无线系统中面向时延与能耗感知流分配的基于后决策状态的在线学习

Mahesh Ganesh Bhat，Shana Moothedath，*IEEE*高级会员，以及 Prasanna Chapor kar

arXiv:2601.03108v1 [eess.SP] 6 Jan 2026

摘要—**我们提出一种结构感知的强化学习（RL）方法，用于5G用户面功能（UPF）中的时延与能耗感知流分配。我们考虑一个动态系统，其中包含 K 个容量各异的异构用户面功能，用于处理 M 种流类型的随机到达，每种流类型都有不同的速率要求。我们将该系统建模为马尔可夫决策过程（MDP），以刻画流到达和离开（可能未知）的随机性，以及流分配对系统的影响。为解决这一问题，我们提出一种基于后决策状态（PDS）的值迭代算法，该算法利用MDP的内在结构。通过将动作控制的动态与外生因素分离，PDS能够实现更快的收敛和高效的自适应流分配，即使缺乏关于外生变量的统计知识也不例外。仿真结果表明，与标准Q学习相比，所提出的方法收敛更快，长期成本更低，凸显了基于PDS的强化学习在无线网络资源分配中的有效性。**

索引词—**资源分配、UPF 流分配、强化学习、能耗感知决策。**

一、引言

现代 5G 及后续网络（B5G）的发展，使各种服务需求各异的应用成为可能，包括高吞吐量宽带、对时延要求严苛的控制应用，以及大规模物联网连接。这导致网络流量不断增长，服务质量（QoS）需求也呈现异构且日益严苛的特点。为支持大量流量而快速扩建基础设施，进一步增加了能源消耗。因此，可持续性已成为现代网络设计的首要目标。ITU-R M.2160 建议书在“动机与社会考量”和“用户与应用趋势”条款中，将可持续性列为重点之一。这促使人们采用能耗感知的资源管理方式，并部署更绿色的网络。鉴于这一目标，尽管已有大量研究探讨无线接入网（RAN）中的资源分配与调度， [1], [2],但同样需要关注用户面功能（UPF），因为它在 5G 核心网中发挥着关键作用。

用户面功能是 5G核心网中的核心数据平面互连点，负责数据转发、流量路由以及执行服务质量策略。将流量分配到用户面功能会直接影响网络性能和可靠性。此外，用户面功能是部署在各种基础设施上的虚拟功能，其地理位置、计算能力、与用户的距离以及能源特征（运行所消耗的能源中来自绿色能源的比例）各不相同。这些因素会影响用户面功能的选择，也凸显了采用有原则的流分配方法，而非简单的数据转发方式的必要性。

设计智能且自适应的流分配策略，对于在下一代移动网络中实现可扩展、对时延敏感且资源高效的性能至关重要 [3]。流分配问题与任务调度问题具有相似特征，在计算与通信系统中已得到广泛研究。早期研究主要针对静态（优化版本）的分配问题 [4], [5], ，因此难以适应动态且不确定的环境。强化学习（RL）提供了一个有前景的框架，可通过在动态且不确定的网络环境中实现数据驱动的自适应决策，解决流分配这一挑战 [6], [7]。与传统方法不同，RL 能通过与系统交互来学习最优策略，同时考虑随机到达、离开量和系统约束，因此尤其适用于下一代网络中对时延敏感且资源受限的现代应用。例如， [8]提出了一种结合 RL 和随机梯度下降的混合方法，用于处理具有随机任务到达的移动边缘计算（MEC）卸载调度。然而，该方法只考虑了单个边缘云。另有研究提出了一种深度强化学习（DRL）方法，用于卫星部署的用户面功能中的时延与资源感知服务优化 [3] ，但其考虑的是固定的到达量。文献 [9], [10] 提出了基于 DRL 的资源分配方案，以降低 MEC 中的时延。基于 DRL 的方法面临两大局限：它们学习到的是近似策略而非最优策略，且缺乏形式化保证。此外，这些方法依赖函数近似，通常需要大量训练数据。而且，所有这些研究都假设任务到达与离开是固定且已知的。然而，任务到达与离开往往是外生且未知的。本研究旨在利用问题结构，开发高效且最优的强化学习算法，实现快速且样本高效的学习。

分配和调度问题固有的结构性质使得能够开展基于后决策状态（PDS）的强化学习分析。PDS分析已应用于调度和分配问题。针对能效点对点通信中的时延敏感型无线传输调度以及加速学习，已有研究探讨了PDS [11], [12]。文献 [13], 提出了一种基于PDS的在线学习方法，用于数据中心的服务器分配，旨在降低电力成本。文献 [14], 研究了虚拟化无线接入网中的无线资源调度，其中到达量和离开量具有随机性，且重点关注移动设备效用；该研究将一个多智能体马尔可夫决策过程分解为多个单智能体马尔可夫决策过程，

M. G. Bhat 和 S. Moothedath 任职于爱荷华州立大学电气与计算机工程系。电子邮箱：{mgbhat,mshana}@iastate.edu。P. Chaporkar 任职于印度理工学院孟买分校电气工程系。电子邮箱：chaporkar@ee.iitb.ac.in。本研究由美国国家科学基金会（NSF-CNS 2415213）和印度电子和信息技术部（MeitY）资助。

并提出了一种基于后决策状态的在线局部化算法。文献 [15], [16]研究了在物联网应用中使用PDS的基于强化学习和深度强化学习的隐私感知卸载方法。

尽管已有若干研究探讨了无线移动网络中的流调度问题，但这些研究依赖大量训练数据、已知的系统动态或固定的业务流。不同于这些研究，本文关注在流到达量和离开量未知的情况下，对多流进行动态分配建模。本文提出一种基于强化学习的框架，用于在具有多个用户面功能和异构业务类别的5G网络中动态分配流。我们将该问题建模为马尔可夫决策过程（MDP），以刻画网络流量的随机性，并提出一种新颖的强化学习算法，在MDP的转移概率未知时学习最优分配策略。我们旨在利用结构性质，通过基于后决策状态的强化学习来求解时延与能耗感知流分配问题，同时考虑容量约束。所提出的方法能够实现实时自适应决策，从而克服高度动态环境下静态或基于模型的技术的局限，以优化性能。据我们所知，此前尚未有研究探讨具有随机（未知）到达量和离开量的多服务器场景下的这一问题。

本工作的主要贡献有三点。

- 我们将多流动态分配问题建模为时延与能耗感知的马尔可夫决策过程，在固定容量约束下刻画随机业务到达与离开，从而实现自适应且有原则的分配。
- 我们提出一种基于后决策状态（PDS）的值迭代（VI）算法，利用问题结构，特别是外生动态与可控动态之间的分离，简化学习、提高样本效率并加速收敛。我们还证明了所提出的PDS-VI算法的收敛性。
- 我们通过数值仿真评估了所提出方法的性能，结果表明，与传统基线相比，该方法收敛更快且成本性能更优。

II. 预备知识：马尔可夫决策过程

马尔可夫决策过程（MDP）是对随机动力系统建模并计算其最优控制策略 [17] 的标准框架。设 $\mathcal{S}$ 和 $\mathcal{A}$ 分别为描述控制器（智能体）状态和动作的紧集，$r:\mathcal{S}\times\mathcal{A}\to\mathbb{R}$ 为奖励函数。系统动态由概率转移结构 $\mathbb{P}$ 表征，其中 $\mathbb{P}(s'|s,\ a)$ 表示在控制动作 a 下从状态 s 转移到状态 s' 的概率。策略$\pi:\mathcal{S}\to\mathcal{A}$ 是条件分布 $\pi(a|s)$，用于指导智能体的决策过程。在时刻 $t\in\{0,\ 1,\ \ldots,\ \}$，智能体观察当前状态 s_t，并根据策略 $\pi(a|s)$ 选择一个动作a_t^π，然后观察奖励 $r(s_t,\ a_t^\pi)$。智能体在某个状态下选择的动作会以概率（w.p.） $\mathbb{P}(s_{t+1}|s_t,\ a_t^\pi)$使其转移到下一个状态。智能体的目标是找到一个最优策略 $\pi^\star$，使累积奖励 $\lim_{T\to\infty}\sum_{t=0}^{T}\gamma^t r(s_t,\ a_t^\pi)$ 最大化，其中 $\gamma\in(0,\ 1]$ 是折扣因子，用于刻画智能体的短视程度。

III. 问题建模：5G用户面功能中的时延敏感型流分配

A. 系统模型

考虑一个 5G 网络。时间被划分为时隙。每个时隙开始时，网络中以概率 p 到达一个新流。每个到达的流可以属于 M 种类型之一。令$\mathbf{b}_m$ 表示到达流属于类型 m 的概率，其中 $m\in[M]$，$[Z]:=\{1,\ 2,\ \ldots,\ Z\}$，且 Z 为任意整数。我们还假设不同流的到达及其类型相互独立。流类型表示其平均流速率需求。令 R_m 表示类型 m流的平均速率要求。不失一般性，对每个 $m\in[M-1]$，$R_m<R_{m+1}$。对于每个到达的流，网络必须决定是否接纳该流。具体而言，如果无法保证其所需速率，则拒绝接纳该流；否则予以接纳。

如果网络接受某个流，则会为其分配一个用户面功能，由该功能负责处理该流，直至其离开，并根据流的类型为其提供所需的数据速率。假设网络中有 K 个用户面功能。每个用户面功能在可用内存、计算速度和交换速度方面的能力可能各不相同。根据这些能力，令 C_k 表示用户面功能 $k\in[K]$ 可支持的最大数据速率（比特/秒）。最后，在每个时隙结束时，至多有一个流以概率q_k从用户面功能 k^{th} 离开，其中 $k\in[K]$；各时隙中的流离开相互独立。离开的流以相同概率为该用户面功能中的任意现有流。在某时隙到达的流也可能在同一时隙离开。下面我们将动态UPF分配问题建模为马尔可夫决策过程。

B. 马尔可夫决策过程建模

5G网络中的流分配本质上是一个动态决策问题。马尔可夫决策过程为刻画此类动态过程并优化长期性能提供了系统化框架。本节将用户面功能（UPF）中对时延和能耗敏感的流分配问题建模为马尔可夫决策过程，并详细说明其关键组成部分。

1）状态空间：我们将系统状态定义为一个包含分配矩阵 $\boldsymbol{n}$ 和流到达状态f的元组，即 $s=(\boldsymbol{n},\ f)$。对于状态 $s=(\boldsymbol{n},\ f)$，令 $\boldsymbol{n}$为一个 $K\times M$矩阵，其中 $(k,\ m)^{\text{th}}$项 $\boldsymbol{n}_{km}$表示在 s 由 k^{th} 用户面功能（UPF）处理的类型为 m 的流的数量，而 $f\in\{0,\ 1,\ ,\ .....,\ M\}$ 表示在 s 到达的流的类型。这里，$f=0$ 表示 s 没有流到达。令 $R_k(s)$ 表示用户面功能（UPF） k 在 s需要支持的总平均速率。注意，

$$\tilde{R}_k(s)=\sum_{m\in[M]}\boldsymbol{n}_{km}\bar{R}_m. \tag{1}$$

定义状态空间 $\mathcal{S}$ 如下：

$$\mathcal{S}=\left\{s: C_k>\tilde{R}_k(s)\ \text{for all}\ k\in[K], f\in\{0,\ldots,M\}\right\}.$$

注意，$\lfloor C_k/R_1\rfloor$ 是用户面功能k 在任意时刻能够支持的最大流数。因此，$\mathcal{S}$ 是一个有限集。

2）动作空间与控制策略：考虑一个状态 $s=(\boldsymbol{n},\ f)$。设 $\mathcal{A}(s)$ 为 s中的可行动作集合。那么 $\mathcal{A}(s)\subseteq\{0,\ 1、\ldots\ldots、\ K\}$，其中，若 $C_k>R_k(s)+R_f$，则 $k\in\mathcal{A}(s)$，对于 $k\in[K]$。对于 $a\in\mathcal{A}(s),\ \ a\in[K]$表示将流 f 分配给用户面功能 a，而 $a=0$ 表示该流被阻塞。只要满足 $\mathcal{A}(s)\neq\{0\}$，就必须接纳 s中的流 f，并由分配给它的用户面功能处理，直至其离开。动作集合的维度为 $K+1$。对于 $a\in\mathcal{A}(s)$，为便于分析，我们定义一个$K\times M$ 指示矩阵 $\boldsymbol{a}$。

$$\boldsymbol{a}_{km}=\begin{cases}1, & \text{if } a=k \text{ and } f=m,\\ 0, & \text{otherwise.}\end{cases}\tag{2}$$

因此，$\boldsymbol{a}$是一个稀疏指示矩阵，其中至多有一个非零元素（值为 1），其余元素均为 0。

控制策略 $\pi:\ \mathcal{S}\to\mathcal{A}$ 将状态映射到可行动作。我们假设 π 是因果的，即为状态 s 选择的动作可以依赖于过去的状态和已采取的动作，但不能依赖于未来的演变。此外，π 也可以是随机策略，即对于某个状态，可以从可行动作集合中随机选择一个动作。

3）成本函数：令 $\xi(s)$ 表示状态 s下产生的总成本。我们考虑

$$\xi(s)=\sum_{k\in[K]}(\alpha_k(s)+\delta_k(s)),\tag{3}$$

其中，$\alpha_k(s)$表示功耗成本，$\delta_k(s)$表示状态 s下 UPF k 的时延成本。这些成本定义为

$$\alpha_k(s)=c_k\tilde{R}_k(s)\text{ and }\delta_k(s)=\frac{C_k}{C_k-\tilde{R}_k(s)}.\tag{4}$$

这里，c_k 表示在 UPF k切换一位所产生的功耗成本。

4）系统动态：系统动态由概率转移矩阵 $\mathbb{P}$刻画，其中 $\mathbb{P}(s'|s,\ \boldsymbol{a})$ 表示在控制动作 $\boldsymbol{a}$下，从状态 s转移到状态 s' 的概率。考虑当前状态 $s=(\boldsymbol{n},\ f)$以及下一状态 $s'=(\boldsymbol{n}',\ f')$，其中 $\boldsymbol{n},\ \boldsymbol{n}'$表示流分配矩阵，$f,\ f'$表示流到达。

转移动态由两个部分组成：(i) 分配矩阵 $\boldsymbol{n}$ 的演变，它取决于当前状态、控制动作和随机离开；(ii) 到达过程 f，它是外生的，且独立于控制动作，遵循一个固定但未知的概率分布。形式上，转移概率

$$\mathbb{P}(s'|s,\boldsymbol{a})=\mathbb{P}(\boldsymbol{n}'|s,\boldsymbol{a})\cdot\mathbb{P}(f'),\tag{5}$$

其中，到达概率为

$$\mathbb{P}(f')=(1-p)\mathbb{1}_{\{f'=0\}}+p\sum_{m=1}^{M}b_m\mathbb{1}_{\{f'=m\}}.\tag{6}$$

分配矩阵 $\boldsymbol{n}$ 的演化取决于离开过程。为了定义 $\mathbb{P}(\boldsymbol{n}'|s,\ \boldsymbol{a})$，我们首先定义离开过程。设 $\boldsymbol{n}_k$ 和 $\boldsymbol{n}'_k$ 分别为当前状态 s 和下一状态 s'下，分配矩阵中对应于用户面功能 k 的行向量。定义 $\mathbb{R}^M$ 中的标准向量如下：

$$\boldsymbol{e}_m\in\{0,1\}^M,\qquad(\boldsymbol{e}_m)_j=\begin{cases}1, & \text{if } j=m,\\ 0, & \text{if } j\neq m\end{cases}$$

考虑一种类型为 m 的流离开的情形。令 $\mathcal{D}_k(\boldsymbol{n}'_k,\ \boldsymbol{n}_k,\ f)$ 为用户面功能 k的转移概率，即在到达 f的情况下，从 $\boldsymbol{n}_k$ 转移到 $\boldsymbol{n}'_k$ 的概率。我们先考虑没有新流到达 s，或者有类型为 $\tilde{m}$ 的流到达但 $\mathcal{A}(s)$ 为空的情况。此时，要么 $\boldsymbol{n}=\boldsymbol{n}'$，要么分配矩阵由于流离开而转移到一个不同的状态。于是，

$$\mathcal{D}_k(\boldsymbol{n}'_k,\boldsymbol{n}_k,f)=\begin{cases}1-q_k, & \text{if } \boldsymbol{n}'_k=\boldsymbol{n}_k\\ q_k\cdot\frac{\boldsymbol{n}_{km}}{\sum_{m'}\boldsymbol{n}_{km'}}, & \text{if } \boldsymbol{n}'_k=\boldsymbol{n}_k-\boldsymbol{e}_m\\ 1, & \text{if } \boldsymbol{n}'_k=\boldsymbol{n}_k=\boldsymbol{0}\\ 0, & \text{otherwise.}\end{cases}\tag{7}$$

现在考虑一种情况：在状态 s 下，有一个类型为 $\tilde{m}$ 的新到达，且 $\mathcal{A}(s)$ 非空。不失一般性，令 $a=k'$。有两种情况。**情况 1：** 对于任意用户面功能 $k\neq k'$，此时的离开概率与式 (7) 相同。**情况 2：** 对于用户面功能 k'。则，

$$\mathcal{D}_{k'}(\boldsymbol{n}'_{k'},\boldsymbol{n}_{k'},f)=\begin{cases}1-q_{k'}, & \text{if } \boldsymbol{n}'_{k'}=\boldsymbol{n}_{k'}+\boldsymbol{e}_{\tilde{m}}\\ q_{k'}\cdot\frac{\boldsymbol{n}_{k'\tilde{m}}+1}{\sum_{m'}\boldsymbol{n}_{k'm'}+1}, & \text{if } \boldsymbol{n}'_{k'}=\boldsymbol{n}_{k'}\\ q_{k'}\cdot\frac{\boldsymbol{n}_{k'm}}{\sum_{m'}\boldsymbol{n}_{k'm'}+1}, & \text{if } \boldsymbol{n}'_{k'}=\boldsymbol{n}_{k'}+\boldsymbol{e}_{\tilde{m}}\\ & -\boldsymbol{e}_m \text{ and } m\neq\tilde{m}\\ 0, & \text{otherwise.}\end{cases}$$

转移概率

$$\mathbb{P}(\boldsymbol{n}'|s,\boldsymbol{a})=\prod_{k=1}^{K}\mathcal{D}_k(\boldsymbol{n}_k,\boldsymbol{n}'_k,f).\tag{8}$$

式 (5)、(6)、(8) 完成了对系统动态的建模。

5) 最优策略：设 $s^\pi(t)$ 为策略 π 下时隙 t 的状态。对于任意函数 $g:\ \mathcal{S}\to\mathbb{R}$，我们用简写 $\tilde{}\ g^\pi(t)$ 表示 $g(s^\pi(t))$。例如，$R_k^\pi(t)$ 表示在时刻 t、采用策略 $\tilde{}\ \pi$ 时，用户面功能 k 必须支持的总速率，即 $R_k(s^\pi(t))$；而 $\xi^\pi(t)$ 表示时刻 t 产生的成本，即 $\xi(s^\pi(t))$。我们的目标是学习使累积折扣奖励最小化的最优策略 $\pi^\star$

$$\xi^\pi=\lim_{T\to\infty}\sum_{t=0}^{T}\gamma^t\,\xi^\pi(t).$$

IV. 利用强化学习方法的拟议结构

在实际场景中，到达概率（$p,\ \mathbf{b}_m$）和离开概率（q_k）通常未知，因此经典动态规划方法不适用。为解决这一问题，我们提出一种无模型强化学习算法。

A. 基于决策后状态的学习

考虑从当前状态 $s=(\boldsymbol{n},\ f)$ 到下一状态 $s'=(\boldsymbol{n}',\ f')$的状态转移。根据式（5），我们知道转移动态由两个部分组成。这种分解有助于设计能够有效处理环境部分知识的强化学习算法。

令 $\tilde{s}=(\tilde{\boldsymbol{n}},f)$ 表示采取动作后、随机到达与离开产生影响之前的虚拟状态，这称为后决策状态。这里，$\tilde{\boldsymbol{n}}$ 是相应的分配矩阵。给定该虚拟状态 $\tilde{s}$，我们可以将式（8）改写为

$$\mathbb{P}(\boldsymbol{n}'|s,\boldsymbol{a})=\mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}})\cdot\mathbb{P}(\tilde{\boldsymbol{n}}|s,\boldsymbol{a}).$$

这使我们能够将式（5）中的系统动态分解为动作控制（已知）的 $\mathbb{P}^k(\cdot|\cdot,\cdot)$ 转移，以及随机（未知）的 $\mathbb{P}^u(\cdot|\cdot,\cdot)$ 转移。

$$\mathbb{P}(s'|s,\boldsymbol{a}) = \underbrace{\mathbb{P}(\tilde{\boldsymbol{n}}|s,\boldsymbol{a})}_{\substack{\text{action controlled}\\ \mathbb{P}^k(\tilde{s}|s,\boldsymbol{a})}} \cdot \underbrace{\mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}})\cdot\mathbb{P}(f')}_{\substack{\text{exogenous}\\ \mathbb{P}^u(s'|\tilde{s},\boldsymbol{a})}}. \tag{9}$$

分配矩阵在纯动作控制下的演化是确定性的，即对于 $\tilde{s}=(\tilde{\boldsymbol{n}},\ f)$,

$$\mathbb{P}^k(\tilde{s}|s,\boldsymbol{a}) = \mathbb{P}(\tilde{\boldsymbol{n}}|s,\boldsymbol{a}) = 1. \tag{10}$$

系统动态中的不确定性主要源于随机到达和离开过程，从而能够采用结构化且灵活的建模方法进行基于学习的控制。我们可以利用这些结构特性，发挥决策后状态分析的潜力，以降低复杂度 [18], [19]。下一节将介绍一种基于部分可观测决策状态的强化学习算法，以便在部分已知的系统动态下高效学习最优策略。

B. 所提算法：基于*PDS*的价值迭代

设 $s=(\boldsymbol{n},f)$ 为某个时隙中的系统状态。设 $a\in\mathcal{A}(s)$ 为在状态 s 下选择的动作，$\boldsymbol{a}$ 为相应的矩阵。则由 $\tilde{s}$ 表示的后决策状态为 $\tilde{s}=(\tilde{\boldsymbol{n}},f)=(\boldsymbol{n}+\boldsymbol{a},f)$。设 $\boldsymbol{u}$ 为在该时隙结束时观测到的离开量 $K\times M$矩阵，其行向量 $\boldsymbol{u}_k$、$k\in[K]$ 由下式给出

$$\boldsymbol{u}_k = \begin{cases} \boldsymbol{e}_m, & \text{flow of type } m \text{ departs from UPF } k \\ \boldsymbol{0}, & \text{if no departures.} \end{cases} \tag{11}$$

考虑随机到达与离开后得到的下一个实际状态称为决策前状态，$s'=(\boldsymbol{n}',f')$，其中 $\boldsymbol{n}'=\tilde{\boldsymbol{n}}-\boldsymbol{u}=\boldsymbol{n}+\boldsymbol{a}-\boldsymbol{u}$。考虑PDS的状态转移序列如图1所示。

我们的马尔可夫决策过程的贝尔曼方程可写为：

$$V(s)=\min_{a\in\mathcal{A}(s)}\left\{\xi(s)+\gamma\sum_{s'}\mathbb{P}(s'|s,\boldsymbol{a})\cdot V(s')\right\}. \tag{12}$$

利用公式（9）中的决策后状态转移结构，并代入 s'，可 (10)

将公式（12）重写为

$$V(s)=\min_{a\in\mathcal{A}(s)}\left\{\xi(s)+\gamma\sum_{f',\boldsymbol{u}}\mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}})\cdot\mathbb{P}(f')\cdot V(\tilde{\boldsymbol{n}}-\boldsymbol{u},f')\right\}.$$

~现在我们定义决策后状态价值函数 $V:\mathcal{S}\to\mathbb{R}$ 它是从该决策后状态可达的所有决策前状态 s'的期望价值，

$$\begin{aligned}\widetilde{V}(\tilde{s}) &= \mathbb{E}_{s'}[V(s')]\\ \widetilde{V}(\tilde{s}) &= \sum_{s'}\mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}})\cdot\mathbb{P}(f')\cdot V(s').\end{aligned} \tag{13}$$

根据此定义，贝尔曼方程变为

$$V(s)=\min_{a\in\mathcal{A}(s)}\left\{\xi(s)+\gamma\widetilde{V}(\tilde{s})\right\}. \tag{14}$$

令 $a'\in\mathcal{A}(s')$ 表示在状态 s' 下选择的动作，$\boldsymbol{a}'$ 表示相应的指示矩阵。另外，令下一个决策后

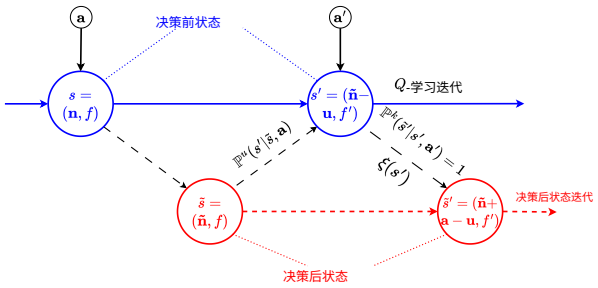


图1：基于PDS的状态转移示意图 ion.

算法1 基于决策后状态的值迭代

输入： 用户面功能数量 K，流类型数量 M，重新-~源需求 R_m、容量 C_k、单位成本 c_k

输出： 策略 π

~1: 初始化后状态 $\tilde{s}=(\tilde{\boldsymbol{n}},\ f)$，$\tilde{s}\in\mathcal{S}$、$V\leftarrow\boldsymbol{0}$ 和 $t=0$

2: **对于** $t=0,1,2,3,\cdots$ **执行**

3: 观察离开量并计算离开矩阵 $\boldsymbol{u}$ 使用公式 (11) //离开量

4: 观测流到达 f' // 到达

5: 计算 $s'=(\tilde{\boldsymbol{n}}-\boldsymbol{u},\ f')$

6: 确定可行动作，$\mathcal{A}(s')$

7: ~使用公式（16）更新 $V_{t+1}(\tilde{s})$ 并获得 $\boldsymbol{a}'$ 相关-与最小化动作 $a'\in\mathcal{A}(s')$ 相对应

8: 计算 $\tilde{s}'=(\tilde{\boldsymbol{n}}-\boldsymbol{u}+\boldsymbol{a}',\ f')$

9: 更新 $\pi(s')=a'$ 和 $\tilde{s}=\tilde{s}'$

10: **结束循环**

状态（在 s' 下选择 a' 后）为 $\tilde{s}'=(\boldsymbol{n}'+\boldsymbol{a}',f')$。状态 s' 的价值方程可写为

$$V(s')=\min_{a'\in\mathcal{A}(s')}\left\{\xi(\boldsymbol{n}',f')+\gamma\widetilde{V}(\boldsymbol{n}'+\boldsymbol{a}',f')\right\}.$$

将其代入公式（13），得到构成所提出的价值迭代算法基础的方程，

$$\widetilde{V}(\tilde{s})=\sum_{f',\boldsymbol{u}}\mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}})\cdot\mathbb{P}(f')\min_{a'\in\mathcal{A}(s')}\Big[\xi(\tilde{\boldsymbol{n}}-\boldsymbol{u},f')+\gamma\widetilde{V}(\tilde{\boldsymbol{n}}-\boldsymbol{u}+\boldsymbol{a}',f')\Big]. \tag{15}$$

备注 1. 与标准贝尔曼方程相比，利用该公式的结构性质，公式（15）中的期望运算位于最小化运算之外。这使我们能够提出一种基于随机逼近的价值学习方法。

注 2. 式 (15)是 *PDS* ~价值函数 V的贝尔曼方程。将 *PDS* 分析 $V^\star$所对应的最优价值代入式(14)，即可得到式 (12)的最优价值向量 $V^\star$。进而，也可得到最优策略 $\pi^\star$。

接下来，我们将说明如何根据公式（15）得到一种学习算法，即基于PDS的价值迭代（PDS-VI）。令 $\{\alpha_t\}_{t\geqslant 0}$ 为一个正的步长序列，满足 Robbins-Monro 条件 $\sum_t\alpha_t=\infty$ 和 $\sum_t(\alpha_t)^2<\infty$。利用这些步长，我们进行随机逼近

价值向量，并根据以下更新规则，在每个时间步根据观测到的到达量和离开量更新每个状态的估计值

$$\widetilde{V}_{t+1}(\tilde{s}) = \widetilde{V}_t(\tilde{s}) + \alpha_t \Bigg[\min_{a' \in \mathcal{A}(s')} \Big(\xi(\tilde{\boldsymbol{n}} - \boldsymbol{u}, f') + \gamma \widetilde{V}_t(\tilde{\boldsymbol{n}} - \boldsymbol{u} + \boldsymbol{a}', f') \Big) - \widetilde{V}_t(\tilde{s}) \Bigg], \tag{16}$$

$$\widetilde{V}_{t+1}(\tilde{s}'') = \widetilde{V}_t(\tilde{s}''), \qquad \text{for all } \tilde{s}'' \neq \tilde{s}.$$

算法1给出了基于给定更新进行价值估计的迭代算法。下面我们证明，这些迭代结果会收敛到~PDS价值向量 $V^\star$ 的最优值。

定理1. 公式~ ~中的*PDS*价值函数迭代会收敛到最优 (16) *PDS*价值函数 $V_t \to V^\star$。

证明. 令 $\mathcal{T}$: $\mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ 定义如下:

$$\mathcal{T}(\widetilde{V})(\tilde{s}) = \sum_{f', \boldsymbol{u}} \mathbb{P}(\boldsymbol{n}'|\tilde{\boldsymbol{n}}) \cdot \mathbb{P}(f') \min_{a' \in \mathcal{A}(s')} \Big[\xi(\tilde{\boldsymbol{n}} - \boldsymbol{u}, f') + \gamma \widetilde{V}(\tilde{\boldsymbol{n}} - \boldsymbol{u} + \boldsymbol{a}', f') \Big]. \tag{17}$$

^将 $\mathcal{T}$ 定义为在时间 t针对某个观测到的 f' 和$\boldsymbol{u}$ 的采样算子，则

$$\widehat{\mathcal{T}}(\widetilde{V}_t)(\tilde{s}) = \min_{a' \in \mathcal{A}(s')} \Big[\xi(\tilde{\boldsymbol{n}} - \boldsymbol{u}, f') + \gamma \widetilde{V}_t(\tilde{\boldsymbol{n}} - \boldsymbol{u} + \boldsymbol{a}', f') \Big].$$

^我们可以用 $\mathcal{T}$和 $\mathcal{T}$改写公式（16），得到

$$\widetilde{V}_{t+1}(\tilde{s}) = \widetilde{V}_t(\tilde{s}) + \alpha_t \left(\mathcal{T}(\widetilde{V}_t)(\tilde{s}) - \widetilde{V}_t(\tilde{s}) + \epsilon_{t+1}(\tilde{s}) \right),$$

^ ~ ~其中，$\epsilon_{t+1}(\tilde{s}) = \mathcal{T}(V_t)(\tilde{s}) - \mathcal{T}(V_t)(\tilde{s})$。重新整理可得,

$$\widetilde{V}_{t+1}(\tilde{s}) - \widetilde{V}_t(\tilde{s}) = \alpha_t \left(\mathcal{T}(\widetilde{V}_t)(\tilde{s}) - \widetilde{V}_t(\tilde{s}) + \epsilon_{t+1}(\tilde{s}) \right). \tag{18}$$

令 $\mathcal{F}_t = \{(s_{t'}, a_{t'}, \xi_{t'}, s'_{t'})\}_{0 \leqslant t' \leqslant t}$ 表示截至时间 t的信息。由此可知，$\{\epsilon_{t+1}(\tilde{s})\}_{t \geqslant 0}$ 构成一个鞅差序列，且 $\mathbb{E}[\epsilon_{t+1}(\tilde{s})|\mathcal{F}_t] = 0$。式（18）中的迭代等价于常微分方程（ODE） $V = \mathcal{T}(V) - V$. 的离散化版本。由[20], 可以论证，这些更新的收敛等价于上述常微分方程的收敛。此外， $\mathcal{T}$在上确界范数下是一个 γ-压缩映射，即:

$$||\mathcal{T}(f) - \mathcal{T}(g)||_\infty \leqslant \gamma ||f - g||_\infty \quad \forall f, g.$$

~因此，可以保证迭代 V_t 收敛到~最优值 $V^\star$。

□

注3. $\mathcal{S}$ 的基数上界为 $|\mathcal{S}| = (M+1) \cdot \prod_{k=1}^{K} (\lfloor C_k / \bar{R}_1 \rfloor)^M$，并且会随着 K 和 M呈指数级增长。*PDS-VI* 同时节省计算量和内存：*PDS-VI* 算法需要存储每个状态的价值估计，因此其复杂度为 $|\mathcal{S}|$; 而 *Q*学习 需要 $|\mathcal{S}| \times |\mathcal{A}|$ 来存储所有状态-动作对。*PDS-VI* 和 *Q*学习 每次迭代的计算复杂度均为 $|\mathcal{A}|$ ，然而，*PDS-VI* 收敛所需的迭代次数为 $|\mathcal{S}|$ 量级，而 *Q*学习 所需的迭代次数为 $|\mathcal{S}| \times |\mathcal{A}|$量级。

注4。在我们提出的模型中应用*PDS*分析，通过利用标准*PDS*分解之外的结构性质，加快了收敛速度。在我们的模型中，*PDS*值独立于当前流量。因此， $|M|+1$ 状态可以同时更新，从而进一步提高收敛速度。

V. 数值仿真

数据生成： 我们在一个包含 5 个用户面功能和 2 种流类型的合成数据集上评估了所提出的算法。对于所有 $k \in [K]$，流到达量按照概率为 $p = 0.7$ 的伯努利分布进行采样，流离开量则根据 $q_k = 0.3$ 动态采样。它们的概率分别为 $b_1 = 0.6$ 和 $b_2 = 0.4$。每种流类型的平均速率¯ ¯需求设为 $R_1 = 30$ 和 $R_2 = 35$。每个用户面功能的最大数据速率 C_k 设为 100。我们假设折扣因子为 0.96。状态总数为 98304。功耗成本为 $c_1 = 5, c_2 = 4, c_3 = 3, c_4 = 2$和 $c_5 = 1$。仿真运行 4.5×10^6 个时隙，其中3.5×10^6 用于训练， 1×10^6 用于评估。

基线与指标： 我们通过值迭代（动态规划）获得真实值。值迭代的更新方程为 $V_{t+1}(s) = \xi(s) + \min_{\boldsymbol{a} \in \mathcal{A}} \gamma \sum_{s'} \mathbb{P}(s'|s, \boldsymbol{a}) V_t(s) \quad \forall s \in \mathcal{S}.$。该值更新利用了转移动态的知识，因此可作为真实值基线，用于评估性能指标。我们根据收敛速度和平均奖励对算法进行比较。我们将算法的时间平均成本作为

$$\bar{\xi}_t = \frac{1}{t} \sum_{t=1}^{N} \xi(s_t),$$

成本性能指标，并以相对贝尔曼误差衡量收敛情况。由于不同状态的访问频率各不相同，我们选择采用加权误差，定义如下:

$$RBE = \frac{\sum_s w_s (|V(s) - V^\star(s)|)}{\sum_s w_s (|V^\star|(s))},$$

其中， w_s 是状态 s 被访问的次数， $V^\star$是真实值。我们还将各算法丢弃的流数量作为一项评估指标进行比较。

结果与讨论： 通过Q学习，我们使用 $V(s) = \min_a Q[s, a]$获得价值函数。它作为无模型学习的基线。我们的目标是证明，所提出的PDS方法能够以更高的效率（即使用更少的数据样本和更短的计算时间）实现与Q学习相当的性能。我们将成本评估分为两部分：训练期间产生的成本，以及训练后对学习得到的策略进行的简短评估。我们使用Python实现了所提出的算法。仿真在搭载*Intel(R) Xeon(R) W-1370*处理器的Windows计算机上运行。图2中的仿真结果为1000次独立蒙特卡洛运行的平均值。

如图 2a 所示，PDS-VI 的收敛速度快得多，在 7.5×10^5 次迭代内即可达到 90% 的准确率，而 Q学习 约需 3.5×10^6 次迭代。正如预期，

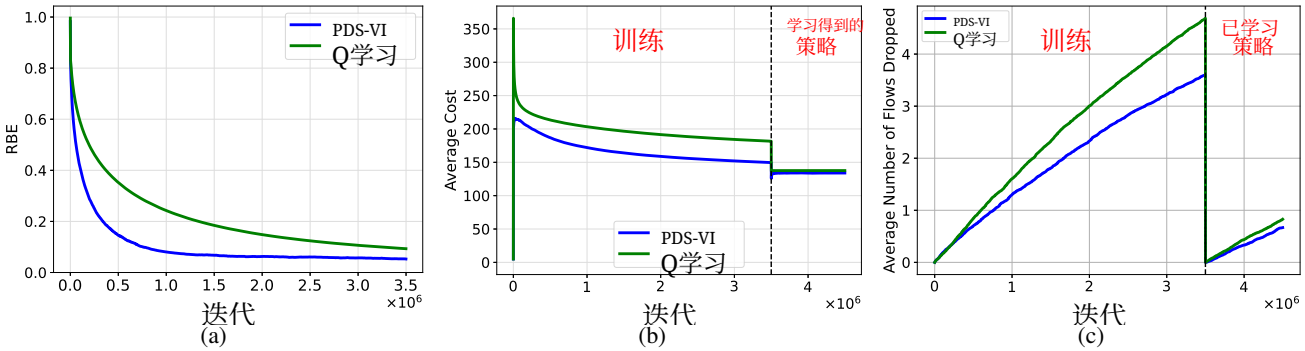


图 2：基于PDS的价值迭代与 Q学习的比较。图 2a 展示了平均相对贝尔曼误差（RBE）的曲线，图 2b 展示了平均成本，而图 2c 表示 5 个 UPF 情况下被阻塞的流数量随迭代次数的变化。

由于收敛速度更快，如图 2b 所示，PDS-VI 在训练期间的成本低于 Q学习。在使用学习到的策略进行策略评估时，两种方法的表现相近，但 PDS-VI 始终略胜一筹。图 2c 显示，与 Q学习 相比，PDS-VI 在训练和评估期间阻塞的流更少。因此，PDS-VI 凭借更快的收敛速度、更低的训练成本以及接纳更多流，明显优于 Q学习。

备注 5. *3GPP TS 23.501* 中定义的协议数据单元（*PDU*）会话锚点 *UPF*，会维护 *PDU* 会话上下文，在移动性变化期间提供会话连续性，而服务于该流的其他 *UPF* 可能会发生变化。会话管理功能（*SMF*）会根据包括地理服务区域限制在内的多种因素选择此锚点 *UPF*。服务于同一网络的 *UPF* 通常部署在地理位置集中的小型集群中。因此，锚点 *UPF* 的实际选择范围仅限于一小部分 $KUPF$，其中 K 通常是一个较小的数值。

VI. 结论

本文研究了资源约束下无线系统中的时延与能耗感知流分配问题。我们将该问题建模为一个马尔可夫决策过程，并提出了一种基于后决策状态（PDS）学习的无模型强化学习算法。通过利用系统动态的可分解结构，即将可控因素与外生因素分离，我们的方法能够实现高效学习并加快收敛速度。我们证明了所提算法的收敛性，并将其性能与标准Q-learning算法进行了比较，从而验证了其有效性。

参考文献

[1] X. Wu、J. Farooq 和 J. Chen，“O-RAN中URLLC业务的联合接纳控制与资源配置：一种受约束多智能体强化学习方法”，载于*IEEE*国际通信会议（*ICC*），2025年，第1426–1431页。[2] A. Vatankhah 和 R. Liscano，“QoS感知的节能时隙信道跳频调度算法”，载于*IEEE*国际通信会议（*ICC*），2025年，第3538–3544页。[3] C. Wang、X. Ma、R. Xing、S. Li、A. Zhou 和 S. Wang，“面向时延和资源的卫星UPF服务优化”，*IEEE*移动计算汇刊，2024年。[4] T. D. Braun、H. J. Siegel、A. A. Maciejewski 和 Y. Hong，“具有依赖关系、优先级、截止期限和多个版本任务的异构计算环境静态资源分配”，《并行与分布式计算杂志》，第68卷，第11期，第1504–1516页，2008年。

[5] Z. Han 和 K. R. Liu，无线网络资源分配：基础、技术与应用。剑桥大学出版社，2008年。[6] Y. S. Nasir 和 D. Guo，“无线网络中的动态功率分配多智能体深度强化学习”，*IEEE*通信选定领域期刊，第37卷，第10期，第2239–2250页，2019年。[7] F. Tang、Y. Zhou 和 N. Kato，“高移动性5G异构网络中的动态上行/下行资源分配深度强化学习”，*IEEE*通信选定领域期刊，第38卷，第12期，第2773–2782页，2020年。[8] S. Huang、B. Lv、R. Wang 和 K. Huang，“具有用户随机到达的移动边缘计算调度——一种近似马尔可夫决策过程与强化学习方法”，*IEEE*车辆技术汇刊，第69卷，第7期，第7735–7750页，2020年。[9] J. Wang、L. Zhao、J. Liu 和 N. Kato，“移动边缘计算的智能资源分配：一种深度强化学习方法”，*IEEE*新兴计算主题汇刊，第9卷，第3期，第1529–1541页，2021年。[10] Y. Liu、H. Yu、S. Xie 和 Y. Zhang，“车载边缘计算与网络中的卸载和资源分配深度强化学习”，*IEEE*车辆技术汇刊，第68卷，第11期，第11 158–11 168页，2019年。[11] N. Mastronarde 和 M. van der Schaar，“面向能效无线通信的快速强化学习”，*IEEE*信号处理汇刊，第59卷，第12期，第6262–6266页，2011年。[12] F. Fu 和 M. van der Schaar，“无线多媒体传输动态调度的结构化解决方案”，*IEEE*视频技术电路与系统汇刊，第22卷，第5期，第727–739页，2012年。[13] J. Yang、S. Zhang、X. Wu、Y. Ran 和 H. Xi，“基于在线学习的服务器配置以降低数据中心电费”，*IEEE*控制系统技术汇刊，第25卷，第3期，第1044–1051页，2017年。[14] X. Chen、Z. Han、H. Zhang、G. Xue、Y. Xiao 和 M. Bennis，“利用随机学习进行虚拟化无线接入网中的无线资源调度”，*IEEE*移动计算汇刊，第17卷，第4期，第961–974页，2018年。[15] X. He、R. Jin 和 H. Dai，“面向隐私感知卸载的深度后决策状态学习：应用于MEC赋能的物联网”，*IEEE*物联网期刊，第6卷，第3期，第4547–4555页，2019年。[16] M. Min、X. Wan、L. Xiao、Y. Chen、M. Xia、D. Wu 和 H. Dai，“结合能量采集的医疗物联网基于学习的隐私感知卸载”，*IEEE*物联网期刊，第6卷，第3期，第4307–4316页，2019年。[17] R. S. Sutton 和 A. G. Barto，强化学习：导论。麻省理工学院出版社，2018年。[18] N. Salodkar、A. Bhorkar、A. Karandikar 和 V. S. Borkar，“衰落信道上面向能效和时延约束调度的在线学习算法”，*IEEE*通信选定领域期刊，第26卷，第4期，第732–742页，2008年。[19] J. Zhang、X. He 和 H. Dai，“面向智能物联网的盲后决策状态强化学习”，*IEEE*物联网期刊，第10卷，第12期，第10 605–10 620页，2023年。[20] V. S. Borkar 和 S. P. Meyn，“随机逼近与强化学习收敛的常微分方程方法”，《*SIAM*控制与优化杂志》，第38卷，第2期，第447–469页，2000年。