Design and Implementation of Power Control Programming Interface

Song Liu[1, a], Yi Liu[1, b], Hailong Yang[1, c], Yucong Zhou[1, d]

[1]School of Computer Science and Engineering, Beihang University, Beijing 100191, China;

[a]song.liu@buaa.edu.cn, [b]yi.liu@buaa.edu.cn, [c]hailong.yang@buaa.edu.cn, [d]yucong.zhou@buaa.edu.cn

Keywords: HPC, Power Control, Operating Characteristics, Power Control Programming Interface.

Abstract. High performance cluster need to consume lots of power. High power consumption is an important factor that restricts the development of system in both technical and economic ways. The power control technology has become a hot issue in the area of high performance computing systems and data center management. For different programs, reducing power consumption can exert different influence on performance. In order to minimize the losses of performance, adjustments of power consumption should be made based on the characteristics of programs. This paper aims at solving the problem of power control of nodes in high performance computing system. PCPI (Power Control Programming Interface) is designed to acquire real-time power consumption state and to provide function of dynamic power consumption control, so as to make effective power adjustments according to the operating characteristics of programs. In this way, power consumption can be reduced while the losses of performance are minimized.

1. Introduction

In recent years, with the rapid development of high performance computing and cloud computing technology and with the growing scale of high-performance systems and applied computation amount, the problem of energy consumption has become more and more serious. High power consumption brings lots of problems: it not only reduces the reliability and availability of the system, but also increase the operation costs such like cooling devices and maintenance. As a result, the costs of power and cooling account for more than 70% of the total cost of facilities in the computer room. What's worse, this problem can causes serious environmental pollution and energy waste. Therefore, the way of efficiently manage power consumption becomes increasingly important.

For a long time, low power consumption is a hotspot in the area of computer technology research. The purpose of low power consumption control is to reduce energy consumption of the system as much as possible while bringing zero or little impact on performance. The operating characteristics in the running process are often not fixed, For example, it sometimes suddenly runs computing operations on a large scale, sometimes runs large amount of I/O operations. As the CPU utilization ratio of high performance computing systems is relatively high, in order to achieve the goal of energy saving, reducing power consumption blindly may produce great impact on performance. Therefore, it is necessary to design a scheme that can effectively monitor and control power consumption of the whole system according to the features of programs.

This paper analyzes the task characteristics and power control technology in high-performance environment, design and implement the power control programming interface (PCPI). By calling the PCPI interface, real-time acquisition and control of the system energy consumption can be realized. In the program, use different power control measures according to different situations, or choose different corresponding scheme and come up with more fine-grained optimization strategy according to the power consumption changes of the whole system. Experiments show that in this way, under the premise of small performance losses, the effect of reducing power consumption can be realized.

2. PCPI Function Analysis

In the running process of high performance parallel program, the running program's characteristics change will sometimes cause CPU utilization rate to increases suddenly, resulting in a high peak power. CPU resources can be wasted due to large amounts of I/O operations at some times. Therefore, in the process of programming, we can implement corresponding adjustment mechanism to control power consumption according to the characteristics of different situations.

2.1 Problem Analysis.

Cluster system consists of a number of nodes and some peripheral devices. Node consists of components such like CPU, memory, hard disk and so on. Peripheral equipment includes refrigeration equipment, network equipment, etc.

According to empirical data, it can be known that power consumption of CPU of system power is approximately proportional to CPU's frequency and utilization, namely $P_{cpu} = \beta \times U \times F$, U for the CPU utilization, F for the CPU running frequency while β for a coefficient related CPU. As CPU utilization of program changes, the power consumption of CPU fluctuates, but other devices' power consumption change are far smaller. As a result, we can regard the total power consumption of cluster as consists of two parts: dynamic power consumption and static power consumption. Dynamic power consumption changes with different working conditions of components while static power consumption remains basically the same. CPU power consumption changes as the working frequency and utilization of the processor change. So it can be regarded as the dynamic part of power consumption. Other parts can be viewed as static power consumption. The total power consumption of cluster system can be expressed as $P = P_{dynamic} + P_{static} = \sum_{i=1}^{n}\sum_{j=1}^{m}\beta(i,j)U(i,j)F(i,j) + P_{static}$. Therefore, as long the CPU frequency can be reduced, so can the whole system's power consumption can be reduced.

2.2 Basic Consideration.

Commonly used power adjustment technology mainly start from the perspective of the whole system or devices. The third party software has to be run to realize control. Real-time acquisition and control of system's power consumption cannot be implemented during the running process of program. This method can achieve the purpose of reducing power consumption, but also easily lead to system's performance losses. Power consumption caused by running program is influenced by many factors. As for the process of large amounts of processor calculation, reduction of CPU frequency will cause large loss of performance. For the process of frequent I/O operations, as the bottleneck of performance lies in the I/O devices, and the speed of CPU is much faster than the I/O devices, thus the performance losses brought by reducing CPU frequency is much smaller. If we can adopt different frequency control strategy according to different running characteristics, power consumption can be reduced while minimizing the loss of performance.

Aiming at solving these problems, trying to reduce power consumption while not producing high performance losses, in the process of high-performance programming, we need to consider how to capture and control the real-time practical power, and design the corresponding power consumption control strategy according to the characteristics of the code in the corresponding events. Therefore, we provide a series of programming interface to achieve the functions above.

2.3 Realization.

In order to provide the function of real-time power consumption acquisition so as to make the power consumption optimization easier, we designed the power consumption control programming interface PCPI. The interface implement the adjustment of power consumption of the whole system mainly by changing the frequency of the CPU.

PCPI consists of node end and monitoring end, data communication is realized by using asynchronous Socket.

1. Monitoring End

Monitoring end is deployed in the Windows system, its functions include reading the real-time power consumption of cluster from the power meter, and implementing orders of power queries and frequency adjustments sent by executive node end.

The monitoring end of PCPI supports obtaining power consumption from two kinds of power meter. One is getting the real-time power data through reading power meter connected with serial port of monitoring end's host. Another is getting real-time power data from the PDU device through the network using SNMP protocol. Data source can be switched by changing configuration files. In addition, the monitoring terminal also provides extended function of showing real-time power consumption curves.

2. Node End

Node end is deployed on each node of Linux host. Functions include PCPI interface and Socket monitoring. PCPI interface function provides various functions of collecting and adjusting power consumption. The Socket acoustic monitoring function is responsible for the interactions of node end and monitoring end, including sending orders of collecting real-time power consumption to the monitoring end and sending and receiving requests of power consumption adjustment for cluster as a whole, etc.

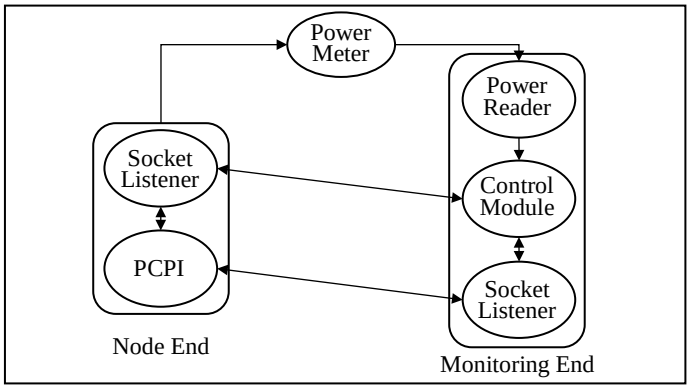

Fig. 1 Interaction Design between Node End and Monitoring End of PCPI

PCPI interface uses cpufreq's kernel subsystem of Linux to control power consumption. The energy-saving principle of cpufreq's subsystem is from DVFS (Dynamic Voltage and Frequency Scaling). It realizes the adjustment of power consumption by changing the frequency of CPU.

Cpufreq offers various control mode, including performance, powersave, ondemand, userspace and so on. Performance is to set the CPU run at maximum performance. Powersave is to set the CPU run at minimum power consumption. Ondemand provides a power adjustment strategy change according to the load status. And userspace provides users with manual mode of frequency control. Userspace control mode is for the users, realizing the adjustment to the processor's frequency in user space. Users can manually set the frequency of processer by modifying the following interface: /sys/devices/system/cpu/cpuX/cpufreq/scaling_setspeed, or dynamically through applications. PCPI supports changing the control modes of CPU, with the default control mode initialized as userspace.

The main interfaces provided by PCPI include:

Table 1: Introduction of Primarily APIs of PCPI

Interface	Function	Introduce
pcpi_initial(char *ip, int port)	Initialize PCPI	Establish the socket connection，set mode as "userspace", port is set default as 10000.
get_total_power()	Get the total power of the cluster	Using socket to connect the moniter part to get real-time power.
get_local_power()	Get the local power	Only available when using PDU device to get power, if not ,return 0.
get_total_energy()	Get the total energy	Can be used before and after the program running, to calculate the total energy consumption.
get_ctrl_mode()	Get control mode	Get the local cpufreq control mode.
set_ctrl_mode(char *mode)	Set control mode	Set the local cpufreq control mode.
get_freq()	Get current frequency	
set_freq(double freq, int local)	Set frequency	If local=0, only set local frequency; otherwise set frequency of all nodes in the cluster.
set_freq_up(int m, int local)	Upgrade m levels of frequency	If local=0, only set local frequency; otherwise set frequency of all nodes in the cluster.
set_freq_down(int m, int local)	Reduce m levels of frequency	If local=0, only set local frequency; otherwise set frequency of all nodes in the cluster.
set_cgroup(int pid, int percent)	Set the cpu usage of the process or thread	
pcpi_close()	Close PCPI	Disconnect socket connection.

While using PCPI to control power consumption, first we should initialize the interface so that we can establish socket connection and set cpufreq control mode as userspace before changing frequency. Then before the cpu intensive events such as large scale CPU calculation, we can call get_total_power() to get power of the whole cluster and analysis it. If the total power is too high, we may call set_freq_down(m, 1) to reduce the frequencies of all nodes by m level, or just call set_freq(x, 1) to set frequencies of all nodes as x. When the program is doing I/O operation we can also reduce the frequency to save power, and set it back after the I/O statement.

At the end of the codes, closing statement is necessary, which is used to close socket connection and recovery the system environment.

3. Experiments

3.1 Experiment Environment

The environment of the experiment includes node ends, power meter and monitoring end. The monitoring end is using to read real-time power and total energy from the power meter. We use GDW1200C electric parameter measuring instrument, which is a professional power meter to get real-time power and energy of the system under test.

The system of monitoring end is Windows 7 while node end is CentOS 7 (kernel：Linux 3.10.0-327.el7.x86_64，CPU：Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz，two CPUs，24 cores，128G Memory)

The test programs of the experiment are bzip2 and mcf in SPEC CPU 2006 and iozone. Among them, bzip2 is CPU bound program, mcf is memory bound and iozone is I/O bound.

3.2 Experiment Method

The experiments adopt two running situation of the same programs, and compare the real-time power diagram with each other.

1. No power control strategy;
2. Calling PCPI in the codes and try to do fine-grained power optimizing.

3.3 Results and Analysis

1. No control strategy is used in this situation, and cpufreq control mode is set as performance. We can see the waved power curve and the higher power peak.

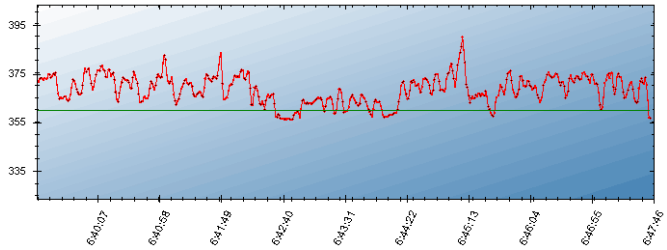

Fig. 2 Real-time Power without Control

2. PCPI is used in this situation while power is too high (while a reference power consumption is set) or doing large scale I/O operation. Comparing the two power curve, we can recognize the effect of PCPI.

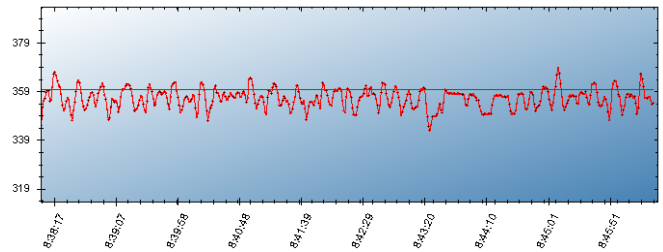

Fig. 3 Real-time Power Controlled by PCPI

4. Conclusion

This paper focus on the problem of large power consumption and high power peak exists in high performance computing nodes. We design a programming interface library PCPI that can realize data acquisition and control, so as to make it more convenient for developers to obtain real-time energy consumption of program and customize a kind of effective power consumption optimization scheme based on corresponding operational characteristics. Experiments show that PCPI interface can reduce the power consumption of program during the running process.

Reference

[1] Sarood O, Langer A, Kalé L, et al. Optimizing power allocation to CPU and memory subsystems in overprovisioned HPC systems[C]//Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013: 1-8.

[2] Marathe A, Bailey P E, Lowenthal D K, et al. A run-time system for power-constrained HPC applications[C]//High Performance Computing. Springer International Publishing, 2015: 394-408.

[3] Van H N, Tran F D, Menaud J M. Performance and power management for cloud infrastructures[C]//Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. IEEE, 2010: 329-336.

[4] Sarood O, Langer A, Gupta A, et al. Maximizing throughput of overprovisioned hpc data centers under a strict power budget[C]//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Press, 2014: 807-818.

[5] Ge R, Feng X, Song S, et al. Powerpack: Energy profiling and analysis of high-performance systems and applications[J]. Parallel and Distributed Systems, IEEE Transactions on, 2010, 21(5): 658-671.