基于深度强化学习的分离式数据中心存储资源调度优化方法

袁政利[1]，郭少勇[1]，胡鑫[1]，仝杰[2]，郝佳恺[3]，Michel Kadoch[4]，喻鹏[1]
（1.北京邮电大学网络与交换技术全国重点实验室，北京 100876；2.中国电力科学研究院有限公司，北京 100192；
3.国网北京市电力公司，北京 100031；4.魁北克大学高等技术学院，蒙特利尔 QC G1K 9H7）

摘　要：为解决分离式数据中心存储系统中调度、部署和算法运行开销大的问题，提出将存储调度方法内嵌于白盒交换机的部署方案，通过交换机与数据处理单元（DPU）交互实现数据自动化迁移，释放存储访问的I/O占用资源。综合访问量和访问周期，设计基于贝叶斯的数据热度灵敏度优化策略，进而设计基于深度强化学习的存储资源调度优化方法，融合数据热度信息，实现更低的访问时延。仿真结果表明，与传统基于规则的LRU和FIFO算法相比，所提方法在访问总时延方面降低38.4%~67.2%；与DQN、PPO和DDPG算法相比，所提方法在访问总时延方面降低12.8%~43.2%.

关键词：分离式数据中心；多层级存储；数据迁移；深度强化学习；资源调度
中图分类号：TP393
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2025135

Storage resource scheduling optimization method for separated data center based on deep reinforcement learning

YUAN Zhengli[1], GUO Shaoyong[1], HU Xin[1], TONG Jie[2], HAO Jiakai[3], Michel Kadoch[4], YU Peng[1]
1. State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
2. China Electric Power Research Institute, Beijing 100192, China
3. State Grid Beijing Electric Power Company, Beijing 100031, China
4. École de Technologie Supérieure (ETS), University of Quebec, Montreal QC G1K 9H7, Canada

Abstract: To address the high overhead of scheduling, deployment, and algorithm execution in disaggregated data center storage systems, a deployment solution was proposed that embeded the storage scheduling mechanism into white-box switches. By enabling the interaction between the switches and data processing unit (DPU), the system achieved automated data migration, thereby freeing up I/O resources occupied by storage access. Through a Bayesian-based data temperature sensitivity optimization strategy integrating access volume and access interval, it fed the heat information to a deep reinforcement learning (DRL) scheduler that minimized access latency. Simulations show that, against rule-based LRU and FIFO, latency drops from 38.4% to 67.2%. Against DQN, PPO and DDPG, latency drops from 12.8% to 43.2 %.
Keywords: disaggregated data center, multilevel storage, data migration, deep reinforcement learning, resource scheduling

0　引言

随着数据中心存储数据量的激增，传统服务器打包式架构面临着共享需求难以满足、弹性资源分配能力弱、可扩展性差等问题。近些年，学术界和工业界开始广泛关注一种分离式数据中心架构[1]，这种架构将硬件资源按照类别彻底解耦，形成不同

收稿日期：2025–04–12；修回日期：2025–07–24
通信作者：郭少勇，syguo@bupt.edu.cn
基金项目：国家自然科学基金资助项目（No.62322103）；中央高校基本科研业务费专项资金资助项目（No.2023ZCTH11）；北京市自然科学基金资助项目（No.4232009）
Foundation Items: The National Natural Science Foundation of China (No.62322103), The Fundamental Research Funds for Central Universities (No.2023ZCTH11), The Natural Science Foundation of Beijing (No.4232009)

的硬件资源池，通过高通量数据总线将其连接，以支持资源的独立弹性扩展和按需灵活调度。然而，新型架构也面临一些挑战。首先，在存储层级方面，引入了新型存储介质，如持久性内存（PM, persistent memory）、高速固态硬盘等，打破了原有“内存–硬盘”的存储层级结构的边界，转变为“内存–持久性内存–硬盘”的新的层级结构，使存储资源调度优化更为复杂[2]，现有的数据缓存、分级策略等不能充分发挥其硬件效能。其次，在数据管理模式方面，传统由专用存储服务器管理数据的模式[3]，一旦产生大量输入/输出（I/O，input/output）操作时，有限资源的I/O总线会成为存储性能瓶颈。最后，在数据存储调度算法方面，数据迁移操作过于频繁或慵懒，均会加剧I/O资源开销，引起访问时延过长[4]。因此，通过数据存储调度算法部署与优化，破解数据中心分散式存储的性能瓶颈成为关键。

目前已有的数据存储分层调度算法，大多根据数据访问频次计算数据热度[5]。此方法更新数据热度时未考虑到数据的访问周期性，易导致冷热数据误判问题，造成数据访问频率与存储性能不匹配，进而导致访问时延增长。目前已提出的基于深度强化学习的数据调度算法[6]能够通过学习复杂的数据访问模式，以更少的数据迁移量形成更优的存储布局，为使用深度强化学习解决存储资源调度问题提供了新的思路。然而，该方法仅能在相邻层级的存储设备之间迁移数据，无法适配数据迁移更加灵活的分离式数据中心存储池。针对存储调度算法的部署问题，白盒交换机和数据处理单元（DPU, data processing unit）提供了解决方案。白盒交换机能够在高通量数据总线中发挥其强大的可编程性和灵活性，存储数据状态、进行存储决策并将迁移请求下发到DPU中；DPU能够适配不同类别的存储硬件，卸载存储相关的操作。以上述问题为指导，本文主要进行了如下研究。

1) 提出了基于白盒交换机和DPU交互的存储资源调度算法部署方案。存储调度算法部署在白盒交换机中，负责分析接收到的I/O操作信息。在固定周期到达后，交换机进行迁移决策，然后将迁移信息下发到对应存储池，最后存储池中部署的DPU负责执行具体的迁移操作。

2) 设计了基于贝叶斯的数据热度灵敏度优化方法，结合数据访问特性的周期性变化，改进经典牛顿冷却定律公式对数据周期性访问反应差的问题。设计基于深度强化学习的多层级存储资源调度优化方法，根据存储资源状态、数据热度等信息，输出数据迁移策略。

3) 在阿里云用户行为数据集上进行了仿真测试，测试结果表明，对比传统基于规则的最近最少使用（LRU，least recently used）、先进先出（FIFO，first in first out）算法，以及深度Q网络（DQN，deep Q-network）、近端策略优化（PPO，proximal policy optimization）、深度确定策略梯度（DDPG，deep deterministic policy gradient）这3种深度强化学习算法，本文存储调度优化方法访问总时延分别降低了38.4%~67.2%、12.8%~43.2%，在分离式架构中实现了更低的访问时延。

1 相关工作

随着存储需求的不断升级，企业通常采用混合存储架构来提高存储容量和访问性能。除了优化存储架构外，企业通常将更频繁访问的热数据存储在性能更高的存储设备上，以提高系统访问性能[7]。数据冷热识别算法大致分为两类：一类是基于传统缓存机制的冷热识别算法，如LRU[8]、最近最少使用K次（LRU-K，least recently used - K）[9]、最不常使用（LFU，least frequently used）[10]、自适应替换缓存（ARC，adaptive replacement cache）[11]等；另一类是基于数据特征进行建模研究，例如，文献[12]优化了原生Mongo数据库切片迁移的自动切片机制，采用朴素贝叶斯算法，根据访问特征和热负载差异来建立数据迁移机制。文献[13]在云边协调场景下，根据指定时间窗口内相邻请求的内容来预测数据的热度，同时建立模型确定文件价值，以提高边缘节点的存储利用率，减少与云服务器交互次数。然而这些方法具有局限性，无法综合考虑数据访问周期性变化，在某些动态环境中适应性较差。

目前的存储数据分层调度算法依赖基本规则或浅层学习策略，根据数据访问模式分析结果进行数据迁移。文献[14]提出的自动存储分层（AST， automated storage tiering）系统，使用深度神经网络（DNN, deep neural network）和支持向量机（SVM, support vector machine）学习过去数据的访问行为，预测云存储中文件的热度，并生成文件优先级列

表，AST 系统再结合热度和文件优先级列表来确定文件的存储位置，节约了成本、提高了系统的灵活性。但 AST 在构建时训练深度神经网络耗时长，面对数据中心指数级增长的数据，需要更高效率的算法。文献[15]设计的 ChewAnalyzer 块级存储感知框架，使用层次分类器分析数据块的访问频率、数据大小等特性，为数据块分配存储池，优化存储效率。但该框架忽略了存储池间连接成本和带宽对数据迁移效率的影响。文献[4]提出了三层存储策略，三层存储设备包含动态随机存取存储器（DRAM，dynamic random access memory）、非易失性存储器（NVM，non-volatile memory）、固态硬盘（SSD，solid state drive），该策略结合懒惰迁移和积极迁移策略优化数据放置，但仅针对特定的存储层级组合设计，未能测试其他存储组合的需求。文献[16]设计的 Data Jockey 数据管理系统，以“目标驱动”管理数据，实现了自动化的数据管理。但该系统需要频繁更新数据迁移策略，在数据作业变化频繁时可能影响系统效率和响应时间。与此同时，大量学者开始研究深度强化学习，并将深度强化学习应用于自适应资源分配问题。例如，文献[6]提出了基于强化学习的迁移策略，根据模拟软件和云框架的测试结果，尽管强化学习算法的使用增大了计算复杂度，但可以减少存储层级之间数据传输量，同时能够基于传入数据访问模式形成更优的存储布局。文献[17]基于深度强化学习构建了一种分层架构，用于自适应整体资源分配，降低数据中心能耗。文献[18]提出了一种基于异步优势演员–评论家（A3C，asynchronous advantage actor-critic）算法的资源分配方法，设计云数据中心统一的资源分配模型。这些研究为使用基于强化学习的数据分层迁移策略，提供了新的研究思路和框架。

以上方法在数据分层调度上有所突破，但是在面对大规模、动态变化的存储环境时，仍存在效率不足、适应性差和计算成本高等问题。文献[5]提出的 Olsync 引入了基于软件定义网络（SDN, software-defined networking）的 PIPO（packet in packet out）通信框架，在数据平面对存储节点进行管理，做出分层决策，但该方法仍未彻底解决集中控制可能产生的瓶颈问题。与之前工作不同的是，本文提出了白盒交换机和 DPU 交互的存储调度部署方案。白盒交换机[19]概念近些年飞速发展，允许用户按需定制交换机的网络功能，使交换机能够满足数据中心逻辑复杂的场景，并为基于深度强化学习的资源调度算法的部署提供了可能。现代交换机中配备的高性能硬件，例如中央处理器（CPU，central processing unit）、专用集成电路（ASIC，application specific integrated circuit）、现场可编程门列阵（FPGA，field programmable gate array）等，能够实现并行化处理，快速生成数据迁移决策，减少计算时延，避免传统服务器计算瓶颈对存储调度的影响。

2　系统设计

针对叶脊交换机组网的分离式数据中心，本文提出了存储资源调度优化方法，通过迁移数据优化存储池中数据分布，实现存储资源的自动化调度。分离式数据中心存储调度优化框架如图 1 所示，系统框架包含两大关键部分：部署存储调度算法的白盒交换机和内嵌 DPU 的存储池。当数据请求到达数据中心后，由计算池中的 CPU 进行处理后生成对应的数据 I/O 请求，I/O 请求经由高通量数据总线到达存储池。存储池中的 DPU 负责卸载存储相关操作，得到对应数据的具体信息，将具体信息通过数据总线返回，DPU 同时负责根据接收到的数据包执行具体的迁移操作。脊交换机通过分析接收的数据包，获得全局存储资源分布情况，在固定的周期内进行数据迁移决策，并下发数据迁移决策到对应的存储池，由存储池中内嵌的 DPU 完成具体的迁移操作。

2.1　部署存储调度算法的脊交换机

脊交换机位于数据交换的核心位置，数据中心通过脊交换机传递数据包信息。将基于深度强化学习的存储调度算法部署在脊交换机中，脊交换机通过分析计算池和存储池间传递的数据包，提取存储资源的信息，更新存储池全局资源状态表。当到达固定的迁移周期，调度算法根据数据的热度、访问次数等状态信息生成迁移决策。脊交换机通过数据总线向对应存储池发送迁移指令，指明具体数据块的迁移原层级（即数据块所在的原存储池）和目标层级。具体迁移操作由存储池中部署的 DPU 完成。

由于脊交换机位于数据中心网络的核心位置，这使其能够通过解析接收到的数据包获得各个存储池的资源状况和数据访问模式。脊交换机之间也需

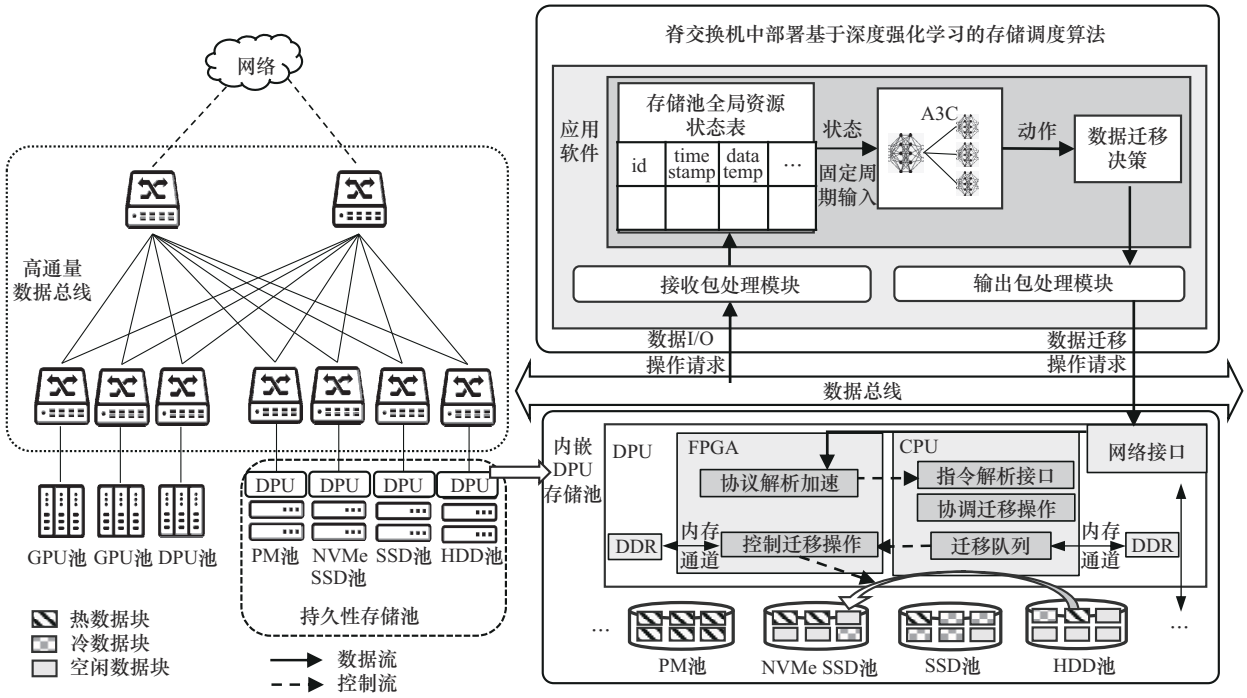


图1　分离式数据中心存储调度优化框架

要在固定周期进行远程镜像，同步数据分布情况。

2.2　内嵌DPU的存储池

在持久性存储池中，每一个存储池都内嵌DPU，负责卸载存储操作并执行迁移任务。DPU主要包含用于数据传输的网络接口、用于解析数据包的CPU处理器和负责迁移加速的FPGA处理器。

1) 网络接口。DPU的网络接口是DPU与外部通信的核心模块，负责与交换机、存储介质进行交互，卸载存储操作和数据迁移操作。网络接口模块需支持远程直接内存访问（RDMA, remote direct memory access）、非易失性内存快速通道（NVMe, non-volatile memory express）、NVMe-oF（NVMe over fabrics）等协议。

2) CPU。CPU负责解析接收到的迁移指令，并负责存储迁移的协调工作，例如判断当前存储介质是否有足够的空间来执行存储迁移操作。最后生成迁移队列，将对应参数和指令传递给FPGA，借助FPGA，执行迁移操作。

3) FPGA处理器。FPGA负责进行协议解析加速，根据CPU传递的迁移队列，与存储池交互执行具体迁移操作。CPU和FPGA通过高速串行计算机扩展总线标准（PCIe，peripheral component interconnect express）总线进行连接。

3　数据热度计算方法

3.1　基于牛顿冷却定律的数据热度计算

数据热度是衡量数据访问频率的指标，会随着时间的推移和业务需求的变化而改变。牛顿冷却定律反映了物体的温度随时间指数下降的过程，因此使用牛顿冷却定律来模拟数据热度随时间下降过程，其数学公式为

$$T'(t) = -\alpha(T(t) - H) \tag{1}$$

其中，$T(t)$是物体的温度，H表示环境温度，α为冷却系数。对式(1)进一步推导，可得

$$T(t) = T_0 e^{-\alpha(t - t_0)} + H(1 - e^{-\alpha(t - t_0)}) \tag{2}$$

对于数据库中的数据来说，环境温度可以用数据库中所有数据热度的平均值来代替。但每个数据是独立于数据库中其他数据的，数据热度根据自身访问模式和频率来确定，并不会受数据库中所有数据热度均值的影响。所以这样的计算方式实际应用意义并不大，故忽略环境温度的影响。考虑到数据在被访问时，热度会有所上升，所以将式(2)进行修改，得到热度计算式为

$$T(t_n) = T(t_{n-1}) e^{-\alpha(t_n - t_{n-1})} + T_{\text{heat}} \cdot c \tag{3}$$

其中，t_{n-1}表示数据上一次被访问的时间，T_{heat}表示数据被访问后上升的温度，c为一个离散函数，

当数据在t_n时刻被访问时，$c=1$，否则$c=0$。

基于牛顿冷却定律的热度计算公式中，数据热度会随着时间的推移不断下降，直至数据成为冷数据。但数据库可能会遇到低峰时期，在此期间数据库中大部分数据很少或者根本没有被访问。例如，在夜间或假日，用户活动减少，导致数据库总访问量下降；在进行系统升级或者维护时，可能会暂时限制用户对数据的访问。在此期间，使用基于牛顿冷却定律的热度计算公式时，会导致数据热度趋于一致，进而导致热数据的误迁移。由于基于牛顿冷却定律的热度计算公式对于数据周期性访问不灵敏，故对其进行改进。

3.2　基于贝叶斯平均的数据热度计算

数据库处于低峰期时，数据的访问量通常较低。相比而言，平稳期的热度值更加稳定，能够反映出数据的访问特性，可以作为数据热度值估计的一个可靠参考。贝叶斯平均引入数据过去的热度信息，将周期性访问模式与实时数据访问结合，增强数据热度的周期性访问灵敏度。针对每一个数据，其贝叶斯平均热度B_i计算式为

$$B_i=\frac{C\cdot m_{\text{newavg}}+\sum_{i=1}^{n_{\text{Bayes}}}T_i}{C+n_{\text{Bayes}}} \tag{4}$$

其中，C表示先验样本大小，选取低峰期间前的一段时间C个热度数据作为样本；m_{newavg}表示先验平均分，即计算所有样本数据的平均热度；n_{Bayes}表示低峰期间数据被访问的次数；$\sum_{i=1}^{n_{\text{Bayes}}}T_i$表示低峰期间使用牛顿冷却定律计算所得热度总和；通过引入C和m_{newavg}，贝叶斯平均法可以减少低峰时期数据热度下降对热数据的影响，使其热度能够向平稳时期数据热度的平均值偏倚。

3.3　阶段性结合数据热度计算方法

在阶段性结合计算方法中，根据数据库中数据的总体访问量判断数据库是否处于低峰期。当数据库处于低峰期时，使用贝叶斯平均来计算其数据热度；当数据库处于平稳期时，转向使用牛顿冷却定律来模拟数据热度随时间自然衰减的过程。

根据先验数据得到数据库在一个周期内访问量的最大值$\text{count}_{\max}$，设ρ为流量临界系数，当数据库访问量观测值$N_t\geqslant\rho\cdot\text{count}_{\max}$时，数据库处于平稳期；观测值$N_t<\rho\cdot\text{count}_{\max}$时，数据库处于低峰期，其中$\rho\in[0,1]$根据实验设定最优值。随着数据库访问量观测值变化，动态调整模型在牛顿冷却定律和贝叶斯平均之间转化，同时对最大值$\text{count}_{\max}$动态更新。阶段性模型公式为

$$T(t_n)=\beta(t_n)B(t_n)+(1-\beta(t_n))\cdot (T(t_{n-1})\mathrm{e}^{-\alpha(t_n-t_{n-1})}+T_{\text{heat}}\cdot c) \tag{5}$$

其中，$T(t_n)$表示数据在t_n时刻的热度，$B(t_n)$表示贝叶斯平均热度，α表示存储介质的冷却系数，$\beta(t_n)=\begin{cases}1, & N_t<\beta\cdot\text{count}_{\max}\\ 0, & N_t\geqslant\beta\cdot\text{count}_{\max}\end{cases}$。需要根据存储介质的性能特性，设置冷却系数$\alpha$。将2种模型结合，使其能够根据不同应用场景进行灵活转化。

4　存储资源调度算法设计

4.1　问题分析

分离式数据中心的存储调度问题主要目标是降低数据的访问时延，同时尽量减少非必要的数据迁移。为了降低访问时延，迁移策略应尽快匹配访问频率，然而当迁移策略较积极时，迁移操作过多，会占用I/O带宽。当限制迁移操作，会导致数据迁移落后于访问频率变化，进而增大访问时延。故需要综合考虑访问时延和迁移操作，建立奖励函数，在迁移后达到合理的存储数据分布。

4.2　问题形式化

在分离式数据中心存储调度场景中，将存储调度问题转化为深度强化学习问题，智能体负责对访问序列中的数据逐个生成迁移决策，并根据全局信息利用策略梯度函数对调度算法进行迭代优化，以适应动态变化的存储池。利用深度强化学习，将存储调度过程形式化为马尔可夫决策过程[20]，包含：1) 该过程表现为智能体与持久性存储池之间的交互；2) 状态空间包含整个存储池的状态和待迁移决策数据的状态；3) 动作空间包含数据可迁移的存储层级；4) 奖励函数综合考虑访问时延和迁移产生时延。存储资源调度问题映射到强化学习框架如图2所示，在固定的迁移周期到达后，内嵌在脊交换机中的智能体根据存储池全局资源状态表获取状态s_t，输出策略分布$\pi_\theta(a_t|s_t)$得到动作决策a_t（即数据的目标迁移层级），DPU根据a_t将数据data_t从原层级迁移到目标层级，存储池更新产生新的状态s_{t+1}，同时计算奖励R_t，并通过R_t更新策略函数。

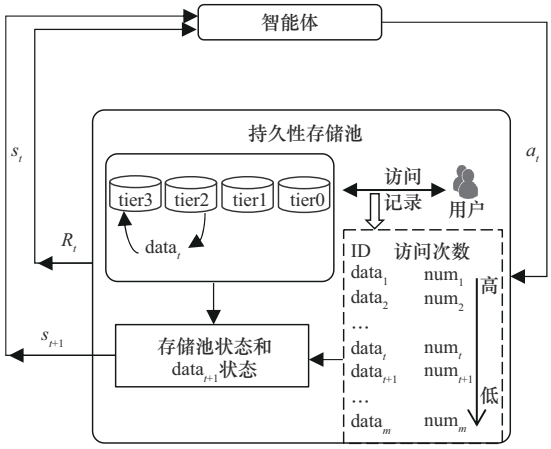


图2　存储资源调度问题映射到强化学习框架

深度强化学习算法的目标是最大化期望奖励，如式(6)所示。

$$\max_{\theta} J(\theta) = \mathbb{E}_{(s_0,a_0,s_1,a_1,\ldots)\sim\pi_\theta}\left(\sum_{t=0}^{P}\gamma^t R_t\right) \tag{6}$$

其中，P为训练结束的时刻，γ为折扣因子。奖励函数R_t指导着智能体的优化方向，根据问题分析，设数据访问时延为L_{mig}，数据迁移时延为L_{move}，将R_t形式化为$f(\cdot)$。

$$R_t = f(L_{\text{mig}}, L_{\text{move}}) \tag{7}$$

在实际的存储资源设备池中，为了保证迁移策略的稳定性，上述优化目标需要在满足一系列限制的前提下实现，具体来说，约束条件如下。

1) 存储容量约束：数据块迁移时，目标层级的空闲空间C_{free}必须大于或等于迁入数据块的大小D_{size}，即式(8)，否则需要将目标层级中的数据驱逐到存储层级更低的存储介质中。

$$C_{\text{free}} \geqslant D_{\text{size}} \tag{8}$$

2) I/O吞吐量约束：持久性存储介质的读写带宽是有限的，需要同时考虑迁移占用带宽（读取带宽R_{mig}、写入带宽W_{mig}）和正常业务带宽（读取带宽R_{app}、写入带宽W_{app}）。设存储设备的最大读取带宽为$R_{\max}$，最大写入带宽为$W_{\max}$，则进行如下约束。

$$R_{\text{app}} + R_{\text{mig}} \leqslant R_{\max} \tag{9}$$

$$W_{\text{app}} + W_{\text{mig}} \leqslant W_{\max} \tag{10}$$

3)决策唯一性约束：同一个数据在一个迁移周期内，只进行一次迁移决策。

4.3　基于深度强化学习的存储调度算法

使用强化学习求解该优化问题，需定义核心三要素：状态空间、动作空间和奖励函数。

4.3.1　状态空间

状态空间中的变量应充分描述影响数据迁移决策的关键信息，包括存储池的状态信息和待决策数据的状态信息。设T为数据迁移周期（单位为h），管理员可以根据业务需求自行设置T。存储系统每经过一个T时间后，对该时间段内被访问到的所有数据按访问次数大小排序，并逐个进行迁移决策。具体来说，每个时间段T内，共有m条数据被访问，按照数据被访问次数由高到低进行排列，得到数据id序列$\text{data}_1,\text{data}_2,\cdots,\text{data}_t,\cdots,\text{data}_m$。统计其数据状态信息以及对应的存储池状态信息，得到状态s_t，状态空间参数及含义如表1所示。

4.3.2　动作空间

动作空间定义了智能体能够采取的所有可能动作，对于存储调度算法，动作空间则是待决策数据可选择的存储池。动作空间$\mathbb{A}$定义为

$$\mathbb{A} = \{0,1,2,\cdots,n_{\text{tier}} - 1\} \tag{11}$$

其中，n_{tier}表示存储池个数。使用策略网络时，输出为动作空间的概率分布，依据概率分布选择

表1　　**状态空间参数及含义**

状态空间变量	参数	含义
存储池的状态信息	avgtemp_t	各个存储池中数据热度的均值。每经过一个周期T，更新所有数据的热度值，并计算每个层级中数据热度的平均值
	free_t	每个存储池存储介质可用的存储空间
	speed_t	每个存储池中的存储介质持续读写速度
待决策数据的状态信息	datatemp_t	当前的热度值，反映了数据在近期被访问的频率
	size_t	待决策数据大小，当数据较大时，迁移时延就会增加，影响系统响应时间
	tier_t	数据当前所在的存储层级
	freq_t	数据在该周期内被访问的次数，反映了数据的活跃度，同时也是智能体预测数据未来访问趋势的参考

动作。

4.3.3　*奖励函数*

在数据迁移调度算法中，完成一次迁移活动，就会计算对应的奖励。迁移算法的优化目标是最小化系统时延。奖励R_t共包含3个部分：λT~T中访问序列的平均访问时延、迁移数据产生的时延、λT~T中快速设备访问次数占比和慢速设备访问次数占比差值，其中$\lambda \in [0,1]$为比例系数，需通过实验确定最佳值。奖励函数计算式为

$$R_t = w_1 d_{\text{avgdelay}} - w_2 d_{\text{remove}} + \text{rate}_{\text{diff}} \tag{12}$$

其中，d_{avgdelay}为平均访问时延部分的奖励，d_{remove}为迁移数据过程产生的时延惩罚，w_1、w_2为权重系数，用于区别不同部分在奖励函数计算中的重要程度。

较低的平均访问时延意味着调度算法更高效，能够快速响应用户的访问请求。平均访问时延avgdelay计算式为

$$\text{avgdelay} = \frac{\sum \frac{\text{size}_{\lambda T\sim T}}{v_{\text{tier}}}}{N} \tag{13}$$

其中，$\sum \frac{\text{size}_{\lambda T\sim T}}{v_{\text{tier}}}$表示迁移后$\lambda T$~$T$时间段内访问序列的总时延，$\text{size}_{\lambda T\sim T}$表示$\lambda T$~$T$时间段内访问数据的大小，$v_{\text{tier}}$表示该数据所在层级的读取速度，$N$表示该段时间访问请求总数。为了确保各个奖励能够处于同一尺度，通过min-max标准化方法处理，将平均访问时延的奖励约束在[0, 1]范围内，$d_{\max}$、$d_{\min}$为访问时延的最值，基于历史数据动态更新。d_{avgdelay}计算式为

$$d_{\text{avgdelay}} = -\left(\frac{\text{avgdelay} - d_{\min}}{d_{\max} - d_{\min}} - 1\right) \tag{14}$$

在最大化奖励函数的过程中，智能体会倾向于选择能够降低平均访问时延的策略，从而提高系统响应速度。

选择λT~T时间段的访问序列计算奖励函数有2个原因：1)相比于完整T时间段的访问序列，λT~T的数据量较小，能够在模型训练和更新策略时占用更少的计算资源，实现更快的数据处理速度，这对于实时性较高的数据迁移任务至关重要；2)λT~T时间段访问序列的时效性更强，更能反映出存储系统近期的访问趋势，从而更有效地指导数据迁移操作。

约束智能体在进行决策时考虑迁移数据造成的系统时延，包含决策数据迁移到目标层级产生的时延、目标层级空闲空间不足时驱逐部分数据产生的时延。当目标层级没有足够的空闲空间时，将该层级中访问次数低的数据驱逐到更低的层级之中。迁移数据过程中产生的时延$\text{delay}_{\text{remove}}$计算式为

$$\text{delay}_{\text{remove}} = \left(\frac{\text{size}_t}{v_{\text{read}}} + \frac{\text{size}_t}{v_{\text{new}_{\text{write}}}}\right) + \sum \text{delay}_{\text{exchange}} \tag{15}$$

其中，size_t表示决策数据大小，v_{read}表示数据所在存储层级读取数据的速度，$v_{\text{new}_{\text{write}}}$表示数据目标层级写入数据的速度，$\sum \text{delay}_{\text{exchange}}$为驱逐数据产生的时延。迁移数据的时延，包含数据原始层级的一次读取和目标层级的一次写入。同样使用min-max标准化方法将迁移时延惩罚约束在[0,1]的范围内，$\text{delay}_{\text{remove}}$最小值为$\text{dr}_{\min} = 0$，即决策数据不迁移，最大值为$\text{dr}_{\max}$，基于历史数据动态更新最大值。$d_{\text{remove}}$计算式为

$$d_{\text{remove}} = \frac{\text{delay}_{\text{remove}} - \text{dr}_{\min}}{\text{dr}_{\max} - \text{dr}_{\min}} \tag{16}$$

在λT~T时间内快速设备访问次数占比和慢速设备访问次数占比差值记作$\text{rate}_{\text{diff}}$。此差值越大，说明快速设备更有效地存储了系统中访问频繁的数据，使高性能介质充分发挥性能优势；慢速设备的访问更稀疏，对于访问频率较低的数据，存储在相对较慢的设备中以优化成本。故将此项加入奖励函数中，引导智能体实现访问频率和存储性能的匹配。$\text{rate}_{\text{diff}}$计算式为

$$\text{rate}_{\text{diff}} = \frac{n_{\text{fast}} - n_{\text{slow}}}{N} \tag{17}$$

其中，n_{fast}表示迁移后快速设备中数据被访问次数，n_{slow}表示迁移后慢速设备中数据被访问次数，N表示该段时间访问请求总数。

4.3.4　A3C 算法

基于A3C算法，求解存储池中数据迁移的最优策略。基于深度强化学习的存储资源调度算法机制如图3所示，在A3C框架中，存在一个全局网络和多个本地网络，它们具有相同的网络结构，包含Actor网络（参数θ）和Critic网络（参数ϕ），可相互复制网络参数。本地网络复制全局网络参数，基于当前参数与环境进行交互，收集数据并计算梯度；全局网络不与环境进行交互，而是聚合本地网

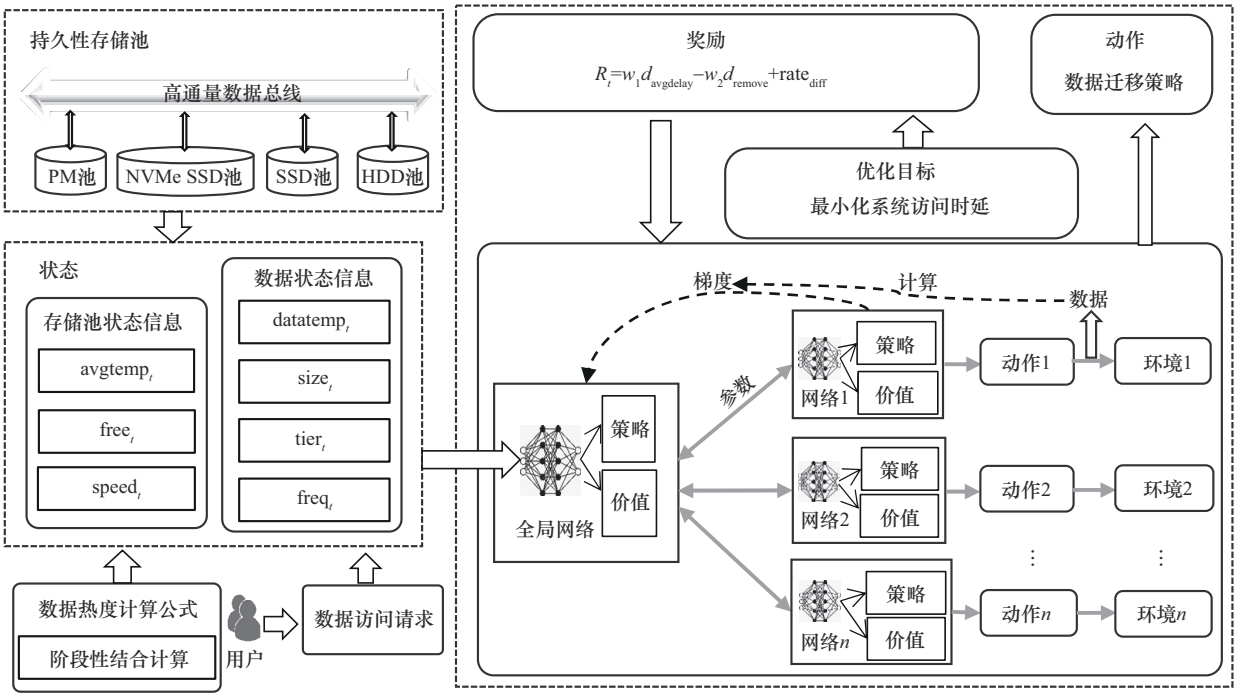


图3　基于深度强化学习的存储资源调度算法机制

络收集的梯度来更新全局网络参数，并将参数同步给本地网络。本地线程构建状态向量s_t，Actor网络输出所有动作的概率分布$\pi_\theta(a_t|s_t)$，按照概率分布选择动作a_t，并执行动作，环境反馈新的状态s_{t+1}，并使用式(12)计算奖励R_t。根据动作执行后的奖励、状态以及优势函数的值更新Actor和Critic网络参数。

1) 优势函数：本地线程缓存n步经验轨迹$\{s_t,a_t,R_t,\cdots,s_{t+n}\}$，使用多步时序差分误差（TD Error, temporal difference error）来计算优势函数的值为

$$A(s_t,a_t)=\sum_{k=0}^{n-1}\gamma^k R_{t+k}+\gamma^n V(s_{t+n};\phi)-V(s_t;\phi) \tag{18}$$

其中，$A(s_t,a_t)$为优势函数，$V(s_t;\phi)$为Critic网络所估计的当前状态的长期价值。

2)策略网络：Actor网络输出动作的概率分布，通过最大化累计奖励的期望值$J(\theta)$来更新策略参数θ，其目标函数为

$$J(\theta)=\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{P}\gamma^t A(s_t,a_t)] \tag{19}$$

其中，P为训练结束的时刻；γ为折扣因子，折扣因子$\gamma=0$时，智能体只注重眼前的奖励，$\gamma=1$时，智能体会重视长期的奖励。基于优势函数计算策略网络的梯度为

$$\nabla_\theta J(\theta)=\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{p}\nabla_\theta \ln\pi_\theta(a_t|s_t)\cdot A(s_t,a_t)] \tag{20}$$

其中，$\nabla_\theta \ln\pi_\theta(a_t|s_t)$为策略网络的对数概率梯度。

3)价值网络：Critic网络通过最小化TD误差更新参数，计算式为

$$L(\phi)=\mathbb{E}_{(s_t,a_t,R_t,s_{t+1})}[(R_t+\gamma V(s_{t+1};\phi)-V(s_t;\phi))^2] \tag{21}$$

4)全局网络：本地线程根据缓存的n步经验轨迹$\{s_t,a_t,R_t,\cdots,s_{t+n}\}$计算本地梯度$\nabla_\theta J(\theta)$和$\nabla_\phi L(\phi)$，根据式(22)更新全局网络参数为

$$\theta_g\leftarrow\theta_g+\xi\cdot\nabla_\theta J(\theta) \tag{22}$$

$$\phi_g\leftarrow\phi_g+\zeta\cdot\nabla_\phi L(\phi) \tag{23}$$

其中，ξ和ζ为学习率，θ和ϕ为线程本地网络参数，θ_g和ϕ_g为全局网络参数。

5)优化目标实现：通过最大化期望奖励，实现降低系统访问时延的优化目标。通过大量的训练，A3C能够学习到数据迁移调度问题的最优策略，在实际应用中，训练好的策略网络能够根据环境的变化，选择最优的迁移策略，持续优化系统性能。基于A3C的分离式数据存储调度算法如算法1所示。

算法1　基于A3C的分离式数据存储调度算法

输入　本地Actor和Critic神经网络参数、全局

最大训练回合数MAX_EP、更新网络的步长ITER、折扣因子γ

输出　训练后Actor和Critic神经网络参数

1)while episode <MAX_EP do：

2)　初始状态：s_0 = env.reset()

3)　从环境中采取数量为ITER的样本：

$$\{s_t,a_t,R_t,\cdots,s_{t+n}\}$$

4)　计算累计奖励

5)　计算梯度：

$$\nabla_\theta J(\theta)=\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{p}\nabla_\theta \ln\pi_\theta(a_t|\ s_t)\cdot A(s_t,a_t)]$$

$$L(\phi)=\mathbb{E}_{(s_t,a_t,R_t,s_{t+1})}[(R_t+\gamma V(s_{t+1};\phi)-V(s_t;\phi))^2]$$

6)　更新网络参数：

$$\theta_g\leftarrow\theta_g+\xi\cdot\nabla_\theta J(\theta),\phi_g\leftarrow\phi_g+\zeta\cdot\nabla_\phi L(\phi)$$

7)　if 本轮结束：

8)　　进入下一轮episode = episode+1

9)　　break

10)end while

5　实验仿真

本节在阿里云用户行为数据集上进行仿真实验，对本文存储资源调度优化方法进行性能分析。首先介绍相关实验环境，然后与传统基于规则的LRU、FIFO算法，以及DQN、PPO、DDPG这3种深度强化学习算法对比，以访问总时延和负载均衡度为指标，分析基于A3C的存储调度算法性能。

5.1　数据集选择和预处理

实验所用数据集为阿里云天池发布的淘宝用户购物行为数据集，该数据集记录了2017年11月25日到2017年12月3日随机用户的购物行为（包括点击、喜欢、加购、购买）。每一行数据由用户id、商品id、商品类目id、行为类型以及时间戳组成，将所有行为都看作对数据库中商品信息的访问，则每一行数据可以代表对该商品对应信息的一次访问行为。该数据集共有100 150 807条记录，4 162 024个商品id。

本实验采用概率抽样的方法，从原始数据集中依据各个商品被访问到的概率分布抽取400个商品id，以及对应的239 959条访问记录作为仿真实验使用的数据，然后依次进行如下操作。

1)去除用户id、商品类目id、行为类型3列无关数据。

2)为每个商品id生成1~5的随机整数表示数据大小，以MB为单位。

3)按照时间戳从小到大排列数据集，每一行数据由商品id、数据大小、时间戳组成，每一行数据代表用户对数据信息的一次访问。

4)以时间戳1511845200为分界线，将数据划分为用于深度强化学习智能体训练的训练集，和用于测试算法性能的测试集，数据集信息如表2所示。

5.2　仿真设备信息

本文在Python3.10和Pytorch1.12.1环境下，对基于A3C的存储调度算法进行仿真实验。实验模拟的存储池由4种存储设备组成，存储设备信息及仿真时延模拟容量大小如表3所示。

表2　　**数据集信息**

数据集	访问记录数	起始时间戳	起始日期时间	结束时间戳	结束日期时间
训练集	89 115	1511550000	2017年11月25日 03:00:00	1511845199	2017年11月28日 12:59:59
测试集	150 844	1511845200	2017年11月28日 13:00:00	1512277199	2017年12月3日 12:59:59

表3　　**存储设备信息及仿真时延模拟容量大小**

存储设备	详细信息	仿真实验模拟容量
Intel® Optane™ Persistent Memory 200 Series	256 GB，读取带宽为8.1 GB/s，写入带宽为3.15 GB/s	64 MB
金士顿PCIe4.0 NVMe M.2 SKC3000S/512 G	512 GB，读取速度为7 000 MB/s，写入速度为3900 MB/s	128 MB
西部数据NAS SATA SSD WDS100T1R0A	1 TB，读取速度为560 MB/s，写入速度为530 MB/s	256 MB
西部数据SATA HDD WD4003VRYZ	4 TB，传输速率为267 MB/s	1 GB

数据迁移以数据块为粒度，存储设备设置1 MB为块大小。合适的数据块大小有助于减少存储中碎片的产生，同时也有利于简化存储调度算法设计。

5.3 对比指标

1)访问总时延

使用每个迁移周期内访问请求的总时延作为评价算法性能的指标。数据迁移是数据库中不可避免的操作，特别是在云环境，数据迁移会频繁发生。为了更全面地评估系统时延，将数据迁移产生的时延纳入总时延的计算之中，确保更合理地评估系统性能。

2)负载均衡度

负载均衡度能够衡量多个存储设备间负载的均匀程度。存储设备的负载可以由每秒读写操作次数（IOPS，input/output operation per second）来间接估计。IOPS表示存储设备在1 s中内处理读写操作的次数，反映了存储设备的繁忙程度，计算式为

$$\mathrm{IOPS}=\frac{\text{总操作数}}{\text{迁移周期}} \tag{24}$$

在一个迁移周期结束后，计算得到每个设备的IOPS，再通过计算标准差来衡量存储池的负载均衡度。负载均衡度计算式为

$$\text{负载均衡度}=\sqrt{\frac{\sum(\mathrm{iops}_i-\overline{\mathrm{iops}})^2}{n_{\mathrm{tier}}}} \tag{25}$$

其中，iops_i表示每个存储池的IOPS，$\overline{\mathrm{iops}}$表示IOPS的平均值，n_{tier}表示存储池个数。当IOPS的标准差越小时，负载均衡度就越小，负载越平衡，反之负载均衡越不平衡。

5.4 参数设置

为了评估关键参数对本文优化方法的影响，对奖励函数中的权重系数w_1、w_2进行网格搜索，分别取值为{0,0.2,0.4,0.6,0.8,1}，计算不同(w_1,w_2)组合对应的访问总时延。不同权重组合对应的访问总时延灰度热力图如图4所示。

根据图4可知，$w_2\in[0,0.2]$时的访问总时延均小于$w_2\in[0.4,1.0]$时的访问总时延，且随w_1的增大，访问总时延降低。在(w_1,w_2)取(0.8, 0)、(1.0, 0)、(1.0, 0.2)时，均能使优化方法具有较小的访问总时延。由于λT~T中比例系数λ和流量临界系数ρ同样影响优化方法效果，故选取以上3种(w_1,w_2)组合，在不同组合上对$\lambda\in\{\frac{3}{4},\frac{4}{5},\frac{5}{6},\frac{7}{8}\}$和$\rho\in\{0.2,0.3,0.4\}$进行网格搜索，寻找优化方法性能的帕累托最优解，不同(w_1,w_2)组合取值对应的访问总时延如图5所示。

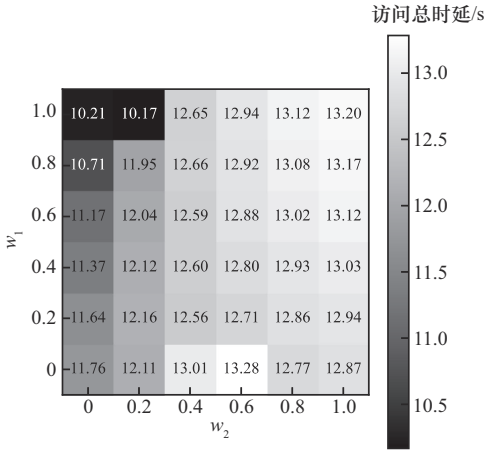
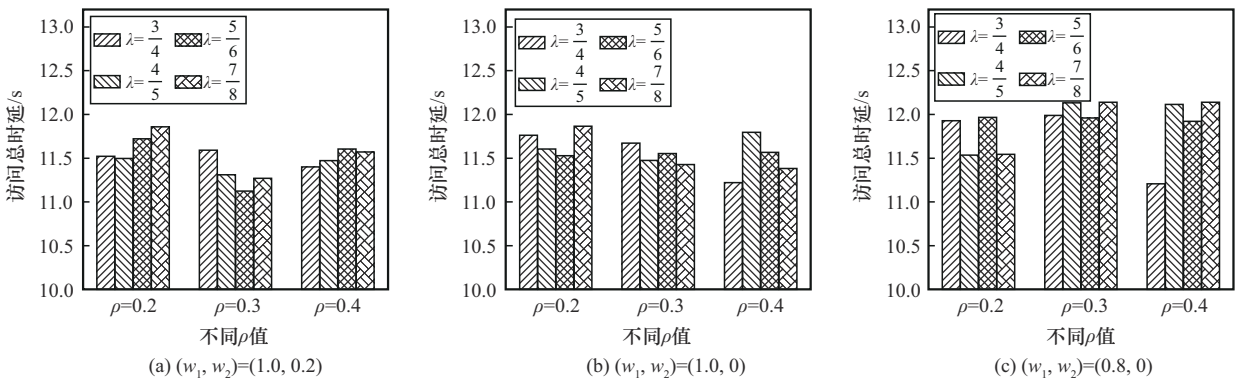


图4 不同权重组合对应的访问总时延灰度热力图

根据图5可知，当(w_1,w_2)取(1.0, 0.2)时，访问总时延普遍更低，故最终选择权重系数组合为(1.0, 0.2)。在$\lambda=\frac{5}{6},\rho=0.3$时，访问总时延效果最优，故最终选取$\lambda=\frac{5}{6},\rho=0.3$。

除了影响优化方法性能的参数外，深度强化学



(a) (w_1, w_2)=(1.0, 0.2) (b) (w_1, w_2)=(1.0, 0) (c) (w_1, w_2)=(0.8, 0)

图5 不同(w_1, w_2)组合取值对应的访问总时延

习算法训练过程也对折扣因子γ、更新网络步长ITER、学习率等参数敏感。对于学习率，本实验在训练A3C的过程中使用自适应优化器Adam自适应更新学习率，不需要搜索优化。对于γ和ITER，由于2个参数对于深度强化学习训练过程的影响不同，所以对其分别进行单因素实验。取$\gamma\in${0.60, 0.70, 0.80, 0.90, 0.92, 0.94, 0.96, 1.00 }，ITER∈{10, 16, 32, 64, 128}，对2个参数分别进行模型训练，记录训练过程奖励值变化。不同γ取值对应的训练过程中奖励值变化如图6所示。不同ITER取值对应的训练过程中奖励值变化如图7所示。

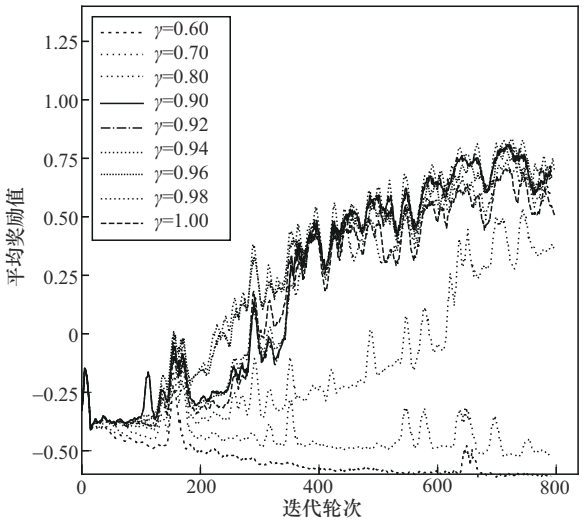


图6　不同γ取值对应的训练过程中奖励值变化

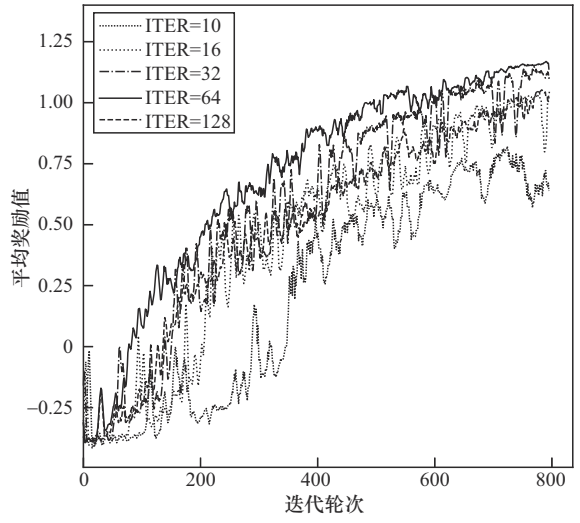


图7　不同ITER取值对应的训练过程中奖励值变化

根据图6，$\gamma\in$[0.90,1.00]时训练效果更优，且在该区间内，奖励值变化对γ并不敏感，故最终选取算法默认参数γ=0.90。根据图7，ITER=64时奖励值收敛速度更快，故最终选取ITER=64。

以上详细说明了影响优化方法性能的超参数选值过程。此外，参数迁移周期T、上升温度T_{heat}、数据热度样本数C、冷却系数α和存储池个数n_{tier}并非影响优化方法性能的关键因素，故直接设定其值。访问量最大值$count_{max}$和最值d_{max}, d_{min}, dr_{max}，通过统计训练集数据确定。最终实验参数选值如表4所示。

表4　　参数选值

参数	选值
奖励函数权重系数(w_1, w_2)	(1.0, 0.2)
λT~T中的比例系数λ	$\frac{5}{6}$
流量临界系数ρ	0.3
折扣因子γ	0.90
更新网络的步长ITER	64
迁移周期T/h	1
数据被访问后上升的温度T_{heat}	1
低峰期间前的一段时间数据热度样本数C	5
4种存储设备(速度由慢到快)冷却系数值α	[0.01,0.005, 0.002, 0.001]
存储池个数n_{tier}	4
数据库在一个周期内访问量的最大值$count_{max}$	2 889
最值d_{max},d_{min},dr_{max}	d_{max}=0.001 7 d_{min}=0.000 2 dr_{max}=0.160 0

5.5　结果分析

本文首先以2种传统的缓存算法作为对比方法。1)基于LRU的数据迁移策略，访问到的数据不在速度最快的存储介质之中，则向上迁移一层；若上层存储介质已满，则替换掉其中最近访问时间最早的数据。2)基于FIFO数据迁移策略，则替换掉最先进入该设备中的数据。然后与DQN、PPO、DDPG这3种深度强化学习对比访问总时延和负载均衡度。实验结果如下。

1)访问时延实验

本文优化方法与传统的LRU、FIFO算法的访问总时延对比如图8所示。基于A3C的数据迁移调度算法相较于其他2种算法，在降低访问总时延方面效果显著。与LRU算法对比，A3C算法的总时

延降低了38.4%。与FIFO算法对比，A3C算法的总时延降低了67.2%。

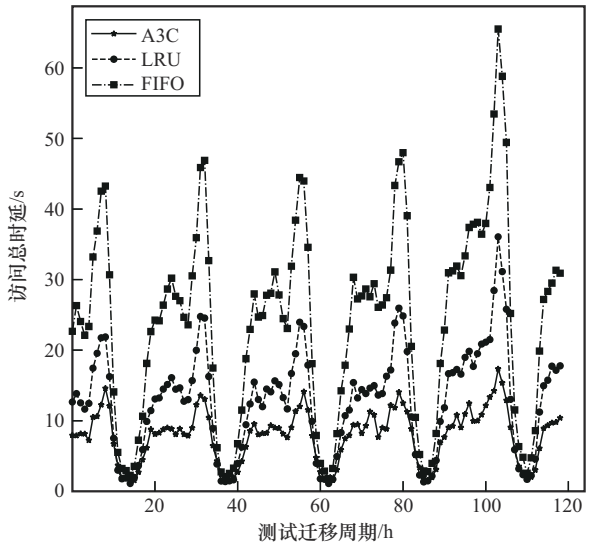


图8　与传统算法的访问总时延对比

本文优化方法与基于深度强化学习的DQN、DDPG和PPO这3种算法的访问总时延对比如图9所示。

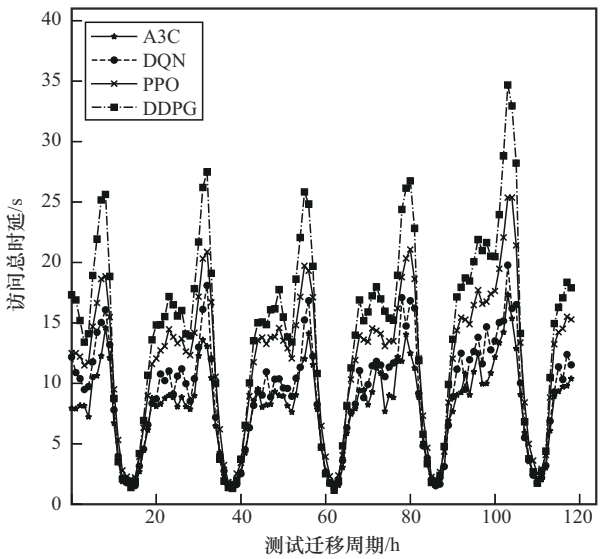


图9　与深度强化学习算法的访问总时延对比

由图9可知，相较于其他3种深度强化学习算法，基于A3C的算法表现出更低的访问总时延。与DQN算法、PPO算法和DDPG算法对比，A3C算法的总时延分别降低了12.8%、33.2%、43.2%。

A3C算法的核心优势在于能够在访问流量变化的非平稳存储资源环境中避免陷入局部最优，在有限的数据和算力情况下，进行并行采样，高效探索动态环境。而DQN算法依赖历史数据，动态适应性不足；DDPG则十分依赖奖励函数设计，且易局部最优；PPO的更新效率较低，需要海量的数据集进行训练。故A3C算法更适用于当前的场景。

2)负载均衡实验

本文优化方法与传统的LRU、FIFO算法的负载均衡度分布对比如图10所示。FIFO算法的负载均衡度波动更大，稳定性较差；而A3C算法波动更小，相对比较均衡，没有出现极端峰值或谷值。A3C算法的负载均衡度小于LFU算法的负载均衡度。

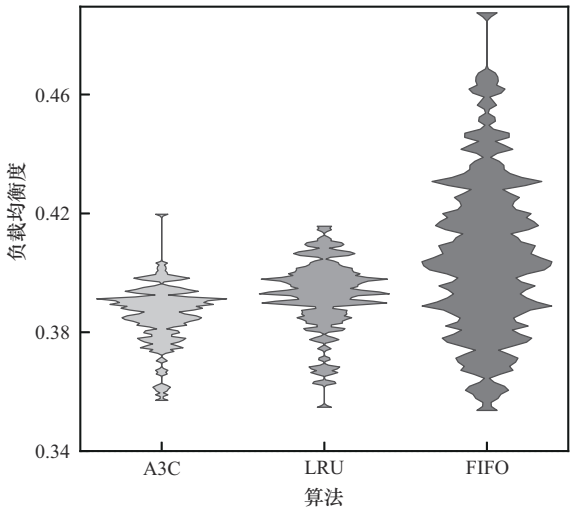


图10　与传统算法的负载均衡度分布对比

本文优化方法与基于深度强化学习的DQN、DDPG和PPO这3种算法的负载均衡度分布对比如图11所示。对比DQN算法，A3C算法的波动和峰值更小；对比PPO算法和DDPG算法，A3C算法的负载均衡度更低。综上，A3C算法在保证了较低时延的同时，更好地实现了负载均衡。

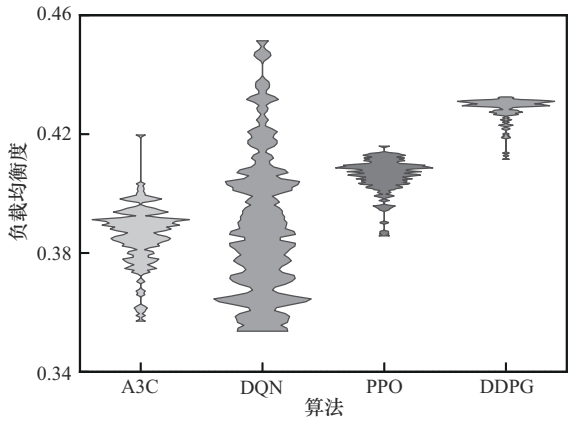


图11　与深度强化学习算法的负载均衡度分布对比

3)数据热度消融实验

为了验证阶段性数据热度计算方法对原牛顿冷却定律的改进作用，本文分别将式(3)、式(5)计算的数据热度进行可视化。使用基于牛顿冷却定律的计算方法如图 12 所示，使用阶段性热度计算方法如图 13 所示，数据在低峰时期的热度值变得更加稳定，向平稳时期数据热度均值偏倚。

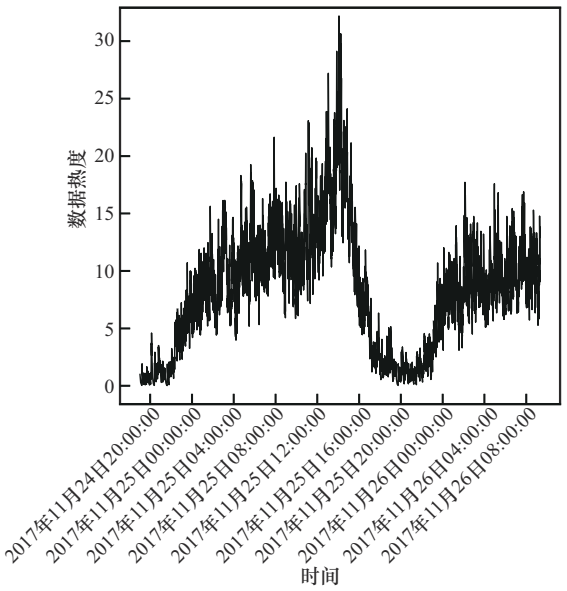


图 12　使用基于牛顿冷却定律的计算方法

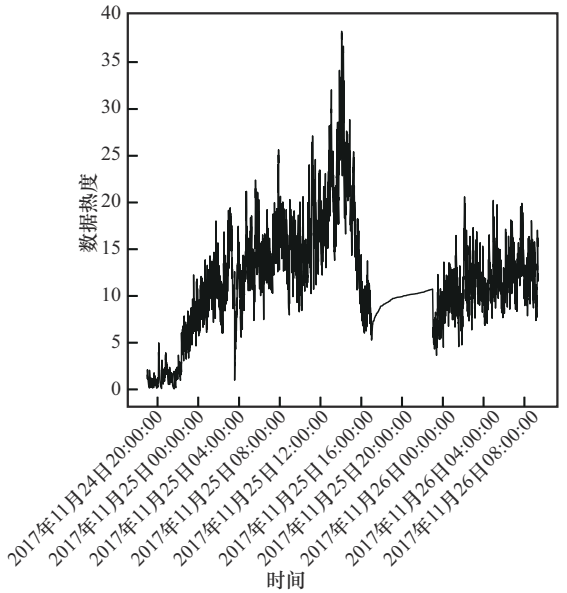


图 13　使用阶段性热度计算方法

为了验证阶段性热度计算方法在数据迁移算法中的有效性，本文设计数据热度消融实验，去除算法中热度相关的部分，比较其与本文算法的访问总时延，数据热度消融实验访问总时延对比如图 14 所示。

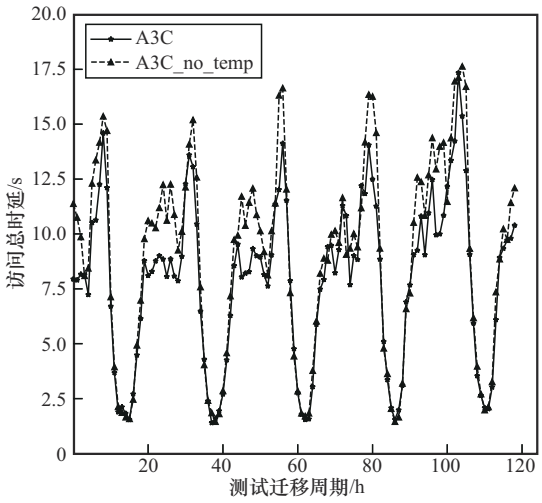


图 14　数据热度消融实验访问总时延对比

由图 14 可知，本文 A3C 算法能够实现更低的访问总时延。与未添加热度的算法相比，本文算法访问总时延降低了 12.3%。

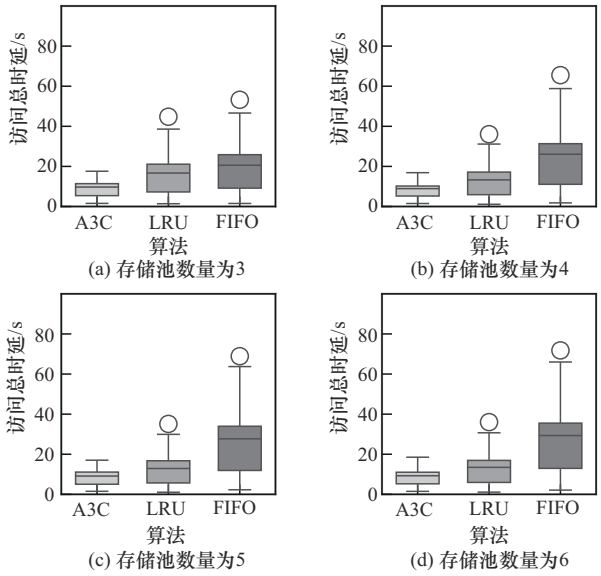
4)不同数量存储池实验

为了进一步验证本文方法的有效性，本文设计了 3 种算法在不同数量的存储池下的实验，总时延对比如图 15 所示。由图 15 可知，随着存储层级数量的增加，A3C 算法的访问总时延的中位数和四分位数范围均低于 LRU 算法和 FIFO 算法，表现出最低的访问总时延。且不同存储层级数量变化下，A3C 算法箱线图波动较小，能够适应不同层级配置的动态需求。说明基于 A3C 的动态决策机制可以优化数据访问时延，并适应不同层级配置。

6　结束语

在分离式数据中心存储系统中，存储调度算法是十分重要的技术。首先，本文对分离式数据中心存储系统进行了调研，概述了数据分层存储调度算法相关工作。然后，提出了基于牛顿冷却定律和贝叶斯平均的阶段性热度计算公式。接着，基于深度强化学习中的 A3C 框架，提出了依据存储池状态和数据状态做出迁移决策的存储调度优化方法，并详细说明了设计过程。最后，进行了仿真实验，实验结果显示基于 A3C 的迁移算法能够有效降低系统访问总时延。

本文虽然在数据迁移调度算法问题上取得了一



(a) 存储池数量为3　(b) 存储池数量为4

(c) 存储池数量为5　(d) 存储池数量为6

图15　不同数量存储池的访问总时延对比

定的成果，并在公开数据集上进行了实验验证，但在实验设计时考虑的因素较为简单，实际应用场景中，还需要考虑网络通信、服务质量要求、成本效益等其他多种因素。

参考文献：

[1] 舒继武, 陈游旻, 汪庆, 等. 分离式数据中心的存储系统研究进展[J]. 中国科学: 信息科学, 2023, 53(8): 1503-1528.
SHU J W, CHEN Y M, WANG Q, et al. Progress on storage systems for disaggregated data centers[J]. Scientia Sinica (Informationis), 2023, 53(8): 1503-1528.

[2] 陈游旻, 李飞, 舒继武. 大数据环境下的存储系统构建: 挑战、方法和趋势[J]. 大数据, 2019, 5(4): 27-40.
CHEN Y M, LI F, SHU J W. Building storage systems in big data era: challenges, methods and trends[J]. Big Data Research, 2019, 5(4): 27-40.

[3] ZHANG Q Z, BERNSTEIN P A, CHANDRAMOULI B, et al. DDS: DPU-optimized disaggregated storage[J]. Proceedings of the VLDB Endowment, 2024, 17(11): 3304-3317.

[4] ARULRAJ J, PAVLO A, MALLADI K T. Multi-tier buffer management and storage system design for non-volatile memory[J]. arXiv Preprint, arXiv:1901.10938, 2019.

[5] LI Z K, WANG Y, NIE S Q, et al. Olsync: Object-level tiering and coordination in tiered storage systems based on software-defined network[J]. Future Generation Computer Systems, 2025, 163: 107521.

[6] ZHANG T R, HELLANDER A, TOOR S. Efficient hierarchical storage management empowered by reinforcement learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(6): 5780-5793.

[7] SONG Y F, FAN S H, XU J X, et al. A novel hot-cold data identification mechanism based on multidimensional data[C]//Proceedings of the 2022 5th International Conference on Data Science and Information Technology (DSIT). Piscataway: IEEE Press, 2022: 1-5.

[8] DAN A, TOWSLEY D. An approximate analysis of the LRU and FIFO buffer replacement schemes[C]//Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems - SIGMETRICS'90. New York: ACM Press, 1990: 143-152.

[9] O'NEIL E J, O'NEIL P E, WEIKUM G. The LRU-K page replacement algorithm for database disk buffering[J]. ACM SIGMOD Record, 1993, 22(2): 297-306.

[10] LEE D, CHOI J, KIM J H, et al. On the existence of a spectrum of policies that subsumes the least recently used (LRU) and least frequently used (LFU) policies[J]. ACM SIGMETRICS Performance Evaluation Review, 1999, 27(1): 134-143.

[11] MEGIDDO N, MODHA D S. ARC: A self-tuning low overhead replacement cache[C]//2nd USENIX Conference on File and Storage Technologies (FAST 03). Berkeley: USENIX Association, 2003: 115-130.

[12] 冯超政, 蒋溢, 何军, 等. 基于冷热数据的MongoDB自动分片机制[J]. 计算机工程, 2017, 43(3): 7-10,17.
FENG C Z, JIANG Y, HE J, et al. Auto-sharding mechanism in MongoDB based on cold and hot data[J]. Computer Engineering, 2017,43(3): 7-10, 17.

[13] ZHANG L, LI L, CHEN H L, et al. A cache replacement algorithm for industrial edge computing application[J]. Computer Research and Development, 2021, 58(7) :1533-1543.

[14] HSU Y F, IRIE R, MURATA S, et al. A novel automated cloud storage tiering system through hot-cold data classification[C]//Proceedings of the 2018 IEEE 11th International Conference on Cloud Computing (CLOUD). Piscataway: IEEE Press, 2018: 492-499.

[15] GE X Z, XIE X C, DU D H C, et al. ChewAnalyzer: workload-aware data management across differentiated storage pools[C]//Proceedings of the 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). Piscataway: IEEE Press, 2018: 94-101.

[16] SHIN W, BRUMGARD C D, XIE B, et al. Data jockey: automatic data management for HPC multi-tiered storage systems[C]//2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS). Piscataway: IEEE Press,2019:511-522.

[17] LIU N, LI Z, XU J L, et al. A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning[C]// 2017 IEEE 37th International Conference on Distributed Compu-ting Systems (ICDCS). Piscataway: IEEE Press,2017: 372-382.

[18] CHEN Z Y, HU J, MIN G Y, et al. Adaptive and efficient resource allocation in cloud datacenters using actor-critic deep reinforcement learning[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 33(8): 1911-1923.

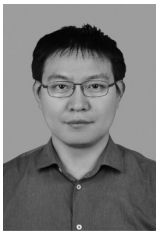
[19] 张成林, 宋玲玲. 面向未来网络的白盒交换机体系综述[J]. 信息技术与网络安全, 2022, 41(3): 2-8.
ZHANG C L, SONG L L. A review on architecture of white box

switches for future networks[J]. Information Technology and Network Security, 2022, 41(3): 2-8.

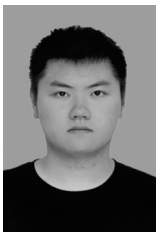
[20] LIU M Y, PAN L, LIU S J. RLTiering: a cost-driven auto-tiering system for two-tier cloud storage using deep reinforcement learning[J]. IEEE Transactions on Parallel and Distributed Systems, 2023, 34(2): 501-518.

[作者简介]

袁政利（2001– ），女，河北邯郸人，北京邮电大学博士生，主要研究方向为DPU、存储优化、强化学习等。



郭少勇（1985– ），男，河北邢台人，博士，北京邮电大学教授、博士生导师，主要研究方向为工业互联网网络管控、电力智联网等。



胡鑫（1999– ），男，重庆人，北京邮电大学博士生，主要研究方向为边缘计算、模型推理、分布式机器学习系统等。

仝杰（1983– ），男，山西运城人，博士，中国电力科学研究院用电与能效研究所教授级高级工程师，主要研究方向为算电协同、电力人工智能、电力物联网等。

郝佳恺（1972– ），男，北京人，国网北京市电力公司特级专家、正高级工程师，主要研究方向为电力信息通信建设、运维等。

Michel Kadoch（1944– ），男，博士，魁北克大学高等技术学院教授，主要研究方向为无线自组网、5G和6G网络、网络管理等。

喻鹏（1986– ），男，湖北随州人，博士，北京邮电大学副教授、博士生导师，主要研究方向为5G/6G网络智能管控。