计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2021.20200366
58(12):2783-2797,2021

基于深度强化学习的自适应虚拟机整合方法

余 显[1,2] 李振宇[1] 孙 胜[1,2] 张广兴[1] 刁祖龙[1] 谢高岗[1]
[1](中国科学院计算技术研究所 北京 100190)
[2](中国科学院大学 北京 100049)
(yuxian@ict.ac.cn)

Adaptive Virtual Machine Consolidation Method Based on Deep Reinforcement Learning

Yu Xian[1,2], Li Zhenyu[1], Sun Sheng[1,2], Zhang Guangxing[1], Diao Zulong[1], and Xie Gaogang[1]
[1](*Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)
[2](*University of Chinese Academy of Sciences, Beijing* 100049)

Abstract The problem of service quality optimization with energy consumption restriction has always been one of the big challenges for virtual machine (VM) resource management in data centers. Although existing work has reduced energy consumption and improved system service quality to a certain extent through VM consolidation technology, these methods are usually difficult to achieve long-term optimal management goals. Moreover, their performance is susceptible to the change of application scenarios, such that they are difficult to be replaced and will produce much management cost. In view of the problem that VM resource management in data center is hard to achieve long-term optimal energy efficiency and service quality, and also has poor flexibility in policy adjustment, this paper proposes an adaptive VM consolidation method based on deep reinforcement learning. This method builds an end-to-end decision-making model from data center system state to VM migration strategy through state tensor representation, deterministic action output, convolution neural network and weighted reward mechanism; It also designs an automatic state generation mechanism and an inverting gradient limitation mechanism to improve deep deterministic strategy gradient algorithm, speed up the convergence speed of VM migration decision-making model, and guarantee the approximately optimal management performance. Simulation experiment results based on real VM load data show that compared with popular VM consolidation methods in open source cloud platforms, this method can effectively reduce energy consumption and improve system service quality.

Key words data center; VM resource management; VM consolidation; reinforcement learning; deep deterministic policy gradient (DDPG)

摘 要 能耗限制的服务质量优化问题一直以来都是数据中心虚拟机资源管理所面临的巨大挑战之一.尽管现有的工作通过虚拟机整合技术一定程度上降低了能耗和提升了系统服务质量,但这些方法通常难以实现长期最优的管理目标,并且容易受到业务场景变化的影响,面临变更困难以及管理成本高等难

收稿日期:2020-06-02;**修回日期**:2021-01-27
基金项目:国家自然科学基金项目(61725206,U20A20180);中科院-奥地利合作项目(171111KYSB20200001)
This work was supported by the National Natural Science Foundation of China (61725206, U20A20180), and the CAS-Austria Project Plan (171111KYSB20200001).
通信作者:李振宇(zyli@ict.ac.cn)

题.针对数据中心虚拟机资源管理存在的能耗和服务质量长期最优难保证以及策略调整灵活性差的问题,提出了一种基于深度强化学习的自适应虚拟机整合方法(deep reinforcement learning-based adaptive virtual machine consolidation method, RA-VMC).该方法利用张量化状态表示、确定性动作输出、卷积神经网络和加权奖赏机制构建了从数据中心系统状态到虚拟机迁移策略的端到端决策模型;设计自动化状态生成机制和反向梯度限定机制以改进深度确定性策略梯度算法,加快虚拟机迁移决策模型的收敛速度并且保证近似最优的管理性能.基于真实虚拟机负载数据的仿真实验结果表明:与开源云平台中流行的虚拟机整合方法相比,该方法能够有效地降低能耗和提高系统的服务质量.

关键词 数据中心;虚拟机资源管理;虚拟机整合;强化学习;深度确定性策略梯度

中图法分类号 TP393

现代化数据中心采用“现收现付制”,并且通过广泛的网络连接支持各种类型的用户终端访问,提供用户弹性的计算、存储、网络等资源服务[1].但随着数据中心应用业务的日益繁荣和规模的不断扩大,如何实现低能耗和高服务质量的双重目标逐渐成为虚拟机(virtual machine, VM)资源管理所面临的巨大挑战之一.

目前,一种非常流行的方法是通过整合虚拟机资源来降低数据中心能耗、提升系统服务质量[2-16].这种方法通过实时感知虚拟机的动态资源需求,利用虚拟机迁移技术,避免主机过载所导致的性能下降和减少因主机资源利用率低而导致的能耗成本.考虑到虚拟机整合问题的复杂性,大部分现有工作[2-13]将其分解为主机过载检测、主机欠载检测、虚拟机选择和虚拟机重分配 4 个子问题,重点针对某个子问题进行研究.主机过载检测用于判断主机何时过载,是避免系统性能因资源不足而下降的关键;主机欠载检测用于发现资源利用率低的主机,通过关闭这些主机能有效节省主机的静态功耗;虚拟机选择主要解决过载主机上哪些虚拟机需要迁移的问题;虚拟机重分配又称之为虚拟机动态迁移,为过载主机上的待迁移虚拟机以及欠载主机上的所有虚拟机选择新的托管主机.

尽管这种解耦合式的虚拟机整合方法极大地降低了虚拟机资源管理的复杂度,但是由于这种方法只考虑了单一子问题的解决策略,没有考虑各种子问题策略之间的相互影响,从而难以实现最佳的能效和服务质量.此外,差异化的数据中心业务场景、动态时变的任务资源需求,以及多样化的服务目标要求数据中心管理员能够快速灵活地完成虚拟资源管理方法的动态调整.然而,大多数现有工作采用静态方法,不具备自动化学习能力,这使得数据中心管理员需要具备非常专业的领域知识,并耗费大量的人力和时间投入去针对每个子问题配置独立的最优策略,进而容易造成极大的成本浪费.故目前仍旧缺少一种灵活的、低成本和高服务质量的虚拟机资源整合方法.

针对上述问题,本文提出了一种基于深度强化学习[17]的自适应虚拟机整合方法(deep reinforcement learning-based adaptive virtual machine consolidation method, RA-VMC).具体而言,RA-VMC 利用张量化状态表示、确定性动作输出、卷积神经网络和加权奖赏机制构建了从数据中心系统资源状态到虚拟机迁移方案的端到端决策模型.该模型能够根据实时的主机资源占用状态、虚拟机资源需求状态以及虚拟机的主机位置分布动态地制定虚拟机迁移策略.RA-VMC 提出改进深度确定性策略梯度(deep deterministic policy gradient, DDPG)算法[18],设计自动化状态生成机制和反向梯度限定机制,加速模型的稳定收敛并有效保障收敛性能,从而获得能耗和服务质量的最优性能.得益于强化学习本身强大的自主学习能力,RA-VMC 还能够支持在不同场景下自动化完成目标优化,极大地降低人力优化的运行管理成本.

本文的主要贡献有 3 个方面:

1) 提出了一种从数据中心系统状态到虚拟机迁移策略的端到端决策模型,设计张量化状态表示、确定性动作输出、卷积神经网络和加权奖赏机制;

2) 提出了一种改进深度确定性策略梯度算法,设计自动化状态生成机制和反向梯度限定机制,加快收敛速度并保障收敛性能;

3) 基于真实虚拟机负载数据集进行仿真实验,结果表明,相比于开源云平台中流行的虚拟机整合方法所提 RA-VMC 方法能够实现约 90%的服务质量提升.

1　相关工作

考虑到本工作引入强化学习来实现虚拟机的自适应整合,故此部分主要从虚拟机整合方法和强化学习在调度优化问题中的应用2个方面来介绍现有的研究工作.

1.1　虚拟机整合方法

考虑到虚拟机整合问题的复杂性,很多工作都采用启发式方法来实现虚拟机的动态资源管理.为了尽量降低决策时间对性能的影响,现有的开源云平台(例如 OpenStack[19] 和 CloudStack[20])大多采用最先匹配(first-fit, FF)或最先降序匹配(first-fit-decrease, FFD)算法进行虚拟机分配.Shen 等人[4]同样是采用 FFD 算法,并结合预测得到的虚拟机资源需求来完成虚拟机分配.Panigrahy 等人[5]定义主机剩余资源为当前状态和满资源占用状态之间的欧几里得距离,并将 FFD 算法扩展到多维资源场景.Chen 等人[16]在 Panigrahy 等人[5]工作的基础上将 FFD 算法扩展到了时间维度,能进一步提升数据中心的资源利用率,但该方法需要估计每个任务全周期内的资源需求情况,导致实际应用困难.Beloglazov 等人提出将虚拟机整合问题拆分为主机过载和主机欠载检测、虚拟机选择、虚拟机重分配4个子问题进行研究,并提出多种启发式策略[2],随后开发了集成于 OpenStack 的虚拟机整合插件[3],极大地简化了研究人员解决虚拟机资源动态管理问题的复杂度,但同时导致难以实现最佳的性能目标,并且增加了数据中心管理员的维护和管理成本.

为了尽可能地实现虚拟机资源管理的最优化目标,许多研究人员从理论角度研究虚拟机整合问题.针对数据中心虚拟机资源管理所涉及到的不同应用场景,部分研究人员[21-22]首先通过线性规划、混合非线性整数规划等理论模型对原问题建模,分析问题的最优解,然后为了降低问题求解的复杂度,采用贪婪算法或近似启发式算法确定虚拟机迁移方案.这些工作一定程度上提高了性能需求,但它们都只考虑当前时间下的虚拟机迁移,难以实现服务目标的长期最优化.针对这一问题,Han 等人[23]将虚拟机的动态资源管理问题转化为求解大规模 Markov 决策过程(Markov decision process, MDP),考虑了主机状态的多步时间下的转化关系,促进了虚拟机迁移决策的准确性;同样,Shen 等人[24]通过有限 Markov 决策过程模型来实现数据中心主机的长期负载均衡.然而,这种基于 MDP 的方法[23-24]要求知道所有状态相互间的转移概率,难以在实际动态变化的环境中获得,而且求解复杂度高;此外,这些工作的优化目标要么是提升数据中心资源利用率,要么旨在实现主机间负载均衡,与本工作研究的能耗和服务质量双重优化目标有所差异.Li 等人[10]通过将主机托管虚拟机后的状态变化关系转化为 Web 网页之间的链接关系,然后采用 PageRank[25]值来定义主机托管虚拟机的优先级.这种方法考虑了主机放置虚拟机后的变化状态,有利于提高数据中心资源占用率和降低能耗,但是其无法保障得到最佳的服务质量.

为了兼顾数据中心网络的有关性能,Huang 等人[26]将考虑网络成本的虚拟机整合问题建模成 M-convex 优化问题,然后通过求解 M-convex 来决定虚拟机迁移位置,具有较高的求解复杂度.Cui 等人[27]针对网络动态变化的数据中心场景,通过形式化建模分析了该场景下虚拟机迁移问题属于 NP-hard 问题,然后提出了一种近似算法实现了多项式时间复杂度和更高的虚拟机吞吐量.但此工作关注的是软件交换机中的虚拟机调度问题,与本工作研究的主机上虚拟机调度不同,没有考虑主机过载对性能的影响.

此外,部分工作[5,8,11-13]通过一定的预测算法来预测虚拟机未来的资源需求,然后通过启发式算法或者规划算法确定虚拟机的迁移位置.考虑虚拟机未来需求的方式有助于更好地做出虚拟机迁移决策,但是其性能受限于预测算法的准确性.

1.2　强化学习在调度优化问题中的相关应用

针对真实网络环境和定制化的用户视频体验目标条件下难以实现最佳性能问题,Mao 等人[28]提出了一种基于强化学习的自适应码率算法,能够为视频块动态选择合适的传输码率.为了解决软件定义网络场景下的服务编排问题,Zhang 等人[29]引入 Q-Learning 模型来学习最佳的服务放置策略,能够有效降低用户的累积服务成本.Chen 等人[30]重点关注如何优化数据中心的网络流量,提出了基于深度强化学习模型的最优化流量调度方法;为了减少模型决策时间对用户服务质量的影响,作者进一步提出了一种长短流分开的多级处理机制,极大地提升流量调度方法的扩展性和可用性.Liang 等人[31]引入演员-评论家模型有效地解决了数据中心高度异构资源场景下的用户任务调度问题,提升了数据中心资源利用率,保障了用户服务质量.针对传输网络的

异构性和动态变化特点,以及服务质量的多目标需求等问题,Zhang 等人[32]设计了一种基于深度强化学习的多路径数据包调度方法,实现了自动化数据包选路传输,提高了用户服务质量.考虑到资源受限场景中的神经网络压缩算法无法兼容不同应用程序性能和底层物理资源的可用性问题,Liu 等人[33]利用深度学习模型自动化为输入模型选择合适的压缩算法,很好地实现了指定性能目标和资源约束的平衡.

上述工作充分反映了强化学习方法在调度优化领域的应用,但是由于不同问题中对应场景和目标的差异,导致这些工作中提出的方法难以用于解决虚拟机资源管理问题.

为此,Masoumzadeh 等人直接从主机过载检测和虚拟机选择 2 个问题的角度出发,分别提出了一种基于模糊 Q-Learning 的自动化主机过载检测机制[6]和基于模糊 Q-Learning 的虚拟机选择机制[7].Farahnakian 等人[34]则是利用 Q-Learning 模型解决主机电源状态的自动化切换.这些方法一定程度上有助于降低能耗或提升服务质量,但它们没有考虑其他问题(例如:主机欠载检测和虚拟机重分配)策略对性能的影响,难以实现最佳性能.区别于上述工作,RA-VMC 实现的是数据中心系统状态到虚拟机迁移策略(即虚拟机重分配)的端到端解决方案,整合了各种子问题最优策略,具有强大的优化能力和自适应能力.Pahlevan 等人[35]提出了一种基于强化学习的超启发式虚拟机分配算法,能够有效提升数据中心资源利用率,降低网络通信开销.但此方法并没有考虑资源整合过程中的虚拟机服务性能,因此难以保障系统的服务质量.相反,RA-VMC 综合考虑了包含能耗、服务质量和虚拟机迁移在内的优化目标.

2 研究动机和挑战

本文主要通过强化学习[17]来构建虚拟机迁移决策模型,从而实现端到端的虚拟机动态资源管理.本节首先介绍强化学习方法的基本原理,然后论述引入强化学习模型解决虚拟机整合问题的研究动机和所面临的主要挑战.

2.1 强化学习原理介绍

图 1 展示了标准强化学习[17]模型中智能体(agent)和环境(environment)之间交互的一般过程.在时刻 t 下,智能体观察到环境的状态 $\boldsymbol{s}_t$,并产生动作 $\boldsymbol{a}_t$.在智能体动作 $\boldsymbol{a}_t$ 的作用下,环境的状态转变为 $\boldsymbol{s}_{t+1}$,并且返回给智能体大小为 r_t 的奖赏.环境状态之间的转换以及产生的奖赏具有一定的随机性,并且假定满足 Markov 特性.智能体起初由于不知道环境状态的转化关系以及动作所产生的奖赏情况,因此无法合理地控制其产生的动作.通过与环境的交互,智能体结合环境状态、自身产生的动作以及对应的奖赏来进行训练和学习.优化目标是最大化期望累积折扣奖赏,表达式为:$\mathbb{E}\left[\sum_{t=0}^{\infty}\lambda^t r_t\right]$,其中 λ 为累积折扣奖赏的折扣因子.

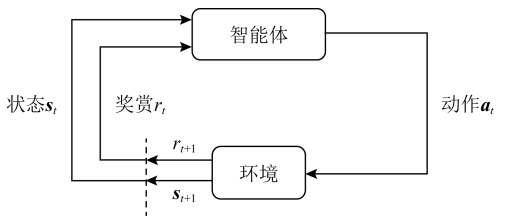


Fig. 1 Structure of standard reinforcement learning model

图 1 标准强化学习模型结构

2.2 研究动机和挑战

强化学习得益于其端到端特性和优秀的自学习能力和表达能力,已经被广泛应用于解决各种类型的决策控制问题[28-33].考虑到本文主要针对数据中心的虚拟机资源动态管理问题,研究合理的虚拟机动态迁移策略,本文引入强化学习来解决虚拟机整合问题的主要依据有 3 个:1)基于强化学习的虚拟机迁移决策模型在与环境不断交互过程中,能充分准确地学习到不同环境状态下最佳的虚拟机迁移方案,从而尽可能实现最佳的服务质量和能效目标;2)借助强化学习的端到端特性能够有效避免虚拟机整合问题中所面临的复杂中间过程,增加了虚拟机整合方法使用的灵活性;3)强化学习的自主学习机制能够极大地降低因环境或目标需求变化而导致人工更新所带来的维护和管理成本.

然而,复杂的数据中心环境对强化学习方法在虚拟整合问题中的有效应用产生了巨大挑战:

1) 难以设计合理的虚拟机迁移决策模型

在本问题中,环境状态表示数据中心的主机和虚拟机状态以及虚拟机的分布,即使是在 10×10 的主机-虚拟机规模下,环境状态的排列组合数量也超过了 10^{10}.同理,智能体动作表示虚拟机迁移方案,其空间大小随主机规模增加呈指数级增长.这导致无法存储所有的环境状态和智能体动作,也无法通

过构建确定性参数的 MDP 模型来直接计算出最佳的虚拟机迁移策略.能耗和服务质量的多目标需求更加加剧了决策模型的构建难度.

2) 难以保障决策模型训练过程成功收敛

决策模型在训练的过程中,由于无法事先获知虚拟机未来时刻的资源需求,使得环境状态在动作作用下所转化而成的下一时刻状态具有很强的随机性,即状态之间的转移概率和对应产生的奖赏并非由当前时刻的环境状态和智能体动作完全决定.这就导致决策模型在训练过程中难以收敛.另一方面,由于虚拟机被托管的主机位置有限,这种有限性可能导致参数梯度传递时朝着极端发展,即容易使得个别参数过大或过小,从而导致模型训练失效.

3　自适应虚拟机整合方法

针对 2.2 节所提到的两大挑战,本文分别提出一种基于深度确定性策略梯度(DDPG)[18]的虚拟机迁移决策模型和一种离线的自适应训练算法.RA-VMC 通过该决策模型动态制定虚拟机迁移方案.此部分首先介绍该决策模型和数据中心环境交互的基本框架,然后重点分析了该决策模型的设计过程,最后详细阐述了如何有效训练该虚拟机迁移决策模型.

3.1　整体框架

图 2 展示了 RA-VMC 中虚拟机迁移决策模型和数据中心环境交互过程的基本框架.

图 2 中左侧"云数据中心"图描述了一种典型的集中控制模式下的数据中心环境,右侧"智能体"图则表示基于 DDPG 的虚拟机迁移决策模型(或智能体).DDPG 算法包括演员(actor)和评论家(critic)共 2 部分.演员负责观察环境动作并产生相应动作;评论家则是用于评估演员动作作用下当前环境状态的价值.当前环境状态的动作价值表示的物理含义是:数据中心在当前时刻执行演员给出的虚拟机迁移动作后,综合服务质量和能耗变化所对应的期望累积折扣奖赏.演员分为动作网络和目标网络,评论家分为值网络和目标网络.演员的动作网络和目标网络分别用于产生环境当前状态和下一时刻状态对应的虚拟机迁移动作.评论家的值网络和目标网络则是分别用于评估当前环境状态采取当前智能体动作的价值,以及下一时刻环境状态采取下一时刻智能体动作的价值.

离线训练阶段,假定当前时刻为 t.智能体观察到数据中心环境所对应的系统状态 $\boldsymbol{s}_t$,并通过演员的动作网络产生虚拟机迁移方案 $\boldsymbol{a}_t$.数据中心根据此虚拟机迁移方案完成虚拟机迁移,计算此次虚拟机迁移所导致的服务质量等性能变化所对应的奖赏值 r_t,并转换为下一时刻状态 $\boldsymbol{s}_{t+1}$.评论家中值网络根据 $\boldsymbol{s}_t$ 和 $\boldsymbol{a}_t$ 计算出 $\boldsymbol{s}_t$ 在 $\boldsymbol{a}_t$ 下的状态动作价值 $Q(\boldsymbol{s}_t,\boldsymbol{a}_t)$.$Q(\boldsymbol{s}_t,\boldsymbol{a}_t)$被用于评估演员中动作网络在 $\boldsymbol{s}_t$ 状态下做出的动作是否合理,即优化演员的动作网络参数.演员的目标网络根据 $\boldsymbol{s}_{t+1}$ 计算出 $\boldsymbol{a}_{t+1}$.同理,

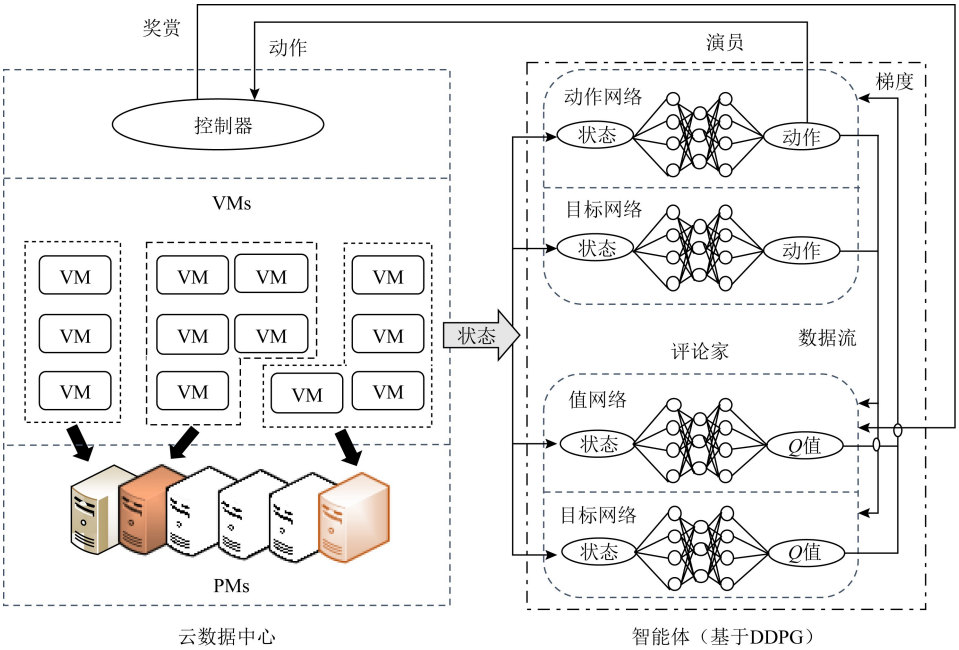


Fig. 2　Overall framework for the interaction between the agent and cloud data center

图 2　智能体与云数据中心交互的整体框架

评论家的目标网络可以评估出下一时刻环境状态在 $\boldsymbol{a}_{t+1}$ 动作作用下的 $Q'(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1})$，并且 $Q(\boldsymbol{s}_t,\boldsymbol{a}_t)$ 在数值上应该等于 $Q'(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1})+r_t$，故而可以通过评论家的目标网络来优化值网络，使之得到更加准确的状态-价值模型.

应用阶段，智能体演员的动作网络根据数据中心的状态实时计算出虚拟机迁移方案，然后交付数据中心予以实施.

3.2 虚拟机迁移决策模型

针对难以设计合理的虚拟机迁移决策模型的挑战，此部分分别重点介绍决策模型中环境状态、智能体动作、智能体的网络结构以及动作奖赏的设计方法.

3.2.1 状态张量化表示

数据中心的系统状态由主机的资源占用情况、虚拟机的资源需求情况以及虚拟机所在主机分布共同组成.针对状态空间爆炸的问题，以及为了有效表达各个部分之间的依赖关系，本文采用张量来形式化描述数据中心的系统状态，结构如图 3 所示.假定主机和虚拟机分别用 p 和 v 表示，资源用 d 表示，三者的数量分别用 N,M,D 表示，那么环境状态可以定义为一个由虚拟机、主机、资源种类组成的张量 $\boldsymbol{s}_t$，并且 $\boldsymbol{s}_t=(p,v,d)$.$\boldsymbol{s}_t$ 中任意一点表示 $\boldsymbol{s}_{p,v,d}$ 是主机 p 上虚拟机 v 的资源 d 的需求大小为 l.

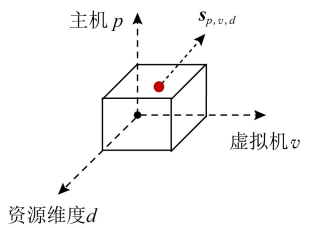


Fig. 3 Representation of the environment state in the VM migration decision-making model

图 3 虚拟机迁移决策模型的环境状态表示

3.2.2 确定性动作策略

智能体动作定义为虚拟机和主机的映射向量.由于虚拟机所在主机的映射空间大小为 N^M，无法直接通过随机动作策略来穷尽所有状态的概率大小，然后找出最佳的动作.故本文采用确定动作策略[18]，每次只输出一个概率最大的动作，从而能有效地解决因动作空间爆炸而导致决策模型无法正常学习的问题.确定性动作策略和随机性动作策略的主要差别体现在智能体演员动作网络的输出形式上.下面给出了智能体在采用确定性动作策略时的具体形式化定义：动作 $\boldsymbol{a}=(n_1,n_2,\cdots,n_M)$，$n_i\in\{1,2,\cdots,N\}$.例如：假定 $\boldsymbol{a}=(1,4,\cdots,20)$，那么该动作表示的物理含义为将第 1 个虚拟机迁移到编号 1 的物理主机，将第 2 个虚拟机迁移到编号为 4 的物理主机.依此类推，将最后一个虚拟机迁移到编号为 20 的物理主机.对于某特定虚拟机而言，如果动作给出的主机编号和其自身所在的主机编号相同，则说明该虚拟机无需迁移；否则，需要迁移该虚拟机.

3.2.3 网络结构

从 3.1 节可知 DDPG 主要由演员和评论家这 2 个部分组成，并且根据 DDPG 的原理可知，演员中的动作网络和目标网络，以及评论家中的值网络和目标网络结构上完全一致.因此，此部分重点分析演员动作网络和评论家值网络的优化及它们的基本结构.

1) 动作网络.负责根据观测到的环境状态产生虚拟机迁移动作.假定动作网络用包含参数 $\boldsymbol{\theta}^u$ 的函数 $\boldsymbol{u}$ 表示，则有：$\boldsymbol{a}=\boldsymbol{u}(\boldsymbol{s}|\boldsymbol{\theta}^u)$.演员的训练目标是通过优化参数 $\boldsymbol{\theta}^u$ 来得到最佳的动作函数 $\boldsymbol{u}$，此优化过程等同于最大化任意状态的动作价值，最优化动作网络参数计算为

$$\boldsymbol{\theta}^{u^*}=\arg\max_{\boldsymbol{\theta}^u}Q(\boldsymbol{s},\boldsymbol{u}(\boldsymbol{s}|\boldsymbol{\theta}^u)),$$

其中，Q 值由评论家网络计算得到，表示智能体观测环境状态 $\boldsymbol{s}$ 产生动作 $\boldsymbol{u}(\boldsymbol{s}|\boldsymbol{\theta}^u)$ 的期望累计折扣奖赏.为了得到最佳的动作函数 $\boldsymbol{u}^*$，评论家会根据评估得到的 Q 值来更新演员的动作网络参数 $\boldsymbol{\theta}^u$，更新过程为

$$\nabla_{\boldsymbol{\theta}^u}J\approx$$

$$\frac{1}{N}\sum_i\nabla_{\boldsymbol{a}}Q(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}^Q)|_{\boldsymbol{s}=\boldsymbol{s}_i,\boldsymbol{a}=\boldsymbol{u}(\boldsymbol{s}_i)}\nabla_{\boldsymbol{\theta}^u}\boldsymbol{u}(\boldsymbol{s}|\boldsymbol{\theta}^u)|_{\boldsymbol{s}_i},\quad(1)$$

其中，$\boldsymbol{\theta}^Q$ 表示评论家值网络参数，N 表示批次训练的样本数量.

考虑到环境状态的张量化结构具有较明显的空间化特征，如图 3 所示，故本文首先采用 3 层卷积神经网络(convolution neural network, CNN)[36]来从

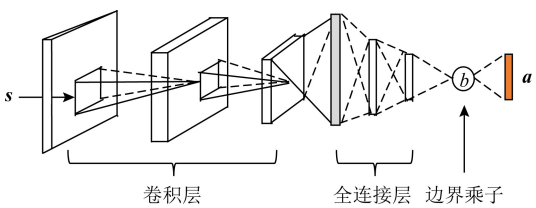


Fig. 4 Structure of actor network in VM migration decision-making model

图 4 虚拟机迁移决策模型的动作网络结构

状态中充分挖掘潜在的有用信息,同时还能提高演员动作的产生速度,相对其他深度神经网络而言更加高效.演员动作网络结构具体如图 4 所示.图 4 中卷积神经网络后紧接着是 3 层全连接层,用于进一步增强演员动作网络的表达能力.在此之后,通过一个边界乘子 b 来限制全连接层的输出,以此保证模型制定的动作能够使虚拟机映射到有效主机上.该乘子的基本原理是当全连接层产生的虚拟机迁移目的主机编号超过规定的主机编号范围时,按主机编号的最大最小进行处理.

2) 值网络.负责评估演员网络产生的动作价值.值网络用 Q 函数表示,参数为 $\boldsymbol{\theta}^Q$.通过计算 Q 值计算演员产生的动作梯度,进而优化演员动作网络.为了得到准确的 Q 函数,评论家目标网络计算下一时刻状态-动作价值,然后结合当前时刻的奖赏来评估值网络计算得到的状态-动作价值是否准确,即通过最小化损失函数来优化值网络,其中损失函数 $L(\boldsymbol{\theta}^Q)$ 的计算为

$$L(\boldsymbol{\theta}^Q)=\sqrt{\frac{1}{N}\sum_i(r_i+\lambda Q'(\boldsymbol{s}_{i+1},\boldsymbol{u}'(\boldsymbol{s}_{i+1}\mid\boldsymbol{\theta}^{u'})\mid\boldsymbol{\theta}^{Q'})-Q(\boldsymbol{s}_i,\boldsymbol{a}_i\mid\boldsymbol{\theta}^Q))^2}\,, \tag{2}$$

其中,$\boldsymbol{u}'$ 表示演员的目标网络函数,参数为 $\boldsymbol{\theta}^u$;Q' 表示评论家的目前网络函数,参数为 $\boldsymbol{\theta}^Q$.

值网络的网络结构如图 5 所示.状态 $\boldsymbol{s}$ 首先经过 3 层的 CNN 层和 2 层的全连接层,然后和同样经过全连接层处理后的动作 $\boldsymbol{a}$ 进行求和,然后进一步通过 3 层的全连接层计算产生 $Q(\boldsymbol{s},\boldsymbol{a}\mid\boldsymbol{\theta}^Q)$.详细的评论家的 Q 值计算过程可参考文献[18].

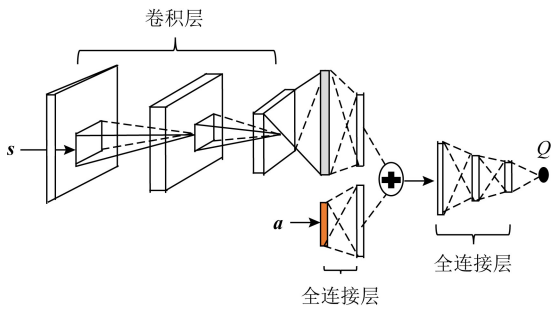


Fig. 5　Structure of critic network in decision-making model

图 5　决策模型的值网络结构

3.2.4　加权奖赏机制

本文主要考虑系统服务质量和能耗 2 方面优化目标,由于服务质量优化可以转化为尽可能地降低主机过载和虚拟机迁移对虚拟机性能的影响.因此本文主要从能耗、主机过载和虚拟机迁移数量 3 个方面来计算环境奖赏.

1) 能耗方面.通过统计数据中心正常运行过程中空闲主机的占比.空闲主机比重越低,说明总体的能耗也就越高.因此根据空闲主机占比来计算能耗相关的奖赏:

$$r_{\text{power}}(\boldsymbol{s}_t,\boldsymbol{a}_t)=1-\frac{2}{\mathrm{e}^{idle\_rate}+1}. \tag{3}$$

2) 主机过载方面.统计主机过载时,超出的资源量占虚拟机总共所需资源的比重.该比重越高,说明主机过载越严重,即性能下降越多.因此可以根据此比重来计算主机过载方面产生的奖赏:

$$r_{\text{overload}}(\boldsymbol{s}_t,\boldsymbol{a}_t)=\frac{1}{over\_rate+1}. \tag{4}$$

3) 虚拟机迁移方面.本文重点考虑 TCP/IP 网络下的虚拟机迁移,此种情况下,虚拟机迁移次数越多,意味着更多的 CPU 和网络带宽资源占用[37].因此可以根据虚拟机迁移次数来计算虚拟机迁移方面的奖赏:

$$r_{migrs}(\boldsymbol{s}_t,\boldsymbol{a}_t)=\frac{1}{1.1^{migrs}+1}. \tag{5}$$

其中 $idle\_rate$,$over\_rate$,$migrs$ 分别表示活跃主机数占比、过载主机数占比、虚拟机迁移数量.于是,智能体动作的单步奖赏为

$$r_t(\boldsymbol{s}_t,\boldsymbol{a}_t)=\alpha\times r_{\text{power}}(\boldsymbol{s}_t,\boldsymbol{a}_t)+\beta\times r_{\text{overload}}(\boldsymbol{s}_t,\boldsymbol{a}_t)+\gamma\times r_{migrs}(\boldsymbol{s}_t,\boldsymbol{a}_t), \tag{6}$$

$$\alpha+\beta+\gamma=1. \tag{7}$$

3.3　自适应离线训练算法

3.2 节中主要介绍虚拟机迁移决策模型的设计,此节则重点介绍如何训练该决策模型.首先介绍了在虚拟机负载未知的情况下如何表达数据中心系统状态之间的转化,然后引入一种反向梯度限定机制[38]来防止决策模型训练过程中动作越界,最后给出了该训练算法的伪代码.

3.3.1　自动化状态生成机制

传统的强化学习案例中,环境根据智能体的动作能够直接得出下一时刻状态.这种状态的转换具有一定的随机性,但是通常被认定符合 Markov 特性.而此问题中,环境状态包含了虚拟机动态负载情况.当环境接收到智能体的动作并完成虚拟机迁移后,其由于不知道虚拟机下一时刻的负载大小,因此无法确定下一时刻的状态变化,这也就可能导致在相同的环境状态和动作下,环境反馈智能体一种

完全不同、甚至完全相反的奖赏激励,使得智能体难以收敛,无法做出准确合理的虚拟机迁移决策.

为了解决这一问题,一种可行的办法是通过统计每个虚拟机的历史负载数据,分析出该虚拟机负载的分布特性(比如虚拟机 CPU 利用率在某时间段内呈正态分布),然后据此可以在该分布内对虚拟机下一时刻的负载做出一定的概率假设,从而可以判断出环境下一时刻状态.这种机制在准确获得虚拟机负载分布的前提下,极大地降低了状态空间的规模,有助于智能体高效地完成训练.然而,这种机制需要每个虚拟机大量的历史负载数据做支撑,并且容易受到虚拟机突发负载的影响,难以得到准确的概率分布.

综合随机虚拟机负载和基于特定统计概率的虚拟机负载 2 种机制的优缺点,本文在每个时刻为虚拟机按照均匀分布来选择其负载大小.均匀分布假设的依据是能够较为公平地分布虚拟机负载.在这种假设下,本文给出环境状态的变化过程,如图 6 所示.其中虚拟机负载仿真器遵循均匀概率分布产生虚拟机负载,任意时刻 t 下所有虚拟机负载用 $\boldsymbol{l}_t$ 表示,在已知主机总资源的前提下,环境时刻 t 的状态可以通过 $\boldsymbol{l}_t$ 和智能体时刻 $t-1$ 动作 $\boldsymbol{a}_{t-1}$ 唯一确定.

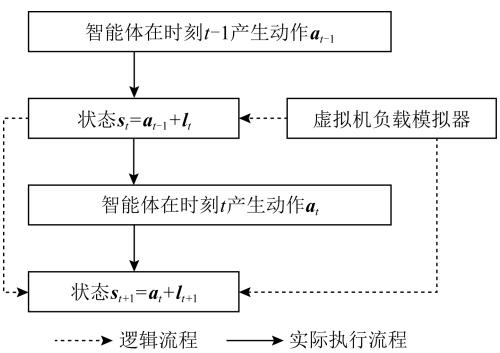


Fig. 6 Mechanism of automated state generation in RA-VMC

图 6 RA-VMC 自动化状态生成机制

3.3.2 反向梯度限定机制

从 3.2.2 节定义的智能体动作可知,每个动作具有一定边界限制,即表示每个虚拟机选择的新托管主机必须在数据中心给定的主机范围之内.这种边界限制可能导致智能体在训练过程中无法收敛.图 7 给出了这种问题的示例.图 7 描述了包含 30 个主机的数据中心场景下,随机选择的 10 个虚拟机在训练过程中被托管主机编号和迭代次数的关系.当训练进行到第 10 000 次时,绝大部分虚拟机都被放置到了 0 号和 29 号主机上.迭代继续,但大部分虚拟机的托管位置都无变化,这说明此时决策模型的训练已经失效.

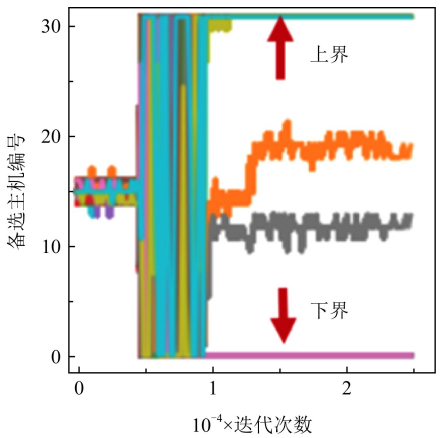


Fig. 7 Example of action out-of-bounds problem

图 7 动作越界问题示例

网络在更新参数后,产生越界动作并得到了较好的奖赏,那么评论家在传递给演员的动作梯度中可能会继续鼓励演员增大已经越界的动作值,以获得更高的奖赏,从而导致多次迭代后决策模型输出的动作远远超出合理范围.为此,本文引入反向梯度(inverting gradients)[38]限定机制来控制评论家传播梯度大小,旨在一定程度内控制动作范围.训练过程中,当演员输出的动作靠近边界或已经越界时,并且评论家传播的动作梯度促进演员动作朝 2 个极端发展时,该机制会缩小梯度,使得演员网络安全更新.该机制修正梯度 ∇_p' 的计算为

$$\nabla_p' = \nabla_p \times \begin{cases} \dfrac{p_{\max}-p}{p_{\max}-p_{\min}}, & \text{如果}\nabla_p\text{建议增加}\ p, \\ \dfrac{p-p_{\min}}{p_{\max}-p_{\min}}, & \text{否则}, \end{cases} \tag{8}$$

其中,∇_p 表示评论家传给演员网络的动作梯度,参数 p,$p_{\max}$,$p_{\min}$ 分别表示当前动作取值(即虚拟机托管的主机编号)、最大主机编号和最小主机编号.

3.3.3 离线训练算法伪代码

本文通过改进 DDPG 算法[10]来完成虚拟机迁移决策模型的训练.训练算法的伪代码如算法 1 所示.

算法 1. RA-VMC 中虚拟机迁移决策模型训练算法.

① 随机初始化评论家网络 $Q(\boldsymbol{s},\boldsymbol{a}\mid\boldsymbol{\theta}^Q)$ 及权重 $\boldsymbol{\theta}^Q$,演员网络 $\boldsymbol{u}(\boldsymbol{s}\mid\boldsymbol{\theta}^u)$ 及权重 $\boldsymbol{\theta}^u$;

② 初始化目标网络 Q' 和 $\boldsymbol{u}'$,权重分别为 $\boldsymbol{\theta}^{Q'}$ 和 $\boldsymbol{\theta}^{u'}$;

③ 初始化一个重放缓冲池 R,用于存放当前状态、动作、下一时刻状态和反馈;
④ 初始化动作探索概率 ε、探索概率衰减因子 Δ、累计奖赏折扣因子 λ;
⑤ for $episode=1$ to K do
⑥　随机产生一个虚拟机到主机的映射分布 $\boldsymbol{f}$ 和所有虚拟机负载 $\boldsymbol{l}$;
⑦　for $t=1$ to T do
⑧　　随机产生当前时刻观测状态 $\boldsymbol{s}_t \leftarrow \boldsymbol{f}+\boldsymbol{l}$;
⑨　　if $\boldsymbol{s}_t$ 已被探索 then
⑩　　随机产生一个虚拟机映射动作 $\boldsymbol{a}_t$;
⑪　　else
⑫　　选择动作 $\boldsymbol{a}_t \leftarrow u(\boldsymbol{s}_t \mid \boldsymbol{\theta}^u)$;
⑬　　end if
⑭　　按均匀分布产生虚拟机时刻 $t+1$ 负载 $\boldsymbol{l}'$;
⑮　　计算时刻 $t+1$ 的环境状态 $\boldsymbol{s}_{t+1} \leftarrow \boldsymbol{a}_t+\boldsymbol{l}'$;
⑯　　计算奖赏值:

$$r_t \leftarrow \alpha \times r_{\text{power}}(\boldsymbol{a}_t)+\beta \times r_{\text{overload}}(\boldsymbol{s}_{t+1})+\gamma \times r_{migrs}(\boldsymbol{f},\boldsymbol{a}_t);$$

⑰　　在 R 中存储 $(\boldsymbol{s}_t,\boldsymbol{a}_t,r_t,\boldsymbol{s}_{t+1})$;
⑱　　从 R 中随机采样 N 条样本;
⑲　　计算 $y_i \leftarrow r_i+\lambda Q'(\boldsymbol{s}_{i+1},\boldsymbol{u}'(\boldsymbol{s}_{i+1} \mid \boldsymbol{\theta}^{u'}) \mid \boldsymbol{\theta}^{Q'})$;
⑳　　计算损失函数

$$L=\sqrt{\frac{1}{N}\sum_i (y_i-Q(\boldsymbol{s}_i,\boldsymbol{a}_i \mid \boldsymbol{\theta}^Q))^2},$$

　　更新评论家值网络;
㉑　　if R 已经存满 then
㉒　　使用采样的策略梯度更新演员动作网络:

㉓　　$$\nabla_{\boldsymbol{\theta}_u} J \leftarrow \frac{1}{N}\sum_i inverting\_grad(\omega)\nabla_{\boldsymbol{\theta}_u}\boldsymbol{u}(\boldsymbol{s} \mid \boldsymbol{\theta}_u)\mid_{\boldsymbol{s}_i},\omega=\nabla_{\boldsymbol{a}}Q(\boldsymbol{s},\boldsymbol{a} \mid \boldsymbol{\theta}^Q)\mid_{\boldsymbol{s}=\boldsymbol{s}_i,\boldsymbol{a}=\boldsymbol{u}(\boldsymbol{s}_i)},\boldsymbol{u}(\boldsymbol{s}_i);$$

㉔　　更新目标网络参数:
㉕　　$\boldsymbol{\theta}^{Q'} \leftarrow \tau\boldsymbol{\theta}^Q+(1-\tau)\boldsymbol{\theta}^{Q'}$;
㉖　　$\boldsymbol{\theta}^{Q'} \leftarrow \tau\boldsymbol{\theta}^u+(1-\tau)\boldsymbol{\theta}^{u'}$;
㉗　　更新动作探索概率:
　　$\varepsilon \leftarrow \varepsilon \times \Delta$, if $\varepsilon \geqslant \varepsilon_{\min}$;
㉘　　end if
㉙　　重置 $\boldsymbol{f} \leftarrow \boldsymbol{a}$, $\boldsymbol{l} \leftarrow \boldsymbol{l}'$;
㉚　end for
㉛ end for

算法开始阶段,初始化演员和评论家网络;初始化一个缓冲池,用于存储当前状态、动作、下一刻状态以及奖赏;初始化探索率 ε、探索率的衰减因子 Δ、累积奖赏折扣因子 λ(行①～④).

本文采用多时序差分的学习方法来训练决策模型.每个回合(行⑤～㉛),首先随机产生所有虚拟机到主机的映射 $\boldsymbol{f}$,以及随机产生虚拟机负载 $\boldsymbol{l}$,根据 $\boldsymbol{f}$ 和 $\boldsymbol{l}$ 可以得到当前回合环境的起始状态 $\boldsymbol{s}$(行⑥).每个回合都包含 T 步处理,表示环境从状态 $\boldsymbol{s}$ 出发,每个回合历经 T 次与智能体的交互(行⑦～㉚).每个阶段,首先得到环境的当前状态 $\boldsymbol{s}_t$,并由演员的动作网络产生动作 $\boldsymbol{a}_t$(行⑨～⑬).进一步根据 3.3.1 节中的自动化状态生成机制产生环境的下一刻状态(行⑭～⑮).计算单步奖赏,并将相关结果存储到缓冲池中(行⑯～⑰).然后,每次从缓冲池中随机抽取得 N 条记录,每条记录均代表环境与智能体的一次交互(行⑱),并以此来训练评论家网络(行⑲～⑳).演员网络参数的更新采用了反向梯度限定机制(行㉒～㉓).紧接着更新目标网络(行㉔～㉖).随后,减少动作的探索概率,表示决策模型具备越来越准确的决策能力,直到小于探索概率(行㉗).最后,当前回合的一个阶段执行结束,环境切换到下一时刻状态(行㉙),继续执行.

4　仿真结果

本文采用开源流行的云仿真平台 CloudSim[39] 来对比评估 RA-VMC 的有关性能,实验中只考虑 CPU 一种类型资源,CloudSim 模拟数据中心运行 24 h.此节首先介绍基本的实验环境配置,然后从不同的性能指标展开评估.

4.1　实验环境配置

4.1.1　实验数据

实验中共仿真了 2 种不同类型的主机和 6 种不同类型的虚拟机,其具体的硬件配置参考来源于 Amazon Ec2[40] 中的主机和虚拟机实例数据.主机和虚拟机详细的硬件配置信息如表 1 和表 2 所示.2 种主机的功耗模型如表 3 所示.实验中,主机数量和虚拟机数量分别设置为 20 和 30.每种类型主机或虚拟机数量相等.

CloudSim 在运行初始阶段,会为每一个虚拟机分配一个事先定义的虚拟机负载文件,用于模拟虚拟机运行过程中的资源需求变化.本文共采用了 2 个公开的真实数据中心虚拟机负载数据集来作为仿真

虚拟机的运行依据:1)PlanetLab 数据集[2].该数据集记录了 2011 年 3 月、4 月某 10 天中共 11 746 个虚拟机的 CPU 资源利用率情况.2)GWA-T-12 fastStorage[41].该数据集来源于 Bitbrains 分布式数据中心,记录了共 1 237 个虚拟机长达 1 个月的 CPU 和内存资源占用情况.其中 2 组数据集的采样频率均为 5 min.实验首先分别随机抽取 2 组数据集中 1 天的数据,然后从当天的数据集中随机抽取指定数量的虚拟机负载文件,并以此作为虚拟机运行时负载.

Table 1 Host Types and Configurations

表 1 主机类型及配置

类型	硬件配置
M3	高频 Intel Xeon E5-2670 v2(Ivy Bridge)处理器,8 核 CPU,内存 64 GB
C3	高频 Intel Xeon E5-2680 v2(Ivy Bridge)处理器,8 核 CPU,内存 75 GB

Table 2 Virtual Machine Types

表 2 虚拟机类型

类型	硬件配置
m3.medium	1 核 CPU,0.6 GHz,内存 3.75 GB
m3.large	2 核 CPU,0.6 GHz,内存 7.50 GB
m3.xlarge	4 核 CPU,0.6 GHz,内存 15.0 GB
m3.2xlarge	8 核 CPU,0.6 GHz,内存 30.0 GB
c3.large	2 核 CPU,0.7 GHz,内存 3.75 GB
c3.xlarge	4 核 CPU,0.7 GHz,内存 7.50 GB

Table 3 Power Consumption Model of Host

表 3 主机功耗模型

CPU 利用率/%	功耗/(kW·h)	
	E5-2670	E5-2680
0	334	394
20	349	408
40	364	425
60	378	442
80	396	463
100	418	489

4.1.2 对比方法

本文通过组合不同的主机过载检测、主机欠载检测、虚拟机选择和虚拟机重分配(或虚拟机迁移)4 个问题策略来对比分析 RA-VMC 的有关性能,下面分别介绍各种策略情况.

1) 主机过载检测.通过固定阈值来判断主机是否过载.当主机 CPU 资源利用率超过该阈值时,认为该主机过载,策略包括:Thr(1.0),Thr(0.9),Thr(0.8),其分别表示是阈值为 1.0,阈值为 0.9,阈值为 0.8.

2) 主机欠载检测.采用贪婪的判定方法,遍历所有主机,每次认定资源利用率最低的主机为欠载主机,直到所有主机均处于过载、欠载或者虚拟机迁移状态[39].

3) 虚拟机选择[2].通过 3 种不同的策略来从过载主机上选择需要迁移的虚拟机,其分别是:随机选择一个虚拟机(random selection, RS);选择 CPU 资源占用最低的虚拟机(minimum utilization, MU);选择和其他虚拟机 CPU 资源使用情况相关性最大的虚拟机(maximum correlation, MC).虚拟机选择过程按照上述策略执行,直到去除待迁移虚拟机后,当前主机不过载.

4) 虚拟机迁移.本文首先选取诸如 OpenStack[19], CloudStack[20] 等开源云平台中 2 种流行的虚拟机分配方法:最先匹配(FF)和最先降序匹配(FFD),其中 FFD 每次都是选择放置虚拟机后剩余资源最少的主机.除此以外,本文还对比了 CloudSim 中默认配置的虚拟机分配方法:最小能耗变化匹配(minimum energy, ME)[39].ME 为正在迁移的虚拟机选择在放置该虚拟机后,能耗变化最小的主机.

4.1.3 评估指标

本文主要从服务质量、能耗以及虚拟机整合决策时间 3 个角度来对比评估虚拟机整合方法的性能.

1) 服务质量方面.采用服务等级协议的违反程度(service level agreement violation, SLAV)来评估各方法的服务质量性能,其具体由主机过载和虚拟机迁移 2 部分导致的性能下降(例如:规定正在迁移的虚拟机性能下降 90%,源主机的性能整体下降 10%[39])程度的乘积得到.前者表示仿真过程中主机过载时长占总运行时长的比例;后者表示所有虚拟机实际获得的 CPU 资源占所需总 CPU 资源的比例.

2) 能耗方面.统计了整个仿真过程中所有主机能耗.

3) 虚拟机整合决策时间方面.对比方法的决策

时间由主机过载、欠载检测、虚拟机选择、虚拟机迁移4个操作的时间共同组成;而RA-VMC则是统计决策模型每次从状态输入到动作输出所花时间.

此外,本文还评估了RA-VMC在3种不同的奖赏权重配置下的收敛性能.

4.2　性能评估

4.2.1　服务质量评估

如图8所示,图8中展示了各种不同虚拟机整合方法在5次重复实验下的平均SLAV情况.对比方法均是由虚拟整合各种子问题的不同策略组合而成,组合策略表示示例为:FF+RS+Thr(1),其表示采用由过载阈值为1的过载检测策略、随机虚拟机选择策略RS、虚拟机迁移方法FF,共同组合而成的虚拟机整合方法.RA-VMC对应的方法中,从左至右,其能耗权重依次为0.7,0.5,0.3.

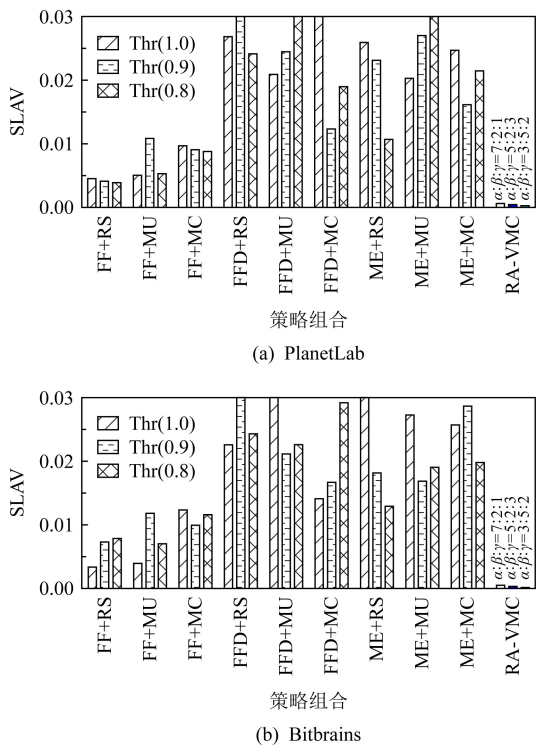


Fig. 8　SLAV comparison among VM consolidation methods

图8　虚拟机整合方法的SLAV性能对比

实验结果表明RA-VMC不管在PlanetLab还是Bitbrains数据集下,相对其他方法均能有效地提升服务达90%以上.RA-VMC能取得如此优秀的服务质量性能是因为:它通过最优化累积折扣加权奖赏,能够训练非常有效的决策模型,即使在动态变化的系统状态下,也能做出最佳的虚拟机迁移决策.而且,RA-VMC实际上可以等价于整合了各种不同子问题的最佳策略,并且综合考虑了各种策略之间的影响,从而相对于其他虚拟机整合方法能实现更高的服务质量.此外,从图8中明显可以看出虚拟机整合方法在不同的子问题策略组合下服务质量存在显著差异(例如:FF+RS在不同过载阈值下的性能差异,或者相同过载检测阈值下不同虚拟机选择或虚拟机迁移方法的性能差异).不同于这些方法,RA-VMC不需要关心虚拟机整合的中间环节,因此无需担心多种策略配置混乱所造成的成本和性能问题,能够有效提升虚拟机资源管理的灵活性.

4.2.2　能耗评估

图9展示了各种虚拟机整合方法分别在2种不同数据集下5次重复实验的平均能耗情况.

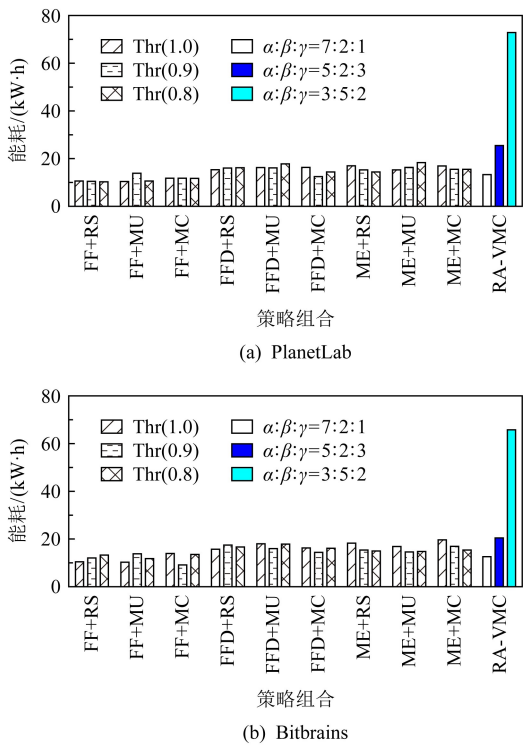


Fig. 9　Comparison of power consumption among VM consolidation methods

图9　虚拟机整合方法的能耗性能对比

实验结果表明不同能耗奖赏权重下RA-VMC的能耗性能差异明显,并且奖赏权重越小,其产生的能耗越高.当能耗奖赏的权重大小设置为0.7时,即不同奖赏权重比为$\alpha:\beta:\gamma=7:2:1$时,RA-VMC能耗最低.在PlanetLab数据集上,RA-VMC相对其他方法而言,能耗差异水平为$-23\%\sim+26\%$,表示RA-VMC相对其他方法至少降低了约23%的能耗,最多增加约26%,平均能耗减少约5%;而在Bitbrains数据集上,RA-VMC相对其他方法的能耗差异水平为$-31\%\sim+22\%$,平均能耗下降约13%.

4.2.3 决策时长评估

图 10 展示了各种虚拟机整合方法在 MacBook Pro 主机上(操作系统版本为 MacOS 10.13.6)5 次重复实验的平均迁移决策时间.主机硬件配置为:Intel® Core™ i5-5257U CPU @ 2.70 GHz, 8 GB 1 867 MHz DDR3.为了公平起见,所有方法均采用 Python 编程语言实现,其中 RA-VMC 基于 Tensorflow 框架(版本号 1.14.0)实现.

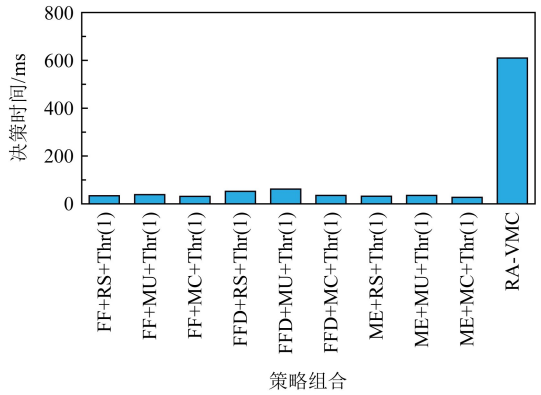


Fig. 10 Comparison of the decision-making time consumption among different VM consolidation methods

图 10 不同策略组合虚拟机整合方法决策时间对比

实验结果表明:RA-VMC 的决策时间相比其他方法增加约一个数量级.造成这一问题的主要原因在于 RA-VMC 为了能够准确学习到各种状态下不同动作激励,在构造决策模型时采用了比较复杂的网络结构,从而相对其他方法产生了更多的计算开销.这种问题在实际使用过程中可以通过模型压缩、剪枝等优化技术来降低模型的复杂度,从而减少决策时间.此外,还可以进一步采用 GPU 等硬件设备提升模型的决策效率.

如图 11 所示,本文进一步评估了 RA-VMC 在不同主机和虚拟机规模下的平均决策时间.其中保持主机和虚拟机数量之比为 2∶3不变,当主机规模等数量增大时,RA-VMC 的决策时间基本呈线性增长.

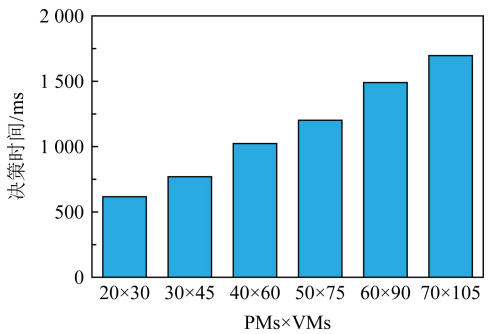


Fig. 11 Decision-making time consumption of RA-VMC varies with PM-VM scales

图 11 RA-VMC 在不同主机-虚拟机规模下的决策时间

4.2.4 收敛性能评估

图 12 展示了 RA-VMC 在 3 种不同的奖赏权重配置下虚拟机迁移决策模型的训练收敛情况.由于训练过程中采用了 ε-greedy 动作探索机制,故此部分列出了是否考虑探索动作 2 种情况下的累积平均收益随迭代次数的变化情况.其中 STD 曲线表示的含义是:在探索动作阶段,根据探索的动作来计算奖赏;而在非探索阶段,根据虚拟机迁移决策模型输出的动作来计算奖赏.ORI 曲线则表示整个训练阶段,均采用决策模型输出的动作来计算奖赏.

实验结果表明:在图 12 中 3 种不同的奖赏权重配置下,RA-VMC 的虚拟机迁移决策模型在训练过程中均能于第 70 000 次迭代时开始稳定收敛.虚拟机迁移决策模型在实际应用过程中可以通过模型优化、分布式训练、硬件加速以及结合实际的虚拟机负载的数据分布特征来进一步提高收敛速度.

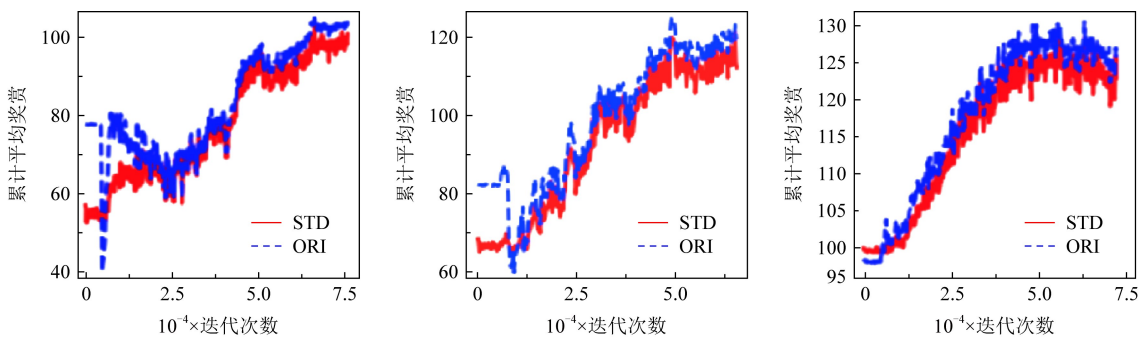


Fig. 12 The curves of the cumulative average gain of decision-making model in RA-VMC

图 12 RA-VMC 中决策模型累积平均收益曲线

5　结论和展望

针对数据中心虚拟机资源管理难以达到最佳性能和管理成本高的问题,本文提出了一种基于深度强化学习的自适应虚拟机整合方法(RA-VMC).RA-VMC 通过新构造的虚拟机迁移决策模型,实现了直接从系统状态到虚拟机迁移的动态决策,避免了虚拟机整合复杂的中间过程,增强了使用的灵活性,节约了运行管理成本;通过改进深度确定性策略梯度算法实现了虚拟机迁移决策模型的成功快速收敛,从而能够在动态变化的系统状态下持续做出最佳的虚拟机迁移策略.基于真实虚拟机负载数据进行仿真实验,实验结果表明 RA-VMC 能够有效提升系统服务质量.

未来工作中,我们计划将 RA-VMC 方法及最新的部分工作集成到 OpenStack 系统中,以此完成在真实数据中心场景下对 RA-VMC 的性能评估,并重点考虑通过模型压缩、剪枝等技术手段,以及结合实际的虚拟机负载特征来优化虚拟机迁移决策模型的收敛速度,提升模型的决策效率.

参　考　文　献

[1] Mell P, Grance T. The NIST definition of cloud computing [OL]. Special Publication (NIST SP), 2011[2020-06-01]. https://www.nist.gov/publications/nist-definition-cloud-computing

[2] Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers [J]. Concurrency and Computation Practice and Experience, 2020, 24(13): 1397-1420

[3] Beloglazov A, Buyya R. OpenStack neat: A framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds [J]. Concurrency and Computation Practice and Experience, 2015, 27(5): 1310-1333

[4] Shen Zhiming, Subbiah S, Gu Xiaohui, et al. Cloudscale: Elastic resource scaling for multi-tenant cloud systems [C] // Proc of the 2nd ACM Symp on Cloud Computing. New York: ACM, 2011

[5] Panigrahy R, Talwar K, Uyeda L, et al. Heuristics for vector bin packing [OL]. [2020-06-01]. https://www.microsoft.com/en-us/research/publication/heuristics-for-vector-bin-packing/

[6] Masoumzadeh S S, Hlavacs H. An intelligent and adaptive threshold based schema for energy and performance efficient dynamic VM consolidation [C] //Proc of European Conf on Energy Efficiency in Large Scale Distributed Systems. New York: ACM, 2013: 85-97

[7] Masoumzadeh S S, Hlavacs H. Integrating VM selection criteria in distributed dynamic VM consolidation using fuzzy q-learning [C] //Proc of Int Conf on Network & Service Management. Piscataway, NJ: IEEE, 2013: 332-338

[8] Chen Liuhua, Shen Haiying. Considering resource demand misalignments to reduce resource over-provisioning in cloud datacenters [C] //Proc of IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2017: 1-9. doi: 10.1109/INFOCOM.2017.8057084

[9] Chang B J, Lee Yuwei, Liang Y H. Reward-based Markov chain analysis adaptive global resource management for inter-cloud computing [J]. Future Generation Computer Systems, 2018, 79(2): 588-603

[10] Li Zhuozhao, Shen Haiying, Miles C. PageRankVM: A PageRank based algorithm with anti-collocation constraints for virtual machine placement in cloud datacenters [C] //Proc of IEEE Int Conf on Distributed Computing Systems. Piscataway, NJ: IEEE, 2018: 634-644

[11] Shaw R, Howley E, Barrett E. An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions [J]. Simulation Modelling Practice and Theory, 2019, 93(3): 322-342

[12] Farahnakian F, Pahikkala T, Liljeberg P, et al. Energy-aware VM consolidation in cloud data centers using utilization prediction model [J]. IEEE Transactions on Cloud Computing, 2019, 7(2): 524-536

[13] Nguyen T H, Di Francesco M, Yla-Jaaski A. Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers [J]. IEEE Transactions on Services Computing, 2020, 13(1): 186-199

[14] Lin Hao, Qi Xin, Yang Shuo, et al. Workload-driven VM consolidation in cloud data centers [C] //Proc of Int Parallel and Distributed Processing Symp. Piscataway, NJ: IEEE, 2015: 207-216

[15] Guo Liangmin, Hu Guiyin, Dong Yan, et al. A game-based consolidation method of virtual machines in cloud data centers with energy and load constraints [J]. IEEE Access, 2017, 6: 4664-4676

[16] Chen Liuhua, Shen Haiying. Consolidating complementary VMs with spatial/temporal-awareness in cloud datacenters [C] //Proc of IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2014: 1033-1041

[17] Sutton R S, Barto A G. Reinforcement Learning: An Introduction [M]. Cambridge, MA: MIT Press, 1998

[18] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning [C] //Proc of Int Conf on Learning Representations. San Juan, Puerto Rico: arXiv, 2016. doi: 10.32657/10356/90191

[19] NASA, Rackspace. OpenStack Project [OL]. [2020-06-01]. https://www.openstack.org/

[20] Liang Sheng. CloudStack [OL]. [2020-06-01]. https://cloudstack.apache.org/

[21] Chen Liuhua, Shen Haiying, Platt S. Cache contention aware virtual machine placement and migration in cloud datacenters [C] //Proc of IEEE Int Conf on Network Protocols. Piscataway, NJ: IEEE, 2016: 1-10. doi: 10.1109/ICNP.2016.7784447

[22] Benbrahim S E, Quintero A, Bellaïche M. Live placement of interdependent virtual machines to optimize cloud service profits and penalties on SLAs [J]. IEEE Transactions on Cloud Computing, 2019, 7(1): 237-249

[23] Han Zhenhua, Tan Haisheng, Wang Rui, et al. Energy-efficient dynamic virtual machine management in data centers [J]. IEEE/ACM Transactions on Networking, 2019, 27(1): 344-360

[24] Shen Haiying, Chen Liuhua. Distributed autonomous virtual resource management in datacenters using finite-Markov decision process [J]. IEEE/ACM Transactions on Networking, 2017, 25(6): 3836-3849

[25] Page L, Brin S, Motwani R, et al. The PageRank citation ranking: Bringing order to the Web [C] //Proc of the Web Conf. New York: ACM, 1999: 161-172

[26] Huang Zhe, Tsang D H K. M-convex VM consolidation: Towards a better VM workload consolidation [J]. IEEE Transactions on Cloud Computing, 2016, 4(4): 415-428

[27] Cui Yong, Yang Zhenjie, Xiao Shihan, et al. Traffic-aware virtual machine migration in topology-adaptive DCN [J]. IEEE/ACM Transactions on Networking, 2017, 25(6): 3427-3440

[28] Mao Hongzi, Netravali R, Alizadeh M. Neural adaptive video streaming with pensieve [C] //Proc of ACM Special Interest Group on Data Communication. New York: ACM, 2017: 197-210

[29] Zhang Ziyao, Ma Liang, Leung K K, et al. Q-placement: Reinforcement-learning-based service placement in software-defined networks [C] //Proc of Int Conf on Distributed Computing Systems. Piscataway, NJ: IEEE, 2018: 1527-1532

[30] Chen Li, Lingys J, Chen Kai, et al. Auto: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization [C] //Proc of ACM Special Interest Group on Data Communication. New York: ACM, 2018: 191-205

[31] Liang Sisheng, Yang Zhou, Jin Fang, et al. Job scheduling on data centers with deep reinforcement learning [OL]. [2020-06-01]. http://arxiv.org/abs/1909.07820

[32] Zhang Han, Li Wenzhong, Gao Shaohua, et al. ReLeS: A neural adaptive multipath scheduler based on deep reinforcement learning [C] //Proc of Conf on Computer Communications. Piscataway, NJ: IEEE, 2019: 1648-1656

[33] Liu Sicong, Lin Yingyan, Zhou Zimu, et al. On-demand deep model compression for mobile devices: A usage-driven model selection framework [C] //Proc of Annual Int Conf on Mobile Systems, Applications, and Services. New York: ACM, 2018: 389-400

[34] Farahnakian F, Liljeberg P, Plosila J. Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning [C] //Proc of Euromicro Int Conf on Parallel, Distributed, and Network-Based Processing. Piscataway, NJ: IEEE, 2014: 500-507

[35] Pahlevan A, Qu Xiaoyu, Zapater M, et al. Integrating heuristic and machine-learning methods for efficient virtual machine allocation in data centers [J]. IEEE Transactions on Computer Aided Design of Integrated Circuits & Systems, 2018, 37(8): 1667-1680

[36] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks [C] //Proc of Int Conf on Neural Information Processing Systems. New York: ACM, 2020: 1106-1114

[37] Clark C, Fraser K, Hand S, et al. Live migration of virtual machines [C] //Proc of Symp on Networked Systems Design & Implementation. New York: ACM, 2005. doi: 10.5220/0006682803840391

[38] Hausknecht M J, Stone P. Deep reinforcement learning in parameterized action space [C] //Proc of Int Conf on Learning Representations. San Juan, Puerto Rico: arXiv, 2015. doi: 10.1109/ism.2018.00025

[39] Calheiros R N, Ranjan R, Beloglazov A, et al. Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms [J]. Software Practice and Experience, 2011, 41(1): 23-50

[40] Amazon. Ec2 instance types [OL]. [2020-06-01]. https://aws.amazon.com/ec2/instance-types/

[41] Shen Siqi, van Beek V, Iosup A. Statistical characterization of business-critical workloads hosted in cloud datacenters [C] //Proc of Int Symp on Cluster, Cloud and Grid Computing. Piscataway, NJ: IEEE, 2015: 465-474

Yu Xian, born in 1992. PhD. His main research interests include cloud computing, edge cache and AIOps.

余 显,1992 年生.博士.主要研究方向为云计算、边缘缓存和 AIOps.

Li Zhenyu, born in 1980. PhD, professor. Member of IEEE. His main research interests include Internet architecture and Internet measurement.

李振宇,1980 年生.博士,研究员,IEEE 会员.主要研究方向为网络体系架构和网络测量.

Sun Sheng, born in 1990. PhD. Her main research interests include mobile edge computing and edge intelligence.

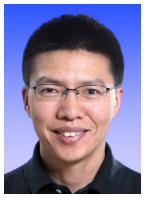
孙　胜,1990 年生.博士.主要研究方向为移动边缘计算和边缘智能.

Zhang Guangxing, born in 1978. PhD, associate professor. His main research interests include SDN/NFV system, Internet measurement, and future Internet architecture.

张广兴,1978 年生.博士,副研究员.主要研究方向为 SDN/NFV 系统、网络测量和未来网络架构.

Diao Zulong, born in 1988. PhD, assistant professor. His main research interests include machine learning, edge computing and network security.

刁祖龙,1988 年生.博士,助理研究员.主要研究方向为机器学习、边缘计算和网络安全.



Xie Gaogang, born in 1974. PhD, professor, PhD supervisor. His main research interests include SDN/NFV system, Internet measurement, future Internet architecture and AIOps.

谢高岗,1974 年生.博士,研究员,博士生导师.主要研究方向为 SDN/NFV 系统、网络测量、未来网络架构和 AIOps.