Predictive Power Management for Multi-Core Processors

William Lloyd Bircher, Lizy John

Predictive Power Management for Multi-Core Processors

W. Lloyd Bircher and Lizy John
The University of Texas at Austin

ABSTRACT
Predictive power management provides reduced power consumption and increased performance compared to reactive schemes. It effectively reduces the lag between workload phase changes and changes in power adaptations since adaptations can be applied immediately before a program phase change. To this end we present the first analysis of prediction for power management under SYSMark2007. Compared to traditional scientific/computing benchmarks, this workload demonstrates more complex core active and idle behavior. We analyze a table based predictor on a quad-core processor. We present an accurate run-time power model that accounts for fine-grain temperature and voltage variation. By predictively borrowing power from cores, our approach provides an average speedup of 7.3% in SYSMark2007.

1 INTRODUCTION

The challenge in applying power management to increase efficiency and performance is identifying when to adapt performance capacity. In the case of controlling dynamic voltage and frequency scaling, the ubiquitous commercial solution has been to react to changes in performance demand. While this approach is simple, it performs sub-optimally [4][22] for workloads with many distinct and/or short phases. Each time a workload transitions from a phase of low performance demand to a phase of high demand, reactive power management increases performance capacity some time *after* the transition. During the time between the change in demand and capacity, performance is less than optimal. Similarly, power consumption is sub-optimal on transitions from high to low demand. The amount of performance loss is proportional to the number of phase changes in the workload and the lag between demand and capacity. For increasing performance in power-limited situations, reactions must be fast enough to prevent overshooting the power limit or missing opportunities to increase performance.

To date analysis in this field has focused on predicting phase changes within scientific/computing workloads such as SPEC CPU. Since these workloads contain only active phases in which the processor is non-halted, a significant amount of valuable phase information is ignored. In a "real" system much of the time is spent idle waiting for user input or responses from the I/O subsystem. The power difference between the active and idle phase is much larger than what is observed within a fully active phase. To address this missed opportunity, we analyze a table-based phase predictor under the SYSMark2007 workload.

This benchmark is better than conventional scientific/computing benchmarks at providing complex workload variation for power management analysis. GUI interaction, keypress delays, I/O, thread migration and a wide range of processor activity levels are provided. It includes popular desktop/personal computing applications such as Microsoft Word, Excel, PowerPoint, Adobe Photoshop, Illustrator, etc. Unlike traditional, benchmarks which are composed of a single active phase, this one spends greater than 40% of the time in phases less than 1 second in duration. The analysis is performed using our proposed tabled-based activity predictor. This predictor simplifies existing implementations by using core activity as the predicted metric. It achieves accuracy of 98% for 43% of SYSMark2007. The remaining portion is detected as unpredictable and is handled reactively. To estimate power consumption of this approach, we present a detailed run-time power model for an AMD quad-core processor, utilizing core-level performance metrics. We improve upon existing models by accounting for leakage power and temperature effects and the impact of power management. This model has an average error rate of 0.89% across SPEC CPU 2006 and SYSMark2007.

2 PREDICTION FOR POWER MANAGEMENT

Unlike prior research in prediction for power management [9][6][11][17][1] we choose core activity and power as the predicted metric. Most others predict IPC[6][11] or memory-boundedness[9][11]. Their approach relies on detecting program phases of execution which benefit less from high processor clock frequency. These phases are usually defined as memory-bound. The degree of memory-boundedness is usually estimated using IPC, memory access intensity or a combination of both. The problem with this approach is that no application is 100% memory bound. Therefore, any reduction in processor clock frequency will result in a performance loss. These approaches rely on the tradeoff of a small performance loss for a larger power savings.

Instead we choose to predict core activity levels. In this case core activity is defined as the ratio of non-halted cycles to non-halted + halted cycles. Since by definition a halted (idle) core has 0% performance dependence on core frequency, there is no need to accept a performance loss in order to save power. This is the essence of existing c-state power management techniques [3]. Since most existing dynamic power management (clock gating and power gating) use core halting as the key to entry, core activity can represent program phase changes which

have the largest impact on power consumption. For the quad-core processor considered in this study, IPC variation is only responsible for power fluctuations of 20% of peak power compared to nearly 90% for core activity. Core activity is used by all existing Windows and Linux operating systems for control of DVFS. For this reason, it is also an appropriate prediction metric for out of band techniques such as power boosting [14][8][24] and power limiting. These techniques respond to changes in power consumption, as dictated by operating systems and applications, to improve performance or ensure safe operation

3 PROGRAM PHASE CHARACTERIZATION

To understand power-relevant phases in real world workloads, we characterize the program behavior of the desktop/client SYSmark® 2007 benchmark [19]. This benchmark represents a wide range of desktop computing applications. The major categories are e-learning, video creation, productivity, and 3D. The individual subtests are listed in Table 1. This benchmark is particularly important to the study of dynamic power adaptations because it provides realistic user scenarios that include GUI interface in I/O interactions.

To illustrate the potential of this workload for the analysis of core activity prediction, we present traces of core activity level and duration. Figure 1 shows the number of cores used concurrently by the workload over time. At each point where the number of cores changes, a phase transition occurs as idle cores become active or vice versa. This is distinct from traditional benchmarks such as SPEC CPU which uses a fixed number of processors for the entire benchmark run. Therefore, there are essentially no core activity changes. As a result power fluctuations are much less since variation comes only from changes in pipeline utilization (IPC).

Table 1: Benchmarks Used: SYSmark® 2007

E-Learning	Video Creation
Adobe® Illustrator®	Adobe After Effects®
Adobe Photoshop®	Adobe Illustrator
Microsoft PowerPoint®	Adobe Photoshop
Adobe Flash®	Microsoft Media Encoder
	Sony Vegas
Productivity	**3D**
Microsoft Excel®	Autodesk® 3Ds Max
Microsoft Outlook®	Google™ SketchUp
Microsoft Word®	
Microsoft PowerPoint	
Microsoft Project®	
WinZip®	

The actual number of phase changes in SYSMark is even greater since the traces in Figure 1 uses averaging of samples to smooth the trace for readability. Figure 2 quantifies actual phase variation under Windows Vista. The majority of phases are much shorter than one second. In all cases an average of at least 40% of phases are shorter than one second. There are two factors that cause high rates of phase changes: low overall activity and thread migration by the operating system. Considering only Figure 2 it might be concluded that two or more cores remain idle for almost the entire benchmark. However, this is not the case.

Figure 2 shows that all four cores experience large numbers of phase transitions. The reason is thread migration. Most threads only remain on any given core for 10s of milliseconds before being preempted. Since Windows does not aggressively maintain thread locality (application controlled affinity does), preempted threads tend to continuously migrate among all available cores. Since there are consistently more cores than are required by the workload, there are frequent transitions from idle-active and active-idle.

4 METHODOLOGY

4.1 Power Measurement

To measure power consumption for the development of a power model, we instrument an AMD Phenom2 3GHz system. Processor core power consumption is measured using a hall-effect sensor placed in-line with the core power rail (Core VDD). This sensor produces an output voltage which is linearly proportional to current. We also measured voltage levels at the point where the current enters the processor socket. We perform all sampling at a rate of 1 MHz, using a National Instruments NIUSB-6259 [15]. This granularity allows the measurement of most power phases which were sufficiently long to perform adaptations. Though shorter duration phases exist, current adaptation frameworks are not able to readily exploit them.

4.2 Performance Counter Measurement

We use performance monitoring counters (PMCs) to track core activity and power. We develop an on-line, PMC-based, core-level power model. It is necessary to use a power model since it is impossible to measure core-level power consumption since all cores share a single power plane [3]. Physical instrumentation thus cannot provide core-level power, whereas the power model using performance counters can provide power estimates from individual cores. Prior researchers have shown that accurate power models can be built using performance counter measurements [2][10][5]. Power models from prior researchers are extended to include leakage power, temperature effects and the impact of power management that are present in modern multi-core processors. These models are in fact preferred for dynamic power management since there is no need to measure power with out-of-band instrumentation. The accuracy of the models was verified with controlled core level and aggregate measurements.

To sample performance monitoring counters we develop a small kernel which provides periodic sampling of the four AMD performance counters. This kernel uses a device driver to give ring-0 access to user-mode applications. This approach is preferred over existing user-mode

performance counter APIs as it affords more precise control of sampling and lower overhead. In all experiments the sampling overhead for performance counter access average less than 1%.

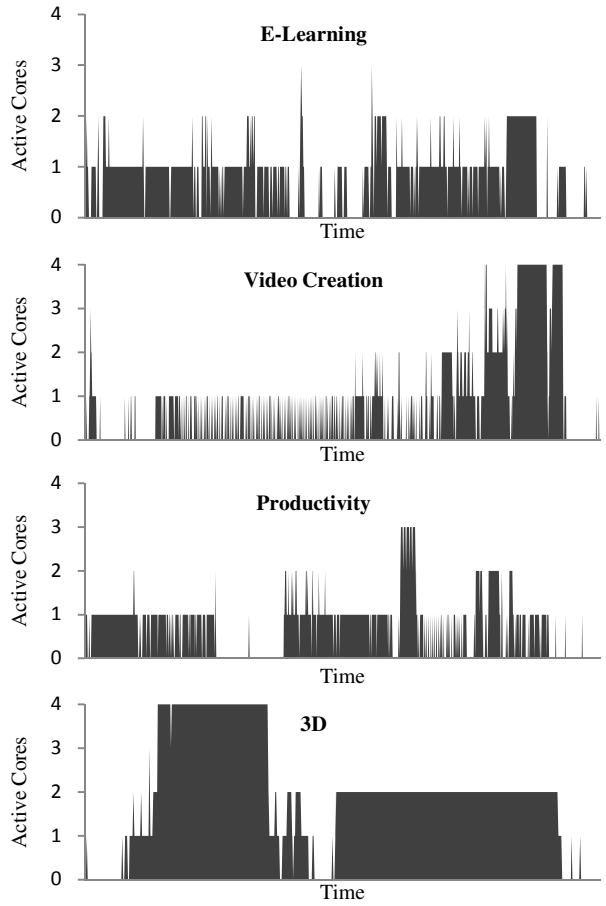

Figure 1: Concurrent Core Activity by SYSmark® 2007 Benchmark

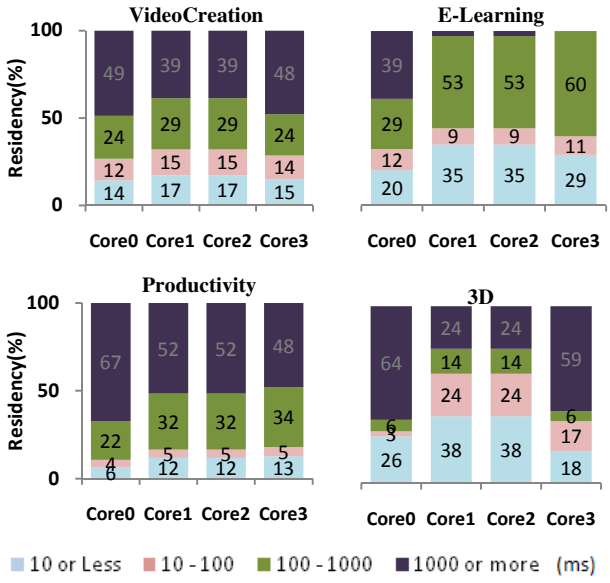

Figure 2: Core Activty Duration

Another benefit of our device driver is that it provides access to others registers besides performance counters. In particular, our approach requires access to model specific registers (MSRs) and PCI configuration registers. These registers allow our application to take control of processor frequency, voltage, power management. It also gives access to on-die processor temperature sensors. This is required to account for static power consumption. Finally, sampling is invoked at a user-specified periodicity user the built-in OS multimedia timer [20].

5 CORE ACTIVITY PREDICTOR

This predictor uses table-based prediction structures and the repetitive nature of core activity phases to predict performance demand and/or power consumption. The predictor is shown in Figure 3.

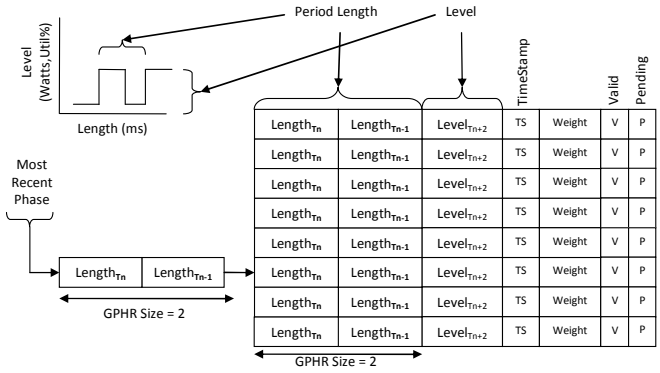

Figure 3: Power Phase Predictor

Like traditional table-based predictors the main components are: a global phase history register (GPHR), pattern history table (PHT) and predicted level. The distinction is that the predictor uses a PHT length of two entries and records the level of the next phase only. Since we define core activity to have only two possible values: 1 and 0 (100% and 0%), there is no need to record the level of each phase as is done in traditional run length encoding predictors [18]. For example if the predicted level is 1, then the previous level was 0 and the level before that was 1. The PHT size is modest at 48 entries. This compares to Isci [9] which uses a GPHR of 8 and PHT of 128 in order to obtain accuracy better than statistical predictors. Each predictor entry also contains several fields that capture quality, validity, and pending prediction status. The fields are described in Table 2.

Table 2 Predictor Field Descriptions

Predictor Field	Description
Length	Duration of phase. This is also the table index. When a phase is detected, it is used to index the prediction table.
Level	Predicted level at next transition. For the activity predictor this is active or idle. For power prediction this is the last power level seen when this phase occurred.
Timestamp	Records timestamp of when predicted phase

	change is to occur. This is the most critical value produced by the predictor. It is used by the power manager to schedule changes in power/performance capacity of the system. This value allows for optimal selection of performance capacity given the anticipated duration of operation at a particular performance demand.
Weight	"Quality" of phase as a function of past predictions and duration. The weight is used by the power manager to determine if a prediction will be used or not. It is also used by the replacement algorithm to determine if the phase will be replaced if the predictor is full. All newly detected phases start with a weight of 1. If the phase is subsequently mispredicted, the weight is reduced by a fixed ratio.
Valid	Indicates whether this entry has a valid phase stored with a "true" or "false."
Pending	Indicates if this phase is predicted to occur again. This value is set "true" on the occurrence of the phase and remains true until the predicted transition time has past.

6 CORE-LEVEL CPU POWER MODEL

Using a real system instrumented for power measurement we develop polynomial, regression models for power consumption. The details of the model are given in Tables 3 and 4. The model improves on existing on-line models [2][5][10] by accounting for power management and temperature variation. All model coefficients are tuned empirically using a real system instrumented for power measurement. Like existing models this one contains a workload dependent portion which is dominated by the number of instructions completed per second. In this case we use the number of fetched operations per second in lieu of instructions completed. The fetched ops metric for Phenom2 processors is preferred as it also accounts for speculative execution. The distinction of our model is that it contains a temperature dependent portion. Using workloads with constant activity, we vary processor temperature and voltage to observe the impact on static leakage power. Temperature is controlled by adjusting the speed of the processor's fan. Temperature is observed with 0.125 degree Celsius resolution using an on-die temperature sensor[3]. This sensor can be accessed by the system under test through a built-in, on-chip register. Voltage is controlled using the P-State control register. This allows selection of one of five available voltage/frequency combinations. Voltage is observed externally using our power instrumentation. Like the workload dependent model, we tune the coefficients of the static power model using regression techniques. Note that the static power model is highly process dependent. Processors with different semiconductor process parameters require the model to be re-tuned.

The dominant power management effects (voltage/frequency scaling, clock gating) are further accounted for using the gateable and ungateable power models. Gateable power is found by measuring the effect of enabling/disabling idle core clock gating. Ungateable represents the portion of power which cannot be gated. These components are also found experimentally. The resultant, average error in the model was 0.89%. The error distribution for SPEC CPU2006 and SYSmark2007 has the first standard deviation with less than 1% error. Worst-case error is 3.3%.

Table 4: Measured Quantities (Model Inputs)

Quantity	Description
N	Core Number.
FetchOps	Micro operations fetched. Includes speculative operations.
FloatPointOps	Floating point operations retired. Accounts for difference in power between INT/FP.
DCAccess	Data cache access. Accounts for power consumed in caches.
%Halted	% of cycles in which the core was halted.
%Halted	% of cycles the core was not halted.
Voltage	Maximum requested voltage for all cores. Due to shared voltage plane.
Frequency	Current core frequency. This can be read via AMD model specific register.
Temperature	Current processor temperature. This can be read via AMD model specific register.
Coeff	Model coefficient. The values are determined empirically using measurement/regression.

Table 3: AMD Quad-Core Phenom2 Power Model

Power Models	Equation
Total Power	$\left(\sum_{N=0}^{3} WorkloadDependent_N + Ungateable_N + Gateable_N\right) + Static_{Volt,Temp}$
Workload Dependent	$((\text{FetchOps}_N/\text{Sec})\cdot\text{Coeff}_F+(\text{FloatPointOps}_N/\text{Sec})\cdot\text{Coeff}_{FP}+(\text{DCAccess}_N/\text{Sec})\cdot\text{Coeff}_{DC})\cdot\text{Voltage}^2$
Gateable	$(\%\text{Halted}_N)\cdot\text{Coeff}_{Gateable}\cdot\text{Voltage}^2\cdot\text{Frequency}_N$
Ungateable	$(\%\text{NonHalted}_N)\cdot\text{Coeff}_{Ungateable}\cdot\text{Voltage}^2\cdot\text{Frequency}_N$
Static	$(\text{Temp}^2\cdot\text{Coeff}_T^{\,2}+\text{Temp}^1\cdot\text{Coeff}_T^{\,1}+\cdot\text{Coeff}_T^{\,0})_{\text{VoltageN}}$

7 RESULTS

This section provides power and performance estimates of our phase predictor in comparison to commercial reactive schemes. First we consider the prediction accuracy and coverage. Then, we estimate performance speedup due to borrowing power from under-utilized cores using prediction.

7.1 Performance Metric Definitions

To analyze the effectiveness of core activity prediction, we define three metrics: weighted match ratio, weighted hit ratio, and prediction coverage. These metrics differ from traditional branch predict / cache hit/miss metrics in that they describe the portion of the total program execution they represent. For example, a frequently predicted phase may only cover a small portion of the time if the phase is short. These get proportionally less weight. A long phase that occurs less gets a greater weight as it represents more time. A system with slow

adaptation rates may assign more value in predicting long phases, while a fast-adapting system may need greater precision for shorter phases since they are viable for adaptation. These metrics are similar in function to those proposed by Isci [11] for the analysis of a phase duration predictor.

The *weighted match ratio* is a measure of predictor utilization weighted by phase length. When a phase transition occurs, if a phase of that length has been observed previously by the predictor, it is considered a match. This is similar to a tag match. It does not mean the prediction was correct -- just that a similar phase has been seen recently.

Weighted hit ratio indicates the outcomes of predictions weighted by phase length. For example, a 10ms phase prediction has twice the weight of a 5ms phase prediction. The net result is that weighted hit ratio favors correct outcomes on long-duration phases.

Prediction Coverage. Due to the cost of misprediction, predictions are only considered valid if the outcome has a high probability of success, similar to a confidence predictor[12]. Using the *Weight* value, maintained for each PHT entry, difficult-to-predict phases are ignored. This approach shifts the assessment of prediction quality from hit rate to the portion of the workload that can be accurately predicted. The power manager only adapts predictively for high-probability phases. The remainder of the time, the power manager behaves reactively.

7.2 Quantitative Comparison

Table 5 summarizes the performance of our best case predictor configuration, 48-entry PHT, at anticipating phase changes. In all subtests, the predictor maintained an average weighted match ratio over 95%. This indicates that the predictor is fully utilized, and is not oversized. Increasing the PHT size beyond 128 caused weighted match ratio to drop below 50%. This suggests that the complete range of unique phases can be captured in a predictor of about 64 entries. Hit rates show diminishing returns above a PHT size of 48. Average weighted hit rates were also above 95% for all subtests (98% overall). Since our predictor only predicts phase transitions it has previously observed, the most critical result is prediction coverage percentage. In the worst-case, Productivity, was predictable 33.5% of the time. During that time, the predictions were almost always correct. For the cases in which a prediction is not possible, the predictor behaves as a *reactive* predictor.

We compare these results to the reactive p-state slection scheme used by Windows Vista[16]. This configuration is representative of the majority of commercial, operating system directed, reactive DVFS algorithms in use today [4]. Since the reactive "predictors" do not make use of any success metrics, the prediction coverage is always 100%. Note, weighted match ratio is not applicable here since no comparison is made to past phases. For the reactive scheme, the critical metric is weighted hit ratio. Hits/misses are determined by configuring the Windows Vista DVFS algorithm to sample core activity on 100ms intervals [16]. By tracing core activity and the resultant DVFS selections made by the operating system we are able to detect when correct predictions are made. For example, if an active phase occurs yet the operating system selects a low core frequency, a miss is counted. Since the AMD Phenom2 supports five DVFS frequency/voltage operating points, the "correct" frequency was determined using the operating systems own algorithm. The algorithm works on the concept of an ideal average activity. If the reactive scheme achieved the ideal activity with the selected frequency, it is counted as a hit.

On average the reactive scheme is wrong 35% of the time, though it is able to "predict" 100% of the time. Note that since the predictor can identify the non-predictable phases it is possible to increase the coverage of the predictor by simply performing reactively when a prediction is not possible. If the workload is assumed to have a uniform distribution of predictable phases, then the predictor should equal it in non-predictable phases (57% of the time) and surpasses it in predictable phases.

Table 5: Prediction Outcomes – Core Activity

Predictor, 48-Entry PHT				
Workload	**Weighted Match Ratio**	**Weighted Hit Ratio**	**Prediction Coverage**	**Effective Hit Ratio**
E-Learning	92.8%	98.8%	57.0%	82.6%
Productivity	95.5%	95.8%	33.5%	73.8%
Video Creation	95.1%	97.7%	43.0%	76.4%
3D	98.3%	99.1%	37.9%	72.7%
Reactive				
Workload	**Weighted Match Ratio**	**Weighted Hit Ratio**	**Prediction Coverage**	**Effective Hit Ratio**
E-Learning	100%	66.4%	100%	66.4%
Productivity	100%	65.2%	100%	65.2%
Video Creation	100%	63.5%	100%	63.5%
3D	100%	59.7%	100%	59.7%

7.3 Predictive Frequency Boosting

The second application of core activity prediction is boosting performance in active cores by borrowing power from under-utilized cores. To estimate the impact of our approach we scale power, performance, and core activity traces from a real system. For each point in the traces we consider the available power surplus and the probability that the predictor is able to identify a predictable phase. If it is determined that performance can be increased, we scale (shrink) the duration of the current phase at 75% of the frequency increase, to account for the speedup. Our analysis showed that, on average, SYSMark07 core frequency sensitivity is 75%. If a phase is not predictable, then no frequency change is applied. The major distinction between this analysis and that of the previous section is that the predictor predicts power rather than core activity. Phases are still defined in terms of core activity. However, an additional field is included to record the next power level. This is similar to

Duesterwald's cross-metric predictors [6] which predict one metric based on another.

Note that the maximum performance increase is assumed to be 20%. This limit is determined by critical path delay at the maximum supported voltage. This processor, like many multi-core processors produced in the last few years is performance limited by the total power budget, not critical path delay of a given core. If total processor power consumption is less than the maximum, due to under-utilized cores, active cores can reach higher frequencies (at higher voltages).

A comparison of average error results for the predictor versus a reactive scheme are presented in Figure 4. These results were somewhat surprising given the high prediction rates observed in the previous section. On average the predictor achieved 86% accuracy compared to 83% for the reactive scheme. This suggests that the 35% of execution time identified as predictable, yielded only three percentage points of increase in accuracy compared to the reactive scheme. Since core activity phases are being predicted with high accuracy, the likely cause of lower than expected accuracy is the use of cross-metric prediction. Since power is predicted indirectly using activity level, aliasing of different power levels could be causing increased error. Considering individual subtests, the results were better. In the best case, video creation accuracy was 86% compared to 77% for the reactive scheme. In all cases, the predictor outperformed the reactive scheme.

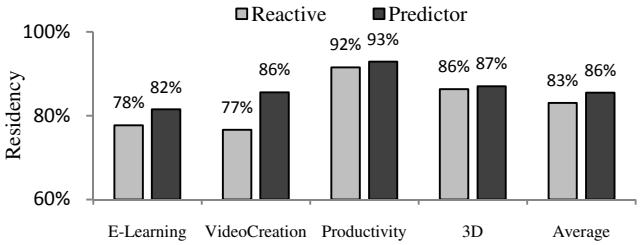

Figure 4: Core Power Accuracy Predictive vs Reactive

The best-case speedup of 11.1% was achieved on e-learning; a highly predictable and mostly single-threaded subtest. 3D had the lowest speedup at 5% partly due to low predictability, but more due to a lack of power surplus. Average speedup was 7.3%. The results are summarized in Figure 5. Speedup results are not provided for the reactive case is there is no foreknowledge of phase transition outcomes for avoiding boosting in unstable phases.

8 CONCLUSION

In this paper we have presented an analysis of core activity prediction in SYSMark2007. We show that this benchmark contains a large number of short duration phases that are relevant for power management. We demonstrate that these phases can be predicted with 98% accuracy 43% of the time. An estimate of performance increase due to power borrowing is presented, showing a potential speedup 7.3%. The analysis makes use of a temperature and voltage sensitive run-time power model that achieves 99.1% accuracy.

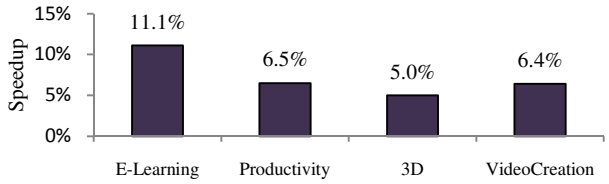

Figure 5: SYSmark 2007 Speedup

8 REFERENCES

[1] Annavaram, M., et al. The Fuzzy Correlation between Code and Performance Predictability. International Symposium on MIcroarchitecture, pp 93-104, 2004.

[2] Bellosa, F. The Benefits of Event-Driven Energy Accounting in Power-Sensitive Systems. Proceedings of 9th ACM SIGOPS European Workshop, pp. 37-42, September 2000.

[3] BIOS and Kernel Developer's Guide for AMD Family 10h Processor. http://www.amd.com. November 2007.

[4] Bircher, W. L. and John, L. Analysis of Dynamic Power Management on Multi-Core Processors. International Conference on Supercomputing, pp. 327-338, (Kos, Greece, June 2008).

[5] Bircher, W.L and John, L., Complete System Power Estimation: A Trickle-Down Approach based on Performance Events. International Symposium on Performance Analysis of Systems and Software, pp. 158-168, April 2007.

[6] Duesterwald, E., Cascaval, C., and Dwarkadas, S. Characterizing and Predicting Program Behavior and its Variability. Parallel Architectures and Compilation Techniques, pp. 220-231, September 2003.

[7] Govil, K., Chan, E., Wasserman, H. Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU. International Conference on Mobile Computing, pp 13-25 (Berkeley, California, 1995).

[8] Intel® Turbo Boost Technology in Intel® Core™ Microarchitecture (Nehalem) Based Processors, November 2008. http://download.intel.com/design/processor/applnots/320354.pdf?iid=tech_tb+paper

[9] Isci, C., Contreras, G., and Martonosi, M. Live, Runtime Phase Monitoring and Prediction on Real Systems with Application to Dynamic Power Management. International Symposium on Microarchitecture, pp. 359-370, December 2006.

[10] Isci, C. and Martonosi, M. Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data. In 36th ACM/IEEE International Symposium on Microarchitecture, pp 93Dec. 2003.

[11] Isci, C., Buyuktosunoglu, A., Martonosi, M. Long-term workload phases: duration predictions and applications to DVFS. IEEE MICRO, Vol 25, no. 5, pp 39-51, 2005.

[12] Jacobson, E., Rotenberg, E., and Smith, J., "Assigning Confidence to Conditional Branch Predictions", Proceedings of the 31st International Symposium on Microarchitecture, 1998.

[13] Lau, J., Schoenmackers, S., and Calder, B. Structures for Phase Classification. IEEE International Symposium on Performance Analysis of Systems and Software, pp. 57-67 (Austin, Texas, March 2004).

[14] McGowen, R., Poirier, C., Bostak, C., Ignowski, J., Millican, M., Parks, W., and Naffziger, S. Temperature Control on a 90-nm Itanium Family Processor. IEEE Journal of Solid State Circuits, Vol. 41, No. 1, January 2006.

[15] National Instruments Data Acquisition Hardware. http://www.ni.com/dataacquisition/. April 2008.

[16] Processor Power Management in Windows Vista and Windows Server 2008. http://www.microsoft.com. November 2007.

[17] Shen, X., Zhong, Y., Ding, C., Locality Phase Prediction. International Conference on Architectural Support for Programming Languages and Operating Systems, October 2004.

[18] Sherwood, T., Sair, S., Calder, B. Phase Tracking and Prediction. International Symposium on Computer Architecture, June 2003.

[19] An Overview of SYSMark 2007 Preview. http://www.bapco.com/techdocs/SYSmark2007Preview_WhitePaper.pdf, May 2008

[20] Windows Multimedia:timeEndPeriod(). http://msdn.microsoft.com/en-us/library/ms713415(VS.85).aspx. November 2008.

[21] Zagacki, P., Ponnala, V. Original 45nm Intels Coret2 Processor Performance. Intel Technology Journal. http://www.intel.com/technology/itj/2008/v12i3/7-paper/ 1-abstract.htm. October 2008.

[22] Diao, Q., Song, J. Prediction of CPU Idle-Busy Activity Pattern. International Symposium on High-Performance Computer Architecture. February 2008.

[23] V. Pallipadi, S. Li, and A. Belay. cpuidle: Do nothing, efficiently... In Proceedings of the Linux Symposium, June 2006.

[24] Charles, J., et al. Evaluation of the Intel® Core™ i7 Turbo Boost feature. International Symposium on Workload Characterization. October 2009.