accuracy afforded by imprecise models, the manager maintains control of the system due to continuous monitoring that provides opportunities to re-evaluate p-state choices, every 10-15 ms.

6 Conclusion

In this work, we demonstrate a multi-dimensional, goal-oriented approach to achieving high performance within power and thermal constraints. We describe **PET**, a run-time manager that translates macro objectives from the user level to micro directives in the hardware, tailored to workload and environmental conditions.

We illustrate the key features with a prototype implementation in user-level software that manages a Pentium M system. In this paper, we demonstrate that the PET approach continuously adapts to workload behavior as it strives to meet target operating points in two situations. First, we show how PET maximizes performance within power and thermal budgets by adapting DVFS settings to dynamic workload behavior with the SPEC CPU2000 suite, both integer and floating-point benchmarks, with the REF input set. In this case, the prototype manager generally adheres to hard constraints throughout execution and overshoots soft limits with minor excursions. In the second set of experiments, both the workload behavior and the goal modulate during execution. In this scenario, the manager violates soft limits more often as it handles a wider range of operation; in the worst case, it violates power soft limits for 12.5% of execution time, though it maintains performance within preferred range 99.7% of the time.

This paper demonstrated the feasibility of the PET approach with a single-processor system. The prototyped approach is extensible to management of additional constraints, and holds promise for large-scale systems with a hierarchy of PET managers that each control a local region and together manage a larger system. In a cluster-level PET manager, goal specifications would be determined by system administrators, either tuned by human operators or automatically updated by operational utilities such as IBM's PowerExecutive [6].

References

[1] Advanced Micro Devices. PowerNow with optimized power management. http://www.amd.com/us-en/0,,3715_12353,00.html, Jan. 2006.

[2] K. Cameron, R. Ge, and X. Feng. High-performance, power-aware distributed computing for scientific applications. *IEEE Computer*, pages 40–47, November 2005.

[3] C. Domshlak, F. Rossi, B. Venable, and T. Walsh. Reasoning about Soft Constraints and Conditional Preferences: Complexity Results and Approximation Techniques. In *Proceedings of International Joint Conferences on Artificial Intelligence*, pages 215–220, 2003.

[4] PRIMERGY BX300 Datasheet. issue date: November 16, 2004, http://www.fujitsu-siemens.com.tr/le/products-/standard_servers-/blade/primergy_bx_300.html .

[5] ProLiant Datasheet. issue date: October 8, 2004, http://h18000.www1.hp.com/products/quickspecs-/11686_div/11686_div.pdf.

[6] IBM. Managing Server Energy Consumption Using IBM PowerExecutive. ftp://ftp.software.ibm.com/common/ssi/rep_wh/n/-XSW02410USEN/XSW02410USEN.PDF, May 2006.

[7] 2.0 GHz Pentium M Integrated XSeries Server. http://www-03.ibm.com/systems/i/bladecenter/ixs/4812001.html .

[8] Intel. Pentium M Processor on 90 nm Process with 2-MB L2 Cache Datasheet, January 2005. http://www.intel.com/design/ mobile/ datashts/ 302189.htm.

[9] Intel Corp. Enhanced Intel SpeedStep technology. http://support.intel.com/support/processors/mobile/pm/sb/ CS-007981.htm, Jan. 2006.

[10] P. Juang, Q. Wu, L.-S. Peh, M. Martonosi, and D. W. Clark. Coordinated, Distributed, Formal Energy Management of Chip Multiprocessors. In *Proceedings of the 2005 International Symposium on Low Power Electronics and Design (ISLPED)*, pages 127–130, 2005.

[11] A. Nanduri. Dynamic Power Coordination. http://www.intel.com/products/processor/coreduo-/dynamicpowercoordination.htm, 2006.

[12] C. Poirier, R. McGowen, C. Bostak, and S. Naffziger. Power and Temperature Control on a 90nm Itanium-Family Processor. In *IEEE International Solid-States Circuits Conference 2005*, pages 304–305, March-April 2005.

[13] Radisys Corporation. Endura LS855 Product Data Sheet, October 2004. http://www.radisys.com/oem_products/ ds-page.cfm? productdatasheetsid=1158.

[14] K. Rajamani, H. Hanson, J. Rubio, S. Ghiasi, and F. Rawson. Application-Aware Power Management. In *Proceedings of the IEEE International Symposium on Workload Characterization*, October 2006.

[15] P. Ranganathan, P. Leech, D. Irwin, and J. Chase. Ensemble-level Power Management for Dense Blade Servers. In *Proceedings of ISCA*, pages 66–77, 2006.