文章编号:1009-3087(2015)02-0112-05 DOI:10.15961/j.jsuese.2015.02.017

# 云计算环境下基于能耗感知的弹性资源管理机制

熊 伟[1,3],李 兵[2*]

(1.武汉大学 软件工程国家重点实验室 计算机学院,湖北 武汉 430072;2.武汉大学 国际软件学院,湖北 武汉 430072;
3.湖北文理学院,湖北 襄阳 441000)

**摘 要**:目前云计算环境中弹性资源管理的研究,主要考虑了系统的功能和性能,较少关注到能耗问题。设计了一种结合请求率预测与能耗感知的弹性资源管理方法。该方法首先根据获取的请求变化信息学习基于小波变换的模型以准确预测用户对应用的请求率,然后根据请求率与资源开销信息将任务映射到资源开销相对固定的应用执行单元,以达到对系统能耗的弹性管理。通过实验对不同的弹性策略的性能进行比较,验证了该方法在保证服务质量的前提下能更有效地降低能耗。

**关键词**:云计算;能耗模型;弹性;预测控制

**中图分类号**:TP311.5 **文献标志码**:A

## An Elastic Resource Management Mechanism Based on Perception of Energy Consumption in Cloud Computing Environment

*XIONG Wei*[1,3], *LI Bing*[2*]

(1. State Key Lab. of Software Eng., School of Computer, Wuhan Univ., Wuhan 430072, China;
2. International School of Software, Wuhan Univ., Wuhan 430072, China;
3. Hubei Univ. of Arts and Sci., Xiangyang 441000, China)

**Abstract**: The existing solutions in present cloud environment focused more on the functionality and performance of the system, and less on energy-consuming. An elastic resource management approach based on perception of energy consumption in cloud computing environment was designed. This approach accessed to information of resources cost and request changes firstly, then learned a wavelet-transform-based model to predict request rate, and mapped tasks to application execution units, where resources costs are relatively fixed, in order to achieve flexible management of system energy consumption according to request rate and cost information. The results showed that this approach achieves higher performance than different ones, which can more effectively reduce energy consumption on the premise of guaranteeing the quality of service.

**Key words**: cloud computing; energy-consuming model; elasticity; predictive control

对于云计算服务的提供者来说,他们的主要任务是保证针对用户的服务质量,并尽可能地减少系统资源开销。采取弹性的资源管理机制,可以在保证服务质量规约(service level agreement)的前提下,有效地提高系统的资源利用率,从而减少能耗和运维成本。

根据美国环保署发布的一项数据表明,当前美国数据中心所消耗的电力已达到消耗总电力的2%[1]。而根据对典型数据中心耗能分析,实际计算设备占总耗能的52%,而其他辅助系统占总耗能

**收稿日期**:2014-08-10

**基金项目**:国家重点基础研究发展计划资助项目(2014CB340400);国家自然科学基金资助项目(61273216;61202032;61202031;61202048);湖北省科技创新项目资助(2013AAA020);国家科技支撑计划资助项目(2012BAH07B01);武汉市青年科技晨光计划资助项目(2014070404010232);武汉大学软件工程国家重点实验室开放基金资助项目(SKLSE-2012-09-21)

**作者简介**:熊 伟(1973—),男,博士生,讲师.研究方向:开放系统、服务计算和软件工程.E-mail:xwei9093@126.com

*通信联系人 E-mail:bingli@whu.edu.cn

**网络出版时间**:2014-12-3 17:24:00 **网络出版地址**:http://www.cnki.net/kcms/detail/51.1596.T.20141203.1724.001.html

的 48%[2]。在现有的数据中心设计之初,由于较少考虑到能耗问题,为应对突发情况、提高网络的可靠性和保证服务质量采用了冗余设计和留有资源余量的方法,而维持这些设备消耗了大量的能量。

云平台的耗能问题作为云计算环境的一个重要问题,已被众多国内外研究者高度关注。通过对作业运行规律的调研,发现具有较大的节能空间。通过运用虚拟化技术和硬件节能技术,该问题可望被解决[3-4]。该问题的研究主要考虑 3 个方面:1)实现多种能耗监控与测量方法,及时准确获取原始数据;2)设计节能机制并进行优化,在降低能耗、满足性能 2 方面进行权衡;3)如何建立能耗模型,能预测能量消耗的趋势和逻辑关系。

已有研究工作大多基于应用的请求率来优化资源的供给方案。其中:有些研究通过实时的请求变化情况,判断当前应用的运行情况,动态地增减系统资源[5];有些研究通过预测应用的请求率来达到提前供给资源,保证服务质量的目的[6];还有一些研究同时依据预测和实时响应来为应用提供弹性资源[6]。

Nathuji 等[5]实现了 VirtualPower 系统,综合利用如 DVFS 等硬件节能技术和减少所使用的硬件数量的软件技术。通过建立软件层面定义的电能状态和硬件的不同状态(ACPI 标准,硬件可以并不支持这么多状态)之间的映射,可以很方便地使用软件方法进行 CPU 的动态调节。在单机支持软件调节的基础上,综合性能、平均负载、最高负载等多种因素,采用重放置和迁移等策略从不同的层次进行配置实现全局最优化节能配置。测试结果表明 VirtualPower 系统最高能节省 34% 的能耗。

Wood 等[6]实现了 Sandpiper 系统,通过在线负载预测和过载发现,使用黑盒和灰盒探测结合的方法,并利用时间序列模型,使得应用程序可以根据其在不同情况下对于能耗变化做出相应调整。

Wang 等[7]提出另一种方法,给出一个混合了慢时标非定常性和快时标随机性的分析模型,能自适应性配置更新服务器的容量,以匹配当前系统负载强度的要求。

另外,一些工作关注了系统资源的快速分配,降低资源实时分配的时间和性能开销,从而快速对负载的变化进行响应,以提高资源利用率。

Xiao 等[8]设计并实现了一个可对资源进行自动伸缩的 IaaS 平台,该方法只适用于 CPU 密集型应用,对于 PaaS 上大量的在资源开销方面不同的应用而言,该方法没有充分考虑它们的差异。针对 CPU 密集型应用,该方法提出一种基于 CPU 占用情况的弹性策略,系统依据 CPU 开销情况执行弹性操作。作者参照该方法实现了一个基于 CPU 占用情况的弹性策略,并与提出的方法进行了实验对比。

Gong 等[9]对应用进行了基于向量的特性描述,但在其实际系统的实验验证里,并没有对 CPU、I/O 等多种资源的消耗情况同时进行考虑,其应用部署策略也没有采取资源互补的搭配部署方式。

Qin 等[10]关注了分布式系统上的 I/O 密集型应用,分析了 I/O 密集型应用的特点和原因,提出针对 I/O 密集型应用的 2 种调度算法。但是该方法对资源的搭配使用考虑不够,如果系统的 I/O 资源不是性能的瓶颈的话,也很难证明该方法的有效性。

综上所述,目前的相关研究工作较少有对平台上应用的特性进行综合考虑,它们或者忽视了应用往往是对多种资源进行竞争,而只局限地考虑了单一资源,或者忽视了应用请求随时间变化体现出的重要规律。作者提出一种云计算环境下基于能耗感知的弹性资源管理机制,在保证应用服务质量的前提下,尽可能地减低能耗。

使用一个名为 CloudCRM 在线 SaaS 平台来进行案例研究。该平台的业务部门负责营销策略,其通过多租户(multi-tendency)的方法来满足客户个性化定制的需求;该平台的技术部门则负责开发和维护系统。根据历史监测(图 1),其业务负载通常表现趋势性,周期性和随机性。能否将其在保持性能的前提下,降低能耗,并保持系统的稳定性和可靠性?

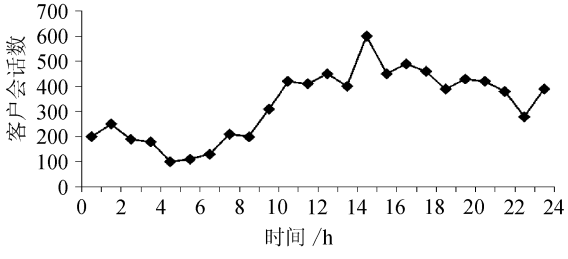


**图 1　一日的客户会话数演化**

**Fig. 1　Evolution of client sessions one day**

作者的主要贡献包括:

1)通过挖掘应用的访问日志,找出应用请求率随时间的变化规律,以实现对用户请求率的预测。并且,通过结合监测记录,建立起各项资源开销等指标的特征模型。

2)提出一种结合请求率的预测与资源开销感知,基于应用执行单元的请求分配机制。该机制根

据应用请求将任务映射到资源开销相对固定的应用执行单元,在保证了用户服务质量的同时降低能耗。并且由于引入了请求率的预测,其能有效地减少了弹性操作的频率,从而降低了弹性操作带来的额外开销。

## 1 问题定义

为了能提供可靠、高效的应用服务,SaaS 平台上应用的服务往往由多个应用执行单元共同承担完成,并且通常单个应用执行单元不会独自占尽虚拟机上的全部资源。具体地,任务被分发给各个应用执行单元,由其负责管理所在服务器提供的资源,完成对请求的响应和处理,并将结果返回给用户。

问题的形式化描述如下:

给定 $D$ 为一个 3 元组集合,其每个元素表示为 $(i,t,d_{it})$。其中,$i$ 为应用执行单元,$t$ 为时刻,$d_{it}$ 为主控制节点在时间 $t$ 对于应用执行单元 $i$ 的控制操作。控制操作包括当预测结果或监测数据满足特定条件时,会触发的扩展或伸缩操作。

给定 $RC$ 表示应用资源消耗与其请求率之间的函数关系:

$$RC = \{C_1(r),\cdots,C_i(r),\cdots,C_n(r)\}。$$

其中:$n$ 为资源种类数;$C_i(r)(1 \le i \le n)$ 为第 $i$ 种资源消耗随请求率 $r$ 的变化函数,特别地,规定 $C_1(r)$、$C_2(r)$ 和 $C_3(r)$ 分别表示 CPU、内存和磁盘的能耗情况。

给定 $Y$ 为一个 3 元组集合,其每个元素表示为 $(i,t,y_{it})$。其中,$i$ 为应用执行单元,$t$ 为时刻,$y_{it}$ 为主控制节点在时间 $t$ 观察到的应用执行单元 $i$ 的请求率。未知的 $y_{i(t+1)}$ 可以根据日志信息 $y_t = (y_{1t},y_{2t},\cdots,y_{it},\cdots,y_{kt})$ 来预测,然后根据预测的 $y_{i(t+1)}$、$RC$ 与 3 元组集合 $D$ 生成时间 $t+1$ 时的控制操作向量 $\boldsymbol{d}_{(t+1)}$。其目的是将请求的预测、负载均衡和节省能耗结合起来,达到最优。

## 2 整体框架

### 2.1 能耗指标的获取

运行在 SaaS 平台上的应用执行单元会消耗系统资源,这些资源主要包括 CPU 时间、磁盘时间、内存占用等。这些资源的消耗量往往随应用的请求率有规律地变化。根据调研,应用的 CPU 等资源占用率与请求率存在非线性关系,而 IO、内存等资源占用率与请求率则存在明显线性关系。据此,下面分别定义 CPU、内存、磁盘等子系统的能耗模型。

#### 2.1.1 CPU 能耗模型

在 $t$ 时刻,应用执行单元的 CPU 单位时间能耗非线性关系模型定义为:

$$C_1(r) = \alpha_{\text{cpu}} \times r(t)^{\beta_{\text{cpu}}} + \gamma_{\text{cpu}}(t) \tag{1}$$

其中:$r(t)$ 为在 $t$ 时刻的请求率;$\alpha_{\text{cpu}}$、$\beta_{\text{cpu}}$ 和 $\gamma_{\text{cpu}}$ 为能耗模型中的相关参数,其值可以采用非线性回归的方法拟合得到。

当系统处于稳定工作状态时,应用执行单元的单位时间能耗可以表示为:

$$C_1(r) = \alpha_{\text{cpu}} \times r^{\beta_{\text{cpu}}} + \gamma_{\text{cpu}} \tag{2}$$

其中,$r$ 为稳定工作状态时的请求率。

#### 2.1.2 内存能耗模型

应用执行单元的内存单位时间能耗线性关系模型可以定义为:

$$C_2(r) = \alpha_{\text{mem}} \times r + \gamma_{\text{mem}} \tag{3}$$

其中,$\alpha_{\text{mem}}$ 和 $\gamma_{\text{mem}}$ 为能耗模型中的相关参数,其值可以采用线性回归的方法拟合得到。

#### 2.1.3 磁盘能耗模型

磁盘能耗模型相对较难建立,在数据中心服务器中,磁盘以 RAID 的形式存在,有单独的控制器,因此无法知道磁盘的功耗状态以及磁盘硬件缓冲的影响,只能简化处理,这样其模型类似于内存能耗模型,可以定义为:

$$C_3(r) = \alpha_{\text{disk}} \times r + \gamma_{\text{disk}} \tag{4}$$

其中,$\alpha_{\text{disk}}$ 和 $\gamma_{\text{disk}}$ 为能耗模型中的相关参数,其值可以采用线性回归的方法拟合得到。

### 2.2 $t$ 时刻请求率 $r(t)$ 的预测

傅里叶变换可以将时间序列数据转换为频率序列数据以抽取时间序列的特征。但是由于傅里叶变换,而其本质上具有时域和频域局部化矛盾,使用小波变换分析和描述应用在各个单位时间内的平均请求率随时间的变化规律,该变换较好地解决了上述矛盾。

回顾在问题定义中的内容,未知的 $y_{i(t+1)}$ 可以根据日志信息 $y_t = (y_{1t},y_{2t},\cdots,y_{it},\cdots,y_{kt})$ 来预测。

具体地,可将时间序列 $y_{i(t+1)}$ 展开为:

$$y_{i(t+1)} = \sum_{j=1}^{l} \omega_j h(j) \tag{5}$$

其中,$\omega_j$ 为隐含层到输出层权值,$h(j)$ 为第 $j$ 个隐含层节点的输出,$l$ 为隐含层节点数。更进一步,$h(j)$ 定义为式(6):

$$h(j) = h_j\left(\frac{\sum_{i=1}^{k} \omega_{ij}x_i - b_j}{a_j}\right) \tag{6}$$

其中:$j = 1,2,\cdots,l$;$\omega_{ij}$ 为输入层和隐含层的连接权值;$b_j$ 为小波基函数 $h_j$ 的平移因子;$a_j$ 为小波基函数 $h_j$ 的伸缩因子。

采用 Morlet 母小波基函数,其定义为:

$$y = \cos(1.75x)e^{x^2/2} \tag{7}$$

小波网络权值参数修正公式如下:

1)计算预测误差

$$e = y_n - y \tag{8}$$

其中,$y_n$ 为期望输出,$y$ 为预测输出。

2)根据误差修正相关权值和系数

$$\omega_{ij}^{(i+1)} = \omega_{ij}^{i} + \Delta\omega_{ij}^{(i+1)} \tag{9}$$

$$a_j^{(i+1)} = a_j^{i} + \Delta a_j^{(i+1)} \tag{10}$$

$$b_j^{(i+1)} = b_j^{i} + \Delta b_j^{(i+1)} \tag{11}$$

其中,梯度公式如下:

$$\Delta\omega_{ij}^{(i+1)} = -\eta\frac{\partial e}{\partial\omega_{ij}^{i}} \tag{12}$$

$$\Delta a_j^{(i+1)} = -\eta\frac{\partial e}{\partial a_j^{i}} \tag{13}$$

$$\Delta b_j^{(i+1)} = -\eta\frac{\partial e}{\partial b_j^{i}} \tag{14}$$

其中,$\eta$ 为学习速率。

算法的时间复杂度为 $O(nkl)$,其中,$n$ 为迭代次数,$k$ 为输入层节点的数目,$l$ 为隐含层节点的数目。

### 2.3　基于能耗指标的弹性机制

回顾在问题定义中的内容,将根据预测的 $y_{i(t+1)}$、$RC$ 与 3 元组集合 $D$ 生成时间 $t+1$ 时的控制操作向量 $\boldsymbol{d}_{(t+1)}$。

为寻求控制操作向量 $\boldsymbol{d}_{(t+1)}$,采用代价函数

$$J(t) = Q \parallel w \cdot C(y_{(t+1)}) \parallel^2 + R \parallel \boldsymbol{d}_{(t+1)} - \boldsymbol{d}_t \parallel^2 \tag{15}$$

其中,$Q$ 为能耗系数,$R$ 为稳定系数,$w$ 为不同子系统能耗的权重,$\boldsymbol{d}_t$、$\boldsymbol{d}_{(t+1)}$ 为控制操作向量。

采用梯度下降法来获取优化的控制操作向量 $\boldsymbol{d}_{(t+1)}$,梯度公式为:

$$\frac{\partial J}{\partial \boldsymbol{d}_{(t+1)}} = -2R \cdot (\boldsymbol{d}_{(t+1)} - \boldsymbol{d}_t) \tag{16}$$

算法的时间复杂度为 $O(nk)$,其中,$n$ 为迭代次数,$k$ 为应用执行单元的数目。

## 3　实验评估

### 3.1　实验设计

为了评估该系统,实验采用如下配置环境:

1)该集群包括 18 个节点,每个节点有 32 位 8 核 CPU 和 16 GB RAM。

2)所有节点都用 CentOS 6.2,Java 1.6.022。

为达到实验目的,部署了 1 个名为 CloudCRM 的 SaaS 平台来进行验证,是用 Java 实现的。在该平台上分别实现了 4 种弹性策略(详见 3.2 节),并用相应的评测指标(详见 3.3 节)进行比较。为采集相关信息,在平台上部署了监控程序,其监控系统的状态并将其记录在日志中。

### 3.2　比较方法

对比了以下 4 种弹性策略,分别是:

1)基于 CPU 占用的弹性策略 C(CPU based elastic approach)。参照 Xiao 等的工作[8]实现了该策略,其以单一的 CPU 开销情况作为系统执行弹性操作的依据。

2)基于请求的弹性策略 R(request based elastic approach)。参照 Wood 等在 Sandpiper 系统[6]上的工作实现了该策略,其以应用实时的请求率作为系统执行弹性操作的依据。

3)基于资源消耗特征的弹性策略 CR(based elastic management-resource consumption)。这是作者提出的策略,但仅考虑应用的资源消耗,未考虑请求变化。这个策略用来评估考虑请求预测后的效果。

4)请求率预测与资源开销实时监测相结合的基于能耗感知的弹性策略 ERMPEC。这是作者提出的完整的策略,包含了预测机制。

### 3.3　评价指标

为评测不同的弹性策略,使用效能功耗比(performance per Watt)、全局响应时间(response-time)和可靠性(reliability)作为评价指标。其中,效能功耗比为每瓦特的电能消耗可以完成的任务数。

### 3.4　实验结果分析

通过对日志的分析,获得的实验结果如表 1 所示。表 1 中,效能功耗比、全局响应时间和可靠性(当平台处于高负载状态时,可能会产生应用执行单元执行的失败,其可以被日志记录下来,由此计算可靠性)都是取多个不同应用的均值。

**表 1　性能比较**

**Tab. 1　Performance comparison**

| 方法 | 响应时间/ms | 吞吐率/% | 可靠性/% | 效能功耗比 |
|---|---|---|---|---|
| C | 688.75 | 33.37 | 89.95 | 230.6 |
| R | 878.73 | 30.34 | 90.07 | 251.0 |
| CR | 1 473.92 | 18.32 | 89.74 | 295.0 |
| ERMPEC | 925.66 | 28.86 | 91.81 | 321.1 |

从表 1 中可知,与 C 策略相比,ERMPEC 平均可以增加 28.2% 的功耗比。ERMPEC 在保证服务

质量的表现上,与 R 策略相比并无明显劣势。

与 R 策略相比,ERMPEC 平均可以增加 21.9% 的功耗比。ERMPEC 在保证服务质量的表现上,在保证服务质量的表现上,两者表现相当。

与 CR 相比,由于引入了预测,该 ERMPEC 无论是在节省资源功耗方面,还是在保证服务质量方面,都得到了有效提高。

综上所述,ERMPEC 能够在保证服务质量的情况下,有效减少节省资源功耗。

图 2 反映了 4 种方法的效能功耗比随时间演化的过程。图 2 中:横轴表示时间,每 43.6 min 对应用所对应的效能功耗比进行一次统计,因此 24 h 的实验周期内共有 33 个统计点;纵轴表示统计所得的效能功耗比。

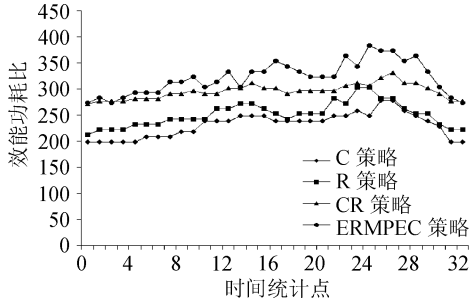


**图 2 不同弹性策略下应用的效能功耗比比较**

**Fig. 2 Performance per Watt comparison under different elasticity strategy**

实验表明,基于应用功耗的 2 种弹性策略(CR 策略与 ERMPEC 策略),相比其他 2 种策略(C 策略与 R 策略),获得了更高的效能功耗比,因此基于应用功耗的弹性策略更加节省服务器资源。进一步看,结合了应用请求率预测的 ERMPEC 策略相比 CR 策略,获得了更高的效能功耗比,这表明通过对应用请求率的预测,弹性机制性能得到了进一步提高。需要注意的是,该方法适用于请求率规律相对较好的应用,对于请求规律性较差的应用,基于请求率预测的方法就失去了其优越性。

## 4 结 论

提出一种云计算环境下基于能耗感知的弹性资源管理方法 ERMPEC。通过分析应用的日志信息,对应用请求率进行预测;并结合对应用所在系统环境的监测记录分析,建立起涵盖应用请求变化信息和资源占用信息的模型,来达到云环境场景中在满足功能需求前提下降低能耗的目的。今后考虑把系统的应用特征作为参数加入到模型中,比如更好地支持各种不同的 SaaS 系统。

**参考文献:**

[1] Koomey J G. Worldwide electricity used in data center [J]. Enviromental Research Letters, 2008, 3(3): 034008.

[2] Lin Chuang, Tian Yuan, Yao Min. Green network and green evaluation: Mechanism, modeling and evaluation[J]. Chinese Journal of Computers, 2011, 34(4): 593 – 612.

[3] Stoess J, Lang C, Bellosa F. Enenrgy management for hyper-visor-based virtual machines[C]//Proceedings of the USENIX Annual Technical Conference. Berkeley: Usenix Association, 2007: 1 – 14.

[4] McIntosh-Smith S, Wilson T, Crisp J, et al. Energy-aware metrics for benchmarking heterogeneous Systems[J]. ACM SIGMETRICS Performance Evalution Review, 2011, 38(4): 88 – 94.

[5] Nathuji R, Schwan K. VirtualPower: Coordinated power management in virtualized enterprise systems[C]//Proceedingss of the 21th ACM SIGOPS Symposium on Pperating Systems Principles (SOSP'07). New York: ACM, 2007: 265 – 278.

[6] Wood T, Shenoy P, Venkataramani A, et al. Black-box and gray-box strategies for virtual machine migration[C]// Proceedings of 4th USENIX Symposium on Networked Systems Design & Implementation. Berkeley: Usenix Association, 2009: 229 – 242.

[7] Wang Kai, Lin Minghong, Ciucu F, et al. Characterizing the impact of the workload on the value of dynamic resizing in data centers[C]//Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems. New York: ACM, 2012: 405 – 406.

[8] Xiao Z, Chen Q, Luo H. Automatic scaling of internet applications for cloud computing services[J]. IEEE Transaction on Computers, 2014, 63(5): 1111 – 1123.

[9] Gong Z, Ramaswamy P, Gu X, et al. Siglm: Signature-driven load management for cloud computing infrastructures[C]// Proceedings of the 17th International Workshop on IEEE Quality of Service, IWQoS 2009. Oakland: IEEE Press, 2009: 1 – 9.

[10] Qin X, Jiang H, Zhu Y, et al. Improving the performance of I/O—Intensive applications on clusters of workstations [J]. Cluster Computing, 2006, 9(3): 297 – 311.

(编辑 杨 蓓)