第三章 云原生微服务应用的资源配置与软件参数协同优化

（分析：概述微服务应用配置优化挑战及协同优化方案的必要性）

微服务架构非常适合用于设计面向用户的在线应用，在这种类型的应用中，对用户和应用提供方来说，性能都是非常重要的因素。通常一个完整的面向用户的在线微服务应用包括前端服务（如 Nginx）作为服务网关，提供流量管理及认证授权等功能；业务逻辑服务，根据用户请求数据提供逻辑处理和数据转换等功能；存储服务，通常各业务逻辑服务拥有独立的存储后端（如 Redis、MongoDB），提供用户数据存储以及加速缓存功能。典型的在线微服务应用架构如图 3.1 所示。

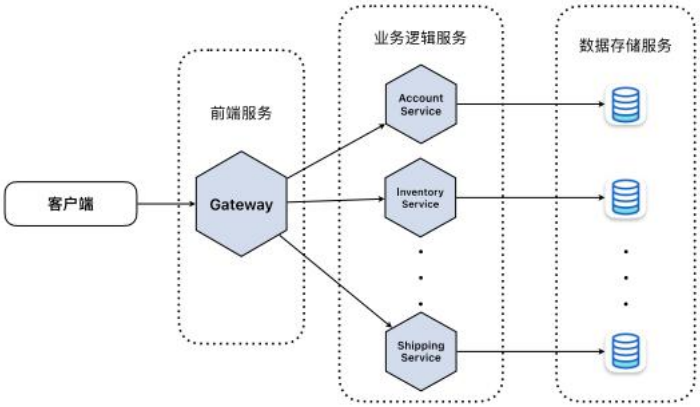

图 3.1 微服务应用组成

Figure 3.1 Microservice Application Composition

（分析：指出现有研究局限性，提出协同优化方案与贝叶斯优化方法的应用）

尽管微服务架构具有众多优点，但是也给应用持续稳定高效的运行带来了巨大挑战。配置参数会对应用性能产生重大影响，促使研究人员对应用的配置优化进行了大量研究。然而，现有的相关工作主要集中于传统软件的配置优化问题，很难直接迁移到微服务架构的应用。少量微服务场景下的相关工作主要关注系统资源的分配和管理，最近几篇工作[38-40]开始关注到微服务应用中软件参数的优化，目前尚未有相关研究将微服务应用的资源配置与软件参数协同考虑进行配置优化，这导致了应用性能优化不充分。本章提出了云原生微服务应用资源配置和软件参数的协同优化的方案，将贝叶斯优化应用到微服务应用配置优化问题上，并在本地集群上进行实验，验证了协同优化方案的有效性，相比单独调整资源配置或软件参数的方案能够进一步提升应用性能。

3.1 问题描述

（分析：明确优化目标为寻找最优参数配置以提升性能指标）

本章重点关注如何通过协同调整云原生环境中微服务应用资源配置和软件参数，进行微服务应用性能优化。微服务应用的配置参数调整可以看作一个优化问题。设表示在参数配置下运行微服务应用并对其进行负载测试后所得到的性能。其中，参数配置是由微服务应用中所有可调节参数组成的向量，该向量的维数非常高，假设的维数为，则(表示参数配置中第个参数的取值。注意这里所说的参数配置是指所有参数取值组成的向量。设为参数配置的所有可能取值的集合，* ∈ 表示微服务应用性能指标达到最优时的参数配置，即：

综上所述，云原生环境下微服务应用配置优化问题的目标就是找到最优参数配置* ，以优化微服务应用的性能。

在云原生环境下微服务应用的场景中，为了最大化应用性能，可以取性能指标的负数形式作为目标函数。例如，对延迟指标取反，则在达到最大性能指标时，微服务应用的延迟是最低的。在微服务场景下无法对配置优化目标函数的具体结构做出假设，此外，结合实际情况，找到最优参数配置需要在合理的时间内完成。需要特别说明的是，微服务应用中有许多指标（如每秒请求数量，延迟）可用于评估应用性能[56]，目标函数可以代表任何能够反应微服务应用性能的指标，本章选取工业和学术界常用的 99 端到端延迟[57-59]来评估微服务应用的性能。

3.2 研究挑战

（分析：总结微服务配置优化的三大核心挑战——依赖复杂、非线性关系、高维度搜索空间）

传统软件的配置优化工作已经是一项十分具有挑战的任务，在微服务架构下，通过资源配置与软件参数协同优化的方式，对微服务应用进行性能优化更加难以实施，因为它面临着如下挑战：

（1）参数之间依赖复杂。首先单个服务内部的参数存在相互依赖，一个参数往往不只对服务的一个方面产生影响。例如，在 MongoDB 中调整缓存参数，不仅会影响到缓存的大小，还会影响到数据库处理并行读事务的能力。其次微服务应用中不同的服务依赖彼此的输出完成自身功能逻辑，因此单个服务上参数的影响范围会扩展到与其存在依赖关系的其它服务。最后微服务应用中服务通常以容器形式部署，不同的容器可能部署在同一主机上,共享同一物理资源，例如内存、磁盘空间、CPU 等，因此存在资源竞争的现象，系统资源的设置存在干扰

关系。总的来说微服务应用的配置优化问题需要考虑参数之间存在复杂的依赖关系。

（2）配置参数与应用性能之间关系非线性。云原生环境下微服务应用性能与配置参数的取值之间并不是简单的线性关系。系统线程池是一个很好的例子，并不是线程池越大微服务应用性能越高：如果设置的值过低，会导致 CPU 资源利用率较低，线程池资源不足；而如果设置的值过高，则可能导致 CPU 资源的竞争。因此，线程池大小过低或者过高都会降低应用性能。配置参数与性能之间非线性的关系，使得预测微服务应用配置优化的结果非常困难，很难通过训练为云原生环境下微服务应用的配置优化问题建立一个准确的性能预测模型。

（3）高维度的参数搜索空间。云原生环境下的微服务应用规模庞大，大型微服务应用通常由数百至数千个相互作用的服务组成，每个服务都可以对其 CPU、内存等系统资源进行配置。此外，微服务应用部署的软件（如 Nginx、Redis、MongoDB 等）还额外增加了大量可调节的软件参数，并且这些参数的取值可能是连续的，也可能是离散的，因此可配置的参数共同组成了高维度的参数搜索空间。

为了解决上述挑战，需要正确、高效的方法来描述微服务应用的配置优化问题。

3.3 参数搜索空间降维策略

（提出关键服务识别策略）

本节将对上述云原生环境下微服务应用具有高维的参数搜索空间这一挑战进行深入讨论，并介绍了基于关键服务识别的有效解决方法。

在微服务应用中，假设存在个服务，每个服务有(个可供调节的参数，因此，参数配置向量的维度。如果平均每个参数有个可能的取值，那么所有可能参数配置的数量|，这表明随着服务数量的增加，参数的搜索空间呈指数级增长。为了缩小参数搜索范围需要采取适当的策略，以使配置优化工作集中在微服务应用中某一服务子集上。关键服务识别方法就是一种有效的解决方案，它可以确定哪些服务是影响微服务应用性能的关键，并将配置优化工作的重点放在这些服务上，以有效地控制参数搜索空间的大小。