

# 浙 江 大 学

## 硕士学位论文开题报告

(专业学位)

论文题目: 云原生环境下微服务配置优化方法研究及框  
架实现

姓 名: 刘京宗

学 号: 22451040

专 业: 软件工程

院 别: 软件学院

导 师: 才振功

二零二五 年 四月

# 目 录

1. 课题来源及类型 .....	- 1 -
2. 课题的意义及国内外现状分析 .....	- 1 -
3. 课题的研究目标、研究内容和拟解决的关键问题 .....	- 4 -
3.1 课题研究目标 .....	- 4 -
3.2 课题研究内容 .....	- 5 -
3.3 拟解决的关键问题 .....	- 6 -
4. 课题的研究方法、设计及试验方案，可行性分析 .....	- 7 -
4.1 课题的研究方法 .....	- 7 -
4.2 课题设计方案 .....	- 8 -
4.3 课题可行性分析 .....	- 9 -
5.课题计划进度和预期成果 .....	- 9 -
5.1 计划进度 .....	- 9 -
5.2 预期成果 .....	- 10 -

## 1. 课题来源及类型

本课题来源于云原生环境下微服务架构的配置优化问题。随着云计算和微服务技术的飞速发展，越来越多的企业选择在云原生环境中部署应用程序。微服务架构因其拆分单一应用为多个独立的小型服务，提高了灵活性和扩展性，但也带来了配置优化的复杂性。具体而言，微服务架构中服务数量众多，且每个服务都有多个可配置参数，服务之间依赖关系复杂，如何合理优化微服务的资源配置和软件参数以提升整体性能，成为了该研究的核心课题。

该课题属于计算机科学与技术领域的应用研究，聚焦于云计算和微服务架构的优化方法。具体而言，研究的核心为微服务架构下资源配置与软件参数协同优化的问题，是典型的性能优化领域的研究课题。此课题结合了云原生环境下微服务应用的性能提升策略，旨在提出新的优化方法，解决传统微服务架构中存在的配置优化难题。

## 2. 课题的意义及国内外现状分析

随着云计算、微服务架构和云原生技术的迅猛发展，越来越多的企业和组织选择在云环境中部署和运行应用程序。微服务架构将传统的单一应用程序拆分为多个独立的小服务，通过每个服务的独立部署与管理，实现了高度的灵活性、可扩展性和高效性。然而，微服务架构的成功实施带来了新的挑战，尤其是在如何对微服务应用进行高效的配置优化上。随着服务数量的激增，服务间复杂的依赖关系和配置参数的多样性，使得微服务应用的资源配置和软件参数优化成为亟待解决的重要问题。

在微服务架构下，应用的性能不仅依赖于服务间的交互，还与微服务所需的系统资源（如 CPU、内存、网络带宽等）以及每个服务所部署的软件参数（如数据库缓存、Web 服务器参数等）密切相关。然而，由于微服务应用通常包含大量服务，每个服务的配置参数也非常复杂，因此如何高效地调整这些参数，以最大化应用性能，成为了当前云原生环境下研究的热点。

优化微服务架构中的资源配置和软件参数，能够大幅提高系统的运行效率，减少资源浪费。例如，通过合理的资源配置和软件调优，可以显著提升微服务应用的吞吐量、响应速度和系统稳定性。同时，优化方案能够降低云平台的运营成本，提升系统的可靠性和弹性，减少硬件资源浪费，提高资源利用率。在云原生环境下，合理配置微服务应用的资源和软件参数，不仅能够提高性能，还能够提升系统的可维护性和可扩展性。

此外，由于微服务架构的组件化特点，开发人员可以通过灵活的配置调整应对不断变化的负载需求，但与此同时也面临着配置调整的复杂性。在这种环境下，如何快速且有效地进行配置优化，使得微服务在保证性能的同时减少资源浪费，是一项具有重要理论和实践价值的课题。

因此，针对云原生环境下微服务配置优化问题，提出资源配置与软件参数协同优化方案并进行框架实现，不仅能够有效提升微服务应用的性能，还可以为微服务架构的进一步发展提供指导，为企业提供更具成本效益的云原生应用配置解决方案。

传统的软件配置优化问题，通常集中在提升大规模软件系统性能、优化计算资源的使用效率上。在早期的大数据框架（如 Hadoop、Spark、Flink）中，研究者们通过调节系统资源配置参数（如 CPU、内存、磁盘等）来优化性能。此外，应用层的性能优化也主要依赖于数据库等软件的调优，例如，调整数据库的缓存大小、查询优化等。这些传统的优化方法主要针对单一应用程序或单一软件栈，忽略了不同软件栈之间的相互依赖和多层次的配置需求。

在云计算和大数据领域，已有一些针对软件配置优化的研究提出了有效的解决方案。例如，Herdotu 等提出的 Starfish 系统，利用成本模型模拟 MapReduce 执行过程，并结合历史执行数据自动调整任务调度策略，成功优化了大数据框架的性能。然而，这些方法主要集中于传统的单体应用系统或特定的大数据处理框架，未能适应微服务架构中的高度分布式和服务间复杂依赖的特点。

随着云计算和微服务架构的流行，传统的软件配置优化方法在新兴的云原生环境中面临挑战。微服务架构中，应用程序被拆分成多个小服务，服务间的依赖关系复杂，每个服务都可能拥有独立的资源配置需求，且服务间的资源共享和竞争问题也更加突出。因此，传统的单一优化方法难以应对微服务架构下的配置优化

问题，需要进一步拓展和创新配置优化的策略。

目前，针对微服务架构下的配置优化问题，国内外研究的重点主要集中在两类优化方案上：一类是通过优化微服务的资源配置（如 CPU、内存、存储等）以提高应用性能，另一类则是通过优化微服务内部的软件参数（如数据库配置、Web 服务器配置等）来提升性能。

在资源配置优化方面，研究者们主要关注如何根据微服务的特点，动态调整资源分配，以应对不同负载下的性能需求。例如，Gias 等设计了一个微服务资源自动扩缩控制器，通过动态调整服务副本数和 CPU 资源，提升微服务的性能。Sinan 模型则通过动态监控各层服务的资源使用情况，实现了基于 QoS 目标的资源优化。

然而，单独优化资源配置或软件参数往往无法获得最优的性能提升，尤其在微服务架构下，由于服务之间的复杂依赖，单独调整某一方面的配置可能导致其他服务性能的下降。因此，近年来有部分研究开始关注资源配置与软件参数的协同优化。例如，Somashekar 等提出的微服务配置优化方法，将服务部署的多个软件参数（如 Nginx、Memcached、MongoDB 等）与资源配置结合，通过协同优化提升微服务应用性能。然而，这些研究仍然未能全面解决资源配置和软件参数之间复杂的依赖关系，且大多数研究仅通过理论分析进行探讨，缺乏足够的实践验证。

另一个研究方向是自动化配置优化。在微服务架构中，服务和参数数量庞大，手动调优的复杂性和错误率较高。为此，研究者们尝试通过机器学习、贝叶斯优化等方法，自动化地进行微服务应用的配置优化。例如，使用贝叶斯优化算法，通过智能搜索算法调整微服务配置，避免传统优化方法中的高维度搜索空间问题。这种方法能够在较短时间内找到接近最优的配置，并自动调整各服务的资源配置和软件参数，减少人为干预。

尽管国内外在微服务配置优化方面已有一定的研究进展，但整体上仍面临许多挑战。首先，微服务应用的配置空间非常庞大，传统的优化方法往往无法应对复杂的配置搜索空间；其次，微服务应用中的资源配置与软件参数之间存在强烈的依赖关系，这使得协同优化变得异常复杂；最后，如何在保证配置优化效果的同时，提高优化过程的自动化水平和效率，也是目前研究的一个难点。

综上所述，国内外在微服务配置优化领域的研究虽已取得一定成果，但仍存在许多不足和挑战。现有的研究大多侧重于单一优化策略，未能充分考虑资源配置与软件参数之间的协同作用。随着微服务架构在云原生环境中的广泛应用，如何通过协同优化提升微服务应用的整体性能，尤其是在高维配置空间和复杂依赖关系下，成为了亟待解决的关键问题。因此，提出一种资源配置与软件参数协同优化的方法，并结合贝叶斯优化算法和自动化框架进行优化，具有重要的学术价值和实际应用意义。

### 3. 课题的研究目标、研究内容和拟解决的关键问题

#### 3.1 课题研究目标

本课题的主要目标是针对云原生环境下微服务架构中资源配置与软件参数的协同优化问题，提出一种基于贝叶斯优化算法的高效优化方法，并设计和实现一个在线配置优化框架。通过本课题的研究，旨在：

提出资源配置与软件参数协同优化方案：分析云原生微服务架构中的性能瓶颈，研究资源配置与软件参数之间的相互依赖关系，提出一种综合优化方案，旨在提高微服务应用的整体性能。

设计并实现云原生微服务应用配置优化框架：基于现代云原生技术栈（如容器、Kubernetes、DevOps 工具等），设计并实现一个自动化的在线配置优化框架，使得微服务应用能够动态地根据当前工作负载自动调整资源配置与软件参数。

提升微服务应用的性能和资源利用率：通过优化微服务应用的配置，提高系统的吞吐量、响应时间和稳定性，同时提升系统资源（如 CPU、内存、网络等）的利用效率，降低企业的运营成本。

优化微服务架构下配置优化的自动化过程：避免手动调整配置的繁琐过程，通过自动化优化框架提高配置优化的效率和精度，并减少人为错误。

## 3.2 课题研究内容

本课题的研究内容主要集中在云原生微服务架构下配置优化问题的理论与实践，涉及以下几个方面：

微服务架构配置优化的背景与挑战分析：

- 研究微服务架构在云原生环境中的配置需求，分析微服务的资源配置与软件参数对应用性能的影响。
- 识别微服务架构中的性能瓶颈，分析微服务服务之间复杂的依赖关系以及高维度配置空间带来的优化难题。
- 研究当前微服务配置优化领域的现有方案和技术，分析其优缺点以及适用场景。

基于贝叶斯优化算法的资源配置与软件参数协同优化方法研究：

- 采用贝叶斯优化算法，解决传统配置优化方法无法有效应对高维配置空间的难题。
- 研究如何将资源配置与软件参数的优化过程进行协同，避免单独优化带来的性能损失，提升微服务应用整体性能。
- 提出参数空间降维的有效策略，例如通过关键路径识别和服务抽取减少优化空间，提升优化效率。

云原生微服务应用在线配置优化框架设计与实现：

- 设计框架的整体架构，确定框架的功能需求和技术选型，结合容器编排技术（如 Kubernetes）和 DevOps 工具实现框架的高效运行。
- 实现在线配置优化功能，使微服务能够根据运行状态和负载自动调整资源配置和软件参数，优化性能。
- 提供用户自定义接口和算法接口，增强框架的可扩展性，满足不同企业 and 应用场景的需求。

实验设计与性能验证：

- 设计实验环境，模拟不同负载情况下微服务应用的性能表现。
- 对比单独调整资源配置、单独调整软件参数和协同优化三种方案的性能提升效果，验证协同优化方法的有效性。
- 通过性能指标（如延迟、吞吐量、资源利用率等）对优化效果进行定量

分析，评估框架的性能改进幅度。

- 框架的自动化配置和智能化调优：
- 研究如何进一步提升配置优化的自动化程度，减少人工干预的需求。
- 引入机器学习方法，结合历史数据和实时反馈，进一步提升框架的智能化调优能力。
- 设计一个自适应优化模块，能够根据负载变化和应用需求动态调整优化策略，优化微服务的配置流程。

### 3.3 拟解决的关键问题

本课题的关键问题主要集中在以下几个方面：

高维参数优化问题：微服务应用包含大量的配置参数，且这些参数的取值空间较大，如何有效地在庞大的高维参数空间中进行搜索，找到最佳配置，是配置优化中最具挑战性的问题。传统的优化方法无法高效应对这样的问题，因此本课题拟采用贝叶斯优化算法，结合降维策略，减少参数搜索空间，提高优化效率。

资源配置与软件参数的协同优化：微服务应用中的资源配置与软件参数之间存在复杂的依赖关系，单独优化其中一项配置，可能会影响到另一项配置，甚至可能导致系统性能下降。因此，如何实现资源配置与软件参数的协同优化，并在多个参数间找到最优的平衡点，是本课题的重要研究问题。

优化框架的自动化与智能化：微服务架构的配置优化需要在复杂的环境中自动进行，如何实现优化框架的自动化配置和智能化调优，减少人工干预，并使框架具有较好的适应性，是本课题的一大挑战。通过自动化框架，能够使企业在微服务的运行中实时、动态地调整配置，提升系统的可靠性和灵活性。

性能评价与实验验证：在配置优化过程中，如何量化不同优化方案的性能改进效果，并对优化方案进行科学的实验设计，是本课题需要解决的关键问题。通过合理的性能评价指标（如吞吐量、响应时间、资源利用率等）来衡量优化效果，确保优化方案能够在实际场景中取得显著的性能提升。

微服务架构下的资源竞争问题：微服务应用中，多个服务可能共享相同的资源（如 CPU、内存、网络带宽等），因此存在资源竞争的情况。如何在进行配置优化时，考虑资源竞争对性能的影响，避免过度分配资源或资源不足的问题，



是一个需要解决的关键问题。

框架的可扩展性与定制化能力：本课题的优化框架需要具备较强的可扩展性，以适应不同规模的微服务应用和多变的业务需求。如何设计一个灵活、易于扩展的框架，允许用户根据自身的需求定制化配置优化策略，是本课题需要解决的重要问题。

## 4. 课题的研究方法、设计及试验方案，可行性分析

### 4.1 课题的研究方法

本课题的研究方法主要包括以下几个方面：

理论与文献调研：

通过广泛调研国内外在云原生环境下微服务配置优化方面的研究成果，了解现有的配置优化方法、框架和技术，分析当前研究中的不足，明确本课题的创新点。

分析微服务架构下资源配置与软件参数优化的关键挑战，如服务间复杂的依赖关系、参数之间的非线性影响、高维度参数搜索空间等，为后续的优化算法设计和框架实现提供理论基础。

贝叶斯优化算法：

由于微服务应用中存在庞大的高维配置参数空间，传统的优化方法在搜索效率和精度上无法满足要求。贝叶斯优化算法作为一种有效的全局优化方法，能够通过较少的评估次数高效地寻找最优解。本课题将采用贝叶斯优化算法来解决配置优化中的高维度搜索问题，并结合协同优化策略同时优化资源配置和软件参数。

贝叶斯优化算法能够处理不确定性高的复杂系统，适用于本课题中高维复杂配置优化的需求。通过构建代理模型，贝叶斯优化能够在保证优化效率的同时减少计算资源消耗。

协同优化方法设计：

本课题将重点设计资源配置与软件参数之间的协同优化方法，考虑到微服务架构中的服务间复杂依赖关系，提出一种协同优化方案，以避免单独优化带来的性能损失。通过对微服务应用中的关键路径和关键服务进行识别，减少优化参数的搜索空间，从而提高优化过程的效率。

利用优化算法中的降维策略，聚焦于影响应用性能的核心服务和参数，降低搜索空间的维度，提高优化效率。

自动化框架设计：

基于云原生技术栈（如 Kubernetes、Docker、DevOps 工具等），设计并实现一个自动化的微服务配置优化框架。该框架能够自动化进行资源配置与软件参数的优化，并根据微服务的运行状态和负载动态调整配置，以提高系统的性能和资源利用率。

框架将包含自动化配置模块、优化算法模块、性能监控模块、服务调度模块等，能够通过云原生环境的集成，实现灵活的配置调整与优化。

实验验证与性能评估：

设计合理的实验方案，对比不同优化策略（单独调整资源配置、单独调整软件参数和协同优化）下的微服务性能表现，验证协同优化方法的有效性。

通过对比实验，分析各个方案在不同负载条件下的性能提升情况，重点考察吞吐量、响应时间、系统资源利用率等指标的变化，验证优化方案对微服务应用的实际影响。

## 4.2 课题设计方案

本课题的设计方案分为理论设计与实践实现两个层次，具体包括以下几个方面：

研究框架设计：

本课题将构建一个基于贝叶斯优化算法的资源配置与软件参数协同优化方案，设计优化框架的整体架构。优化框架将包括资源配置优化模块、软件参数优化模块、协同优化模块以及性能监控模块等。

框架将能够自动化收集微服务应用的运行状态、负载信息和资源利用情况，结合贝叶斯优化算法进行资源配置和软件参数的协同优化。

微服务配置优化算法的实现:

设计并实现基于贝叶斯优化的协同优化算法,结合微服务架构的特点,分析服务间的依赖关系,并通过关键服务的识别减少搜索空间。算法将能够在给定的时间内,基于历史数据和实时负载动态调整微服务应用的资源配置与软件参数,以实现性能优化。

自动化框架的实现:

基于云原生技术栈(如 Kubernetes、Docker、Ansible、Prometheus、Grafana 等)实现微服务应用在线配置优化框架。框架将能够与微服务部署环境集成,自动化进行容器资源配置和软件参数调整。

该框架将包括前端界面、优化算法接口、性能评估模块、自动化测试模块等,能够支持用户自定义优化目标和策略,适应不同的应用场景。

实验设计与评估方案:

设计多种实验场景,模拟微服务在不同负载下的运行情况,比较不同优化方案(资源配置优化、软件参数优化和协同优化)对系统性能的影响。

通过对微服务应用的吞吐量、延迟、资源利用率等多个指标进行综合评估,验证协同优化方案的有效性。

## 4.3 课题可行性分析

基于以上分析结果,我们认为本文的研究方法和设计方案切实可行,可操作性高,研究结果也具有很高的实用性,课题具有一定的研究价值。

## 5.课题计划进度和预期成果

### 5.1 计划进度

开始时间	结束时间	主要工作内容
2025 年 06 月	2026 年 03 月	前期知识储备、技术积累,方向调整

2026 年 03 月	2026 年 06 月	查阅方向相关论文，写开题报告
2026 年 07 月	2026 年 09 月	写文献综述，设计实验与算法
2026 年 09 月	2026 年 11 月	实验验证与优化，开始写论文初稿
2026 年 11 月	2027 年 1 月	论文修改、定稿

## 5.2 预期成果

论文一篇；  
结合公司环境与项目进行落地实践；  
相关代码开源到社区。