doi:10.12068/j.issn.1005-3026.2026.20250021

基于ACO算法及可变性管理的SaaS多租户服务仿真技术

印 莹[1]，霍胤彤[1]，刘影梅[2]
（1.东北大学 计算机科学与工程学院，辽宁 沈阳 110169；
2.北京仿真中心 复杂系统建模与仿真全国重点实验室，北京 100854）

摘 要：为了满足不同租户的具体业务需求，软件即服务（SaaS）通常提供仿真定制功能.通过仿真定制，租户可根据自身的业务需求，对软件即服务进行个性化配置，从而更好地满足其业务需求.但是现有的租户定制服务存在租户要求无法充分满足以及算法运行和响应速度慢的问题.因此，本文提出一种基于蚁群优化（ACO）算法和可变性遍历的软件即服务多租户服务仿真定制技术，实现仿真优化服务部署，引入可变性模型，实现服务定制组装的适应性和可复用性.实验结果显示，在评估SaaS租户服务资源使用情况时，该技术在实例a和b上的平均值高于对比算法；执行时间在不同配置方案中有所波动，最短为1 426 ms，最长为1 652 ms；切换资源耗费占空比相对较为稳定，波动范围在1.12%~1.51%，较低的占空比意味着在相同时间内，SaaS能够更有效地利用资源，减少因资源切换而带来的性能损耗.不同SaaS租户的配置方案及运行时间的数据表明，租户能够有效派生服务配置方案.所提技术可为SaaS的仿真定制性能优化提供技术参考.

关 键 词：ACO算法；云计算；SaaS；可变性管理；多租户
中图分类号：TP 311 **文献标志码**：A **文章编号**：1005-3026(2026)01-0082-08

SaaS Multi-tenant Service Simulation Technology Based on ACO Algorithm and Variability Management

YIN Ying[1], *HUO Yin-tong*[1], *LIU Ying-mei*[2]
（1. School of Computer Science & Engineering, Northeastern University, Shenyang 110169, China; 2. Modeling and Simulation of Complex Systems National Key Laboratory, Beijing Simulation Center, Beijing 100854, China. Corresponding author: LIU Ying-mei, E-mail: lymcasic0812@163.com）

Abstract: In order to meet the specific business needs of different tenants, software as a service (SaaS) usually provides simulation customization functions. Through simulation customization, tenants can personalize SaaS according to their own business needs, so as to better meet their business needs. However, existing tenant customization services have the problems of failing to fully meet tenant requirements and having slow algorithm operation and response speeds. Therefore, an SaaS multi-tenant service simulation and customization technology based on an ant colony optimization (ACO) algorithm and variability traversal was proposed. The simulation optimization service deployment was achieved. By introducing a variability model, the adaptability and reusability of service customization and assembly were realized. The experimental results show that in the evaluation of SaaS tenant service resource utilization, the average values of the proposed technology are slightly higher than those of the comparison algorithm on instances a and b. The execution time fluctuates among different configuration schemes, with the shortest being 1 426 ms and the longest being 1 652 ms. The duty cycle of switching resources is relatively stable, with a fluctuation range between 1.12% and 1.51%. A lower duty cycle means that SaaS can more effectively utilize resources and reduce performance losses caused by resource switching at the same time. Based on the configuration schemes and

收稿日期：2025-03-11
作者简介：印 莹（1980—），女，辽宁铁岭人，东北大学副教授，博士生导师.
通信作者：刘影梅，E-mail:lymcasic0812@163.com.

running time data of different SaaS tenants, it is indicated that tenants can effectively derive service configuration schemes. The proposed technology can provide technical references for optimizing the simulation customization performance of SaaS.

Key words: ACO algorithm; cloud computing; SaaS; variability management; multi-tenant

软件即服务(software as a service,SaaS)应用的多元化和租户(即用户)的专业化对SaaS多租户服务定制能力和仿真技术提出了新的要求[1].SaaS是一种基于云计算的软件交付模式,允许用户通过互联网访问和使用软件,而无需在本地安装和维护.传统SaaS服务定制应用只能为租户的一次租赁维护一份定制内容,而新的SaaS服务定制应用需要支持租户对同一应用的多份定制,以满足不同的业务需求,同时减少重复定制,方便租户统一管理[2].多级定制模型实现租户内所组建定制的业务应用系统之间的定制元数据共享,减少重复定制,方便租户统一管理自己的机构、用户、角色、代码、业务数据、费用等[3].因此,面向SaaS多租户的服务定制被广泛应用.文献[4]提出一种面向多租户数据中心的基于三元演化模型参数的通信开销优化算法,通过结合最优局部模型和三元向量化模型参数的演化方向来减少租户与数据中心模型参数传输之间的冗余通信;同时,基于联邦学习的隐私研究论证分析了在传输通信过程中所提算法能有效保障参与训练租户的隐私信息.文献[5]设计了一个用于共享数据敏感应用程序的云信息系统,在高职院校体育管理应用过程中对该系统进行了实用性验证.随着业务流程变得越来越复杂,当前的流程定制技术对复杂流程定制开发的效率较低.文献[6]基于多租户应用场景,提出一种支持多租户模式的业务流程动态定制模型,证实了该研究方向在提升定制有效性上的实践价值.文献[7]提出一种基于软件定义网络的新的接入网络架构,为不同的接入技术提供了弹性支持能力,实验结果说明了该研究在网络节能、资源利用和成本方面的有效性和实用性.在不断变化的环境中,管理和控制用户隐私是SaaS的一个重要目标.文献[8]提出一种隐私策略自动更新方法,以提高复合服务中服务参与者变更时的用户隐私保护效果.当服务数量达到50时,监控器成功检查服务变化的比例为81%,隐私策略的正确更新率为93.6%.针对云环境下多租户数据库的可伸缩性挑战,文献[9]提出基于内存负载预测的组合预测模型和弹性伸缩策略,通过增强系统稳定性、降低预测误差和减少资源浪费,从而提升响应速度和系统性能.网络虚拟化通过其隔离和多租户机制,有效增强了网络安全.文献[10]提出一种基于果蝇优化的虚拟网络映射算法,该算法在负载均衡、请求接受率和控制延迟方面表现出较好的性能.

综上所述,针对SaaS服务定制仿真,研究者们对SaaS业务服务的处理时延、共享数据、SaaS模式与微服务技术结合、隐私策略、SaaS多租户云网站群体系、多租户架构数据识别都有一定的涉及和研究,但是对于优化算法改进SaaS软件服务定制的应用还不够深入.因此,如何构建有效的可变性模型,以达成SaaS软件服务的高复用性、强适应性并降低开发与维护成本,以及在多租户环境下,如何提升服务定制效率与质量、保障数据安全、实现资源高效利用成为SaaS服务应用的重要目标.在此背景下,本文提出一种基于蚁群优化(ant colony optimization,ACO)算法及可变性管理的SaaS多租户服务仿真技术,研究SaaS多租户服务MapReduce任务调度中满足租户多样化功能需求的最优策略,并构建有效的可变性模型,目标是提升SaaS软件服务的高复用性和强适应性,降低开发与维护成本,为SaaS多租户个性化服务定制提供技术基础.

1　ACO算法和可变性模型驱动的SaaS多租户仿真技术

1.1　融合ACO算法的SaaS多租户服务定制

SaaS多租户服务是一种基于云计算的软件服务模式,多个租户(通常为企业或组织)可共享一套软件系统的资源.每个租户在使用过程中,能根据自身业务需求对软件功能进行个性化配置,同时保持数据的独立性与安全性.在本研究中,SaaS多租户服务主要聚焦于提供多租户共享的业务流程定制化功能,涵盖客户关系管理、资源调度管理等场景.SaaS多租户是一种软件架构风格,它允许多个租户共享相同的系统资源,并提供数据隔离,在降低运营成本、提高资源利用率和加快服务部署速度等方面具有显著优势.在设计系统时,需要考虑如何在硬件资源有限的情

况下提供最佳租户体验,例如合理的数据库设计、缓存策略和资源隔离等[11-12].但系统在如何满足租户的定制化需求、确保数据隔离和安全性以及优化资源分配等方面依旧面临着挑战.

在SaaS多租户中,MapReduce是一种编程模型,用于处理大型数据集.在运行MapReduce任务时,可通过调整任务的并行度、数据就近访问性、减少磁盘I/O、使用更有效的数据结构等[13]进行优化.针对SaaS多租户环境中的MapReduce优化以及传统SaaS多租户服务中用户需求无法满足的问题,本文基于MapReduce框架引入ACO算法改进MapReduce框架以满足多租户的SaaS服务定制需求.对比传统MapReduce调度策略(如基于任务优先级的调度、随机调度等),ACO算法具有多样性解生成与区域探索、自适应跳出局部最优、动态搜索策略调整机制和并行多区域搜索能力的优点.在SaaS多租户环境中,不同租户的需求复杂多样,传统调度策略难以全面兼顾,而ACO算法可以更好地适应这种复杂环境,通过蚂蚁路径信息素的迭代更新机制,寻找更优的服务组合方案,满足租户多样化的需求,提升整体服务质量.

进一步提出一种新型的仿真优化算法以更好地满足多租户的SaaS服务定制需求.首先将服务定制映射成一个加权图,节点表示服务,边表示服务之间的依赖关系或组合方式;每只蚂蚁代表一个可能的服务组合方案,通过遍历加权图寻找最优路径(即最优服务组合);根据服务组合的质量(如满足租户需求的程度、成本等)来更新信息素浓度,以指导后续搜索.具体框架如图1所示,共分为三部分.第一部分是一代进行计算工作,包括Map函数和Reduce函数,其中Reduce函数用于处理信息素的更新,为下一代Map函数提供输入信息;第二部分是迭代过程,通过云计算不断迭代;第三部分是蚂蚁族群求解过程.由于云计算需要耗费一定时间,而蚂蚁独立求解是并行性蚁群算法中最耗时的过程,因此ACO算法运行需要保证蚁群数量合理,避免额外的时间耗费.

ACO算法包括启发函数、部分解、转移概率集合等,表达式如式(1)所示:

$$\left.\begin{aligned}\delta_{ij}&=r_{ij}/b_i,\\ \bar{\delta}_j&=\left(\sum_{i=1}^{m}\delta_{ij}\right)/m,\\ \partial_j&=Q_j/\bar{\delta}_j.\end{aligned}\right\} \tag{1}$$

其中:δ_{ij}表示蚂蚁在某时刻的部分解;$\bar{\delta}_j$表示蚂蚁构造解时的转移概率集合;∂_j表示正反馈机制参数;r_{ij}表示蚂蚁移动距离;b_i表示位置标记;i表示位置编号(如任务编号);m表示访问次数;j表示具体项(如资源节点编号);Q_j表示第j项的价值.

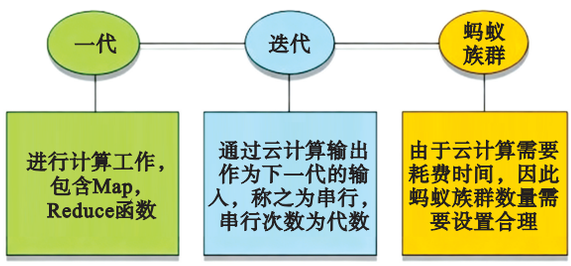

图1 ACO算法改进MapReduce框架

Fig. 1 Improved MapReduce framework based on ACO algorithm

在MapReduce调度场景下,蚁群算法通过模拟蚂蚁觅食行为,将任务分配问题转化为路径寻优过程,有效提升调度效率与全局优化能力.以下结合具体应用给出蚁群算法变量的含义:1)蚂蚁某时刻的部分解δ_{ij}代表蚂蚁在MapReduce调度过程中某时刻已确定的任务分配方案片段,其中i表示任务编号,j表示分配的资源节点编号.例如,$\delta_{3,5}$表示任务3已分配至资源节点5,构成了当前解空间的局部状态.δ_{ij}作为构建完整调度方案的基础单元,约束了后续任务分配的可行空间.蚂蚁需基于已有δ_{ij}信息,结合任务依赖关系(如数据处理的先后顺序)与资源约束(如节点计算能力、内存容量),动态扩展解空间,避免生成不可行调度方案.2)转移概率集合($\bar{\delta}_j$).在MapReduce调度中,转移概率集合$\bar{\delta}_j$通过综合任务间依赖关系、资源匹配度及信息素浓度计算转移概率,决定蚂蚁下一步选择任务的可能性.例如,若某蚂蚁当前处于任务A,转移概率集合会根据任务A与其他任务的依赖关系、资源需求等因素,计算出该蚂蚁选择下一个任务B,C等的概率.蚂蚁依据$\bar{\delta}_j$中的概率分布进行随机选择,平衡探索(选择低概率但潜在更优路径)与利用(选择高概率路径),避免陷入局部最优.例如,当资源节点负载不均衡时,算法可通过转移概率动态调整任务分配方向.3)正反馈机制参数(∂_i).在MapReduce调度中,正反馈机制参数∂_i影响着信息素对蚂蚁选择的影响程度,初期设置较小∂_i(比如1.0,1.2等较小的值)以鼓励全局探索,后期增大参数会使蚂蚁更倾向于选择信息素浓度高的路径,即更有可能选择已被证明较好的任务组合方式,适应数据规模与任务复杂度的变化情况,提升调度方案的全局最优性.

Map函数对应的概率计算如式(2)所示:

$$P_i(a)=\frac{[\lambda_i]^{\alpha}[\eta_i]^{\beta}}{\sum_{i}^{m}[\lambda_i]^{\alpha}[\eta_i]^{\beta}},i\in \text{allowed}_k(t). \quad (2)$$

式中:P_i表示概率;a表示任务;allowed_k表示候选集;t表示迭代次数以及第t个蚂蚁;α,β为参数;$[\lambda_i]$,$[\eta_i]$表示启发式信息.本研究对于ACO算法的改进,旨在解决计算过程中局部最优、卡顿的问题.因此,引入启发函数,进行信息素更新,输出对应问题的值和解,保存最优值和最优解.具体包括成本目标函数和时间目标函数,启发函数如式(3)所示:

$$\left.\begin{aligned}\eta T_{ij}&=(\max T_i-T_{ij}+1)/(\max T_i-\min T_i+1),\\ \eta a_{ij}&=(\max C_i-C_{ij}+1)/(\max C_i-\min C_i+1).\end{aligned}\right\} \quad (3)$$

式中:T_{ij}表示时间属性;C_{ij}表示费用属性;i,j表示两种服务;$\max T_i$,$\min T_i$表示服务i的时间属性的上、下限;$\max C_i$,$\min C_i$表示服务i的费用属性的上、下限;η表示启发函数值.启发函数值与时间、成本成负相关,并且每个服务中包括了对应的信息素,其更新公式如式(4)所示:

$$\tau_{c_{ij}}=(1-\rho)\tau_{c_{ij}}+\Delta\tau_{c_{ij}}. \quad (4)$$

其中:$\tau_{c_{ij}}$表示C_{ij}信息素量;ρ表示信息素衰减系数;$\Delta\tau_{c_{ij}}$表示C_{ij}的信息素增量.

多租户服务定制是指在一个共享的系统实例中为不同的租户提供定制化的服务和功能.这种服务模式允许每个租户根据自己的需求对系统进行个性化配置,同时保证数据和业务的隔离[14].本文的多租户服务框架首先将租户的需求转化为业务流程,然后分解为抽象服务,进一步转化为具体的服务环节,最后反馈到服务器中进行访问和处理.提出的多租户服务定制流程基于ACO算法的SaaS多租户MapReduce调度,根据租户的功能需求、基本要求和租户偏好实现服务定制.

1.2　基于可变性模型的SaaS软件服务仿真技术

在SaaS多租户服务中,已经对多租户服务定制进行优化,但是SaaS软件服务存在重复使用和定制问题,难以满足多租户多样化的需求,且开发和维护成本较高.尽管一些公共服务可以在不同的流程实例之间重用,这些流程实例通常部署在服务容器中,但流程定义本身是不能重用的(即必须维护不同的流程示例),因此该模型在开发效率和维护复杂性方面存在不足[15-17],在此背景下,本文提出了可变性模型进行解决.

本文提出的抽象服务组装引入了可变性模型,将其应用于服务组装中.可变性模型引入了可变性设计思想,SaaS可变性管理包括服务组装模型和个性化定制.服务组装模型通过提供一个抽象服务组装模型,支持平台在运行阶段解释执行该模型,根据租户的个性化需求派生不同的流程实例[18-19].这些流程实例多态共存、互不影响,可支持多实例多租户的交付.个性化定制是SaaS软件针对特定租户的个性化需求进行定制,不影响其他租户[20].这种定制基于远程定制工具,软件供应商可以快速响应租户的变化需求,使得SaaS软件能在不断变化的环境中保持高度的灵活性和适应性,满足各种不同的业务需求.

本文提出的基于可变性模型的SaaS软件服务框架如图2所示.图2框架主要分为3个阶段,分别为多租户需求分析、抽象服务组装模型构建和服务组装规格部署及执行.在多租户需求分析中,设置必选服务和可选服务,以构成租户的需求集合.必选服务是指租户使用SaaS服务时不可或缺的基础功能模块,如用户认证、数据存储基础架构等;可选服务则是租户可根据自身业务特点选择性启用的功能,例如特定行业的数据分析模板、个性化的用户界面布局等.在抽象服务组装模型构建中,对相关协议和规格进行说明,在变体标签中添加租户信息的描述,对规格说明采用相应的租户配置文件描述租户信息;在服务组装规格部署及执行中,采用相应的支持平台解释执行抽象服务组装模型,根据用户的个性化需求创建不同的流程实例,实现租户需求多态共存.在应用程序的逻辑层面实现租户之间的数据隔离,通过使用租户标识符来实现,每个租户在系统中都有一个唯一的标识符,应用程序在访问数据库时,会根据租户标识符来限制访问和操作的范围,确保每个租户只能访问自己的数据;多租户架构的设计使多个租户在共享同一套程序的同时,保证数据隔离.在多租户架构中,将查询任务分布到多个节点上并行处理,通过分布式查询技术,这种架构模式降低了成本,提高了资源利用率,增强数据安全性和优化查询效率.

该模型的优点是:在SaaS多租户环境中,不同租户虽然有不同的业务需求,但往往存在诸多共性需求.可变性模型通过共性分析,识别并提取这些共性需求,将它们作为软件系统中的稳定部分.通过将这些共性需求封装为可重用的组件

或服务,可变性模型为SaaS软件提供了基础的功能框架,支持多个租户共享相同的核心功能.由于可变性模型将共性需求封装为可重用的组件或服务,这些组件或服务可以在不同的租户之间重复使用,从而降低软件开发的成本和时间.因此,该模型可基于同一问题领域内不同用户的共性需求和个性需求,实现SaaS多租户软件服务的适应性和可复用性.可变性模型在共性基础上还能进一步分析系统可变要素,针对可能发生变化的属性,明确系统中可配置或可定制的模块.通过对变化备选方案的抽象处理,每个变异点可以包含多个变体,每个变体代表一种特定的实现或配置方案.当新租户加入时,SaaS软件可通过寻找不同租户或系统间的共同要素,快速复制现有的基础功能框架,并根据新租户的具体需求,通过配置或定制变异点和变体来快速适应新环境.

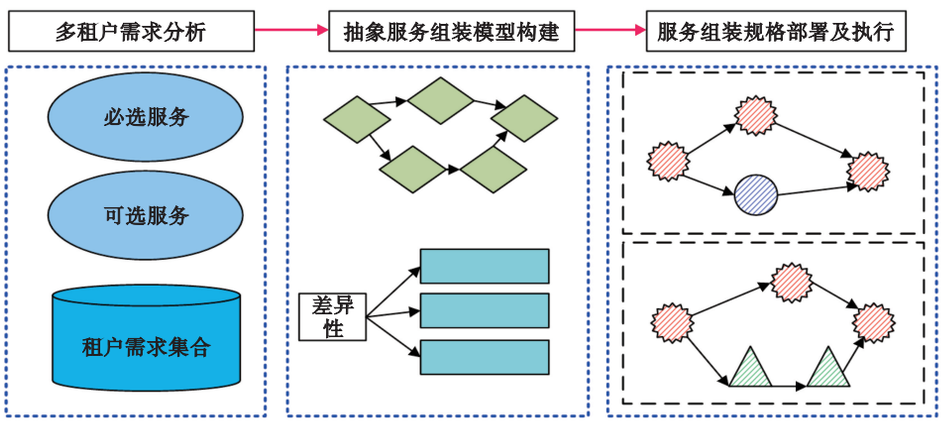

图2 基于可变性模型的SaaS软件服务框架

Fig. 2 SaaS framework based on variability model

2 SaaS多租户服务可变性管理仿真实验

为验证提出的基于ACO算法及可变性管理的SaaS多租户服务仿真技术,本文通过分析设计参数和实验结果,验证研究算法的优势和可行性.实验环境配置如下:操作系统为WIN 10-64bit,CPU为Intel(R) Core(TM)i5-4590 CPU@3.30 GHz,GPU为NVIDIA GeForce GTX 1050 Ti,内存为8 GB DDR5.

实验数据采用国际通用的ORLIB(operations research library)基准测试数据集a,b,c,d实例作为测试对象.该数据集包含不同规模与复杂度的任务组合,能够全面覆盖多样化应用场景,为算法性能测试提供充分的验证依据.在对比实验设计中,本文采用文献[8]算法作为对比基线算法,这是因为该算法在复合服务场景下服务参与者动态变化时的用户隐私保护方面具有典型代表性,并且与研究的SaaS多租户服务场景存在部分共性,通过对比可凸显本文算法在资源使用效率、收敛速度等方面的优势.实验方案旨在从多个维度(如资源使用情况、迭代代数、运行时间等)全面评估本研究算法的性能,验证其在SaaS多租户服务中的有效性和优越性.

图3为研究算法与对比算法的SaaS租户服务资源使用情况,对比算法为文献[8]提出的一种隐私策略自动更新方法,其能在复合服务中服务参与者发生变化时增强用户隐私保护.a,b,c,d分别为来自通用基准测试数据集ORLIB的不同测试实例,a(5.100-00),b(5.150-10),c(5.150-30),d(5.200-10).实验模拟了10个租户的配置方案以验证隔离情况,表1为不同SaaS租户配置方案及运行时间,可以看出,其中User7的运行时间最短,为512 ms,User5的运行时间最长,为951 ms,但是整体来看,运行时间规律并不明显,可见租户能够有效派生服务配置方案,且租户之间数据隔离是有效的.

下面针对在SaaS及ODE(on-premises deployment)部署下对某服务组装进行评估.ODE是一种将软件安装在租户自己服务器上的部署方式,租户需要自行购买和维护服务器硬件、网络设备等基础设施,同时负责软件的安装、配置、升级等操作.在该实验中,不同租户(User0~User9)的配置方案由a1,a2,a3的不同组合构成,以此实现租户的定制化服务.a1,a2,a3是预定义的3种不同的服务功能变体(或配置模板),通过组合来适配不同租户的业务需求,表1展示的是通过不同

配置方案a1,a2,a3实现了10个租户的需求配置.实验结果如下:ODE部署下执行时间在各配置方案中相对稳定,波动范围在1 422~1 650 ms,SaaS部署中执行时间在不同配置方案中有所波动,最短为1 426 ms,最长为1 652 ms;ODE部署切换资源耗费在各配置方案中波动较大,最短为218 ms(配置方案8),最长为251 ms(配置方案3);SaaS部署切换资源耗费相对较为稳定,波动范围在18~24 ms,远低于ODE部署下的耗费;ODE部署切换资源耗费占空比在各配置方案中波动较大,最低为12.36%(配置方案6),最高为14.6%(配置方案4);SaaS部署切换资源耗费占空比相对较为稳定,波动范围在1.12%~1.51%,远低于ODE部署下的占空比.以上数据说明,无论是从绝对数值还是波动范围看,SaaS在切换资源耗费上明显优于ODE.这表明SaaS在资源切换方面更加高效,能够更快地完成资源调整.与切换资源耗费类似,SaaS在切换资源耗费占空比上也表现出显著的优势.较低的占空比意味着在相同时间内,SaaS能够更有效地利用资源,减少因资源切换导致的性能损耗.

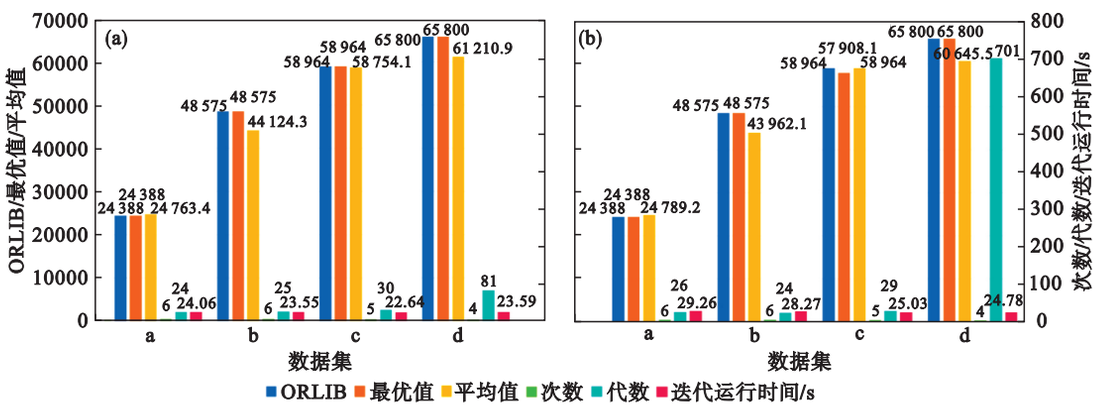

图3　研究算法与对比算法的SaaS租户服务资源使用情况

Fig. 3　SaaS tenant service resource utilization of proposed algorithm and comparison algorithm

(a)—研究算法;(b)—对比算法.

表1　不同SaaS租户配置方案及运行时间

Table 1　Configuration schemes and running time of different SaaS tenants

租户	配置维度								运行时间/ms
	1	2	3	4	5	6	7	8	
User0	a1	a2	a3	a3	a2	a1	a3	a3	772
User1	a1	a3	a1	a3	a1	a1	a1	a2	685
User2	a3	a3	a1	a2	a3	a1	a1	a3	635
User3	a1	a1	a2	a1	a1	a2	a1	a3	616
User4	a1	a2	a3	a1	a2	a3	a2	a3	684
User5	a2	a1	a2	a1	a3	a3	a2	a3	951
User6	a3	a2	a1	a3	a3	a3	a3	a3	547
User7	a2	a1	a3	a1	a2	a1	a3	a2	512
User8	a2	a3	a3	a1	a1	a1	a3	a2	543
User9	a3	a2	a2	a3	a3	a1	a1	a1	721

简单查询和复杂查询考察的是SaaS租户在不同服务组装软件交互的时间使用情况[6],在此场景的简单语句查询下,传统服务交互和可变性管理服务交互的时间使用变化区间分别为1 100~1 250 ms,1 000~1 150 ms,可见研究的可变性管理服务交互能够提高服务查询和定制的效率.在此场景的复杂语句查询下,传统服务交互和可变性管理服务交互的使用时间变化区间分别为1 125~1 250 ms,1 025~1 180 ms,使用时间变化相差不大.说明研究提出的可变性服务交互时间耗费始终小于传统服务交互,可见所提技术对数据查询和处理效率的影响是积极的.

3 结 语

在SaaS多租户服务中,可变性管理是指对服务中可变服务的管理和配置,包括功能模块、业务流程、用户界面等,其目标是确保SaaS服务能够灵活地适应不同租户的需求变化,同时保持服务的稳定性和一致性.本研究一方面将ACO算法深度应用于SaaS多租户服务的MapReduce任务调度,通过独特的启发函数设计和信息素更新机制,能有效解决计算过程中局部最优和卡顿的问题,提升了任务调度的效率和服务定制的质量,相较于传统调度策略,能够更精准地匹配租户需求.另一方面,提出的可变性模型打破传统服务组装软件交互模式的局限,通过抽象服务组装和变异点、变体设计,实现SaaS软件服务的适应性和可复用性,提高开发效率,降低维护成本.这种将优化算法与可变性管理相结合的方式,为SaaS多租户服务的发展提供了新的技术思路和方法.然而,在SaaS多租户应用中的用户安全隐私方面研究深度还不足,因此在后续的研究中需要对此进一步探索.

参考文献:

[1] 李凌书,邬江兴,刘文彦.SaaS云环境下基于容器指纹匿名的网络欺骗方法[J].信息安全学报,2022,7(2):72-86.
(Li Ling-shu, Wu Jiang-xing, Liu Wen-yan. An anonymous network deception method based on container fingerprint modification for SaaS applications [J]. *Journal of Cyber Security*, 2022,7(2):72-86.)

[2] 管婉青,张海君,路兆铭.基于DRL的6G多租户网络切片智能资源分配算法[J].北京邮电大学学报,2020,43(6):132-139.
(Guan Wan-qing, Zhang Hai-jun, Lu Zhao-ming. Intelligent resource allocation algorithm for 6G multi-tenant network slicing based on deep reinforcement learning [J]. *Journal of Beijing University of Posts and Telecommunications*, 2020, 43(6): 132-139.)

[3] Kuru K. Management of geo-distributed intelligence: deep insight as a service (DINSaaS) on forged cloud platforms (FCP)[J]. *Journal of Parallel and Distributed Computing*, 2021, 149: 103-118.

[4] 程华盛,敬超.面向多租户数据中心的联邦学习架构下通信开销优化方法[J].计算机应用研究,2024,41(9):2823-2830.
(Cheng Hua-sheng, Jing Chao. Optimization method for communication overhead in federated learning architecture for multi-tenant data center [J]. *Application Research of Computers*, 2024,41(9):2823-2830.)

[5] 邓彭冲.高职院校体育管理云信息系统设计与实现[J].自动化技术与应用,2020,39(12):47-50,65.
(Deng Peng-chong. Design and implementation of sports management cloud information system in higher vocational colleges [J]. *Techniques of Automation and Applications*, 2020, 39(12): 47-50, 65.)

[6] 张纪林,邵玉曹,任永坚,等.支持多租户模式的业务流程动态定制模型[J].计算机科学,2022,49(增刊1):705-713.
(Zhang Ji-lin, Shao Yu-cao, Ren Yong-jian, et al. Dynamic customization model of business processes supporting multi-tenant[J]. *Computer Science*, 2022, 49(sup1): 705-713.)

[7] 王廷,刘刚.支持网络切片和绿色通信的软件定义虚拟化接入网[J].计算机研究与发展,2021,58(6):1291-1306.
(Wang Ting, Liu Gang. Software defined virtualized access network supporting network slicing and green communication [J]. *Journal of Computer Research and Development*, 2021, 58(6): 1291-1306.)

[8] Ke C B, Xiao F, Huang Z Q, et al. A user requirements-oriented privacy policy self-adaption scheme in cloud computing [J]. *Frontiers of Computer Science*, 2022, 17(2): 172203.

[9] 徐海洋,刘海龙,陈先,等.基于组合负载预测模型的多租户数据库弹性伸缩方法[J].软件学报,2025,36(3):981-994.
(Xu Hai-yang, Liu Hai-long, Chen Xian, et al. Elastic scaling method for multi-tenant databases based on hybrid workload prediction model[J]. *Journal of Software*, 2025, 36(3): 981-994.)

[10] 冉金鹏,王翔,赵尚弘,等.基于果蝇优化的虚拟SDN网络映射算法[J].信息网络安全,2020,20(6):65-74.
(Ran Jin-peng, Wang Xiang, Zhao Shang-hong, et al. Virtual SDN network embedding algorithm based on fruit fly optimization[J]. *Netinfo Security*, 2020, 20(6): 65-74.)

[11] 文宇鸿,周游,吴秋霖,等.多租户固态盘服务质量保障技术综述[J].计算机研究与发展,2023,60(3):555-571.
(Wen Yu-hong, Zhou You, Wu Qiu-lin, et al. Quality of service guaranty technology of multi-tenant solid-state drives: a survey [J]. *Journal of Computer Research and Development*, 2023, 60(3): 555-571.)

[12] 李妍,郭得科,曹晓丰,等.应用感知的数据中心网络多租户共享方法[J].计算机学报,2021,44(7):1363-1377.
(Li Yan, Guo De-ke, Cao Xiao-feng, et al. Application-aware network sharing for multi-tenant data center [J]. *Chinese Journal of Computers*, 2021, 44(7): 1363-1377.)

[13] Ortegat G, Grolaux D, Riviere E, et al. Engineering the transition of interactive collaborative software from cloud computing to edge computing[J]. *Proceedings of the ACM on Human-Computer Interaction*, 2022, 6: 1-31.

[14] 刘海龙,王硕,侯舒峰,等.云多租数据库资源规划调度技术综述[J].软件学报,2025,36(1):446-468.
(Liu Hai-long, Wang Shuo, Hou Shu-feng, et al. Survey on resource planning and scheduling technologies for multi-tenant cloud databases[J]. *Journal of Software*, 2025, 36(1): 446-468.)

[15] 朱斌庚,陈晓曼.基于云计算SaaS的电能质量在线监测平台设计和开发[J].自动化与仪器仪表,2023(8):49-51,55.
(Zhu Bin-geng, Chen Xiao-man. Design and development of power quality online monitoring platform based on cloud computing SaaS [J]. *Automation & Instrumentation*, 2023(8): 49-51, 55.)

[16] 胡宇翔,冯旭,董永吉,等.基于深度强化学习的多租户算网资源分配算法[J].物联网学报,2024,8(4):34-44.
(Hu Yu-xiang, Feng Xu, Dong Yong-ji, et al. Multi-tenant computing network resource allocation algorithm based on deep reinforcement learning[J].*Chinese Journal on Internet of Things*, 2024,8(4):34-44.)

(下转第98页)