基于 Deep-Q-network 的多智能体强化学习的多目标工作流调度方法研究

重庆大学硕士学位论文

（学术学位）

学生姓名：王元斗

指导教师：夏云霓　教授

学科门类：工　学

学科名称：计算机科学与技术

研究方向：云计算

答辩委员会主席：李传东

授位时间：2019 年 6 月

Research on Multi-objective Workflow Scheduling with Deep-Q-network-based Multi-agent Reinforcement Learning

A Thesis Submitted to Chongqing University

in Partial Fulfillment of the Requirement for the

Master's Degree of Engineering

By

Yuandou Wang

Supervised by Prof. Yunni Xia

June , 2019

摘　要

云计算灵活的资源配置和现收现付的付费模式为执行大规模的科学和经济问题提供了一个高效、经济的分布式计算平台。大规模的科学和经济问题通常被建模成工作流模型，这些工作流上的应用程序不断增长的数据和计算需求使如何在云计算平台上高效的调度和部署这些应用程序的研究受到广泛关注。从用户的角度，最大完成时间和总花费是两个重要的服务质量指标，他们希望工作流应用程序可以尽可能快的完成，同时期望降低付出的总花费。然而，如何解决 IaaS 云环境下的多个工作流调度问题仍存在各种挑战，特别是针对多个相互冲突的目标的最优调度问题，仍有待妥善解决。现有的云工作流-多目标优化调度方法在很多方面仍然存在局限性，如在处理动态调度问题时编码受到先验或后验专家知识的限制，严重影响调度的性能。针对上述问题，本文重点研究了在不需要大量的专家知识和人为干预的情况下，同时优化最大完成时间和总花费的多工作流调度方法。

在本文中，我们首先分析了云工作流调度问题的最小化最大完成时间和总成本的双目标优化问题的形式化建模。为了优化多个工作流的最大完成时间和用户成本，我们考虑了一个以工作流应用程序和异构云主机的数量为状态输入，以最大完成时间和成本为奖励的马尔可夫博弈模型，通过合适的选择机制以及奖励函数的设计使博弈模型的解收敛于相关均衡。然后本文在多智能体学习场景中运用 Deep-Q-network 模型求解该马尔可夫博弈模型，以指导 IaaS 云上的多工作流调度。该方法将两个优化目标抽象成两个智能体，并考虑了一个随机的、动态交互的环境，旨在使智能体之间通过相互协作以及与环境的交互学习基于相关均衡的动态调度策略。为了验证本文提出的模型和方法，我们基于多个著名的科学工作流模板以及 Amazon EC2 云实例进行了广泛的案例研究，并与传统算法，如多目标粒子群优化算法、非主导排序遗传算法-II 和基于博弈理论的贪心算法进行对比实验。实验结果表明，我们提出的算法在生成调度计划的最大完成时间的最优性方面明显优于传统算法，最低水平差值优势超过 53.4%，而总成本相较于对比算法的差值比率最高不超过 9.9%。

关键词：多工作流-多目标优化调度；Deep-Q-network；多智能体强化学习；博弈论；IaaS 云计算

ABSTRACT

Cloud Computing provides an effective and cost-effective distributed computing platform for executing large-scale and complex scientific and economic problems with its flexible resource provisioning model and pay-as-you-go model. Generally, large-scale scientific and economic issues are modeled as workflows, and the ever-growth data and computational demands of these applications have led to widespread research on how to efficiently schedule and deploy them on cloud environments. From the user's perspective, it is important to consider the two Quality-of-Service metrics, i.e., the maximum completion time and total cost, basically, they want workflow applications can be completed as quickly as possible while expecting to reduce the total cost. Nevertheless, various challenges, especially its optimal scheduling for multiple conflicting objectives and diverse different workflow applications, are yet to be addressed properly. Existing multi-objective workflow scheduling approaches are still limited in many ways, e.g., encoding is restricted by prior or posteriori experts' knowledge when handling dynamic multi-objective scheduling problems, which strongly influences the performance of scheduling. In view of the above problems, we study the research on multi-objective workflow scheduling with Deep-Q-Network-based multi-agent reinforcement learning that optimizes the make-span and total cost without requiring enormous experts' knowledge and human intervention.

In this paper, we first formulate the problem with two optimization goals, i.e., minimizing make-span and total cost. To optimize multi-workflow completion time and user's cost, we consider a Markov game model which takes the number of workflow applications and heterogeneous virtual machines as state input and the maximum completion time and cost as rewards. Then a suitable selection mechanism and reward functions are designed respectively, which can guarantee the game model to converge to a unique correlated equilibrium. Then we apply a Deep-Q-Network model in a multi-agent reinforcement learning setting to solve the Markov game model, aims to guide the scheduling of multi-workflows over Infrastructure-as-a-Service clouds. The method is capable of seeking for correlated equilibrium by two agents' collaboration and learning from environment interaction. To validate our proposed approach, we conduct extensive case studies based on multiple well-known scientific workflow

templates and Amazon EC2 cloud, and make compare with traditional algorithms, e.g., multi-objective particle swarm optimization, non-dominated sorting genetic algorithm-II, and game-theoretic-based greedy algorithm. Experimental results clearly suggest that our proposed approach outperforms traditional ones in terms of optimality of scheduling plans generated. The advantage of our proposed method exceeds 53.4% at the lowest level of make-span, and the highest level of total cost does not exceed 9.9% in terms of the percentage of difference between baseline algorithms and our proposed method.

Keywords: Multi-workflow multi-objective optimization scheduling; Deep-Q-network; multi-agent reinforcement learning; game theory; Infrastructure-as-a-Service Cloud

目 录

1 绪论

1.1 研究背景及意义

云计算正逐渐成为一个具有大规模、异构的自主系统集合和灵活的计算架构的高性能的计算环境[1]-[3]。它提供了构建数据密集型或计算密集型并行应用程序的工具和技术，同时与传统的并行计算技术相比，它的价格要便宜得多。因此，在云计算领域，如调度、虚拟机放置、能耗管理、隐私与策略、安全等领域的研究工作越来越多[4]-[7]。

近年来，云环境下的工作流调度由于其在科学和经济领域的广泛应用而备受关注[8]-[13]。工作流通常被数学形式化建模为一个有n个任务节点的有向无环图，同时这些任务需要满足优先次序约束条件。云环境中的工作流调度是指将多个任务匹配到可支撑的计算资源m上，即在基础设施即服务（IaaS）云平台上创建的虚拟机或云主机实例。对于多目标调度问题而言，其调度的多个目标之间有时会发生冲突。例如，为了最小化执行时间，具有高性能配置的云主机比低性能配置的云主机更可取；然而，高性能配置的云主机通常更昂贵，因此最小化执行时间可能与降低成本目标相矛盾。在分布式平台上调度多任务工作流是一个 np-complete 问题[14]，这一点也得到了广泛的认同。因此，通过基于遍历的算法生成最优调度方案是非常耗时的。幸运的是，具有多项式复杂度的启发式和元启发式算法能够以可接受的最优性损失为代价生成调度方案的近似解或近似最优解[15]。多目标粒子群优化算法（MOPSO）[16]和非支配排序遗传算法-II（NSGA-II）[17]就是很好的例子。虽然这些算法提供了令人满意的解决方案，但通常在编码方案方面，它们需要大量的先验或后验专家知识和人为干预。另外，值得注意的是，博弈理论模型和方法在处理基于云的工作流调度问题也有很强的能力，这方面的研究贡献也有很多[18]-[21]。

近年来，随着新型机器学习算法的日益广泛和强大，利用强化学习和基于 Q-learning 的算法寻找近似最优工作流调度方案的研究也越来越多[22]-[25]。然而，大多数现有的研究贡献都集中在解决具有服务水平协议（SLA）约束条件的单目标工作流调度上。虽然在很多领域中已经存在各种多智能体强化学习模型和方法，如针对多机器人控制、分散网络路由、分布式负载均衡、电子拍卖和交通控制等问题，但基于多智能体强化学习的多工作流-多目标优化调度方法仍然不存在（至少截止到目前，已发表出来的研究工作中）。

本文将多工作流调度问题转化为离散事件和多准则交互的马尔可夫博弈模型，提出了一种针对多目标工作流调度的基于 DQN 的多智能体强化学习（DQN-based

MARL）算法。多个 DQN 智能体在马尔可夫博弈学习环境中进行训练，并从遗留系统，如神经网络中的启发式算法，中获取数据。本文考虑每个 DQN 智能体可观察到其他所有智能体的行为和奖励，并根据训练结果和选择机制选择各自的联合分布行为策略以及更新环境。也就是说，所得到的工作流调度计划是通过自我学习和自我优化的方式生成的。本文提出的方法具有以下优点：1）可以针对流程模型类型不同的工作流和资源配置不同的异构云主机对智能体进行训练；2）无需人为干预或大量的专家知识即可获得多目标工作流调度计划。我们基于真实的第三方 IaaS 云资源数据，以及使用多个科学工作流模板进行了广泛的案例研究。实验结果表明，本文提出的方法在最大完成时间这项指标上完全打败了传统的基准对比算法，至少优于对比算法 53.4%；成本优化方面尚有改进之处，但即使在最坏情况下，本文提出的算法比基准对比算法在调度总花费的结果上，最高不超过 9.9%。

1.2 国内外研究现状

众所周知，在分布式平台上安排多任务工作流是一个 NP-complete 问题。因此，通过基于遍历的算法产生最佳时间表非常耗时。幸运的是，具有多项式复杂性的启发式和元启发式策略能够以可接受的最优性损失为代价产生近似或接近最优的解，如：

Choudhary[8]等人提出了一种针对单工作流的基于元启发式的双目标优化调度算法，考虑最大限度地缩短最大完成时间并最小化成本。该算法是流行的元启发式-重力搜索算法（GSA）与流行的启发式-异构完成时间算法（HEFT）的混合体，用于调度工作流应用程序，并通过引入一个称为成本-时间等效的因子将双目标优化问题转化为一个带约束条件的单目标优化问题来求解。然后将金钱成本（MCR）比和调度长度比（SLR）作为性能指标与现有算法进行性能比较。

Kaur[26]等人提出一种多目标细菌觅食优化算法（MOBFOA），通过应用帕累托最优前沿技术对原始的细菌觅食优化算法（BFOA）进行改进以处理多目标优化调度问题。其改进主要是从占优和非占优前沿中选择细菌的位置以获得所获得的解的多样性。通过引入自适应步长改善 BFOA 的收敛的准确性和速度，并使用新的适应度分配方法和细菌选择过程同时优化多个目标，即最小化弹性工作时间、最大完成时间和资源使用成本，重点考虑了多目标权衡的独立工作的调度。

Zhang[27]等人提出了一种用于工作流调度的低能耗和高稳定性的双目标遗传算法（BOGA）。Casas[28]等人提出了一种用于云系统科学应用的 GA-ETI 调度器，旨在通过提供几个以帕累托前沿的方式的高效的解决方案来优化工作流的最大完成时间和成本。GA-ETI 通过有目的地/定制地修改其交叉和突变操作符来增强原始遗传算法，并且它使用增强的交叉来组合基因簇，而不是随机分割的染色体；

GA-ETI 还使用增加/减少突变来添加/删除给定染色体上的虚拟机。与原始遗传算法相比，这两种修改都降低了固有的随机性。

Verma[29]等人针对工作流的复杂性和云环境的动态性，以及 IaaS 云上具有多个冲突目标的云工作流调度问题，提出了一种基于非优先排序的混合粒子群优化（HPSO）算法。HPSO 启发式算法旨在优化两个具有冲突性的目标，即在截止日期和预算约束条件下的最大完成时间和成本，并利用帕累托前沿技术得出一组帕累托最优解供用户做选择。该算法是预算和截止期限制的异构最早完成时间（BDHEFT）算法和多目标粒子群优化（MOPSO）算法的混合。

Zhou[30]等人研究了在 IaaS 云中调度工作流的成本和最大完成时间的联合优化，并将模糊优势排序机制和启发式的 HEFT 列表算法紧密整合起来，提出了一种基于模糊优势排序的异构最早完成时间（FDHEFT）算法。

最近，博弈论和强化学习模型和方法也被广泛应用于多约束过程调度问题中[31]-[37]。在很多研究工作中，博弈论中的均衡概念和多智能体训练方法在处理多约束和多目标优化问题时具有很强的有效性。比较典型的有：

Iranpour[19]等人提出一种新的两层（区域-局部）分布式资源管理算法，可用于处理大规模的请求接纳控制和负载均衡等管理任务。该算法每个组件都执行独立管理资源的任务，其中控制任务在应用服务器和代理服务器之间划分。为了控制请求的接收，其引入了包含两个模糊控制器的自适应模糊 2 型控制器；为了在代理服务器中分配请求和平衡负载，其提出了一种基于博弈论的算法，使结果收敛于纳什均衡。

Duan[20]等人将调度问题形式化建模为序贯合作博弈模型，并提出了一种通信和存储感知的多目标算法，旨在满足网络带宽和存储要求两个约束条件的同时优化执行时间和运营成本两个用户目标。Wang[18]等人提出了一种基于动态博弈理论模型的多目标优化工作流调度方法，以减少工作流的最大完成时间和总成本，并最大化异构云虚拟机中工作负载分配的系统公平性。

Wu[22]等人以任务调度时间最小化和负载均衡优化为调度目标，建立云工作流环境下的马尔可夫决策过程（MDPs）模型，并提出了一种基于强化学习的多目标优化云工作流调度算法。该算法结合 Q-learning 的特点，在 Q-learning 奖励函数中加入一个带有加权适应值函数的函数使其实现多目标优化调度。其调度方案也是一个帕累托最优解集合，根据用户的偏好来选择最优的调度方案。Cui[24]等人针对云环境下多个不同优先级提交的DAG工作流应用程序的调度问题，应用Q-learning算法获取云工作流调度方案。

Mao[32]等人针对多资源集群调度问题，考虑构建直接从经验中学习管理资源的系统，并提出了一个名为 DeepRM 的示例解决方案，将具有多种资源需求的打

包任务问题转化为学习问题。其实验结果表明，直接从经验中学习资源管理策略，如果能够使它作用于实际环境中，可以为当前基于启发式的方法提供真正的替代方案。

1.3 本文的主要工作

本文面向 IaaS 云系统中多工作流调度问题，以解决其马尔可夫博弈与多智能体强化学习建模、多 QoS 指标相关均衡解分析、多 QoS 指标工作流调度策略获取这几个关键科学问题为主要目标：以多目标优化、博弈理论和多智能体强化学习理论为出发点，开展 IaaS 云环境下多工作流调度策略和多目标优化调度量化建模的研究，研究多个冲突性 QoS 指标的机制设计与相关均衡的存在性以及设计基于 DQN 的多智能体强化学习算法得出收敛于相关均衡的动态调度策略；以 Amazon EC2 云主机单位服务价格、性能参数和五种著名的科学工作流模板作为实验的输入数据进行模拟实验，并与传统的基准算法作对比得出最终实验结果。具体而言，本文主要在以下几个方面展开了深入研究：

1.3.1 多工作流-多目标优化调度问题的动态博弈建模

以多目标优化问题、博弈理论为形式化描述工具，对 IaaS 云环境下动态的多工作流-多目标优化调度问题形式化建模为马尔可夫博弈模型，量化分析该博弈模型下的相关均衡解的存在性及其证明，实现对 IaaS 云环境下的多工作流-多 QoS 指标优化调度过程的精确形式化描述。这部分的研究包括以下具体内容：

① 工作流建模；

② 云主机资源建模；

③ 基于最大完成时间和总花费优化的调度问题的量化建模；

④ 基于马尔可夫博弈模型的云工作流调度问题建模及其相关均衡解分析。

1.3.2 基于 DQN 的多智能体强化学习的工作流调度算法

经过马尔可夫博弈的形式化建模和转换，复杂而具体的 IaaS 云环境下的多工作流-多目标优化调度问题可以转换为抽象精炼和具有状态概率转移的动态博弈过程的形式化模型。接下来，从马尔可夫博弈的形式化模型及相关均衡解的分析，导出量化的基于 DQN 算法的多智能体强化学习（DQN-based MARL）的系统模型，再通过设计合适的奖励函数和选择机制保证多个 DQN 智能体在马尔可夫博弈学习环境下的学习策略收敛于相关均衡解。接下来，将真实的实验数据代入多个 DQN 智能体强化学习模型中，通过 DQN 智能体之间、DQN 智能体与环境之间的交互和状态更新，最终获得自我学习和自我优化的最大完成时间和总花费的调度结果。最后与基准算法，即基于博弈的贪心算法（GTBGA）、非支配排序遗传算法-II（NSGA-II）和多目标粒子群优化算法（MOPSO）进行对比实验，对取得的调度

策略和最大完成时间和总花费指标进行结果分析和检验。该部分研究主要包括以下具体内容：

① 基于多个 DQN 智能体的强化学习框架分析；

② DQN 智能体的奖励函数设计以及策略选择机制设计；

③ 基于 DQN 的多智能体强化学习的云工作流调度算法设计；

④ 案例研究、实验结果分析与检验。

1.4 本文的组织结构

本文总共分为六个章节，以下是每个章节的主要内容安排情况：

第一章首先介绍了云计算和云环境下工作流调度问题研究的背景以及发展现状，指出了云工作流调度问题是当前云计算领域的研究热点之一。之后对目前国内外学者在云计算环境下多目标工作流调度方面的研究成果进行简要的概括性介绍和总结，指出了目前研究存在的不足之处，进而引出本文重点关注的问题、研究方向以及研究内容，并给出全文的组织结构。

第二章主要对云环境下多工作流-多 QoS 指标优化调度问题建模和求解方法所涉及到的相关理论知识进行简要的介绍。首先简要阐述了云计算、云环境下工作流应用的基本概念，然后针对多目标工作流调度问题对本文建模过程中所用到的数学理论基础进行详细的介绍，主要涉及到多目标优化问题及其求解方法、博弈模型与均衡解，以及强化学习的相关理论知识，包括单个智能体案例和多智能体案例分析，并分别给出了部分经典的强化学习算法。这一章涉及的数学定义、公式推导、模型以及求解方法为第三章的形式化建模和第四章的系统模型及算法设计提供了理论基础。

第三章是本文的核心章节之一。首先给出了云工作流应用模型和云资源性能-价格模型，以及在 IaaS 云环境下基于马尔可夫博弈的多工作流-多目标优化调度模型。这一章涉及的数学定义及公式推导为下一章的系统模型分析及算法设计提供了理论支持。

第四章是本文的核心章节。首先在第三章提出的问题模型的基础上，给出了求解该调度问题的系统模型。接着分析了单智能体学习场景下的 DQN 算法。然后将其扩展到多智能体学习场景下，针对基于 DQN 算法的多工作流-多目标优化的调度问题，为每个 DQN 智能体设计了合适的奖励函数，以及选择机制设计使 DQN 智能体学习基于相关均衡的动态策略并保证其收敛性，最终形成了基于 DQN 的多智能体强化学习的多目标云工作流调度算法。

第五章为了验证前两章提出的模型和算法，采用五种著名的科学工作流模板以及异构的云资源，如 Amazon EC2 云主机，进行模拟实验。对每个科学工作流模板，

本文考虑了不同的任务类型，当不同类型的任务被部署在不同配置的云主机上运行时，其性能和服务价格都是有差异的。本文采用基于 Geekbench 平台公信的测试数据为验证算法提供输入数据，将多目标粒子群优化算法（MOPSO）、非支配排序遗传算法-II（NSGA-II）和基于博弈理论的贪心算法（GTBGA）作为基准对比算法得出实验结论。

第六章对全文进行了系统性的概括和总结，指出了当前多工作流-多目标优化调度模型及求解方法中存在的局限性和不足之处，并给出了下一步的改进方案以及未来的研究方向。

2 云工作流及调度相关理论基础

2.1 云计算简介

云计算中的“云”一词在不同的时代与技术发展背景下有着不同的比喻义。早在1977年，云符号就被用于原始的阿帕奇网络（ARPANET）中，描述计算设备的网络，同样它也被用于表示开始于1981年的计算科学网络（CSNET），这两种网络都是互联网的前身。因此“云”这个词常被用作互联网的隐喻，并且一个标准的云形状被用来表示电话示意图上的网络。到了1993年，“云”这个术语被用来指代分布式计算平台，当苹果公司剥离了通用魔术（General Magic）通信公司后，AT&T公司就将其用于描述他们的配对远程脚本（Telescript）和个性化链接（PersonaLink）技术。从2000年开始，云计算时代正式来临，一大批与云计算相关的技术开始涌现。比如亚马逊公司的Amazon Web Service子公司于2006年推出的弹性云计算（EC2），谷歌公司于2008年发布Google App Engine等。此后，云计算的发展经历了最迅猛的十年。2010年，微软发布了Microsoft Azure，同年，Rackspace Hosting和美国航空航天局（NASA）联合建立了开源的云软件OpenStack项目，到2011年云计算成为IBM SmartCloud框架最重要的组成部分。2012年甲骨文公司的Oracle Cloud成为首个可以为用户提供一套集成的IT解决方案，包括应用程序（SaaS），平台（PaaS）以及基础设施（IaaS）层。

现在，“云”的隐喻更多地被表述为：整个提供商管理的硬件和软件套件被视为无形的云，提供服务的网络元素组不需要由单独处理或管理。由此比喻义引申，云计算的定义普遍被总结为：

“云计算是一种基于互联网的计算方式，通过这种方式，共享的软硬件资源和信息，可以按需求提供给计算机各种终端和其他设备。”

云服务供应商通过不同的服务模式为用户提供“服务”。尽管现如今面向服务的架构倡导“一切皆服务”，但这些服务模式中每一种都来自于美国国家标准和技术研究院定义的三种标准服务模式，它们是：在应用程序层的软件即服务（Software-as-a-Service），平台层的平台即服务（Platform-as-a-Service）以及基础设施层的基础设施即服务（Infrastructure-as-a-Service）。图2.1给出了在这一框架下云计算的概念图。

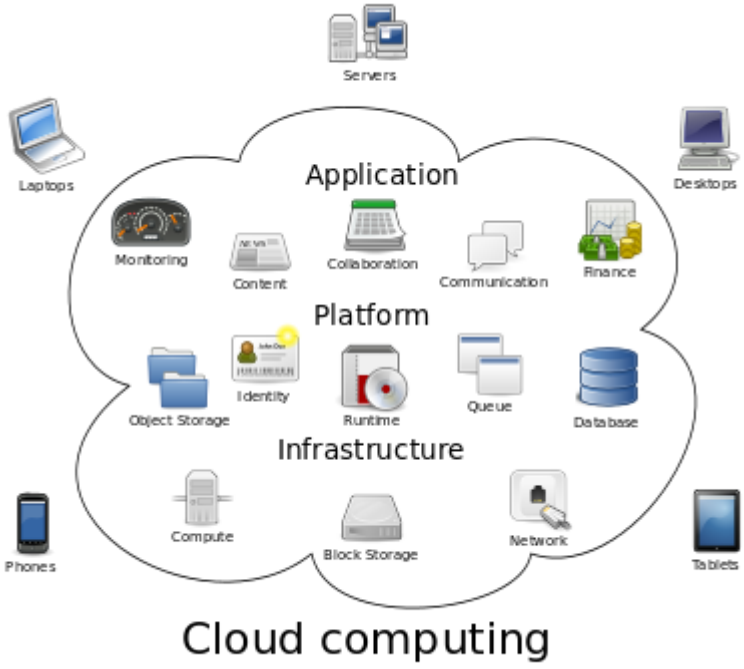


图 2.1　云计算概念图[38]

Fig 2.1 The conceptual structure of cloud computing

软件即服务，简称 SaaS，是一种软件许可和交付模式。在这种模式中，云端或软件服务供应商集中式代管软件或相关数据信息，以租赁的概念提供客户服务，软件仅需通过网络访问，而不需通过安装即可使用。通常，用户使用精简客户端经由一个网页浏览器来访问 SaaS 服务。例如：Microsoft CRM 与 Salesforce.com。

平台即服务，简称 PaaS，是一种基于平台的服务模式。它提供了一个平台，允许用户开发、运行以及管理应用程序，而不需要构建和维护通常与开发和启动应用程序相关的基础架构的复杂性。例如：Google App Engine。

基础设施即服务，简称 IaaS，是提供高级的应用程序接口（API）的在线服务。它用于取消对底层网络基础设施的各种低级细节的应用，如物理计算资源、位置、数据分区、扩展、安全和备份等。典型的应用实例包括：Amazon AWS 公司的 Amazon EC2、Rackspace 公司的 OpenStack。

美国国家标准与技术研究院根据云计算的定义，确定了云计算服务应该具备的五个基本特征，本文将其总结在表 2.1 中。

表 2.1 云计算的五个基本特征[38]

Table 2.1 Units for five basic characteristics of Cloud Computing

特点	描述
按需自助服务 (On-demand self-service)	用户可以根据需求单方面地供给计算能力，如服务器时间和网络存储，而不必与每个服务供应商进行人为的交互。
广泛的网络访问 (Broad network access)	可通过网络获得服务并通过标准的机制访问，这些机制促进异构的精简或密集的客户端平台（如移动电话、平板电脑、笔记本电脑和工作站等）的使用。
共享资源池 (Resource Pooling)	供应商的计算资源被池化，并使用一个多租户模式为多个用户提供服务，同时根据用户的需求动态地分配和重新分配不同的物理资源以及虚拟资源。
快速弹性 (Rapid elasticity)	根据需求快速地向内和向外扩展，弹性地供应和释放资源的能力。对于用户而言，可用于供给的资源通常是无限的，并且可以随时以任何数量进行分配。
可监控与可度量的服务 (Measured service)	通过在适合于服务类型的某种抽象级别上利用计量能力来自动控制和优化资源使用。可以监视、控制和报告资源使用情况，从而为所使用的服务的供应商和用户提供透明性。

根据上述特点和优势，现如今云计算平台常常被用于大规模的、复杂的经济、科学或工程问题中，如 DNA 结构、基因图谱定序、解析癌症细胞等高端科学计算。

2.2 云工作流简介

随着与日俱增的按需付费的云服务，越来越多的企业和社区选择云计算平台来部署他们的商业或科学工作流应用程序。云工作流是一个被广泛应用的模型，用于描述在 IaaS 云上部署和托管的商业的、科学的以及工程相关的数据密集型或计算密集型的应用程序。

工作流的概念根植于商业企业，作为一种业务流程的建模工具。这些业务工作流程旨在自动化和优化组织的流程，它们被视为有序的活动序列，并且是由 1993 年成立的工作流管理联盟（WfMC）领导的成熟研究领域。之后这种工作流的概念延伸到科学界，被统一称作科学工作流。科学工作流一般支持大规模、复杂的科学过程，它们旨在通过管理、分析、模拟和可视化科学数据来进行实验和证明科学假设。因此，尽管商业和科学工作流共享相同的基本概念，但两者都有特定的要求，需要单独考虑。本文重点关注了科学工作流的调度情况，后文都将其简称为工作流。

各种科学领域都使用工作流来分析大量数据，并在云计算环境中有效地运行复杂的模拟和实验。例如，Montage 工作流是一个天文学应用程序，其特点是 I/O 密集，用于在一组输入图像的基础上创建天空的自定义镶嵌。它使天文学家能够生成一个合成图像，该图像显示的天空区域太大，无法由天文照相机产生，并且已经用不同波长和仪器进行了测量。在工作流执行过程中，将根据输入图像的几何图形计算输出图像的几何图形。然后，重新投影输入数据，使其具有相同的空间比例和旋转。接下来是对所有图像的背景进行标准化。最后，将所有处理后的输入图像进行合并，形成天空区域的最终马赛克图像。再如，Cybershake 工作流是一个由南加州地震中心使用的数据和内存密集型地震危险特征描述应用程序。该工作流首先通过模拟为感兴趣的区域生成应变格林张量。然后使用这些应变格林张量数据为每个预测破裂生成合成地震图，然后为给定区域创建加速度和概率危险曲线。其他例子则包括 Inspiral、Sipht 和 Epigenomics 工作流。Inspiral 是一种内存密集型应用，用于物理领域，目的是探测引力波。在生物信息学中，Sipht 用于在国家生物技术信息中心数据库中自动搜索所有细菌复制体的小 RNA 编码基因。同样在生物信息学领域，Epigenomics 工作流是一个 CPU 密集型应用程序，可自动执行各种基因组测序操作。这五种著名的科学计算问题常被抽象为五种工作流模板被广泛应用于云工作流调度研究中，常常被作为案例研究进行模拟实验。

云工作流调度问题可简单的理解为，在满足任务的数据和结构依赖关系的约束条件下，将工作流中的多个任务匹配到合适的云主机中，为了满足某种或多种服务质量需求目标而采取不同的调度策略。

2.3 多目标优化

多目标优化是一个涉及多个目标函数同时优化的数学优化问题的多准则决策领域。目前多目标优化已应用于许多科学领域，包括工程、经济学和物流等领域，这类问题常常需要在两个或多个相互冲突的目标之间存在权衡的情况下做出最优决策。

2.3.1 多目标优化问题

一般多目标优化问题（MOP，multi-objective optimization problem）可以总结为以下通用模型：

$$\begin{cases} \min/\max\ f_1(x) \\ \quad\cdots \\ \min/\max\ f_i(x) \\ \quad\cdots \\ \min/\max\ f_n(x) \end{cases} \tag{2.1}$$

$$s.t. \qquad g_j(x) \le 0, j = 1, \dots, m_1 \tag{2.2}$$

$$h_l(x) = 0, l = 1, \dots, m_2 \tag{2.3}$$

其中，$x = (x_1, \dots, x_n) \in X$是决策变量，$X$是变量空间，$f_i(x),\ (j = 1,2, \dots, k)$是目标函数，$g_j(x)$和$h_l(x)$一起被称作为约束函数。

元素$x^* \in X$被称作一个可行的解决方案或者一个可行的决策。可行解x^*的向量$z^* \coloneqq f(x^*) \in \mathbb{R}^n$称为目标向量或结果。在多目标优化问题中，通常不存在可同时最小化或最大化所有目标函数的可行解决方案。因此，帕累托最优解（Pareto optimal solutions）被赋予了更多的关注。

2.3.2 帕累托最优解

定义 2.3.1 在数学术语中，一个可行解$x^1 \in X$被称为帕累托支配另一个解$x^2 \in X$，即满足

对所有的$i \in \{1,2, \dots, n\}$，满足$f_i(x^1) \le f_i(x^2)$，且

对至少一个$j \in \{1,2, \dots, n\}$，满足$f_j(x^1) < f_j(x^2)$.

如果不存在其它的解支配$x^* \in X$，则x^*被称为帕累托最优（Pareto optimal）。帕累托最优结果的集合通常被称为帕累托前沿（Pareto front, or Pareto frontier）或者帕累托边界（Pareto boundary）。

多目标优化问题的帕累托前沿如果是有限的话，它们受到所谓的最低点目标向量z^{nad}和理想目标向量z^{ideal}的限制。其中，最低点目标向量被定义为

$$z_i^{nad} = \sup_{x \in X \ is \ Pareto \ optimal} f_i(x),\ \forall\ i = 1,2, \dots, n \tag{2.4}$$

同时，理想目标向量被定义为

$$z_i^{ideal} = \inf_{x \in X} f_i(x),\ \forall\ i = 1,2, \dots, n \tag{2.5}$$

换言之，最低点和理想目标向量的分量分别定义了帕累托最优解的目标函数值的上限和下限。在实践中，最低点目标向量只能近似，因为通常整个帕累托最优解集是未知的。另外，一个乌托邦的目标向量$z^{utopian}$用表示，即

$$z_i^{utopian} = z_i^{ideal} - \epsilon,\ \forall\ i = 1,2, \dots, n \tag{2.6}$$

其中，$\epsilon > 0$是个很小的常量，通常由于数值原因而定义。

图 2.2 就是一个使用帕累托最优的例子，横纵轴表示优化的目标函数，中间的点集是可行解集，其中红色的点即为帕累托前沿。由于多目标优化问题通常存在多个帕累托最优解，解决多目标优化问题的意义并不像传统的单目标优化问题那么简单。下一小节总结了针对该类问题求解的常用方法以及它们所适用的场景。

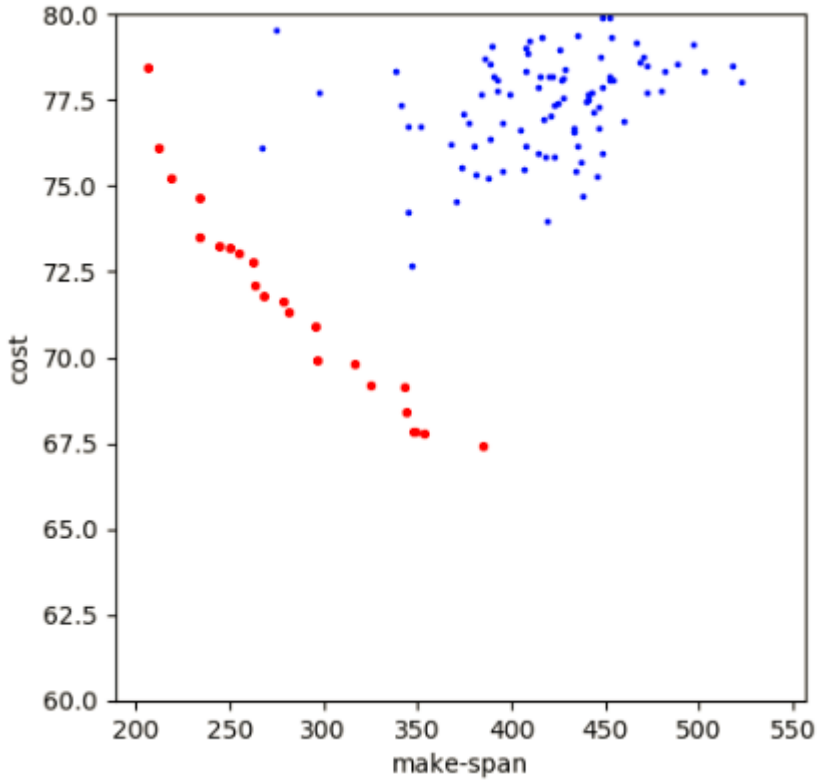


图 2.2 帕累托最优前沿例子

Fig 2.2 Example of a Pareto frontier

2.3.3 多目标优化方法

求解一个多目标优化问题有时被理解为近似或计算所有或一组具有代表性的帕累托最优解[39]。当强调做出决策时，解决多目标优化问题的目标是支持决策者根据其主观偏好找到最优选的帕累托最优解。其潜在的假设是，必须确定一个问题的解决方案，以便在实践中实施。在这种情况下，人类决策者扮演着重要的角色，决策者通常是相关问题领域的专家。多目标优化方法通常分为四类，包括无偏好方法、先验方法、后验方法以及交互方法。

① 无偏好方法

当决策者没有明确表达任何偏好信息时，多目标优化方法可以归类为无偏好方法。一个众所周知的例子是全局准则的方法，也就是将具有多个目标的原始问题转化为单目标优化问题，即一个形如标度化问题，如

$$\min \| f(x) - z^{ideal} \| \tag{2.7}$$

$$s.t. \quad x \in \mathrm{X} \tag{2.8}$$

被求解。在上述的问题中，$\|\cdot\|$可以是任意的L_p标准式，普遍的选择包括L_1, L_2以及L_∞。全局准则的方法对目标函数的缩放敏感，因此，建议将目标归一化为统一的无量纲标度。如果标度化处理得当，则可以保证得到的解为帕累托最优。

② 先验方法

先验方法要求在求解过程之前表达足够的偏好信息。众所周知的先验方法示例包括：效用函数方法，词典编排方法和目标规划方法。

1）在效用函数方法中，假设决策者的效用函数是可用的。映射u：$\mathrm{Y} \to \mathbb{R}$是一个效用函数，如果对所有的$y^1, y^2 \in Y$，如果决策者更偏好$y^1$而不是$y^2$，则满足$u(y^1) > u(y^2)$；并且如果决策者对$y^1, y^2$的选择持中立，则满足$u(y^1) = u(y^2)$。效用函数指定了一个决策向量的排序，一旦取得了$u$，就足以解决

$$\max u(f(x)) \tag{2.9}$$

$$s.t. \quad x \in \mathrm{X} \tag{2.10}$$

但实际上很难构建一个能够准确表示决策者偏好的效用函数，特别是在优化开始之前帕累托前沿是未知的。

2）词典编排方法假设目标可以按重要性排序。在不失一般性的情况下，我们可以假设决策者的目标函数的重要性顺序为f_1是最重要的，f_k是最不重要的。词典编排方法包括解决一系列形如单目标优化问题

$$\min f_l(x) \tag{2.11}$$

$$s.t. \quad f_j(x) \le y_j^*, j = 1,2,\dots,l-1 \tag{2.12}$$

$$x \in \mathrm{X} \tag{2.13}$$

其中，y_j^*是上述问题在$l = j$情况下的最优值。因此，$y_1^* := \min\{f_1(x) | x \in X\}$以及在上述问题中这种形式的每个新问题都会增加一个新的约束，当l按顺序从1到k时。需要注意的是，此时没有为任何目标指定目标值，这使它与词典目标规划方法不同。

3）目标规划是线性规划的扩展和泛化，旨在处理多个通常相互冲突的目标度量。这些措施中的每一项都有一个要实现的目标或目标值。当要实现的目标之间存在明确的优先级排序时，通常使用词典目标规划。它将不需要的偏差排序为若干优先级别，最高优先级别的偏差最小化比低优先级别的偏差更为重要。

③ 后验方法

后验方法旨在产生所有帕累托最优解或者帕累托最优解的代表性子集。大多数后验方法属于以下两类中的任何一种：基于数学规划的后验方法，该算法不断重复并且该算法的每次运行产生一个帕累托最优解。比较著名的算法包括正常边界交集（NBI），修正法向边界交集（NBIm）、法向量约束（NC）、连续帕累托（SPO）、定向搜索域（DSD）等方法。另外一种是进化算法，该算法的一次运行产生一组帕累托最优解集，该类方法中比较著名的算法有非支配排序遗传算法-II（NSGA-II）、多目标粒子群优化算法（MOPSO）、强度帕累托进化算法-2（SPEA-2）等。

进化算法的主要缺点是速度较慢，无法保证解的帕累托最优性。只知道所产

生的解决方案都不会占优其它解决方案。

④ 交互方法

在使用交互方法解多目标优化问题时，其求解过程是迭代的，并且决策者在搜索最优选的解决方案时会不断地与该方法交互。换句话说，决策者应该在每次迭代时表达偏好，以便获得决策者感兴趣的帕累托最优解，并了解可以获得哪种解决方案。以下是交互方法包含的主要步骤：

1）初始化；

2）生成一个帕累托最优起始点；

3）寻求决策者的偏好信息；

4）通过偏好生成新的帕累托最优解，并且将其解与问题的一些可能的其他信息传递给决策者；

5）如果生成了一些决策解，询问决策者选取到目前为止最好的解；

6）停止。

在数学优化方法中，人们常常将数学收敛作为一种停止准则，而在交互方法中，人们往往强调心理收敛。一般来说，当决策者确信他/她找到了最适合的解决方案时，方法就终止。

2.4 博弈论基本概念

2.4.1 博弈论简介

博弈论是对理性决策者之间策略交互的数学模型的研究[40]。所谓博弈，可以理解为一些个人或组织在一定的环境和规则下，同时或先后，一次或多次，从各自允许选择的行为或策略中进行选择并加以实施，各自取得相应结果的过程[41]。图 2.3 给出了按照不同的原则，常见的博弈分类情况。

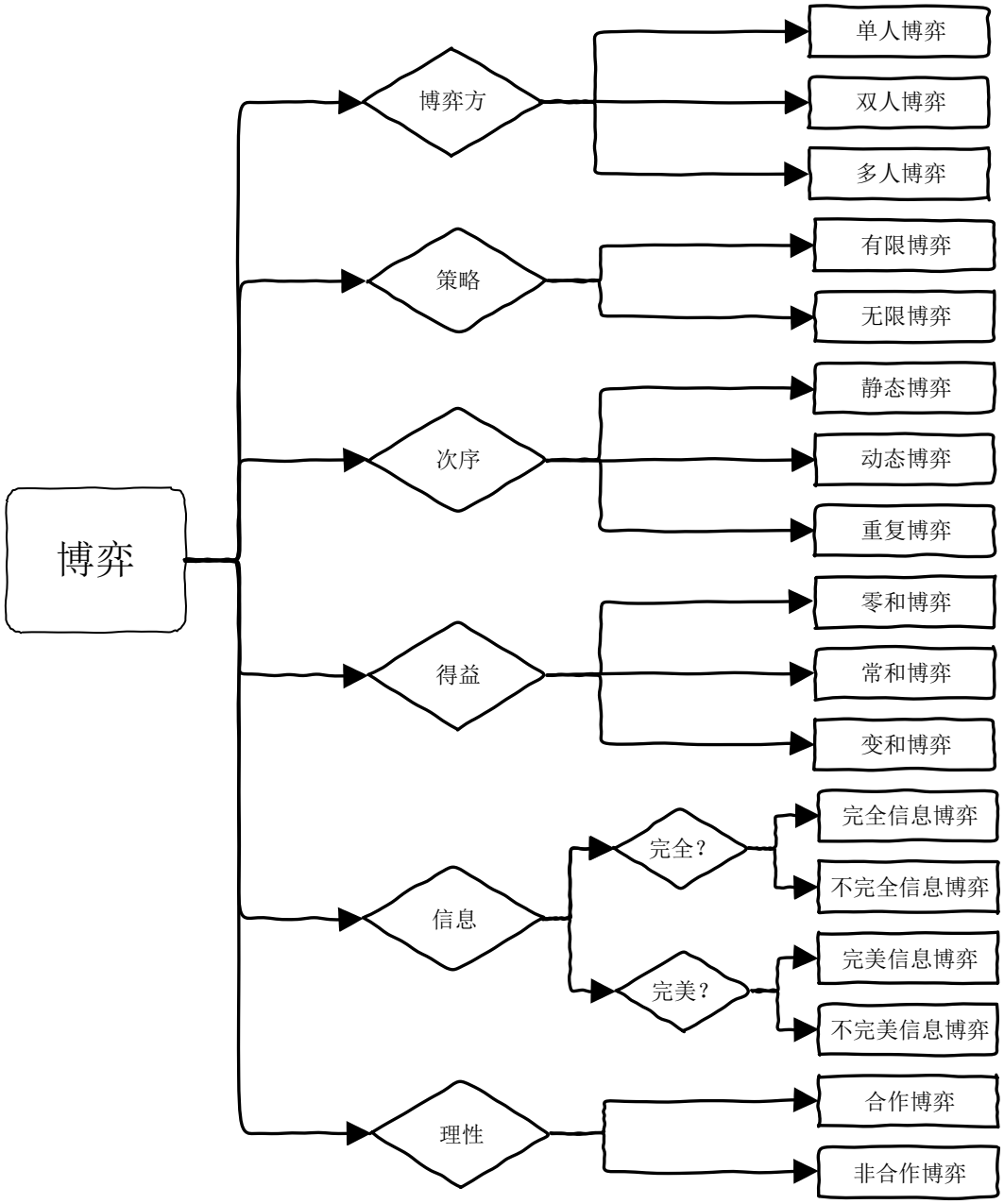


图 2.3 常见的博弈分类

Fig 2.3 Classification of common games

一般的，博弈的表现形式包括策略式、扩展式和特征函数式。现实中任何不同的博弈问题都可根据这三种表现形式进行更加复杂的分析和建模。

① 策略式博弈是最基础的一类博弈，也是博弈论最早研究的一类，因此也被称为标准式博弈。策略式博弈主要用来描述静态博弈，它由三个基本的要素组成，即博弈参与者集合、可选的策略集以及支付矩阵。

定义 2.4.1 一个（有限的）策略式博弈是一个三元组$\Gamma = (i \in I, S_i, u_i)$，其中

1) $I = \{1, \ldots, n\}$是博弈中参与者的集合，称为玩家；

2) S_i是玩家i单独的（有限的）可能采取的策略的集合，每当参与者i选定一个策略$s_i \in S_i$后，就形成了一个局势$\boldsymbol{s} = (s_1, \dots, s_n) \in \prod_{i \in N} S_i$；

3) $u_i(\boldsymbol{s}) = u_i(s_1, \dots, s_n)$是玩家$i$的支付函数，指定了它在局势$\boldsymbol{s}$收到的预期收益。

为了区分参与者i与其他参与者$-i$，对$\forall i \in I$，有$-i = I \backslash \{i\}, \boldsymbol{s} = (s_1, \dots, s_n) = (s_i, s_{-i})$，其中$s_{-i} = (s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n) \in S_{-i} = \prod_{k \neq i} S_k$. 从而$(t_i, s_{-i})$表示参与者$i$把局势$\boldsymbol{s}$中他的策略$s_i$换成$t_i$，其他参与者的策略不变得到新的局势。

② 扩展式博弈通常用来表示动态的或者不完全信息博弈，它可以用博弈树结构表示。其主要构成要素包括：博弈的参与者、树形图、行动集合、支付函数、信息集以及自然状态的概率分布。

定义 2.4.2 一个（有限的）扩展式博弈是一个六元组$\Gamma = (N, T, A_{1,\dots,n}, u_{1,\dots,n}, H_{1,\dots,n}, P)$，其中

1) $k \in N = \{1, \dots, n\}$是博弈中参与者的集合，称为玩家；

2) T是博弈的树形图结构；

3) $A_k \equiv \bigcup_{h_k \in H_k} A(h_k)$是玩家 k 在参考历史信息的情况下单独的（有限的）可能采取的行动的集合；

4) $u_k: A_1 \times \dots \times A_n \rightarrow \mathbb{R}$是玩家$k$单独的支付函数，指定了它因一次行动$a \in A_1 \times \dots \times A_n$而收到的预期收益；

5) H_k是玩家 k 在做决策时可参考的历史信息集；

6) P是自然状态下外生事件的概率分布。

③ 在合作博弈中，通过引入特征函数的概念来刻画联盟的总支付。特征函数式博弈包含两个要素：博弈的参与者集合和特征函数，因此记为$\Gamma = (n, v)$。

定义 2.4.3 设v是定义在 N 的子集族2^N上的实值函数，且满足条件，

1) $v(\emptyset) = 0$;

2) $\forall S, T \subseteq N, S \cap T = \emptyset$，有$v(S) + v(T) \leq v(S \cup T)$

则称v为特征函数[42]。

在这三种基础的表现形式上，可以根据不同的应用场景或问题，建立更加复杂而形式各异的博弈模型，对于更加复杂的分析，三种表现形式也可以相互转换。

2.4.2 博弈模型的解

在博弈论中，一个核心概念就是均衡，它也被称为博弈问题或模型的解。对不同类别的博弈，其均衡解也有不同的定义。

例如，被广泛讨论的纳什均衡概念，由数学家约翰·纳什（John Forbes Nash Jr）在他的博士论文中提出并命名。在非合作博弈中，每个博弈参与者都是选择对自己最有利的策略，进而最后形成大家都不愿去改变的稳定局势，即为纳什均衡局

势。

定义 2.4.4 设s^*是完全信息静态博弈$\Gamma = (i \in I, S_i, u_i)$的一个局势，如果

$$u_i(s_i, \boldsymbol{s}^*_{-i}) \leq u_i(\boldsymbol{s}^*), \forall i \in I, \forall s_i \in S_i \tag{2.14}$$

则称$\boldsymbol{s}^*$是Γ的一个纳什均衡局势，简称为纳什均衡。纳什均衡$\boldsymbol{s}^*$的第i个分量成为参与者i的均衡策略。也就是说，对于参与者i，纳什均衡$\boldsymbol{s}^* = (s_1^*, \dots, s_n^*)$是当其他参与者 j 都采取s_j^*时，参与者i为了得到最大利好，也只能选择均衡策略s_i^*，因此，各方的竞争在局势$\boldsymbol{s}^*$下达到一个稳定状态。

在此基础上，博弈论研究者们相继开发出多种均衡。比如对于完全信息静态博弈的混合策略有混合纳什均衡、在不完全信息静态博弈中发展出了（混合）Bayes-Nash 均衡，完全且完美信息动态博弈中的子博弈精炼纳什均衡，以及不完全信息动态博弈中定义的精炼 Bayes 均衡也被广泛讨论。

然而，在研究纳什均衡时，研究者们发现了一些有意思的现象，比如著名的“囚徒困境”、“信任困境”、“多重纳什均衡”等。为了进一步完善均衡理论，研究者循着问题发现规律，于是分别提出了集体理性与帕累托上策均衡、策略的风险与风险上策均衡、策略的多重性与聚点均衡，以及机制设计与相关均衡等均衡理论。相关均衡于 1974 年由数学家罗伯特·奥曼（Robert Auman）首次探讨，是比纳什均衡更一般的博弈论中的解的概念[43][44]，本文在第三章做了详细的讨论。

2.5 强化学习

强化学习是机器学习中介于监督学习和非监督学习之间的一个领域，强调如何基于环境行动，以取得最大化的预期收益[45]。强化学习在很多领域都有研究，比如博弈论、控制论、运筹学、信息论、仿真优化、群体智能、统计学领域等，都发挥着它的影响力。

强化学习的诞生离不开心理学的一些基本思想，比如效益法则和强化理论。效益法则为美国心理学家爱德华·桑代克（Edward Thorndike）经过猫的习得实验中得出的一种有机体普遍存在的趋利避害的规律，即“有机体跟随满意结果的反应，以后出现的概率会越来越大；相反，其出现概率会跟随着不满意结果的反应而减少”。而强化理论则来源于心理学中的行为主义理论，由美国心理学家、行为学家伯尔赫斯·弗雷德里克·斯金纳（B.F. Skinner）提出，他实施了称为斯纳金盒子（Skinner Box）的实验，通过提供（积极的）反馈来训练动物的行为，得到“有机体如何在环境给予的奖励或惩罚的刺激下，逐步形成对刺激的预期，产生能获得最大利益的习惯行为”的理论。这两种思想奠定了强化学习中“通过与环境的交互来学习”的基础。

此外，强化学习也糅合了控制理论中的一些概念。比如“设计一个控制器来最

小化一个动态系统的行为的一些指标”，与美国应用数学家理查德·贝尔曼（Richard E. Bellman）相关的“系统状态和值函数的使用”和著名的贝尔曼方程，以及动态规划的基本概念——“通过求解贝尔曼方程解最优控制问题”等。

这两条线在 1980 年代结合在一起，形成了现代强化学习领域。以下简要列举了强化学习的几个关键特点，包括：

① 学习者不会被告知将采取哪种行动；

② 试错（trial-and-error）的方法学习；

③ 延迟奖励的可能性——牺牲短期收益以求获取更大的长期收益；

④ 需要平衡探索和利用（exploration-exploitation trade-off）；

⑤ 介于监督学习和非监督学习之间的机器学习算法。

本文从最优控制理论、博弈论、机器学习问题的语境下，探讨了强化学习领域的不同学习环境，以及在单智能体和多智能体强化学习案例中的不同的算法。图 2.4 给出了强化学习根据不同的状态和智能体数目的学习环境分类，包括单个智能体单个状态（无状态或固定状态）的单个自动机、单个智能体多个状态的马尔可夫决策过程、多个智能体单个状态的策略式博弈以及多个智能体多个状态的马尔可夫博弈。本节重点讨论了单个智能体多状态的马尔可夫决策过程和多智能体多状态的随机的马尔可夫博弈学习环境中的强化学习。

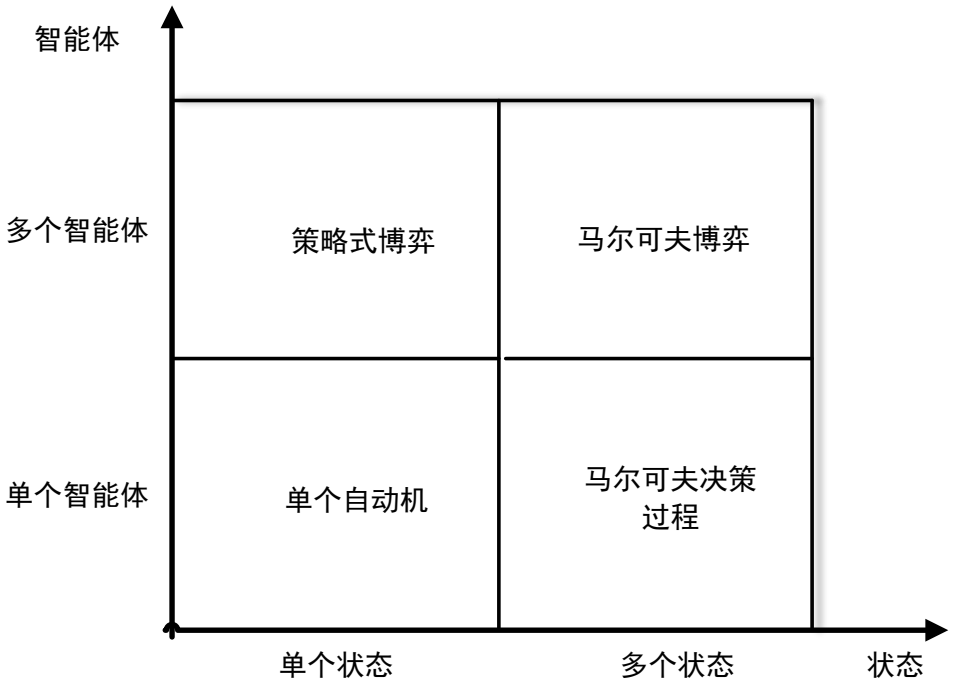


图 2.4 学习环境概述

Fig 2.4 Overview of the learning settings

2.5.1 单个智能体案例

在机器学习问题中，单个智能体强化学习的环境通常被建模为一个马尔可夫决策过程（MDP）。在博弈论的语境下，马尔可夫决策过程也可以看作为只有一个参与者或智能体的随机博弈。

在概率论和统计学中，马尔可夫决策过程是离散时间随机控制过程，它为结果部分随机且部分在决策者控制的情况下的决策建模提供了数学框架，以俄罗斯数学家安德雷·马尔可夫（Andrey Markov）命名。

定义 2.5.1 一个（有限的、贴现的）马尔可夫决策过程是一个五元组(S, A, T, R, δ)，其中

1) S是一个有限的状态集合；

2) A是一个有限的行动集合，A_s是来自状态s的可行的有限的行动集合；

3) $T: S \times A \times \mathrm{S} \to [0, 1]$表示概率转移函数；

4) $R: S \times A \times \mathrm{S} \to \mathbb{R}$是奖励函数；

5) $\delta \in [0, 1]$是贴现因子，表示未来（或预期）的奖励与现在的奖励之间的重要差异。

马尔可夫决策过程由状态集、行动集、状态之间的转移以及一个定义的奖励函数和贴现因子组成。环境状态集合S被定义为有限集合$\{s^1, \dots, s^N\}$，其中状态空间的大小为N，即$|S| = N$，每一个状态s对建模的问题状态中所有重要特征作唯一表征。行动集合A被定义为行动空间大小为K的有限集合$\{a^1, \dots, a^K\}$，$|A| = K$，行动集合可用于控制系统的状态。通过在状态$s \in S$中应用行动$a \in A$，系统根据从s到新状态$s' \in S$可能的转移集合上的概率分布做出一次转移。奖励函数R为所处的状态或者在状态中做出一些行动指定了奖励，同时它隐式地规定了学习的目标。T定义了一个在可能的下一个状态中的合适的概率分布。在状态s中进行行动a之后以状态s'结束的概率表示为$T(s, a, s')$，并且要求对所有的行动a，所有的状态s以及s'，满足$T(s, a, s') \in [0, 1]$；且对所有的状态s和行动a，有$\sum_{s' \in S} T(s, a, s') = 1$成立。另外，如果行动的结果不是依赖先前的行动以及访问过的历史状态，而仅仅是依靠当前状态，即满足

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \dots) = P(s_{t+1} = s'|s_t = s, a_t = a) = T(s_t, a_t, s_{t+1}) \quad (2.15)$$

则被控制的系统具有马尔可夫特性。图 2.5 给出了马尔可夫决策过程的一个例子，通常被描述成一个状态转移图，其中绿色的节点表示状态，红色的节点表示行动，黑色的边是转移概率，粉色箭头指向的数值是奖励值。

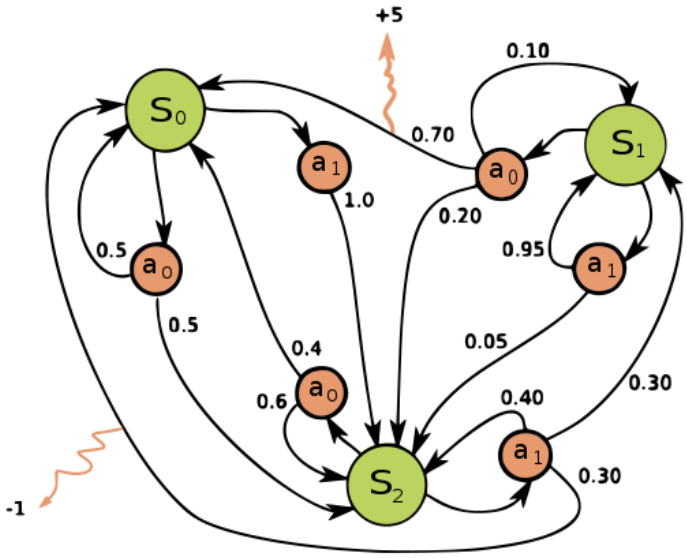


图 2.5　马尔可夫决策过程图例[46]

Fig 2.5 The example of Markov decision process

马尔可夫决策过程的核心问题是为决策者找到一个策略：函数π指定决策者在状态s将选择行动$\pi(s)$。因为在状态s被选择的行动完全由$\pi(s)$决定，并且由$P(s_{t+1}=s^{'}|s_t=s,a_t=a)$可推导至$P(s_{t+1}=s^{'}|s_t=s)$，所以产生的行为会表现得像一个马尔可夫链。对于无限期贴现的马尔可夫决策过程，策略π将最大化随机奖励的累积函数，通常是在潜在的无限期内的预期的贴现总和，

$$\sum_{t=0}^{\infty}\delta^t R_{a^t}(s_t,s_{t+1}),a_t=\pi(s_t) \tag{2.16}$$

任何给定的马尔可夫决策过程的目标都是找到一个最优策略π^*，以获得最大的奖励。

尽管动态规划算法能够在马尔可夫决策过程（MDPs）环境的完美模型存在的情况下计算最优策略，但是对于许多应用程序来说，模型可用的假设是很难保证的。相反，强化学习则主要关注在模型不可用或无模型（model-free）时如何获得最优策略，即不依赖先验已知的转移和奖励模型（MDPs 模型）的可用性，因此强化学习对无模型的 MDPs 问题的求解是更可取的方法，更具有实际应用意义。

在使用强化学习求解 MDPs 问题时，研究者们通过定义价值函数，将最优性指标与策略联系起来。以下给出了 MDPs 问题模型的最优策略推导过程。

$V^{\pi}(s)$是策略π下状态s的值，表示从状态s开始并且跟随策略π后的预期回报，其形式化描述为，

$$V^{\pi}(s)=E_{\pi}\{\sum_{k=0}^{\infty}\delta^k R_{t+k}|s_t=s\} \tag{2.17}$$

状态-行动值函数$Q:S\times A\rightarrow\mathbb{R}$被定义为从状态$s$开始，采取措施$a$，然后遵循

策略π的预期回报，

$$Q^{\pi}(s,a)=E_{\pi}\{\sum_{k=0}^{\infty}\delta^{k}R_{t+k}|s_t=s,a_t=a\} \tag{2.18}$$

价值函数的一个基本性质是满足某些递归性质。对于任何策略π和任何状态s，表达式(2.17)可以递归地定义为如下的贝尔曼（Bellman）方程，

$$\begin{aligned}V^{\pi}(s) &= E_{\pi}\{R_t+\delta R_{t+1}+\delta^2R_{t+2}+\cdots|s_t=t\}\\ &= E_{\pi}\{R_t+\delta V^{\pi}(s_{t+1})|s_t=s\}\\ &= \sum_{s'}T(s,\pi(s),s')(R(s,a,s')+\delta V^{\pi}(s'))\end{aligned} \tag{2.19}$$

方程(2.19)表明状态的预期值是根据直接的奖励、贴现因子以及可能的下一个状态的值定义的，这些值由它们的转移概率加权而得。可见，V^{π}是这一系列方程的唯一的解，尽管多种策略可能有相同的价值函数，但是对于一个给定的策略π，V^{π}是唯一的。

因此，最优策略意味着对所有的$s\in S$，最大化方程(2.17)的价值函数。更为形式化的表述为，一个最优策略π^*表示对所有的状态$s\in S$以及所有的策略π，满足$V^{\pi^*}(s)\geq V^{\pi}(s)$，令$V^*=V^{\pi^*}(s)$，可得出著名的贝尔曼最优化方程，

$$V^*(\mathrm{s})=\max_{a\in A}\sum_{s'\in S}T(s,a,s')(R(s,a,s')+\delta V^*(s')) \tag{2.20}$$

它指出，最优策略下的状态值必须等于该状态下最佳的行动的预期回报。要在给定最优状态值函数V^*的情况下选择一个最优行动，可应用以下规则：

$$\pi^*(\mathrm{s})=\arg\max_{a}\sum_{s'\in S}T(s,a,s')(R(s,a,s')+\delta V^*(s')) \tag{2.21}$$

类似的，状态-动作值函数为，

$$Q^*(s,a)=\sum_{s'}T(s,a,s')(R(s,a,s')+\delta\max_{a'}Q^*(s',a')) \tag{2.22}$$

Q函数是有用的，因为它们使不同方案的加权求和不需要使用转换函数。在一个状态下，不需要向前推理步骤来计算最优动作。这就是为什么在无模型（model-free）方法中，即在T和R未知的情况下，学习Q函数而不是V函数的原因。Q^*和V^*之间的关系由下式给出，

$$Q^*(s,a)=\sum_{s'\in S}T(s,a,s')(R(s,a,s')+\delta V^*(s')) \tag{2.23}$$

$$V^*(\mathrm{s})=\max_{a}Q^*(s,a) \tag{2.24}$$

类似的，最优行动的选择可简单的表示为，

$$\pi^*(\mathrm{s})=\arg\max_{a}Q^*(s,a) \tag{2.25}$$

即最优行动是根据采取该行动后可能出现的下一个状态，具有最高预期效用的行动。

在强化学习中，有两种常用的方法用于求解上述贝尔曼方程以找到 MDPs 问题的最优策略。它们分别是通过反复地改进值函数来获取最优策略的“值迭代”方法和通过反复评估和改进策略来学习最优策略的“策略迭代”方法。这两种方法被证实都可以收敛于最优值函数Q^*。

此外，在强化学习中，一个关键的问题是，智能体不能简单地执行当前估计回报最高的最优策略（目标策略），还需要尝试新的与环境交互产生数据的策略（行为策略），以尝试发现更好的策略，这个问题通常被称为探索-利用权衡问题（exploration-exploitation trade-off）。根据行为策略和目标策略是否一致，常分为on-policy 和 off-policy 学习方法。

Q-learning 算法是 off-policy 算法的一个典型例子。它采用随机逼近技术对最优值Q^*进行连续估计，使用融合了蒙特卡洛和动态规划的时间差分法学习算法，在贝尔曼方程的基础上对马尔可夫决策过程求解最优策略。在每个时间步，系统都处于特定的状态s，Q-learner 执行由其控制策略选择的行动a。在执行动作a后，Q-learner 观察其即时奖励R和新系统状态s'。然后使用表达式(2.26)更新状态-动作对(s,a)的Q值估计，即

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha[R + \delta \max_{a'} Q\,(s',a')] \tag{2.26}$$

其中$\alpha \in (0,1)$表示 Q-learner 的学习率，$\delta \in [0,1]$是贴现因子。在特别一般的情况下，Q-learning 算法的估计值会收敛于Q^*。事实上，只要它不断更新所有状态-动作对，就可以使用任何行为策略。为了满足理论上的收敛要求，控制策略需要无限频繁地访问每个状态-动作对。然而，在实践中，通常会测试Q值是否稳定，并随着算法的进行相应地减少探索量。因为 Q-learning 算法用于生成行动的行为策略不需要与算法所学习的目标策略相对应，所以它是典型的 off-policy 学习算法。尽管 off-policy 确保了数据的全面性，所有的行为都能覆盖并具有良好的通用性，但同时也带来了收敛慢的问题。

on-policy 是 off-policy 学习的一种特殊情况，智能体使用的行为策略与正在学习的目标策略相同。比如在 SARA 算法中，基于当前的策略直接执行一次动作选择，然后用这个样本更新当前的策略。该方法带来到问题是仅仅利用目前已知的最优选择，可能学不到最优策略，容易收敛到局部最优，而加入探索又降低了学习效率。

为了解决探索-利用矛盾，ϵ-greedy 探索机制在强化学习中被广泛使用。它以较大的概率$(1-\epsilon)$去选择现在最好的动作，如果没有选择最优动作，就在剩下的动作中随机选择一个。ϵ是一个可调参数，更小的ϵ意味着算法会更加贪心，如果ϵ逐渐衰减，则 Q-learning 和 SARA 的结果都收敛到近似最优解。表 2.2 罗列了强化学习中部分经典的算法，并根据其策略、行动空间和状态空间的连续性以及运算子做了简要的比较。

表 2.2 强化学习算法比较[45]

Table 2.2 Comparison of reinforcement learning algorithms

算法	策略	行动空间	状态空间	运算子
Monte Carlo	off-policy	离散的	离散的	样本均值
Q-learning	off-policy	离散的	离散的	Q 值
SARSA	on-policy	离散的	离散的	Q 值
Q-learning-λ	off-policy	离散的	离散的	Q 值
SARSA-λ	on-policy	离散的	离散的	Q 值
DQN	off-policy	离散的	连续的	Q 值
DDPG	off-policy	连续的	连续的	Q 值
A3C	off-policy	连续的	连续的	Q 值
NAF	off-policy	连续的	连续的	优势
TRPO	on-policy	连续的	连续的	优势
PPO	on-policy	连续的	连续的	优势

2.5.2 多个智能体案例

随着多智能体系统不断增长的需求，如机器人组织、分散网络路由、分布式负载均衡、电子竞拍和流量控制等领域，加上处理交互学习者或智能体的复杂性，导致了多智能体强化学习领域的发展。在这种系统下，每个智能体的最佳策略不仅取决于环境，还取决于其他智能体的策略。因此，它面临的挑战除了继承了来自于单个智能体强化学习的维度灾难和探索-利用权衡问题外，还包括智能体难以确定学习目标、学习问题的非稳定性以及协调的需要等困难[47]。

将马尔可夫决策过程引入多个智能体，其学习环境变成了一个马尔可夫博弈。表 2.3 给出了基于智能体之间策略相互作用的多智能体强化学习方法的概述。列出的技术根据它们在多智能体系统中学习时的适用性和信息种类进行分类。研究者们将博弈的场景主要区分为无状态博弈技术和马尔可夫博弈技术，前者侧重于在假设环境是静止的情况下处理多智能体之间的交互，后者同时处理多智能体交互和动态环境。表 2.3 还展示了智能体用于学习的信息：独立学习者只根据他们自己的奖励观察来学习，而联合行动学习者也使用行为观察和其他智能体可能的奖励。

表 2.3 多智能体强化学习方法概述[48]

Table 2.3 Overview of MARL approaches.

		博弈设置		
		无状态博弈	团队马尔可夫博弈	一般马尔可夫博弈
强化学习	Independent Learner	Stateless Q-learning Learning Automata IGA FMQ Commitment Sequences Lenient Q-learners	Policy Search Policy Gradient	MG-ILA (WOLF-)PG Learning of Coordination Independent RL CQ-learning
	Joint Action Learner		Distributed-Q Sparse Tabular Q Utile Coordination	Nash-Q Friend-or-Foe Q Asymmetric Q Joint Action Learning Correlated Q

2.6 本章小结

本章首先介绍了云计算的基本概念，包括云计算的定义、三种标准的服务模式和五个基本特征。接着简要介绍了云环境下的工作流的背景和基础概念，并概述了五种著名的科学工作流。然后对云环境下的多工作流-多目标优化调度问题中涉及到的多目标优化、博弈论以及强化学习的基础知识进行了详细的讲解。这些都是本文进行云工作流调度问题建模、模型求解以及算法设计所需要的基础知识，包括多目标优化问题数学模型、帕累托最优解及多目标优化方法，博弈模型的三种表现形式及均衡解，以及马尔可夫决策过程环境下的单智能体强化学习和多智能体强化学习方法概述，为下文的展开打下了坚实的理论基础。

3　云工作流调度问题及其建模

3.1 云工作流调度问题

3.1.1 工作流模型

单个工作流用一个有向无环图（DAG），$W=(T,E)$表示。其中$T=\{t_1,t_2,\dots,t_n\}$是一组n个任务（或应用程序）的集合，E是边的集合，由一组优先级依赖关系项组成。每个任务t_i代表一个单独的应用程序，其中某个任务在一台计算实例上的运行时间表示为rt_i。优先级依赖项$e_{ij}=(t_i,t_j)$表示t_j仅在完成t_i并且接收到来自t_i的数据之后才开始执行。依赖项e_{ij}的起点和终点分别称为父任务和子任务。工作流分别由进入（entry）任务开始和退出（exit）任务结束。如果原始工作流具有多个进入/退出而不是单个进入/退出任务，则可以添加具有零执行时间的虚拟进入/退出任务作为唯一进入/退出任务。图 3.1 给出了一个简易的工作流模型的表现形式，如下所示：

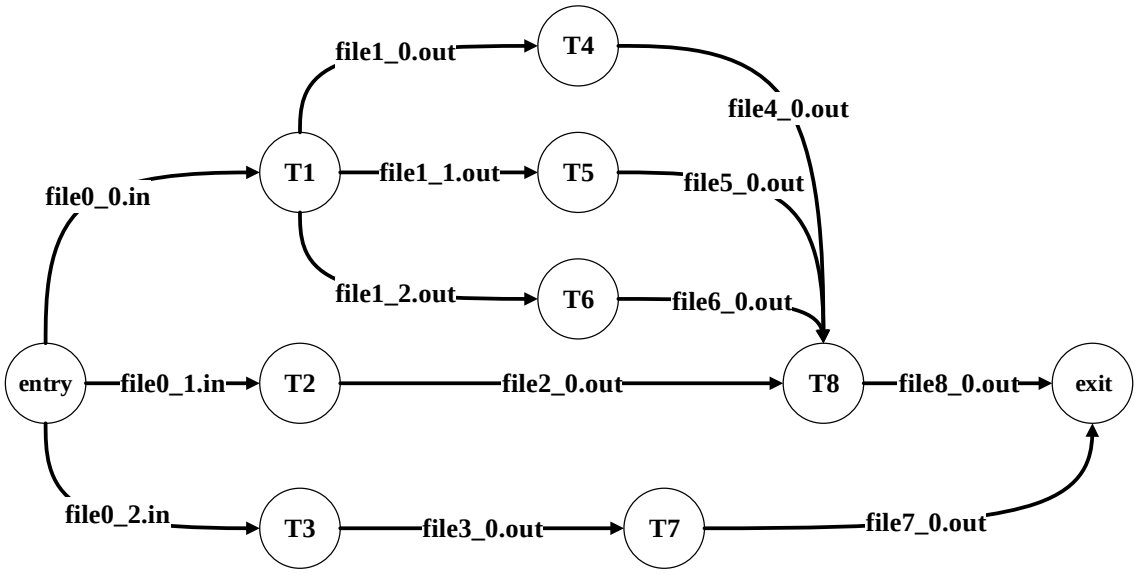


图 3.1 工作流模型——DAG 图

Fig 3.1 The model of workflow - DAG

在本文中，我们研究了多工作流调度的情况，并且被调度的每个工作流不仅在拓扑结构上存在差异而且其任务节点也来自不同的任务类型，即多个不同的 DAG 图同时被调度的情况。

3.1.2 云资源模型

云资源模型是一系列异构的虚拟云主机集合，$V=\{V_1,V_2,\dots,V_m\}$。云主机的异

构性体现在不同的云主机类型，包括其虚拟 CPU、内存、可用区以及服务单价等参数有差异性，记$V_j = (\text{vType}, \text{vCPU}, \text{Memory}, \text{Availability Zone}, \text{unit price})_j$为第$j$台云主机的配置参数。为了简化云资源模型，本文将云主机的vCPU、Memory和Availability Zone三个配置参数用性能分数（performance score）代替，于是有$V_j = (\text{vType}, \text{performance score}, \text{unit price})_j$。当$V_j$执行不同的任务$t_i$时，其性能分数也存在差异，因此$\text{performance score}_j = (ps_{t_1}, \dots, ps_{t_n})$，任务的执行时间与性能分数直接相关。在云资源的配置策略上，考虑了弹性资源池，云主机的服务单价按照按需付费的价格模式，并且假设云资源供应或取消供应不存在传输时间延迟。

3.1.3 多 QoS 指标优化调度问题

在本文中，我们将 IaaS 云视为多个工作流的支持平台。IaaS 云为执行多个工作流的任务提供了许多具有不同资源和定价配置的异构虚拟云主机。为了促进基于多智能体强化学的多目标优化调度方法的发展，本文提出了下述假设条件：

① 每一个任务只能被一台云主机执行；

② 任务的运行时间是任务开始与结束之间的时间间隔；

③ 不考虑资源供应或取消供应的延迟时间；

④ 不考虑任务之间传输的延迟时间。

本文考虑了两个 QoS 指标，即最大完成时间（make-span）和总花费（cost），作为云工作流调度的目标，是一个双目标优化问题，其优化问题可以形式化表述为：

$$\min f_1 = makespan = \max\{FT(V_j, n_{k,i}) * x_{k,i,j}\} \tag{3.1}$$

$$\min f_2 = cost = \sum_{k=1}^{M} FT(V_j, n_{k,i}) * x_{k,i,j} * p_j \tag{3.2}$$

$$s.t. \quad i \in [1, n_k], j \in [1, M], k \in [1, K] \tag{3.3}$$

$$FT(V_j, n_{k,i}) = st_{k,i,j} + rt_{k,i,j}, FT(V_j, n_{k,i,j}) \geq 0 \tag{3.4}$$

其中，f_1，f_2分别表示量化的最大完成时间和总花费。$n_{k,i}$表示工作流中的任务，p_j表示云主机V_j的服务单价，$x_{k,i,j}$是一个布尔值：当任务$n_{k,i}$被分配到V_j上执行时，$x_{k,i,j} = 1$；否则，$x_{k,i,j} = 0$。$FT(V_j, n_{k,i})$代表 V_j上执行完$n_{k,i}$之后的完成时间，由起始时间$st_{k,i,j}$和运行时间$rt_{k,i,j}$确定。

3.2 基于马尔可夫博弈的云工作流调度模型

针对上述双目标优化问题，本文以马尔可夫博弈为形式化和量化描述工具，对 IaaS 云环境下动态的工作流调度过程进行形式化建模，分析该博弈模型下的相关均衡以及机制设计，实现对 IaaS 云上的多工作流-多 QoS 指标优化的调度过程的精确形式化描述。

3.2.1 基于马尔可夫博弈的云工作流调度模型

马尔可夫博弈可以看作马尔可夫决策过程在多智能体情形的延伸，其中联合行动是多智能体选择一个行动的结果。

定义 3.2.1[49]一个（有限的，贴现的）马尔可夫博弈是一个元组$\Gamma^{\delta} = (i \in I, S, A, R, P)$，其中，

① I是一个有限的参与者或智能体的集合；

② S是一个有限的状态集合；

③ $A = \prod_{i \in I, s \in S} A_i(s)$，其中$A_i(s)$表示智能体$i$在状态 s 的有限的纯策略集合。为了区分智能体i与其他智能体$-i$，我们定义$A(s) \equiv \prod_{i \in I} A_i(s)$且$A_{-i}(s) = \prod_{j \neq i} A_j(s), A(s) = A_{-i}(s) \times A_i(s)$；$a = (a_{-i}, a_i) \in A(s)$，其中$a_i \in A_i(s)$且$a_{-i} \in A_{-i}(s)$；状态-行动对集合$\mathcal{A} = \bigcup_{s \in S} \bigcup_{a \in A(s)} \{(s, a)\}$。

④ R: $\mathcal{A} \to [\alpha, \beta]^I$，其中$R_i(s, a) \in [\alpha, \beta]$表示智能体$i$在状态 s 以及在行动策略为$a \in A(s)$的奖励；

⑤ P是一个转移概率系统，即对所有的$s \in S, a \in A(s)$，有$P[s'|s, a] \geq 0$且$\sum_{s' \in S} P[s'|s, a] = 1$；我们将$P[s'|s, a]$解释为给定当前状态$s$以及当前行动策略$a$，到下一个状态$s'$的概率；

⑥ $\delta \in [0, 1]$表示贴现因子。

我们将上述工作流调度过程看成是一个有两个智能体的马尔可夫博弈，两个调度目标，即最大完成时间（make-span）和总花费（cost），被分别抽象成两个智能体。假设在时间步t，每个智能体都可以观察彼此的行动和奖励，然后他们选择联合分布π_{s^t}，即所有智能体的行动选择组合。每个智能体进一步决定在一个行动a_i^t，并且产生可执行的纯策略行动集合$a^t = (a_1^t, \ldots, a_I^t)$。基于当前状态$s^t$和动作策略$a^t$，每个参与者得到奖励$R_i(s^t, a^t)$并且系统伴随着转移概率为$P[s^{t+1}|s^t, a^t]$演变到状态$s^{t+1}$。上述过程在$t+1$时间步重复。状态空间$S$由当前可用的云主机以及紧接着那些在之前的状态已经被映射到目标云主机的任务之后的后续的任务来决定。动作空间A由某个任务映射到某台云主机的映射概率组成。奖励函数$R: A \to \mathbb{R}^I$由公式(3.1)和(3.2)推导得出。值得注意的是，调度的性能直接受到奖励机制的影响，同时这些奖励机制伴随着多个智能体的交互作用。

3.2.2 博弈模型的相关均衡

在博弈问题模型中，均衡策略集是具有稳定结果的解决方案。相较于纳什均衡，本文考虑了更加一般的“相关均衡”。它满足多个参与者的策略之间的依赖关系，即在行动的联合分布上，任何参与者都不会单方面偏离。

给定由博弈参与者集合$i \in I$，每个参与者i的行动集合A_i以及效用函数u_i组成的策略式博弈$\Gamma = (i \in I, A_i, u_i)$。对于参与者$i$，一次策略调整表示为函数$\phi_i: A_i \to A_i$。

也就是说，在实施行动a_i时，ϕ_i告诉参与者i通过实行$\phi_i(a_i)$来调整他或她的行为。

定义 3.2.2 设(Ω,π)是一个可数的概率空间。对每个参与者i，设P_i为的信息分区，q_i是i的后验概率且$s_i:\Omega\to A_i$，在i的信息分区的相同的单元格中为状态指定相同的值。如果对所有的参与者i以及每一种策略调整ϕ_i满足，

$$\sum_{\omega\in\Omega} q_i(\omega)u_i\big(s_i(\omega),s_{-i}(\omega)\big)\geq\sum_{\omega\in\Omega} q_i(\omega)u_i\big(\phi_i\big(s_i(\omega)\big),s_{-i}(\omega)\big) \tag{3.5}$$

则$\big((\Omega,\pi),P_i,s_i\big)$是策略式博弈$\Gamma=(i\in I,A_i,u_i)$的一个相关均衡。换句话说，如果没有参与者能通过策略调整改进他或她的期望效用，则$\big((\Omega,\pi),P_i,s_i\big)$为一个相关均衡。相关均衡通常和机制（即策略调整）设计共同讨论。

本文试图找到一种固定策略$\pi\in\prod_{s\in S}\Delta(A(s))$，该策略满足使基于马尔可夫博弈的云工作流调度问题模型的解收敛于相关均衡。为了将相关均衡概念适配到本文的马尔可夫博弈模型上，令$T^{\pi}_{s,s'}$是状态s到状态s'的转移概率，其动作策略根据智能体的分布π_s遵循

$$T^{\pi}_{s,s'}=\ \sum_{a\in A(s)}\pi_s(a)P[s'|s,a] \tag{3.6}$$

在t时间步之后，从状态s到状态s'的转移概率表示为$\big(T^{\pi}_{s,s'}\big)^t$，值函数$V_i^{\pi}(s)$表示智能体$i$的期望奖励，满足

$$V_i^{\pi}(s)=(1-\delta)\sum_{t=0}^{\infty}\sum_{s'\in S}\delta^t\big(T^{\pi}_{s,s'}\big)^t\sum_{a\in A(s')}\pi_{s'}(a)R_i(s',a) \tag{3.7}$$

Q-值函数$Q_i^{\pi}(s,a)$表示如果智能体i的行动策略a是在状态s以及策略π下实施的期望奖励值，满足

$$Q_i^{\pi}(s,a)=(1-\delta)(R_i(s,a)+\delta*$$
$$\sum_{s'\in S}P[s'|s,a]\,(\sum_{t=0}^{\infty}\sum_{s''\in S}\delta^t\big(T^{\pi}_{s',s''}\big)^t\sum_{a\in A(s'')}\pi_{s''}(a)R_i(s'',a))) \tag{3.8}$$

归一化常数$1-\delta$确保V_i^{π}和Q_i^{π}的范围均落在$[\alpha,\beta]$中。根据表达式(3.7)和(3.8)以及马尔可夫特性的推导，可得到定理 3.2.3。

定理 3.2.3 给定一个马尔可夫博弈Γ^{δ}，对任意的$V:S\to[\alpha,\beta]^I$，任意的$Q:\ \mathcal{A}\to[\alpha,\beta]^I$，以及任意的固定策略$\pi$，有$V=V^{\pi}$，$Q=Q^{\pi}$成立，当且仅当对所有的$i\in I$有，

$$V_i(s)=\sum_{a\in A(s)}\pi_s(a)Q_i(s,a) \tag{3.9}$$

$$Q_i(s,a)=(1-\delta)R_i(s,a)+\delta\sum_{s'\in S}P[s'|s,a]\,V_i(s') \tag{3.10}$$

为了代替表达式(3.7)和(3.8)，递归地定义了V_i^{π}和Q_i^{π}作为满足表达式(3.9)和(3.10)的唯一函数对。

令$\pi_s(a_i)=\sum_{a_{-i}\in A_{-i}(s)}\pi_s(a_{-i},a_i)$，对任意的$\pi_s(a_i)>0$有$\pi_s(a_{-i}|a_i)=\frac{\pi_s(a_{-i},a_i)}{\pi_s(a_i)}$。

定理 3.2.4 给定一个马尔可夫博弈Γ^{δ}，一个固定的策略π是一个相关均衡，当且仅当对所有的$i\in I$，对所有的$s\in S$，具有$\pi_s(a_i)>0$的所有的$a_i\in A_i(s)$，以及所有的$a_i'\in A_i(s)$，满足

$$\sum_{a_{-i}\in A_{-i}(s)}\pi_s(a_{-i}|a_i)Q_i^{\pi}\big(s,(a_{-i},a_i)\big)\geq\sum_{a_{-i}\in A_{-i}(s)}\pi_s(a_{-i}|a_i)Q_i^{\pi}\big(s,(a_{-i},a_i')\big) \tag{3.11}$$

也就是说，在状态s，当智能体i被推荐执行a_i，则它更倾向于执行a_i，因为对所有的a_i'，a_i的期望效益大于或等于a_i'的期望效益。

如果除了智能体i之外的所有其他智能体$-i$都是根据固定策略π来执行动作，那么从智能体i的角度看，它的环境是一个马尔可夫决策过程。因此，这种 MDPs 的博弈偏差原理建立了定理 3.2.4。在定理 3.2.4 的基础上，可进一步得出推论 3.2.5。

推论 3.2.5 给定一个马尔可夫博弈Γ^{δ}，一个固定的策略π是一个相关均衡，当且仅当对所有的$i \in I$，$s \in S$，以及所有的$a_i, a_i' \in A_i(s)$，满足

$$\sum_{a_{-i} \in A_{-i}(s)} \pi_s(a_{-i}, a_i) Q_i^{\pi}\left(s, (a_{-i}, a_i)\right) \geq \sum_{a_{-i} \in A_{-i}(s)} \pi_s(a_{-i}, a_i) Q_i^{\pi}\left(s, (a_{-i}, a_i')\right) \quad (3.12)$$

表达式(3.12)是表达式(3.11)乘上$\pi_s(a_i)$。由于$\pi_s(a_{-i}, a_i)$是未知的，因此$Q_i^{\pi}\left(s, (a_{-i}, a_i)\right)$也是未知的，并且公式(3.12)是一个非线性不等式系统。

需要说明的是，当状态空间、行动空间和策略空间是在欧式空间的子集，即为非空、紧的凸集的前提条件下，可通过建立“不动点”的方法证明推论 3.2.5 中的相关均衡的存在性[49]。

3.3 本章小结

本章首先给出了云工作流模型，包括工作流应用模型、云资源模型和多 QoS 指标优化调度问题的形式化数学描述。接着在多 QoS 指标优化调度的数学模型的基础上，将复杂而具体的多目标工作流调度问题转换为抽象精炼和具有状态概率转移的动态博弈过程的形式化模型，构建出基于马尔可夫博弈的云工作流调度模型，并分析了该博弈模型下的相关均衡概念及推导过程。为下一章的系统模型及算法设计提供了理论分析依据。

4　基于 DQN 的多智能体强化学习的云工作流调度方法

4.1 系统模型

为了求解上述马尔可夫博弈模型，我们考虑了一个基于 DQN 算法的自适应多智能体强化学习（DQN-based MARL）框架，其总体框架描述如图 4.1 所示。与随机的马尔可夫博弈中求解纳什均衡的算法不同，本文讨论了该博弈模型下的机制设计与相关均衡，DQN-based MARL 学习框架的目标是通过智能体与环境及其他智能体的交互，动态地自我学习和自我优化策略，不断迭代使最终收敛于稳定的相关均衡策略，而不需要大量的先验或后验的专家知识和人为干预。

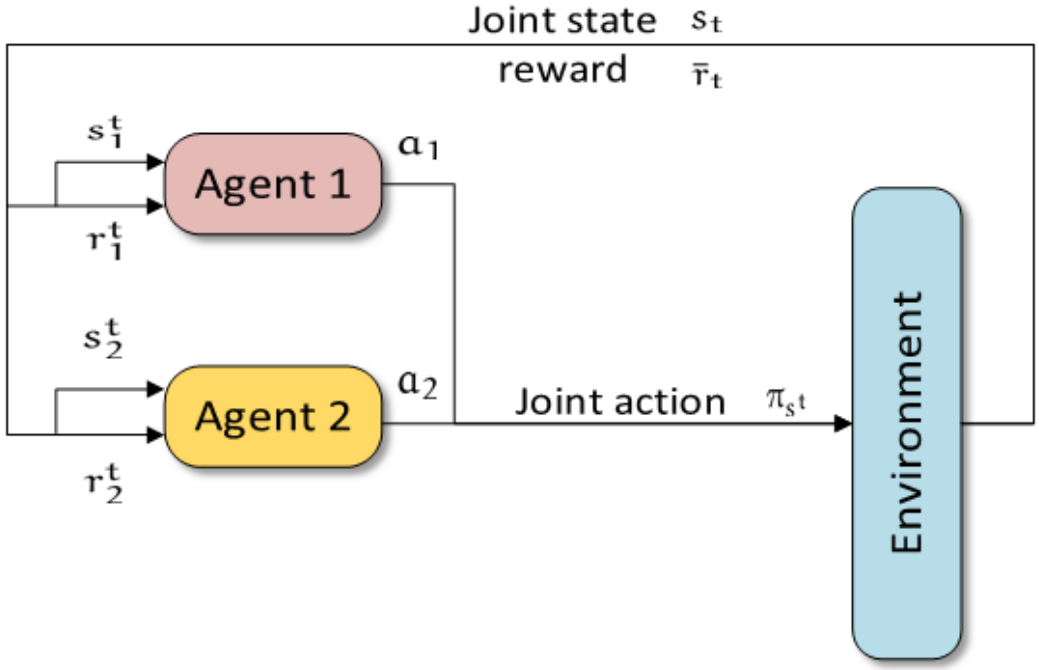


图 4.1 DQN-based MARL 模型

Fig 4.1 Overview of DQN-based MARL framework for multi-objective workflow scheduling

结合马尔可夫博弈模型进一步分析，本文将其调度问题模型中的两个优化目标，即最大完成时间（make-span）和花费（cost）分别抽象为两个智能体，即图 4.1 中的 Agent-1 和 Agent-2，每一个智能体都是基于 DQN 算法的智能体，通过与环境和其他智能体的交互进行自适应学习和自我优化过程；智能体的学习环境则对应云工作流调度问题下的马尔可夫博弈模型。在每个时间步$t = 1,2,\ldots$，每个智能体可以观察到当前所有的状态$\forall\ i \in I, s_t \in S$，多个智能体都是根据一种联合行动$\pi_{s^t}$学习，不断地优化调整直到收敛于相关均衡策略。用于计算全局均衡策略的

迭代算法在每个状态基于本地更新Q值和策略。通常在时间步t给出Q值，即对于所有$i \in I$，对于所有$s \in S$，并且对于所有$a \in A(s)$，有$Q_i^t(s,a)$。

4.2 DQN 算法

DQN[50][51]算法是深度强化学习领域中的一种流行方法。如图 4.2 所示，它的主要模块包括：环境模块、损失函数、经验重放模块和两种结构完全相同但参数不同的神经网络，即估计值网络和目标值网络。DQN 算法通过最小化损失来学习与最优策略相对应的动作值函数Q^*，

$$\mathcal{L}(\theta) = \mathbb{E}_{s,a,r,s'}[(Q^*(s,a|\theta) - y)^2] \tag{4.1}$$

$$y = r + \delta \max_{a'}(Q^*(s',a'|\theta^-)) \tag{4.2}$$

其中，y是目标Q函数，其参数周期性地用最新的θ更新，这有助于稳定学习。

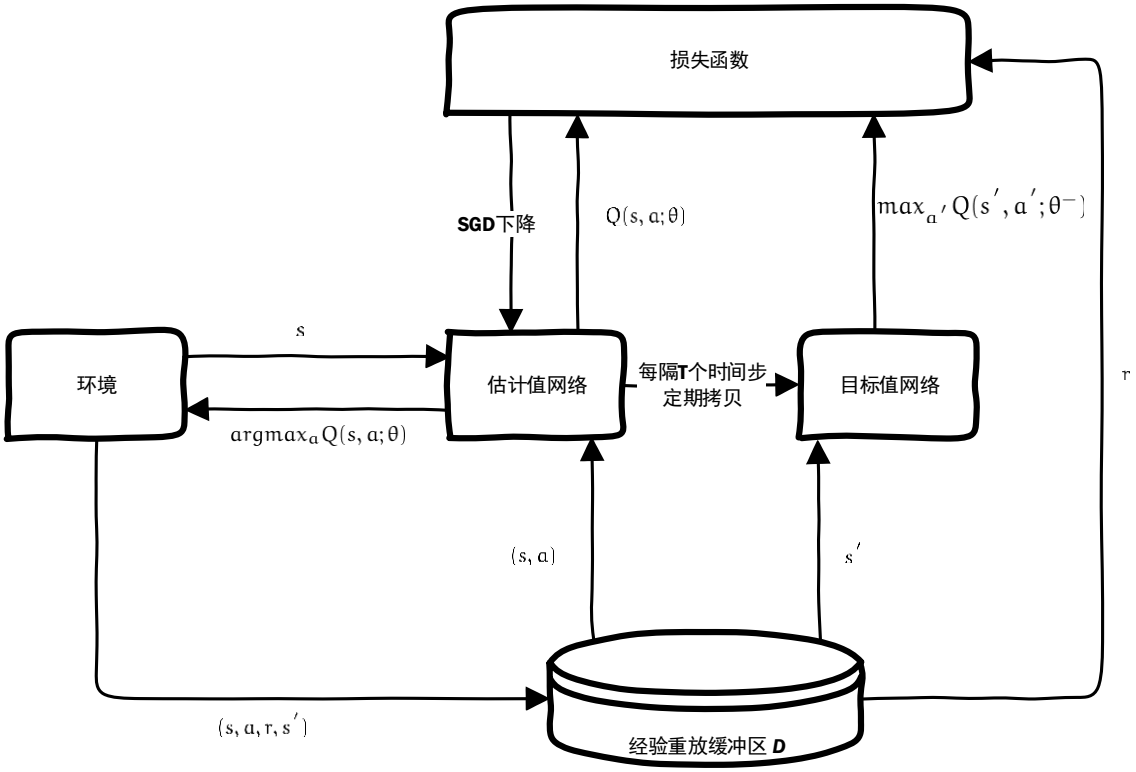


图 4.2 Deep Q-Network 算法框架

Fig 4.2 algorithm framework of Deep Q-Network

DQN 与 Q-learning 都是基于值迭代的强化学习算法，相较于后者使用 Q-table 存储每个状态-动作对的Q值，DQN 用神经网络提取复杂特征分析产生Q值。估计Q值网络用于预测Q估计值，它的输入来自于当前环境最新的参数，每次迭代参数都会更新，用θ表示其权重，$Q^*(s,a|\theta)$表示当前估计值网络的输出；目标Q值网络的

输入参数每隔一段时间才会更新，通常用θ^-表示它的权重比估计Q值网络更新得慢一些，$\max_{a'}(Q^*(s',a'|\theta^-)$表示目标值网络的输出。神经网络的训练目标是最优化由这两种Q值构造的损失函数，接着通过反向传播使用随机梯度下降的方法来更新估计Q值网络的参数，每隔一定次数的迭代，会将估计Q值网络的参数定期拷贝给目标Q值网络。使用一组较旧的参数生成目标会在对Q进行更新的时间和更新影响目标y的时间之间增加一个延迟，在一定程度上降低了估计Q值和目标Q值的相关性，使发散或振荡变得更加不可能，因此提高了算法的稳定性。

稳定 DQN 算法的另一个关键组成部分是使用了包含元组(s,a,r,s')的经验重放缓冲区D。在每一个时间步，智能体与环境交互得到的数据样本存储在经验缓冲区，要训练时再随机拿出一块 minibatch 大小的部分数据来训练，也打乱了训练样本的相关性。下列给出了单个智能体在 MDP 环境下的 DQN 的算法的经典伪代码。

算法 4-1 具有经验重放池的深度 Q-learning 算法

Algorithm 4-1 deep Q-learning with experience replay

Initialize replay memory *D* to capacity *N*
Initialize action-value function Q with random weights θ
Initialize target action-value function $\bar{Q}$ with weights $\theta^- = \theta$
For *episode=1, M* **do**
    Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
    For *t=1, T* **do**
        With probability ϵ select a random action a_t
        otherwise select $a_t = argmax_a Q(\phi(s_t), a; \theta)$
        Execute action at in emulator and observe reward r_t and s_{t+1}
        Set $s_{t+1} = s_t,\ a_t,\ s_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
        Store transition ($\phi_t,\ a_t,\ r_t,\ \phi_{t+1}$) in *D*
        Sample random minibatch of transitions ($\phi_j,\ a_j,\ r_j,\ \phi_{j+1}$) from D
        if *episode terminates at step j+1* **do**
            Set $y_j = r_j$
        else
            Set $y_j = r_j + \delta \max_{a'} \bar{Q}(\phi_{j+1}, a'; \theta^-)$
        Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ
        Every *C* steps reset $\bar{Q} = Q$
    End For
End For

DQN 中神经网络模块的使用克服了单智能体强化学习的高维数据灾难，并且通过目标值网络、经验回放池的使用，以及利用基于ϵ-greedy 方法的探索机制在一定程度上平衡了探索-利用（exploration-exploitation）矛盾。

4.3 基于 DQN 的多智能体强化学习算法

尽管 DQN 算法在一定程度上克服了维度灾难和探索-利用权衡问题，但仍面临很多困难：在多智能体系统中，还需要解决包括智能体难以确定学习目标、学习问题的非稳定性以及协调的需要等困难。此外，对于博弈模型，它的均衡可能不是唯一的。本文提出的算法通过合适的奖励函数以及策略选择机制设计，可保证每个基于 DQN 的智能体的学习结果收敛于唯一的相关均衡策略。

4.3.1 奖励函数设计

在多智能体学习系统中，通常面临着智能体难以确定学习目标、学习问题不稳定以及处理协调的挑战。我们发现，伴随着合适的奖励函数设计，可以保证在多智能体学习场景中该算法的稳定性和收敛性。对于最大完成时间智能体，本文为其设计的奖励函数为，

$$\mathfrak{R}_1 = \left[\frac{ET_{k,i,j}(a)-(makespan'-makespan)}{ET_{k,i,j}(a)}\right]^3 \tag{4.3}$$

类似的，花费的奖励函数被设计为，

$$\mathfrak{R}_2 = \left[\frac{worest-ET_{k,i,j}(a)*p_j}{worest-best}\right]^3 \tag{4.4}$$

最大完成时间的奖励函数$\mathfrak{R}_1$和花费的奖励函数$\mathfrak{R}_2$的值域都落在[0,1]内，其中公式(4.3)表示在执行策略a，更新最大完成时间时，使增加的最大完成时间的值越小越好，对应的奖励值越接近 1；否则，趋近于 0。同理，公式(4.4)代表花费的增加值越小，其策略越可取，其奖励值越趋近于 1；否则，没有奖励，值为 0。

4.3.2 选择机制设计

在学习由实际问题衍生出来的一般的马尔可夫博弈的相关均衡策略中的困难之一是，存在多个值的多个均衡。另一方面，在任意多智能体强化学习中，其均衡解的选择问题都在增多，这一点是毋庸置疑的。为了解决这个问题，本文考虑了四种比较常见的均衡策略选择机制，包括功利主义的（utilitarian）、平等主义的（egalitarian）、富豪的（plutocratic）以及独裁的（dictatorial）选择机制f。

① 功利主义的：最大化所有智能体的奖励值的求和：在状态 s，有

$$\max_{\pi_s\in\Delta(A(s))} \sum_{j\in I}\sum_{a\in A(s)}\pi_s(a)Q_j(s,a) \tag{4.5}$$

② 平等主义的：最大化所有智能体的奖励的最小值：在状态 s，有

$$\max_{\pi_s\in\Delta(A(s))} \min_{j\in I}\sum_{a\in A(s)}\pi_s(a)Q_j(s,a) \tag{4.6}$$

③ 富豪的：最大化所有智能体的奖励的最大值：在状态 s，有

$$\max_{\pi_s \in \Delta(A(s))} \max_{j \in I} \sum_{a \in A(s)} \pi_s(a) Q_j(s,a) \tag{4.7}$$

④ 独裁的：最大化任何单个智能体的奖励值的最大值：对于智能体 i，在状态 s，有

$$\max_{\pi_s \in \Delta(A(s))} \sum_{a \in A(s)} \pi_s(a) Q_i(s,a) \tag{4.8}$$

为了使动态调度结果收敛于相关均衡，每个 DQN 智能体都学习相关均衡策略 π^t，其中$\pi_s^{t+1} \in f(Q^{t+1}(s))$。因此，原问题的求解被更为一般的描述为，在马尔可夫博弈中，对所有的$i \in I$，所有的$s \in S$，以及所有的$a \in A(s)$，在时间步t给定Q值，即为$Q_i(s,a)$；给定一个策略π^t；并给定一种选择机制f，也就是将多工作流-多目标优化调度问题的博弈映射到联合分布的集合；在时刻 t+1，有

$$V_i^{t+1}(s) = \sum_{a \in A(s)} \pi_s^t(a) Q_i^t(s,a) \tag{4.9}$$

$$Q_i^{t+1}(s,a) = (1-\delta) R_i(s,a) + \delta \sum_{s' \in S} P[s'|s,a]\, V_i^{t+1}(s') \tag{4.10}$$

$$\pi_s^{t+1} \in f(Q^{t+1}(s)) \tag{4.11}$$

本文选取表达式(4.5)中功利主义的选择机制作为（均衡）策略调整机制。

4.3.3 算法设计与分析

本文将单个智能体强化学习中经典的 DQN 算法和多智能体强化学习中的 correlated Q-learning 算法思想相结合，提出了一种适用于云环境中多工作流-多目标优化调度问题的算法，命名为基于 DQN 的多智能体强化学习（DQN-based MARL）算法。算法 4-2 给出了其伪代码。

算法 4-2 基于 DQN 的多智能体强化学习的多目标工作流调度方法

Algorithm 4-1 DQN-based MARL method

Input: game Γ^{δ}, selection mechanism f

Output: Q-values Q, stationary policy π^*, reward r

Initialize replay memory D, action-value function Q with random weights θ;
Initialize state s, action profile a;
observe initial state S;
while *not at max_episode* **do**
    if *with probability* ε **then**
        select a random action a;
    else
        select $a \in f$;
    carry out action a;
    observe reward r and new state s';
    store experience $< s, a, r, s' >$ in replay memory D;
    sample random transitions $< ss, aa, rr, ss' >$ from replay memory D;

13　　calculate target for each minibatch transition;

14　　**if** ss' *is terminal state* **then**

15　　$tt = rr$;

16　　**else**

17　　$tt = rr + \delta \max_{a'} Q(ss', aa')$;

18　　train the Q-network using $\left(tt - Q(ss', aa')\right)^2$ as loss;

19　　$s = s'$

20 return Q-values Q, action profile a, reward r

从理论上分析，DQN-based MARL 算法通过神经网络、经验回放缓冲区、ϵ-greedy 探索机制、智能体协作、奖励机制和选择机制的设计，在一定程度上解决了前文中提到的维度诅咒、探索-利用权衡问题，以及学习目标不明确、学习效果（奖励值）不稳定等问题。

4.4 本章小结

本章是本文的核心内容。首先根据第三章提出的基于马尔可夫博弈的云工作流调度模型，进一步导出量化的基于多个 DQN 智能体强化学习的系统模型。接着详细介绍并分析了单个智能体强化学习中基于马尔可夫决策过程的 DQN 算法，包括环境模块、损失函数、经验重放模块和两种结构完全相同但参数不同的神经网络，即估计值网络和目标值网络。在此基础上，本文针对多智能体强化学习目前面临的一些挑战，结合经典的博弈论中的相关均衡与机制设计理念，提出了基于 DQN 算法的多智能体强化学习的多目标工作流调度算法。通过为每个 DQN 智能体设计合适的奖励函数以及策略选择机制使其在一定程度上克服了在多智能体系统中学习目标不确定、学习问题不稳定以及协调的需要等问题，并使其调度结果收敛于相关均衡策略。

5 案例研究和实验结果分析

本文的案例分析和算法的实现是基于仿真实验环境进行的。为了验证模型和算法，本文基于图 5.1 所示的五种著名的科学工作流模板和真实的第三方商业云，如表 5.1 所示的 Amazon EC2 云主机实例，进行了大量的案例研究。

5.1 实验设置

① 实验环境。本章的实验环境配置如下：

1) 操作系统：Windows 10 家庭版
2) 处理器：Intel(R) Core i7-5500U CPU @ 2.40GHz 2.39 GHz
3) 已安装的内存（RAM）：8.00 GB（7.70 GB 可用）
4) 集成开发环境：PyCharm 2017.3.3（专业版）
5) 程序语言及版本：Python 3.6

② 输入数据。本实验中主要包括两类输入数据：IaaS 云资源数据和云工作流数据。实验考虑了异构的云资源模型，即七种不同的云主机类型（vType），如表 5.1 所示。其资源的异构性主要体现在计算资源，如云主机的虚拟 CPU（vCPU）、内存资源（Memory）、可用区（Availability Zone），以及单位时间服务价格（Price）。前者决定了云主机的计算性能，后者决定了单位时间花费。

表 5.1 Amazon EC2 云主机的参数配置

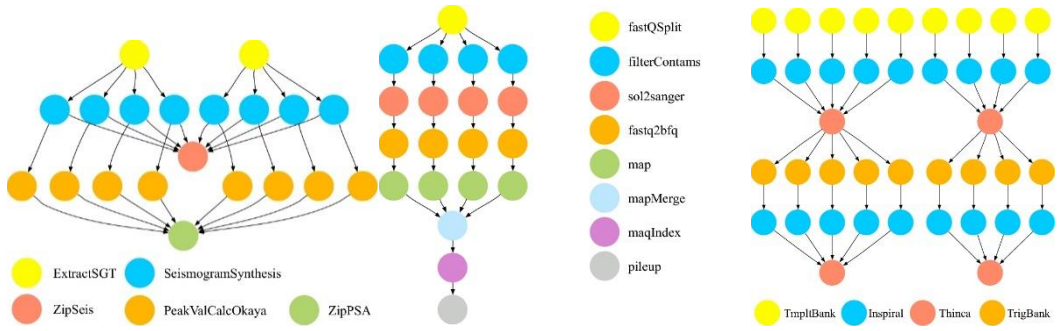
Table 5.1 Units for parameter configuration of Amazon EC2 instances

vType	vCPU	Memory	Availability Zone	Price (USD$/hr)
t3.medium	2	8	us-east-2a	0.0418
t3.large	2	8	us-east-2c	0.0835
c5.large	4	16	us-east-2b	0.0850
m5.large	4	12	us-east-2a	0.0960
c5n.large	8	32	us-east-2c	0.1080
r5a.large	8	32	us-east-2b	0.1260
a1.4xlarge	8	32	us-east-2a	0.4080

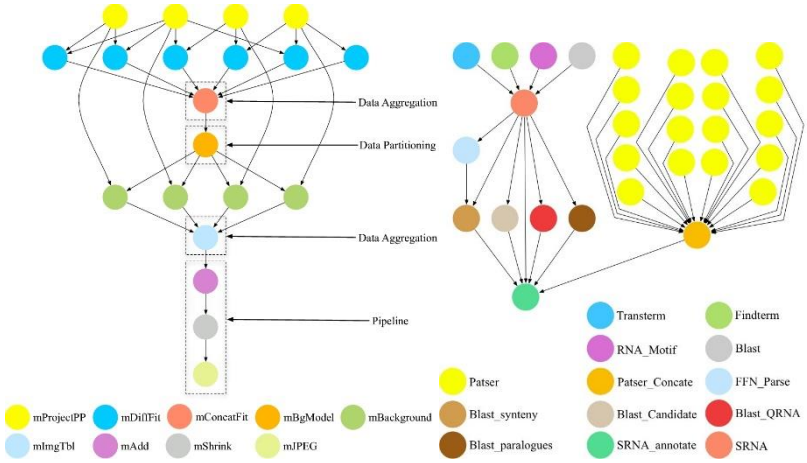
此外，本文采用五种著名的科学工作流模板作为工作流模型，包括 CyberShake、Epigenomics、Inspiral、Montage 和 Sipht。这五种工作流分别来自不同的项目。其中，CyberShake 工作流是被南加州地震中心用来描述一个地区的地震灾害数据；

Epigenomics 工作流是由南加利福利亚大学的表观基因组中心和 Pegasus 团队创建的表观基因组学工作流程，用于自动化基因组序列处理中的各种操作；LIGO 的 Inspiral Analysis 工作流用于生成和分析紧凑二元系统合并期间收集的数据的重力波形；Montage 是由美国国家航空航天局/红外处理与分析中心创建，Montage 工作流的应用程序将多个输入图像拼接在一起，以创建天空的自定义镶嵌图案；Sipht 工作流是来自哈佛大学生物信息学的 SIPHT 项目，用于在 NCBI 数据库中自动搜索细菌复制子的非翻译 RNA（sRNA）。

如图 5.1 所示，这五种科学工作流模型分别由不同的拓扑结构以及不同的任务类型构成。因此，当对多工作流进行调度时，既要考虑到多个工作流的任务是否可并行执行的情况，又要考虑到任务的异构性。此外，本文考虑了不同任务规格的科学工作流，即五种工作流的总任务数量为 138，252，358 和 497 的情况。



(a) CyberShake (b) Epigenomics (c) Inspiral



(d) Montage (e) Sipht

图 5.1 五种著名的科学工作流模板概览.

Fig 5.1 Overview of five workflow templates. (a) CyberShake, (b) Epigenomics, (c) Inspiral, (d) Montage, (e) Sipht.

很明显，对于同一种类型的任务，它在不同类型的云主机上运行的执行时间是不同的；同样，当不同类型的任务在同一台云主机上执行时，其完成时间也是有差异的。为了模拟不同类型的任务在异构的云主机上被执行的场景，同时为了简化实验，我们一共考虑了五种不同的任务类型，包括 AES，LZMA，JPEG，Canny 和 Lua，它们作用于图 5.1 所示的五种工作流模型中。表 5.2 给出了这五种不同的任务在七种异构 Amazon EC2 云主机的性能数据以及平均值，它们自于 Geekbench[52]平台的公测数据。

表 5.2 Amazon EC2 云主机多核性能分数

Table 5.2 Units for multicore performance scores of Amazon EC2 instances

vType	AES	LZMA	JPEG	Canny	Lua	Average
t3.medium	3427	2967	3405	3176	3704	4065
t3.large	3611	3809	4420	4327	4831	5035
c5.large	3648	3664	4402	4152	4761	4521
m5.large	3453	3310	3997	3897	4319	4575
c5n.large	3774	3831	4414	4311	4864	5105
r5a.large	4956	3234	3569	3701	2643	4549
a1.4xlarge	1509	2221	2092	2662	1605	16861

③ 训练流程。基于多工作流-多目标优化的动态调度系统，首先系统会实时地获取工作流和虚拟机的信息，为 DQN-based MARL 算法提供训练样本；然后这些样本存储在一个经验池中；之后更新模型，并将更新后的模型应用于多工作流调度。这个流程不断反复，模型动态更新，在特定的机制设计下调整策略，最终收敛于相关均衡。

④ 参数设置。对于每个 DQN 智能体（工作流调度环境），主要涉及的超参数θ包括神经网络结构、学习率α、奖励衰减率δ、ϵ-贪心因子ϵ、最大值ϵ_{max}、增长率ϵ_{incre}、经验池存储大小m、minibatch 大小以及替换目标值网络参数的间隔迭代次数 replace_target_iter 等。

本文采用三层全连接神经网络结构来构建估计Q值网络和目标Q值网络，输入是长度为 8 的向量。网络隐层的大小分别是 20/7。前一层用的是 Relu 激活函数，后一层用的线性激活函数。其余参数设置为$\alpha = 0.002$，$\delta = 0.9$，ϵ=0.7，ϵ_{max}=0.95，ϵ_{incre} =1e-5，m=10000，minibatch=128，replace_target_iter=500。

在经过 500 次训练过程迭代后，图 5.2 给出了本文提出的算法，即基于 DQN 的多智能体强化学习的工作流调度算法（DQN-based MARL method）的收敛结果

图。从图 5.2 中可以看出，DQN-based MARL 算法的收敛性是很明显的，伴随着小范围的波动，最大完成时间和总花费智能体分别收敛于各自的最高奖励值。

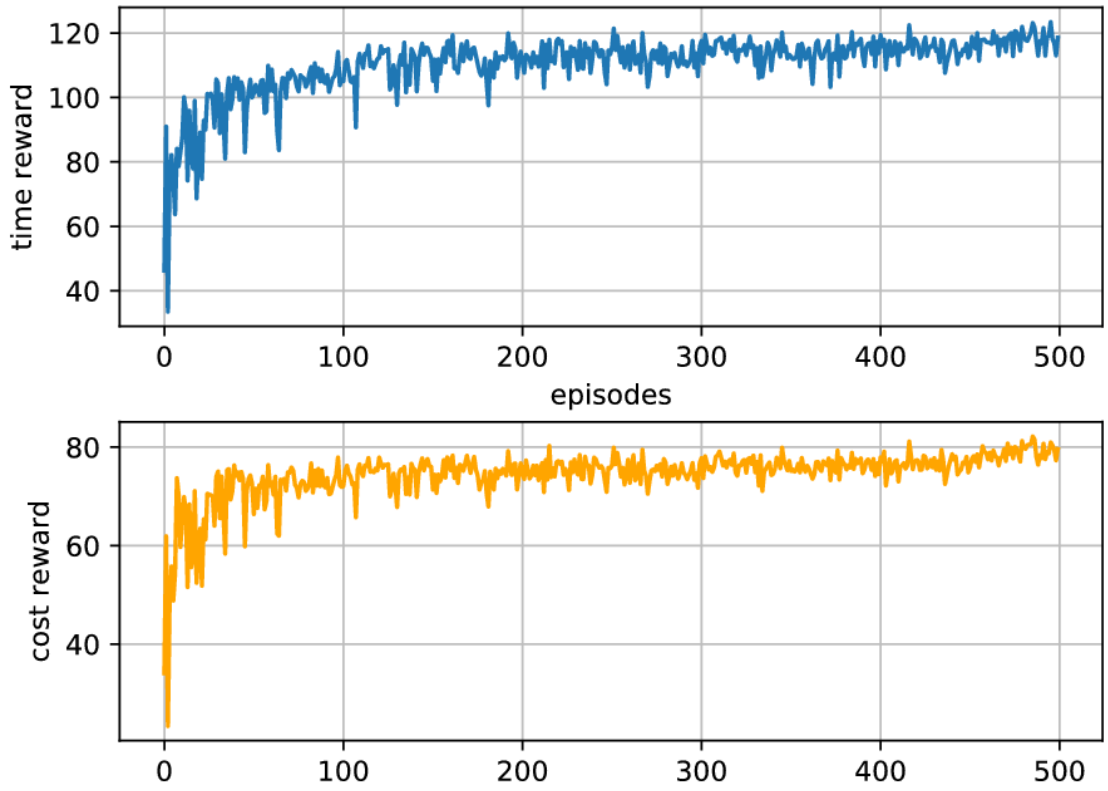


图 5.2 基于 DQN 多智能体强化学习的工作流调度方法的收敛性

Fig 5.2 The convergence of DQN-based MARL method for workflow scheduling

5.2 对比算法

为了验证模型和算法的有效性，出于比较原因，本文将基于（元）启发式和帕累托前沿的 MOPSO[16]，NSGA-II[17]算法，以及基于博弈的贪心算法[18]（GTBGA）作为基准对比算法。

GTBGA 结合了博弈思想以及贪心策略。首先对不同的科学工作流的任务进行切分，将其划分为不同阶段可被执行的任务包，对阶段性可被分配的任务和可用的云主机之间的博弈进行均衡匹配求解，由于存在多个阶段，而调度过程只考虑了单个阶段的最优，因此是贪心的。

MOPSO 和 NSGA-II 都是基于启发式的多目标优化算法，并结合了帕累托最优技术。前者由 Coello 等人于 2004 年提出，目的是允许 PSO 算法可以应用于求解复杂的多目标优化问题。在 PSO 算法的基础上，加入了一个外部存储库（Archive）以及结合特殊的变异操作来寻找帕累托最优解集。后者是 Deb 等人在 NSGA 算法的基础上提出的改进算法，通过使用精英选择策略和拥挤距离保证了解的多样性

且不必依赖于共享参数。

本文的算法与上述对比算法最大的不同在于它不需要大量的先验或后验专家知识和人为干预，而是通过与环境和其他智能体的交互来学习相关均衡策略。此外，相较于帕累托最优策略以一个静态全局的角度看待问题，它更注重分析问题的局部性和动态性，对于在线部署或在线调度等动态更新的问题是更适用的。

5.3 实验结果

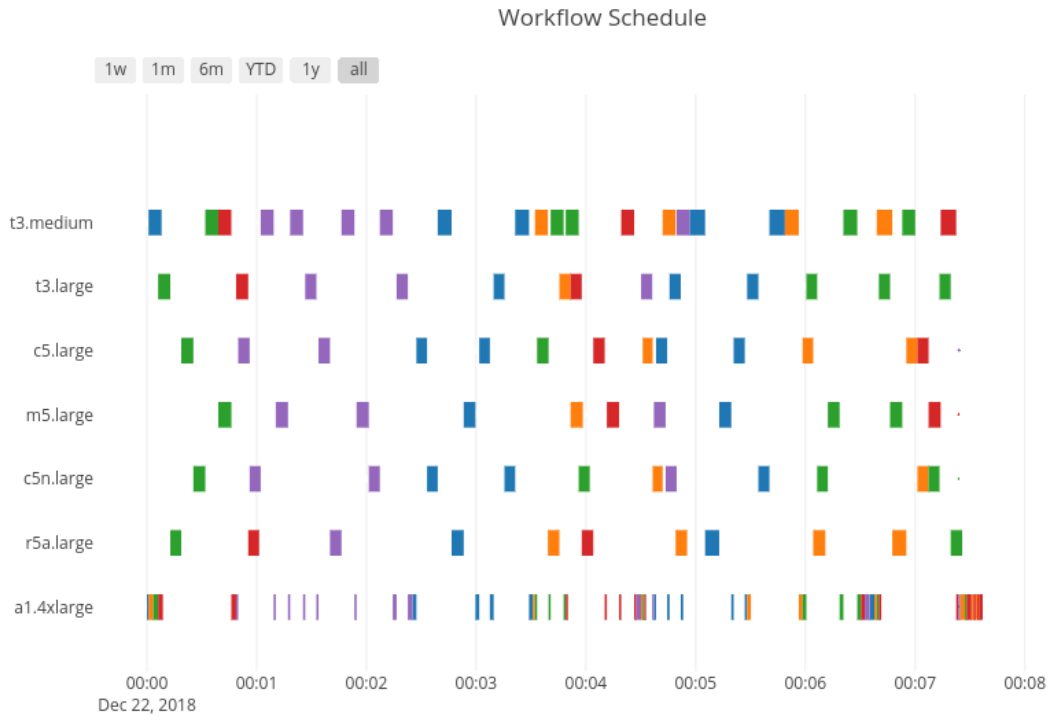
本文主要考虑了五种工作流模板被同时调度的场景，考核了调度策略计划、最大完成时间和总花费两种 QoS 指标的完成情况。同时，也考虑了四种不同总任务数目，即 138，252，358 和 497 的调度情况。

图 5.3 给出了四种算法的调度结果，每种算法对应一张甘特图。其中，一共有 138 个来自五种不同工作流，即 CyberShake_30，Epigenomics_24，Inspiral_30，Montage_25 以及 Sipht_29 的任务被调度。每种颜色代表一种工作流。从图 5.3 中可以看出，我们提出的算法（DQN-based MARL method）在最大完成时间方面明显优于对比算法（GTBGA，NSGA-II 以及 MOPSO）。直观地讲，由于我们的算法留下较少的任务间的停留时间并且更多地利用了 Amazon EC2 平台提供的底层并行性，因此实现了这种优势。相比之下，对比算法倾向于首先遵循工作流的拓扑结构约束，而没有充分利用任务之间潜在的并行性。表 5.3 给出了总任务数为 138 时，四种算法的最大完成时间、总花费以及对比算法与本文提出的算法在总花费数值上的差值百分比。当考虑到最大完成时间和总花费的权衡时，MOPSO 算法的性能明显超过 GTBGA 和 NSGA-II 的性能。虽然 MOPSO 算法得出的花费比我们提出的算法更便宜，但是我们提出的算法在最大完成时间最优性方面明显打败了基准对比算法，并且我们的算法比 GTBGA 便宜 5.938%，仅比 NSGA-II 和 MOPSO 算法分别贵 2.899%，4.303%。

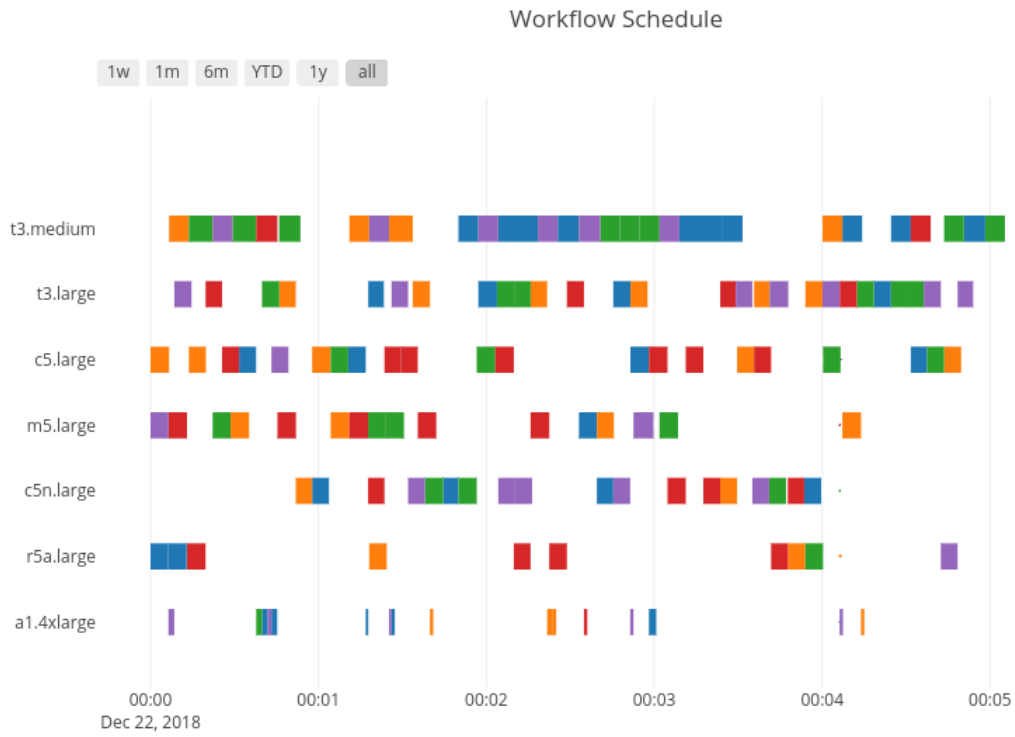
表 5.3 总任务数为 138 时的对比结果表

Table 5.3 Units for comparisons of algorithms when task size = 138

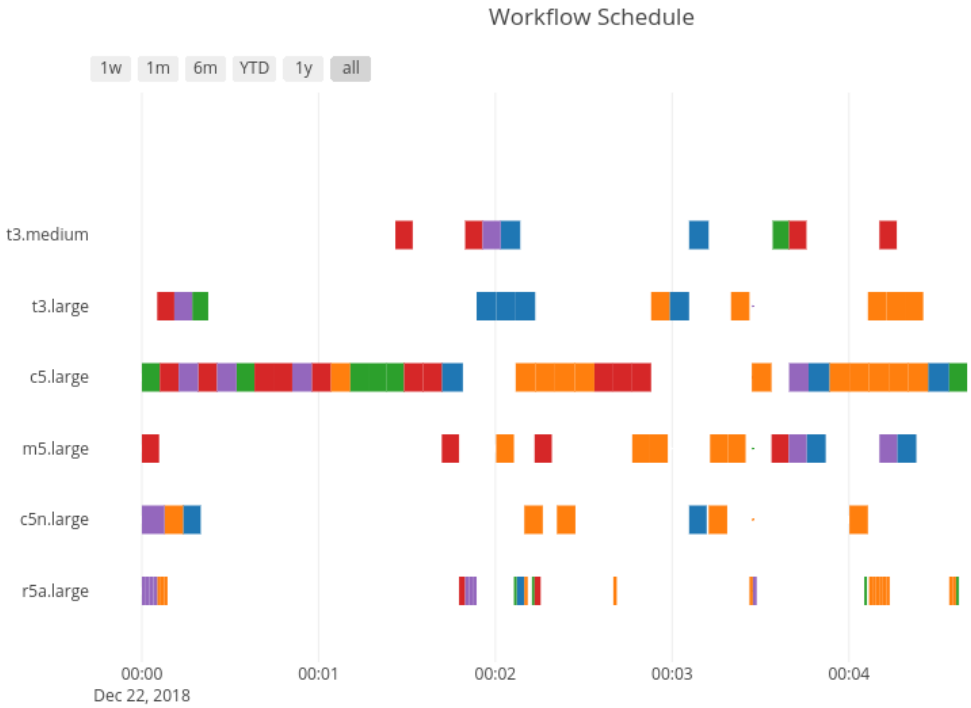
算法	最大完成时间	总花费	花费差值百分比
GTBGA	456.961304	0.022345547	-5.9382614%
NSGA-II	305.309292	0.020426507	2.8987017%
MOPSO	287.696525	0.020151471	4.3031031%
DQN-based MARL	102.6682	0.02101861	0



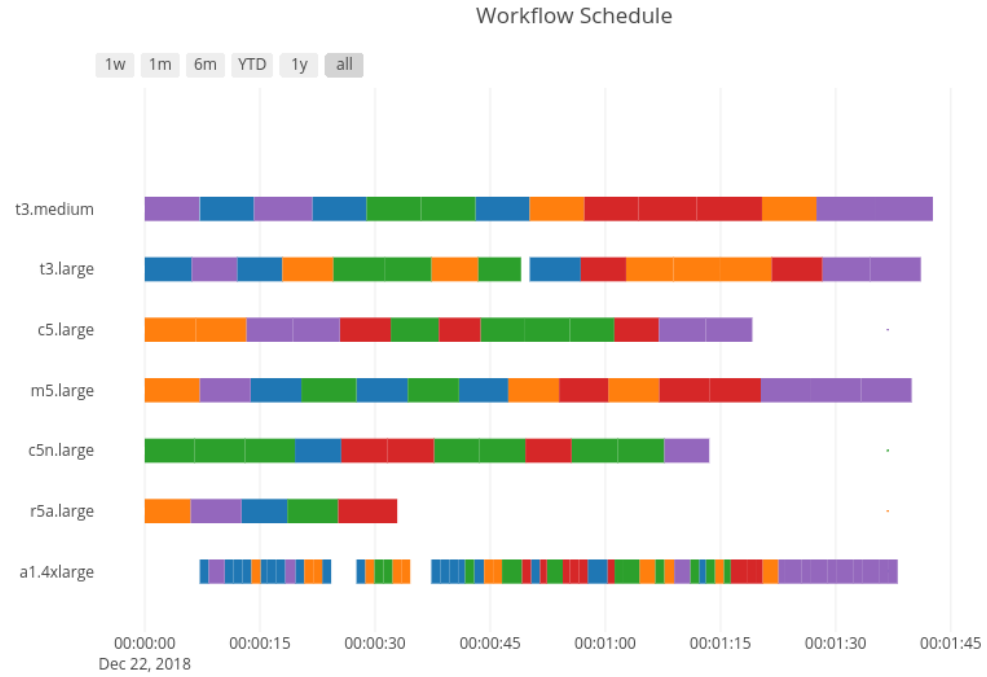
(a) GTBGA



(b) NSGA-II



(c) MOPSO

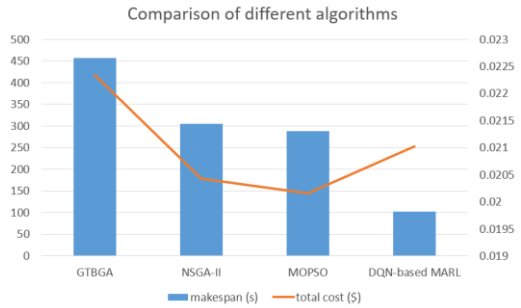


(d) DQN-based MARL

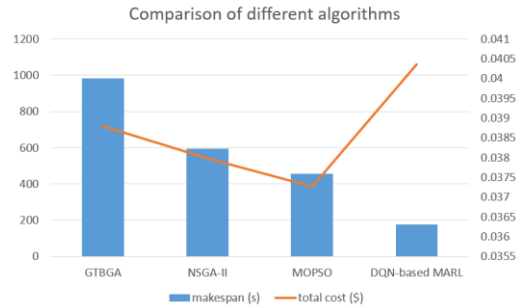
图 5.3 NSGA-II，MOPSO，GTBGA 以及 DQN-based MARL 在总任务数为 138 时的工作流调度结果示意图

Fig 5.3 Illustration of workflow scheduling with 138 tasks over NSGA-II, MOPSO, GTBGA and DQN-based MARL algorithms. (a) GTBGA; (b) NSGA-II; (c) MOPSO and (d) DQN-based MARL algorithm.

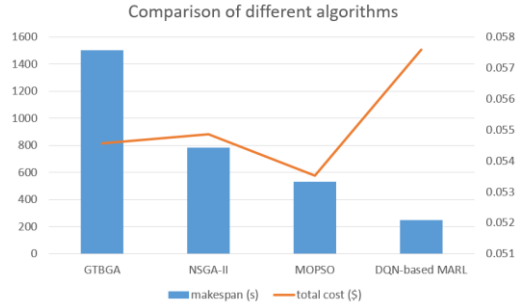
图 5.4 罗列了当任务总数为 138、252、358 以及 497 时的对比结果。每张子图用双坐标轴图表示，其中条形图表示最大完成时间（make-span），折线图表示总花费（cost）。针对最大完成时间指标，总任务数的增加并不会影响四种算法的最大完成时间上的对比趋势，DQN-based MARL 算法总是最优的，其优势很明显，最大完成时间低于最好的对比算法的差值概率超过 50%；相反，总花费的变化相对比较明显。



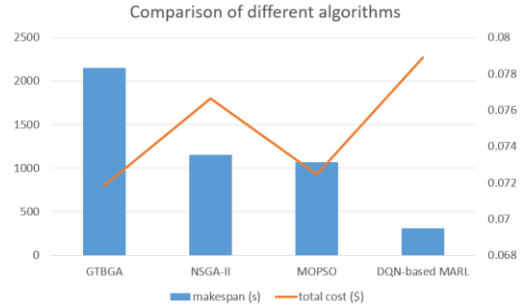
(a) 任务总数= 138



(b) 任务总数= 252



(c) 任务总数= 358



(d) 任务总数= 497

图 5.4 不同的任务大小下的对比结果示意图. (a) 138; (b) 252; (c) 358 以及 (d) 497.

Fig 5.4 Illustration of comparisons of baseline algorithms and DQN-based MARL method with four different task sizes. (a) 138; (b) 252; (c) 358 and (d) 497.

图 5.5 单独分析了四种任务大小下的总花费的对比结果。从图中可以看出，在同一总任务数下，四种算法的总花费的差距并不大，基本持平。表 5.4 和表 5.5 进一步对比了实验结果，分别给出了 DQN-based MARL 相对于其它算法在最大完成时间和花费方面的增长百分比的对比数据。其中，在最大完成时间（make-span）指标上本文提出的算法绝对优于对比算法（>53.4%）；相反，随着任务数的增加其支付的总花费（cost）波动性则高于对比算法，但其长势控制在 2%到 9.9%之间。

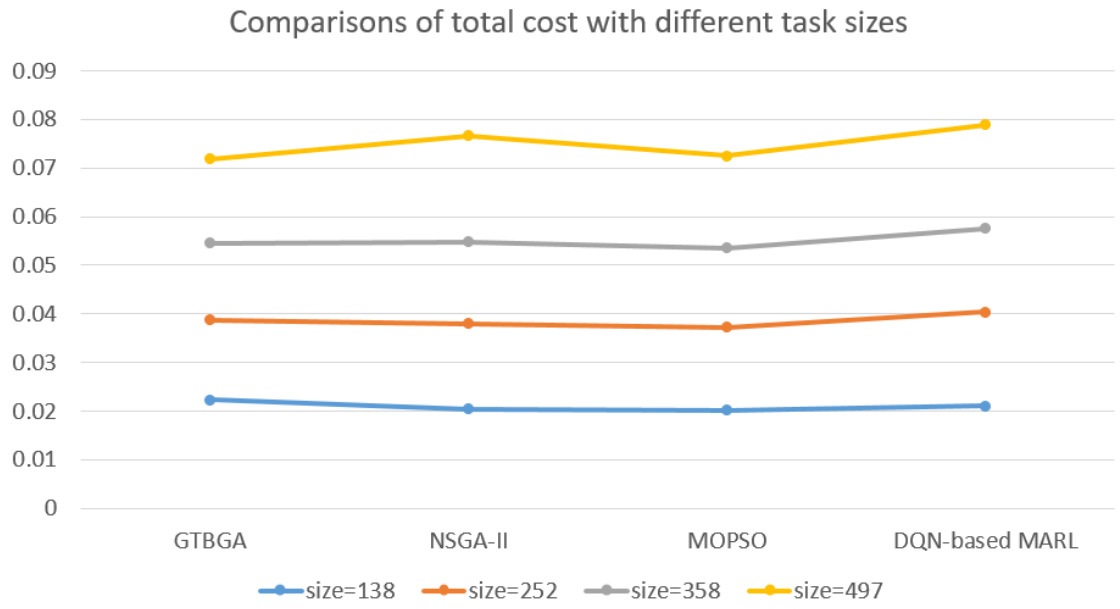


图 5.5 四种任务大小下四种算法的总花费对比

Fig 5.5 Comparisons of total cost of four algorithms in terms of four task sizes

表 5.4 DQN-based MARL 相对于基准算法的 make-span 增长率

Table5.4 Units for make-span increment rate of DQN-based MARL compared to others

算法	任务总数			
	138	252	358	497
GTBGA	-0.77532	-0.82219	-0.83579	-0.85493
NSGA-II	-0.66372	-0.70589	-0.68539	-0.73011
MOPSO	-0.64314	-0.61799	-0.53457	-0.70908

表 5.5 DQN-based MARL 相对于基准算法的花费增长率

Table 5.5 Units for cost increment rate of DQN-based MARL compared to others

算法	任务总数			
	138	252	358	497
GTBGA	-0.059383	0.040311	0.055401	0.098098
NSGA-II	0.028987	0.062725	0.049967	0.029534
MOPSO	0.043031	0.083047	0.076154	0.0891

由上述分析可得，本文提出的算法得出的调度结果在最大完成时间 QoS 指标上绝对优于对比算法（>53.4%）。即使随着任务数的增加其支付的总成本高于对比算法，但同比增长并不高（<9.9%）。如果对时间要求比较高而成本又在可考虑的

范围内，本文的算法是完全占优的。

5.4 本章小结

本章的重点就是通过案例研究与模拟实验，来验证本文所提出调度问题模型以及调度算法。首先给出了实现本文介绍的算法采用的实验设置，包括实验平台配置环境、输入数据、训练流程以及参数设置，对案例研究中采用的模拟实验环境进行了介绍。然后选取了三个传统的工作流调度算法，即 NSGA-II、MOPSO、GTBGA，作为基准对比算法进行实验，主要对比了两个 QoS 指标，即最大完成时间和总花费，并分别给出了四种算法得出的任务调度策略的甘特图比较结果。此外，本文还对比了四组不同总任务数量大小，即 138，252，358，497 的实验结果。实验结果显示，本文提出的调度算法在最大完成时间优化指标上占绝对优势，差值百分比超过 53.4%；尽管本文提出的算法的总成本花费高于部分基准对比算法，但是高出的成本不超过 9.9%。

6 总结与展望

6.1 工作总结

各式各样的、大规模的科学问题通常被建模为工作流程。近年来，云计算的发展带来了以低成本运行科学工作流程的巨大机会，它提供了几乎无限的资源池作为这些复杂的科学应用程序运行的支撑平台，并根据实际需求，以及灵活的付费模式付费使用。这些应用程序不断增长的数据和计算需求导致了对如何在云环境中有效地调度和部署它们的广泛研究。近年来国内外对云环境下工作流调度的研究进展已取得一定的成果，然而，现有的大多数研究都是基于大量的先验或后验的专家知识和人为干预实现的，而在基于非专家知识、自适应学习算法解决云系统下多工作流-多目标优化调度的云工作流调度方法的研究仍不存在。基于上述存在的问题和不足，本文研究的主要工作内容如下：

本文首先将复杂而具体的 IaaS 云环境下的多工作流-多目标优化调度问题量化并形式化抽象建模成马尔可夫博弈模型，并讨论了该问题模型下的相关均衡解。接着在马尔可夫博弈的形式化模型的基础上，进一步导出基于 DQN 的多智能体学习框架的系统模型。然后分析了基于 DQN 算法的单个智能体的算法实现情况。接着分析了在多个 DQN 智能体系统下，通过为每个 DQN 智能体设计合适的奖励函数和选择机制来辅助达成相关均衡，使每个 DQN 智能体不断地在马尔可夫博弈学习环境中学习动态的策略收敛于相关均衡策略。为了验证模型和算法的可用性，本文将实验数据代入多 DQN 智能体强化学习模型中，完成其收敛性验证，再通过与基于启发式的多目标优化算法与基于博弈的贪心算法进行对比，得出最大完成时间和总花费的数据，证明了我们提出的理论模型的正确性和有效性。实验结果表明，多智能体系统的强化学习能够应用在多目标-多工作流调度中，并取得了可观的结果。本文提出的算法在最大完成时间指标上占绝对优势，其完成时间低于所有对比算法至少 53.4%，尽管本文提出的算法的总成本花费高于部分基准对比算法，但是高出的百分比不超过 9.9%。

6.2 研究展望

本文的模型和算法虽然可以较好的解决 IaaS 云环境下的多工作流-多 QoS 指标优化调度问题，但是还有许多改进的地方。未来将会通过以下方面对模型进行改进。

① 对实验数据的完善。本文的实验采用的是基于真实数据的模拟实验，实验数据比较单一、简单。然而，在现实的云计算系统中，特别是在互联网上部署的

大规模的工业云上运行的工作流的任务更加复杂，且其规模也更大。除此之外，工作流的状态也是在线动态更新的，需要更多的学习训练过程提取特征。在以后的研究中将会考虑规模更大、任务更加复杂的工业级的工作流调度的情况，完善数据预处理模块并考虑开发 Demo，并部署于真实的云环境中使其在线学习云工作流调度过程；

② 目前我们使用了多个工作流的最大完成时间和总花费指标来评估调度模型和算法。未来将开发分析出更多的 QoS 指标，如最小化弹性时间、资源使用成本和可靠性，甚至高维目标。针对不同的侧重点，从多个方面对调度问题的建模和调度算法进行分析；

③ 目前的研究工作采用基于 DQN 的智能体根据与环境和其它智能体交互学习，是基于值函数迭代的算法。在未来进一步的研究中，我们将采用基于策略迭代的模型和算法来快速计算评估指标，如 DDPG 或 MADDPG 算法，都是值得探索和尝试的；

④ 目前的研究仅考虑同一家云服务供应商中的异构云主机，也没有考虑任务在不同云主机上的传输时间。在以后的研究中，应考虑在多个云服务厂商之间进行远程扩展，并考虑任务的传输延迟时间。

参考文献

[1] Rodriguez M A, Buyya R . A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments[J]. Concurrency & Computation Practice & Experience, 2016, 29(8).

[2] Xia Y N, Zhou M C, Luo X, et al. Stochastic Modeling and Performance Analysis of Migration-enabled and Error-prone Clouds[J]. IEEE Transactions on Industrial Informatics, 2015, 11(2):495-504.

[3] Xia Y N, Zhou M C, Luo X, et al. A Stochastic Approach to Analysis of Energy-Aware DVS-Enabled Cloud Datacenters[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2015, 45(1):73-83.

[4] Yuyu Y , Yueshen X , Wenting X , et al. Collaborative Service Selection via Ensemble Learning in Mixed Mobile Network Environments[J]. Entropy, 2017, 19(7):358-.

[5] Yu J , Kuang Z , Zhang B , et al. Leveraging Content Sensitiveness and User Trustworthiness to Recommend Fine-Grained Privacy Settings for Social Image Sharing[J]. IEEE Transactions on Information Forensics and Security, 2018, 13(5):1317-1332.

[6] Yuyu Y , Fangzheng Y , Yueshen X , et al. Network Location-Aware Service Recommendation with Random Walk in Cyber-Physical Systems[J]. Sensors, 2017, 17(9):2059-.

[7] Yu J , Zhang B , Kuang Z , et al. iPrivacy: Image Privacy Protection by Identifying Sensitive Objects via Deep Multi-Task Learning[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(5):1005-1016.

[8] A. Choudhary, I. Gupta, V. Singh, et al. A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing[J]. Future Generation Computer Systems, 2018, 83: 14–26.

[9] Q. Peng, M. Zhou, Q. He, Y. Xia, et al. Multi-objective optimization for location prediction of mobile devices in sensor-based applications[J]. IEEE Access, 2018, 6: 77123–77132.

[10] W. Li, Y. Xia, M. Zhou, X. Sun, and Q. Zhu. Fluctuation-aware and predictive workflow scheduling in cost-effective infrastructure-as-a-service clouds[J]. IEEE Access, 2018, 6: 61488–61502.

[11] Xu R , Wang Y , Luo H , et al. A sufficient and necessary temporal violation handling point selection strategy in cloud workflow[J]. Future Generation Computer Systems, 2018, 86: 464-479.

[12] Deng S , Huang L , Taheri J , et al. Computation Offloading for Service Workflow in Mobile

Cloud Computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 26(12):1-1.

[13] Yu J , Tao D , Wang M , et al. Learning to Rank Using User Clicks and Visual Features for Image Retrieval[J]. IEEE Transactions on Cybernetics, 2015, 45(4):767-779.

[14] Ullman J D. NP-complete scheduling problems[J]. Journal of Computer & System Sciences, 1975, 10(3):384-393.

[15] D. Nasonov, A. Visheratin, N. Butakov, N. Shindyapina, M. Melnik, and A. Boukhanovsky. Hybrid evolutionary workflow scheduling algorithm for dynamic heterogeneous distributed computational environment[J]. Journal of Applied Logic, 2017, 24: 50–61.

[16] Coello C A C , Pulido G T , Lechuga M S . Handling multiple objectives with particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3):256-279.

[17] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182–197.

[18] Y. Wang, J. Jiang, Y. Xia, Q. Wu, X. Luo, and Q. Zhu. A multi-stage dynamic game-theoretic approach for multi-workflow scheduling on heterogeneous virtual machines from multiple infrastructure-as-a-service clouds[C]. SCC, Lecture Notes in Computer Science, 2018, 10969: 137–152.

[19] E. Iranpour and S. Sharifian. A distributed load balancing and admission control algorithm based on Fuzzy type-2 and Game theory for large-scale SaaS cloud architectures[J]. Future Generation Computer Systems, 2018, 86: 81–98.

[20] R. Duan, R. Prodan, and X. Li. Multi-objective game theoretic scheduling of bag-of-tasks workflows on hybrid clouds[J]. IEEE Transactions on Cloud Computing, 2014, 2(1): 29–42.

[21] L. Wu and Y. Wang. Scheduling Multi-Workflows Over Heterogeneous Virtual Machines With a Multi-Stage Dynamic Game-Theoretic Approach[J]. International Journal of Web Services Research, 2018, 15(4): 82–96.

[22] Jiahao W, Zhiping P, Delong C, et al. A Multi-object Optimization Cloud Workflow Scheduling Algorithm Based on Reinforcement Learning[C]. ICIC, 2018, 10955: 550–559.

[23] Y. Wei, D. Kudenko, S. Liu, et al. A Reinforcement Learning Based Workflow Application Scheduling Approach in Dynamic Cloud Environment[C]. CollaborateCom, 2018, 252: 120–131.

[24] D. Cui, W. Ke, Z. Peng, and J. Zuo. Multiple DAGs Workflow Scheduling Algorithm Based on Reinforcement Learning in Cloud Computing[C]. 2016, 575: 305–311.

[25] B. Waschneck, A. Reichstaller, L. Belzner, T. Altenmüller, T. Bauernhansl, A. Knapp, and A.

Kyek. Optimization of global production scheduling with deep reinforcement learning[J]. Procedia CIRP, 2018, 72: 1264–1269.

[26] M. Kaur and S. Kadam. A novel multi-objective bacteria foraging optimization algorithm (MOBFOA) for multi-objective scheduling[J]. Applied Soft Computing Journal, 2018, 66: 183–195.

[27] L. Zhang, K. Li, C. Li, and K. Li. Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems[J]. Information Sciences, 2017, 379: 241–256.

[28] I. Casas, J. Taheri, R. Ranjan, L. Wang, and A. Y. Zomaya. GA-ETI: An enhanced genetic algorithm for the scheduling of scientific workflows in cloud environments[J]. Journal of Computational Science, 2018, 26: 318–331.

[29] A. Verma and S. Kaushal. A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling[J]. Parallel Computing, 2017, 62: 1–19.

[30] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, and S. Hu. Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT[J]. Future Generation Computer Systems, 2019, 93: 278–289.

[31] D. P. Bertsekas. Feature-based aggregation and deep reinforcement learning: A survey and some new implementations[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2018: 1–31.

[32] H. Mao, M. Alizadeh, I. Menache, and S. Kandula. Resource Management with Deep Reinforcement Learning[C]. Proceedings of the 15th ACM Workshop on Hot Topics in Networks - HotNets ’16, 2016: 50–56.

[33] L. Xue, C. Sun, D. Wunsch, Y. Zhou, and F. Yu. An adaptive strategy via reinforcement learning for the prisoner’s dilemma game[J]. IEEE/CAA Journal of Automatica Sinica, 2018, 5: 301–310.

[34] Y. Zhan, H. B. Ammar, and M. E. Taylor. Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer[C]. IJCAI’16, AAAI Press, 2016, 2315–2321.

[35] H. Wang, T. Huang, X. Liao, H. Abu-Rub, and G. Chen. Reinforcement Learning for Constrained Energy Trading Games with Incomplete Information[J]. IEEE Transactions on Cybernetics, 2017, 47(10): 3404–3416.

[36] L. Zheng, J. Yang, H. Cai, et al. MAgent: A Many-Agent Reinforcement Learning Platform for Artificial Collective Intelligence. 2017: 1–2.

[37] R. Lowe, Y. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch. Multi-agent actor-critic

for mixed cooperative-competitive environments[C]. Neural Information Processing Systems 30, 2017: 6379–6390.

[38] Wikipedia. Cloud Computing[EB/OL]. https://en.wikipedia.org/wiki/Cloud_computing

[39] Matthias Ehrgott. Multicriteria Optimization[M]. Springer, 2005.

[40] Wikipedia. Game theory[EB/OL]. https://en.wikipedia.org/wiki/Game_theory.

[41] 葛泽慧等编著.博弈论入门[M].清华大学出版社,2018.

[42] 谢政,戴丽等.博弈论[M].高等教育出版社,2018.

[43] Aumann R J. Subjectivity and correlation in randomized strategies[J]. Journal of Mathematical Economics, 1974, 1(1):67-96.

[44] Aumann R J. Correlated Equilibrium as an Expression of Bayesian Rationality[J]. Econometrica, 1987, 55(1):1-18.

[45] Wikipedia. RL[EB/OL]. https://en.wikipedia.org/wiki/Reinforcement_learning.

[46] Wikipedia. MDP.[EB/OL]. https://en.wikipedia.org/wiki/Markov_decision_process.

[47] Lucian Buşoniu, Robert Babuška, Schutter B D. Multi-agent Reinforcement Learning: An Overview[M]// Innovations in Multi-Agent Systems and Applications - 1. Springer Berlin Heidelberg, 2010.

[48] Macro Wiering, Martijin van Otterlo. Reinforcement Learning: State-of-the-Art[M]. Springer, 2012.

[49] A. Greenwald and K. Hall. Correlated-q learning[C] in Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03, , AAAI Press, 2003: 242–249.

[50] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning[J]. CoRR, vol. abs/1312.5602, 2013.

[51] Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540):529-533.

[52] Wikipedia. Geekbench[EB/OL]. https://en.wikipedia.org/wiki/Geekbench.

附 录

A. 作者在攻读学位期间发表的论文目录：

[1] 作者***, Hang Liu***, Wanbo Zheng, 导师*, et al. Multi-Objective Workflow Scheduling With Deep-Q-Network-Based Multi-Agent Reinforcement Learning[J], IEEE Access, 2019,7(1): 39974-39982, doi: 10.1109/ACCESS.2019.2902846. (SCI 2 区, Factor: 3.557)

[2] 作者***, Jiajia Jiang, 导师*, Quanwang Wu, Xin Luo, Qinsheng Zhu. A Multistage Dynamic Game-Theoretic Approach for Multi-Workflow Scheduling on Heterogeneous Virtual Machines from Multiple Infrastructure-as-a-Service Clouds[C], SCC, Lecture Notes in Computer Science, 2018, 10969: 137–152. (CCF-C, EI-indexed)

[3] Lei Wu, 作者*. Scheduling Multi-Workflows Over Heterogeneous Virtual Machines With a Multi-Stage Dynamic Game-Theoretic Approach[J], International Journal of Web Services Research (IJWSR), 2018, 15(4): 82-96. (SCI 4 区, Factor: 0.667)

[4] Weiling Li, Yongbo Wang, 作者, 导师*, Xin Luo, Quanwang Wu. An Energy-Aware and Under-Sla-Constraints VM Consolidation Strategy Based on the Optimal Matching Method[J], International Journal of Web Services Research (IJWSR), 2018, 14(4): 75-89. (SCI 4 区, Factor: 0.667)

[5] Li Zhu, 作者, Wanbo Zheng, Lei Wu, Ye Yuan, Peng Chen, 导师*. Percentile Performance Analysis of Infrastructure-as-a-Service clouds with task retrials[C], IEEE 14th International Conference on Networking, Sensing and Control (IEEE ICNSC), 2017: 270-274. (EI-indexed)

[6] Weiling Li, Lei Wu, 导师*, 作者, Kunyin Guo,Xin Luo, Mingwei Lin, Wanbo Zheng. On Stochastic Performance and Cost-aware Optimal Capacity Planning of Unreliable Infrastructure-as-a-Service Cloud[C], International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), 2016: 644-657. (CCF-C)

B. 缩略语对照表:

缩略词	英文全称	中文对照
IaaS	Infrastructure as a Service	基础设施即服务
PaaS	Platform as a Service	平台即服务
SaaS	Software as a Service	软件即服务
QoS	Quality of Service	服务质量
SLA	Service of Level Agreement	服务水平协议
DAG	Directed Acyclic Graph	有向无环图

MOP	Multi-objective Optimization Problem	多目标优化问题
MDP(s)	Markov Decision Process(es)	马尔可夫决策过程
RL	Reinforcement Learning	强化学习
DQN	Deep Q-Network	DQN 算法
DDPG	Deep Deterministic Policy Gradient	深度确定性策略梯度算法
MARL	Multi-agent Reinforcement Learning	多智能体强化学习
NSGA-II	Non-dominated Sorting Genetic Algorithm-II	非支配排序遗传算法-II
MOPSO	Multi-objective Particle Swarm Optimization	多目标粒子群优化算法
GTBGA	Game-theoretic-based greedy algorithm	基于博弈的贪心算法
DQN-based MARL	Deep Q-Network-based Multi-agent Reinforcement Learning method	基于 DQN 的多智能体强化学习方法

C. 符号对照表:

符号	符号描述
N	工作流个数的最大值
K	任务包的个数
n_k	第k个任务包的任务个数
$n_{k,i}$	第k个任务包的第i个任务
M	云主机的个数
V_j	第j台云主机
p_j	第j台云主机的单位服务价格
$st_{k,i,j}$	任务$n_{k,i}$在云主机V_j执行时的开始时间
$rt_{k,i,j}$	任务$n_{k,i}$在云主机V_j执行时的运行时间
$FT(V_j, n_{k,i})$	任务$n_{k,i}$在云主机V_j执行时的结束时间
$x_{k,i,j}$	布尔值。云主机V_j被作为任务$n_{k,i}$执行平台时值为 1；否则，为 0
I	博弈的参与者或智能体的集合
S	博弈的状态空间
A, $\mathcal{A}$	博弈的（纯）行动、混合行动空间
R	参与者或智能体的奖励
P	博弈的转移概率系统
a_t	在时间步t的行动策略
a_i^t	在时间步t第i个参与者的动作
a_{-i}^t	在时间步t其它参与者的动作
s^t	在时间步t博弈的状态

R_i 第i个参与者的奖励

π 固定的策略

π_{s^t} 在时间步t的状态s下联合分布式的策略

π^t 在时间步t的联合分布策略

δ^t 在时间步t的贴现因子

f 选择机制

D. 学位论文数据集:

关键词	密级	中图分类号
多工作流-多目标优化调度；Deep-Q-network；多智能体强化学习；博弈论；IaaS 云计算	公开	TP3

学位授予单位名称	学位授予单位代码	学位类别	学位级别
重庆大学	10611	学术学位	硕士

论文题名	并列题名	论文语种
基于 Deep-Q-network 的多智能体强化学习的多目标工作流调度方法研究	无	中文

作者姓名	王元斗	学号	20161402026t
培养单位名称		培养单位代码	
重庆大学		10611	
学科专业	研究方向	学制	学位授予年
计算机科学与技术	云计算	3 年	2019 年
论文提交日期	2019/5/28	论文总页数	62
导师姓名	夏云霓	职称	教授
答辩委员会主席		李传东	

电子版论文提交格式

文本（√） 图像（） 视频（） 音频（） 多媒体（） 其他（）

致 谢

“人生天地之间，若白驹之过隙，忽然而已”。 ——庄周 《庄子·知北游》

按中国人口的综合平均寿命 76.04 岁来算，研究生三年的岁月只占了人生总时长的 3.95%不到，真的很短暂，但也是人生中非常关键的三年。这段时间内，我经历了很多值得铭记的瞬间，也遇见了一些值得尊敬的师长、热心的朋友。他们的陪伴使我渡过了最难过的日子，为我指引了生活的方向，并使我坚定了我的内心、追逐我的心之向往。在此我诚挚的向他们表达内心的感谢。

首先我要感谢的人是我的硕士研究生导师夏云霓教授。他是一位非常好的导师！在学术、科研项目方面孜孜不倦年轻有为，在生活中平易近人，非常受学生们欢迎。他亲自为学生修改小论文初稿，为学生慷慨解囊，鼓励学生要敢于追求更大更好的平台，只为将来能谋得更好的职业生涯和出路。非常感谢他在我踌躇不前时及时推我一把，让我下定决心追求心中的向往，愿学成归来为团队做更大的贡献。

其次我要感谢在申请学校时联系过的老师和优秀学者。他们是 Shiping Chen，Qinghu Lu，Guo Chen，Joe Dong，感谢他们的帮助和鼓励，让我在求学的路上多一份欣慰和向往，坚定自己的职业生涯目标，愿你们诸事顺利！

再者我要感谢帮助过我的师兄师姐。他们是陈强、喻可、况黎、张海川、李蔚凌、郑万波、肖旋、彭青蓝师兄以及孙晓宁、王胜男师姐，愿你们（学）事业、家庭双丰收。此外，感谢同师门的师弟师妹们，刘航、潘谊、马堉银、王熹佳、顾郑东，以及帮助过我很多次的同实验室的王俊杰师弟、过年早早归校在实验室一起分享零食的姚军师弟，愿你们早日达成目标！

感谢我的父母、爷爷奶奶、舅舅舅妈，感谢你们尊重我的选择并尽可能地给予我帮助。我会努力向前，尽可能保持健康的体魄与成熟的心态以迎接未知的挑战。我爱你们！

三年研究生时光，有欣喜、感动、热爱、失落、悲伤、懊悔、妥协、迷茫、怯懦、踌躇、振作、任性、坚定、固执和坚持……那些历历在目的记忆碎片和情感伴随着我成长，我要感谢我的室友陈冰洁、虞甜静以及周璟琰陪伴我渡过最低落的时期，祝你们快乐幸福健康。

最后，对本文写作中用到的国内外研究成果、著作、文献的作者表示真挚的感谢。

前路道阻且长，行则将至！

王元斗

二 O 一九年四月 于重庆