一种基于 K 均值聚簇的虚拟机分类与部署方法

李俊雅[1] 牛思先[2*] 程 星[3]

[1](济源职业技术学院 河南 济源 459000)

[2](百色学院信息工程学院 广西 百色 533000)

[3](郑州大学计算机学院 河南 郑州 450001)

摘 要 云数据中心环境下,虚拟机部署结果对主机能耗与服务等级协议 SLA 的遵守均具有重要影响。为了降低数据中心能耗与 SLA 违例,提出一种基于三门限值的高能效虚拟机部署优化算法。基于历史数据集,设计一种中档四分位的 K-均值聚簇方法以产生主机 CPU 利用率的三个门限值;依据三个门限值,将主机划分为低载主机、轻量负载主机、正常负载主机和重载主机四种类型;为了对重载主机实施虚拟机迁移,分别针对计算密集型任务和 I/O 密集型任务设计两种虚拟机迁移选择方法,实现虚拟机优化部署;通过现实负载流数据对算法进行仿真分析。结果表明,该算法不仅可以有效降低能耗,而且 SLA 违例也较低,相比单纯降低能耗而忽略性能的同类算法,具有更高的能效。

关键词 云计算 数据中心 虚拟机部署 虚拟机迁移 迁移选择

中图分类号 TP393 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2019.08.003

A VIRTUAL MACHINE CLASSIFICATION AND PLACEMENT METHOD BASED ON K-MEANS CLUSTERING

Li Junya[1] Niu Sixian[2*] Cheng Xing[3]

[1](*Jiyuan Vocational and Technical College*, *Jiyuan 459000*, *Henan*, *China*)

[2](*School of Information Engeering*, *Baise University*, *Baise 533000*, *Guangxi*, *China*)

[3](*School of Computer*, *Zhengzhou University*, *Zhengzhou 450001*, *Henan*, *China*)

Abstract In cloud data center environment, virtual machine placement greatly affects on the energy consumption of hosts and complied with Service Level Agreement (SLA). To reduce the energy consumption and SLA violation, we proposed a high energy-efficient virtual machine placement optimization algorithm based on three thresholds. Based on the historical data set, we designed a K-means clustering algorithm with Midrange-Interquartile range to produce three thresholds of CPU utilization on hosts. According to the three thresholds, we divided all hosts into four types: hosts with little load, hosts with lightly load, hosts with moderate load and hosts with heavy load. Then, for migrating some virtual machines from the hosts with heavy load, we designed two kinds of virtual machine migration selection algorithms for the computation intensive tasks and the I/O intensive tasks respectively. Finally, we performed some simulation experiments using the real-world workload. The results show that our algorithm not only can reduce the energy consumption, but can maintain low SLA violation. Compared with the same type of algorithms that only reduce the energy consumption without considering the performance, our algorithm can get higher energy-efficiency.

Keywords Cloud computing Data center Virtual machine placement Virtual machine migration Migration selection

---

收稿日期:2018 - 12 - 14。国家自然科学基金项目(61807031)。**李俊雅**,讲师,主研领域:云计算技术。**牛思先**,教授。**程星**,讲师。

0 引 言

云计算技术的出现使得大规模数据中心的建立可以更好地满足社会对于计算能力的巨大需求[1]。然而,这种大规模云数据中心正在消耗庞大的电力资源,进而导致的高能耗和巨量的碳排放问题。研究表明[2],2013 年,全球数据中心的总体电力消耗约 4.35 千兆瓦特,并且以每季增长速率 15% 递增。云数据中心的高能耗会导致一系列问题,包括:能量浪费、较低的投入产出率、系统稳定性以及带来温室效应的碳排放[3]。

另一个关键问题是如何在确保服务质量 QoS 的前提下降低数据中心能耗。云系统中的 QoS 通常以服务等级协议 SLA 的形式表达。然而,目前数据中心的资源利用率仅为 50% 左右,较低的资源利用率会导致巨大的能量浪费,改善主机资源利用率将有助于降低能耗。但是,一味改进主机资源利用率反过来会影响系统的 QoS 交付。因此,对于云数据中心而言,必须均衡地考虑能耗降低和 SLA 违例问题,设计高能效的部署方法。

为了降低能耗和 SLA 违例,实现数据中心的高能效,本文设计一种新的虚拟机部署算法,算法将根据处理负载,以自适应的方法设置三个门限值,将主机划分为具有不同负载的四种类型,包括:低载主机、轻量负载主机、正常负载主机和重载主机,并通过设计的虚拟机迁移选择算法对低载主机和重载主机进行虚拟机迁移,同时维持轻量负载主机和正常负载主机不变,以此在能耗降低和性能保障上作出均衡。

1 相关研究

目前,云数据中心的能耗管理方法分为三种:动态性能扩展 DPS 方法[4-6]、基于门限值的启发式方法[7-11]和基于历史数据统计分析的决策方法[12-14]。DPS 方法中,系统组件可动态调整其性能以节省能耗,如逐渐降低 CPU 的频率或电压。DPS 可在资源未充分利用的情况下大幅降低能耗。采用 DPS 方法的关键技术是动态电压/频率调整 DVFS,DVFS 可分为三类:基于间隔的 DVFS、基于任务内的 DVFS 和基于任务间的 DVFS。基于间隔的 DVFS 方法利用过去时段的 CPU 利用数据预测未来的利用率,并调整 CPU 性能,该方法在多核系统中比较最优在线算法具有更好的性能。相比而言,基于任务内的 DVFS 方法区分不同的任务并将其分配至不同速率的 CPU 上,该方法在负载提前预知的情况下表现得简单而高效,但不适用于负载变化的云系统。基于任务间的 DVFS 方法利用程序的结构知识,在任务内调整处理器频率进而节省能耗,但全局数据中心的能耗仍然较高。

基于门限值的启发式方法通过设置门限改善资源利用率,进而降低能耗并确保系统交付的 QoS。文献[7]提出一种单门限方法 ST,ST 设置一个利用率门限值以确保所有主机的 CPU 占用在该门限以下,以此控制虚拟机迁移,其能耗节省优于 DVFS。文献[8]提出双门限值方法 DT,包括上门限和下门限,并确保所有主机的 CPU 占用处于两个门限值之间。对于 CPU 占用未处于两个门限值之间的主机,则需要进行虚拟机迁移。文献[9]基于双门限 DT 设计了一种改进 PSO 算法 MPSO,该算法可以减少活动主机的数量和虚拟机迁移次数,以此降低能耗。文献[10]研究云数据中心的能效问题,并提出基于 DT 的虚拟机选择算法,该算法重点考虑了 CPU 占用率和资源满意度。文献[11]提出一种静态三门限虚拟机部署算法,在考虑能效的情况下将最优门限间隔设置为 40%,实现了能效优化。以上基于门限值的启发式方法尽管可以节省大量能耗,但在未知可变的负载状况下显得并不适用。

基于历史数据统计分析的决策方法是改进数据中心能效的有效方法。文献[12]提出一种自适应双门限算法,该算法通过利用近期负载数据特征预测资源占用情况。文献[13]利用权重线性回归预测未来负载并优化资源分配。文献[14]通过虚拟机部署预测改进数据中心能效,但所考虑的负载类型为单一计算密集型,且仅考虑了能耗降低,而未考虑能量使用效率,即忽略了性能保障和 SLA 违例降低问题。尽管基于历史数据统计分析的决策方法可以大幅降低能耗,但已有工作未考虑负载类型,且仅考虑虚拟机部署过程中的能耗降低,而未考虑 SLA 违例等性能因素,全局提高能效。

2 优化算法设计

2.1 自适应门限值部署模型

为了解决高能耗问题,将数据中心的主机基于 CPU 利用率进行划分,设置三个 CPU 利用率的门限值为 T_a、T_b 和 T_c,且 $0 \leqslant T_a < T_b < T_c \leqslant 1$。三个门限值将主机划分为四种类型:低载主机、轻量负载主机、正常负载主机和重载主机。低载主机的 CPU 利用率范围为 $[0, T_a]$,轻量负载主机的 CPU 利用率范围为 $[T_a, T_b]$,正常负载主机的 CPU 利用率范围为 $[T_b, T_c]$,重

载主机的 CPU 利用率范围为[T_c,1]。图 1 为自适应三门限值算法的执行过程,具体思想为:对于每个数据中心中的主机,首先获取其 CPU 利用率 U_i,然后将 U_i 与三个门限值进行比较,将主机进行分类:① 若 $U_i \geqslant T_c$,则该主机为重载主机,为了避免高 SLA 违例,需要从该主机迁移部分虚拟机迁移至具有最高能效的轻量负载主机上;②若 $T_b \leqslant U_i < T_c$,则该主机为正常负载主机,该主机无需进行虚拟机迁移;③ 若 $T_a \leqslant U_i < T_b$,则该主机为轻量负载主机,该主机无需进行虚拟机迁移;④ 若 $U_i < T_a$,则该方机为低载主机,该主机上的所有虚拟机需要被迁移至具有最高能效的轻量负载主机上,且需要将该主机转换为休眠模式,以降低静态能耗。当需求增加时,该主机将再次激活。

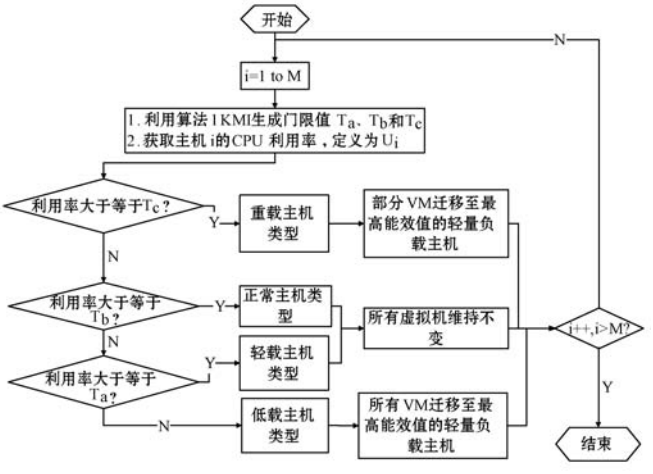

图 1　自适应三门限值下的处理流程

该模型重点需要解决两个问题:① 如何决定三个门限值的具体大小;② 如何在重载主机上选择需要迁移的虚拟机,以适应于计算密集型任务和 I/O 密集型任务的需求。以下分别进行讨论。

2.2　自适应三门限值取值机制

提出一种基于中档四分位的 K-均值聚簇算法 KMI 计算三个门限值的大小。令 $H = \{h_1, h_2, \cdots, h_M\}$ 表示云数据中心中 M 个主机的集合,令 $D = \{U_1, U_2, \cdots, U_n\}$ 表示数据集,且 $U_t \in D(1 \leqslant t \leqslant n)$ 表示时间 t 时主机 $h_f \in H$ 的 CPU 利用率。算法 1 给出 KMI 算法的伪代码。

算法 1　KMI

Input:D, k, d

Output:T_a, T_b, T_c

1. $R = \text{Cluster}(D, k)$　　//K-均值聚簇算法
2. **for** $i = 1$ to R. length **do**
3. $MR_i = (\max(R_i) + \min(R_i))/2$
4. $MR[i] = MR_i$
5. **end for**
6. $IQR = TQ_3(MR) - TQ_1(MR)$
7. 计算 T_a、T_b、T_c

算法 1 详细说明:算法输入参数包括 CPU 利用率数据集 D、聚簇数量 k 和预定义的合并因子 d,算法输出为三个 CPU 利用率的门限值 T_a,T_b,T_c。KMI 算法的步骤 1 利用 K-均值聚簇算法将 D 划分为 k 个簇:表示为$\{R_1, R_2, \cdots, R_k\}$,使得:

1) $R_i = \{U_{pi-1} + 1,\ U_{pi-1} + 2, \cdots, U_{pi}\} \in D$

2) $0 = P_0 < P_1 < P_2 < P_3 < P_4 < \cdots < P_i = n$

3) $R_i \cap R_j = \varnothing$,对于 $1 \leqslant i, j \leqslant k$。

步骤 3 中,对于每个簇 R_i,算法获取的中位值为:

$$MR_i = (\max(R_i) + \min(R_i))/2 \quad 1 \leqslant i \leqslant k \tag{1}$$

式中:参数 $\max(R_i)$ 表示簇 R_i 的最大值,$\min(R_i)$ 表示簇 R_i 的最小值,步骤 4 可以获得簇 i 的中位值 MR_i,该过程可产生 $MR = \{MR_1, MR_2, \cdots, MR_i, \cdots, MR_k\}$。算法在步骤 6 可获得集合 $MR = \{MR_1, MR_2, \cdots, MR_i, \cdots, MR_k\}$ 的四分位差为:

$$IQR = TQ_3 - TQ_1 \tag{2}$$

式中:TQ_3 为 MR 的第三个四分位,TQ_1 为 MR 的第一个四分位。式(2)的目标是得到集合 MR 的 IQR。算法在步骤 7 通过以下三个公式分别计算三个门限值为:

$$T_c = 1 - d \times IQR \tag{3}$$

$$T_b = 0.9 \times (1 - d \times IQR) \tag{4}$$

$$T_a = 0.5 \times (1 - d \times IQR) \tag{5}$$

式中:d 表示合并因子,描述系统进行虚拟机合并的积极程度。d 值越小,能耗越小,SLA 违例越高,反之亦然。

2.3　重载主机上虚拟机迁移选择

考虑数据中心的执行负载为计算密集型任务或 I/O 密集型任务。当一个应用任务的完成时间主要由 CPU 的速度决定时即认为是计算密集型任务;当一个应用任务的完成时间主要由等待输入输出操作完成时间决定时即认为是 I/O 密集型任务。换言之,此时将花费更多时间等待数据而非处理数据。这种情况下,CPU 和内存将消耗更多能量。

令 S_{VM} 表示给定一个重载主机上当前运行的虚拟机集合,令 C^i_{VM} 和 M^i_{VM} 分别表示虚拟机 i 的 CPU 和内存利用率,$1 \leqslant i \leqslant S_{VM}$,令 C^e_{VM} 和 M^e_{VM} 分别表示任意虚拟机 e 的 CPU 和内存利用率,$1 \leqslant e \leqslant S_{VM}$,且 $e \neq i$。

1) MRCU 算法:CPU 利用率与内存利用率之比最大化算法。若某一主机由于计算密集型任务成为重载主机,则利用 MRCU 算法进行迁移虚拟机选择。MRCU 算法选择迁移的虚拟机 v 需要满足:

$$C^v_{VM}/M^v_{VM} > C^e_{VM}/M^e_{VM} \quad v \in S_{VM}, e \in S_{VM}, v \neq e \tag{6}$$

若主机由于执行计算机密集型任务出现重载，相比其他系统组件（如内存），CPU 将占据总体能耗的大部分。式（6）表明 C^v_{VM} 越高，M^v_{VM} 越低，C^v_{VM}/M^v_{VM} 将越大。因此，式（6）将选择一个拥有最高 C^v_{VM}/M^v_{VM} 值的虚拟机进行迁移，由于 CPU 利用率越高，能耗越多，而内存利用率越低，则虚拟机迁移的能耗越少。MRCU 算法在选择迁移虚拟机时同步考虑 CPU 和内存因素。

2）MPCU 算法：当主机由于执行 I/O 密集型任务发生重载时，为了降低潜在的 SLA 违例，可利用 CPU 利用率与内存利用率之积最小化算法 MPCU 处理。MRCU 算法选择迁移的虚拟机 v 需要满足：

$$C^v_{VM} \times M^v_{VM} > C^e_{VM} \times M^e_{VM} \quad v \in S_{VM}, e \in S_{VM}, v \neq e \tag{7}$$

若主机由于 I/O 密集型任务出现重载，此时 CPU 和内存将消耗大部分能量，即 CPU 因素和内存因素具有同等重要性。式（7）将选择具有最小 CPU 利用率与内存利用率之积的虚拟机迁移，由于此时迁移能耗较少，对应的由虚拟机迁移带来的性能下降也有所减少，从而可以降低 SLA 违例。MPCU 算法在选择迁移虚拟机时也同步考虑了 CPU 因素和内存因素。

2.4 能效最大化的虚拟机部署

基于三门限值和重载主机的虚拟机迁移选择，本节设计一种考虑同步降低能耗和 SLA 违例的虚拟机部署算法，并命名为最大能效算法 VPME。过程如算法 2 所示。

算法 2 VPME

Input：vmlist，hostlist，T_a，T_b，T_c
Output：allocation of virtual machines
1. vmlist. sortByDeseasingUtilization( )
2. **for**(vm：vmlist) **do**
3. minEnergyEfficiency = min，allocatedHost = ∅
4. **for**(host：hostlist) **do**
5. **if**(host meets the requirements of vm) **then**
6. CpuUtilize = getUtilizationAfterVm(host，vm)
7. **if**($T_a \leqslant$ CpuUtilize $\leqslant T_b$) then
8. power = host. getPower( )
9. powerDiff = powerAfterAllocation-power
10. firstSlaBefore = getSlaTimePerActiveHost(host)
//第一次 SLA 违例计算
11. firstSla = firstSlaAfterVM-firstSlaBefore
12. secondSlaBefore = getSlaMetric(host. getVmList( )
//第二次 SLA 违例计算
13. SLA = fristSla × secondSla//通过式(8)计算 SLA 违例
14. EnergyEfficiency = 1/(powerDiff × SLA) //计算能效
15. **if**(EnergyEfficiency > minEnergyEfficiency) **then**
//寻找满足虚拟机分配具有最高能效值的主机
16. minEnergyEfficiency = EnergyEfficiency
17. allocatedHost = host
18. **if**(allocatedHost ≠ null) **then**
19. allocated the Vm to host
20. return allocation of virtual machines

算法 2 详细说明：算法输入为虚拟机列表 vmlist、主机列表 hostlist 以及三个门限值 T_a、T_b、T_c，算法输出为虚拟机的分配结果。步骤 1 中，VPME 算法首先按 CPU 利用率将所有虚拟机降低排列。步骤 3 对最小能效值进行初始化，并将已分配的主机集合初始化为空集，以备后续分配的更新。然后，对于数据中心的每台主机（步骤 4），检测是否当前主机在可用 CPU 和内存大小上能够满足虚拟机的请求（步骤 5）。若满足，进行虚拟机分配后获取该主机的 CPU 利用率（可见步骤 6 的变量 CpuUtilize）。步骤 7 将维持该主机为轻量负载主机。步骤 8 – 步骤 9 计算虚拟机分配前后该主机的功能差值。步骤 10 – 步骤 11 计算虚拟机分配前后第一次的 SLA 违例（第一次 SLA 违例定义可参见实验部分式（9））。步骤 13 计算虚拟机分配前后第二次的 SLA 违例（第二次 SLA 违例定义可参见实验部分式（10））。步骤 13 计算 SLA 违例（见式（8）），步骤 14 计算能效值（见式（13）），步骤 15 – 步骤 17 寻找满足虚拟机分配具有最高能效值的主机，步骤 19 完成虚拟机分配。算法时间复杂度为 $O(M \times N)$，M 为主机数量，N 为虚拟机数量。

2.5 虚拟机部署优化

虚拟机部署优化是对 2.4 节执行虚拟机部署后的优化过程，过程如算法 3 所示。

算法 3 虚拟机部署优化过程

Input：T_a，T_b，T_c，hostlist
Output：migrationMap
1. **for**(host：hostlist) **do**
2. **if**(utilization ≥ T_c) **then** //若为重载主机
//则迁移虚拟机至拥有最高能效值的主机
3. overUtilizedHosts = getOverUtilizedHosts( )
4. vmsToMigrate = getVmsToMigrateFromHosts(overUtilizedHosts)
5. migrationMap = getNewVmPlacement (vmsToMigrate)
6. **else if**(utilization < T_a) **then**
//若为轻载主机，维持不变
7. lowUtilizedHosts = getLowUtilizedHost( )
8. vms ToMigrateB = getVmsToMigrateFromLowHosts (lowUtilizedHosts)
9. migrationMapB = getNewVmPlacement(vmsToMigrateB)
10. migrationMap. addAll(migrationMapB)
11. **return** migrationMap

算法 3 详细说明:算法输入为三个门限值 T_a、T_b、T_c、主机列表 hostlist,算法输出为虚拟机迁移后的重部署结果。步骤 3 – 步骤 5 通过利用虚拟机迁移选择算法(2.3 节)在重载主机上选择需要迁移的虚拟机,并将其迁移至拥有最高能效值的主机上。步骤 6 – 步骤 9 将维持正常负载主机和轻量负载主机不变。步骤 10 选择低载主机上的所有虚拟机迁移至拥有最高能效效值的主机上。算法 3 的时间复杂度为 $O(M)$,M 为主机数量。

3　仿真实验

3.1　实验环境搭建

考虑到实验的可重复性,利用 CloudSim 工具包[15]搭建云数据中心环境。实验模拟了一个包括 800 台物理主机的数据中心,由 HP Proliant G4 和 HP Proliant G5 两种主机类型组成,数量各 400 台。主机具体参数如表 1 所示。虚拟机特征参考 Amazon EC2 的 VM 类型建立,具体描述如表 2 所示。

表 1　机类型

主机类型	CPU 类型	频率/GHz	核心数	RAM/GB
HP Proliant G4	Intel Xeon 3040	1.86	2	4
HP Proliant G5	Intel Xeon 3075	2.66	2	4

表 2　虚拟机类型

虚拟机类型	CPU/MIPS	RAM/GB
High-CPU medium instance	2 500	0.85
Extra large instance	2 000	3.75
Small instance	1 000	1.70
Micro instance	500	0.61

实验中的测试负载数据利用现实负载 CoMon 项目中的 PlanetLab 提供的负载数据,负载流数据参数如表 3 所示。

表 3　负载数据特征/CPU 利用率

虚拟机数量	均值/%	第 1 个四分位/%	第 3 个四分位/%
1 052	12.31	2	15
898	11.44	2	13
1 061	10.7	2	13
1 516	9.26	2	12
1 078	10.56	2	14
1 463	12.39	2	17
1 358	11.12	2	15
1 233	11.56	2	16
1 054	11.54	2	16
1 033	10.43	2	12

两种主机类型不同负载状况下的能耗情况如表 4 所示。

表 4　不同 CPU 利用率下的能耗

CPU 利用率/%	HP Proliant G4 功耗/W	HP Proliant G5 功耗/W
0	86	93.7
10	89.4	97
20	92.6	101
30	96	105
40	99.5	110
50	102	116
60	106	121
70	108	125
80	112	129
90	114	133
100	117	135

3.2　SLA 违例指标

云数据中心拥有两种类型的 SLA 违例:单个活动主机的 SLA 违例 *SLAVH* 和由于虚拟机迁移导致的 SLA 违例 *SLAVM*。本文将该指标综合定义为:

$$SLAV = SLAVH \times SLAVM \tag{8}$$

SLAVH 表示 PH 的 SLAV 时间与 PH 活动时间的比例,即:

$$SLAVH = \frac{1}{M}\sum_{j=1}^{M}\frac{T_{s_j}}{T_{a_j}} \tag{9}$$

式中:M 表示 PHs 的数量,T_{s_j} 表示 PH_j 所经历的 CPU100% 占用从而导致 SLA 违例的总时间,T_{a_j} 表示 PH_j 在活动状态下经历的总时间。

SLAVM 表示由于迁移所导致的虚拟机性能下降与虚拟机请求的总体 CPU 能力间的比例,即:

$$SLAVM = \frac{1}{N}\sum_{i=1}^{N}\frac{C_{d_i}}{C_{r_i}} \tag{10}$$

式中:N 表示虚拟机数量,C_{d_i} 表示由于迁移导致的虚拟机 i 的性能降低估算,C_{r_i} 表示在其生命周期内虚拟机 i 请求的总体 CPU 能力。实验中将 C_{d_i} 估算为虚拟机 i 所有迁移中 CPU 利用的 10%。这意味着每次迁移可能导致 SLA 违例。一次动态迁移的时间长短取决于虚拟机使用的内存总量和可用的网络带宽。

虚拟机 i 带来的性能下降为:

$$C_{d_i} = \int_{t_0}^{t_0+T_i^m} u_i(t)\,\mathrm{d}t \times 10\% \tag{11}$$

$$T_i^m = RAM_i / BW_i \tag{12}$$

式中：t_0 表示迁移的起始时间，T_i^m 表示完成迁移的时间，$u_i(t)$ 表示虚拟机 i 对主机的 CPU 占用，RAM_i 表示虚拟机 i 使用的内存总量，BW_i 表示可用网络带宽。

3.3 能效指标

能效指标包括两个方面：能耗和 SLA 违例。基于前文定义，能效 EE 定义为：

$$EE = \frac{1}{(p_{\text{power}} \times I_{SLA})} \tag{13}$$

式中：p_{power} 表示能耗，I_{SLA} 表示 SLA 违例指标。EE 值越高，能效越高。

3.4 实验结果

实验 1 观察三门限的设置对于性能的影响。分别与基于单门限的虚拟机部署算法 STVMP[7]、基于双门限值的虚拟机部署算法 DTVMP[10] 及基于平均绝对中位差的多门限值算法 KAMVMP[14] 进行比较。单门限值设置为 0.7，双门限值设置为 0.2 和 0.8，三门限值由本文的 KMI 算法产生，算法中的 d 值设置为 1，测试负载以计算密集型任务为主，因此，利用 MRCU 算法进行迁移虚拟机选择，即本文算法 = KMI-MRCU-1，1 代表 d 值。实验比较分析了能耗、SLA 违例、SLAVH、SLAVM、虚拟机迁移量以及算法得到的能效值等性能指标，结果如图 2 – 图 4 所示。总体来看，单门限值算法的性能是最差的，主要原因是单门限值仅设置了 CPU 利用率的上限，会导致很多主机利用率较低，静态功耗较多。双门限值同时设置了 CPU 利用率的上下限，可以避免部分主机利用不充分的问题，但上下限的区间跨度仍然较大，难以适应动态的负载需求。KAMVMP 算法虽然是基于多门限值思想，但仅是以最小化能耗为目标的，其能效并不一定最优。且其虚拟机迁移选择是基于内存最小原则设计的，无法适应于不同类型的负载执行。综合比较，本文算法具有更高的能效（能耗和 SLA 违例）。

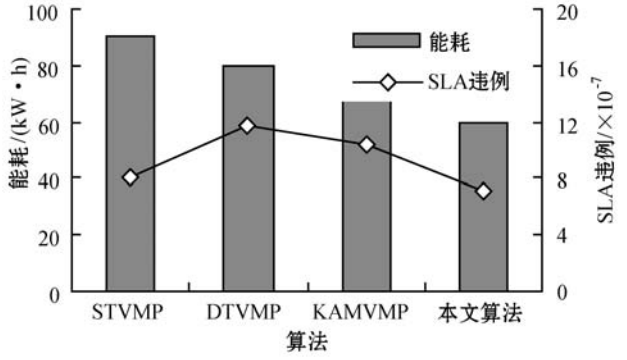

图 2 能耗与 SLA 违例

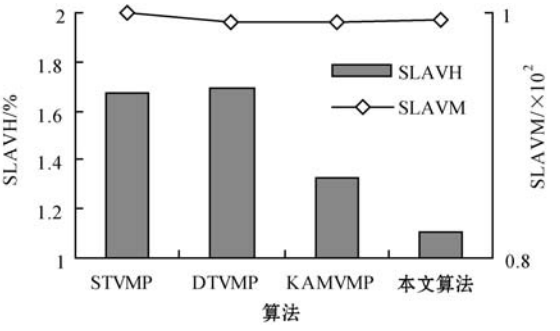

图 3 两种类型的 SLA 违例

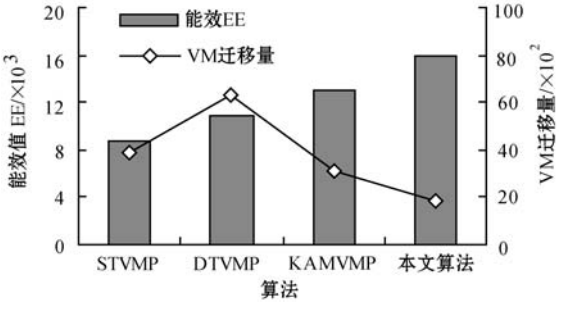

图 4 能效与 VM 迁移量

实验 2 测试执行计算密集型任务时 d 值的变化对算法性能的影响。选取 3/3/2011 数据集合作为计算密集型任务的测试数据。测试算法 = KMI-MRCU-d，d 的取值范围为[0.5,3]，递增步长为 0.5，结果如图 5 – 图 7 所示。由图 5 可知，d = 2 时能耗最小，但 d = 1 时 SLA 违例最小。同步考虑到能耗与 SLA 违例（即能效），并结合图 7 中的 EE 指标，d = 1 时算法具有最优能效。总体来看，过高的 d 值反映出虚拟机具有更高的合并积极性，能耗与 SLA 违例指标均表现出先降低后升高的趋势，因此 d 值的选取至关重要。

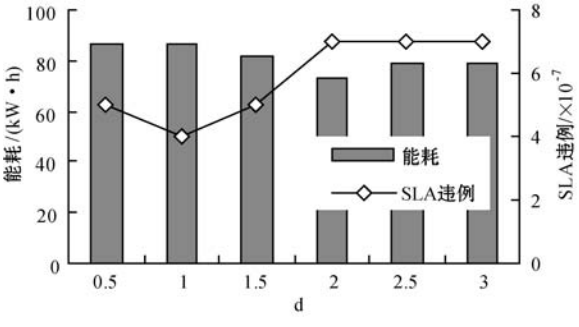

图 5 能耗与 SLA 违例

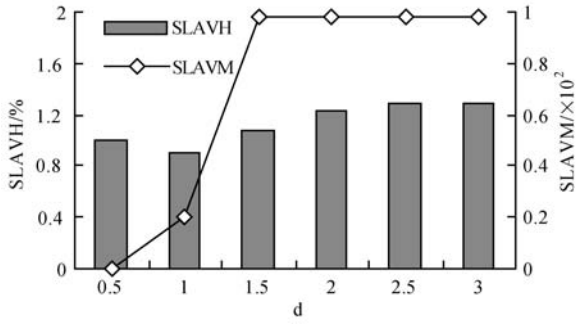

图 6 两种类型的 SLA 违例

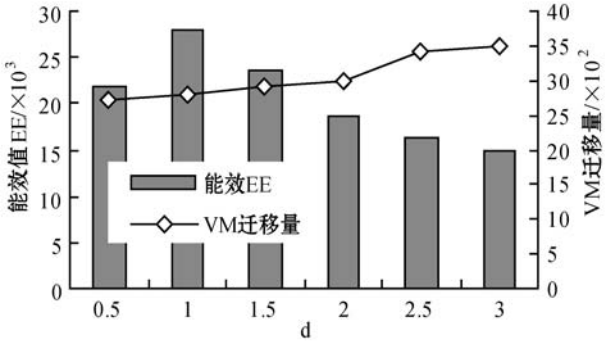

图 7　能效与 VM 迁移量

表 5 将本文算法 KMI-MRCU-1 与同类型算法进行系统比较,同样选取能耗、SLA 违例、SLAVH、SLAVM、虚拟机迁移量以及算法得到的能效值等性能指标。对比算法为非功耗感知算法 NPA(以满载形式执行所有任务)、动态电压/频率调整算法 DVFS、THR-MMT-1 算法[12]、THR-MMT-0. 8 算法[12]、MAD-MMT-2. 5 算法[12]、IQR-MMT-1.5 算法[12]及 KAM-MMS-2 算法[14]。指标中,能效 *EE* 越高越好,而能耗、SLA 违例、SLAVH 和 SLAVM 则越小越好。对于虚拟机迁移量,过多或过小的迁移量均不利于能效的提高。NPA 未采用任何能量优化机制,能耗是最高的,DVFS 通过降低 CPU 性能可以降低部分闲置能耗。NPA 和 DVFS 均未涉及虚拟机迁移,因此其他性能指标均以“ – ”表示。

表 5　计算密集型任务下的性能系统比较

算法	EE	能耗/kW · h	SLA 违例/10^{-7}	SLAVH	SLAVM	迁移量
NPA	–	2 410	–	–	–	–
DVFS	–	803	–	–	–	–
THR-MMT-1	38	99	2 718	25	0.1	19 841
THR-MMT-0.8	170	122	483	5	0.1	26 738
MAD-MMT-2.5	169	114	516	4.5	0.1	25 673
IQR – MMT-1.5	166	119	516	1.74	0.1	26 518
KAM-MMS-2	7 393	104	15	2	0.1	7 618
KMI-MRCU-1	28 425	87	4	0.8	0.003	2 889

综合来看,本文算法 KMI-MRCU-1 比较其他几种算法,在能效、能耗、SLA 违例、SLAVH、SLAVM 以及虚拟机迁移量等指标上均有 5 ~ 10 倍的性能提升,其原因可概括为:1) THR-MMT-1、THR-MMT-0. 8、MAD-MMT-2.5 和 IQR-MMT-1.5 均还是双门限值框架,而 KMI-MRCU-1 是基于自适应三门限值的,多门限值的效率更高;2) 四种比较算法更侧重于虚拟机部署过程的能耗优化,而本文算法考虑的是能效的最大化;3) 对于重载主机的发现,四种比较算法均利用历史数据的统计分析进行计算,而本文算法则是通过基于历史数据的 K-均值聚簇方法计算的;4) 对于迁移虚拟机的选择,四种比较算法均只考虑 CPU 或内存利用率的单一因素,而本文算法则同时考虑了这两个因素。

与 KAM-MMS-2 算法比较,本文算法仍是更优的,原因在于:1) KAM-MMS-2 算法仍仅是考虑能耗优化的,不是能效;2) 对于计算密集型任务,本文采取的重载主机发现与迁移选择机制更优。

实验 3 测试执行 I/O 密集型任务时 d 值的变化对算法性能的影响。选取 22/3/2011 数据集合作为 I/O 密集型任务的测试数据。测试算法 = KMI-MPCU-d,d 的取值范围为[0.5,3],递增步长为 0.5,结果如图 8 – 图 10 所示。由图 8 可知,d = 0.5 时能耗最小,而 d = 3 时 SLA 违例最小(d = 0.5 时算法部署失效)。同步考虑到能耗与 SLA 违例(即能效),并结合图 10 中的 *EE* 指标,d = 2 时算法具有最优能效。对于 I/O 密集型任务,能耗较计算密集型任务更少,但 SLA 违例更高。

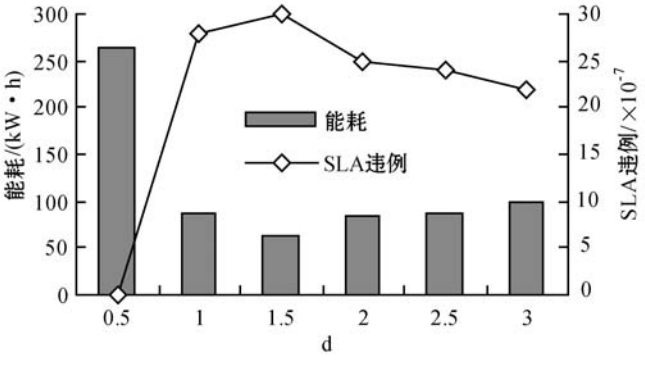

图 8　能耗与 SLA 违例

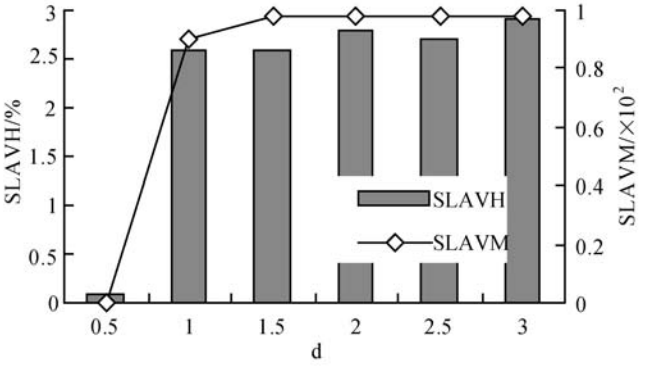

图 9　两种类型的 SLA 违例

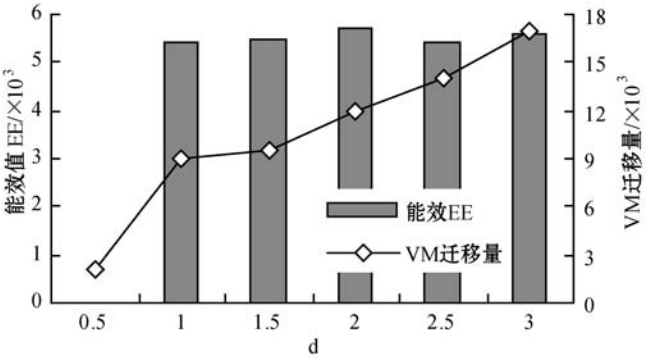

图 10 能效与 VM 迁移量

表 6 对 I/O 密集型任务下对算法性能进行了系统比较。同样地,本文算法在各个性能指标上也拥有更好的效果。其原因可参见表 5 的注释。

表 6 I/O 密集型任务下的性能系统比较

算法	EE	能耗/kW·h	SLA 违例/10^{-7}	SLAVH	SLAVM	迁移量
NPA	–	2 415	–	–	–	–
DVFS	–	1 012	–	–	–	–
THR-MMT-1	33	101	3112	27	0.11	24 254
THR-MMT-0.8	234	120	609	5.46	0.11	33 243
MAD-MMT-2.5	156	117	625	5.22	0.11	32 213
IQR-MMT-1.5	134	121	24	5.46	0.01	33 115
KAM-MMS-2	4 542	103	21	2	0.01	8 157
KMI-MRCU-1	5 678	67	26	1.8	0.01	12 318

4 结 语

为了降低数据中心能耗与 SLA 违例,提出了一种基于三门限值的高能效虚拟机部署优化算法。算法基于历史数据集,设计了一种中档四分位的 K-均值聚簇方法以产生主机 CPU 利用率的三个门限值;然后,依据三个门限值,将主机划分为低载主机、轻量负载主机、正常负载主机和重载主机四种类型;最后,为了对重载主机实施虚拟机迁移,分别针对计算密集型任务或 I/O 密集型任务设计了两种虚拟机迁移选择方法。结果表明,所提算法不仅可以有效降低能耗,而且 SLA 违例也较低,具有更高的能效。

参 考 文 献

[1] Carretero J, Blas J G. Introduction to cloud computing: platforms and solutions[J]. Cluster Computing, 2014, 17(4): 1225 – 1229.

[2] Wang L. Review of performance metrics for green data centers: a taxonomy study[J]. Journal of Supercomputing, 2013, 63(3): 639 – 656.

[3] Rincón D, Agustítorra A, Botero J F, et al. A Novel Collaboration Paradigm for Reducing Energy Consumption and Carbon Dioxide Emissions in Data Centres[J]. Computer Journal, 2013, 56(12): 1518 – 1536.

[4] Shojafar M, Canali C, Lancellotti R, et al. An Energy-aware Scheduling Algorithm in DVFS-enabled Networked Data Centers[C]// Special Session on Tools for an Energy Efficient Cloud. 2016: 387 – 397.

[5] Zhang Y, Wang Y, Hu C. CloudFreq: Elastic Energy-Efficient Bag-of-Tasks Scheduling in DVFS-Enabled Clouds [C]// 2015 IEEE 21st International Conference on Parallel and Distributed Systems(ICPADS). IEEE, 2016: 585 – 592.

[6] Arroba P, Moya J M, Ayala J L, et al. DVFS-Aware Consolidation for Energy-Efficient Clouds[C]// Proceedings of the 2015 International Conference on Parallel Architecture and Compilation(PACT). IEEE, 2016: 494 – 495.

[7] Buyya R, Ranjan R, Calheiros R N. Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities[C]// 2009 International Conference on High Performance Computing & Simulation. IEEE, 2009: 1 – 11.

[8] Beloglazov A, Buyya R. Energy Efficient Allocation of Virtual Machines in Cloud Data Centers[C]// Ieee/acm International Conference on Cluster, Cloud and Grid Computing. IEEE, 2010: 577 – 578.

[9] Li H, Zhu G, Cui C, et al. Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing[J]. Computing, 2016, 98(3): 303 – 317.

[10] Xiong F U, Zhou C. Virtual machine selection and placement for dynamic consolidation in Cloud computing environment[J]. Frontiers of Computer Science, 2015, 9(2): 322 – 330.

[11] Zhou Z, Hu Z G, Song Y, et al. A novel virtual machine deployment algorithm with energy efficiency in cloud computing[J]. Journal of Central South University, 2015, 22(3): 974 – 983.

[12] Beloglazov A, Buyya R. Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers[C]// Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science. ACM, 2010: 1 – 6.

[13] Guenter B, Jain N, Williams C. Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning[C]// 2011 Proceedings IEEE INFOCOM. IEEE, 2011: 1332 – 1340.

[14] Zhou Z, Hu Z, Li K. Virtual Machine Placement Algorithm for Both Energy-Awareness and SLA Violation Reduction in Cloud Data Centers[J]. Scientific Programming, 2016, 21(1): 1 – 11.

[15] Goyal T, Singh A, Agrawal A. Cloudsim: simulator for cloud computing infrastructure and modeling[J]. Procedia Engineering, 2013, 38(4): 3566 – 3572.