强化学习下能耗优化的虚拟机放置策略

卢海峰 顾春华 罗 飞 丁炜超 袁 野 任 强
(华东理工大学信息科学与工程学院 上海 200237)

摘 要 云数据中心的高速发展带来了非常强大的计算能力,但是伴随产生的能耗问题也日益严重。为了降低云数据中心内物理服务器的能耗开销,首先利用强化学习对虚拟机放置问题进行建模,随后结合实际问题从状态聚合和时间信度两个方面对Q-Learning(λ)算法进行优化,最后通过云仿真平台CloudSim和实际数据集对虚拟机放置问题进行实验。实验结果表明,与Q-Learning算法、Greedy算法和PSO算法相比,优化后的Q-Learning(λ)算法更有效地降低了物理服务器的能耗开销,同时针对不同数量的虚拟机放置请求也能够保证更好的结果,具有较强的实用价值。

关键词 云计算,虚拟机放置,强化学习,能耗优化,Q-Learning(λ)算法
中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.09.044

Virtual Machine Placement Strategy with Energy Consumption Optimization under Reinforcement Learning

LU Hai-feng GU Chun-hua LUO Fei DING Wei-chao YUAN Ye REN Qiang
(School of Information Science and Engineering, East China University of Science and Technology, Shanghai 200237, China)

Abstract Although the rapid development of cloud data centers has brought very powerful computing power, the energy consumption problem has become increasingly serious. In order to reduce the energy consumption of physical servers in cloud data centers, firstly the virtual machine placement problem is modeled by reinforcement learning. Then, the Q-Learning(λ) algorithm is optimized from two aspects: state aggregation and time reliability. Finally, the virtual machine placement problem is simulated through cloud simulation platform CloudSim and actual data. The simulation results show that the optimized Q-Learning(λ) algorithm can effectively reduce the energy consumption of the cloud data center compared with the Greedy algorithm, PSO algorithm and Q-Learning algorithm, and can ensure better results for different numbers of virtual machine placement requests. The proposed algorithm has strong practical value.

Keywords Cloud computing, Virtual machine placement, Reinforcement learning, Energy consumption optimization, Q-Learning(λ) algorithm

1 引言

随着云服务需求的日益增长,云数据中心作为一种物理服务器和网络设备的高性能计算机集合,其规模不断扩大,但这也造成了云数据中心面临着低效率、高成本和高能耗等问题。有研究表明单个数据中心消耗的电力相当于25000个家庭所消耗的电力[1]。另外,能源消耗不仅取决于硬件的数量,还取决于资源的低效利用。据统计,我国云数据中心的平均资源利用率只维持在10%左右,同时云服务器的按需服务形式使其在大部分时间都处于空闲状态,即便如此空闲服务器也会消耗满负荷状态下60%的能耗[2]。由此可见,服务器利用率过低是造成大量能源浪费的主因,同时也间接增加了CO_2的排放量。因此,如何在保证服务质量的基础上提高云数据中心服务器的资源利用率,以降低服务成本和能源损耗,是云服务提供商极为重视的一个问题。

通过虚拟化技术,云数据中心利用虚拟机单元将各类物理服务器上的CPU、内存、存储和网络等进行统一整合,形成对外提供服务的虚拟机资源池,其典型应用之一就是基础设施即服务(Infrastructure as a Service, IaaS),例如Amazon EC2和Google Compute Engine。在IaaS中,云数据中心根据租户请求对虚拟机资源进行灵活放置,以优化云数据中心的资源配置。同时由于不同虚拟机与物理机之间的映射关系会造成不同的资源利用率,因此云数据中心需要在保证SLA(Service-Level Agreement)的同时,利用资源分配算法在软件层面上优化虚拟机放置策略,使其既可以满足应用的性能需求,又能提高资源利用率并减少能源消耗,从而降低云数据中

到稿日期:2018-08-28 返修日期:2018-11-26 本文受国家自然科学基金面上项目(61472139),华东理工大学2017年教育教学规律与方法研究项目(ZH1726107)资助。

卢海峰(1993-),男,博士生,主要研究方向为云计算和强化学习;顾春华(1970-),男,博士,教授,博士生导师,主要研究方向为物联网、云计算、软件工程,E-mail:chgu@ecust.edu.cn(通信作者);罗 飞(1978-),男,博士,副教授,主要研究方向为分布式计算;丁炜超(1989-),男,博士,讲师,主要研究方向为云资源调度、分布式计算、多目标优化等;袁 野(1995-),男,硕士,主要研究方向为云计算;任 强(1993-),男,硕士生,主要研究方向为云计算。

心的运营成本。目前云计算中的资源分配方案主要分为两种:细粒度资源分配和粗粒度资源分配[3]。细粒度资源分配是指通过调整单一物理机上多个虚拟机的具体资源配置,使不同虚拟机之间的资源分配配额随着用户的需求而改变,以保证各虚拟机的服务质量;粗粒度资源分配是指通过建立所有虚拟机与物理机之间的映射关系,确保多个虚拟机共享同一个物理机资源,提高应用负载。因此,细粒度资源分配是针对个体物理机进行优化,相比较于粗粒度资源分配的整体优化存在一定程度的局限性。

由于云数据中心的物理机和虚拟机数量较多,针对虚拟机放置(Virtual Machine Placement,VMP)问题找到唯一最优解的计算复杂度过高,因此为了避免较大的时间开销通常采用一些启发式算法来解决该问题。启发式算法具有实现简单、收敛速度快等优点,但其容易陷入局部最小值,且不能达到整体放置效果最优。相对地,元启发式算法能够较好地寻找全局最优解,但是算法参数过多,计算结果的复用性差,不能快速有效地进行参数调优。针对以上缺陷,本文利用强化学习算法来解决 VMP 问题,该类算法具有自学习和自适应等特征,需要提供的参数很少,有较好的全局搜索能力;同时该算法遵循马尔可夫性质,即模型的下个状态只与当前状态信息有关,而与更早之前的状态无关,因此其每次迭代的计算量很小,在一定程度上降低了时间复杂度。

本文主要针对云数据中心的粗粒度资源分配进行研究,将数据中心的能耗最小化作为优化目标,利用强化学习算法自动寻找虚拟机放置的最优策略,以实现节能减排。

本文第 2 节介绍 VMP 问题和强化学习的研究现状;第 3 节介绍强化学习的基础理论以及 Q-Learning(λ)算法的计算流程;第 4 节介绍基于强化学习的 VMP 问题建模方法,并结合实际问题对维数灾和时间信度分配问题进行改进;第 5 节利用仿真实验对各类算法在 VMP 问题中的能耗进行比较和分析;最后对基于强化学习的 VMP 问题进行总结。

2 相关工作

VMP 问题是一个 NP-Hard 的大规模组合优化问题,目前国内外学者主要采用一些启发式或元启发式算法进行求解,常见的启发式算法包括贪心算法、首次适应算法和最佳适应算法等,元启发式算法则包括遗传算法、粒子群算法和蚁群算法等。文献[4]对比了 8 种启发式算法分别在不同应用场景下求解最小化能耗的性能表现。文献[5]基于 BFD(Best Fit Decrease)算法,提出了改进的最佳适应降序算法(Modified Best Fit Decrease,MBFD)来进行虚拟机放置。文献[6]使用蚁群算法实现了一种服务器负载均衡方法,但由于每只蚂蚁仅对信息素进行局部更新,导致其收敛速度过慢。文献[7]通过拍卖模型将动态虚拟机分配问题描述为多种资源类型的整数规划问题,利用贪婪算法为优先级高的用户提供虚拟机以及费用账单,从而提高云提供商的总收入。文献[8]根据 GRASP 的启发式算法提出了一种 GraspCC-fed 方法,该方法通过对每个工作流的虚拟机数量进行最佳(或接近最佳)估计来确定基于并行云的虚拟集群的最佳配置。

强化学习是一种最接近自然界动物学习本质的工作范式,其通过建立智能体从环境到行为的映射关系,不断利用“试错-评价”过程与环境进行交互,观察采取某一动作后环境状态的变化情况,同时将即时奖励反馈给强化学习系统来实现行为决策的优化,随后不断重复该过程并使最终生成的策略获得最大的长期奖励[9]。目前强化学习在各领域的应用非常广泛。文献[10]提出了一种基于强化学习的选择算法,其可用于解决大规模机器之间的通信阻塞;文献[11]利用强化学习生成了一种基于集群的路由方案,其在一定程度上解决了认知无线网络中路由开销泛滥的问题;文献[12]提出了一种基于强化学习的集群调度框架,该框架是由一个学习智能体和几个调度智能体组成的多智能体结构。当发生任务调度时,调度智能体将实际调度产生的奖励值转发给学习智能体,随后学习智能体更新全局表中对应的值,并将更新后的全局表共享给调度智能体,最后调度程序根据更新的全局表进行决策。

3 强化学习

3.1 MDP 模型

在强化学习中,马尔可夫决策过程(Markov Decision Process,MDP)是对完全可观测的环境进行描述的,其可以定义为一个五元组(S,A,P,R,γ),其中 S 表示状态空间,A 表示动作空间,P 表示状态转移概率矩阵,R 表示奖励函数,$\gamma\in(0,1)$表示折扣因子。假设 $P_{ss'}(a)$表示智能体 agent 在时间步 t 由于采取动作 a 而使环境从状态 s 转移到状态 s'的转移概率,计算则公式为:

$$P_{ss'}(a)=P[S_{t+1}=s' \mid S_t=s,A_t=a] \tag{1}$$

其中,$S_t\in S$ 表示 agent 在时间步 t 接收的状态,$S_{t+1}\in S$ 为采取动作后接收的状态,对应采取的动作为 $a\in A$。在时间步 t 能获得的奖励期望为:

$$R_s=E[R_{t+1} \mid S_t=s] \tag{2}$$

时间步 t 开始,之后的长期奖励 G_t 的公式为:

$$G_t=R_{t+1}+\gamma R_{t+2}+\cdots=\sum_{k=0}^{\infty}\gamma^k R_{t+k+1} \tag{3}$$

其中,折扣因子 γ 用于表示未来奖励在当前时间步的价值比例。γ 越接近于 0,则价值越偏向于考虑短期利益;γ 越接近于 1,则价值越偏向于考虑长期利益。

3.2 Q-Learning(λ)算法

Q-Learning 属于一种无模型强化学习方法,该方法由于不知道环境的状态转移函数和奖励函数,因此无法直接评估策略质量,只能通过采样的方式与环境进行交互学习,随后根据观察到的转移状态和奖励反馈值来优化策略,最终得到累计奖励最大化的最优策略。Q-Learning 算法充分利用了 MDP 性质,能够在执行每一步策略后立即对值函数进行更新,保证了算法具有较高的学习效率[13]。同时 Q-Learning 算法采用了异策略(off-policy)方法,即策略评估时使用的是 ε-greedy 策略,策略执行时使用的是原始策略。异策略方法具有很好的通用性,并且保证了数据的全面性和所有行为的覆

盖性，因此不会收敛到局部最优。Q-Learning 算法对 MDP 中行为价值函数的迭代计算公式为：

$$Q(s,a)=Q(s,a)+\alpha(r+\gamma\max_{a'}Q(s',a')-Q(s,a)) \quad (4)$$

其中，s'表示状态 s 在动作 a 作用下的下一状态，而 a'为状态 s'后所有可能的动作；γ 为价值累积过程中的折扣系数，决定了未来回报相对于当前回报的重要程度；α 为学习速率，该值越大，则历史训练结果的影响就越小。

虽然 Q-Learning 算法具有较好的收敛速度，但是其在每次采取动作并获得即时奖励后只对前一步的 $Q(s,a)$进行更新。相比之下 Q-Learning(λ)算法是通过额外维护一张 E 表来存储在路径中所经历的每一步，保证在每次更新时对之前经历的所有步的 $Q(s,a)$进行更新，因此 Q-Learning(λ)算法具有更高的收敛速度，其中 λ 在[0,1]之间取值，当 λ=1 时表示更新获取到奖励值的前所有步。Q-Learning(λ)算法的伪代码如算法 1 所示。

算法 1　Q-Learning(λ)算法

Input：Requested Discount rate γ，Learning rate α，Step number λ
Output：Q table
1. Initialize Q(s,a)=0，and s∈S，a∈A(s)
2. While Iteration is not terminal
3.　Initialize E(s,a)=0，and s∈S，a∈A(s)
4.　Initialize state s and action a
5.　For i=1 to A. size() do
6.　　Take action a，get immediate reward r and next state s′
7.　　According to the strategy generated in the Q table (e. g ε-greedy)，action a′ is taken from state s′
8.　　ε←r+γ $\max_{a'}$Q(s′,a′)−Q(s,a)
9.　　E(s,a)←E(s,a)+1
10.　　For all s∈S，a∈A(s) do
11.　　Q(s,a)←Q(s,a)+αεE(s,a)
12.　　E(s,a)←γλE(s,a)
13.　　EndFor
14.　s←s′；a←a′
15.　EndFor
16. EndWhile
17. Return Q table

4　基于强化学习的 VMP 问题建模

强化学习虽然能够有效解决不确定环境中的决策问题，利用环境反馈的奖励值和状态信息来不断优化策略参数，而不需要人为构建推理模型，但是其具有的试错搜索和延迟回报等特点也带来了维数灾难和时间信度分配问题。其中维数灾难是指由于状态和动作的维数过高，导致 agent 没有足够的能力和资源在有限的时间和空间内找到一种合理的策略；时间信度分配问题是由于环境的反馈信息比较稀疏并且有较大的延时性，因此在 agent 执行多次动作并收到一个奖励后再决定先前动作的奖励分配方案是比较困难的。针对数量巨大的物理机和虚拟机而言，基于强化学习的 VMP 模型需要采取措施来避免维数灾难和时间信度分配问题，本文主要通过状态聚合以及改进 CloudSim 执行流程来解决上述问题。

4.1　VMP 模型

本文将云数据中心的虚拟化资源池用(V,H,HV)表示，其中 $V=\{v_1,v_2,\cdots,v_m\}$表示含有 m 个虚拟机的集合，$H=\{h_1,h_2,\cdots,h_n\}$表示具有 n 个物理主机的集合；HV 表示虚拟机与物理机之间的映射关系，当某一物理机的资源满足虚拟机运行所需资源时，则该虚拟机能成功放置到物理机上，其关系可以表示为 $HV=\{\langle v_i,h_j\rangle \mid v_i\in V,h_j\in H\}$。基于上述描述，下面分别从状态空间、动作空间和奖励函数 3 个方面对基于强化学习的 VMP 问题进行建模，使其能够形成通用有效的强化学习模型。

4.1.1　状态空间

在文献[14]中通过长达 3 个月的监控实验总结出主机的功耗与其 CPU 利用率的相关系数高达 0.990 667，可见物理机的能耗与其 CPU 利用率呈线性关系，因此为了优化物理机集群的总能耗，将第 i 个物理机的 CPU 利用率定义为 s_i，那么在时间步 t 时物理机集群的状态空间为 $S_t=(s_1,s_2,\cdots,s_i,\cdots,s_n)$，其中 n 为物理机的数量。

4.1.2　动作空间

为了将虚拟机放置到合适的物理机上，在强化学习中规定动作空间与物理机集合呈对应关系，$(0/1)_i^j$ 用表示第 i 个虚拟机是否放置到第 j 个物理机上，例如动作空间 $A_i=(0,0,1,\cdots,0)$说明第 i 个虚拟机根据策略被放置在第 3 个物理机上。因此对于包含 n 个物理机的集群，其动作空间为 $A=(h_1,h_2,\cdots,h_n)$。

4.1.3　即时奖励函数

因为本文是将物理机集群的能耗最小化作为优化目标，所以考虑每次状态更新时的即时奖励为物理机集群的总能耗。同时针对物理机的 CPU 负载状态，本文定义集群中第 i 台物理机的功率模型公式如下[15]：

$$P_i(u)=\begin{cases}K*P_i^{\max}+(1-K)*P_i^{\max}*u, & \text{if } u>0\\ 0, & \text{otherwise}\end{cases} \quad (5)$$

其中，K 表示空闲物理机消耗能耗的百分比，$P_i^{\max}$ 表示集群中第 i 台物理机在满负载时所消耗的能量，u 为 CPU 利用率。另外，物理机负载是随时间而不断变化的，现假设 $u(t)$为单位时间内物理机 CPU 利用率的变化函数，则从 t_0 时刻开始，一个物理机在单位时间 t 内的能耗为：

$$E_i(t_0)=\int_{t_0}^{t_0+t}P_i(u(t))\mathrm{d}t \quad (6)$$

为了解决异构物理机功率差异在每一个时间步所造成的奖励值偏差，利用每一个物理机的前一时间步能耗与当前时间步能耗之比来保证归一化，最后总能耗为所有物理机能耗之比的和，其公式为：

$$E(t_0)=\sum_{i=1}^{n}\left(\frac{E_i(t_0-t)}{E_i(t_0)}\right) \quad (7)$$

其中，n 为集群中物理机的数量。同时为了符合 Q-Learning(λ)算法的最大化奖励选择策略，$\frac{E_i(t_0-t)}{E_i(t_0)}$表示用前一单位时间内的总能耗除以当前单位时间内的总能耗。如果强化学习生成的策略有效降低了物理机能耗，则有 $E_i(t_0)\leqslant E_i(t_0-t)$，即$\frac{E_i(t_0-t)}{E_i(t_0)}\geqslant 1$，相反则有$\frac{E_i(t_0-t)}{E_i(t_0)}<1$。因此，物理机总

能耗之比的和 $E(t_0)$ 越大，说明当前单位时间内物理机集群产生的能耗越少，Q-Learning(λ)算法生成的策略越有效，随后根据该算法的特性会逐步增大长期奖励，最终生成总能耗最小的策略[16]。

4.2 状态聚合

随着 VMP 问题中物理机数量和迭代次数的增加，其状态空间的维数会呈指数增长，最终将由于维度过高而对收敛速度和优化值产生负面影响。因此为了加快算法学习过程，本文利用状态聚合的方法，该方法可以在一定程度上解决该问题，其公式为：

$$E=\begin{cases} S_k, & \text{if } s_i\in[10k,10(k+1)] \text{ and } k\in[0,9] \\ & \text{and } i\in[1,n] \\ S_l, & \text{if } s_i=100 \end{cases} \tag{8}$$

其中，s_i 为一个物理机的 CPU 利用率，k 为 0 到 9 之间的整数，n 为集群中物理机的数量。式(9)利用 S_k 表示 CPU 利用率在[0,100)之间的状态空间，并根据 $k\in[0,9]$将利用率按 10%为单位离散化成 10 个状态；利用 S_l 表示 CPU 利用率为 100%时的状态。随后根据 S_k 的 10 个状态将其映射成 0 到 9 之间的 10 个数字，同时设置 $S_l=9$。最终状态空间中所有物理机的 CPU 利用率都能用一串离散化后的数字表示，例如 10 个物理机的 CPU 利用率状态可以表示为 $s=$(0015409403)，这极大简化减少了 VMP 问题中状态空间的维度和计算量[17]。

4.3 CloudSim 执行流程的改进

针对时间信度分配问题，本文主要通过改进仿真程序的执行流程来降低奖励延时性。本文主要采用 CloudSim 来进行 VMP 问题的仿真实验，其通常采用事件机制来进行相关流程的模拟。CloudSim 的事件执行流程为：事件被存放在事件队列 future 中，模拟程序开始后，不断地检查 future 队列，取出事件并处理(往往伴随新事件的入队)，随后更新时间轴和云数据中心的主机状态，直到 future 为空则开始执行模拟程序的终止操作，否则不停触发事件并更新时间轴。由于真实环境中云数据中心的运转是连续的，每个物理机和虚拟机的负载都实时变化，因此云数据中心会根据当前时间的需求和负载来进行虚拟机的创建和放置[18]。而在 CloudSim 中时间轴是依据事件来更新，为方便模拟，在程序开始时对虚拟机进行批量创建和统一放置，这将造成 VMP 问题的模拟是批处理而不是真实环境下的流处理。同时该机制对强化学习中的时间信度分配问题是非常不利的，这是因为虚拟机的批处理放置造成了物理机集群总能耗的巨大延时，使得动作的即时奖励分配方案产生了很高的误差[19]。为了有效避免该问题，本文对 CloudSim 中的 DatacenterBroker 和 Datacenter 两个类进行执行流程改进，使其虚拟机的创建和放置更符合实际环境中的流处理过程，降低强化学习算法在批处理过程中因奖励更新和动作选择所造成的不合理结果，避免生成的最优策略不能收敛到最小值。改进后的 CloudSim 执行流程如图 1 所示，结合强化学习的 VMP 算法的伪代码如算法 2 所示。

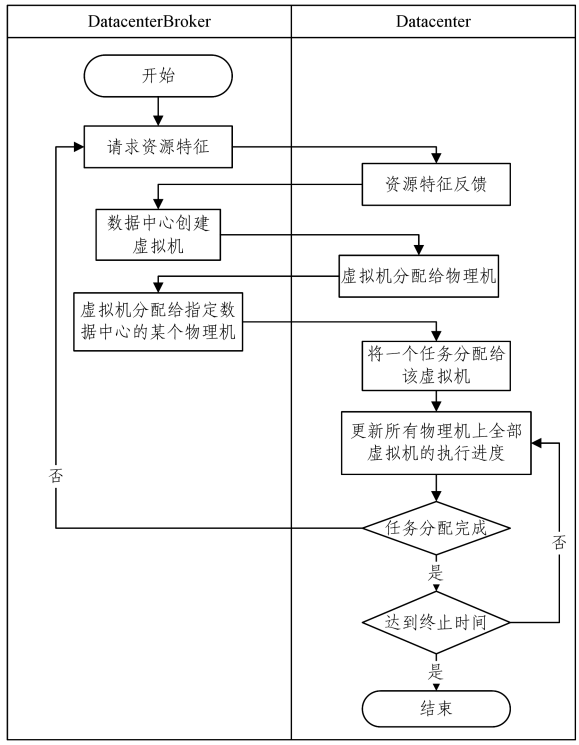

图 1 改进后的 CloudSim 执行流程图

Fig. 1 Execution flowchart of improved CloudSim

算法 2 结合强化学习的 VMP 算法

Input: Requested all physical machines HostList, VM that needs to be assigned

Output: The destination of VM

1. Get CPU utilizations of all physical machines CPUList
2. Initialize state aggregation list convertList, total energy consumption value totalPower
3. For i=1 to |CPUList| do
4. Use state aggregation to convert the CPU utilization of a physical machine to a number between 1－9
5. Add the conversion result to the convertList
6. The energy consumption hostPower of the physical machine is calculated by CPU utilization and energy consumption model PowerModelLinear.
7. totalPower＋＝hostPower
8. EndFor
9. Pass state convertList and reward value totalPower into the algorithm Q-Learning(λ) for calculation and selecting action PM
10. If VM can be placed on PM then
11. Get mapping 〈VM,PM〉
12. Else
13. Pass penalty value into the algorithm Q-Learning(λ) to recalculate and select action
14. Re-execute line 10 of code
15. EndIf
16. Return 〈VM,PM〉

5 仿真实验与结果分析

5.1 仿真环境

本文采用 CloudSim4.0 对 VMP 问题进行仿真实验，通

过比较强化学习算法和其他经典算法的收敛情况来检验对应策略在数据中心总能耗最小化方面的有效性。仿真实验模拟了一个包括 300 台异构物理主机的数据中心，其中物理机按照 HP ProLiant ML110 G5，HP ProLiant DL360 G7 和 HP ProLiant DL360 Gen9 3 种类型平均分布，具体配置参数如表 1 所列。同时为了保证结果的真实性，本文选取 Planet Lab 提供的 200 台真实虚拟机在一天内的 CPU 利用率数据，并且实验中虚拟机的配置信息参考 Amazon EC2 的实例类型[20]，虚拟机数量按照几种配置信息平均分配，其详细配置如表 2 所列。

表 1　物理机配置

Table 1　Physical machine configuration

物理机	HP ProLiant ML110 G5	HP ProLiant DL360 G7	HP ProLiant DL360 Gen9
处理器/Mips	2660	3067	2300
核心数	2	12	36
内存/GB	4	16	64
带宽/(GB/s)	1	1	1

表 2　虚拟机配置

Table 2　Virtual machine configuration

虚拟机	Large	Medium	Small	Micno	Nano
处理器/Mips	2500	2000	1000	500	250
内存/GB	2048	2048	1024	1024	512
带宽/(GB/s)	1	1	1	1	1

本文将分别比较启发式算法 Greedy、元启发式算法 PSO (Particle Swarm Optimization)以及强化学习算法 Q-Learning 和 Q-Learning(λ)在 VMP 问题中的能耗，其中 PSO 算法、Q-Learning 算法和 Q-Learning(λ)算法的参数设置如表 3 所列。

表 3　各算法的参数设置

Table 3　Parameter setting of each algorithm

算法	PSO	Q-Learning	Q-Learning(λ)
迭代次数 Iteration	100	100	100
学习速率 α		0.8	0.8
折扣系数 γ		0.9	0.9
更新步系数 λ			1
粒子种群 N	300		
惯性权重 w	0.1		
加速因子 c_1，c_2	2		

5.2　实验结果分析

在强化学习中，折扣率 γ 的取值可以有效影响算法的收敛结果，当 $r=0$ 时，Q-Learning(λ)算法只考虑即时奖励，而随着 γ 值的不断增长，该算法中未来奖励的权重增大。如图 2 所示，本文在相同模拟环境下对不同折扣率 γ 值进行实验，可以发现当 $r=0.9$ 时数据中心的总能耗最小，因此后续实验中强化学习算法的折扣率统一设置为 0.9。

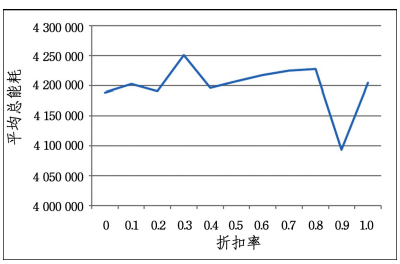

图 2　不同折扣率下 Q-Learning(λ)算法的平均总能耗图

Fig. 2　Average total energy consumption diagram of Q-Learning(λ) algorithm under different discount rates

针对强化学习中的状态聚合和时间信度问题，本文设计了两个实验来分别比较模型在改进前后的性能差异。图 3 是针对 VMP 问题 Q-Learning(λ)算法是否进行状态聚合的比较结果。由图 3 可知，在相同模拟环境下，经过状态聚合的算法比未经过状态聚合的算法有更快的收敛速度，这对解决大规模以及实时性要求较高的问题有很强的实用性。图 4 是针对 VMP 模型是否进行时间信度改进的比较结果，由图可知虽然在收敛速度上两者相差不大，但是改进时间信度后，模型在收敛值方面要优于改进前的模型。

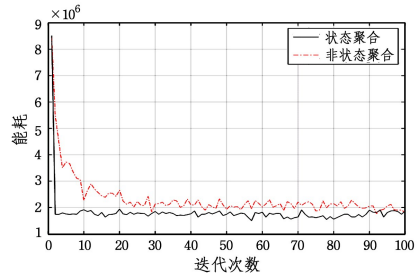

图 3　状态聚合与收敛结果的关系

Fig. 3　Relation of state aggregation and convergence results

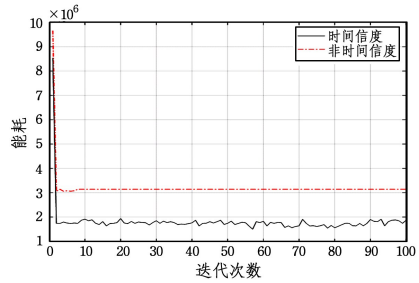

图 4　时间信度与收敛结果的关系

Fig. 4　Relation of time reliability and convergence results

图 5 是各类算法在 VMP 问题中的能耗比较，由图可知 Q-Learning(λ)算法比 PSO 算法和 Q-Learning 算法具有更快的收敛性，同时该算法收敛的最优值要远小于 Greedy 算法和 PSO 算法的值，因此在 VMP 问题中 Q-Learning(λ)算法既保证了收敛速度又具有更好的最优值。

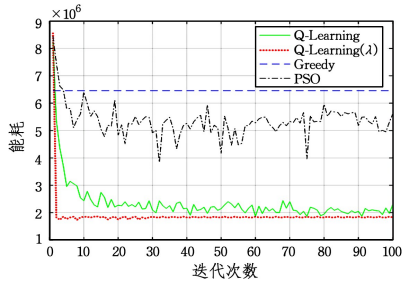

图 5　各类算法在 VMP 问题中的能耗比较

Fig. 5　Energy consumption comparison of various algorithms in VMP problem

表 4 是 VMP 问题中的能耗数据以每 10 次迭代为单位所求的平均值。由表 4 可知，当各类算法最终收敛后，Q-Learning(λ)算法的平均收敛值要比 Greedy 算法大约降低了 71.7%，比 PSO 算法大约降低了 65.4%。

表 4 各算法的平均收敛值

Table 4 Average convergence values of algorithms

	Q-Learning	Q-Learning(λ)	Greedy	PSO
1	3.88×10^6	2.49×10^6	6.46×10^6	6.31×10^6
2	2.42×10^6	1.82×10^6	6.46×10^6	5.38×10^6
3	2.26×10^6	1.82×10^6	6.46×10^6	5.23×10^6
4	2.22×10^6	1.83×10^6	6.46×10^6	4.97×10^6
5	2.12×10^6	1.83×10^6	6.46×10^6	5.17×10^6
6	2.20×10^6	1.84×10^6	6.46×10^6	5.01×10^6
7	2.16×10^6	1.83×10^6	6.46×10^6	5.27×10^6
8	2.03×10^6	1.83×10^6	6.46×10^6	5.32×10^6
9	2.06×10^6	1.84×10^6	6.46×10^6	5.59×10^6
10	2.07×10^6	1.83×10^6	6.46×10^6	5.29×10^6

图 6 是保持物理机数量不变，各类算法在不同虚拟机数量下的能耗变化。由图 6 可知，随着虚拟机数量的不断增加，Q-Learning(λ)算法相较于其他算法所增加的能耗更小，因此该算法具有更好的稳定性。表 5 是各类算法在不同虚拟机数量下的具体能耗值。由表 5 可知，随着虚拟机数量的增加，Q-Learning(λ)算法的能耗变化最小，当虚拟机数量为 300 时该算法比 Greedy 算法的能耗值大约低 71.5%，比 PSO 算法的能耗值低 63.3%，比 Q-Learning 算法的能耗值约低16.4%。

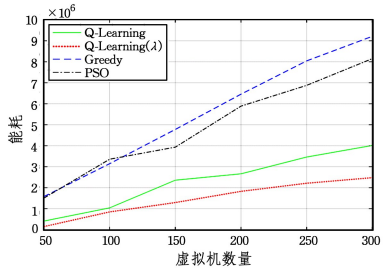

图 6 各类算法在不同虚拟机数量下的能耗比较

Fig. 6 Energy consumption comparison of various algorithms under different virtual machine numbers

表 5 各算法在不同虚拟机数量下的能耗值

Table 5 Energy consumption values of each algorithm under different virtual machine number

	Q-Learning	Q-Learning(λ)	Greedy	PSO
50	0.51×10^6	0.11×10^6	1.56×10^6	1.54×10^6
100	1.04×10^6	0.92×10^6	3.06×10^6	3.18×10^6
150	2.36×10^6	1.12×10^6	4.89×10^6	3.98×10^6
200	2.74×10^6	1.91×10^6	6.39×10^6	5.97×10^6
250	3.48×10^6	2.08×10^6	8.01×10^6	6.91×10^6
300	4.03×10^6	2.42×10^6	9.11×10^6	8.06×10^6

通过比较图 5 和图 6 可知，Q-Learning(λ)算法在收敛到最小值时所需的迭代次数要小于 PSO 算法和 Q-Learning 算法，同时该算法计算出的总能耗最优值也远小于 Greedy 算法和 PSO 算法，因此总体来说强化学习算法在收敛速度和最优值上都要优于启发式算法和元启发式算法。

本文采用了 HP ProLiant ML110 G5，HP ProLiant DL360 G7 和 HP ProLiant DL360 Gen9 等 3 种配置的物理机，参考 SPEC(Standard Performance Evaluation Corporation)的数据可知，各类型物理机的平均性能功耗比分别为 431，2979 和 10118，而该值越大表明物理机在相同性能下的能耗越少。表 6 是各算法根据 CPU 利用率对不同配置物理机进行分类的详细数据。由表 6 可知，当 CPU 利用率为 80%～100%时，Q-Learning(λ)算法是所有算法中使用 G5 物理机数量最少的，同时也是使用 Gen9 物理机数量最多的；当 CPU 利用率为 0～20%时，Q-Learning(λ)算法在所有算法中使用的 G5 物理机数量最多，同时其使用的 Gen9 物理机数量最少。根据平均性能功耗比的定义可知，Gen9 物理机的数量越多，则云数据中心能够提供的性能越高，同时产生的能耗越少；相反 G5 物理机的数量越多，则云数据中心能够提供的性能越低，同时产生的能耗越多。因此在以能耗为标准的 VMP 问题中，根据 Q-Learning(λ)算法生成的放置策略符合实际情况并且所产生的总能耗最小，同理 Q-Learning 算法次之，Greedy 算法最差，PSO 算法要稍优于 Greedy 算法。

表 6 在不同 CPU 利用率下各算法的物理机数量分布

Table 6 Quantity distribution of physical machines of each algorithm under different CPU utilization

	Q-Learning	Q-Learning(λ)	Greedy	PSO
[80%,100%] G5	39	1	72	50
[80%,100%] G7	8	4	9	14
[80%,100%] Gen9	8	19	0	4
(20%,80%) G5	26	0	24	20
(20%,80%) G7	32	22	35	29
(20%,80%) Gen9	29	26	38	24
[0,20%] G5	35	99	4	30
[0,20%] G7	60	74	56	57
[0,20%] Gen9	63	55	62	72
[80%,100%] 总数	55	24	81	68
(20%,80%) 总数	87	48	97	73
[0%,20%] 总数	158	228	122	159
物理机数量	300	300	300	300

结束语 本文以云数据中心内所有物理机产生的总能耗为奖励值，将物理机集合的 CPU 利用率和索引分别作为状态空间和动作空间，结合 VMP 问题并针对维数灾难和时间信度分配两个方面的问题对 Q-Learning(λ)算法进行优化，最终利用 CloudSim 仿真程序对其进行建模和实验。在仿真实验中，本文首先设计状态聚合实验和时间信度实验，证明了 Q-Learning(λ)算法的优化有效性；随后通过比较 4 种算法在 VMP 问题中的收敛速度和最优值证明了 Q-Learning(λ)算法的性能；最后分析了 4 种算法在不同类型物理机之间的数量分布，证明了根据 Q-Learning(λ)算法生成放置策略的实用性和科学性。

参 考 文 献

[1] GAI K, QIU M, ZHAO H, et al. Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing [J]. Journal of Network & Computer Applications, 2016, 59(C): 46-54.

[2] HAMEED A, KHOSHKBARFOROUSHHA A, RANJAN R, et al. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems[J]. Computing, 2016, 98(7): 751-774.

[3] GAI K, QIU M, ZHAO H. Cost-Aware Multimedia Data Allocation for Heterogeneous Memory Using Genetic Algorithm in Cloud Computing[J]. IEEE Transactions on Cloud Computing, 2016, PP(99): 1-1.

[4] LINDBERG P, LEINGANG J, LYSAKER D, et al. Comparison

and analysis of eight scheduling heuristics for the optimization of energy consumption and makespan in large-scale distributed systems[J]. Journal of Supercomputing，2012，59(1)：323-360.

[5] BELOGLAZOV A，ABAWAJY J，BUYYA R. Energy-aware resource alocation heuristics for eficient management of data centers for cloud computing[J]. Future Generation Computer Systems，2012，28(5)：755-768.

[6] GAO Y，GUAN H，QI Z，et al. A multi-objective ant colony system algorithm for virtual machine placement in cloud computing [J]. Journal of Computer & System Sciences，2013，79(8)：1230-1242.

[7] NEJAD M M，MASHAYEKHY L，GROSU D. Truthful Greedy Mechanisms for Dynamic Virtual Machine Provisioning and Allocation in Clouds[J]. IEEE Transactions on Parallel & Distributed Systems，2015，26(2)：594-603.

[8] COUTINHO R D C，FROTA Y，OLIVEIRA D D. Optimizing virtual machine allocation for parallel scientific workflows in federated clouds [J]. Future Generation Computer Systems，2015，46(C)：51-68.

[9] MAO H，ALIZADEH M，MENACHE I，et al. Resource Management with Deep Reinforcement Learning[C]// ACM Workshop on Hot Topics in Networks. ACM，2016：50-56.

[10] RUPASINGHE N，GÜVENÇ I. Reinforcement learning for licensed-assisted access of LTE in the unlicensed spectrum[C]// Wireless Communications and Networking Conference. IEEE，2015：1279-1284.

[11] SALEEM Y，YAU K L A，MOHAMAD H，et al. Clustering and Reinforcement-Learning-Based Routing for Cognitive Radio Networks[J]. IEEE Wireless Communications，2017，24(4)：146-151.

[12] MORADI M. A centralized reinforcement learning method for multi-agent job scheduling in Grid[C]// International Conference on Computer and Knowledge Engineering. Mashhad：IEEE，2017.

[13] BOTVINICK M，WEINSTEIN A，SOLWAY A，et al. Reinforcement learning，efficient coding，and the statistics of natural tasks[J]. Current Opinion in Behavioral Sciences，2015，5：71-77.

[14] ZHENG Q，LI R，LI X，et al. A Multi-Objective Biogeography-Based Optimization for Virtual Machine Placement[C]// 2015 15th IEEE/ACM International Symposium on Cluster，Cloud and Grid Computing. Shenzhen：IEEE，2015：687-696.

[15] YOU C，HUANG K，CHAE H，et al. Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading[J]. IEEE Transactions on Wireless Communications，2017，16(3)：1397-1411.

[16] GAI K，QIU M. Optimal resource allocation using reinforcement learning for IoT content-centric services [J]. Applied Soft Computing，2018，70：12-21.

[17] KUMAR M，YADAV A K，KHATRI P，et al. Global host allocation policy for virtual machine in cloud computing[J]. International Journal of Information Technology，2018，10(3)：279-287.

[18] SANTRA S，MALI K. A new approach to survey on load balancing in VM in cloud computing：Using CloudSim[C]// International Conference on Computer，Communication and Control. IEEE，2016：1-5.

[19] DUONG T，CHU Y J，NGUYEN T，et al. Virtual Machine Placement via Q-Learning with Function Approximation[C]// IEEE Global Communications Conference. San Diego：IEEE，2015：1-6.

[20] HABIB A，KHAN M I. Reinforcement learning based autonomic virtual machine management in clouds[C]// International Conference on Informatics，Electronics and Vision. Univ Dhaka：IEEE，2016：1083-1088.

[21] XU ZX，et al. Deep Reinforcement Learning with Sarsa and Q-Learning：A Hybrid Approach[J]. IEICE Transactions on Information and Systems，2018，E101d(9)：2315-2322.

[22] TENG L，BIN T，YUN A，et al. Parallel reinforcement learning：a framework and case study[J]. IEEE/CAA Journal of Automatica Sinica，2018，5(4)：827-835.

[23] NISHIYAMA R，YAMADA S. Reinforcement Learning with Multiple Actions[C]// Proceedings of the 3rd International Conference on Intelligent Technologies and Engineering Systems. New York：Springer2016：207-213.

[24] HOMEM T P D，PERICO D H，SANTOS P E，et al. Improving Reinforcement Learning Results with Qualitative Spatial Representation[C]// Brazilian Conference on Intelligent Systems. Brazil：IEEE，2017：151-156.

[25] DUAN Y，CHEN X，HOUTHOOFT R，et al. Benchmarking deep reinforcement learning for continuous control[C]// International Conference on International Conference on Machine Learning. New York：ACM，2016：1329-1338.

[26] LITTMAN M L. Reinforcement learning improves behaviour from evaluative feedback[J]. Nature，2015，521(7553)：445-451.

[27] THERRIEN A S，WOLPERT D M，BASTIAN A J. Effective reinforcement learning following cerebellar damage requires a balance between exploration and motor noise[J]. Brain，2016，139(1)：101-114.

[28] CUTLER M，WALSH T J，HOW J P. Real-World Reinforcement Learning via Multifidelity Simulators[J]. IEEE Transactions on Robotics，2017，31(3)：655-671.

[29] LEONG Y C，RADULESCU A，DANIEL R，et al. Dynamic Interaction between Reinforcement Learning and Attention in Multidimensional Environments[J]. Neuron，2017，93(2)：451-463.

[30] KIM B G，ZHANG Y，SCHAAR M V D，et al. Dynamic Pricing and Energy Consumption Scheduling With Reinforcement Learning[J]. IEEE Transactions on Smart Grid，2016，7(5)：2187-2198.

[31] XIONG R，CAO J，YU Q. Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle[J]. Applied Energy，2018，211：538-548.

[32] SAMBROOK T D，GOSLIN J. Principal Components Analysis of Reward Prediction Errors in a Reinforcement Learning Task [J]. Neuroimage，2016，124(Pt A)：276-286.

[33] CHEN H，LI X，ZHAO F. A Reinforcement Learning-Based Sleep Scheduling Algorithm for Desired Area Coverage in Solar-Powered Wireless Sensor Networks[J]. IEEE Sensors Journal，2016，16(8)：2763-2774.