基于云计算的资源调度算法优化研究

张渝新
（重庆邮电大学软件工程学院，重庆 400000）

摘　要：启发式算法在云计算资源调度中的优化和评估模型是一个关键的研究领域。对启发式算法在云计算资源调度中的优化方法进行了详细说明，并建立了资源调度算法评估模型。通过研究，可以深入理解启发式算法在云计算资源调度中的优化和评估模型，并为实际应用提供指导和参考。

关键词：云计算；资源调度；启发式算法
DOI:10.16184/j.cnki.comprg.2023.10.015

1　概述

云计算作为一种灵活、可扩展的计算模式，已经在各领域广泛应用。然而，随着云计算规模的扩大和应用场景的复杂化，资源调度成为云计算中的一个关键问题[1]。

采用合理的资源调度算法，可以最大限度地利用云计算环境中的资源，实现资源的合理分配和利用，提高整体资源利用率，避免资源的过度分配和能源的浪费，降低云计算系统的能源消耗，提升系统的能源效率，减少资源竞争和冲突，提高系统的响应速度和性能，提供更好的用户体验。通过有效的资源调度策略，可以确保各类任务和服务的及时响应，提供高质量的服务，提升用户满意度和用户体验。因此，研究基于启发式算法的资源调度算法优化在优化云计算资源调度效果、提高系统性能和能源效率方面具有重要的研究价值和实际应用意义。

2　云计算资源调度算法应用现状

云计算是一种基于互联网的计算模式，通过共享的计算资源和服务，按需提供可扩展的计算、存储和网络资源。云计算提供了不同的服务模型，包括基础设施即服务（IaaS）、平台即服务（PaaS）和软件即服务（SaaS）。云计算可以部署为公有云、私有云、混合云和边缘云等不同的部署模型[2]。

云计算环境涉及多种资源，包括计算资源、存储资源、网络资源等，调度需要考虑不同资源之间的关系和约束。资源调度的目标是实现高效的资源利用和优化性能，以满足用户的需求和服务级别协议（SLA）要求。云计算需要具备弹性和可伸缩性，需要根据实际需求动态调整资源分配，并能够应对负载的变化。资源调度算法需要考虑能源消耗和环境影响，以实现节能和可持续发展目标[3]。

现有的云计算资源调度算法包括以下类别。

（1）基于优先级的调度算法。根据任务的优先级，对高优先级的任务优先分配资源，保证重要任务的执行。

（2）基于队列的调度算法。按照任务的到达时间或提交顺序进行调度，实现公平性和先来先服务的原则。

（3）基于负载均衡的调度算法。通过动态分配任务和资源，实现负载均衡，提高资源利用率和性能。

（4）基于预测的调度算法。通过分析历史数据和趋势预测，预测未来的资源需求，进行提前调度和规划。

（5）基于启发式算法的调度算法。利用启发式算法的特点和优势，设计和优化资源调度算法，例如，遗传算法、蚁群算法和粒子群优化算法等。

上述算法只是云计算资源调度算法中的一部分，每种算法都有其特点和适用场景。综合考虑资源调度的挑战和需求，以及现有算法的优缺点，发现基于启发式算法的资源调度算法可以提供更灵活、高效的资源管理和调度策略，优化云计算环境的性能和能源利用效率。

3　启发式算法在云计算资源调度中的优化

3.1　启发式算法

启发式算法是一类基于经验和启发性信息的问题求解方法，主要用于解决复杂的优化问题。与传统的确定性算法相比，启发式算法具有以下特点。

（1）近似解。启发式算法不保证找到全局最优解，而是寻找一个较好的近似解。它通过对问题进行简化、约束或限制搜索空间提高求解效率。

（2）可扩展性。启发式算法可以应用于各种问题，无论问题规模的大小和复杂度如何，它都能够灵活地适应和扩展。

（3）非确定性。启发式算法通常基于随机性策略，使用随机性的操作来引导搜索过程，以避免陷入局部最优解。

作者简介：张渝新（2001—），男，本科，研究方向为软件工程。

（4）启发信息利用。启发式算法利用问题的特定启发信息或经验知识指导搜索过程，加速求解速度和提高解的质量。

3.2 遗传算法在资源调度中的应用

遗传算法是一种启发式优化算法，模拟了生物进化的过程，通过对候选解的群体进行选择、交叉和变异等操作，逐步迭代搜索最优解。在资源调度问题中，遗传算法可以优化资源的分配和调度，以提高资源利用率和系统性能。

（1）遗传算法在资源调度中的实现方法如图 1 所示。

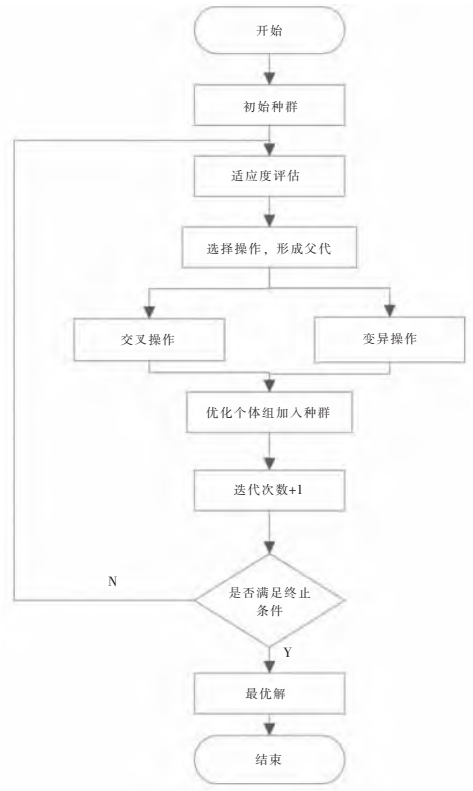

图 1 遗传算法在资源调度中的实现方法

1）初始化种群。设种群大小为 N，种群表示为 $P=\{p_1,p_2,...,p_N\}$，其中，p_i 为第 i 个个体，每个个体由 D 个基因组成，表示资源的分配情况。基因可以用二进制编码来表示，例如 $p_i=\{g_1,g_2,...,g_D\}$，其中，g_i 为第 i 个基因，g_i=0 或 1，表示资源的分配情况。

2）适应度评估。定义适应度函数 $f(p_i)$，根据问题的具体要求和性能指标计算每个个体的适应度值。适应度函数可以是资源利用率、性能指标的函数。

3）选择操作。根据适应度值选择父代个体。采用轮盘赌选择策略，计算每个个体的选择概率 $P(p_i)=f(p_i)/\Sigma f(p_j)$，其中，$\Sigma f(p_j)$ 为种群中所有个体的适应度值之和。

随机生成 N 个随机数 $r_1,r_2,...,r_N$，满足 $0<r_i<1$。

根据选择概率，选取满足 $r_i<P(p_i)$ 的个体作为父代，形成父代群体 $Q=\{q_1,q_2,...,q_N\}$。

4）交叉操作。对父代个体进行交叉操作，生成子代个体。采用单点交叉方式，随机选择一个位置 k，将两个父代个体在位置 k 之后的基因进行交换，形成新的子代个体。

交叉操作可以表示为子代个体 $r=\text{crossover}(q_i,q_j)$，其中 q_i 和 q_j 为父代个体，r 为子代个体。

5）变异操作。对子代个体进行变异操作，引入一定的随机性。随机选择一个位置 k，将子代个体 r 在位置 k 处的基因取反，形成新的变异后的子代个体。

6）更新种群。将生成的子代个体加入种群中，形成新的种群 $P'=P\cup\{r\}$，其中，$\cup$ 表示集合的并集操作。

7）终止条件。根据预定的迭代次数或是否满足终止条件来判断是否停止迭代。终止条件可以是达到一定的适应度值、种群没有显著改进等。

8）输出最优解。根据迭代过程中保存的最优个体，输出最优解作为最优的资源调度方案。

（2）优化遗传算法的方法如下。

1）调整选择策略。选择策略影响个体被选中的概率，因此可以根据问题的特点调整选择策略，例如，引入选择压力，提高较优解的选中概率。

2）改变交叉和变异操作。调整交叉和变异的方式和概率，以增加解的多样性和搜索空间的覆盖范围。

3）考虑约束条件。在选择、交叉和变异操作中，加入对约束条件的处理，确保生成的解满足问题的约束条件。

4）多种群策略。引入多种群的思想，将种群分为多个子群体，每个子群体独立进行遗传操作，以增加搜索的多样性和效率。

3.3 蚁群算法在资源调度中的应用

蚁群算法是一种启发式优化算法，模拟了蚂蚁在寻找食物过程中的行为。在资源调度问题中，蚁群算法可以用于优化资源的分配和调度，以提高资源利用率和系统性能。蚁群算法流程如图 2 所示。

（1）蚁群算法的初始化。

设定蚂蚁数量为 M，资源分配方案数量为 N。初始化蚂蚁的位置，将每只蚂蚁的位置表示为一个 M 维向量，其中，每个维度代表一种资源的分配方案，初始值为随机选择的一种方案。初始化信息素浓度，使用一个 $N\cdot M$ 的矩阵 $\boldsymbol{\tau}$，其中，$\boldsymbol{\tau}(i,j)$ 为第 i 个蚂蚁在第 j 个位置上的信息素浓度。

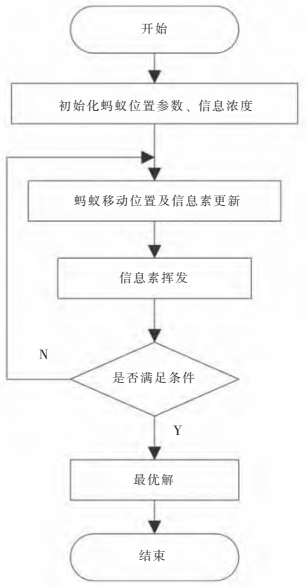

图 2　蚁群算法流程

（2）蚂蚁的移动和信息素更新。

对于每只蚂蚁，根据当前位置的信息素浓度和启发函数计算选择下一个位置的概率。选择下一个位置时，使用轮盘赌选择方法，根据概率分布进行随机选择。更新蚂蚁的位置，将当前位置更新为选择的下一个位置。在移动过程中，每只蚂蚁会释放信息素到经过的路径上。

更新路径上的信息素浓度，可以使用如公式（1）所示：

$$\tau(i,j)=(1-\rho)\times\tau(i,j)+\Delta\tau(i,j) \quad (1)$$

其中，ρ 为信息素挥发系数（$0<\rho<1$）；$\Delta\tau(i,j)$ 为蚂蚁在路径上释放的信息素增量，可以根据蚂蚁的适应度值或其他指标来计算。

（3）信息素的挥发。

对于每个位置 j，对信息素进行挥发操作，可以使用如公式（2）所示进行更新：

$$\tau(i,j)=(1-\rho)\times\tau(i,j) \quad (2)$$

其中，ρ 为信息素挥发系数。

（4）终止条件。

根据预定的迭代次数或是否满足终止条件来判断是否停止迭代。

（5）输出最优解。

在迭代过程中，保存每次迭代后的最优解。最优解是具有最优适应度值的资源分配方案。

以上是蚁群算法在资源调度中的具体实现步骤，可以根据具体问题和需求进行调整和优化。在实际应用中，需要根据具体问题进行参数设置和适应度函数的设计，以及选择合适的信息素更新策略和挥发策略。

3.4　粒子群优化算法在资源调度中的优化方法

粒子群优化算法（PSO）是一种基于群体智能的优化算法，模拟了鸟群或鱼群的集体行为。在资源调度问题中，粒子群优化算法可以优化资源的分配和调度，以提高资源利用率和系统性能。粒子群优化算法在资源调度中的详细步骤如下。

（1）初始化粒子群。

设定粒子群的大小为 N，每个粒子表示一种云计算资源分配方案。每个粒子的位置表示为一个 D 维向量，其中，每个维度代表一种资源的分配情况。

初始化粒子的位置和速度，位置用一个 D 维向量表示，速度也用一个 D 维向量表示，初始值为随机选择的一种方案。

（2）适应度评估。

对于每个粒子，根据问题的具体要求和性能指标计算其适应度值。可以根据具体问题设计适应度函数，例如，资源利用率、成本最小化等。

（3）更新粒子的位置和速度。

对于每个粒子，根据其当前位置和速度，以及全局最优解和个体最优解，进行位置和速度的更新。

粒子的位置更新公式如公式（3）所示：

$$x(t+1)=x(t)+v(t+1) \quad (3)$$

粒子的速度更新公式如公式（4）所示：

$$v(t+1)=w\times v(t)+c_1\times r_1\times[\text{pbest}-x(t)]+c_2\times r_2\times[\text{gbest}-x(t)] \quad (4)$$

其中，$x(t+1)$ 为粒子在下一时刻的位置；$v(t+1)$ 为粒子在下一时刻的速度；w 为惯性权重；c_1 和 c_2 为学习因子；r_1 和 r_2 为随机数；pbest 为粒子的个体最优解；gbest 为粒子群的全局最优解。

（4）更新个体最优解和全局最优解。

对于每个粒子，根据当前的适应度值和个体历史最优适应度值进行比较，更新个体最优解。

对于粒子群，根据所有粒子的适应度值和全局历史最优适应度值进行比较，更新全局最优解。

（5）终止条件。

根据预定的迭代次数或是否满足终止条件判断是否停止迭代。

（6）输出最优解。

在迭代过程中，保存每次迭代后的最优解。最优解是具有最优适应度值的资源分配方案。

以上是粒子群优化算法在云计算资源调度中的具体实现步骤，可以根据具体问题和需求进行调整和优化。在实际应用中，需要根据具体问题进行参数设置和适应

(下转第 89 页)

参考文献

[1] 张浩斌. 基于开放式云平台的开源在线评测系统设计与实现 [J]. 计算机科学, 2012 (S3): 339-343.

[2] 曾棕根. 源程序在线评测系统技术改进 [J]. 计算机工程与应用, 2011, 47 (4): 68-71.

[3] 李文新, 郭炜. 北京大学程序在线评测系统及其应用 [J]. 吉林大学学报: 信息科学版, 2005 (S2): 170-177.

[4] 杨志伟, 曾艳姗. 基于 Linux 的 ACM 在线评测系统研究 [J]. 计算机与现代化, 2010 (6): 166-169.

[5] 王涛春, 罗永龙, 左开中. 基于在线评测的数据结构实践教学探讨 [J]. 计算机教育, 2010 (10): 88-91.

[6] 周高嵌, 彭四伟. 源代码在线评测系统中剽窃检测技术的研究与实现 [J]. 计算机与信息技术, 2005 (12): 85-87.

[7] 李博, 孟成博. 对 HUSTOJ 在线评测系统的若干优化与创新 [J]. 现代计算机: 中旬刊, 2013 (12): 47-50.

(上接第 85 页)

度函数的设计，以及选择合适的学习因子和惯性权重。

4 资源调度算法评估模型

为了采用云计算资源调度算法进行定量评估，建立以下评估模型。

（1）资源利用率。

定义总可用资源为 R_total、实际使用的资源为 R_used，则资源利用率 U 可以如公式（5）所示：

$$U=\text{R\_used}/\text{R\_total} \quad (5)$$

（2）响应时间。

定义任务的排队时间为 T_queue、调度时间为 T_schedule、执行时间为 T_execution，则响应时间 T_response 可以表示为公式（6）所示：

$$\text{T\_response}=\text{T\_queue}+\text{T\_schedule}+\text{T\_execution} \quad (6)$$

（3）成本效益。

定义能耗为 E、硬件资源使用费用为 C，则成本效益 B 可以表示为公式（7）所示：

$$B=1/(E+C) \quad (7)$$

（4）可扩展性。

定义系统的吞吐量为 T_throughput，并发处理能力为 C_concurrent，则可扩展性 S 可以表示为公式（8）所示：

$$S=\text{T\_throughput}/\text{C\_concurrent} \quad (8)$$

（5）公平性。

定义不同任务的需求为 D_task1,D_task2,...,D_taskn，定义任务的执行时间为 T_task1,T_task2,...,T_taskn，则公平性 F 可以表示为公式（9）所示：

$$F=\min(\text{D\_task1}/\text{T\_task1},\text{D\_task2}/\text{T\_task2},...,\text{D\_taskn}/\text{T\_taskn}) \quad (9)$$

根据以上定义和描述，可以构建一个综合的评估模型来评估云计算资源调度算法的性能。该模型可以根据具体的场景和需求进行扩展和调整，包括引入权重因子、优化目标等。

通过建立这样的评估模型，可以定量地评估和比较不同的云计算资源调度算法，并根据实际需求选择最适合的算法。这样的模型能够提供决策支持和指导，以优化资源调度过程、提高系统性能和效益。

5 结语

研究表明，不同的启发式算法可以针对不同的调度问题和目标进行优化。遗传算法适用于全局优化问题，蚁群算法适用于离散优化问题，粒子群优化算法适用于连续优化问题。选择合适的启发式算法可以根据具体的需求和问题特性。

对于启发式算法的评估模型，可以综合考虑多个性能指标，例如，资源利用率、响应时间、成本效益等。通过实验设计和数据集选择，可以进行定量的评估并比较不同启发式算法在云计算资源调度中的性能。根据评估结果，可以选择最优的启发式算法作为云计算资源调度的解决方案。此外，还可以进一步优化和改进启发式算法，例如，调整参数、引入新的启发式规则等，以进一步提高调度性能和效果。

启发式算法在云计算资源调度中具有重要的优化能力，并且可以通过合适的评估模型进行性能评估和比较。选择最优的启发式算法能够提高资源利用率、降低成本、改善响应时间等，为云计算系统提供高效的资源调度解决方案。

参考文献

[1] 胡程鹏, 薛涛. 基于遗传算法的 Kubernetes 资源调度算法 [J]. 计算机系统应用, 2021, 30 (9): 152-160.

[2] 王镇道, 张一鸣, 石雪倩. 基于竞争粒子群算法的云计算资源调度策略 [J]. 湖南大学学报 (自然科学版), 2021, 48 (6): 80-87.

[3] 贾晓前. 云计算环境下的资源调度算法探究 [J]. 电脑编程技巧与维护, 2021 (11): 94-96.