哈爾濱工業大學

本科毕业论文（ 设计 ） 中期报告

题　目：面向公平性的云边微服务系统部署方法

专　　业　软件工程

学　　生　付书煜

学　　号　2021111824

指导教师　贺祥

日　　期　2025 年 3 月 日

哈尔滨工业大学教务处制

目 录

1 论文（设计）工作是否按开题报告预定的内容及进度安排进行 1
2 已完成的研究工作及成果 2
2.1 云-边协同架构的系统设计与建模 2
2.1.1 变量定义 2
2.1.2 约束条件 4
2.1.3 优化目标 5
2.2 算法实现 5
2.2.1 算法实现思路 5
2.2.2 算法具体流程 5
2.2.3 对比算法 6
2.3 实验平台开发 6
2.3.1 集群网络模块 6
2.3.2 数据源管理模块 6
2.3.3 演化模块 7
2.4 仿真实验 7
2.4.1 实验设计 7
2.4.2 实验过程 7
2.4.3 仿真结果与分析 8
3 后期拟完成的研究工作及进度安排 12
4 存在的问题与困难 12
5 论文按时完成的可能性 13

1 论文（设计）工作是否按开题报告预定的内容及进度安排进行

随着互联网的迅速发展，微服务架构因其高效、灵活的特性被广泛应用于视频流媒体、在线游戏等对实时性和灵活性要求较高的场景中。微服务通过将单体应用拆分为独立、细粒度的模块，使各模块能够独立部署和扩展，并通过轻量级通信协议进行协作，从而大幅提升了系统的灵活性和扩展性。然而，这种架构也带来了网络延迟、系统管理复杂、资源利用效率低等问题，尤其是在高并发访问和复杂任务处理时，如何高效分配和管理资源成为亟待解决的关键问题。

为应对这些挑战，云-边协同计算模式应运而生。该模式通过在云端和边缘节点之间合理分配任务，最大化各自的资源优势。云端资源丰富，适合处理大规模数据和复杂任务，但由于其物理距离远，网络传输延迟较高；边缘节点则位于用户附近，能够就近处理请求，显著降低延迟从而提升用户体验。然而，边缘节点资源有限，易在高并发或复杂任务处理中出现性能瓶颈。因此，在云-边协同环境中，不仅要合理分配任务和资源，还需根据计算需求和网络条件，灵活调整服务实例的部署，以更好地发挥云端和边缘节点的资源优势，从而提升整体 QoE（服务质量体验）。

现有的云-边协同部署方案多集中于最大化整体 QoE，但在优化整体用户体验时，未充分考虑不同用户之间的体验差异的公平性问题。由于地理位置、网络条件、节点资源负载的差异，不同用户的响应时间波动较大，尤其是远离边缘节点的用户，可能因网络延迟较高，体验显著低于接近边缘节点的用户。对于视频流媒体等对实时性和一致性要求较高的应用场景，这种 QoE 不均衡会严重影响用户体验，特别是在资源有限的情况下，公平性问题变得更加突出。

公平性问题可以从两个方面来考虑。首先，不同优先级的用户在服务响应上存在显著差异。例如，在视频流媒体平台上，购买 VIP 会员的高优先级用户期望能够以更高的画质、更快的加载速度观看视频，边缘节点可以优先处理这些高优先级用户的请求，从而减少加载时间和缓冲现象；而普通用户则仅仅关注视频的流畅播放，不希望出现长时间的加载或低画质的问题。高优先级用户的期望和实际体验之间的差距，可能会影响他们对平台的满意度。其次，即使在同一优先级的用户之间，网络条件和设备性能差异也可能导致体验上的不公平。例如，尽管同为 VIP 会员的用户，他们的网络条件不同，可能会造成体验上的巨大差异。网络较差的高优先级用户可能在观看视频时出现缓冲或延迟现象，而设备较旧的普通用户可能会面临视频卡顿或加载缓慢的问题，这些因素都可能导致同一优先级用户之间的体验差异。对于视频流媒体等对实时性和一致性要求极高的应用场景，这种 QoE 不均衡会在资源紧张时加剧，严重影响用户的体验。

因此，优化服务实例部署和资源分配，合理控制资源受限条件下不同优先级用户之间的响应时间差异，同时均衡同一优先级用户的 QoE，已成为云-边协同计算中的关键挑战。在资源充足时，公平性问题较少，但在资源受限的环境下，如何结合云端和边缘资源，利用智能化手段优化部署策略，提升系统公平性和增强用户体验一致性，成为本研究的核心目标。我的工作进度安排如表 1-1 所示：

表 1-1 工作进度安排表

工作安排	周数	起止时间
系统建模与设计阶段	3	2024.11.20-2024.12.15
算法设计与实现	5	2024.12.16-2025.01.19
仿真实验与数据收集	4	2025.01.20-2025.02.16
敏感性分析与模型调优	3	2025.02.17-2025.03.09
成本效益分析与优化验证	4	2025.03.10-2025.04.06
结题，论文编写	2	2025.04.07-2025.04.20

目前为止，基本按照开题报告中的规划进行，已完成了系统建模、算法设计以及部分仿真实验。

2 已完成的研究工作及成果

如开题报告中的场景描述：作为一家提供智能网络服务的公司，我们负责为一个大规模的用户群体设计和优化其服务部署。用户分布在不同的地理位置，既有处于城市中心的用户，也有偏远地区的用户，他们通过不同的终端设备连接到我们的服务系统。为了确保服务质量，特别是在响应时间上的公平性，我们引入了云-边协同架构，在多个位置部署边缘节点，并将计算资源丰富的云服务器与边缘节点紧密结合。

在这个场景中，用户的优先级（例如高付费用户和普通用户）会影响资源的分配，优先级较高的用户会获得更多的计算资源，从而保证更低的响应时间。响应时间不仅受到服务器资源的影响，还与数据的传输延迟和处理能力密切相关。此外，边缘节点和云服务器的部署成本和资源负载也需要考虑在内，以避免过度依赖边缘节点导致的超载现象。

我们的目标是在边缘服务器资源有限的前提下，通过优化资源分配和服务实例的部署，使得不同优先级的用户都能够在合理的响应时间内获得服务，尽可能提高系统的公平性。同时，我们还需要确保云边协同架构在部署实例时能够平衡成本和资源消耗，避免不必要的资源浪费或过度负载，从而实现系统的高效运行和经济可行性。

为了实现这一目标，我的研究工作从以下几个方面展开：云-边协同架构的系统设计与建模、算法设计、实验相关平台开发及仿真实验。接下来，我将详细介绍我已完成的研究工作及所取得的成果。

2.1 云-边协同架构的系统设计与建模

在上述背景下，我构建了以下数学模型，该模型综合考虑了用户、服务器以及它们之间的交互关系，通过一系列变量定义、目标函数设定和约束条件限制，实现在云-边协同架构中对服务部署、资源分配以及负载均衡等关键问题进行求解，旨在优化系统性能并提高系统公平性。

2.1.1 变量定义

2.1.1.1 用户相关变量

（1）**用户集 U** 定义 $U = \{u_1, u_2, \cdots, u_n\}$ 代表所有用户，不同用户处于不同地理位置，

位置用二维坐标 (x_i, y_i) 表示。

（2）**用户优先级 Q_i**　每个用户 u_i 被赋予一个优先级 Q_i，其根据用户重要性（如付费用户、普通用户等）确定。在进行资源分配时，优先级高的用户将优先获得更多资源，进而影响其响应时间和服务质量。

（3）**用户权重 W_i**　优先级为 Q_i 的用户对应的权重为 W_i，Q_i 越大越大，W_i 越大。W_i 用于计算用户响应时间的加权 Jain 公平性指数，对用户响应时间进行加权。在公平性评估和优化中起着关键作用，其影响着不同优先级用户响应时间在整体公平性指标中的贡献程度。

（4）**请求数据大小 D_i**　用户 u_i 发往服务器（云或边缘）的数据量 D_i，该变量是计算资源需求、传输延迟和服务器处理时间等关键指标的重要依据。

2.1.1.2 服务器相关变量

（1）**服务集 S**　定义 $S = \{s_1, s_2, \cdots, s_m\}$ 表示所有可供用户连接的服务器集合，包括边缘服务器集 S_{edge} 和云服务器集 S_{cloud}，位置用二维坐标 (x_i, y_i) 表示。每个服务器 s_j 上可能部署了一个或多个服务实例；如果没有用户连接到某个服务器，该服务器可以不部署任何服务实例。

（2）**服务器资源集合 R_j^{server}**　定义 $R_j^{server} = \{r_j^{cpu\_total}, r_j^{mem\_total}, r_j^{b\_total}\}$ 表示服务器 j 的 CPU、内存以及带宽总量。

（3）**服务器处理能力 P_j^x**　定义 P_j^x 表示服务器 s_j 的总处理能力，边缘服务器为 P_j^e，云服务器为 P_j^c。通常情况下，$P_j^e < P_j^c$，即云服务器的处理能力较强。

2.1.1.3 连接与资源、能力分配变量

（1）**连接变量 x_{ij}**　用二进制变量 x_{ij} 表示用户 u_i 是否连接到服务器 s_j。x_{ij} 为 1 表示用户 u_i 连接到服务器 s_j，为 0 则表示没有连接。

（2）**计算资源分配集合 R_i^{user}**　表示用户 u_i 连接到服务器 s_j 时，s_j 需要分配给用户的资源集合，$R_i^{user} = \{r_i^{cpu}, r_i^{mem}, r_i^b\}$。其中 $r_i^{cpu} = f_{cpu}(D_i) \cdot W_i$，$r_i^{mem} = f_{mem}(D_i) \cdot W_i$，$r_i^b = f_b(D_i) \cdot W_i$，分别表示用户所需的 CPU、内存以及带宽资源量。其中，$f_x(D_i)$ 为用户数据请求量 D_i 与 CPU、内存以及带宽等资源分配量的转换函数；W_i 为用户权重，使得高优先级用户能获取更多资源，体现了模型对不同优先级用户资源分配的差异化处理。

（3）**处理能力分配变量 P_{ij}**　定义 $P_{ij} = \left(\frac{R_i^{user}}{\sum_{u_k \epsilon U} R_k^{user}}\right) \cdot P_j$，$x_{kj} = 1$，表示服务器 s_j 分配给用户 u_i 的处理能力。其中，$\frac{R_i^{user}}{\sum_{u_k \epsilon U} R_k^{user}}$ 表示用户 u_i 在服务器 s_j 上所获得的资源量占连接到该服务器的所有用户分配的资源总和的比例，通过分配比例，每个用户获得的计算能力与其分配到的资源量成正比。

2.1.1.4 时间相关变量

（1）**传输时延 $\mathbf{t_{trans_{ij}}}$**　由物理传输时延 $t_{d_{ij}} = \frac{d_{ij}}{v}$ 和带宽延迟 $t_{b_{ij}} = \frac{D_i}{b}$ 两部分组成。其中，$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ 表示用户与服务器之间的物理距离；v 为信号传播速度；b 为服务器节点可用带宽。

（2）**处理时延 $\mathbf{t_{proc_{ij}}}$**　定义处理时延为 $t_{proc_{ij}} = \frac{D_i}{P_{ij}}$，即用户数据请求量与用户分配到的服务器处理能力的比值。

（3）**响应时间 $\mathbf{t_{ij}}$**　表示连接到服务器 s_j 的用户 u_i 获得服务响应的总时延，由传输时延 $t_{trans_{ij}}$ 和处理时延 $t_{proc_{ij}}$ 两部分组成，即 $t_{ij} = t_{trans_{ij}} + t_{proc_{ij}}$。

（4）**加权响应时间 $\mathbf{t_{ij}^{weight}}$**　定义 $t_{ij}^{weight} = t_{ij} \cdot W_i$。在公平性评估和优化过程中，加权响应时间通过权重 W_i 体现不同优先级用户响应时间的重要程度差异。

（5）**最大平均响应时间 $\mathbf{T_{Q_i}^{max}}$**　表示优先级为 Q_i 的用户的最大平均响应时间。

2.1.1.5 成本相关变量

（1）**实例部署成本 $\mathbf{c_j}$**　表示在服务器 s_j 部署实例的成本，因边缘节点和云节点的差异性，二者成本计算方式不同。

（2）**边缘节点部署成本 $\mathbf{c_j^e}$**　定义 $c_j^e = c_{fixed_j}^e + c_{usage_j}^e$，其中 $c_{fixed_j}^e$ 为租赁边缘服务器的固定成本；$c_{usage_j}^e = \sum_{i=1}^{n} x_{ij} \cdot (r_i^{cpu} \cdot p_{cpu}^e + r_i^{mem} \cdot p_{mem}^e + r_i^b \cdot p_b^e)$ 为边缘服务器在实际运行中产生的资源消耗成本，p_{cpu}^e、p_{mem}^e、p_b^e 分别表示边缘服务器上单位 CPU、内存以及带宽的价格。

（3）**云节点部署成本 $\mathbf{c_j^c}$**　定义 $c_j^c = c_{usage_j}^c + c_{net_j}^c$，其中 $c_{usage_j}^c = \sum_{i=1}^{n} x_{ij} \cdot (r_i^{cpu} \cdot p_{cpu}^c + r_i^{mem} \cdot p_{mem}^c + r_i^b \cdot p_b^c)$ 为云服务器在际运行中产生的资源消耗成本，与边缘服务器的计算方式相似；$c_{net_j}^c = \sum_{i=1}^{n} x_{ij} \cdot D_i \cdot p_{net}^c$ 为云服务的网络流量成本，表示用户从不同区域访问云，产生的额外网络传输费用，根据流量数据量计算。p_{net}^c 为云平台的流量单价。

（4）**综合部署成本 $\mathbf{C_{total}}$**　定义 $C_{total} = \sum_{s_j \in S_{edge}} c_j^e + \sum_{s_j \in S_{cloud}} c_j^c$，即所有边缘节点和云节点成本的总和 。

（5）**部署成本上限 $\mathbf{C_{max}}$**　为服务提供的部署的最大成本预算。

2.1.2 约束条件

（1）**平均响应时间约束**　确保每个优先级类别用户的平均响应时间不会超过设定的上限，如公式（2-1）所示：

$$\frac{1}{|U_{Q_i}|} \sum_{u_j \in U_{Q_i}} \sum_{s_k \in S} x_{jk} \cdot t_{jk} \le T_{Q_i}^{max}, \ \forall Q_i \qquad (2\text{-}1)$$

（2）**边缘节点资源限制**　每个边缘节点 s_j 的资源消耗不得超过其最大可用资源，约束公式如下：

$$\sum_{i=1}^{n} x_{ij} \cdot r_i^{cpu} \le R_j^{cpu\_total}, \forall s_j \in S_{edge} \qquad (2\text{-}2)$$

$$\sum_{i=1}^{n} x_{ij} \cdot r_i^{mem} \le R_j^{mem\_total}, \forall s_j \in S_{edge} \qquad (2\text{-}3)$$

$$\sum_{i=1}^{n} x_{ij} \cdot r_i^{b} \le R_j^{b\_totsl}, \forall s_j \in S_{edge} \qquad (2\text{-}4)$$

（3）**部署成本限制**　边缘节点与云节点的部署成本总和不得超过整体预算。

$$C_{total} = \sum_{s_j \in S_{edge}} c_j^e + \sum_{s_j \in S_{cloud}} c_j^c \leq C_{max} \quad (2\text{-}5)$$

（4）**用户与服务器连接限制**　每个用户 u_i 必须连接到唯一一个服务器。

$$\sum_{j=1}^{m} x_{ij} = 1, \forall u_i \in U \quad (2\text{-}6)$$

2.1.3 优化目标

（1）**公平性目标函数**　用 f 表示，即 $f = \max\left(F_{Jain}\right)$，其中 F_{Jain} 为加权 Jain 公平性指数，其定义为公式（2-7）：

$$F_{Jain} = \frac{\left(\sum_{i=1}^{n} t_{ij}^{weight}\right)^2}{n \cdot \sum_{i=1}^{n} \left(t_{ij}^{weight}\right)^2} \quad (2\text{-}7)$$

公式中，分子为所有用户加权响应时间总和的平方，反映了加权时间的集中程度，若各用户的加权响应时间较为相近，总和会较大，平方后的值也更大，这表明用户间响应时间差异较小；分母是所有用户加权响应时间的平方和，体现了系统中所有用户响应时间的散布程度，若各用户响应时间相差较大，这个和将会较大，意味着公平性较差。加权响应时间能起到调节公平性的作用，若高权重用户的响应时间更短，其加权响应时间会更接近低权重用户的加权响应时间，使分子总和更集中，差异减小；同时会减少分母中高权重用户平方项的贡献，让整体分母变小。

整个目标函数通过最大化 F_{Jain} 来优化公平性，当 趋近于 1 时，代表所有用户的加权响应时间几乎完全相同，系统达到公平状态；当 F_{Jain} 趋近于 0 时，说明加权响应时间差异非常大，系统不公平。该目标函数不仅可用于公平性评估，还能通过引入加权响应时间，根据用户优先级调节不同用户之间的响应时间差异，保证不同优先级用户的响应时间公平。

2.2 算法实现

2.2.1 算法实现思路

在本研究中，我们主要采用了贪心算法来解决云-边协同架构中的资源调度问题。该算法基于用户的优先级，逐一为每个用户分配资源，以最大化系统的公平性。核心思路是每次选择一个局部最优解（即为当前用户选择能使 Jain 公平性指数最大化的服务器），以此逐步优化系统的整体性能。贪心算法通过优化加权 Jain 公平性指数来实现公平性目标，同时在每一步中确保资源分配满足服务器资源的约束。

2.2.2 算法具体流程

贪心算法的具体流程如下：

（1）**初始化**　首先，初始化用户到服务器的分配矩阵，并设置每个服务器的资源使用情况。

（2）**用户优先级排序**　根据用户的优先级（例如，高付费用户优先）将所有用户进行排序。优先级较高的用户将在分配资源时得到优先考虑。

（3）**分配服务器资源**　根据每个用户的需求，从可用的服务器中选择一个最适合的服务器进行资源分配。该选择过程旨在最大化加权 Jain 公平性指数，同时满足服务器的资源限制。

（4）**更新资源使用情况**　每分配一次资源，更新相应服务器的资源使用情况，确保不会超过服务器的最大资源限制。

（5）**检查约束条件**　在每一步分配之后，检查当前资源分配是否满足所有约束条件，包括服务器负载、部署成本以及响应时间上限等。如果所有条件满足，则继续分配下一用户的资源。

（6）**退出条件**　如果成功为所有用户分配服务器并满足所有约束，返回最终的分配矩阵；若在多次尝试后仍未找到满足约束条件的结果，则退出并给出警告。

2.2.3 对比算法

为了验证贪心算法的有效性，本研究将其与几种其他优化算法进行了对比：

（1）**优化求解器（Gurobi）**　该算法使用数学优化工具对资源分配进行全局优化，目标是最大化系统的公平性并满足约束条件。优化求解器能够找到理论上的最优解，但计算开销较大，尤其在大规模问题中。

（2）**遗传算法（GA）**　遗传算法是一种基于自然选择和遗传学原理的启发式优化算法。通过模拟自然选择过程，GA 能够在大规模问题中找到近似最优解，但相比于贪心算法，计算时间较长。

（3）**无公平性算法**　此算法以最小化平均响应时间为优化目标，不考虑用户之间的公平性。它可以在某些情况下降低用户响应时间，但由于忽视了优先级和资源分配的差异，将无法确保系统公平性，在本实验中主要是用来对照。

2.3 实验平台开发

为了支持云-边协同计算系统的研究，特别是在优化公平性和资源分配方面，我参与了 MicroForge 平台前端部分的开发，主要负责集群网络、数据源管理和演化模块三个功能模块的实现。

2.3.1 集群网络模块

（1）**功能描述**　该模块用于模拟不同的网络环境，允许用户设置不同服务器之间的网络延迟、带宽和其他网络参数。通过灵活配置，研究人员可以在平台上复现各种现实环境中的网络特性。该模块不仅支持配置网络拓扑，还能通过实时显示带宽和延迟情况，帮助用户直观了解网络条件对资源分配和响应时间的影响。

（2）**与研究结合**　集群网络模块为本研究提供了对不同网络条件下系统性能的模拟环境。通过调整网络延迟和带宽，可以分析不同网络环境对用户响应时间、资源分配和公平性的影响，为优化算法提供数据支持。

2.3.2 数据源管理模块

（1）**功能描述**　此模块提供了一个数据源管理界面，展示各个数据源的信息，包括

数据类型和资源使用情况（如 CPU、内存、带宽）。用户可以实时查询和监控各数据源的状态，确保系统能够及时响应各种数据请求。

（2）**与研究结合**　数据源管理模块帮助监控云-边协同系统中的各类资源使用情况，为模型的资源分配和性能评估提供关键的数据支持。研究人员可以实时获取各服务器的资源状态，优化资源分配策略，并对高优先级用户进行更精确的资源分配。

2.3.3 演化模块

（1）**功能描述**　该模块包含分析算法和规划算法两个部分，支持用户向平台注册自己的算法，并调用这些算法以获取运行结果。分析算法主要用于对现有网络环境和资源分配策略进行评估，而规划算法则用于优化资源分配和调度，以提高系统的公平性和效率。

（2）**与研究结合**　演化模块与本研究的算法部分直接相关。通过在平台上注册和调用算法，研究人员能够快速测试和验证不同的优化策略，探索如何在保证公平性的同时提高系统性能。特别是分析算法可以用于评估现有资源分配策略的公平性，而规划算法则为云-边协同架构中的资源调度和负载均衡提供了优化方向。

通过这些模块，MicroForge 平台为本研究提供了一个灵活且强大的实验环境，使得不同网络条件和资源使用情况下的算法优化与测试变得更加高效和直观。

2.4 仿真实验

2.4.1 实验设计

为了验证所提出的云-边协同系统在不同网络环境下的性能表现以及资源分配算法的有效性，本研究设计了一系列仿真实验。实验的主要目的是评估不同算法在处理不同用户数量和优先级时的平均响应时间和公平性表现。

实验设置如下：

（1）**用户数量**　选择 100、150 和 200 个用户，来模拟不同负载下的用户场景。

（2）**用户优先级设置**　为每个用户分配一个优先级，本实验中包括优先级 1、2、3 三个类别，以测试在优先级差异下的公平性。

（3）**算法选择**　实验中使用四种算法进行对比，包括贪心算法、Gurobi 优化求解器、遗传算法（GA）以及无公平性的算法。

（4）**实验平台**　本实验应用到了 MicroForge 平台，平台的集群网络、数据源管理模块和演化模块为实验提供了必要的实验环境。

2.4.2 实验过程

（1）**初始化阶段**　为每个实验配置用户数量、优先级以及相应的服务请求。并初始化每个算法的输入参数。

（2）**算法执行**　根据所选算法，执行资源调度与服务实例部署过程。每个算法根据用户的优先级、请求大小和网络延迟等条件进行资源分配。

（3）**数据采集与分析**　收集每个用户的响应时间、系统资源利用率等数据，并进行统计分析。重点关注不同优先级用户的平均响应时间差异，以及不同算法在公平性优化方面的表现。

（4）**实验验证**　通过仿真实验验证模型和算法的正确性，并对不同算法的性能进行对比分析，特别是在平均响应时间和公平性方面的差异。

2.4.3 仿真结果与分析

2.4.3.1 仿真结果展示

在本部分中，我将以 200 个用户为例，展示各算法的运行结果。

（1）**用户与服务器初始分布**　图 2-1 展示了用户与边缘服务器和云服务器的初始分布情况。

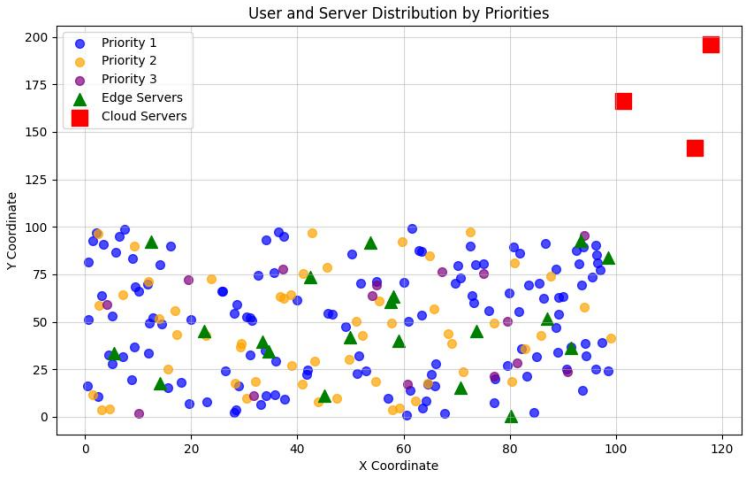
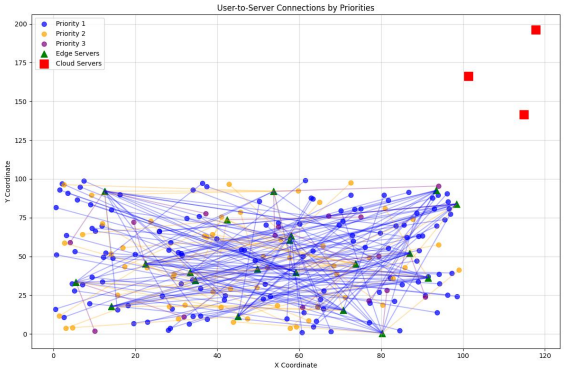
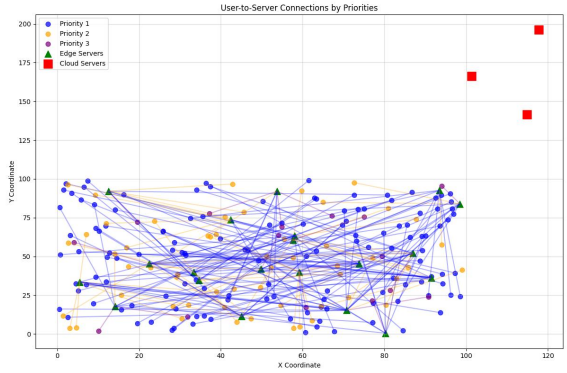


图 2-1 用户与服务器的初始分布

（2）**用户与服务器连接情况**　图 2-2 (a)、2-2 (b)、2-2 (c) 、2-2 (d) 分别展示了贪心算法、Gurobi 优化求解器、遗传算法（GA）以及无公平性的算法运行后的用户与服务器连接情况。

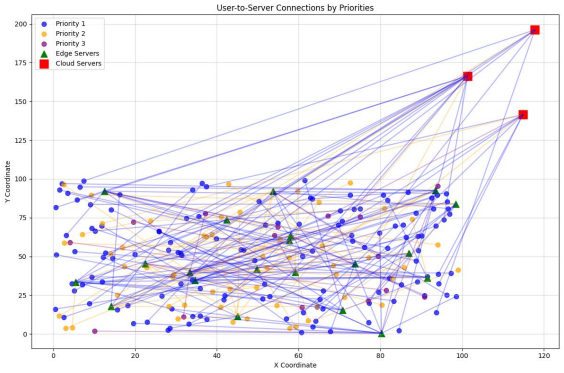


(a) 贪心算法

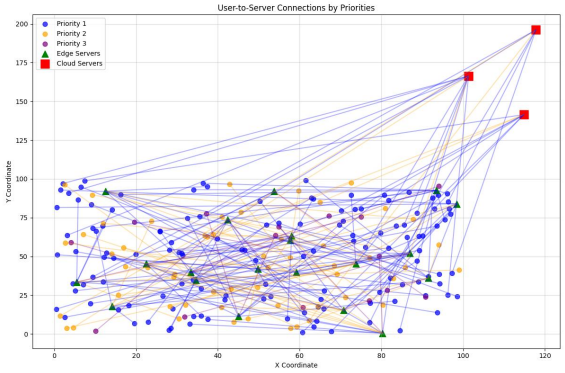


(b) Gurobi 优化求解器

图 2-2 用户与服务器连接情况



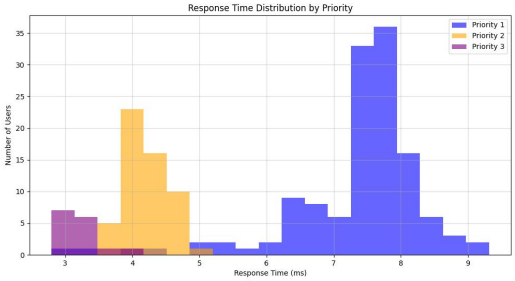
(c) GA 算法



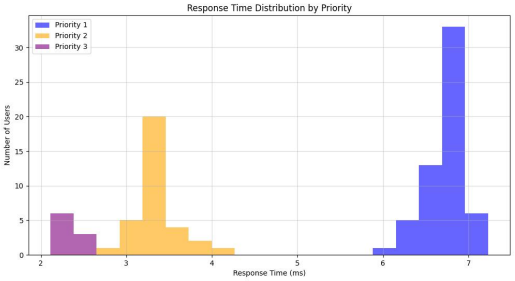
(d) 无公平性算法

图 2-2 用户与服务器连接情况（续）

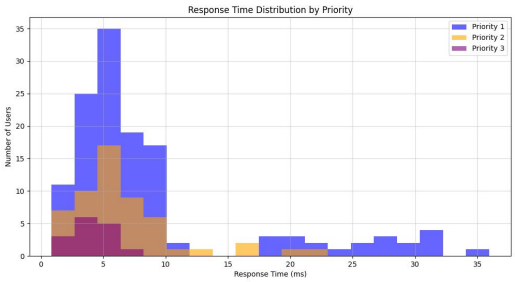
（3）**各优先级用户的响应时间分布** 图 2-3 (a)、2-3 (b)、2-3 (c) 、2-3 (d) 分别展示了贪心算法、Gurobi 优化求解器、遗传算法（GA）以及无公平性的算法运行后的各个优先级用户的响应时间分布情况。



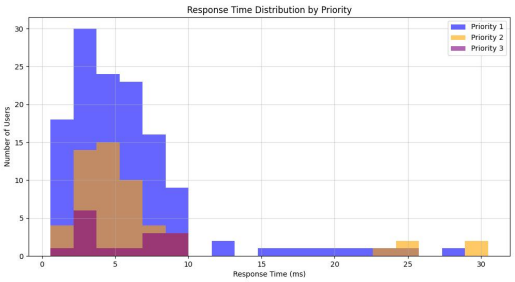
(a) 贪心算法



(b) Gurobi 优化求解器



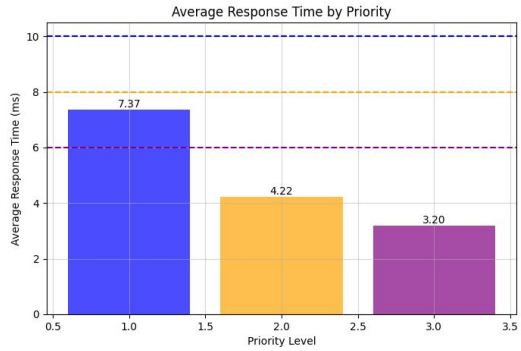
(c) GA 算法



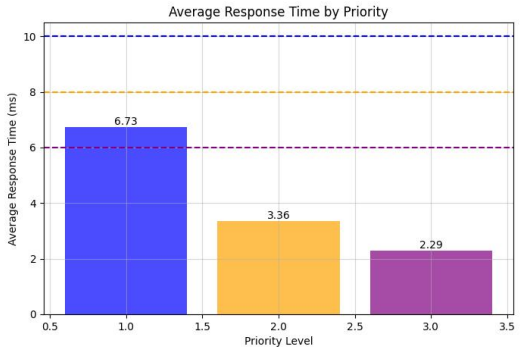
(d) 无公平性算法

图 2-3 各优先级用户的响应时间分布情况

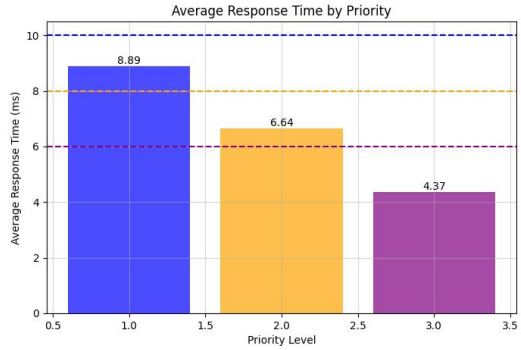
（4）**各个优先级用户的平均响应时间** 图 2-4 (a)、2-4 (b)、2-4 (c) 、2-4 (d) 分别展示了贪心算法、Gurobi 优化求解器、遗传算法（GA）以及无公平性的算法运行后的各个优先级用户的平均响应时间。



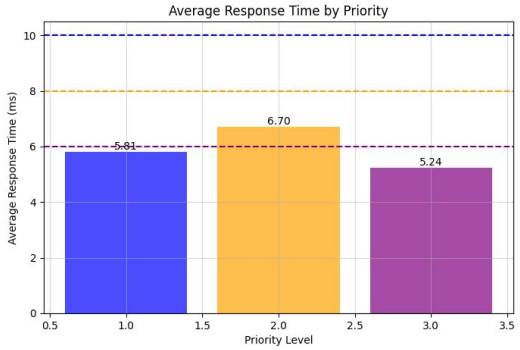
(a) 贪心算法



(b) Gurobi 优化求解器



(c) GA 算法



(d) 无公平性算法

图 2-4 各优先级用户的平均响应时间

2.4.3.2 结果分析

在本部分中，我将展示不同算法在不同用户数量和优先级设置下的性能表现，具体从不同优先级用户平均响应时间和系统公平性表现等方面进行分析。

（1）**不同用户数量下各算法的平均响应时间** 图 2-5 (a)、2-5 (b)、2-5 (c) 分别展示了用户数量为 100、150 和 200 时，各算法不同优先级用户的平均响应时间。

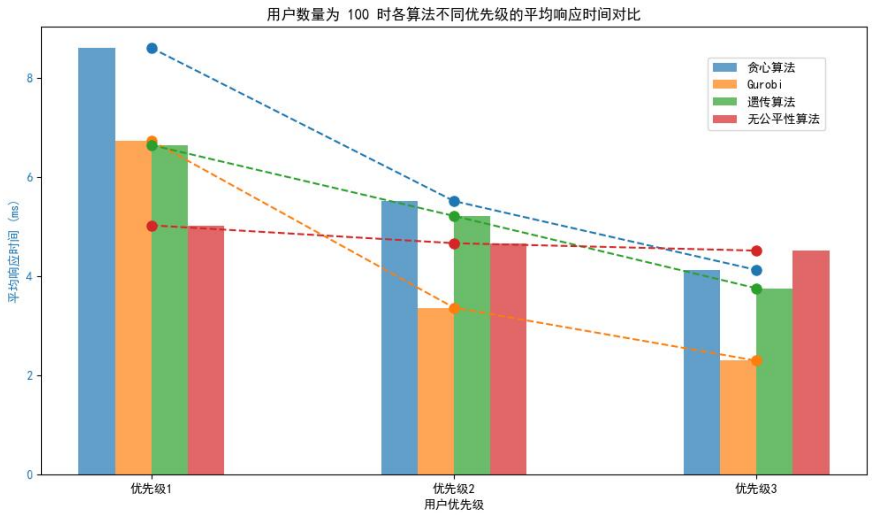


图 2-5 (a) 用户数量为 100 时各算法平均响应时间对比

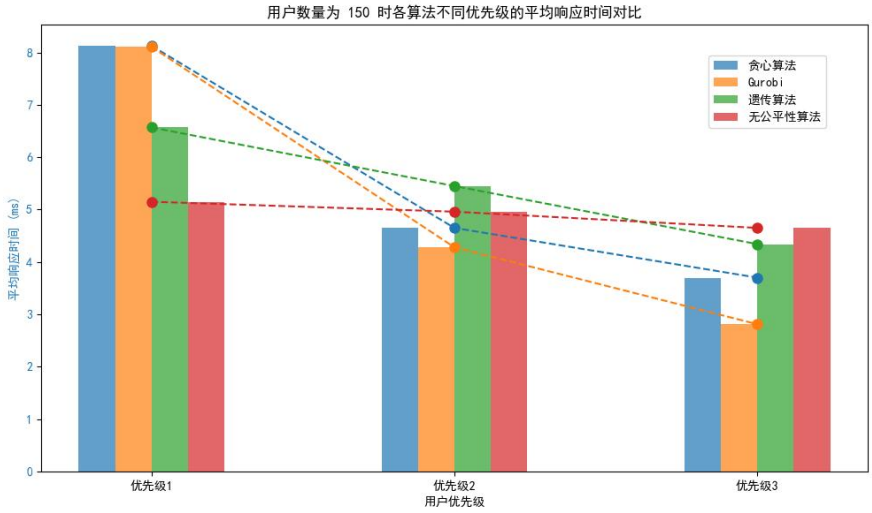


图 2-5 (b) 用户数量为 150 时各算法平均响应时间对比

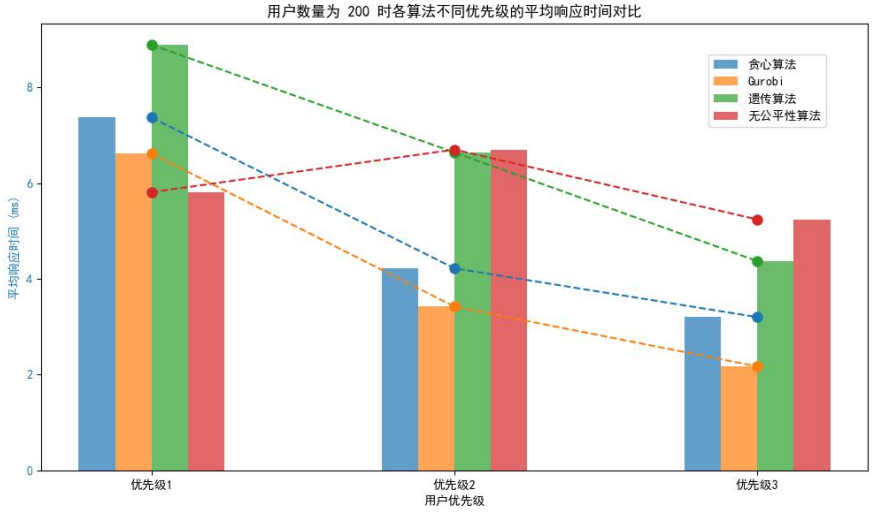


图 2-5 (c) 用户数量为 200 时各算法平均响应时间对比

我从以下两个方面对仿真结果进行分析：

首先，是在同一算法下不同优先级用户的平均响应时间情况。理论上，在资源有限的情况下优先级越高的用户应优先获得资源，因此他们的响应时间应该更短。实验结果表明，贪心算法和 Gurobi 优化算法能够有效根据优先级分配资源，确保高优先级用户的响应时间明显较短，而低优先级用户的响应时间则较长。相比之下，无公平性算法没有考虑优先级差异，导致所有用户的响应时间趋于一致，无法有效区分高低优先级用户，影响高优先级用户的体验。

其次，四种算法的对比结果显示，贪心算法在资源分配方面表现较为均衡，尤其在大规模用户场景下，能够稳定保持低响应时间。Gurobi 优化求解器虽然提供最优解，但计算复杂度较高。遗传算法在小规模场景表现良好，但在用户数量增多时响应时间波动较大，且计算时间较长。无公平性算法忽视了优先级差异，尽管减少了平均响应时间，但无法保证公平性。总的来说，贪心算法在平衡公平性和响应时间方面表现最佳，尤其适合大规模

用户场景。

（2）**不同用户数量下各算法的加权 Jain 指数** 图 2-6 展示了不同用户数量下，各算法的加权 Jain 指数，该指数用于衡量系统的公平性，值越大表示系统的公平性越好。

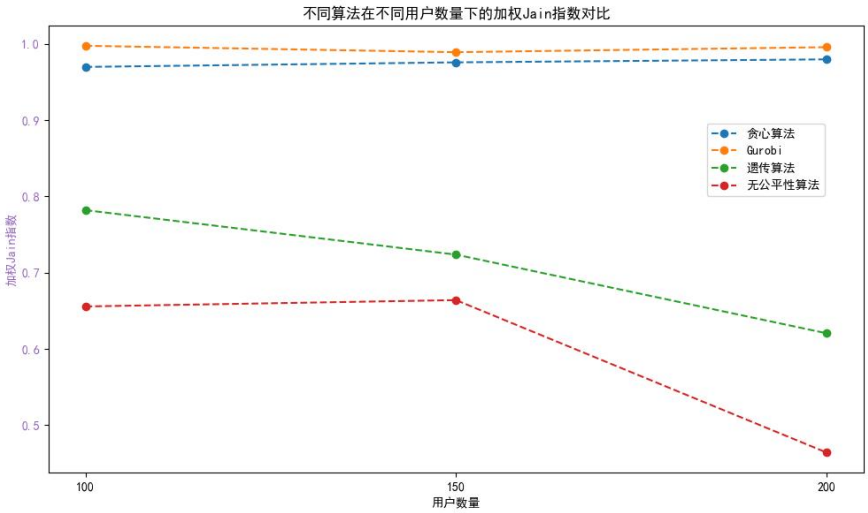


图 2-6 不同用户数量下各算法的加权 Jain 指数

从实验结果可以看出，随着用户数量的增加，所有算法的加权 Jain 指数均有所下降，表明系统的公平性在用户数量增多时有所降低。然而，贪心算法和 Gurobi 优化求解器在所有用户数量下的加权 Jain 指数始终较高，显示出较好的公平性，尤其在大规模用户场景中，二者的公平性明显优于其他算法，说明它们能提供更优化的资源分配方案。尽管如此，由于贪心算法的实现简单且计算开销较低，它在实际应用中表现更为优越，特别是在资源有限的情况下。相比之下，无公平性算法的加权 Jain 指数最低，表明该算法未能考虑用户优先级和公平性，导致不同用户之间的响应时间差异较大。总体来看，贪心算法在平衡实现简单性和系统公平性方面表现突出，特别适合应用于大规模用户场景。

3 后期拟完成的研究工作及进度安排

后期工作将按以下进度安排完成：

（1）**模型调优与验证**（2025.03.10-2025.03.31） 进一步对现有模型进行验证，评估模型在不同环境下的表现，确保其具备实际应用的可行性。

（2）**算法优化**（2025.04.01-2025.04.30） 优化算法性能，提升算法的计算效率和稳定性，确保大规模问题处理的可行性。

（3）**论文撰写与总结**（2025.04.01-2025.04.30） 总结研究成果，并完成论文的编写与完善。

4 存在的问题与困难

（1）**实验数据的收集与分析困难** 由于实验需要在实际的云边协同环境中运行，数据收集面临一定的困难。尤其是在多种网络环境和设备条件下进行实验时，数据的准确性和代表性可能会受到不同网络延迟、带宽波动以及设备性能差异的影响。尽管通过仿真实

验模拟了部分场景，但实际环境中的动态变化仍然是一个需要克服的问题。此外，由于平台的复杂性和多样性，实验数据的整合和分析也存在一定的难度。

（2）**公平性评估的主观性** 在公平性优化过程中，不同用户的需求差异较大，如何平衡这些差异并避免过于主观的判断，仍是一个挑战。

5 论文按时完成的可能性

根据目前的进度安排，研究工作和论文编写的各个环节都在按计划推进。系统建模、算法设计以及部分仿真实验已经完成，且模型调优和成本效益分析也已规划到位。在接下来的两个月内，重点将放在完成算法优化和模型验证上，以确保系统的实际应用可行性和稳定性。所有的实验和算法优化工作预计将在 3 月底完成，论文撰写阶段将从 4 月开始并顺利进入总结和完善阶段。

尽管存在一些实验数据收集和公平性评估的挑战，但通过已有的研究成果和数据支持，项目按时完成的可能性较高。在接下来的时间内，确保顺利完成剩余的工作，应该能够按计划完成毕业论文的撰写和提交。