基于状态管理的服务器节能策略研究

肖志娇 明 仲 蔡树彬
(深圳大学计算机与软件学院 深圳 518060)

摘 要 随着云计算的蓬勃发展,计算机行业的能耗问题日益突出。状态管理一直是优化能耗的有效方法之一。对数据中心的服务器进行合理的状态管理能带来可观的节能收益。针对数据中心等机群环境下服务器的状态能耗进行研究,提出基于状态管理的服务器能耗优化方法,以在保证性能的同时,降低了能耗。首先分析状态管理对服务器能耗带来的影响,根据分析提出服务器的状态优化策略,然后利用 Petri 网及其状态分析技术对该策略的状态能耗模型和性能模型进行分析。实例分析和模拟实验验证了该方法的有效性和优越性。

关键词 状态管理,能耗优化,Petri 网,排队论
中图法分类号 TP31 **文献标识码** A

Study on Energy Optimization of Servers Based on States Management

XIAO Zhi-jiao MING Zhong CAI Shu-bin
(College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China)

Abstract With the development of cloud computing, the problem of huge energy consumption has become more and more critical. State management has always been an effective way of saving energy. State management of servers can also bring an impressive amount of energy saving. A strategy based on state management was proposed to optimize energy consumption of servers in data centers, which can guarantee the performance and bring down the energy consumption at the same time. Petri nets and states analysis were used to analyze the strategy. An instance analysis and some experiments were done to show the validity and superiority of the strategy.

Keywords State management, Energy optimization, Petri net, Queuing theory

1 引言

随着能源问题的日益突出,节能减排已成为全社会参与的大事。以往计算机行业的发展很少考虑能源问题,而随着信息化社会范围的逐步扩大,计算机行业的能源问题也突显出来。许多专家学者针对计算机硬件、网络、数据中心等方面的能耗进行了深入的研究,提出了许多节约能耗的有效措施。这些措施主要从两个层面出发,第一类是从硬件、平台、中心的层面出发,通过低耗设备的研究和开发、绿色平台架构的设计和构造、节能中心的规划和搭建,达到节能减排的目的。工业界常用的措施大多属于此列,例如环保材料的选取、节能设备的应用、温控和电力系统的合理分配等。而学术界致力于低功耗设备的研究[1]、设备状态和服务速率的控制[2]等,许多成果在实际中的应用颇有成效。另一类措施是从软件技术的层面出发,从操作系统到应用软件,寻找有效的能耗管理和优化方法。目前对软件节能技术的研究,主要有静态和动态功耗管理与优化技术[3]、多核调度和资源管理优化[4]、虚拟技术[5]等。

状态管理是减少能耗的有效方法之一,绿色计算的第一批成果之一就是电脑显示器的睡眠模式功能[6]。状态管理是指根据负载的动态变化,在不同的情况下,令设备进入不同的节能状态,如休眠、关闭等。在低峰时期,关闭或者休眠一些空闲的设备不会影响系统性能。而设备处于关闭状态时能耗为0,处于休眠状态时能耗也接近于0。林闯等人[7]针对网络节能,建立绿色网络的评价框架,提出将系统资源的状态作为评价基础,用模型描述资源状态及其之间的转移,从而求解各种评价指标。Gupta 等人[8]的研究表明为 LAN 引入休眠模式是节约能耗的有效方法。Anastasi 等人[9]分析了 Wi-Fi 热点网络行为,将网络空闲分为长空闲和短空闲,采用一定的算法对空闲时间长度进行预测,从而决定如何控制网络交换设备的状态。Liao 等人[10]通过优化调度,在满足性能的前提下,尽量将服务请求安排在连续时间内运行,从而使得设备休眠的时间尽可能长,同时减少设备状态切换次数。

据统计,随着数据中心规模、数量的扩大,目前计算机行业的能耗主要是由数据中心的服务器运行而产生的。而状态管理可以有效地降低服务器能耗,因此很多数据中心都选择关闭一些空闲的服务器,以达到节约能耗的目的。关闭服务器意味着减少提供服务的设备,直接影响到服务性能。但目

到稿日期:2012-06-16 返修日期:2012-09-18 本文受国家自然科学基金项目(61170077),广东高校优秀青年创新人才培养计划项目(LYM09121)资助。

肖志娇(1980—),女,博士,讲师,主要研究方向为绿色计算、智能优化、企业信息化等,E-mail:cindyxzj@yahoo.com.cn;明 仲(1967—),男,博士,教授,主要研究方向为软件工程、云计算、物联网等;蔡树彬(1978—),男,博士,讲师,主要研究方向为物联网、推理逻辑等。

前对如何管理服务器状态、在性能和节能之间达到平衡的研究较少。

本文首先讨论数据中心的服务器各类状态及其能耗情况。随后提出一种服务器的状态优化策略。采用 Petri 网技术，建立服务器的状态切换模型。利用这一模型，采用排队论和马尔科夫链等分析技术，建立服务器状态切换的能耗模型和性能模型，研究如何设置和切换服务器的各种状态，从而在保证系统性能的同时，优化服务器能耗。最后，通过实验验证本文策略的有效性和优越性。

2 服务器的状态及其能耗

2.1 休眠状态的引入

服务器在没有响应某个请求而处于就绪状态时，依然有能耗。有数据显示，处于就绪状态的能耗有时甚至达到处于运行状态能耗的 60%～80%。如果处于就绪状态而不提供服务的时间较长，将造成较大的能源浪费。为了降低能耗，当服务器没有响应请求时，可以让其进入休眠状态后，从而带来能耗的下降。引入休眠状态后，服务器的状态及其状态的转换如图 1 所示。

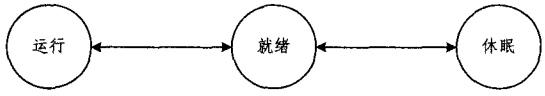

图 1 引入休眠状态后服务器的状态及其切换

休眠状态下的能耗远远低于就绪状态。但进入休眠状态的服务器如果要响应某个请求，就必须首先被唤醒进入到就绪状态，然后再切换到运行状态。这个过程与直接从就绪状态进入到运行状态相比，会带来额外的能耗，以及性能的下降。对于被请求频繁的服务器，状态切换代价太大，不宜为其引入休眠状态。而被请求频率较低的服务器则应引入休眠状态以降低能耗。

引入休眠状态后，所节约的能耗值 P_g 计算如下：

$$P_g=(P_i-P_s)\cdot t_s-P_{i-s}\cdot n_{i-s}-P_{s-i}\cdot n_{s-i} \quad (1)$$

式中，P_i 是服务器处于就绪状态时的能耗，P_s 是服务器处于休眠状态时的能耗，服务器处于休眠状态时的时间是 t_s，P_{i-s} 是服务器从就绪状态切换到休眠状态的能耗，P_{s-i} 是服务器从休眠状态切换到就绪状态的能耗，n_{i-s} 是服务器从就绪状态切换到休眠状态的次数，n_{s-i} 是服务器从休眠状态切换到运行状态的次数。

很显然，只有当式(1)取得一个正值时，引入休眠状态才能带来能耗的降低。一般来说，服务器处于各种状态时的平均能耗大小是固定的，同时每次状态切换所耗费的能耗大小也是一致的。因此，式(1)显示的能耗所能优化的只有两个部分，即服务器休眠与就绪时间的切分和服务器状态切换次数。这两个方面是相互关联和制约的。延长就绪时间可以减少状态切换次数，但延长就绪时间就减少了休眠时间，从而减少由此带来的能耗的降低。

2.2 状态切换的不同情况

将服务器从休眠中唤醒，使其进入就绪状态需要耗费资源。如果服务器醒来后却发现不需要使用它，或者只是短暂的使用，那么这个能耗就无法带来任何收益。因此，希望总是在需要时才唤醒服务器。另一方面，如果针对每个到达的请求都一一唤醒服务器去处理，那么每次请求到达就存在一个状态切换的能耗，频繁的状态转换同样导致得不偿失。合理设置服务器处于休眠和就绪状态的时间段以及服务器的休眠和唤醒的时刻，就可以减少状态转换的次数。

3 服务器状态优化策略

假设当前数据中心一共有 c 台服务器处理某类用户的请求。当用户请求不多时，让部分服务器进入休眠(或关闭)状态，从而达到降低能耗的目的。根据前面的分析，如果当用户请求到达就立即唤醒服务器，容易造成频繁的状态切换，反而达不到节约能耗的目的。为了保证用户的请求能够及时得到响应，而又不频繁地进行状态切换，采用以下状态切换策略对服务器的能耗进行优化。

假设在稳定状态下，让 $r(0\leqslant r\leqslant c)$ 个服务器始终处于就绪或者运行的状态，而不进入休眠状态，用以处理稀疏到达的用户请求。当用户请求的到间隔相对较长时，可以让 $r=0$，充分利用休眠状态带来的能耗下降。而用户请求间隔相对较短时，适当扩大 r 的取值。其他的服务器在不提供服务时，处于休眠状态。根据当前用户请求的数量来决定是否唤醒额外的服务器。在多个服务器加速服务下，用户请求的数量会逐步减少，此时，让这些额外增加的服务器再次进入休眠状态。这样可以在兼顾性能的同时，降低能耗。

在目前的服务器状态管理研究中，一般采用单阈值策略。当排队的请求数量超出阈值时，唤醒处于休眠状态的服务器，为客户提供服务。当排队的请求数量低于阈值时，休眠被唤醒的服务器。显然如果排队的请求数量在阈值左右徘徊，那么就要频繁地将服务器在休眠状态和运行状态间切换。这就是常说的抖动问题。抖动问题的存在很明显会使得服务状态切换次数增多，从而导致能耗的上升。

为解决抖动问题，引入双阈值策略。当排队的请求超出唤醒阈值时，唤醒额外的服务器。一般情况下，服务器的负载应在 80%左右，过高则不利于性能。而服务器负载过低，则虚耗能源。因此，当排队的请求低于休眠阈值时，休眠部分服务器。这一方面有利于解决抖动问题。另一方面，如前文所述，一旦唤醒了处于休眠状态的服务器，就会产生状态切换能耗。为了减少能耗，应当充分利用此次唤醒动作。因此，被唤醒的服务器应当尽可能地处理更多的客户请求。

3.1 服务器状态优化模型

为了采用数学分析技术对服务器状态及其切换进行优化，首先利用 Petri 网技术，建立以上过程的服务器状态及切换模型，如图 2 所示。

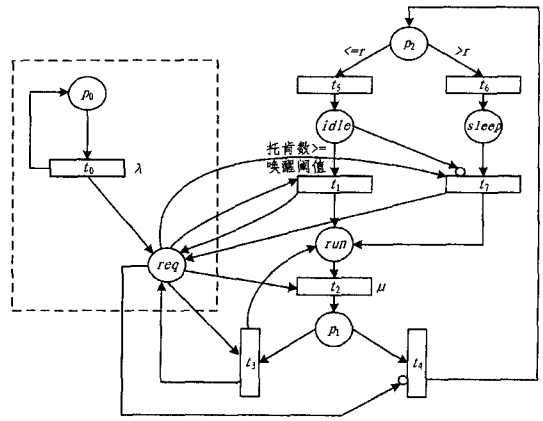

图 2 服务器状态切换模型

其中各库所和变迁的含义如下：

• p_0：为模拟客户请求的到达而引入的辅助库所。

• p_1 和 p_2 为选择库所，根据条件选择将服务器置于何种状态。

• t_0：模拟客户请求的到达满足参数为 λ 的泊松分布。

• t_2：模拟处理客户请求的时间服从参数为 μ 的负指数分布。

• t_1、t_3、t_4、t_5、t_6 和 t_7 为瞬时变迁，辅助进行状态切换。

利用马尔可夫链构造图 2 的状态可达图，如图 3 所示。

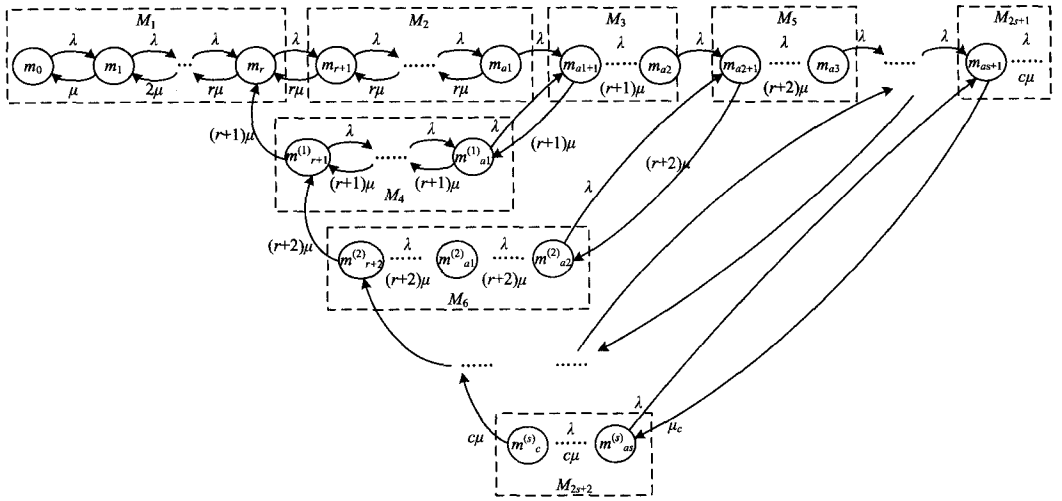

图 3　状态可达图

在图 3 中，处于 M1 状态时，无等待的客户请求。当客户请求数量大于当前可用服务器个数 r 时，进入状态 M2，部分客户请求需排队等待。当排队的请求数量超出阈值 a1，进入状态 M3 时，唤醒 1 个处于休眠状态的服务器，为客户提供服务。当排队的请求数量下降，跌至阈值 a1 以下时，并不是立即让额外唤醒的服务器进入休眠状态，而是继续为排队等待的客户请求提供服务，系统进入状态 M4，直到没有等待的请求，额外唤醒的服务器才再次进入休眠状态。系统设置多个阈值，从而根据客户请求情况，逐一唤醒服务器。利用马尔科夫链在稳定状态下的性质，计算出数据中心的服务器处于各个状态的概率以及进行状态切换的概率。

图 3 中的各个状态下图 2 的托肯分布如表 1 所列。

表 1　各状态下库所的托肯个数

state		req	run	idle	sleep
M_1	m_0	0	0	r	s
	m_1	1	1	r−1	s
	m_r	r	r	0	s
M_2	m_{r+1}	r+1	r	0	s
	m_{a1}	a_1	r	0	s
M_3	m_{a1+1}	a_1+1	r +1	0	s−1
M_4	m'_{r+1}	r+1	r +1	0	s−1
	m'_{a1}	a_1	r +1	0	s−1
	……	……	……	……	……

由于库所 req、run、idle 和 sleep 是区分各个状态的主要库所，而其他库所基本为瞬时库所，起简单分支作用，因此在表 1 中未列出。在表 1 中，$r+s=c$，a_1 和 a_2 等为阈值。当等待的客户超出阈值时，逐一唤醒休眠中的 s 个服务器来提供服务。

3.2　能耗模型

在第 2 节中已提到，引入休眠状态所带来的能耗差异如式(4)所示。一般来说处于各个状态以及进行一次状态切换的能耗是固定的。因此，仅需对式(4)中的 t_s、n_{i-s}、n_{s-i} 几个变量进行计算，即可建立能耗的数学模型。利用马尔可夫链稳态下的性质，可求得如下相关数值：

• 各个服务器处于休眠状态的时间之和 t_s：

$$t_s=\sum_{i=0}^{a_1}(p_i\cdot s)+\sum_{k=1}^{s-1}\sum_{i=a_k+1}^{a_{k+1}}[p_i\cdot(s-k)]+\sum_{k=1}^{s-1}\sum_{i=r+k}^{a_k}[p_i^{(k)}\cdot(s-k)] \tag{2}$$

• 服务从运行状态切换到休眠状态的次数 n_{i-s}：

$$n_{i-s}=\mu\cdot\sum_{k=1}^{s}[(r+k)\cdot p_{r+k}^{(k)}] \tag{3}$$

• 服务从休眠状态切换到运行状态的次数 n_{s-i}：

$$n_{s-i}=\lambda\cdot\sum_{k=1}^{s}p_{a_k} \tag{4}$$

3.3　客户请求性能模型

时间是衡量服务响应客户请求的性能的主要指标之一。本文以客户请求平均等待时间为指标，建立客户请求性能模型。

设某一客户请求到达时，发现系统内已有 j 个客户，若 $j<r$，则该请求等待时间为 0。若 $r\leqslant j<a_1$，则要分两种情况考虑：当前服务器为 r 个和当前服务器为 $r+1$ 个。

• 如果当前服务器为 r 个，那么该客户请求需等待 $(j-r+1)$ 个请求完成才能获得服务。每个客户的平均处理时间为 $1/\mu$，那么 $(j-r+1)$ 个请求平均处理时间为 $(j-r+1)/\mu$。由于当前一共有 r 个服务器，因此当前到达的客户请求的平均等待时间为 $(j-r+1)/r\mu$。

• 如果当前服务器为 $r+1$ 个，那么若 $j<r+1$，则该请求等待时间为 0。否则，当前到达的客户请求的平均等待时间为 $(j-r)/(r+1)\mu$。

其他情况，以此类推，可求得如下客户请求的平均等待时间：

$$\overline{W_q}=\sum_{j=r}^{a_1-1}(p_j\cdot\frac{j-r+1}{r\mu})+\sum_{k=1}^{s-1}\sum_{j=a_k}^{a_{k+1}-1}(p_j\cdot\frac{j-(r+k)+1}{(r+k)\cdot\mu})+\sum_{j=a_s}^{\infty}(p_j\cdot\frac{j-c+1}{c\mu})+\sum_{k=1}^{s}\sum_{j=r+k}^{a_k}(p_j^{(k)}\cdot\frac{j-(r+k)+1}{(r+k)\cdot\mu}) \tag{5}$$

4　实验及分析

4.1　实例分析

本文首先通过分析一个简单实例来演示服务器状态优化策略的有效性和优越性，实例描述如下。

假设对某数据中心的某类请求到达服从泊松分布，平均

单位时间内达到个数为1，请求处理的时间服从负指数分布，平均单位时间内处理个数为2。该数据中心为此类请求提供2台服务器。假设在单位时间内，就绪状态时的能耗 P_i 是5个单位能耗，休眠状态时的能耗 P_s 是1个单位能耗，一次运行状态到休眠状态转换的能耗 P_{i-s} 是2个单位能耗，一次休眠状态到运行状态的能耗 P_{s-i} 也是2个单位能耗。

在此，比较3种不同的策略，分别为不引入休眠状态策略（策略1）、不引入下阈值策略（策略2）和本文策略（策略3）。

• 如果采用策略1，不引入休眠状态，则无法带来能耗的节省，即 $P_g=0$。采用M/M/c排队论模型计算该数据中心的客户请求的平均等待为 $\overline{W_q}=\frac{1}{30}\approx 0.033333$ 单位时间。

• 采用策略2，那么有一个服务器会根据请求情况在休眠和运行状态间切换。利用马尔可夫链稳态性质，计算出如下各状态概率。

$$p_0=\frac{1}{2-\frac{2}{3}\cdot(\frac{1}{2})^{a_1}} \tag{6}$$

$$p_i=(\frac{\lambda}{\mu})^i\cdot p_0,1\leqslant i\leqslant a_1 \tag{7}$$

$$p_i=(\frac{\lambda}{2\mu})^{i-a_1}\cdot(\frac{\lambda}{\mu})^{a_1}p_0,a_1+1\leqslant i \tag{8}$$

那么，客户请求的平均等待时间如下：

$$\overline{W_q}=\sum_{j=1}^{a_1-1}(p_j\cdot\frac{j}{\mu})+\sum_{j=a_1}^{\infty}(p_j\cdot\frac{j-1}{2\mu}) \tag{9}$$

在策略2下，该数据中心服务器的能耗节约值 P_g 如下：

$$P_g=(P_i-P_s)\cdot\sum_{i=0}^{a_1}p_i-P_{i-s}\cdot 2\mu\cdot p_{a_1+1}-P_{s-i}\cdot\lambda\cdot p_{a_1} \tag{10}$$

• 采用策略3，该数据中心保证总有一台服务器处于运行或就绪状态，随时提供服务。而另一台服务器则有时进入休眠状态，从而降低能耗。利用马尔可夫链稳态性质，计算出如下各状态概率。

$$p_0=\frac{1-(\frac{1}{2})^{a_1}}{2-2\cdot(\frac{1}{2})^{a_1}-\frac{a_1}{3}\cdot(\frac{1}{2})^{a_1}} \tag{11}$$

$$p_1=\frac{1}{2}\cdot p_0 \tag{12}$$

$$p_2'=\frac{(\frac{1}{2})^{a_1}}{8\cdot(1-(\frac{1}{2})^{a_1})}\cdot p_0 \tag{13}$$

$$p_{a_1+1}=\frac{4}{3}\cdot[1-(\frac{1}{2})^{2a_1}]\cdot p_2' \tag{14}$$

$$p_i=(\frac{1}{2})^i\cdot p_0-[4-(\frac{1}{2})^{i-3}]\cdot p_2',2\leqslant i\leqslant a_1 \tag{15}$$

$$p_i=(\frac{1}{2})^{2i-2a_1-2}\cdot p_{a_1+1},a_1+1<i \tag{16}$$

$$p_i'=\frac{4}{3}-\frac{4}{3}\cdot(\frac{1}{2})^{2i-2}\cdot p_2',3\leqslant i\leqslant a_1 \tag{17}$$

那么，客户请求的平均等待时间如下：

$$\overline{W_q}=\sum_{j=1}^{a_1-1}(p_j\cdot\frac{j}{\mu})+\sum_{j=a_1}^{\infty}(p_j\cdot\frac{j-1}{2\mu})+\sum_{j=2}^{a_1}(p_j'\cdot\frac{j-1}{2\mu}) \tag{18}$$

采用策略3，该数据中心服务器的能耗节约值 P_g 如下：

$$P_g=(P_i-P_s)\cdot\sum_{i=0}^{a_1}p_i-P_{i-s}\cdot 2\mu\cdot p_2'-P_{s-i}\cdot\lambda\cdot p_{a_1} \tag{19}$$

对以上3种不同的策略，比较它们的性能和能耗，如图4、图5所示。策略1的客户请求平均等待时间为固定值，比策略2和策略3都要小。策略2和策略3则根据阈值 a_1 的不同取值，客户请求平均等待时间逐步变大，直到趋于一个稳定值。从图4可以看出，策略3的平均等待时间始终优于策略2。

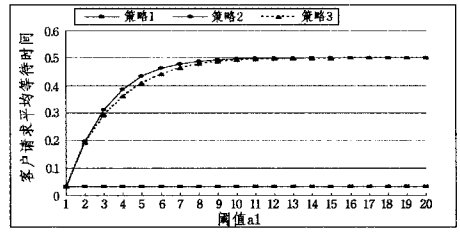

图4 客户请求的平均等待时间比较

由于没有引入休眠状态，策略1的能耗节约为0。策略2和策略3则根据阈值 a_1 的不同取值，能耗节约值逐步变大，直到趋于一个稳定值。由图5可以看出，策略3在能耗节约方面也优于策略2。

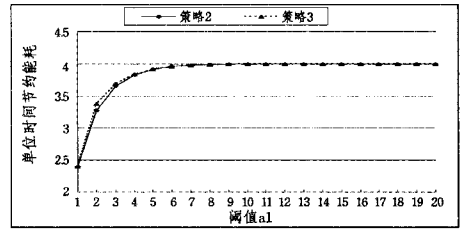

图5 单位时间内节约能耗比较

以上实例的规模很小，通过上面的比较，可以看出本文策略的优势。在大规模数据下，本文策略的优势将更明显。

4.2 实验分析

为了进一步说明本文策略的优越性，采用仿真软件CloudSim，利用历史数据，针对多个不同实例，在实验环境下进行大规模数据的仿真模拟，比较不同策略的性能和能耗等数据。

策略1虽然能最好地保证性能，但是无任何能耗的节省。因此仅比较策略2和策略3。从表2可以看出实验得到和实例分析相同的比较结果，本文策略能在节约更多能耗的同时，获得更优的性能。

表2 部分实验结果比较

实例号	平均等待时间		能耗节约值	
	策略2	策略3	策略2	策略3
1	0.04317637	0.02959146	−2798.283989	472.0522794
2	0.28305495	0.21492324	1633.989326	2250.453611
3	0.30312993	0.21248705	2128.073734	5335.829993
4	0.23066526	0.16292866	1123.434795	3018.821402
5	0.23315242	0.16602028	3058.668973	3221.188964
6	0.15170661	0.10845375	2538.781908	2934.502535
7	0.03473801	0.02401383	4.91021385	1867.061488
8	0.15037049	0.10796982	1119.718566	1144.728877
9	0.15685222	0.10627939	1252.078261	2245.488881
10	0.28021241	0.18156415	1542.284794	3308.003583

注：负数代表无能耗节约，反而产生更多额外的能耗。

结束语 本文在相关研究的基础上，通过对服务器及其服务过程的分析和研究，提出了一种优化服务器状态及其状态切换从而降低能耗的绿色服务器管理策略，即采用Petri网

（下转第30页）

一个给定的应用任务，根据对其应用程序结构的分析，得到它的应用程序超图，然后利用上节中的计算模型，得到对应的体系结构超图，对于体系结构超图的第三层，把每个子派生结构看成是第二层超图的一个结点，这样就和应用程序超图中的子算法对应起来，这可以由子算法的关键粒度决定。

第二种情况：若某种应用的体系结构已建立，就得到了对应的体系结构超图，如果这时应用程序结构发生改变，则需要进行的是超图演化模型，那么需要对体系结构超图进行图变换，利用感知算法和智能决策算法，依据应用程序结构超图中的点或边的变化情况，对应修改体系结构超图中的点（子结构）或边（子结构间的互联关系），使体系结构超图保持与应用程序超图的同构关系。

通过以上方式，我们始终使体系结构超图与应用程序结构超图同构，这样，就能使得不同的应用与其最适合的体系结构匹配，实现可重构的体系结构，以满足不同应用需求。把该计算模型在我们开发的原型样机上进行了实际测试，结果表明，这种可重构计算模型是合理有效的。

结束语 在互联网复杂多样的应用中，单一的体系结构已经不能胜任，对于不同的计算任务，需要构建可重构的体系结构去匹配。我们首先用分层超图刻画应用程序结构和体系结构，然后对不同的应用程序结构超图利用超图的同构原理得到相应匹配的体系结构超图，使得应用程序超图中的子算法簇和关系与体系结构中的子结构和关系达到一一对应，达到整体上的并行计算和最优结构。同时对于子算法簇中的子算法，利用决策算法，在体系结构中使用最适合的子派生结构去匹配，从而达到局部的最优计算效果，这样就能使得不同的应用得到可重构的体系结构匹配，达到了效率最优的结果。

参 考 文 献

[1] 李国杰. 信息科学技术的长期发展趋势和我国的战略取向[J]. 中国科学：信息科学，2010，40(1)：128-138

[2] 邬江兴. 云计算高效能之路[R]. 第三届中国云计算大会报告. 2011

[3] 沈绪榜，等. 计算机体系结构的分类模型[J]. 计算机学报，2005，28(11)：1759-1766

[4] Shoaib K. Reconfigurable hybrid interconnection for static and dynamic scientific applications[C]// Proceedings of the ACM International Conference on Computing Frontiers. 2007：183-194

[5] 陈左宁，金怡濂. 基于多虚空间多重映射技术的并行操作系统[J]. 软件学报，2001，12(10)：1562-1568

[6] 乔磊，齐冀，龚育昌. 一种支持可重构混成系统的操作系统设计与实现[J]. 计算机学报，2009，32(5)：1046-1054

[7] Kiran B, Viktor K. Reconfigurable computing: architectures, models and algorighms[J]. Current Science, Special Section on Computational Science, 2000, 78(7): 828-837

[8] Artundo I, et al. Selective optical broad-cast component for reconfigurable multiprocessor interconnects[J]. IEEE Journal on Selected Topics in Quantum Electronics: Special Issue on Optical Communication, 2006, 12(4): 828-837

[9] Selvakkumaran N. Multiobjective hypergraph partitioning algorithms for cut and maximum subdomain-degree minimization [J]. IEEE Transactions on Computer Aided Design of Intergrated Circuits and System, 2006, 25(3): 504-517

[10] 徐洪珍，曾国荪. 基于超图文法的软件体系结构动态演化[J]. 同济大学学报，2011，39(5)：745-750

[11] 宋锐，等. 基于 RSOM 树和类属超图的分布式图像检索方法[J]. 信号处理，2009，25(8A)：264-267

[12] 王忠杰，等. 基于分层超图的服务价值依赖模型[J]. 计算机集成制造系统，2011，17(8)：1834-1843

[13] Karonski M. The phase transition in a random hypergraph[J]. Journal of Computational and Applied Mathematics, 2002(142): 125-135

(上接第 25 页)

对服务器状态及其切换进行建模，利用马尔可夫链构造服务器状态转换过程，使用排队论等相关数学分析技术建立服务器的能耗模型和性能模型，进而对状态的管理进行合理设置，达到降低能耗的目的。该策略在保证用户的请求能够及时得到响应的同时，使服务器能够获得休眠状态所带来的节能效果，又不会导致频繁地进行状态切换，从而产生额外的能耗。分析和实验验证了本文策略的有效性和优越性。

服务器的能耗和性能随着阈值的变化而变化，如何优化设置甚至动态设置状态转换中的阈值参数，找到使得能耗节省和性能保证的最优阈值平衡点是下一步的研究方向。另外如何优化调度各个服务器，从而进一步优化能耗，也是未来研究的方向。

参 考 文 献

[1] 黄海林，范东睿，许彤，等. 嵌入式处理器中访存部件的低功耗设计研究[J]. 计算机学报，2006，29(5)：815-821

[2] Gunaratne C, Christensen K, Nordman B, et al. Reducing the Energy Consumption of Ethernet with Adaptive Link Rate[J]. IEEE Transctions on Computers, 2008, 57(4): 448-461

[3] 吴琦，熊光泽. 非平稳自相似业务下自适应动态功耗管理[J]. 软件学报，2005，16(8)：1499-1505

[4] Chen Hua-cai, Jin Hai, Shao Zhi-yuan, et al. ClientVisor: Leverage COTS OS Functionalities for Power Management in Virtual Execution Environments[M]. Washington D C. USA: ACM Press, 2009: 62-71

[5] Nathuji R, Schwan K. Virtual Power: Coordinated Power Management in Virtualized Enterprise Systems[C]// Proc. of the 22nd ACM Symposium on Operating Systems Principles. Stevenson, WA, USA: ACM Press, 2007: 265-278

[6] 郭兵，沈艳，邵子立. 绿色计算的重定义与若干探讨[J]. 计算机学报，2009，32(12)：2311-2319

[7] 林闯，田源，姚敏. 绿色网络和绿色评价：节能机制、模型和评价[J]. 计算机学报，2011，34(4)：593-612

[8] Gupta M, Grover S, Singh S. A Feasibility Study for Power Management in LAN Switches[C]// Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP'04). Berlin, Germany, 2004: 361-371

[9] Anastasi G, Conti M, Gregori E, et al. A Performance Study of Power-saving Polices for Wi-Fi Hotspots[J]. Computer Networks, 2004, 45: 295-318

[10] Liao W H, Yen Wen-ming. Power-saving Scheduling with a QoS Guarantee in a Mobile WiMAX System[J]. Journal of Network and Computer Applications, 2009, 32: 1144-1152