面向性能与安全感知的雾计算分布式服务部署

Mohammad Goudarzi、Arash Shaghaghi、Zhiyu Wang 和 Rajkumar Buyya

摘要——物联网（IoT）应用的迅速普及，加剧了对雾计算中高效、安全服务部署的需求。然而，异构资源、动态工作负载以及多样的安全要求，使得实现最优服务部署极具挑战性。现有的大多数解决方案主要关注性能指标，却忽视了部署决策的安全影响。为解决这些问题，本文提出一种安全与性能感知分布式深度强化学习（SPA-DDRL）框架，用于联合优化雾计算中的服务响应时间和安全合规性。该问题被建模为加权多目标优化任务，旨在最小化时延，同时最大化根据雾节点安全能力得出的安全评分。安全评分采用一种新的三级层次结构：配置级检查用于验证设置是否正确，能力级评估用于评估资源安全特性，控制级评估则用于执行严格的策略，从而确保解符合安全要求，同时满足性能目标。SPA-DDRL采用分布式代理—学习器架构，多个代理自主进行服务部署决策，集中式学习器则通过共享优先经验协调全局策略优化。SPA-DDRL融合了三项关键创新：利用长短期记忆网络捕捉动态环境中的时间依赖关系；利用优先经验回放，通过突出关键经验加速收敛；以及采用离策略校正机制稳定分布式训练。基于真实物联网工作负载开展的大量实验表明，与最先进方法相比，SPA-DDRL显著改善了服务响应时间和部署安全性，响应时间提升了 16.3%，收敛速度加快了 33%。此外，SPA-DDRL能够在所有系统规模下始终保持一致、可行且符合安全要求的解，而基线技术则会失效或出现性能下降。

索引词— 雾/边缘计算、物联网（IoT）、深度强化学习（DRL）、安全性、性能 .

arXiv:2601.01125v1 [cs.DC] 3 Jan 2026

1 引言

物联网（IoT）设备已无处不在，广泛应用于智慧医疗、智慧城市和智能交通等多个领域 [1]。这些应用旨在通过分析各类数据流提供高效解决方案。因此，为确保运行完整性，这些计算密集型服务对计算和通信资源提出了很高要求 [2]。然而，资源受限型物联网设备无法在严格的时延限制内处理海量数据流。因此，它们会将任务卸载到替代基础设施，这种范式称为服务部署 [3]。

云计算为物联网部署提供弹性资源 [4]，，但其固有的时延和带宽限制使其不适用于时间敏感型应用。雾计算通过在网络边缘部署雾服务器（FSs）来弥补这一不足，形成一种分层架构，从而有效管理计算密集型和时延关键型工作负载 [5], [6]。

然而，在异构雾计算环境中，服务部署面临双重挑战。首先，与云服务器相比，文件系统的容量有限，导致资源争用，因此需要通过策略性部署，在高负载下平衡性能 [7]。其次，雾计算的开放式架构使系统面临严重的安全漏洞（例如，拒绝服务（DoS）攻击），而高性能节点往往成为攻击的主要目标 [8], [9],[10]。由于传统策略通常优先考虑性能而非安全性，使关键基础设施暴露于风险之中 [11]，因此亟需能够同时优化性能效率与安全合规性的方法。

鉴于雾环境固有的随机性，深度强化学习（DRL）成为一种稳健的自适应决策范式，无需预先掌握系统知识即可推导出最优策略 [12]。然而，标准的 DRL 实现面临显著的可扩展性和效率障碍。在异构、高维环境中，集中式方法往往导致难以承受的收敛时间和高昂的探索开销。此外，现有的分布式深度强化学习技术通常存在数据利用率低的问题，无法有效利用分散在各个执行体中的经验轨迹来实现全局最优。

为克服这些局限，我们提出安全与性能感知分布式深度强化学习（SPA-DDRL）框架，以联合优化安全合规性和服务延迟。该框架采用分布式代理-学习器架构，由自主代理执行去中心化的放置决策，同时由集中式学习器统筹全局策略更新。该框架结合了三种核心机制：用于捕捉时间依赖关系的长短期记忆网络（LSTM）、

•*Mohammad Goudarzi* 任职于澳大利亚莫纳什大学信息技术学院（电子邮箱：*moham- mad.goudarzi@monash.edu*）。

• *Arash Shaghaghi* 任职于澳大利亚新南威尔士大学计算机科学与工程学院（电子邮箱：*a.shaghaghi@unsw.edu.au*）。

• *Zhiyu Wang* 和 *Rajkumar Buyya* 任职于澳大利亚墨尔本大学计算与信息系统学院量子云计算与分布式系统（*qCLOUDS*）实验室（电子邮箱：*zhiywang1@student.unimelb.edu.au*、*rbuyya@unimelb.edu.au*）。

优先经验回放（PER），以加快从关键转移中学习；以及离策略校正，以减轻分布式执行者之间的策略偏离。这种设计有效地兼顾了异构雾环境中的决策可扩展性与学习稳定性。

- 我们提出了一种新颖的服务部署方法，并将其表述为加权优化问题。该方法旨在实现两个关键目标：最小化服务的响应时间，并最大化部署的整体安全评分。安全评分通过严格评估可用雾资源的安全能力来确定。

- 我们提出了 SPA-DDRL，这是一种面向动态随机雾环境的分布式深度强化学习技术。SPA-DDRL 基于演员-评论家架构，能够高效利用从多个分布式代理收集的经验数据，训练出更优的服务部署模型。此外，我们为 SPA-DDRL 设计了奖励函数，以联合优化表示为有向无环图（DAG）的服务的响应时间和服务部署安全评分。

- 为提升学习性能，我们将 LSTM 网络集成到演员-评论家架构中，以捕捉动态雾环境中的时间依赖关系；采用优先经验回放（PER），依据时序差分（TD）误差将学习重点放在关键经验上；并实施离策略校正机制（例如重要性采样和梯度裁剪），以解决分布式代理与学习器之间的策略偏离问题。

- 为全面评估 SPA-DDRL，我们使用一组多样化的合成有向无环图开展了大量实验。这些有向无环图源自真实世界的物联网服务，并将其性能与相关技术进行了比较。

本文其余部分组织如下。第 2 节回顾相关文献。第 3 节介绍系统模型和三层安全层级，而第 4 节将服务放置问题形式化为马尔可夫决策过程（MDP）。第 5节详细介绍所提出的 SPA-DDRL 框架，第6 节则对现有技术进行全面的实验评估。第 7节总结本文，并探讨未来研究方向。

2 相关工作

多项研究（例如， [10], [11], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25]）深入探讨了雾计算中物联网服务放置的优化，主要关注响应时间和能量效率等性能指标。然而，这些研究大多忽略了边缘与雾计算环境中服务部署的一个关键方面：安全性。本节探讨优先兼顾物联网服务性能与安全性的服务部署策略，以关注这一重要但经常被忽视的维度。

现有关于安全感知服务放置的文献大致可分为传统优化方法和基于学习的方法。关于传统优化，Sun等人 [11]提出了一种用于物联网任务调度的启发式算法，该算法根据概率风险模型对任务进行排序，以满足安全要求和严格截止期限。Mann等人 [13] 将服务放置和安全控制选择建模为约束满足问题，并利用Gurobi优化硬件和软件级安全配置。类似地，Elgendy等人 [14] 应用分支定界法优化集成AES加密的计算卸载，同时最小化时延和能耗。Casola等人 [15] 开发了一种用于工业物联网放置的贪心策略，在强制执行特定安全策略约束的同时平衡成本和性能。Javanmardi等人 [16] 引入了一种基于SDN的元启发式调度器，通过动态监控设备安全状态并隔离可疑节点来缓解DDoS风险。在其先前工作的基础上，Mann等人 [17] 使用精确求解器，在模块级和用户级位置安全约束下处理联合应用放置与用户分配问题。Wang等人 [10] 利用元启发式算法，在高性能节点面临更高攻击概率的前提下，同时优化分布式数据放置的响应时间和安全风险。Singh等人 [18] 设计了一种基于标签的启发式算法，根据安全分类将服务映射到资源（例如可信私有云），同时遵守执行截止时间。

转向基于学习的动态方法后，Zhang等人 [19]针对多云边缘网络开发了动作约束深度Q网络（DQN），将AES和RSA加密约束直接整合到计算卸载过程中。Rahmani等人 [20]提出了一种混合框架，将异步优势演员评论家（A3C）与解析层次过程（AHP）相结合，用于区块链赋能的移动边缘计算，在时延、能量和安全交易验证之间实现平衡。Ebrahim等人 [21]采用双重深度Q网络（DDQN）框架解决隐私保护负载均衡问题，在不暴露敏感节点信息的情况下管理工作负载分配，并应对部分可观测性。Sun等人 [22] 将资源分配建模为马尔可夫决策过程，利用动作约束深度Q网络（DQN）根据任务敏感性动态调整保护级别（例如RSA、MD5）。Mohammadi等人 [23]通过引入西格玛缩放改进NSGA-II算法，以解决设备到设备雾计算系统中的多目标资源分配问题，同时优化延迟、能量和泄露成本。Du等人 [24] 为海洋物联网设计了一种改进的遗传算法（GA），引入修正算子和安全量化模型，以确保卸载方案的可行性与安全性。最后，Thangaraj等人 [25] 提出了一种用于区块链赋能的雾计算的混合遗传算法-粒子群优化机制，根据移动模式选择授权服务器，以确保数据完整性并最小化时延。

表 1 全面比较了我们的工作与现有文献在若干关键方面的差异，包括服务属性（如结构、数量、依赖关系、异构性）、环境属性（如设备数量、层级结构、资源分布和异构性）、问题建模（数学建模、参数、约束）、决策引擎属性（放置层、求解器、复杂度、适应性、可扩展性）、安全属性（安全目标、缓解方法、机制）以及所使用的评估指标。

最复杂的雾环境涉及多种物联网设备、具有不同服务需求的服务（通常包含相互依赖的服务）——

表1：相关工作与本文工作的定性比较

属性	服务属性				环境属性				问题形式化属性			决策引擎属性					安全属性		评估属性
	结构	数量	约束	异构性	用户/物联网	层级结构	边数	异构性	公式	参数	约束	放置层	求解器	CMP	ADAP	SCAL	主要目标	缓解	评估指标
[11]	服务	多个	依赖	✓	多个	三层	多个	✓	混合整数非线性规划	Risk 概率	截止期限	Edge	启发式	Low	Low	Mid	满足安全要求 不同应用的。	基于……进行任务排序 安全要求和 将其放置在最佳计算节点上	响应时间， 风险等级， 安全级别
[13]	服务	多个	依赖	×	单个	Two 层	多个	✓	MIQP	性能	CPU Req	NA	优化 求解器	High	Low	Low	应用任务放置与 安全控制器配置。	对应用和服务器的安全控制项进行分类 作为 CSP 问题，并寻找满意的解决方案。	响应时间
[14]	Task	多个	独立	✓	多个	Two 层	单个	×	ILP	时间， 能量	时间， 部署	Edge	分支定界	High	Low	Low	保护敏感的放置 位置。	用于保护的 AES 加密技术 敏感位置。	响应时间， 能量
[15]	服务	多个	依赖	NA	单个	三层	多个	✓	INLP	成本，安全性	安全性	Edge	贪心	低-中	Low	Mid	满足安全要求 的任务。	为每个任务定义安全控制项，并 根据所提供的安全级别满足这些要求。	Cost
[16]	NA	多个	独立	NA	多个	三层	多个	✓	NA	负载均衡， 延迟	×	Edge	元启发式 (NSGA-III)	Mid	Low	Mid	资源管理框架以 考虑物联网设备的安全状态 并建模可疑对象。	将入侵检测系统方法的结果作为输入 输入模糊函数，以隔离物联网设备 发起分布式拒绝服务和扫描攻击。	响应时间， 网络使用
[17]	服务	多个	独立	✓	多个	三层	单个	×	QCMIP	延迟	安全性， 容量	NA	优化 求解器	High	Low	Low	应用放置和用户 分配，同时考虑不同的 安全性和隐私约束	考虑模块级位置、用户级位置、 共址和 k-匿名性约束。	响应时间
[10]	服务 （数据）	多个	独立	✓	多个	三层	多个	✓	NA	时间， 安全性	×	NA	元启发式 （进化算法 ）	Mid	Mid	Mid	尽量缩短 文件系统上分布式文件的读写时间，同时 尽量降低遭受攻击的风险。	在响应时间与安全性之间权衡， 定义基于 文件系统中心性的安全风险评分。	响应时间， 成功 攻击
[18]	服务	多个	独立	✓	多个	三层	多个	×	NA	系统 运行成本， 系统 利用率	截止期限	NA	启发式	Low	Low	Low	雾计算网络中的实时任务调度 同时考虑任务的安全性	为每项服务分配标签，以表示其安全 强度，并为每种资源分配标签，以表示其 信任级别	成功 比率， 吞吐量
[19]	Task	多个	独立	✓	多个	三层	多个	✓	MDP	时间， 能量	截止期限	Edge	DRL （深度Q 网络）	Low	Mid	Low	多云安全资源分配 边缘计算最小化 加权延迟和能量消耗。	采用MD5的AES-RSA混合加密 数据传输完整性验证 MEC和CC层面中的安全性	响应时间， 能量
[20]	Task	多个	独立	✓	多个	三层	多个	✓	MDP	时间， 能量	截止期限	Edge	DDRL (A3C)	Low	Mid	Mid	多用户任务部署优化 在区块链赋能的 MEC-IoT 网络中 平衡能量、时延和安全性	基于区块链的安全交易 记录并生成不可篡改的审计轨迹 以确保数据完整性和信任	响应时间， 能量
[21]	Task	多个	独立	✓	多个	三层	多个	✓	MDP	Time	隐私	Edge	DRL （双重深度 Q网络）	Low	Mid	Low	雾计算中的隐私感知负载均衡 网络中最小化等待延迟 不泄露雾计算节点信息	避免信息隐藏方法 雾计算负载和资源 信息以维护提供商隐私	响应时间
[22]	Task	多个	独立	✓	多个	三层	多个	×	MDP	时间， 能量	安全性	Edge	DRL （深度Q网络）	Low	Mid	Low	优化安全任务部署 决策，同时保护数据 传输过程中的隐私	RSA 非对称加密与 用于数据完整性的 MD5 哈希 以及隐私保护	响应时间， 能量
[23]	Task	多个	独立	✓	多个	三层	多个	×	混合整数非线性规划	时间， 能量	安全性	Edge	元启发式 (NSGA-II)	Mid	Low	Mid	考虑安全性的资源分配 安全漏洞成本	采用自适应加密的多目标优化 基于安全级别的算法选择	响应时间， 能量
[24]	Task	多个	独立	×	多个	三层	多个	✓	混合整数非线性规划	安全性， 能量	资源	Edge	元启发式 (GA)	Mid	Low	Low	海洋物联网中的安全计算卸载 同时防止数据拦截和 保护客户端隐私	采用对称加密算法 安全密钥管理和密码学 用于数据传输安全的配置	安全性
[25]	Task	多个	独立	✓	多个	三层	多个	✓	MIP	时间， 能量	安全性	Edge	元启发式 （混合离散粒子 群优化）	Mid	Low	Low	安全计算卸载于 区块链赋能的环境以 防止恶意节点攻击	采用 PoW 共识的区块链技术 维护去中心化数据的机制 完整性并防止单点故障	响应时间， 能量
我们的工作	服务	多个	依赖	✓	多个	三层	多个	✓	MDP	时间， 安全性	安全性， 截止期限	Edge	DDRL	Low	High	High	联合优化响应时间 与服务放置的安全性 在异构雾计算 环境中	结合硬约束满足的 安全性优化 通过基于惩罚的奖励机制最大化安全性评分 基于惩罚的奖励机制	响应时间， 安全性

QCMIP：二次约束混合整数规划，ILP：整数线性规划，MINLP：混合整数非线性规划，MIQP：混合整数二次规划，MDP：马尔可夫决策过程，MIP：混合整数规划，NA：不适用，Heter：异构性，PSO：粒子群优化，DDRL：分布式深度强化学习，B&B：分支定界，CMP：复杂度，ADAP：适应性，SCAL：可扩展性

、异构雾服务器、多层异构资源的雾计算环境，以及高度动态性。在这些复杂且随机的环境中，为多种物联网应用部署服务以优化响应时间和安全评分，已成为一项至关重要但颇具挑战的问题。尽管一些已有研究（例如 [13], [14], [16], [17], [18], [19], [20],[21], [22], [23], [25]）在部署过程中为雾服务器定义了安全评分，但如何在实践中定义此类评分仍不明确。此外，传统的服务放置求解方法（如启发式算法和元启发式算法，例如 [10], [11], [13], [14], [15], [16], [17], [18], [23], [24], [25]）由于时间复杂度高、适应性有限且可扩展性不足，难以在这些环境中找到高效的解决方案 [26]。此外，用于服务放置问题的深度强化学习技术常因探索成本高、收敛缓慢而面临实际部署挑战，尤其是在特征数量、环境复杂度和放置问题约束不断增加时[26], [27]。为克服这些局限，SPA-DDRL 通过实用的三层分层模型对安全性进行量化，在控制、能力和配置层面刻画安全性，同时纳入全面的响应时间建模，从而消除现有安全评分方法中的模糊性，并实现精确的双目标优化。与传统的集中式方法相比，分布式代理-学习器架构提升了可扩展性和适应性；而长短期记忆网络与优先经验回放（PER）的结合，相较于传统深度强化学习方法，显著缩短了收敛时间并降低了探索成本。此外，先进的离策略校正机制（包括重要性采样和梯度裁剪）确保了即使在分布式环境中出现策略偏离，学习过程仍保持稳定，使该框架尤其适用于大规模动态雾计算部署。

3 系统模型与问题表述

我们考虑一个包含三个资源层级的雾计算系统，如图1所示：云服务器、文件系统和物联网设备。云服务器提供强大的计算能力，但时延较高。部署在网络边缘的文件系统可提供

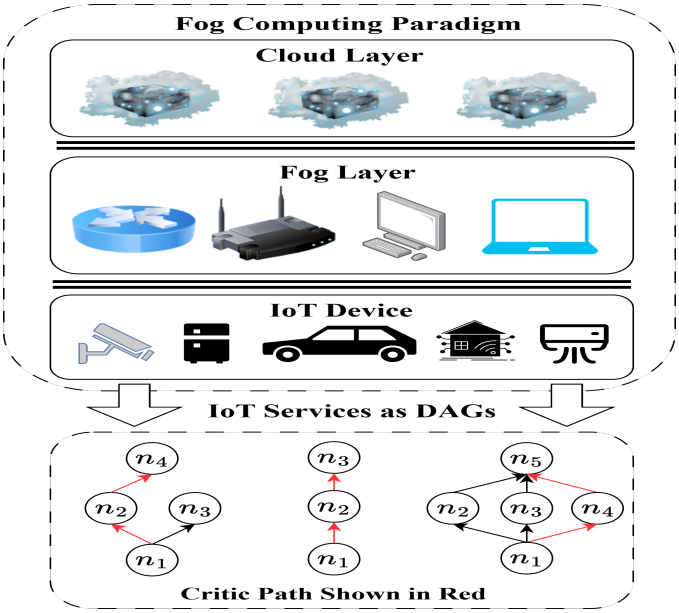


图1：三层雾计算架构，以及一个示例服务有向无环图，其中关键路径以红色高亮显示

中等资源，且时延较低。物联网设备会产生服务请求，但本地处理能力有限。所有参数均汇总于附录中的表1。

3.1 服务建模

服务建模为一个有向无环图 $\Gamma = (\mathcal{N}, \mathcal{A})$，其中 $\mathcal{N}$表示任务集合，$\mathcal{A}$ 表示任务之间的有向边集合。该服务包含 K 个任务，即 $|\mathcal{N}| = K$，每个任务的索引为 n_h，其中 $h \in \{1, 2, ..., K\}$。

每个任务 n_h 由一个五元组$\langle c_h,\ m_h,\ st_h,\ s_h,\ d_h \rangle$ 表征，其中 c_h 表示所需的 CPU 周期，m_h 指定内存需求，st_h 表示存储需求，s_h 表示安全要求，d_h 表示截止时间约束。

有向边 $a_{k,\ h} \in \mathcal{A}$ 从任务 n_k 指向任务 n_h，表示 n_h 依赖于 n_k的输出。每条边都关联一个权重 $w_{k,\ h}$ 用于量化从 n_k 传输到 n_h的数据量。对于任务 n_h，我们将 $\Pi(n_h) = \{n_k \mid a_{k,h} \in \mathcal{A}\}$ 定义为其直接前驱任务的集合。

3.2 问题建模

令 $\mathcal{R}$ 表示可用服务器的集合，其基数为$|\mathcal{R}| = R$。每台服务器 $r^{p,q} \in \mathcal{R}$ 均由类型 p（例如 CS、FS、物联网）和该类型内的索引 q 标识。部署方案 Φ 表示将每个任务分配到一台服务器：

$$\Phi = \{\phi_{n_h} \mid n_h \in \mathcal{N}, \phi_{n_h} \in \mathcal{R}\} \tag{1}$$

其中 $\phi_{n_h} = r^{p,q}$ 表示任务 n_h 被分配至服务器 $r^{p,q}$。每台服务器由其处理能力 $\rho_{r^{p,q}}$、内存容量 $M_{r^{p,q}}$、存储容量 $ST_{r^{p,q}}$、在二维空间中的物理位置坐标 $(x_{r^{p,q}}, y_{r^{p,q}})$ 以及一组已启用的安全配置项 $\mathcal{CNF}_{r^{p,q}}$来表征；这些配置项决定其安全能力的合规性。

3.2.1 响应时间模型

分配至服务器 ϕ_{n_h}的任务 n_h 的完成时间由两部分组成：计算时间和数据等待时间。形式化地，我们将其表示为：

$$T_{\phi_{n_h}} = T^{\text{comp}}_{\phi_{n_h}} + T^{\text{wait}}_{\phi_{n_h}} \tag{2}$$

计算时间由任务的 CPU需求 c_h 和服务器的处理能力 $\rho_{\phi_{n_h}}$决定：

$$T^{\text{comp}}_{\phi_{n_h}} = \frac{c_h}{\rho_{\phi_{n_h}}} \tag{3}$$

数据等待时间表示所有前置输入在分配的服务器上可用之前所经历的时长。对于分配到不同服务器的任务，数据传输同时涉及带宽受限传输和距离相关的传播时延。其公式如下：

$$T^{\text{wait}}_{\phi_{n_h}} = \max_{n_k \in \Pi(n_h)} \left[\left(\frac{w_{k,h}}{\omega(\phi_{n_k}, \phi_{n_h})} + \lambda(\phi_{n_k}, \phi_{n_h}) \right) \cdot \delta_{\phi_{n_k}, \phi_{n_h}} \right] \tag{4}$$

其中，$\omega(\phi_{n_k}, \phi_{n_h})$ 表示两个服务器之间的可用带宽，$\lambda(\phi_{n_k}, \phi_{n_h})$ 表示传播时延。二元指示变量 $\delta_{\phi_{n_k}\ \phi_{n_h}}$ 1 $\phi_{n_k} \neq$ ，在满 ϕ_{n_h} 0 足条件时取值为，否则取值为，从而消除共址任务的网络开销。传播时延根据服务器坐标之间的欧几里得距离和信号传输速度 ν计算：

$$\lambda(\phi_{n_k}, \phi_{n_h}) = \frac{1}{\nu} \sqrt{(x_{\phi_{n_k}} - x_{\phi_{n_h}})^2 + (y_{\phi_{n_k}} - y_{\phi_{n_h}})^2} \tag{5}$$

由于有向无环图结构所固有的并行执行能力，整体服务延迟由关键路径 $\mathcal{B}$决定，该路径表示产生最大累计执行时间的依赖任务序列。我们采用向上排序算法来识别关键路径成员关系，该算法根据每个任务的计算成本及其下游依赖关系对任务进行评估。关键路径指示函数 β_{n_h} 定义为：

$$\beta_{n_h} = \begin{cases} 1, & n_h \in \mathcal{B} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

因此，部署 Φ下的总服务响应时间计算如下：

$$\mathcal{L}(\Phi) = \sum_{h=1}^{K} \beta_{n_h} \cdot T_{\phi_{n_h}} \tag{7}$$

3.2.2 安全模型

雾计算环境开放且分布式的特性使其面临各种安全威胁，包括数据泄露、恶意攻击和未授权访问。不同的物联网应用具有不同的安全要求，范围从简单的数据完整性验证到复杂的端到端加密和访问控制机制。与此同时，异构的计算资源提供商拥有不同的安全能力和保护机制。因此，我们需要一种结构化的安全模型，以精确描述任务安全要求、量化资源提供商的安全能力，并评估二者之间的匹配程度。受我们此前与 Cisco Systems Australia 合作申请专利的解决方案启发 [28], 我们提出了一种三层分层安全模型，提供从抽象安全控制到具体配置实现的细粒度安全要求描述和能力评估框架。该模型不仅能够捕捉现实世界安全要求的复杂度，还能为优化算法提供可量化的安全指标。

分层安全架构：我们假设采用分层雾架构，并由分布式代理做出放置决策。学习器、代理及其通信信道均可信，而 FSs/CSs 不可信，且安全态势存在异构性。通信采用标准协议，但若未强制实施控制项，通信信道可能不安全。每个任务（n_h）都是一项主要资产，具有安全要求（s_h），必须通过适当的服务器放置来满足。我们将各个服务器建模为具有异构安全能力且本质上不可信，因此需要进行明确检查以确保合规性。安全框架会验证服务器固有的配置项（例如已启用的加密算法）是否符合任务需求。

我们的安全模型采用三层渐进式结构：控制项（C）、能力项（CP）和配置项（CI）。安全控制层展示高层安全目标，例如授权和未授权设备清单以及数据加密保护 [29]。每个安全控制项都对应一个特定的安全领域，反映系统需要防范的特定安全风险。对于具有安全要求 s_h 的任务 n_h，可将其表示为一组安全控制项：

$$s_h = \{C_k \mid k \in \mathcal{K}_h\} \tag{8}$$

其中，$\mathcal{K}_h$ 是任务 n_h所需的安全控制项索引集合。安全能力层将每个安全控制项细化为多个具体的安全能力。例如，数据加密保护控制项可能包含传输加密、存储加密和密钥管理等能力。每项能力代表实施相应安全控制项的一个具体技术维度。不同能力在实施安全控制项时的重要性各异，因此我们为每项能力分配权重，同一控制项下所有能力的权重之和为 1。每个控制项 C_k 由多个能力组成：

$$C_k = \{CP_l \mid l \in \mathcal{L}_k\} \tag{9}$$

其中， $\mathcal{L}_k$ 是控制项 C_k的能力索引集合。配置项层是最具体的实施层级，其中每项安全能力由多个配置项组成。配置项表示是否启用了特定安全机制，例如 AES-256 加密算法、双因素认证和审计日志。配置项值为 1 表示已部署该安全机制且运行正常，为 0 则表示未部署或不可用。每项能力 CP_l 由一组配置项组成：

$$CP_l = \{CI_i \mid i \in \mathcal{I}_l\} \tag{10}$$

其中， $\mathcal{I}_l$ 是能力项 CP_l的配置项索引集合，且每个 $CI_i \in \{0,1\}$ 都是二元指示变量。

安全评分计算： 我们设计了一种安全评分算法，用于量化任务需求与资源能力之间的匹配程度。

对于部署分配 ϕ_{n_h}下任务 n_h 的能力项 CP_l ，我们首先计算配置项满足率。设 $\mathcal{CNF}_{CP_l}$ 表示能力项 CP_l 所需的配置项集合（即 $\mathcal{CNF}_{CP_l} = \{CI_i \mid i \in \mathcal{I}_l\}$）， $\mathcal{CNF}_{\phi_{n_h}}$ 表示分配的服务器所支持的配置项集合。能力项 CP_l 的配置项满足率通过 SR_l计算：

$$SR_l(\phi_{n_h}) = \frac{|\mathcal{CNF}_{CP_l} \cap \mathcal{CNF}_{\phi_{n_h}}|}{|\mathcal{CNF}_{CP_l}|} \times 100 \tag{11}$$

接下来，我们引入一个能力级函数，将配置项满足率转换为安全评分。该能力级函数将满足率映射到离散的安全级别，从而创建一种强调完全合规的非线性变换：

$$CapLevel(SR_l(\phi_{n_h})) = \begin{cases} \sigma_{min}, & \text{if } SR_l(\phi_{n_h}) = \tau_{min} \\ f(\sigma,\tau,SR_l(\phi_{n_h})), & \text{if } \tau_{min} < SR_l(\phi_{n_h}) < \tau_{max} \\ \sigma_{max}, & \text{if } SR_l(\phi_{n_h}) = \tau_{max} \end{cases} \tag{12}$$

其中， σ_{min} 和 σ_{max} 分别表示最低和最高安全评分， τ_{min} 和 τ_{max} 为边界满足率，而 $f(\sigma,\tau,SR_l(\phi_{n_h}))$ 是一个离散化函数，它根据预定义阈值将中间满足率映射到相应的安全级别。

对于任务 n_h所需的控制项 C_k ，我们使用 $\mathcal{G}_k$ 对其包含的所有能力评分进行加权聚合，从而计算控制级安全评分，如下所示：

$$\mathcal{G}_k(\phi_{n_h}) = \frac{\sum_{l\in\mathcal{L}_k} w_l^{cp} \times CapLevel(SR_l(\phi_{n_h}))}{\sum_{l\in\mathcal{L}_k} w_l^{cp}} \tag{13}$$

其中， w_l^{cp} 是控制项 C_k中能力 CP_l 的权重。

此外，将控制项评分映射为离散安全级别的控制级函数计算如下：

$$CtrlLevel(\mathcal{G}_k(\phi_{n_h})) = \begin{cases} \kappa_{min}, & \text{if } \mathcal{G}_k(\phi_{n_h}) = \xi_{min} \\ \kappa_{partial}, & \text{if } \xi_{min} < \mathcal{G}_k(\phi_{n_h}) < \xi_{max} \\ \kappa_{max}, & \text{if } \mathcal{G}_k(\phi_{n_h}) = \xi_{max} \end{cases} \tag{14}$$

其中， κ_{min}、 $\kappa_{partial}$和 κ_{max} 分别表示不合规、部分合规和完全合规的安全评分，而 ξ_{min} 和 ξ_{max} 定义了控制评分边界值。

最后，通过加权方式获得部署分配 ϕ_{n_h} 下任务 n_h 的整体安全评分所有相关安全控制项的汇总 ：

$$\mathcal{S}(\phi_{n_h}) = \frac{\sum_{k\in\mathcal{K}_h} w_k^c \times CtrlLevel(\mathcal{G}_k(\phi_{n_h}))}{\sum_{k\in\mathcal{K}_h} w_k^c} \tag{15}$$

其中， w_k^c 是控制项 C_k的权重，反映其在整体安全评估中的重要性。

硬约束与软约束处理： 在实际应用中，安全控制可根据其关键程度分为两类：硬约束和软约束。硬约束代表必须严格满足的关键安全控制，以确保应用的基本安全要求。这些控制通常与法规合规、数据隐私保护或关键任务安全策略相关。另一方面，软约束代表可取的安全增强措施，在优化过程中可与性能或成本等其他目标进行权衡。

我们定义一组硬约束控制项 $\mathcal{H}$，其为所有安全控制的子集，并应在可能的情况下实现完全合规。为有效处理硬约束违规，我们在任务层面引入惩罚机制。在部署分配 ϕ_{n_h} 下，任务 n_h 的安全评分计算如下：

$$\mathcal{S}(\phi_{n_h}) = \begin{cases} \frac{\sum_{k\in\mathcal{K}_h} w_k^c \times CtrlLevel(\mathcal{G}_k(\phi_{n_h}))}{\sum_{k\in\mathcal{K}_h} w_k^c}, & \text{if } \forall k \in (\mathcal{K}_h \cap \mathcal{H}), \\ & \quad CtrlLevel(\mathcal{G}_k(\phi_{n_h})) = 100 \\ P_{constraint}, & \text{otherwise} \end{cases} \tag{16}$$

其中， $P_{constraint}$ 是一个较大的负惩罚值，用于对硬约束违规进行严厉惩罚。

SPA-DDRL 的首要安全目标是确保安全合规性（可行性），该目标由两个子目标定义：A）定量合规性：在确保任务所需安全控制与已部署资源的安全能力项高度匹配的情况下，最大化总体服务安全评分（$\mathcal{S}$（Φ））；B）可行性强制执行：严格执行硬约束（$\mathcal{H}$）。任何违反关键安全控制的服务部署都必须通过目标函数中的严厉惩罚予以淘汰，从而确保优化后的解决方案始终具备安全可行性。

考虑服务中每项任务分配的安全评分，服务的总安全评分为：

$$\mathcal{S}(\Phi) = \frac{\sum_{h=1}^{K} \mathcal{S}(\phi_{n_h})}{K} \tag{17}$$

3.2.3 优化问题

主要优化目标是在最大化服务部署的安全评分的同时，最小化服务的响应时间。为此，需要确定执行服务任务的最优部署方案。为平衡这些可能相互冲突的目标，我们采用加权多目标优化方法，并对目标分量进行归一化：

$$\min \mathcal{W}(\Phi) = \min\Big(\alpha\, NORM_{\mathcal{L}}\big(\mathcal{L}(\Phi)\big) + \beta\, NORM_{\mathcal{S}}\big(\mathcal{S}(\Phi)\big)\Big) \tag{18}$$

其中

$$NORM_{\mathcal{L}}(\mathcal{L}(\Phi)) = \frac{\mathcal{L}(\Phi) - \mathcal{L}_{min}}{\mathcal{L}_{max} - \mathcal{L}_{min}} \tag{19}$$

$$NORM_{\mathcal{S}}(\mathcal{S}(\Phi)) = \frac{\mathcal{S}_{max} - \mathcal{S}(\Phi)}{\mathcal{S}_{max} - \mathcal{S}_{min}} \tag{20}$$

在目标函数中，$NORM_{\mathcal{L}}$ 和 $NORM_{\mathcal{S}}$ 分别是响应时间和安全评分的归一化函数，其中 $NORM_{\mathcal{S}}$ 将安全评分最大化问题转换为等效的最小化形式。$\mathcal{L}_{min}$ 和 $\mathcal{L}_{max}$ 表示可能的最小和最大响应时间，$\mathcal{S}_{max}$ 和 $\mathcal{S}_{min}$ 表示可能的最大和最小安全评分。通过这种归一化，两个目标分量被缩放到相同范围，使权重参数α 和 β 能够有效平衡响应时间与安全目标的相对重要性。我们的目标函数是最小化加权目标：

$$\min(\mathcal{W}(\Phi)) \tag{21}$$

可行部署必须满足若干类约束。对于分配唯一性（$CST1$），每个任务必须恰好映射至一台服务器，通过以下方式实现：

$$CST1: \sum_{r^{p,q}\in\mathcal{R}} \mathbb{I}_{\phi_{n_h}=r^{p,q}} = 1,\ \forall n_h \in \mathcal{N} \tag{22}$$

其中，指示函数$\mathbb{I}_{\phi_{n_h}=r^{p,\,q}}$当任务 n_h 被分配至服务器 $r^{p,\,q}$（即 $\phi_{n_h}=r^{p,\,q}$）时等于 1，否则为 0。

资源容量约束可防止服务器超额订阅。内存（CST2）和存储（CST3）容量约束确保分配至每台服务器的所有任务的总资源需求均不超过其物理容量：

$$CST2: \sum_{n_h\in\mathcal{N}} \mathbb{I}_{\phi_{n_h}=r^{p,q}} \times m_h \le M_{r^{p,q}},\ \forall r^{p,q}\in\mathcal{R} \tag{23}$$

$$CST3: \sum_{n_h\in\mathcal{N}} \mathbb{I}_{\phi_{n_h}=r^{p,q}} \times st_h \le ST_{r^{p,q}},\ \forall r^{p,q}\in\mathcal{R} \tag{24}$$

截止时间约束（$CST4$）确保每个任务在其规定的时间范围内完成：

$$CST4:\ T_{\phi_{n_h}} \le d_h,\ \forall n_h \in \mathcal{N} \tag{25}$$

权重归一化约束确保安全层级与目标函数中的权重分布合理。$CST5$ 确保每个安全控制内的能力权重之和为 1，$CST6$ 确保每个任务的控制权重之和为 1，$CST7$ 确保响应时间权重与安全评分权重之和为 1：

$$CST5: \sum_{l\in\mathcal{L}_k} w_l^{cp} = 1,\ \forall k\in\mathcal{K}_h, \forall n_h\in\mathcal{N} \tag{26}$$

$$CST6: \sum_{k\in\mathcal{K}_h} w_k^{c} = 1,\ \forall n_h\in\mathcal{N} \tag{27}$$

$$CST7:\ \alpha+\beta=1,\ \alpha,\beta\ge 0 \tag{28}$$

该优化问题旨在解决部署决策，同时满足多个运行约束。随着服务器和任务数量增加，解空间呈指数级增长，因此该问题属于NP难类[2]，，无法在多项式时间内求得最优解。

4 深度强化学习模型

服务放置问题需要做出一系列相互依赖的决策，其中每项任务分配都会影响即时性能和未来的放置选项。这种不确定条件下的序贯决策可自然地建模为马尔可夫决策过程（MDP），其定义为元组 $\langle \mathbb{S},\ \mathbb{A},\ \mathbb{P},\ \mathbb{R},\ \gamma\rangle$。

在每个决策周期 t，智能体观察状态 $s_t\in\mathbb{S}$（表征当前任务和系统条件），执行动作 $a_t\in\mathbb{A}$ 以选择服务器，并以概率 $\mathbb{P}(s_{t+1}|s_t,\ a_t)$ 转移到状态 s_{t+1}，同时获得奖励 $r_t=\mathbb{R}(s_t,\ a_t)$。折扣因子 $\gamma\in[0,1]$ 用于平衡即时奖励与未来奖励。智能体学习一个将状态映射到动作的策略π：$\mathbb{S}\to\mathbb{A}$，以最大化期望回报：

$$J(\pi)=\mathbb{E}_\pi\left[\sum_{t=0}^{|\mathcal{N}|-1}\gamma^t r_t \mid s_0\right] \tag{29}$$

我们用于安全感知服务放置的深度强化学习公式由以下关键组件构成：

- **状态空间 $\mathbb{S}$：**在每个时间步 $t\in\mathbb{T}$，状态 s_t ∈$\mathbb{S}$同时反映待放置的当前任务和所有服务器的动态状态：

$$s_t=(F_t^{\mathcal{N}},F_t^{\mathcal{R}}) \tag{30}$$

其中，$F_t^{\mathcal{N}}=\{f_i^{n_h}\mid 1\le i\le k_1\}$ 表示待放置任务 n_h 的特征向量（包括CPU周期 c_h、内存 m_h、存储 st_h、安全要求 s_h和数据依赖关系），而 $F_t^{\mathcal{R}}=\{f_j^{r^{p,q}}\mid r^{p,q}\in\mathcal{R}, 1\le j\le k_2\}$ 则编码所有 R 服务器的特征向量（包括处理能力、内存/存储利用率、带宽、位置和安全能力）。

- **动作空间 $\mathbb{A}$：** 动作空间对应于可用服务器的集合 $\mathbb{A}=\mathcal{R}$。在每个决策步骤 t，智能体选择动作 $a_t\in\mathbb{A}$，将当前任务n_h 分配给特定服务器，形式化表示为：

$$a_t: n_h\mapsto r^{p,q},\quad \text{where } r^{p,q}\in\mathcal{R} \tag{31}$$

- **奖励函数 $\mathbb{R}$：** 由于深度强化学习智能体会针对各个任务进行序贯放置决策，因此我们定义了一个任务级奖励函数，将服务级目标 $\mathcal{W}(\Phi)$ 分解自公式 21。通过序贯决策，智能体学习在考虑服务性能和安全性的累积影响的同时优化任务放置；这些影响通过编码先前放置决策的状态表示体现。奖励函数通过惩罚机制纳入截止时间约束：

$$r_t=\begin{cases}-\mathcal{W}(\phi_{n_h}), & \text{if } T_{\phi_{n_h}}\le d_h\\ P_{failure}, & \text{otherwise}\end{cases} \tag{32}$$

其中，$T_{\phi_{n_h}}$ 和 d_h 分别为任务完成时间和截止时间约束。负号将最小化目标转换为奖励最大化问题。较大的惩罚 $P_{failure}$ 引导智能体避免截止时间违规，从而确保获得可行解。

MDP建模捕捉了我们安全感知服务放置问题的核心属性，同时为应用深度强化学习技术提供了系统化的基础。状态表示涵盖任务需求和提供商能力，动作空间支持灵活的资源选择，而奖励函数直接对应优化目标，确保学习过程与优化结果保持一致。

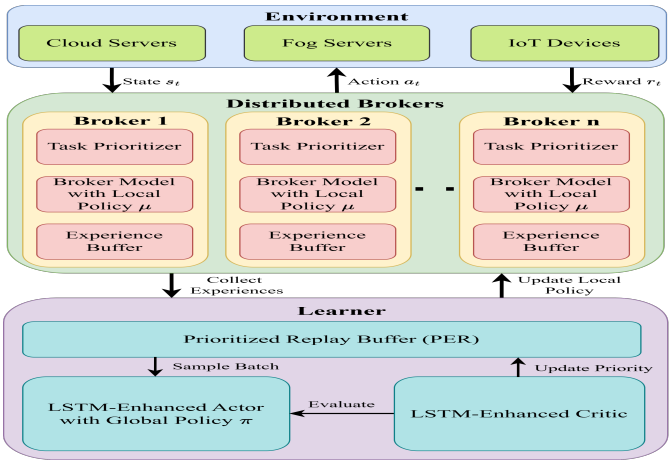


图 2：SPA-DDRL框架概览

5 SPA-DDRL：分布式深度强化学习框架

本节介绍 SPA-DDRL 框架，即面向雾计算环境中高吞吐量服务部署的安全性与性能感知分布式深度强化学习框架。SPA-DDRL 将演员-评论家架构与优先经验回放和 LSTM网络相结合，以在高度异构且动态变化的计算环境中实现安全性与性能双目标的高效优化。

SPA-DDRL采用分布式代理—学习器架构，将多个智能代理部署在异构服务器上，以处理本地放置决策，同时由集中式学习器协调全局策略优化。每个代理利用LSTM增强型演员网络进行实时决策，而学习器则维护包含LSTM组件的演员网络和评论家网络，以开展全面的策略学习与价值估计。这种分布式设计能够在异构雾计算环境中实现可扩展的安全性—性能优化，其中安全要求和性能需求会动态变化。图 2 展示了SPA-DDRL框架的概览。下文将详细介绍代理和学习器的运行机制。

5.1 分布式代理操作

SPA-DDRL中的分布式代理充当遍布雾计算基础设施的自主决策实体。每个代理处理传入的服务请求，并利用本地维护的LSTM增强型演员网络生成安全感知放置决策。分布式部署支持并行处理多个服务请求，同时确保符合安全要求和性能目标。算法 1概述了代理的运行流程。

每个代理都维护一个本地策略 μ，并定期与学习器的全局策略 π 同步，以确保分布式系统中的一致性（第 3 行）。在每个决策周期中，代理会先处理最多 N_{steps} 个放置决策，再与学习器同步。当有新服务到达时，代理会从请求队列 Q_R中检索服务元数据，包括安全约束、性能需求和任务依赖关系（第 7 行）。任务优先级排序函数会分析服务有向无环图的结构，并采用安全性感知调度生成有序的任务序列 Ψ_G，以优先处理安全性

算法1： SPA-DDRL 代理操作 s

输入 ：π：来自学习器的全局策略
/* N_{steps}：每个周期的决策步骤，μ：本地代理策略，EB：经验缓冲区，Q_R：请求队列，G：服务实例 */
1 $status_{new} \leftarrow True$
2 **当** $True$ **时执行**
3 $\mu \leftarrow$同步策略(π)
4 $step \leftarrow 0$
5 **当** $step < N_{steps}$ **时执行**
6 **如果** $status_{new} = True$ **然后**
7 $G \leftarrow Q_R$.dequeue()
8 $\Psi_G \leftarrow$任务优先级排序器(G)
9 $s_t \leftarrow$ 状态组合器($G, \mathcal{R}, \Psi_G$)
10 $status_{new} \leftarrow False$
11 **end**
12 $s_t \leftarrow$状态归一化器(s_t)
13 $a_t \leftarrow$ 决策引擎(s_t, μ) % 长短期记忆网络增强部署决策
14 $r_t \leftarrow$ 奖励评估器($s_t, \ a_t$) % → 式 32
15 $s_{t+1} \leftarrow$ 状态转移(s_t, a_t)
16 $p_t \leftarrow$ 优先级分配器(s_t, a_t, r_t, s_{t+1}) % 优先级计算
17 EB.store($s_t, a_t, r_t, s_{t+1}, p_t$)
18 **如果**服务完成(G) **则**
19 服务评估器(G)
20 $status_{new} \leftarrow True$
21 **end**
22 $step \leftarrow step + 1$
23 **end**
24 传输经验(EB) % 发送优先经验给学习器
25 **结束**

敏感操作和关键路径任务（第 8 行）。

状态组合器() 函数通过组合服务器能力向量 $F_t^{\mathcal{R}}$、当前任务特征 $F_t^{\mathcal{N}}$和安全合规指标，构建决策状态 s_t（第 9 行）。随后，决策引擎()对该状态表示进行处理，并利用长短期记忆网络增强的本地策略μ 生成放置动作 a_t （第 13 行）。奖励评估器()根据公式 32 中定义的双目标函数计算即时奖励 r_t （第 14 行）。经验元组 (s_t, a_t, r_t, s_{t+1}) 由优先级分配器() 函数进行优先级排序，该函数会综合考虑时序差分误差的大小以及安全违规等特定领域因素（第 16 行）。已完成的经验会累积在本地经验缓冲区 EB 中，直至与学习器同步（第 17 行）。

5.2 集中式学习协调

SPA-DDRL 中的学习器通过聚合来自分布式代理的优先经验，并更新演员网络和评论家网络，协调全局策略优化。学习器采用先进的离策略校正技术，以处理代理执行与学习器更新之间固有的策略滞后。算法 2 详细说明了学习协调过程。

离策略校正机制： SPA-DDRL 的分布式特性导致代理策略 μ 与学习器的目标策略 π [30]之间出现策略偏离。为应对这一挑战，我们结合使用重要性采样校正和梯度裁剪。重要性采样机制会针对策略差距[30], 进行调整，而梯度裁剪则可防止破坏性的参数更新，以免使学习过程 [31]失稳。

如算法 2所示，学习器会持续

从活跃代理接收优先经验回放批次，并维护一个集中式经验回放缓冲区 RB （第 4 行）。当积累了足够多的经验（$|RB| \geq BatchSize$）时，*SampleGenerator()* 函数会通过优先采样创建训练批次 SB us- ing（第 5-7 行）。*NetworkOptimizer()*函数（从第 10 行开始）执行基于梯度的更新。对于每个经验元组$(s_j,a_j,r_j,s_{j+1}) \in SB$，算法会计算重要性采样比率（第 12 行）：

$$\chi_j = \frac{\pi_\theta(a_j|s_j)}{\mu(a_j|s_j)} \tag{33}$$

该比率用于校正代理执行策略 μ 与学习器目标策略 π_θ之间的策略偏离。接下来，第 13 行按以下方式计算校正后的时序差分误差:

$$\psi_j = \varrho_j \left(r_j + \gamma V_\phi(s_{j+1}) - V_\phi(s_j)\right) \tag{34}$$

其中，$\varrho_j = \min(\bar{\varrho},\chi_j)$ 是裁剪重要性权重，而 $V_\phi(s_j)$ 表示评论家网络的状态价值估计。

优势估计纳入了时间依赖关系，并在第 14 行进行计算:

$$\hat{V}_j = \sum_{k=0}^{H-1} (\tau\gamma)^k \left(\prod_{l=0}^{k-1} \sigma_l\right) \psi_{j+k} \tag{35}$$

其中，τ 控制偏差-方差权衡，γ 是折扣因子，σ_l 是重要性采样裁剪权重。计算完批次中所有经验对应的这些值后，即计算策略梯度（第 16 行）：

$$\nabla_\theta L_\pi(\theta) = \frac{1}{|SB|} \sum_{j \in SB} \min\left(\chi_j \cdot \hat{V}_j, \text{clip}(\chi_j, 1-\eta, 1+\eta) \cdot \hat{V}_j\right) \tag{36}$$

其中，$\text{clip}(\cdot, 1-\eta, 1+\eta)$ 是阈值为 η 的裁剪操作，可确保策略更新稳定。评论家网络梯度在第 17 行计算:

$$\nabla_\phi L_V(\phi) = \frac{1}{|SB|} \sum_j \left(V_\phi(s_j) - \hat{R}_j\right) \nabla_\phi V_\phi(s_j) \tag{37}$$

其中，ϕ 和$\hat{} R_j$ 分别表示评论家参数和目标价值。随后，使用计算出的梯度更新网络参数：演员参数 θ （第 18 行）和评论家参数ϕ （第 19 行）。最后，*优先级更新器()* 函数根据更新后的时序差分误差调整经验优先级（第 20 行），学习器通过*BroadcastPolicy()* （第 21 行）将更新后的全局策略重新分发给所有活跃代理。

5.3 优先经验管理

SPA-DDRL 实现了一种 PER机制，通过关注学习潜力最大的经验来提高学习效率。在传统的经验回放中，经验是均匀采样的，因此从信息量较少的转移中学习效率低下 [32]。PER 通过根据经验的时序差分误差幅度为其分配优先级权重来解决这一问题。

SPA-DDRL 中的经验优先级计算如下:

$$\text{priority}(e_j) = |\psi_j| + \varepsilon \tag{38}$$

其中 ψ_j 是公式 34中的时序差分误差，且 ε 确保

算法 2： SPA-DDRL 学习协调

输入 ： EB_i: 来自代理的经验批次
/* B: 代理列表， π: 全局策略， RB: 重放缓冲区， SB: 样本批次 */
1 **当** $True$ **时执行**
2 $ready \leftarrow False$， $RB \leftarrow \emptyset$
3 **当** $ready = False$ **时执行**
4 RB.append(EB_i) % 收集优先经验
5 **if** $|RB| \geq BatchSize$ **then**
6 $SB \leftarrow$ 样本生成器(RB) % 优先采样
7 $ready \leftarrow True$
8 **end**
9 **end**
10 网络优化器(SB):
11 **foreach** $(s_j,a_j,r_j,s_{j+1}) \in SB$ **do**
12 $\chi_j \leftarrow \frac{\pi_\theta(a_j|s_j)}{\mu(a_j|s_j)}$ % 重要性采样比率
13 $\psi_j \leftarrow \varrho_j(r_j + \gamma V_\phi(s_{j+1}) - V_\phi(s_j))$ % 时序差分误差
14 $\hat{} V_j \leftarrow \sum_{k=0}^{H-1} (\tau\gamma)^k (\prod_{l=0}^{k-1} \sigma_l)\psi_{j+k}$ % 优势函数
15 **end**
16 $\nabla_\theta L_\pi(\theta) \leftarrow \frac{1}{|SB|} \sum_{j \in SB} \min(\chi_j \cdot \hat{} V_j, \text{clip}(\chi_j, 1-\eta, 1+\eta) \cdot \hat{} V_j)$
17 $\nabla_\phi L_V(\phi) \leftarrow \frac{1}{|SB|} \sum_j (V_\phi(s_j) - \hat{} R_j) \nabla_\phi V_\phi(s_j)$
18 更新演员参数： $\theta \leftarrow \theta + \alpha_\pi \nabla_\theta L_\pi(\theta)$
19 更新评论家参数： $\phi \leftarrow \phi + \alpha_V \nabla_\phi L_V(\phi)$
20 PriorityUpdater(SB) % 更新经验优先级
21 BroadcastPolicy(B, π) % 分发更新后的策略
22 **结束**

所有经验的采样概率均非零。采样概率遵循优先级分布:

$$P_j = \frac{p_j^\nu}{\sum_k p_k^\nu} \tag{39}$$

其中，p_j 是经验 j 的优先级，ν 控制优先级强度（$\nu = 0$ 产生均匀采样）。为纠正采样偏差，在梯度计算过程中应用重要性采样权重:

$$w_j = \left(\frac{1}{N} \cdot \frac{1}{P_j}\right)^\iota \tag{40}$$

其中，N 为缓冲区容量，ι 控制偏差校正强度，通常在训练过程中从 0 逐渐调整至 1。

5.4 SPA-DDRL中的长短期记忆网络增强

SPA-DDRL将长短期记忆网络集成到演员-评论家架构中，以捕捉动态雾计算环境中的时间依赖关系。与传统前馈网络不同，长短期记忆网络能够保持长期记忆，这对于雾环境至关重要，因为当前的服务放置决策会影响未来的资源可用性和安全状态。长短期记忆网络增强使该框架能够对序贯决策模式进行建模，并从历史状态-动作相关性中学习。

长短期记忆网络增强型演员网络： 演员策略集成长短期记忆网络隐藏状态，以在各项放置决策之间保留时间上下文。该设计的主要优势在于能够识别周期性模式，例如服务请求中的时间规律、安全威胁的演变趋势以及资源利用率的变化模式。在时间步 t，长短期记忆网络增强型演员会处理当前状态表示和先前的隐藏状态:

$$h_t^\pi = \text{LSTM}_\theta([F_t^\mathcal{N}, F_t^\mathcal{R}], h_{t-1}^\pi) \tag{41}$$

其中，h_t^π是演员在时间 t的隐藏状态，θ 表示

演员网络参数，$[F_t^{\mathcal{N}}, F_t^{\mathcal{R}}]$ 是拼接状态表示。通过保留历史信息，演员能够做出更明智的决策，尤其是在识别相似服务类型的历史安全合规模式、任务放置与整体服务性能之间的时间相关性，以及资源可用性和安全能力的长期趋势方面。随后将策略分布计算如下：

$$\pi_\theta(a_t|s_t, h_t^\pi) = \text{softmax}(W_\pi h_t^\pi + b_\pi) \tag{42}$$

其中，W_π 和 b_π 是可学习参数，用于将隐藏状态映射为动作概率。

LSTM增强型评论家网络： 评论家网络采用长短期记忆网络组件，在考虑时间上下文的情况下估计状态价值。这对于安全性能评估尤为重要，因为部署决策的价值不仅取决于即时奖励，还取决于长期安全合规性和性能影响。长短期记忆网络使评论家能够对安全威胁的演变进行建模，这些威胁可能影响长期放置策略、安全控制实施与其有效性之间的时间依赖关系、资源安全能力随时间的动态变化，以及历史部署决策与长期服务性能之间的相关性。评论家的长短期记忆网络处理相同的状态输入，但保留独立的隐藏状态：

$$h_t^V = \text{LSTM}_\phi([F_t^{\mathcal{N}}, F_t^{\mathcal{R}}], h_{t-1}^V) \tag{43}$$

其中，h_t^V表示评论家的隐藏状态，ϕ 表示评论家参数。状态价值估计纳入了时间信息：

$$V_\phi(s_t, h_t^V) = W_V h_t^V + b_V \tag{44}$$

时间优势估计：时间上下文通过考虑历史奖励模式，使优势估计更加准确。这种改进后的优势计算能更好地反映决策的长期影响，尤其是在复杂的安全性与性能权衡场景中。优势函数如下：

$$A_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k} + \gamma^{T-t+1} V_\phi(s_{T+1}, h_{T+1}^V) - V_\phi(s_t, h_t^V) \tag{45}$$

其中，γ 为折扣因子，T为回合长度。

含时间依赖的梯度计算：策略梯度通过时间反向传播整合基于长短期记忆网络的时间依赖关系，使网络能够捕捉连续时间步之间复杂的相关模式。这对于建模安全威胁演化和资源动态变化至关重要：

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t, h_t^\pi) A_t \right] \tag{46}$$

其中，$\tau = \{s_0, a_0, r_0, \ldots, s_T, a_T, r_T\}$ 表示轨迹序列。

长短期记忆网络与优先经验回放相结合，可提高学习效率：时间意义重大的经验会获得优先采样，从而加快复杂时间模式的学习。以优化安全性与性能。分布式代理-学习器架构通过支持在雾基础设施中并行探索多种安全性与性能条件，进一步放大这些优势，从而实现更稳健的策略学习。

6 性能评估

本节介绍 SPA-DDRL 在雾计算环境中进行安全性与性能优化的实验评估。我们将说明评估设置，介绍用于比较的基线技术，配置超参数，并针对多个指标分析性能，以展示该框架在兼顾安全合规性与性能目标方面的有效性。

6.1 评估设置

为验证 SPA-DDRL 在异构雾计算中的有效性，我们设计并实现了一个多层计算基础设施仿真平台。该平台构建了一个包含 100 台服务器的分布式环境，其中包括 20 台高性能 CS、30 台中等性能 FS，以及 50 台资源受限型物联网设备。

计算基础设施设计： 遵循 [33], [34],CS 配备高端硬件资源：CPU 核心数为 4 至 32 个，每个核心的处理能力达到 10,000-100,000 MIPS，内存容量配置为 16-128 GB，存储空间为 500-10,000 GB。遵循[35], [36],FS 采用中等硬件配置：CPU 核心数为 2-8 个，处理能力为 5,000-20,000 MIPS，内存容量为 4-32 GB，存储空间为 200-500 GB。遵循 [35], [37], 物联网设备采用资源受限型终端：CPU 核心数为 1-2 个，处理能力为 1,000-5,000 MIPS，内存为 1-2 GB，存储容量为 10-100 GB。

网络连接建模： 系统采用全连接网络拓扑，通过带宽矩阵精确建模服务器/设备之间的通信能力。根据 [38], [39], [40], 每台服务器/设备的类型不同，其网络接口能力也各不相同：物联网设备支持 10-50 Mbps 的双向传输，文件系统支持 50-200 Mbps，云服务器支持 100-1,000 Mbps。

多层安全配置： 安全机制采用三层级结构：15 类基本安全控制措施 [29], ，每类包含 5 项实现能力，每项能力由 3 个配置项组成。不同类型的服务器/设备根据其计算能力和部署需求，实现 15 项安全控制中的不同子集。例如，资源丰富的云服务器可以实施全面的安全控制，而资源受限的物联网设备仅实施必要的安全机制，从而在整个雾计算基础设施中形成异构的安全态势。

在我们的实现中，在不失一般性的情况下，能力级函数（即公式12）定义为：

$$CapLevel(SR_l(\phi_{n_h})) = \begin{cases} 0, & \text{if } SR_l(\phi_{n_h}) = 0 \\ 25, & \text{if } 0 < SR_l(\phi_{n_h}) \leq 33 \\ 50, & \text{if } 33 < SR_l(\phi_{n_h}) \leq 66 \\ 75, & \text{if } 66 < SR_l(\phi_{n_h}) < 100 \\ 100, & \text{if } SR_l(\phi_{n_h}) = 100 \end{cases} \tag{47}$$

类似地，控制级函数（即公式14）为：

$$CtrlLevel(\mathcal{G}_k(\phi_{n_h})) = \begin{cases} 0, & \text{if } \mathcal{G}_k(\phi_{n_h}) = 0 \\ 50, & \text{if } 0 < \mathcal{G}_k(\phi_{n_h}) < 100 \\ 100, & \text{if } \mathcal{G}_k(\phi_{n_h}) = 100 \end{cases} \quad (48)$$

异构服务工作负载构造：我们基于有向无环图设计复杂的服务模型，以模拟真实的计算工作负载。与 [27], [41], 类似，每个服务都包含多个相互依赖的任务，这些任务具有不同的计算、通信和安全要求。服务生成过程分为两个阶段：拓扑结构构造和任务属性分配。在拓扑结构构造阶段，我们通过三个关键参数控制有向无环图的形状特征：任务数量 K 决定服务的规模复杂度，设置为 6 个级别 $K \in \{5, 10, 20, 40, 80, 100\}$；$fat$ 参数控制有向无环图的宽高比，影响任务的并行化程度，设置为 5 个级别 $fat \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$，其中较小的 fat 值会生成更多顺序执行结构，较大的fat值则会生成更多并行执行结构；$density$ 参数控制任务间的连接密度，影响依赖关系的复杂度，同样设置为 5 个级别 $density \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$，其中较小的 $density$ 值会生成松散耦合的任务结构，较大的 $density$ 值则会生成紧密耦合的任务结构。通过组合这三个参数，我们为每种参数配置生成 5 种不同的拓扑变体，以确保结构多样性。在任务属性分配阶段，我们为每种生成的拓扑结构分配具体的资源需求和约束。根据 [37], [42], 计算需求，每个任务的中央处理器工作负载设为 0.5–100 百万条指令，内存需求为 10–1,000 MB；对于时间约束，任务执行截止时间设为 10–1,000 毫秒；对于通信需求，任务间的数据传输量设为 1–1,000 KB；对于安全要求，每个任务会从 15 个安全控制类别中随机选择所需的安全服务。为增加属性配置多样性，每种拓扑结构都与 10 种不同的属性组合配对，使相同的依赖结构具有不同的资源和安全特征。通过上述生成策略，我们构建了一个包含 7,500 个异构服务实例的综合数据集，涵盖从简单线性任务到复杂并行任务的各种服务模式，以及从低安全要求到高安全要求的不同安全级别，为安全性与性能双目标优化提供了充足的测试场景。

实现了以下基线技术：

•X-DDRL：采用文献 [2] 中所述方法的扩展版本作为基线，并使用 IMPALA 作为底层分布式深度强化学习框架。我们更新了奖励函数，以支持安全优化。

• A3C-AHP：采用了 [20] 中方法的改进版本。该方法适用于由多个物联网设备、雾服务器和云服务器组成的异构计算环境中的服务部署。此外，奖励函数也经过更新，以支持服务部署中的安全性和响应时间优化。其底层分布式深度强化学习框架为 A3C。

表2：SPA-DDRL 的超参数

参数	值	参数	值
全连接层	2	学习 g 学习率 lr	0.01
梯度步数	2	折扣因子 γ	0.9
优化技术	Adam	批量大小	128
激活函数	Tanh	缓冲区大小	10000
裁剪常数 ϵ	0.2	V-trace ρ	1.0
GAE Lambda λ	0.95	V-trace c	1.0

表3：基线技术的超参数

超参数	X-DDRL	A3C-AHP	DRLIS	PARL	SCRA
全连接层	2	2	2	2	2
激活函数	TanH	TanH	TanH	ReLU	ReLU
学习率	0.01	0.001	0.01	0.01	0.01
折扣因子 γ	0.99	0.999	0.9	0.999	0.999
批量大小	64	32	128	32	64
缓冲区大小	100000	N/A	N/A	50000	50000

• DRLIS：采用了 [43] 中所述方法的扩展版本作为基线。奖励函数经过更新，以支持安全优化。该方法采用近端策略优化作为深度强化学习框架。

• PARL：将 [21] 中提出的方法的改进版本用作基线。该方法适用于异构计算环境中的服务部署。奖励函数经过更新，以支持安全性和响应时间优化。其底层深度强化学习框架基于双重深度Q网络。

• SCRA：将 [19] 中方法的增强版本用作基线。该技术适用于服务部署，并更新了奖励函数，以优化服务安全评分和响应时间。该方法使用深度Q网络作为深度强化学习框架。

6.2 超参数调优

SPA-DDRL 在所有分布式代理上采用相同的双层全连接神经网络，并使用 Tanh激活函数。通过网格搜索优化，我们将学习率设为 0.01，使用 Adam优化器，折扣因子 γ 设为 0.9，批量大小设为 128。V-trace 重要性采样权重 ρ和 c 均设为 1.0，裁剪阈值 ϵ 设为 0.2，GAE lambda 设为 0.95。经验缓冲区保留 10,000 个样本，每隔 2 步进行一次梯度更新。完整配置见表 2。为确保公平比较，所有基线技术均通过全面的网格搜索进行优化，其配置见表3。

6.3 性能分析

为评估 SPA-DDRL 的有效性，我们开展了多方面的性能分析，涵盖消融研究、收敛行为、可扩展性表现、训练效率和计算开销。

6.3.1 消融分析

为评估 SPA-DDRL 中关键组件的有效性，我们开展了消融分析，考察 LSTM 和 PER 对整体性能的各自贡献。在此实验中，我们使用复杂度各异的服务，其中训练数据集包含任务数量为 $K \in \{5, 10, 20, 80, 100\}$ 的服务，而评估则针对包含 $K = 40$ 个任务的服务进行。

图 3 展示了不同 SPA-DDRL 变体的收敛性能：不含长短期记忆网络和优先经验回放的 SPA-DDRL（基础版）

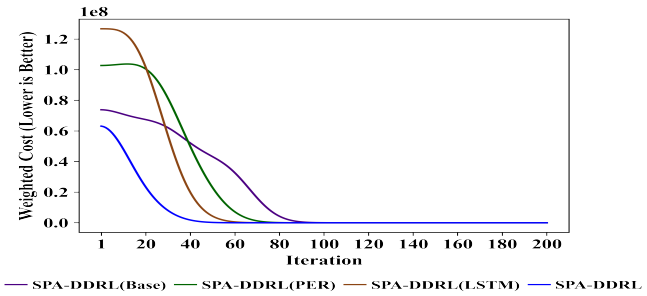


图3：SPA-DDRL 变体的消融分析

仅启用优先经验回放的 SPA-DDRL（PER）、仅启用长短期记忆网络的 SPA-DDRL（LSTM），以及同时包含这两个组件的完整 SPA-DDRL。结果表明，每个组件都显著促进了收敛速度的提升。SPA-DDRL 收敛最快，约在第 40 次迭代时收敛；其次是 SPA-DDRL（LSTM），约在第 60 次迭代时收敛；然后是 SPA-DDRL（PER），约在第 70 次迭代时收敛；最后是 SPA-DDRL（基础版），约在第 90 次迭代时收敛。长短期记忆网络组件提供时间依赖建模，帮助智能体更好地理解动态的雾计算环境，因此与基础版本相比，收敛速度约提升 33%。优先经验回放通过在训练过程中优先采样重要经验，提高了学习效率，使收敛速度提升 22%。在完整的 SPA-DDRL 中，这两个组件的结合产生了协同效应，带来了最高效的学习性能，提升幅度达 56%。总之，长短期记忆网络和优先经验回放对于 SPA-DDRL 框架都至关重要，二者共同使其在复杂的多目标服务部署中实现卓越的收敛性能。

6.3.2 收敛分析

本节从响应时间、安全评分和总体加权成本等方面，分析不同服务部署技术的收敛行为和优化性能。实验设置包括在任务数量为 $K \in \{5, 10, 20, 80, 100\}$ 的服务上进行训练，并在包含 $K = 40$ 个任务的服务上进行评估，确保评估复杂度处于训练范围内，以考察技术在未见过的服务配置上的鲁棒性。图4a、4b和4c 展示了各目标在 200 次迭代中的收敛趋势。

响应时间： 如图4a所示，SPA-DDRL 收敛至约 360 毫秒，显著优于 X-DDRL 的 430 毫秒（提升 16.3%）、A3C-AHP 的 950 毫秒（提升 62.1%）和 DRLIS 的 1050 毫秒（提升 65.7%）。PARL 和 SCRA 未能有效收敛，持续振荡，分别在 1300 毫秒和 1350 毫秒附近波动；与 SPA-DDRL 相比，性能分别下降 72.3% 和 73.3%。在收敛速度方面，SPA-DDRL 约在第 40 次迭代时趋于稳定，而 X-DDRL 约需 60 次迭代，A3C-AHP 约需 70 次迭代，DRLIS 则在持续振荡后约需 200 次迭代。

安全评分： 图4b 表明，SPA-DDRL 在大约第 50 次迭代时收敛至近似最优的安全评分，说明其完全满足硬性安全控制要求。X-DDRL 和 A3C-AHP 也找到了近似最优的可行解，但分别需要 60 次和 70 次迭代。

相比之下，DRLIS 最终停在 -2×10^6，表明其根本无法满足安全要求。PARL 和 SCRA 则出现严重的安全违规，在整个训练过程中持续围绕 -4×10^6 和 -6×10^6 震荡，说明它们的放置策略一再违反三层安全层级中的关键控制层约束。

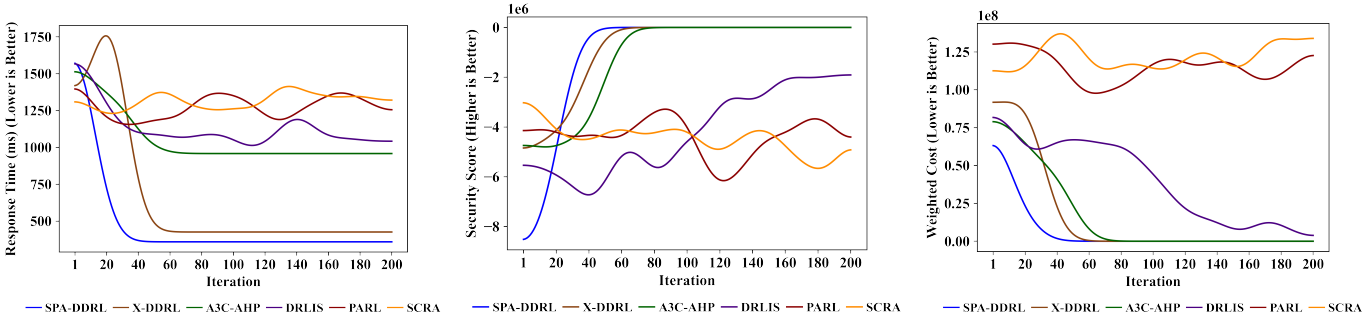
加权成本： 图 4c 表明，SPA-DDRL 具有卓越的联合目标优化能力，在约第 50 次迭代时收敛至接近零的加权成本。X-DDRL 和 A3C-AHP 分别在第 60 次和第 70 次迭代时收敛至接近零，得到的解虽次优但仍可行。DRLIS 在第 0.05×10^8 次迭代时将成本降至该值，但由于安全违规持续存在，仍未能达到真正的可行性。PARL 和 SCRA 的成本高得难以接受，在整个训练过程中始终分别在 1.22×10^8 和 1.32×10^8 附近波动，安全惩罚项主导了它们的目标函数。

这些结果表明，只有 SPA-DDRL、X-DDRL 和 A3C-AHP 成功平衡了安全性与性能之间的权衡；其中，SPA-DDRL 通过协同整合分布式学习、长短期记忆网络和优先经验回放，实现了最优性能和最快收敛。

6.3.3 系统规模分析

本实验评估服务器数量增加时各项服务部署技术的性能。服务器数量增加会扩大搜索空间并提高服务放置问题的复杂度，直接影响优化响应时间和安全目标时的决策过程。本实验使用任务数量为 $K \in \{5, 10, 20, 80, 100\}$的服务进行训练，并使用任务数量为 $K = 40$的服务进行评估，以与之前的实验保持一致。此外，我们还考察不同的服务器数量，其中服务器数量为$\in \{25, 50, 75, 100\}$。我们评估不同迭代次数 $\{25, 50, 100, 200\}$ 下的性能，以观察不同系统规模下的收敛行为。图 5a、5b和5c 展示了不同系统规模下的性能结果。某些技术未显示柱状图，表明它们在特定系统配置下未能找到满足严格安全控制要求的可行解。

响应时间： 如图 5a所示，SPA-DDRL 在所有系统配置下始终实现最低响应时间。随着训练迭代次数增加，所有技术均表现出性能提升。在第 200 次迭代时，SPA-DDRL 收敛至约 300 ms，X-DDRL 为 400-1100 ms，A3C-AHP 为 700-2000 ms，而 DRLIS、PARL 和 SCRA 的响应时间范围分别为 1000-2100 ms、1400-2100 ms 和 1500-2100 ms；与 SPA-DDRL 相比，其性能下降了 80-85%。在相同迭代次数下，随着服务器数量从 25 增加到 100，响应时间通常会因搜索空间扩大和特征复杂度增加而上升。SPA-DDRL 展现出优异的可扩展性，在第 100 次和第 200 次迭代时响应时间趋于稳定。相比之下，基线方法的性能会随着服务器数量增加而显著下降：X-DDRL 从低于 500 ms 增加到超过 1000 ms，A3C-AHP 从 700 ms 增加到 2000 ms，DRLIS 从 1000 ms 增加到 2000 ms，PARL 从 1500 ms 增加到 2000 ms，SCRA 从 1600 ms 增加到 2000 ms，表明其可扩展性较差



(a) 响应时间收敛 (b) 安全评分收敛 (c) 加权成本收敛

图 4：SPA-DDRL 与基线方法在 (a) 响应时间、(b) 安全评分和 (c) 加权成本方面的收敛分析。

较大规模下基线方法的 g规模 y系统。

安全评分： 如图 5b所示，SPA-DDRL 在所有系统配置和迭代次数下均能保持可行的符合安全要求的解，安全评分始终约为 1600。在第 25 次迭代时，SPA-DDRL 是唯一能在所有服务器数量下生成有效解的方法，而 X-DDRL 仅在 25 台服务器的配置下表现出有限的可行性，且安全评分较低（约为 1450）。到第 50 次迭代时，SPA-DDRL 仍保持完全可行，X-DDRL 扩展到更多配置（25 台和 50 台服务器），A3C-AHP 也开始表现出部分可行性（25 台服务器）。在第 100 次和第 200 次迭代时，SPA-DDRL 继续在所有系统规模下展现完全可行性，而 X-DDRL、A3C-AHP 和 DRLIS 在部分配置下实现可行性，安全评分约为 1450。在所有迭代次数和系统规模下，PARL 和 SCRA 均完全没有可行解，证实了它们在异构雾环境中满足严格安全控制的根本能力不足。

加权成本： 如图 5c所示，SPA- DDRL 在安全性与性能联合优化方面展现出卓越的可扩展性。可行性演变呈现出逐步收敛的趋势。在第 25 次迭代时，SPA- DDRL 是唯一一种在所有服务器数量下均能产生可行解的技术，加权成本范围为 0.35-0.50。到第 50 次迭代时，X-DDRL 和 A3C-AHP 在部分配置下实现可行，成本约为 0.40-0.55，而 SPA-DDRL 将成本优化至 0.36-0.40。在第 100 次和第 200 次迭代时，SPA-DDRL 在所有系统规模下均稳定于 0.36 的加权成本；X-DDRL 的成本范围为 0.40-0.55，A3C-AHP 约为 0.47，而 DRLIS 仅在 0.55 时实现有限的可行性。在所有迭代和配置中，PARL 和 SCRA 均完全无法找到可行解；缺失的柱状条表明它们始终未能满足严格的安全约束。

这些结果表明，随着系统规模不断扩大，SPA-DDRL 能够始终保持解的可行性和最优性能，而基线技术的性能则逐渐下降，并且随着环境复杂度的增加，无法满足严格的安全约束。这种卓越的可扩展性源于 SPA-DDRL 的分布式代理-学习器架构支持并行探索、LSTM 增强的时序建模能够捕捉复杂的依赖关系，以及 PER 有助于从大规模异构雾计算部署中的多样化放置场景高效学习。

6.3.4 加速比分析

本分析评估了不同技术在训练期间获取预定义数量经验轨迹的效率。与雾计算环境的交互越快，就能收集到越多样化的经验，从而加速收敛。加速比为：

$$SP = \frac{Time_R}{Time_T} \tag{49}$$

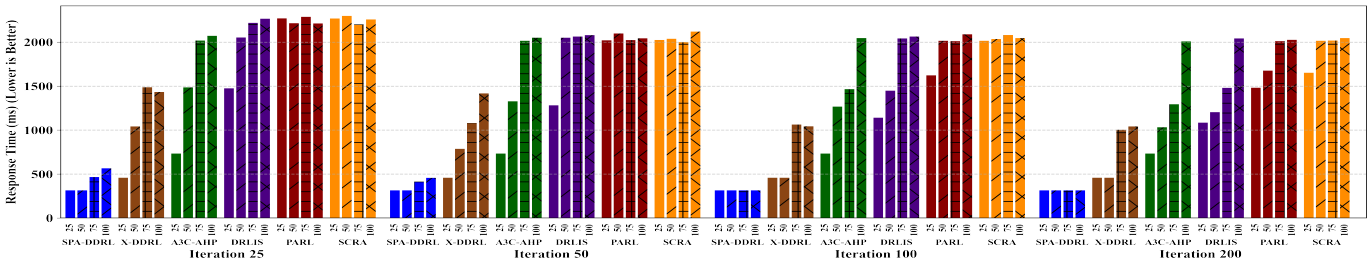
其中，$Time_R$ 表示 SPA-DDRL 使用单个工作节点达到 150000 个环境步所需的时间，$Time_T$ 表示所评估技术达到相同步数所需的时间。

如图6所示，采用 8 个工作节点的 SPA-DDRL 实现了约 2。7×的加速比，优于其他分布式技术。X-DDRL 和 A3C-AHP 的加速比分别为 2。6× 和 2。4× ，而它们的单工作节点变体提升有限。传统的非分布式方法，如 DRLIS、SCRA 和 PARL，表现不如基线（加速比 < 1.0），表明其训练进度较慢。SPA-DDRL 的加速比更高，得益于其有效的并行化机制，使多个工作节点能够探索不同的动作空间，同时共享优先经验。这种设计可显著缩短训练时间，并支持在动态雾计算环境中快速适应，因此 SPA-DDRL 非常适合需要快速收敛和响应能力的实际部署。

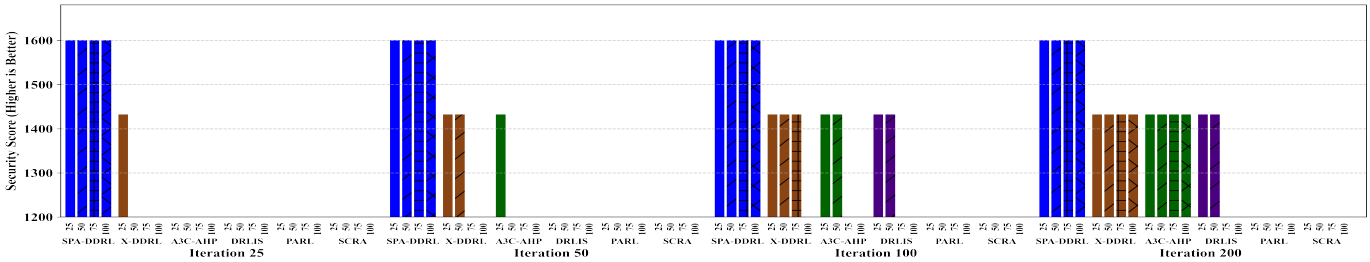
6.3.5 决策时间开销分析

本实验评估了每种服务放置技术的平均决策时间开销（DTO）。DTO 定义为调度过程中为每项服务做出部署决策所需的平均时间。

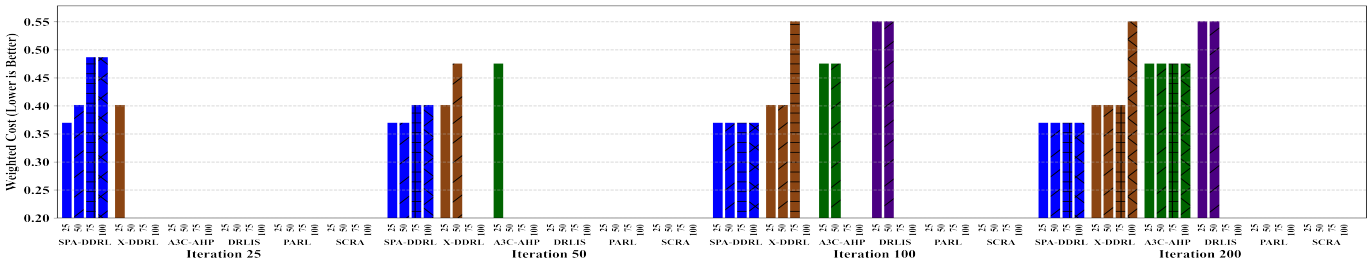
如图所示7，SPA-DDRL 的 DTO 最高，约为 87 ms；相比之下，SPA-DDRL(Base) 为 56 ms，而 X-DDRL、A3C-AHP、DRLIS、PARL 和 SCRA 为 48–58 ms。SPA-DDRL 的开销增加，主要是因为长短期记忆网络（LSTM）进行时间依赖建模，以及优先经验回放（PER）分析历史状态并对经验排序以进行采样所带来的计算成本。尽管 SPA-DDRL 的 DTO 较高，但鉴于其收敛速度、解的质量和可扩展性均更优，这一开销是可以接受的，正如先前实验所示。DTO 的小幅增加是合理的，因为它带来了显著的性能优化和可扩展性表现提升。



(a) 响应时间



(b) 安全评分



(c) 加权成本

图5：系统　　比较不同服务器配置下各技术性能的尺寸分析

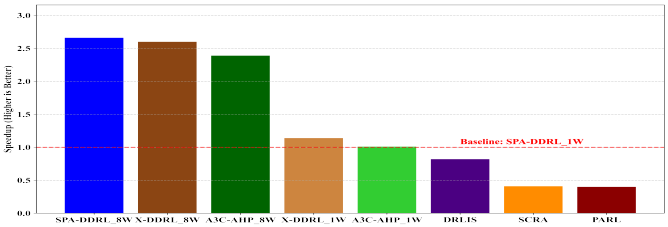


图6：不同工作节点数量下的加速比分析

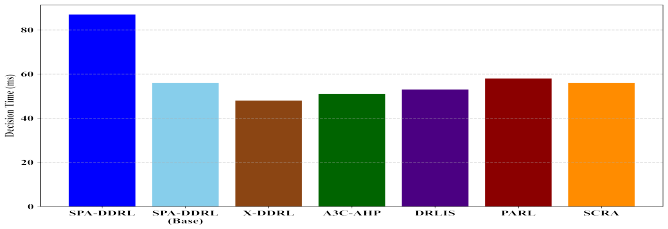


图7：决策时间开销分析

7 结论与未来工作

本文提出了 SPA-DDRL，这是一种分布式框架，旨在异构雾计算环境中联合优化安全合规性与响应时间。我们构建了一个双目标优化模型，并以一种新颖的三层安全量化方案为基础——该方案包含配置、能力和控制级评估——以严格执行数据加密和访问控制等严格策略。通过将这一量化机制融入奖励函数，该框架即使在动态条件下也能确保服务部署符合要求。此外，所提出的分布式代理—学习器架构结合了用于时序建模的 LSTM网络、优先经验回放（PER）和离策略校正，有效兼顾了训练稳定性与可扩展性。大量评估结果表明，SPA-DDRL 在收敛速度和解的质量方面优于最先进基线方法。尤为重要的是，随着系统规模扩大，该框架仍能保持解的可行性和严格的安全合规性，解决了竞争方法常产生不合规解这一关键缺陷。这些能力证明，SPA-DDRL 是一种稳健的解决方案，可用于大规模雾计算生态系统中的关键任务安全部署。

作为未来工作的一部分，我们计划扩展 SPA-DDRL，使其能够处理联邦学习场景；在这些场景中，除了安全性和性能目标之外，隐私保护也至关重要。此外，我们还打算探索自适应安全要求调整机制，根据实时威胁情报和不断演变的攻击模式，动态调整安全级别，从而进一步增强其在雾计算环境中的安全韧性。

参考文献

[1] Q.He,J.Lin,H. Fang,X.Wang,M.Huang,X.Yi, andK.Yu，“融合物联网与6G：边缘智能的应用、挑战与未来方向”，*IEEE*服务计算汇刊，2025年。[2] M. Goudarzi, M. S. Palaniswami, and R. Buyya，“一种用于边缘与雾计算环境中应用放置的分布式深度强化学习技术”，*IEEE Transactions on MobileComputing*，第22卷，第5期，2023年。

[3] A. Du, J. Jia, S. Dustdar, J. Chen, and X. Wang，“分层协作移动边缘计算系统中的在线服务部署、任务调度与资源分配”，IEEE服务计算汇刊，2025年。[4] Q. Deng, M. Goudarzi, and R. Buyya，“Fogbus2：用于将物联网赋能系统与边缘计算和云计算集成的轻量级分布式容器框架”，载于《新兴分布式环境大数据国际研讨会论文集》，2021年，第1–8页。[5] M. Goudarzi, M. Palaniswami, and R. Buyya，“超密集毫微微蜂窝网络中的雾驱动动态资源分配技术”，Journal of Network and Computer Applications，第145卷，文章编号102407，2019年。[6] F. Al-Doghman, N. Moustafa, I. Khalil, Z. Tari, and A. Zomaya，“边缘计算中的人工智能赋能的安全微服务：机遇与挑战”，IEEE服务计算汇刊，第16卷，第2期，2023年。[7] S. N. AL-Jaradi, M. K. Hasan, S. Islam, B. Pandey, J. Baili, M. A. Khan, H. S. Abbas, and N. H. S. Suhai，“利用智能超表面和物联网，在6G移动网络消费电子应用中实时监控通信协议”，IEEE Transactions on Consumer Electronics，2025年。[8] L. Kong等，“边缘计算驱动的物联网：综述”，ACM Computing Surveys，第55卷，第8期，第1–41页，2022年。[9] Q. Deng, M. Goudarzi, A. Shaghaghi, M. Sarvi, and R. Buyya，“集成边云计算环境中容器化物联网应用的安全框架”，Future Generation Computer Systems，文章编号108010，2025年。[10] X. Wang, B. Veeravalli, J. Song, and H. Liu，“面向动态雾环境的安全性与时间感知最优数据放置设计与评估”，IEEE Transactions on Parallel and Distributed Systems，第34卷，第2期，第489–500页，2022年。[11] H. Sun, H. Yu, G. Fan, L. Chen, and Z. Liu，“边缘云中的安全感知且有时间保证的服务部署”，IEEE Transactions on Network and Service Management，第20卷，第1期，第711–725页，2022年。[12] Y. Huang, B. Lu, Y. Luo, J. Fang, Q. Fu, S. Qin, and J. Liu，“动态星载RIS辅助移动边缘计算中的能效任务卸载与深度神经网络推理：基于分解的深度强化学习”，IEEE Transactions on Wireless Co mmunications，2025年。[13] Z. A. Mann，“安全软件部署与配置”，Future Generation Computer Systems，第110卷，第243–253页，2020年。[14] I. A. Elgendy, W. Zhang, Y.-C. Tian, and K. Li，“面向移动边缘计算的资源分配与计算卸载及数据安全”，Future Generation Computer Systems，第100卷，第531–541页，2019年。[15] V. Casola, A. De Benedictis, S. Di Martino, N. Mazzocca, and L. L. L. Starace，“工业物联网中云边系统的安全感知部署优化”，IEEE Internet of Things Journal，第8卷，第16期，第12 724–12 733页，2021年。[16] S. Javanmardi, M. Shojafar, R. Mohammadi, V. Persico, and A. Pescap`e，“S-FOS：一种用于基于软件定义网络的物联网雾网络中性能优化的安全工作流调度方法”，Journal of I nformation Security and Applications，第72卷，文章编号103404，2023年。[17] Z. ´A. Mann，“安全与隐私感知的物联网应用放置及用户分配”，载于《欧洲计算机安全研究研讨会》。Springer，2021年，第296–316页。[18] A. Singh, N. Auluck, O. Rana, A. Jones, and S. Nepal，“雾网络中的实时安全感知任务调度”，IEEE服务计算汇刊，第14卷，第6期，第1981–1994页，2021年。[19] H. Zhang, J. Wang, H. Zhang, and C. Bu，“无服务器多云边缘计算中基于深度强化学习的安全计算资源分配”，Future Generation Computer Systems，第151卷，第152–161页，2024年。[20] A. M. Rahmani, J. Tanveer, F. S. Gharehchopogh, S. Rajabi, and M. Hosseinzadeh，“一种面向区块链赋能的移动边缘计算网络中多用户优化的新型卸载策略，以提升物联网性能”，Computers and Electrical Engineering，第119卷，文章编号109514，2024年。[21] M. Ebrahim and A. Hafid，“雾网络中的隐私感知负载均衡：一种强化学习方法”，Computer Networks，第237卷，文章编号110095，2023年。[22] J. Sun, Q. Gao, C. Wu, Y. Li, J. Wang, and D. Niyato，“通过受约束深度强化学习实现安全资源分配”，载于《机器学习与网络安全国际会议》。Springer，2024年，第1–15页。

[23] M. Mohammadi、F. BahraniPour、S. Ebrahimi Mood 和 M. Farshi，“雾计算中基于元启发式算法的安全感知资源分配”，Cluster Computing，第 28 卷，第 2 期，第 104 页，2025年。[24] R. Du、J. Zhao 和 Y. Gao，“面向海洋物联网的增强遗传算法安全计算卸载”，Journal of Grid Computing，第 23 卷，第 1 期，第 1–19 页，2025年。[25] V. Thangaraj 和 T. Raja Sree，“MSCO：区块链赋能的雾计算环境中的移动性感知安全计算卸载”，Journal of Cloud Computing，第 13 卷，第 1 期，第 88 页，2024年。[26] R. Garaali、C. Chaieb、W. Ajib 和 M. Afif，“面向移动边缘计算的基于学习的任务卸载”，载于 IEEE国际通信会议（ICC）论文集。IEEE，2022年，第 1659–1664 页。[27] J. Wang、J. Hu、G. Min、A. Y. Zomaya 和 N. Georgalas，“基于元强化学习的边缘计算快速自适应任务卸载”，IEEE Transactions on Parallel and Distributed Systems，第 32 卷，第 1 期，第 242–253 页，2020年。[28] “确定网络基础设施安全合规性的方法和设备”，2024年7月。 [在线]。网址：https://ipsearch.ipaustralia.gov.au/patents/2025205598

[29] 20 2025 8 SANS 控制项。

（）CIS 控制项 v。访问日期：2025年11月17日。

[ ] 8 在线。

[30] 网址：https://www.sans.org/blog/cis-controls-v L. Espeholt、H. Soyer、R. Munos、K. Simonyan、V. Mnih、T. Ward、Y. Doron、V. Firoiu、T. Harley、I. Dunning 等，“IMPALA：采用重要性加权 actor-learner 架构的可扩展分布式深度强化学习”，载于国际机器学习会议。PMLR，2018年，第 1407–1416 页。[31] J. Schulman、F. Wolski、P. Dhariwal、A. Radford 和 O. Klimov，“近端策略优化算法”，arXiv 预印本 arXiv:1707.06347，2017年。[32] T. Schaul、J. Quan、I. Antonoglou 和 D. Silver，“优先经验回放”，arXiv 预印本 arXiv:1511.05952，2015年。[33] C. Hu、Y. Deng、W. Luo、Q. Wei 和 G. Min，“基于相关性的通用资源匹配策略：构建异构弹性云服务系统”，IEEE服务计算汇刊，第 17 卷，第 5 期，第 2931–2944 页，2024 年。[34] Y. Li 、 J. Shen 、 P. Vijayakumar、C.-F. Lai、A. Sivaraman 和 P. K. Sharma，“面向云边分布式弹性机器学习的新一代消费电子数据审计方案”，IEEE Transactions on Consumer Electronics，第 70 卷，第 1 期，第 2244–2256 页，2024年。[35] R. Mahmud、S. Pallewatta、M. Goudarzi 和 R. Buyya，“iFogSi m2：面向边缘与雾计算环境中移动性、聚类和微服务管理的扩展版 iFogSim 模拟器”，Journal of Systems and Software，第 190 卷，第 111351 页，2022年。[36] A. Nahar 等，“道路上的云：用于车载自组网络智能资源管理的软件定义雾计算框架”，IEEE Transactions on Mobile Computing，第 23 卷，第 12 期，第 12 778–12 792 页，2024年。[37] M. Goudarzi、M. A. Rodriguez、M. Sarvi 和 R. Buyya，“μ-分布式深度强化学习：一种面向雾计算环境中服务卸载的服务质量感知分布式深度强化学习技术”，IEEE服务计算汇刊，2023年。[38] Z. Wang、M. Goudarzi 和 R. Buyya，“ReinFog：一种由深度强化学习驱动的边缘与云计算环境资源管理框架”，Journal of Network and Computer Applications，第 104250 页，2025年。[39] X. Zhang、M. Wang、X. Zhu、Z. Yan 和 G. Geng，“面向工业物联网的协同边缘云数据传输优化”，IEEE Transactions on Parallel and Distributed Systems，2025年。[40] X. Chen、Y. Zhang、C. Jiang、C. Xu、Z. Yuan 和 G.-M. Muntean，“面向软件定义广域网赋能制造网络的雾云协同收益导向最优服务卸载”，IEEE Transactions on Network Science and Engineering，2025年。[41] Z. Wang、M. Goudarzi 和 R. Buyya，“TF-DDRL：一种用于边缘与云计算环境中物联网应用调度的 Transformer 增强型分布式深度强化学习技术”，IEEE服务计算汇刊，2025年。[42] Y. Chen、S. Zhang、Y. Jin、Z. Qian、M. Xiao、J. Ge 和 S. Lu，“LOCUS：移动边缘计算环境中用户感知时延的服务放置与用户分配”，IEEE Transactions on Parallel and Distributed Systems，第 33 卷，第 7 期，第 1581–1592 页，2022年。[43] Z. Wang、M. Goudarzi、M. Gong 和 R. Buyya，“基于深度强化学习的调度：优化边缘与雾计算环境中的系统负载和响应时间”，Future Generation Computer Systems，第 152 卷，第 55–69 页，2024年。