软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2011,22(Suppl.(2)):199−207

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于 NInO 模型的大规模计算系统功耗控制*

刘勇鹏[1+], 朱 鸿[2], 卢 凯[1], 迟万庆[1], 刘勇燕[3]

[1](国防科学技术大学 计算机学院,湖南 长沙 410073)

[2](School of Technology, Oxford Brookes University, Oxford OX33 1HX, UK)

[3](中国科学技术部 信息中心,北京 100862)

Power Capping of Large Scale Computing Systems with NInO

LIU Yong-Peng[1+], ZHU Hong[2], LU Kai[1], CHI Wan-Qing[1], LIU Yong-Yan[3]

[1](College of Computer, National University of Defense Technology, Changsha 410073, China)

[2](School of Technology, Oxford Brookes University, Oxford OX33 1HX, UK)

[3](Information Center, Ministry of Science and Technology, Beijing 100862, China)

+ Corresponding author: E-mail: liuyp@nudt.edu.cn

Liu YP, Zhu H, Lu K, Chi WQ, Liu YY. Power capping of large scale computing systems with NInO. *Journal of Software*, 2011,22(Suppl.(2)):199−207. http://www.jos.org.cn/1000-9825/11041.htm

Abstract: Power consumption is a huge challenge for large scale systems, and power capping is an important goal of power management. Power overspending accumulative ratio $\Delta P \times T$ is introduced as the metric to power capping. For large scale systems, NInO model is proposed to control the power consumption of clusters. Two example algorithms, NInO-P and NInO-ΔP, are designed based on NInO. Finally, the power capping with NInO is proved in experiments.

Key words: large-scale system; power capping; metric; power overspending; control algorithm

摘 要: 随着规模和计算密度的不断提升,大规模计算系统面临严峻的功耗危机,控制最大功耗是系统功耗管理的一个重要目标.针对大规模计算系统,引入超标功耗持续率$\Delta P \times T$ 作为功耗控制的评价标准,提出面向机群系统的最大功耗控制模型 NInO,并基于 NInO 模型设计了两个功耗控制示范算法 NInO-P 和 NInO-ΔP.最后,对所提出的基于 NInO 模型的各项技术进行综合测评与验证.

关键词: 大规模系统;功耗控制;评价标准;功耗超标;控制算法

为了满足科学应用对高性能计算系统日益增长的性能需求,大规模计算系统的规模越来越庞大,计算密度越来越高.大规模和高密度导致高性能计算系统的功耗呈摩尔定律[1]趋势飞速增长,造成了大规模计算系统的功耗危机.

高性能计算系统的功耗惊人,最新发布的 Top500 中有 29 个系统的功耗超过 100 万瓦(MW),排名前十位的系统功耗平均值高达 4.28MW,其中 Fujitsu 的 K Computer 的功耗最高,高达 9.898MW[2].Earth Simulator 的峰值

* 基金项目: 国家自然科学基金(60903044, 60603061, 60903059); 国家高技术研究发展计划(863)(2009AA01A128); 高效能服务器和存储技术国家重点实验室开放基金(2009HSSA04)

收稿时间: 2011-07-15; 定稿时间: 2011-12-02

功耗甚至高达 12MW,与一个中等城市的耗电量相当[3].2006 年,美国数据中心的耗电总量高达 610 亿度,占全美国电力消耗的 1.5%,电费高达 45 亿美元[4],而且还在快速增长.由于无法满足供电的需求,美国一些数据中心的建设被迫延期或取消.惊人的功耗还造成环境的巨大破坏.根据美国环保署(EPA)的统计,每 1 000 度的能耗将产生 0.72 吨的 CO2[5],以此推算,仅 Top500 中前 10 台系统每年运行即相当于排放 26.9 万吨 CO_2.

面对如此严峻的功耗问题,系统功耗控制已经引起高性能计算领域的广泛重视[6],Top500 中超过一半的系统在发布性能指标的同时,还发布了系统的最大功耗[2],以能效为标准对高性能系统进行排名的 Green500[7]受到业界的广泛关注.为了满足系统供电限制或功耗预算约束,控制最大功耗已成为系统功耗管理的一个重要目标.

1 相关工作

评价标准(metric)是量化功耗管理效果、评价功耗管理技术的重要依据,是功耗管理中进行决策的基础.针对不同系统环境,有很多功耗相关的评价标准[6].比如,能耗与性能相结合的能效标准 $E\times D^n$[8]、高性能计算领域备受关注的 Green500 采用的 FLOPS/W[7]、反映系统成本的 TCO[1]、反映数据中心能效的 PUE[9],等等.但是,已有的评价标准主要集中在能效方面,无法充分反映最大功耗对系统的影响.

系统最大功耗控制,又称功耗封顶(power capping),即控制系统的实际功耗不超过预设的功耗阈值.通过控制最大功耗不仅可以确保系统功耗处于安全阈值之内,而且可以在系统功耗预算内运行尽可能多的结点[10].一般来说,最大功耗控制技术包括功耗检测和功耗遏制两个阶段.Ranganathan 等人[11]设计的机群级功耗控制器在每个服务器上运行一个管理代理,代理负责本地功耗采集,全局控制器根据代理搜集的信息确定系统功耗是否超出预算,如果超出,则根据 SLA 选择目标服务器调节其功耗状态.Femal 等人[12]将功耗预算进行两级分配,机群级功耗管理按系统总功耗预算为每个结点分配合适的功耗预算,结点级功耗管理则按本结点的功耗预算在各个设备间进行再分配.这些功耗控制技术每次只选择 1 个结点作为调节目标.在机群系统中,通常由一组结点协助来共同完成某个应用,这些结点的状态应该统一调节.为此,Wang 等人[13]提出了多输入多输出(mulit input multi output,简称 MIMO)功耗管理算法,统一调节执行某一应用的多个结点的功耗状态.本文与 MIMO 一样,每次功耗调节作用于一个结点集合,与 MIMO 具有相同的输入结点集合和输出结点集合不同,本文根据大规模计算系统功耗控制的需求,对机群结点进行分类,输出集合是输入集合的子集,比 MIMO 更一般化,在保证功耗控制模型有效性的同时,降低了系统复杂性.

2 超标功耗持续率$\Delta P\times T$

为了满足系统供电或功耗预算的限制,控制最大功耗已成为系统功耗管理的一个重要目标.系统运行过程中,实时功耗随运行行为的变化而动态变化.系统功耗可以分为 4 类:空闲时功耗 P_{idle}、平均功耗 $P_{average}$、实测峰值功耗 P_{peak} 和理论峰值功耗 P_{theory}.空闲功耗是指系统没有运行用户应用,处于空闲状态的功耗;平均功耗是指一段时间内,系统能耗与运行时间的比值;实测峰值功耗是运行中系统实际达到最大功耗;理论峰值功耗是系统中所有设备标称最大功耗的代数和.在真实系统中,由于各设备之间的负载不可能完全均衡,所有设备同时处于最大功耗状态的概率基本为 0,所以系统的实测峰值功耗小于理论峰值功耗.系统 4 类功耗满足公式(1)的约束.

$$P_{idle}\leqslant P_{average}\leqslant P_{peak}\leqslant P_{theory} \tag{1}$$

对大规模系统而言,实际应用的并行能力受限,而且系统运行多种不同的应用,所有结点同时处于最大功耗状态的概率极低[10].为了合理控制供电系统的建设和运行投入,供电系统支持的最大功耗可能小于所有结点所有设备的最大功耗的代数和.为确保系统的用电安全,必须对系统的实际最大功耗进行监控,将其控制在合理范围之内.另外,记账系统也会对功耗预算有一定的限制,需要控制系统的最大功耗.

最大功耗控制系统通常设定功耗的安全阈值 P_{th},当功耗超过该安全阈值时,就需要采取相应的措施,将系统功耗降低到安全范围之内.

功耗超出安全阈值造成的危害,不仅与超出阈值的相对功耗大小相关,而且与超出安全阈值的持续时间相关.为此,本文提出超标功耗持续率$\Delta P\times T$来评估最大功耗控制系统对功耗的实际控制效果.$\Delta P\times T$的定义表达式如公式(2):

$$\Delta P\times T=\frac{\int_{p>P_{th}}(P-P_{th})\mathrm{d}t}{\int P\mathrm{d}t} \tag{2}$$

其中,P为实时功耗,P_{th}为功耗安全阈值,t为时间.超标功耗持续率$\Delta P\times T$反映的是功耗超出安全阈值在时间上的累计效果与系统功耗的时间累计效果的比值,如图 1 所示,$\Delta P\times T$即是图中深色阴影面积与总面积的比值.$\Delta P\times T$从功耗与时间两个维度对功耗控制对系统用电安全的影响进行度量,超标功耗持续率越小,表示功耗控制效果越好.

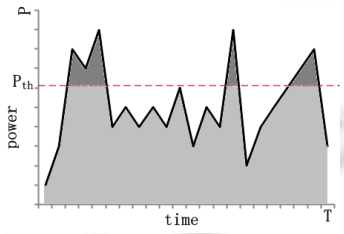

Fig.1 Power overspending accumulative ratio

图 1 超标功耗持续率

$\Delta P\times T$反映了功耗控制对系统用电安全的实施效果.在实际系统中,功耗控制还会对系统运行性能产生影响.因此,评估最大功耗控制系统还应该与能效相结合.

3 面向大规模系统的功耗控制

对大规模计算系统而言,最大功耗控制可以从单个设备的最大功耗、结点最大功耗以及全系统最大功耗这 3 个层次进行考虑.为确保计算结点的安全,稳定运行,结点的供电系统必须能够满足设备和结点的最大功耗需求.在高性能计算中心的实际运行环境中,所有结点同时处于最高功耗状态的机率接近于 0.为了合理控制建设费用和运行维护开销,计算中心设计支持的系统最大功耗或者实际提供的功耗预算可能小于所有结点最大功耗的代数和.也就是说,假设系统一共有N个结点,结点i的最大功耗为P_i,供电系统支持的系统最大功耗为$P_{\max}$,则实际的功耗关系如公式(3)所示:

$$P_{\max}\leqslant\sum_{i=0}^{N-1}P_i \tag{3}$$

因此,功耗管理系统必须对系统的实际最大功耗进行合理的控制,确保系统的实际功耗不超过供电系统支持的最大功耗预算.

3.1 NInO模型

本文根据大规模系统的最大功耗控制需求,提出 NInO(N input n output)功耗控制模型,基于两级阈值控制和多输入多输出控制理论,通过结点性能的动态调节,实现系统功耗的动态控制,如图 2 所示.在实际系统中,功耗供给预算必须满足绝大部分时间的系统正常运行需求,最大功耗控制是为了应对系统在某一时间段的突发式功耗过大,通常只需降低部分结点的实时功耗即可将系统功耗控制在预算范围之内.为方便阐述 NInO 模型,本文首先定义 3 个结点集合:

(1) 统计源集合 A_{total}:对系统功耗产生贡献的所有结点构成的集合,是功耗控制系统感知的最大集合,也即 NInO 模型的输入(input)集合.

(2) 候选集合 $A_{\mathrm{candidate}}$:可以通过相应的功耗控制机制降低功耗的结点集合.在实际系统中,功耗控制机制并

非对所有结点都有效.比如,处理器动态调频机制(DVS)对所有处理器频率都已经处于最低功耗状态的结点无效;休眠机制对已处于最深休眠状态的结点无效;由于 QoS 策略的约束,某些结点不适合进行功耗控制.因此,功耗控制必须确定可以通过相应的功耗控制手段进行功耗控制的结点集合.

(3) 目标集合 A_{target}:被实施相应的功耗控制机制的结点构成的集合,是 NInO 模型的输出(output)集合.

根据上述 3 个集合的定义,可以得出:

$$A_{target} \subseteq A_{candidate} \subseteq A_{total} \tag{4}$$

如图 2 所示的 NInO 模型将整个计算阵列作为 A_{total},功耗测量仪测量 A_{total} 的实时总功耗.空闲结点通常在结点级功耗管理模块的控制下处于最低功耗状态,功耗采样模块只将已分配作业的结点作为候选目标 $A_{\text{candidate}}$.只采样已分配结点在不损失模型正确性的前提下,降低了模型的实现复杂性.NInO 模型中在功耗控制器的控制下选择被调节的结点集合 A_{target}.由此可见,NInO 模型的输出集合通常小于输入集合(这也是 NInO 中第 2 个 n 采用小写字母格式的原因).

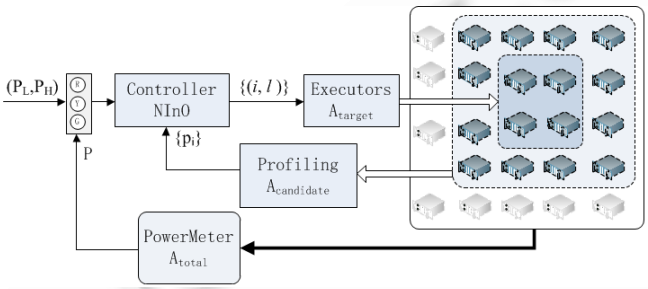

Fig.2 NInO power capping framework

图 2 NInO 功耗控制模型

在图 2 中,P_L 和 P_H 是系统功耗的两级控制阈值,二者满足 $P_L<P_H<P_{\max}$ 约束.系统功耗低于 P_L,表示处于绿色安全状态;超过 P_L 但低于 P_H,表示处于黄色警告状态,功耗控制系统应该采取措施,降低系统的实际功耗;系统功耗大于 P_H,表示系统功耗已进入红色危险状态,必须采取紧急措施,降低系统功耗.

功耗采样模块负责搜集系统中所有候选结点的当前功耗.为此,在每个候选结点运行结点功耗采样代理,负责采样结点的当前功耗.在大规模系统中,一般有功耗测量仪对整个系统的实时功耗进行统计、监控,但是不可能为每个结点配备一个功耗测量仪.因此,结点功耗只能基于结点运行行为特征的实时采样,通过结点功耗模型来推导.在 NInO 模型中,系统功耗通过功耗测量仪精确测量,结点功耗用于结点之间的相对功耗对比,为选择合适的目标结点提供依据(见算法 1 和算法 2 这两种目标结点选择算法),并不需要结点功耗非常精确,因此,本文采用公式(5)所示的结点功耗估计模型来间接获得结点功耗.

$$P = \sum_{all\_cpu} Uti_{cpu} \times P_{cpu} + \frac{Mem_{used}}{Mem_{total}} \times P_{mem} + \frac{Data_{NIC}}{\tau \times BW_{NIC}} \times P_{NIC} \tag{5}$$

其中,P 为结点功耗,Uti_{cpu} 是操作系统统计的 CPU 利用率,P_{cpu} 为处理器的标称最大功耗,所有处理器取和表示结点内所有处理器的功耗,Mem_{used} 是操作系统统计的已分配内存的容量,Mem_{total} 是结点内存总容量,P_{mem} 是所有内存的标称最大功耗,$Data_{\text{NIC}}$ 是采用周期内网卡传输的总数据,通过网卡内的相关计数器获得,BW_{NIC} 是网卡的带宽,τ为采样周期,P_{NIC} 为网卡的标称最大功耗.

性能调节模块根据系统功耗控制器的指令设置目标结点的性能状态.为此,在每个结点运行一个性能调节代理,负责接收系统功耗控制器的指令,并调节结点内各个设备的性能状态.

系统功耗控制器根据当前的系统测量功耗以及 P_L 和 P_H 设置,确定需要进行状态调节的目标结点集合以及目标结点需要调节到的目标状态.为了降低最大功耗控制对应用性能的影响,本文以作业为基本单位,通过调节承载某些作业的结点功耗状态来降低系统功耗.以作业分配的结点集合为单位,而不以结点为单位,是因为并行

作业的性能由所有结点的综合性能决定,一个结点成为性能瓶颈将会降低整个作业的性能,以结点还是以作业所有结点为单位进行调节对并行应用的性能产生的影响差不多,而以作业所有结点为单位进行功耗调节,可以更快地将系统功耗降到安全范围之内.

系统运行过程中,由于大量结点的功耗同时处于较高功耗状态而导致系统实时功耗可能大于系统功耗预算.对于大规模计算系统而言,随着应用行为的动态变化,这种突发式用电高峰并不会持续太久.为了减小功耗控制对应用性能的影响,在用电高峰过后,应将之前由于功耗控制而性能降级的结点恢复到正常性能状态.为此,本文引入系统绿色安全运行持续时间 $Time_g$,用以表示系统进入绿色安全状态($P<P_L$)后运行的时间.相应地,本文进一步引入绿色安全稳定态阈值 T_g,即 $Time_g$ 超过 T_g 后,NInO 认为系统进入稳定的绿色安全态,应该恢复被降级结点的性能.T_g 的引入确保结点性能恢复在系统功耗绿色安全状态持续一段后进行,避免了由于瞬时绿色安全导致系统结点状态的频繁切换.

3.2 基于NInO的功耗控制算法

针对 NInO 模型,本文设计了基于作业的 NInO-P 功耗控制算法,由功耗控制器定时地调用,用于确定目标结点集合及对应目标结点的目标性能状态.

算法 1. Job with maximum power first (NInO-*P*).

输入:整个系统的功耗测量值 P;

所有已分配结点 i 的当前采样功耗$\{p_i(t)\}$.

输出:目标结点集合及其对应的性能等级$\{\{i,l\}\}$.

// A_{degraded} 为已被降级的结点集合

if ($P<P_L$) {

  $Time_g$++;

  if (($Time_g \geqslant T_g$) && ($A_{\text{degraded}} \neq \varnothing$)) { //用电高峰已过,被降级结点恢复正常运行

    目标结点集合为 A_{degraded};

    结点的目标状态为性能最高级;

    $A_{\text{degraded}}=\varnothing$;

  } else

    return ∅;

}

if (($P \geqslant P_L$) && ($P<P_H$)) {

  $Time_g$=0;

  for (结点不在最低功耗状态的所有作业)

    $P_j(t)=\sum_{node_i \in Job_j} P_i(t)$; //计算作业各自功耗

  对上述非最低功耗状态作业 j 按功耗 $P_j(t)$从大到小排序;

  将排序最靠前作业的结点加入目标结点集合; //确定目标结合

  结点的目标状态是性能降一级;

  目标结点加入 A_{degraded};

}

if (($P \geqslant P_H$)) {

  $Time_g$=0;

  目标结点集合是系统所有结点;

  结点目标状态是结点的最低性能态;

  目标结点加入 A_{degraded};

}

由算法 1 的描述可知:

(1) 当系统功耗处于绿色安全范围且持续时间超过绿色安全稳定阈值 T_g 时,功耗控制器恢复之前被降级的结点到正常性能状态.

(2) 当系统功耗处于黄色警戒范围时,选择功耗最大的作业作为目标结点集合,将作业的所有结点性能降一级.目标结点控制在一个作业范围内,并且一次只降 1 级,尽可能减少功耗控制对系统性能的影响.

(3) 当系统功耗处于红色危险范围时,将系统中所有的结点都降低为最低性能态,以确保系统的安全,稳定运行.

黄色警戒状态时,选取功耗最大的作业作为调节目标,功耗降低效果明显,可以尽快地将系统功耗回归到安全范围之内.但是,功耗最大的作业一般是规模最大,结点数目最多的作业.所以,NInO-P 算法容易将每次控制的目标都集中于某个大作业,对大作业来说,很不公平.为此,我们进一步提出以功耗增长率最大的作业为目标的功耗控制算法 NInO-ΔP,见算法 2.

算法 2. Job with maximum power increase ratio first (NInO-ΔP).

输入:整个系统的功耗测量值 P;

所有已分配结点的当前采样功耗$\{p_i(t)\}$.

输出:目标结点集合及其对应的性能等级$\{\{i,l\}\}$.

// A_{degraded} 为已被降级的结点集合

$P_j(t)=\sum_{node_i\in Job_j}P_i(t)$; //计算所有作业的各自功耗

if ($P<P_L$) {

$Time_g$++;

if (($Time_g\geqslant T_g$) && ($A_{\text{degraded}}\neq\varnothing$)){ //用电高峰已过,被降级结点恢复正常运行

目标结点集合为 A_{degraded};

结点的目标状态为性能最高级;

$A_{\text{degraded}}=\varnothing$;

} else

目标结点集合为∅;

}

if (($P\geqslant P_L$) && ($P<P_H$)) {

$Time_g$=0;

for (结点不在最低功耗状态的所有作业)

$\Delta P_j=\dfrac{P_j(t)-P_j(t-1)}{P_j(t-1)}$ //计算作业功耗的增长率

将所有非最低功耗状态作业 j 按功耗增长率ΔP_j从大到小排序;

将排在最前的作业的结点加入目标结点集合; //确定目标集合

结点的目标状态是性能降一级;

目标结点加入 A_{degraded};

}

if (($P\geqslant P_H$)) {

$Time_g$=0;

目标结点集合是系统所有结点;

结点目标状态是结点的最低性能态;

目标结点加入 A_{degraded};

}

记录 $P_j(t)$; //作为下个周期的 $P_j(t-1)$

与 NInO-P 算法的功耗最大作业优先的目标结点选择算法不同,NInO-ΔP 算法选择功耗增长率最大的作业进行性能调节,避免了目标结点过于集中而损害对作业的公平性,更容易找出导致系统功耗超标的"元凶",但是,系统功耗回归到安全范围的收敛速度通常比 NInO-P 算法更慢.基于 NInO 模型,目标结点集合 A_{target} 的选择还可以与作业优先级,服务质量保证等因素相结合,设计不同的目标结点选择算法.

本文设计的 NInO-P 和 NInO-ΔP 算法在系统功耗处于黄色警戒时,采用负反馈方式,进行递减式功耗控制,在实际运行过程中,可能需要多个调解周期才能将系统功耗降低到安全范围之内.通过双阈值设置以及红色危险时的最大力度调节,可以确保系统的最大功耗安全.

在黄色警戒状态,NInO-P 和 NInO-ΔP 算法每次调节都降低一级性能,并没有根据实际功耗超过安全阈值的额度动态调整每次调节的力度.基于 NInO 模型,还可以根据超出阈值的比例,动态调整每次调节的目标结点范围和目标结点状态调整的幅度,比如,超出阈值越多,目标结点集合涵盖的作业也越多,结点性能状态降低幅度也越大.这样一来,可以更快地将系统功耗降低到安全范围之内,但是需引入不同性能状态下的结点功耗预测,将极大地增加算法的复杂性.

4 综合测评

本节对基于 NInO 模型的系统功耗控制技术进行综合测评.测评实验环境中,计算结点由天河-1A 计算主板构成,每个计算结点包含两个 Intel Xeon X5670 6 核处理器,每个处理器配置 6 条 4GB DDR3-1333 内存,构成 2 路 SMP 系统.处理器最大工作频率 2.93GHz,最低工作频率 1.6GHz,共支持 11 个等差的频率等级,实验中通过调节处理器频率来调节结点性能状态.结点间通过主板内置的高速互连接口芯片经由天河-1A 高速网络互连.

本文提出的功耗控制算法基于双功耗阈值 P_L 和 P_H 进行系统级功耗控制,算法的实际运行轨迹与控制效果与阈值的设置密切相关.在实际的机群计算环境中,系统管理员基于计算中心供电系统的设计能力或功耗预算进行设置.本文通过实验学习系统的峰值功耗 P_{peak},并参照 Fan 等人[10]对机群系统用电情况的相关统计,设置 P_L 和 P_H 的设置值.

评测实验环境包括 128 个计算结点,测试程序集合选择 NPB-MPI 测试程序 EP,CG,FT,LU,BT,SP,CLASS 为 D.峰值功耗学习过程中,作业管理系统按算法 3 设计了应用模拟算法,产生并及加载相关的应用,运行 24 小时,运行过程中所有设备工作在最高性能状态,记录运行过程中的峰值功耗,作为本节实验的系统峰值功耗 P_{peak}.

算法 3. NInO 实验作业产生及加载算法.

输入:128 个计算结点构成的空闲结点集合.

IdleNodes=128;

while(1)

{

If (有作业执行完成){

将该作业占用的结点加入空闲结点集合;

IdleNodes 增加该作业释放的结点个数;

}

Job=random{EP,CG,FT,LU,BT,SP}; //应用程序随机

NPROCS=random{8,16,32,64,128,256}; //规模随机

if (*NPROCS*≤*IdleNodes**12) {

在空闲结点集合中按结点号顺序选择$\left\lceil \frac{NPROCS}{12} \right\rceil$个结点运行 Job;

每个处理器核运行一个应用进程;

$$IdleNodes=IdleNodes-\left\lceil \frac{NPROCS}{12} \right\rceil;$$

}

}

Fan 等人[10]对大规模系统用电特征进行长期观察发现,即使运行协调良好的应用,仍有 7%~16%的差额.本文实验环境参照 Fan 等人的统计数据设置功耗控制阈值,即

$$P_H=(1-7\%)\times P_{\text{peak}}=93\%\times P_{\text{peak}},\ P_L=(1-16\%)\times P_{\text{peak}}=84\%\times P_{\text{peak}}.$$

NInO 系统功耗控制模型基于上述 P_H,P_L 设置,T_g 取 10,分别运行 NInO-P 和 NInO-ΔP 各 12 小时,运行过程中,作业管理系统仍按算法 3 产生并加载作业.测试结果如图 3 所示.

Fig.3 Experimental results of the NInO

图 3 NInO 测试结果

图 3 中的性能是用公式(6)计算的系统综合性能,用以描述整个系统的性能损失.

$$Performance=\frac{\sum_{j=1}^{J}\frac{T_j}{T_{nino-j}}}{J} \tag{6}$$

其中,J 为实验阶段完成的作业总数,$T_{\text{nino-}j}$ 为作业 j 在 NInO 算法控制下完成所用的时间,T_j 为该作业在所有处理器保持最高性能状态下完成所需的时间.全性能作业数是指运行过程中承载作业运行的结点未被 NInO 影响的作业的数目.ΔP×T 的计算将 P_L 作为功耗超标阈值 P_{th}.

从图 3 可以看出,通过 NInO 对系统实时功耗的控制,性能损失约 2%,系统的最大功耗则降低了约 10%,功耗超标率的降低则非常明显,NInO-P 和 NInO-ΔP 的超标功耗持续率ΔP×T 相比无功耗控制分别降低了 73%和 66%.在实际系统中,功耗超标的概率很低,因功耗超标而进行结点性能调节是小概率事件,实验中结点状态被调节的作业数占总作业数的约 2%.由于 NInO-P 和 NInO-ΔP 对目标结点集合的选择算法不同,两者的受影响作业数和性能也略有不同,差别分别为 1.4%和 1.1%,造成差别的主要原因是 NInO-P 选择的作业规模一般来说比 NInO-ΔP 更大,一次调节产生的功耗降低也更大,降低到绿色安全态的速率更快.值得注意的是,经过 NInO 的功耗控制,实验过程中系统的最大功耗一直小于 P_H,系统功耗没有进入红色危险状态,进一步验证了 NInO 对确保系统用电安全的重要意义.

5 结束语

本文面向大规模计算系统,提出 NInO 系统功耗控制模型,将机群结点分为统计源、候选、目标这 3 个结点集合,设计了两个基于 NInO 的功耗控制算法,并通过相关实验验证了 NInO 功耗控制模型的有效性.

本文以作业为基本单位,对整个系统的功耗进行控制,没有考虑作业级功耗预算.结合服务质量(QoS)约定、

用户功耗预算、能效等相关因素,设计相应的功耗控制算法,并动态调整调节力度,仍需进一步研究.

References:

[1] Feng W. Making a case for efficient supercomputing. ACM Queue, 2003,1(7):54−64.

[2] Top500. 2011. http://www.top500.org

[3] Feng XZ, Ge R, Cameron KW. Power and energy profiling of scientific applications on distributed systems. In: Proc. of the 19th Int'l Parallel and Distributed Processing Symp. (IPDP 2005). 2005. 34−43.

[4] U.S. Environmental Protection Agency ENERGY STAR program. Report to congress on server and data center energy efficiency, 2007. http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf

[5] U.S. Environmental protection agency. 2009. http://www.epa.gov/cleanenergy/energy-resouces/calculator.html

[6] Liu YP, Zhu H. A survey of the research on power management techniques for high-performance systems. Sotware-Practice and Experience, 2010,13(40):943−964.

[7] Green500. 2011. http://www.green500.org/

[8] Pénzes PI, Martin AJ. Energy-Delay efficiency of VLSI computations. In: Proc. of the 12th ACM Great Lakes Symp. on VLSI (GLSVLSI 2002). New York, 2002. 104−111.

[9] Belady C, Rawson A, Pfleuger J, Cader T. Green grid data center power efficiency metrics: PUE and DciE. White Paper #6, 2007. http://www.thegreengrid.org/gg_content/TGG_Data_Center_Power_Efficiency_Metrics_PUE_and_DCiE.pdf

[10] Fan X, Weber W, Barroso L. Power provisioning for a warehouse-sized computer. In: Proc. of the 34th Int'l Symp. on Computer Architecture (ISCA 2007). San Diego, 2007. 13−23.

[11] Ranganathan P, Leech P, Irwin D, Chase J. Ensemble-level power management for dense blade servers. In: Proc. of the 33rd Int'l Symp. on Computer Architecture (ISCA 2006). Boston, 2006. 66−77.

[12] Femal M, Freech V. Boosting data center performance through non-uniform power allocation. In: Proc. of the 2nd Int'l Conf. on Autonomic Computing (ICAC 2005). Seattle, 2005. 250−261.

[13] Wang X, Chen M. Cluster-Level feedback power control for performance optimization. In: Proc. of the 14th IEEE Int'l Symp. on High-Performance Computer Architecture (HPCA 2008). Salt Lake City, 2008. 101−110.

刘勇鹏(1977－),男,山西忻州人,助理研究员,主要研究领域为功耗管理,容错,高性能计算.

迟万庆(1973－),男,副研究员,主要研究领域为操作系统,功耗管理.

朱鸿(1960－),男,博士,教授,博士生导师,主要研究领域为软件工程,高性能计算.

刘勇燕(1978－),女,博士,高级工程师,主要研究领域为电子商务.

卢凯(1973－),男,博士,教授,博士生导师,主要研究领域为操作系统,高性能计算,虚拟化.