2010 IEEE/ACM International Conference on Green Computing and Communications & 2010 IEEE/ACM International Conference on Cyber, Physical and Social Computing

PCFS: A Power Credit based Fair Scheduler under DVFS for Multi-core Virtualization Platform

Chengjian Wen
School of Computer Science and Technology
Beihang University
Beijing 100191, P.R. China
wenchengjian@gmail.com

Jun He
School of Computer Science and Technology
Beihang University
Beijing 100191, P.R. China
hejun@les.buaa.edu.cn

Jiong Zhang
School of Computer Science and Technology
Beihang University
Beijing 100191, P.R. China
zhangjiong@buaa.edu.cn

Xiang Long
School of Computer Science and Technology
Beihang University
Beijing 100191, P.R. China
long@buaa.edu.cn

***Abstract*—In the area of system architecture, there are two most significant trends: multi-core and system virtualization technology. Both of them have quite close relationship with energy efficient computing. Industry turns to integrate more cores on a single chip instead of increasing frequency to solve the heat problem. Meanwhile system virtualization could decrease the total power consumption by sharing the same platform among different operating systems. So the necessity of power management on the multi-core virtualization platform has become increasingly evident. However, traditional virtual machine monitor schedulers could not make efficient use of DVFS, and thus could not take it into considering that the guest OSes may run at different frequency. In order to address this problem, this paper designs a power efficient scheduler which uses the load and the power level of the guest OSes as feedback, and implements a prototype based on Xen virtual machine monitor. This scheduler allocates power credit to VCPU of each guest OS, accounts the power consumption of VCPUs at different speed levels and makes scheduler decision by integrating it to the credit oriented to CPU time slice sharing. It also uses utilization of processors as feedback and sets frequency according to the load change trends instead of the simple static relationship between load and frequency policies, so as to decrease the speed steps required by response to burst load change. Experiment results show that the scheduling fairness of guest OS improved when using DVFS as main power saving method, and the power consumption of the whole system can be reduced by 5%-30%. Therefore, this framework for feedback scheduling could make efficient use of varieties of power saving methods and maintain ideal balance between overall system power saving and single core over heat.**

***Keywords*-Multicore, power management, DVFS, fair scheduling, virtual machine monitor, Xen**

I. INTRODUCTION

The necessity of power efficiency management on multi-core virtualization platform has become increasingly evident. Two obvious reasons for this are related to the most significant trends in the area of computing systems–multicore and virtualization. To continue on the trajectory of Moore's Law without aggravating design and verification complexity, processor architects turned to multicore[14] design to improve performance by exploiting thread level parallelism. On the other hand, over the past few years, virtualization technology has regained considerable attention in the design of computer systems. These technologies are developed partly to solve the problem of power consumption and cooling limitations either on a single processor socket or in datacenter environments. How to make further use of multicore virtualization platform for better performance per watt becomes a focus of green computing.

Researchers have proposed several approaches to OS directed control over a computer's energy consumption, including user- and service-centric management schemes. Current work about power management exploits hardware support for dynamic voltage and frequency scaling (DVFS)[10][7]. And a plethora of methods for improving the power behavior of individual subsystems, such as network devices[8], were devised. Rich prior approaches about DVFS were borrowed from work in the domains of adaptive and autonomic systems. There have been many works in the area of power model using performance monitoring events and feedback scheduling using DVFS for both the real-time systems and general purpose systems. Researchers are becoming more interesting in how to explore the ability of DVFS on multicore systems.

However, most current approaches on energy management are developed for standard, legacy OSes and not suitable for virtual machine monitor(VMM). This is not only because the VM is lack of the privilege of accessing the DVFS hardware control registers, but also because there is a need for coordination. Researchers from CERCS research center [3][4] have proposed some approaches such as VirtualPower

978-0-7695-4331-4/10 $26.00 © 2010 IEEE
DOI 10.1109/GreenCom-CPSCom.2010.126

IEEE computer society

to solve the coordinated power management problem in virtualized systems. Jan Stoess et al. [5] also developed a similar distributed power management framework on L4 microkernel system. Furthermore, developers from intel OTC added a new feature Xenpm [1] to Xen hypervisor. However, all these solutions could not coordinate well with current schedulers and thus the schedulers would not schedule Virtual CPUs(vcpus) fairly on processor cores, which run at different frequencies, and hence would make many unnecessary migrations between different processor cores. Meanwhile it will bring "DVFS jitter" frequently as well as unnecessary delay when changing frequency and voltage. Furthermore, the coordinate mechanism usually requires to modify guest operating system so as to make it Virtualization-aware and thus cause some scalability problems. We argue that a better solution is to integrate DVFS policy and the hypervisor scheduler, and to make the scheduler frequency-aware by accounting the frequency of each processor core.

In order to integrate DVFS policy to a hypervisor scheduler, we propose the approach of Power Credit based Fair scheduler(PCFS for short). We argue that one effective method to manage power consumption is to codesign the power management subsystem and scheduling subsystem. Scheduler chooses the frequency for each physical core to satisfy the performance requirements and meanwhile it will save power as more as possible. One benefit of this integration is to search more chance to adjust frequency in time and to respond to the load feedback. The contributions of this paper are as follows: (1) consider the power as a kind of system resource and allocate power credit for each guest OSes which is used to account the power consumption for different frequency at which the core may run. (2) utilize the processor load as feedback input, decide the new frequency according to the historical trends instead of the static relation between load and frequency. (3) propose the feedback scheduling framework which can make better use of variety of power saving measures and keep the balance between the whole power consumption and over heat of single physical core. (4) use micro p-state statistics to analyze power consumption of VM for some evaluation.

The remainder of this paper is structured as follows: The related approaches are discussed in Section 2. In section 3, we present the generic design requirements to energy management and fair scheduling of virtualization environments using DVFS as energy efficiency method. And the detail for our prototypical implementation for hypervisor-based systems is in Section 4. In Section 5 we present experiments and results. And finally we draw a conclusion and outline future work in Section 6.

II. Related work

Although a new point in the VMM scheduler design space, the PCFS model is related to much previous work on both scheduler design and power efficient computing.

There have been a bound of research about the power management in traditional operating system especially the real-time embedded system. Limitations in battery capacities and demands for longer device lifetimes have motivated research for mobile and embedded systems [11] which later spread to server and desktop systems. DVFS [7][10]provides a mechanism enabling a processor to operate across a range of clock frequencies and voltages, allow one to trade off performance vs. power consumption. In linux OS CPU frequency driver[18] has been developed based of which users can custom variety policy about the balance among performance and power save. And within OpenSolaris TESLA [17] has some good design highlights for power management on a powerfull OS.

Besides mechanism used in OS, online performance prediction and energy model have also been heavily researched. Predicting the workload to be executed plays a crucial role in all on-line DVFS methods[22]. Bellosa et al. [20] developed a CPU power model for thermal management with 8 predictors in a Pentium 4 system.Stoess et al. [5] and Lee et al. [21] developed power models for hypervisor-based virtual machines and for run-time temperature sensing in high-performance processors respectively. Meanwhile recent years many work try to solve the problem how to using advanced control theories to model,design and analyze feedback scheduler[23][24].

While virtualization is popular and becoming the fundamental platform for datacenter and enterprise IT computing environment, researcher on power management pay great interested on how to combine the virtualization and power management technologies. Researches in this area [4][3][6]have mainly focused on global coordination framework and power policy in large scale data center. Developers from Intel OTC improved power management by adding Xenpm[19] subsystem into Xen hypervisor. Unfortunately Polling is a poor thing for the power management subsystem to have to do. There is a trade off that arises around polling more often to improve responsiveness to changes in utilization, vs. polling less often to minimize overhead. Polling itself is inefficient for it need to wake up to check to see if the system is idle.

A relatively new research area on energy efficiency scheduling is to exploit how to schedule in both homogeneous and heterogeneous multicore systems. Multikernel operating system Barrelfish[15] runs a scheduler on each core and manages the power consumption by system knowledge base[16]. HASS[25] proposes a heterogeneity-aware signature-supported scheduling algorithm for traditional OSes while AASH[26] implements a asymmetry-aware scheduler for hypervisor. Although our work in this paper is not to solve the problem of heterogeneity resulted from AMP systems, we treat the system with DVFS as a dynamic heterogeneous one, and we choose the simplest and most effective principle for designing the scheduler of

a heterogeneous system.

III. Design goals for power efficient scheduler

In the virtualization environment, the basic unit of scheduling is vcpu. The role of scheduler is to assign physical processor cores to each vcpu, and maintain dynamic mapping relation between them. Traditional schedulers are usually designed for load-balance and scheduling fairness to achieve the best performance. With the processor capabilities increasing as well as general-purpose processor's DVFS function enabling, the design goal of scheduler changes from the pursuit of only performance to the pursuit of better energy efficiency, and how to effectively and reasonably employ the system energy-saving measurement functionality, especially the DVFS has become a great challenge in the virtualization platform. General processors typically have multiple P-states and C-states, as shown in figure 1 and formula 1, in the same running time, the higher the frequency and voltage, the more energy will be consumed.

$$P \propto fV^2 \tag{1}$$

The proposed PCFS is implemented on open source Xen hypervisor platform, the remainder of this section will introduce the Xen Credit scheduler, and point out that lack of respect in the energy-saving operation, and finally state design requirements for energy-saving scheduler in the virtualization platform.

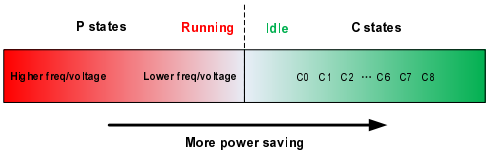

Figure 1: The power saving capability of DVFS

A. Xen credit scheduler

The credit scheduler[2] which is the default scheduler in recent Xen versions is a proportional fair share CPU scheduler built from the ground up to be work conserving on SMP hosts. The credit scheduler regards a time quantum as credit, which is determined by the defined weight and cap for each domain. Each physical core can provide 300 credit in a 30ms accounting period and these credit will be shared by all active VCPUs. Each active VCPU is given the credit calculated by the weight of their domain every credit period and its credit is debited on periodic scheduler tick that occur every 10ms. The credit of a VCPU is used to determine its priority which can be in one of two states: OVER or UNDER(other states include BOOST and IDLE). If a VCPU is in the UNDER state, it has credits remaining. If it is in the OVER state, it has run out of its credit allocation. At each scheduler interrupt, the currently running VCPU is debited 100 credits, and a VCPU can be continuous scheduled at most three tick period (30ms). When making scheduling decisions, VCPUs in the UNDER state are always run before those in the OVER state and the scheduler picks a VCPU at the head of the run queue as a next VCPU. When the time slice of a running VCPU expires, it will be put into the tail of a list that contains VCPUs with the same priority. If a running VCPU does not have any runnable task in spite of time remaining in its time slice, it will be blocked and leave the run queue.

B. Shortage on power management

Credit scheduler is designed to allow the VCPUs to be spanned across multiple packages. Its design is oriented to whole system performance fairness between different virtual machines and load balance between different processors. So the processor cores get little chance to enter a more power-saving state. Besides, an efficient power-aware scheduler needs to know the full state information about the topology of system firstly. However, present Xen Credit scheduler does not support hardware topology very well besides the inter-core topology. So the PCFS can be designed in order that the processor cores can enter more power saving state with large probability.

C. Key design issues for energy efficient scheduler under DVFS

There are three critical requirements for fair energy efficient scheduling on the virtualization platform. Firstly, the scheduler should exploit the variety of energy saving measures supported by the modern processors. The DVFS is the most effective method for processor power management, therefore how to exploit its potential energy saving ability is crucial for the design of whole scheduling framework. The first design issue comes as how to decide the proper frequency according to the load of guest OSes and further more how to integrate the DVFS policy into the scheduler so as to avoid the low efficiency of polling and the semantic gap between subsystem of power management and scheduler. We choose the co-design of power management and scheduler subsystems which means all power management decisions are made in the scheduler and thereby scheduler can trace power states of each core in time.This is shown as figure 2.

Secondly, different load would make some cores run at low frequency while some others run at high frequency, and scheduler may schedule some vcpu always on higher frequency core. Although the time they get are equal but the energy they consume are not equal which is unfair for other VCPUs. We call this phenomenon "energy unfair" which occurs on multicore when using DVFS as power saving method. So a fair scheduler using DVFS must share the cores running at higher frequency fairly among VCPUs of different guest OSes.

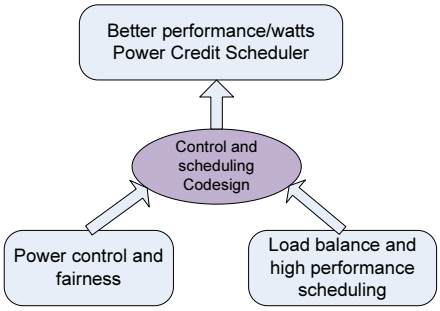

Figure 2: Integration for control and scheduling

Third, the scheduler framework should keep heat balance between each physical core and control the temperature of each core under an threshold. Although the design of each core has been substantially simplified in multicore processors to satisfy power constraints, the total heat energy is increased and processors cooling is still a challenge especially when some cpu-bound applications are run. If the scheduler always keeps one particular core running at the highest frequency, the temperature of that core may beyond the threshold. This brings another design requirement that the scheduler should keep temperature balance and let the different cores keep busy enough in turn. In other words the scheduler should give the different cores fair chance to run at low frequency or go idle.

In conclusion, the design principals are listed here:

- take energy as a kind of system resource, and distinguish statistics of VCPU running time at different frequencies.
- scheduler monitors the load of physical cores and select the frequency according to historical trends not only to the static mapping relations between load and frequency.
- fairly schedule the guest OSes on cores at the higher frequency to meet the fairness requirements of guest OSes.
- fairly schedule frequency levels on the same core to satisfy the physical core cooling limitations.

IV. Design and implementation

In previous section we point out the critical design issues for energy efficiency fair scheduler on virtualization platform. To solve these design challenge We propose PCFS framework as figure 3. The PCFS framework consist of following modules: feedback controller, DVFS selector, topology and power zone management, power credit scheduler. The details will be introduced in the following subsections.

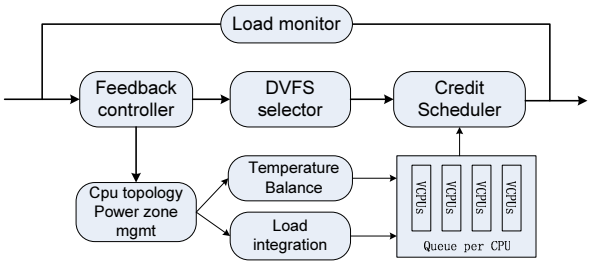

Figure 3: The PCFS framework

A. Unified management of cpu topology and variety of power saving measurements

In the multicore platform, the topology between the physical processor cores affects the performance of the scheduler. Some typical reasons are listed as follows. Different threads scheduled on the cores sharing L2 cache may have a better cache hit ratio, and inter-connection structure of multiple cores in the same socket will decide the communication delay between them. Cores in the same package differ in the cost of vcpu migration with those in different packages. Similarly, an energy effective scheduler more need to fully understand the topology of platform. One of the reasons for that is a number of energy-saving measurements are depended on platform topology. For instance, in order to halt or offline a physical processor package, the scheduler should check whether all the cores in this package have been in idle state.

Linux introduces scheduling domains to manage the processor topology and keep load balanced between a set of cpus. However Xen hypervisor does not support the management of system topology very well and thus could not utilize the topology information effectively to make power saving scheduling decision. We use power zone to manage the system topology related to power management which shares similar principles with scheduling domains. Power zone is a set of processor cores which run with the same frequency and their workloads would be kept balanced by the scheduler of hypervisor. Besides, power zone has reference to other topology including ACPI ID, physical package id, sibling core id. Due to the platform limitations, our current implementation does not consider cost of migration among different cores and the delay of DVFS among different frequencies.

With power zone the variety of power saving measurements are integrated into one framework. When a DVFS decision is made the content of power zones are changed at the way a core belonging to a certain frequency power zone is migrated to another power zone representing other frequency. A power zone is idle when and only when all cores in the zone are idle and we'll take further power saving measurements such as offline a core or whole physical package using cpu hotplug technology according to the

topology.

An especially interest use case is to keep heat and temperature balanced among cores belonging to different power zone. When a threshold is reached at one core the associated power zone will search one core from low frequency power zone to Substitute the over heated core. This will be introduced in the last section of the evolution.

B. Load monitoring and predictions of guest OSes

New x86 platforms support two MSR(model-specific-register)–IA32_APERF and IA32_MPERF, and define that the ratio of these two MSR is the average load of the particular core in the past period. We read the values of these two MSR and calculate the average load of the current core then decide the proper frequency.

$$ratio = \frac{IA32_{APERF}}{IA32_{MPERF}} * 100\% \tag{2}$$

In the virtualization environment the ratio represents the load of the physical core since last time the MSR is reset. Paravirtualized guest operating systems will tell the hypervisor by hypercall[1] it has no work to do and enters the idle state in Xen system. Thus the scheduler will change the according VCPU state to idle and remove from the runqueue of the physical cores. So the ratio representing the physical core's load indicates the average load of guest operating systems. However Circumstances differ when the guest operating systems is hardware-assisted virtual machine(HVM), we will leave the load predictions support for HVM as our further work.

C. Adaptive feedback scheduling framework

In previous sections the method of collecting scheduling information is illustrated and an adaptive feedback scheduling framework is presented in this section. It integrates power management and scheduling together with the scheduling statistics as feedback input thus it works with the basic features of feedback control system.

As shown in Fig.3, feedback controller is the entry of the entire scheduler. It is in charge of choosing one action from DVFS selector and power zone management. We have an array for each physical core to maintain its historical load record. Currently there are 8 items in the array for each core. PCFS will judge wether one core has the load sample under 20% for 8 times. If it is true, PCFS turns to power zone management and migrates the load of this core to the core running at the highest frequency. Meanwhile this core will be set to a more power saving state including offline. Another case choosing power zone management is temperature alert. Once PCFS is informed that one core is over heated, it will migrate the vcpus in the runqueue of the core to the core running at lower frequency. Other cases PCFS will omit the power zone management and take the action of DVFS selector. we define the up-threshold is 95%, when the load is decreased we choose the frequency which can keep cpu load on 90%[18].

D. Runtime power credit management for fair sharing of energy

To support the fair sharing of both computing time and energy consumption between different guest OSes, we introduce a new kind of credit: power credit.Assume that each core in the platform have n frequency level(P-state), which are denoted by $P_0, P_1, P_2, ..., P_{n-1}$. P_0 is the most high frequency and P_{n-1} is the lowest frequency. The total power credits denoted by $Credit_{total}$ allocated in each 30ms time-slice is computed as follows. m denotes the total number of phycical cores in the system. Frequency of core number of i is denoted by f_i.

$$Credit_{total} = \sum_{i=1}^{m} \frac{f_i}{p_0} \tag{3}$$

If one VCPU was scheduled on core i with frequency P_i the $Credit_p$ for this VCPU is decremented by $Credit_{consumed}$ on each 10 millisecond schedulig clock tick as follows:

$$\lambda = (\frac{P_i}{P_0})^2, \quad i = 1, 2, ..., m. \tag{4}$$

$$Credit_{consumed} = \lambda * 100 \tag{5}$$

$$Credit_{left} = Credit_p - Credit_{consumed} \tag{6}$$

In our current implementation scheduling policy is made by both default credit and power credit. Default credit runs as the default Xen credit scheduler and is used to decide the priority of VCPU, the power credit is referred when the scheduler selects the final physical core which is used to run the VCPU. VCPU with high power credit has the high priority to be scheduled on the high frequency physical core. In our previous assumption all physical cores have complete the same P-states distribution which in other words the processor is homogeneous. We expect that our design principles can be expended to heterogeneous multicore systems which we leave as our future work.

V. Evaluation

In this section we evaluate the PCFS scheduler by comparing it to the default Xen scheduler. The idea behind this evolution is to obtain insights into:(1) how effective PCFS can use DVFS as power saving measurement; (2) wether PCFS utilize the variety of power saving measurements in the context of performance conservation; (3) wether PCFS can improve equal sharing capability of the high frequency core? (4) wether PCFS can schedule high frequency on cores to avoid some core become over heated ?

A. Experimental Platform and Benchmarks

We use an Intel Core i5 system with four hard threads and 4GB of RAM. Each core supports 10 P-state and 4 C-state as shows by figure 4 . The maximum frequency is 2.267Ghz, and the frequency step is 133MHz. The version of Xen hypervisor is 3.4.3 while domain0 and domainU both use Linux 2.6.32-5 kernel. The virtualized guest operating system is Debian 5.0 squeeze. Three kinds of applications are selected as benchmarks workload which are listed in Table I as below.

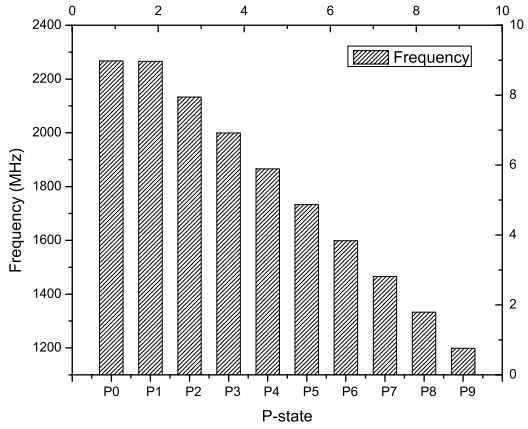

Figure 4: The Intel Core i5 frequency distribution

Table I: Application type

Benchmark	Type	Usage
Spec2006	CPU-bound	nearly 100% workload
Splash2	Parallel application	virtual smp workload
Httperf	Network application	changing workload
Netperf	IO-bound	low workload

B. Evaluation of feedback scheduling ability

In this experiment we evaluate the power-efficient ability of the PCFS scheduling framework and compare it with the default Xen credit scheduler with and without Xenpm which use ondemand governor. In the first set of experiments we ran 4 VMs each configured with one VCPU. One VM ran povray and H264ref from SPEC CPU2006 suite and the other three VMs ran dynamic workload which made cpu utilization changing from 10% to 90%. The lower load was generated by netperf while the dynamic load is generated by httpperf. Figure 5,6 show the results. Figure 5 and figure 6 is overall performance and power results for three kinds of applications. The results shown in figure 6 indicate our scheduling framework can save energy up to 30% while keep the original performance. PCFS can save more power than Xenpm because it gets more chance to keep physical cores run at lower frequency and avoid many unnecessary states transformation. Making effective use of current computing resource is a kind of equal approach for power saving as decreasing the frequency.

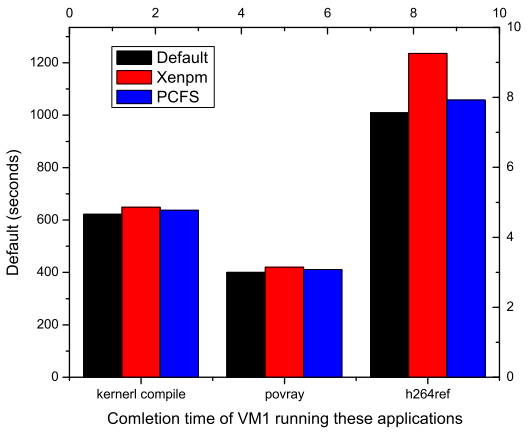

Figure 5: PCFS performance

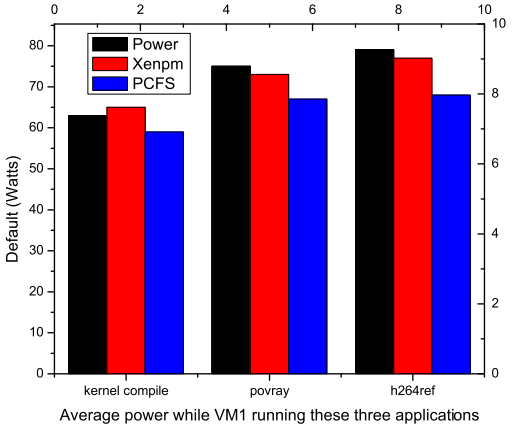

Figure 6: PCFS power consumption

C. Equal sharing capability for cores running at higher frequency

In this section, we collect the scheduling statistics using xenperf tool and xm debug-keys. Firstly we ran three VMs, with one of them has the average CPU utilization of 50%, VM_2 and VM_3 both has the average CPU utilization of 10%. After that, we increased the load of VM_2 in the way of burst. The scheduling statistical data we collected was vpu_wake_up which was based on each core and represented how busy one core was. Results are shown in figure 7.

PCFS-burst and default-burst represent the change when VM_2 has a burst load. Compared to default Xen credit scheduler, PCFS make much more use of high frequency core when the load for VM_2 was increased. That means we make full use of current core running at high frequency instead of change the frequency of the cores running at lower frequency. As explained in previous section it is a equal way to make power efficient scheduling decisions.

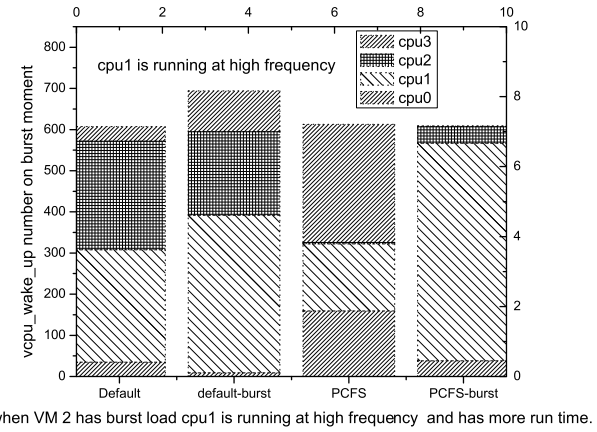

Figure 7: VCPU_wake_up shows the responsiveness of the scheduler

D. Effectiveness of energy-saving measures for low workload

In this experiment we ran four VMs with three other ones running the workload of no more than 10%, we can observe that these three VM were migrated to the same core with two physical core running at C0 state. The physical cores would enter more deep C-state and further more it would be offlined and even unhotpluged when time threshold was reached. Figure 8 presents statistical frequency distribution from which we can observe our scheduling framework gets more chance to decrease the frequency of some cores which had low frequency.

E. Temperature balance between physical cores

In this experiment we run three VMs and each one of them has one vcpu. VM_1 runs mcf from SPEC CPU2006, VM_2 runs netperf which generate average 20% workload, VM_3 runs httperf which generate average 80% load. If the temperature of the CPU exceed 73 degree, an interrupt of temperature above threshold is trigered, and we indirect this interrupt to the scheduler. Then the scheduler will firstly search the low frequency power zoneand migrate the vcpu on that over heat core to one core of the proper power zone. Finally it sets the according frequency in the target power zone.

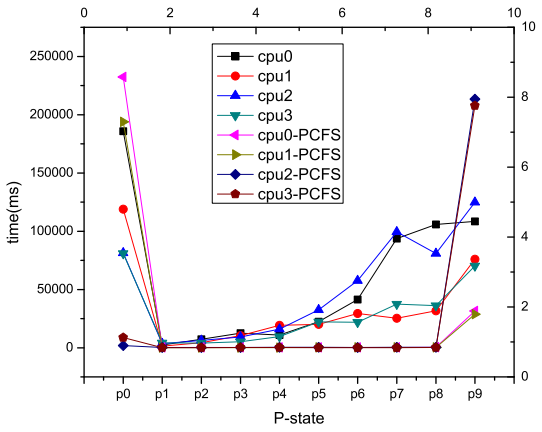

Figure 8: P-states distribution for PCFS and xenpm

VI. Conclusions

PCFS scheduler framework proposed in this paper integrates the power control and scheduling so as to make efficient use of variety of power saving measures especially the DVFS which nearly every modern processor supports. Meanwhile it is aware of different frequency a core may run at when collect the statistical scheduling data which is used to make scheduling decisions. By make effective use of high frequency core PCFS avoids many unnecessary and disorderly frequency change made by power management subsystem usually arising on traditional system. PCFS could keep some cores running at higher frequency and the other at lower frequency as long as possible with the power zone management structure. Experiments show that PCFS can make good use of these power saving measures and support fair sharing of high frequency core between the dynamic changing applications.

Power efficient scheduling on multicore virtualization platform will continue to be one challenging problem. In the future we will evaluate the PCFS on more application type and load configuration and improve the PCFS to adapt to the heterogeneous multicore system and virtual smp guest operating system.

Acknowledgment

This work was supported by the National High Technology Research and Development Program (”863”Program) of China (2007AA01Z118) and Aeronautics Research Funding(20081951033).

References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, “Xen and the art of virtualization”, In *Proceedings of the*

nineteenth ACM symposium on Operating systems principles, Bolton Landing, NY, USA, October 19-22, 2003.

[2] C. Scheduler. *http://wiki.xensource.com/xenwiki/creditscheduler.*

[3] R. Nathuji, K. Schwan, "VirtualPower: coordinated power management in virtualized enterprise systems", In *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, Stevenson, Washington, USA, October 14-17, 2007.

[4] S. Kumar, V. Talwar, P. Ranganathan, R. Nathuji, and K. Schwan. "M-Channels and M-Brokers: Coordinated Management in Virtualized Systems". In *Workshop on Managed Multi-Core Systems*, in conjunction with HPDC'08, 2008.

[5] J. Stoess, C. Lang, and F. Bellosa. "Energy management for hypervisor-based virtual machines". In *Proceedings of the USENIX Annual Technical Conference*, June 2007.

[6] H. Chen, H. Jin, Z. Shao, K. Yu, K. Tian, "ClientVisor: leverage COTS OS functionalities for power management in virtualized desktop environment", In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, Washington, DC, USA, March 11-13, 2009.

[7] C. Isci, G. Contreras, and M. Martonosi. "Live, runtime phase monitoring and prediction on real systems with application to dynamic power management". In *MICRO-39*, December 2006.

[8] R. Kravets and P. Krishnan. "Application-driven power management for mobile communication". In *MOBICOM*, pp. 263-277, October 1998.

[9] J. Kong, I. Ganev, K. Schwan, and P. Widener. "Cameracast: Flexible access to remote video sensors". In *Proceedings of the ACM Multimedia Computing and Networking Conference (MMCN)*, 2007.

[10] P. Pillai, K. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems", In *Proceedings of the eighteenth ACM symposium on Operating systems principles*, Banff, Alberta, Canada, October 21-24, 2001.

[11] A. Miyoshi, C. Lefurgy, E. V. Hensbergen, R. Rajamony, and R. Rajkumar. "Critical power slope: understanding the runtime effects of frequency scaling". In *Proceedings of the16th International Conference on Supercomputing*, pp. 35-44,ACM Press, New York, NY, USA, 2002.

[12] W. Yuan, K. Nahrstedt, "Energy-efficient soft real-time CPU scheduling for mobile multimedia systems", In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, Bolton Landing, NY, USA, October 19-22, 2003.

[13] H. Zeng, C. S. Ellis, A. R. Lebeck, A. Vahdat, "Currentcy: a unifying abstraction for expressing energy management policies", In *Proceedings of the annual conference on USENIX Annual Technical Conference*, pp.4-4, San Antonio, Texas, June 09-14, 2003.

[14] T. G. Mattson, R. Van der Wijngaart, M. Frumkin, "Programming the Intel 80-core network-on-a-chip terascale processor", In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, Austin, Texas, November 15-21, 2008.

[15] A. Baumann , P. Barham , P.E Dagand , T. Harris , R. Isaacs, S. Peter, T. Roscoe, A. Schpbach, A. Singhania, "The multikernel: a new OS architecture for scalable multicore systems", In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, Big Sky, Montana, USA, October 11-14, 2009.

[16] D. Simone. "Power management in a manycore operating system". Master's thesis, ETH Zurich, August 2009.

[17] TESLA. *http://hub.opensolaris.org/bin/view/Project+tesla/CPUPM.*

[18] CPUFREQ. *http://www.kernel.org/pub/linux/utils/kernel/cpufreq/cpufreq.html.*

[19] Xenpm. *http://wiki.xensource.com/xenwiki/xenpm.*

[20] F. Bellosa, A. Weissel, M. Waitz, S. Kellne, "Event driven energy accounting for dynamic thermal management". In *Proceedings of the Workshop on Compilers and Operating Systems for Low Power (COLP03)*, 2003.

[21] K-J Lee, K. Skadron, "Using Performance Counters for Runtime Temperature Sensing in High-Performance Processors". In *Proceedings of the (IPDPS'05) - Workshop 11*, 2005.

[22] D. C. Snowdon, S. M. Petters, G. Heiser, "Accurate online prediction of processor and memory energy usage under voltage scaling", In *Proceedings of the 7th ACM & IEEE international conference on Embedded software*, Salzburg, Austria, September 30-October 03, 2007.

[23] A. Dudani, F. Mueller, Y. Zhu, "Energy-conserving feedback EDF scheduling for embedded systems with real-time constraints", In *Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems*, Berlin, Germany, June 19-21, 2002.

[24] F. Xia, G. Tian, Y. Sun, "Feedback scheduling: an event-driven paradigm", *ACM SIGPLAN Notices*, v.42 n.12, p.7-14, December 2007.

[25] D. Shelepov, J. C. S. Alcaide, S. Jeffery, A. Fedorova, N. Perez, Z. Huang, S. Blagodurov, V. Kumar, "HASS: a scheduler for heterogeneous multicore systems", *ACM SIGOPS Operating Systems Review*, v.43 n.2, April 2009.

[26] V. Kazempour, A. Kamali, A. Fedorova, "AASH: an asymmetry-aware scheduler for hypervisors", In *Proceedings of the 6th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, April 2010.