面向数据库参数调优的协作型多智能体模型

李江敏[1],乔少杰[1],韩 楠[2],吴 涛[3],高瑞玮[1],彭钰寒[1],谢添丞[1],冉黎琼[1]
(1. 成都信息工程大学软件工程学院,四川成都 610225;2. 成都信息工程大学管理学院,四川成都 610103;
3. 重庆邮电大学网络空间安全与信息法学院,重庆 400065)

摘 要: 数据库参数调优是提高数据库性能的重要任务之一. 数据库参数可按照作用域范围和功能进行分类,探究某一类参数之间的相互影响以及不同类别之间的相互影响是非常重要的,但是现有方法尚未考虑这一方面. 提出面向数据库参数调优的协作型多智能体模型DBT-MADDPG(DataBase Tuning-Multi-Agent Deep Deterministic Policy Gradient),设计单智能体预训练模型SA(Single Agent)、多智能体联合训练模型JAM(Joint Action Model)以及基于概率选择的联合训练模型JAPM(Joint Action Probability Model),分阶段对数据库参数进行调优. 实验结果表明,DBT-MADDPG模型能够分功能、分参数级别对数据库参数进行调优,在SA模型训练阶段便能到达主流算法的性能,且比主流算法快17.85%获得较佳配置.

关键词: 多智能体;参数调优;数据库系统;自学习;联合训练模型
基金项目: 国家自然科学基金(No.62272066);四川省科技计划(No.2021JDJQ0021,No.2022YFG0186,No.2022NSFSC0511,No.2023YFG0027);教育部人文社会科学研究规划基金(No.22YJAZH088)
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112(2024)11-3751-06
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230724

A Collaborative Multi-Agent Model for Database Parameter Tuning

LI Jiang-min[1], QIAO Shao-jie[1], HAN Nan[2], WU Tao[3], GAO Rui-wei[1], PENG Yu-han[1],
XIE Tian-cheng[1], RAN Li-qiong[1]
(1. *School of Software Engineering, Chengdu University of Information Technology, Chengdu, Sichuan* 610225, *China*;
2. *School of Management, Chengdu University of Information Technology, Chengdu, Sichuan* 610103, *China*;
3. *School of Cyber Security and Information Law, Chongqing University of Posts and Telecommunications, Chongqing* 400065, *China*)

Abstract: Database parameter tuning is one of the crucial tasks in improving the performance of database systems. Database parameters can be classified based on their scopes and functionalities. It plays an essential role in investigating the mutual influence of parameters within a specific category or between different categories. But, the existing methods do not take into consideration this aspect. A collaborative multi-agent model called DBT-MADDPG (DataBase Tuning-Multi-Agent Deep Deterministic Policy Gradient) is proposed for database parameter tuning. A single-agent pre-training model called SA (Single Agent), a multi-agent joint training model called JAM (Joint Action Model), and a joint training model based on probabilistic selection called JAPM (Joint Action Probability Model) are designed for tuning the database parameters at different stages. The experimental results show that the DBT-MADDPG model is capable of tuning the database parameters at different functional and parameter levels, and can reach the performance of mainstream algorithms in the training stage of the SA model, and is 17.85% faster than the state-of-the-art algorithms to obtain the optimal configuration.

Key words: multi-agent; parameter tuning; database system; self-learning; joint training model
Foundation Item(s): National Natural Science Foundation of China (No.62272066); Sichuan Science and Technology Program (No.2021JDJQ0021, No.2022YFG0186, No.2022NSFSC0511, No.2023YFG0027); Planning Foundation for Humanities and Social Sciences of Ministry of Education of China (No.22YJAZH088)

收稿日期:2023-08-02;修回日期:2024-02-07;责任编辑:李勇锋

1 引言

数据库管理系统(Database Management System, DBMS)的性能与数百个参数配置及特定工作负载有关[1,2]. 大量的可调参数使得手动调优数据库成为了一项挑战工作. 随着人工智能的快速发展,众多学者相继提出了不同的自动化调优方法和工具[3~12],以往的工作主要分为三类:(1) 基于搜索的优化模型:如BestConfig[4]只有在特定工作负载的情况下,该方法才会有较好的结果;(2) 基于贝叶斯的优化模型:如OtterTune[5]和基于元学习[6]的ResTune[7],这类方法的局限性在于其流线型的结构使得前一阶段得到的最佳配置,并不一定保证适用于当前阶段的工作负载;其次,其不支持高维和连续空间上的参数调优. (3) 基于深度强化学习的优化模型:如Qtune[8]为每个查询搜索较佳的参数配置. 其仅在查询级别的工作负载具有较好的表现,在复杂的工作负载环境下,该方法表现并不优秀. CDBTune+[9]只能提供粗粒度调优(例如,只读或只写的工作负载).

本文首次将多智能体算法[13~15]应用于数据库参数调优领域,并做出以下贡献:(1)提出单智能体预训练模型SA,通过调优某一种类型的参数(例如,查询参数,I/O参数),每个Agent负责一种类型的参数,既应用了单智能体系统在有限的样本空间中学习参数配置的优点,又弥补了以往的方法[8, 9, 11]无法探索某一类型参数之间相互影响的不足. (2)提出多智能体联合动作模型JAM,针对数据库场景下,多智能体环境是离散的,Agent的动作执行和环境反馈不同步的问题,利用JAM解决环境离散下Agent之间的协作问题,探究了不同类型参数之间的相互影响,提高了可调参数数量. (3)提出基于概率选择的联合训练模型JAPM,该模型继承JAM的优点和历史经验,赋予每个Agent概率选择的能力,根据概率因子选择是否执行某个Agent的动作. 减少了不重要的Agent调优次数,提高重要Agent的调优次数. (4)提出面向数据库参数调优的协作型多智能体模型DBT-MADDPG,通过阶段式训练,实现分功能、分级别地对数据库参数进行调优. 探究了相同类型参数之间的相互影响以及不同类型参数之间的协作关系.

2 问题描述

为了详细描述算法的流程和框架,本节给出一些基本定义.

定义1 给定一组n个参数的配置$\bar{\theta}_{\text{all}}=\{\bar{\theta}_1,\bar{\theta}_2,\cdots,\bar{\theta}_n\}$作为输入,每个参数的值域为$\Theta_i$,基于一组配置,其配置空间表示为$\Theta_{\text{all}}=\Theta_1\cdot\Theta_2\cdot\cdots\Theta_n$,参数的域可以是连续、离散或分类的.

定义2 给定一组配置$\bar{\theta}_{\text{all}}$和某一特定工作负载$W$,使得数据库在该工作负载下工作的性能满足式(1):

$$\bar{\theta}^*_{\text{all}}=\arg\max_{\bar{\theta}_{\text{all}}\in\Theta_{\text{all}}} f(\bar{\theta}_{\text{all}},W) \tag{1}$$

其中,f为目标函数,性能指标包括系统的吞吐量、第95百分位延迟等.

3 基于多智能体的数据库参数调优

3.1 DBT-MADDPG模型

多智能体环境下,每个Agent单独和环境交互,每个Agent都具有自己的观测和动作空间. 数据库环境下,多智能体的环境是离散的,Agent的动作和状态并不同步,即当Agent做出动作,只有经过工作负载的测试才能获得该动作对数据库性能的影响. 针对多智能体环境,本文提出三种处理方法:(1) 当一个Agent做出动作,将其映射为数据库的推荐配置并激活配置,随后进行负载测试并记录性能数据;(2) 将多个Agent的预测动作联合映射为一组数据库的推荐配置,由于多个Agent参与调优,这要求Agent之间不能存在相同的数据库参数;(3)为每个Agent设置一个概率因子P^i,在算法训练的后期,Agent根据P^i,随机地参与联合动作推荐. 当一个Agent控制的数据库参数对数据库性能的提升不明显时,根据P^i减小该Agent的动作参与,进而降低其对其他Agent所做动作的负面影响. 概选因子的计算式如下:

$$\Delta r_t^i=r_t^i-r_{\text{ready}}^{[1,2,3]} \tag{2}$$

$$w=\Delta r_t^i/(\Delta r_t^1+\Delta r_t^2+\Delta r_t^3) \tag{3}$$

$$P^i=P^i+\frac{1}{\pi}*\tanh(w/\pi) \tag{4}$$

式(2)中,$r_{\text{ready}}^{[1,2,3]}$为环境初始化获得的联合奖励,是所有Agent联合动作所得到的奖励. Δr_t^i表示当前Agent i在t时刻得到的奖励与所有Agent联合奖励的差值. 式(3)中,w表示当前Agent i与其他Agent对数据库影响的权重占比,概选因子P^i的更新如式(4),$(1/\pi)*\tanh(w/\pi)$是关于权重占比w的缩放.

基于上述三种方式,提出协作型多智能体模型DBT-MADDPG,模型由SA、JAM以及JAPM模型组成. SA模型采用第一种方式,JAM模型采用第二种方式,JAPM模型采用第二、三种方式.

3.2 单智能体预训练模型

提出单智能体预训练模型SA,训练初期,每个Agent负责调优一类功能或参数级别相似的参数,目的是探究单个Agent对数据库性能的影响,进而优化其网络,使得其神经网络偏向于调节重要旋钮. 该过程Agent之间相互不影响. 与传统的单智能体模型相比,SA模型的优点在于更深入地探究相似参数之间的关系,也更容易识别出一些重要参数.

动作执行过程:(1)每个Agent根据Actor的策略网络

π生成一个动作信息a_t^i;(2)采用低纬度投影方法[12]将动作信息映射为具体的参数配置;(3)DBMS应用算法推荐的参数配置;(4)激活某一工作负载W进行性能测试.

3.3　多智能体联合训练模型

提出多智能体联合动作模型JAM,训练中期,JAM模型采用组合算法得到所有Agent动作的组合,与SA模型的动作执行阶段相比,执行过程:(1)组合算法得到动作的所有组合,标记为$\hat{a}_t^{\text{list}}$. 以N=3为例,多组推荐配置的集合$\hat{a}_t^{\text{list}}$为{1, 2, 3, [1, 2], [1, 3], [2, 3], [1, 2, 3]};(2)依次将所有配置应用于DBMS,并进行负载测试.

与SA模型相比,JAM模型的优点包括:(1)继承了SA模型训练阶段的成果;(2)探讨了不同类型参数之间的协作关系;(3)扩展了可调优的参数数量.

3.4　基于概率选择的联合训练模型

提出了基于概率选择的联合训练模型JAPM,或称概选模型. 训练后期,JAPM模型在动作执行上多了一步判断的步骤:随机生成一个数P_r^i,当$P^i \leqslant P_r^i$时,Agent i不参与本轮的参数调优. 概率因子的更新如式(2)~(4). 特别地,概率因子P^i在DBT-MADDPG的整个生命周期中一直存在,在SA和JAM模型训练阶段更新但不使用,而在JAPM模型训练阶段使用并且更新. JAPM模型如算法1所示.

算法1　JAPM模型

输入:一组M_1时刻的经验$(\hat{o}_{M_1+M_2}^{[1,2,3]}, \hat{a}_{M_1+M_2}^{[1,2,3]}, \hat{r}_{M_1+M_2}^{[1,2,3]})$,联合训练数$M_3$.

输出:每个Agent的概选因子P^i.

```
FOR episode = M1+ M2 TO M1+M2+M3 DO
  FOR i = 1 TO N DO //遍历每一个Agent
    随机生成一个[0, 1]之间的随机变量r;
    IF r≥ agent_p^i;   //agent_p^i为概选因子P^i
      CONTINUE;   //该Agent没有被选中
    ELSE
      根据Agent i的策略网络π,得到一个动作a_t^i,并将其加入到联合动作â_t^x中;
  END FOR
  环境执行动作â_t^x,得到环境新的观测ô_t^x;
  FOR agent IN item //遍历元组x中的每个Agent
    计算奖励r_t^i = R(ô_{t-1}^{x'}, ô_t^x);
    将(ô_{t-1}^x, â_{t-1}^x, r̂_{t-1}^x, ô_t^x)存入Agent的经验缓冲池D^i中;
    计算Agent i的概选参数P^i;
  END FOR
  更新每个Agent的Actor网络和Critic网络;
END FOR
```

算法输入中,M_1、M_2、M_3是SA、JAM以及JAPM模型各阶段的训练次数,性能观测值$\hat{o}_{M_1+M_2}^{[1,2,3]} = \{\hat{t}_{M_1+M_2}^{[1,2,3]}, \hat{l}_{M_1+M_2}^{[1,2,3]}, \hat{d}_{M_1+M_2}^{[1,2,3]}\}$,$\hat{t}_{M_1+M_2}^{[1,2,3]}$为$M_1+M_2$时刻的吞吐量,$\hat{l}_{M_1+M_2}^{[1,2,3]}$为第95百分位延迟,$\hat{d}_t^x$为测试运行时长. $\hat{r}_{M_1+M_2}^{[1,2,3]}$是环境奖励,由式(5)得出,式中$\hat{o}_{t-1}^{x'}$为上一时刻的性能观测值,元组$x'$包含上一时刻Agent i.

$$r_t^i = R(\hat{o}_{t-1}^{x'}, \hat{o}_t^x) \tag{5}$$

相比于SA和JAM模型,JAPM模型继承了SA和JAM模型的优点和训练成果,所以也避免了两者的缺点:(1)SA模型只能进行同一类型的参数调优;(2)当Agent的数量N过大时,JAM模型算法复杂度过高. 因此,设置Agent的数量N<6.

3.5　DBT-MADDPG算法

DBT-MADDPG模型的完整算法流程如算法2所示.

算法2　分阶段式DBT-MADDPG算法

输入: Agent个数N,Critic网络学习率α,Actor网络的学习率β,目标网络的软更新参数τ,折扣因子γ,每个Agent的经验回放缓冲区大小M,批量样本的大小B,更新目标网络的时间间隔T,Agent概选因子P^i,各个阶段训练轮次M_1,M_2,M_3.

输出: Agent的联合动作$\hat{a}_t^x$.

```
初始化每个Agent的Actor网络θ^i和目标Actor网络θ'^i;
初始化每个Agent的Critic网络φ^i和目标Critic网络φ'^i;
初始化每个Agent的经验回放缓冲区D^i;
使用SA模型推荐单个智能体的动作;
使用JAM模型推荐联合动作;
使用JAPM模型推荐联合动作;
```

4　实验及讨论

4.1　实验设置

实验环境. 实验采用的处理器配置为Intel Core i7-12700HX,24核心,内存为16 GB,Python环境为3.8.10,数据库为PostGreSQL13. 选取了一组影响数据库性能的361个可调参数.

对比算法. 采用典型且具有代表性的算法:BestConfig、OtterTune、CDBTune+、QTune和ResTune,引言部分已做了详细介绍.

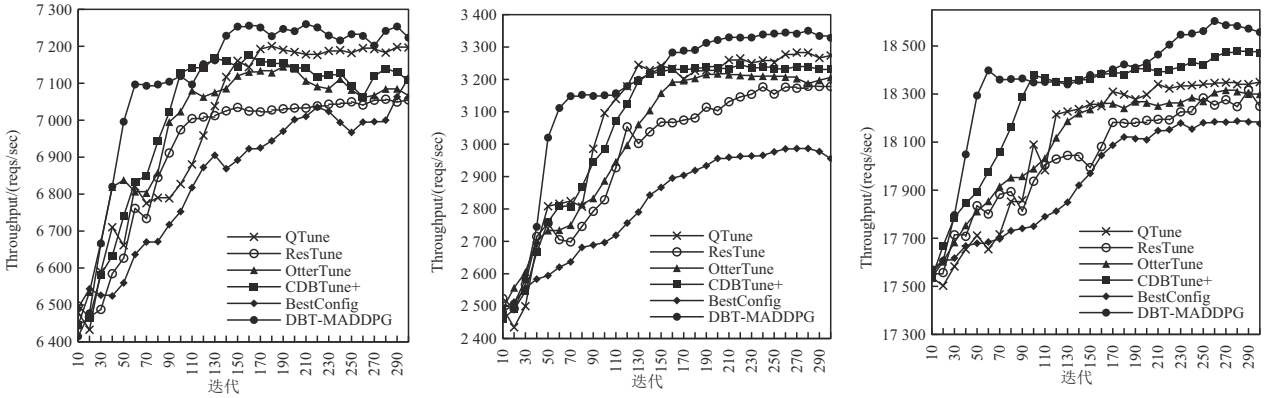
工作负载. 数据库性能测试采用几种典型的基准测试工具. 第一类是YCSB(Yahoo! Cloud Serving Benchmark)组件,YCSB旨在模拟基于主键的随机读取访问模式;第二类为BenchBase,包括一些典型的工作负载:TPC-C(Transaction Processing performance Council benchmark C)是一个常用的事务处理性能基准测试,旨在评估数据库系统在在线事务处理场景下的性能;Twitter可以模拟出实时社交媒体平台中用户的互动行为,这种负载类型可以帮助评估数据库系统在高并发、大规模社交媒体场景下的读取和写入性能,并发现潜在的性能瓶颈和优化机会.

参数设置. Agent 个数 N=3, Critic 网络学习率 α=0.001, Actor 网络的学习率←β=0.001, 目标网络的软更新参数 τ=0.002, 折扣因子 γ=0.9, 批量样本的大小 B=32, 更新目标网络的时间间隔 T=5, Agent 概选因子 P^i=0.7.

评估指标. 数据库性能评价指标为吞吐量 Throughput, 单位为 reqs/sec(request/seconds); 模型评估指标为(CC, PIR), 其中, 收敛轮次 CC(Convergence Cycle)表示模型在多少回合训练之后开始收敛, 性能提升比例 PIR(Performance Improvement Ratio)=提升的性能/初始性能.

4.2 模型性能比较

图 1 给出了在 YCSB-A(50% 读-50% 写)、TPC-C 以及 Twitter 三种工作负载测试下, BestConfig、OtterTune、CDBTune+、QTune、ResTune 以及 DBT-MADDPG 模型的性能对比. 所有模型调优的参数和训练轮次均相同, 调优参数个数为 60, 训练轮次 300, 其中, DBT-MADDPG 模型每个阶段包含 100 轮. 可以发现: 基于搜索的方法 BestConfig 模型表现最差; 基于贝叶斯的方法 OtterTune 和 ResTune 表现良好, 但是没有深度强化学习的方法 CDBTune+和 Qtune 表现优秀. 而 DBT-MADDPG 模型则优于上述方法, 在 YCSB-A, TPC-C 以及 Twitter 三种工作负载下均提供了更优秀的数据库参数配置, 在第一阶段, DBT-MADDPG 模型便能获得与其他方法同样优秀的表现, 并且比之更早找到一组较佳的配置, 且比主流算法快 17.85% 获得较佳配置. 在第二、三阶段, DBT-MADDPG 模型能继续优化其策略网络以及价值网络, 进一步提高数据库的性能.



(a) YCSB-A (b) TPC-C (c) Twitter

图 1 不同工作负载下不同模型吞吐量对比

表 1 展示了在 YCSB-B 工作负载下, 随着调节参数的增加, 模型之间的对比, 评估指标为(CC, PIR). 可以发现: 当参数个数为 20 时, 相比于其他模型, DBT-MADDPG 模型并不占优, 尤其是 Agent 的数量增多时, 其效果甚至不如 CDBTune+等方法, 原因是参数数量过少, 而 Agent 占用的系统资源过多, 并不利于数据库参数调优. 但随着参数数量增加, 尤其是当参数数量在 40~250 之间, DBT-MADDPG 模型将普遍优于其他算法. 当设置 Agent 的个数不同时, 随着参数数量的增多, DBT-MADDPG 模型的参数调优性能不同. 实验结果表明, 随着可调参数的增加, 一个 Agent 控制的参数推荐在 20~50 个之间, DBT-MADDPG 模型均能取得优秀的性能. 但是当参数超过 300 个时, 模型并不能达到收敛, 性能也达不到最优. 但相对于其他方法仍具有一定的优势.

5 结束语

本文提出一种面向数据库参数调优的协作型多智能体模型 DBT-MADDPG, 模型通过多个 Agent 相互协作和学习, 为数据库推荐合理的参数配置, 使得数据库在某一工作负载环境下, 性能得到迅速提升并得到最佳配置. 结果表明 DBT-MADDPG 模型相较于其他主流模型, 在调优参数数量增多的条件下, DBT-MADDPG 能在 SA 模型训练阶段迅速获得较佳配置, 由于每个 Agent 通过控制功能类似或者参数级别类似的参数集合, 在基于 JAM 和 JAPM 模型训练阶段能进一步提高数据库的性能, 验证了面向数据参数调优的协作型多智能体模型的有效性. 未来的工作将对 DBT-MADDPG 模型进行结构优化, 降低其模型时间复杂度, 泛化其对其他数据库系统的应用.

表1　在不同数量参数下模型的(CC,PIR)评价指标对比

参数个数	DBT-MADDPG				CDBTune+	QTune	ResTune	OtterTune	Bestconfig
	N=2	*N*=3	*N*=4	*N*=5					
20	(**31**, **5.25**)	(37, 5.03)	(42, 5.12)	(41, 5.18)	(**25**, 5.23)	(30,**5.45**)	(**24**,**5.44**)	(**28**,**5.89**)	(68,5.16)
40	(**45**, 6.43)	(47, **6.82**)	(50, 6.28)	(62, 5.21)	(**53**,6.54)	(63,6.27)	(74,**6.73**)	(68,6.45)	(83,5.64)
60	(65, 6.64)	(**62**, **6.87**)	(73, 6.12)	(78, 5.34)	(84,6.25)	(**73**,**6.75**)	(133, 5.55)	(92, 5.85)	(110,5.46)
80	(82, 6.54)	(75, 6.63)	(**73**, **6.92**)	(78, 6.38)	(98,6.41)	(**86**,**6.63**)	(158, 5.54)	(126, 5.32)	(154,6.13)
100	(123, 6.39)	(97, 6.74)	(**91**, **7.23**)	(**103**, **7.13**)	(**134**,6.64)	(157,6.21)	(187,6.03)	(167,6.41)	(243,5.23)
150	(168, 6.95)	(137, 7.24)	(125, 7.52)	(**124**, **7.95**)	(158,**6.82**)	(207,6.39)	(279,6.01)	(**154**,6.79)	(289,5.43)
200	(215, 6.83)	(206, 7.03)	(**197**, 7.31)	(201, **7.38**)	(**254**,6.52)	(288, 5.67)	(300+, 5.63)	(264,**6.73**)	(300+,5.56)
250	(300+, 5.43)	(300+, 6.58)	(**289**, **6.73**)	(**277**, **6.89**)	(300+, 5.14)	(300+, 5.13)	(300+, 5.33)	(300+, 5.23)	(300+, 4.73)
300	(300+, 5.36)	(300+, 5.84)	(300+, 6.37)	(300+, **6.61**)	(300+, 4.68)	(300+, 4.39)	(300+, 4.34)	(300+, 4.61)	(300+, 4.57)
350+	(300+, 5.69)	(300+, 5.35)	(300+, **6.04**)	(300+, **6.20**)	(300+, 4.43)	(300+, 4.25)	(300+, 4.13)	(300+, 4.87)	(300+, 4.82)

参考文献

[1] 李国良, 周煊赫, 孙佶, 等. 基于机器学习的数据库技术综述[J]. 计算机学报, 2020, 43(11): 2019-2049.
LI G L, ZHOU X H, SUN J, et al. A survey of machine learning based database techniques[J]. Chinese Journal of Computers, 2020, 43(11): 2019-2049. (in Chinese)

[2] ZHAO X Y, ZHOU X H, LI G L. Automatic database knob tuning: A survey[J]. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(12): 12470-12490.

[3] CEREDA S, VALLADARES S, CREMONESI P, et al. CGPTuner[J]. Proceedings of the VLDB Endowment, 2021, 14(8): 1401-1413.

[4] ZHU Y Q, LIU J X, GUO M Y, et al. BestConfig: Tapping the performance potential of systems via automatic configuration tuning[C]//Proceedings of the 2017 Symposium on Cloud Computing. New York: ACM, 2017: 338-350.

[5] ZHANG B, VAN AKEN D, WANG J, et al. A demonstration of the ottertune automatic database management system tuning service[J]. Proceedings of the VLDB Endowment, 2018, 11(12): 1910-1913.

[6] 李维刚, 甘平, 谢璐, 等. 基于样本对元学习的小样本图像分类方法[J]. 电子学报, 2022, 50(2): 295-304.
LI W G, GAN P, XIE L, et al. A few-shot image classification method by pairwise-based meta learning[J]. Acta Electronica Sinica, 2022, 50(2): 295-304. (in Chinese)

[7] ZHANG X Y, WU H, CHANG Z, et al. ResTune: Resource oriented tuning boosted by meta-learning for cloud databases[C]//Proceedings of the 2021 International Conference on Management of Data. New York: ACM, 2021: 2102-2114.

[8] LI G L, ZHOU X H, LI S F, et al. QTune[J]. Proceedings of the VLDB Endowment, 2019, 12(12): 2118-2130.

[9] ZHANG J, LIU Y, ZHOU K, et al. An end-to-end automatic cloud database tuning system using deep reinforcement learning[C]//Proceedings of the 2019 International Conference on Management of Data. New York: ACM, 2019: 415-432.

[10] TAN J, ZHANG T, LI F, et al. iBTune: Individualized buffer tuning for large-scale cloud databases[J]. Proceedings of the VLDB Endowment, 2019, 12(10): 1221-1234.

[11] DUAN S, THUMMALA V, BABU S. Tuning database configuration parameters with iTuned[J]. Proceedings of the VLDB Endowment, 2009, 2(1): 1246-1257.

[12] KANELLIS K, DING C, KROTH B, et al. LlamaTune[J]. Proceedings of the VLDB Endowment, 2022, 15(11): 2953-2965.

[13] LOWE R, WU Y, TAMAR A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments[C]// Proceedings of the 31st International Conference on Neural Information Processing Systems. New York: ACM, 2017: 6382-6393.

[14] 肖硕, 黄珍珍, 张国鹏,等. 基于SAC的多智能体深度强化学习算法[J]. 电子学报, 2021, 49(9): 1675-1681.
XIAO S, HUANG Z Z, ZHANG G P, et al. Deep reinforcement learning algorithm of multi-agent based on SAC[J]. Acta Electronica Sinica, 2021, 49(9): 1675-1681. (in Chinese)

[15] 饶宁, 许华, 蒋磊, 等. 基于多智能体深度强化学习的分布式协同干扰功率分配算法[J]. 电子学报, 2022, 50(6):

1319-1330.
RAO N, XU H, JIANG L, et al. Allocation algorithm of distributed cooperative jamming power based on multi-agent deep reinforcement learning[J]. Acta Electronica Sinica, 2022, 50(6): 1319-1330. (in Chinese)

参考文献

李江敏 男,1997 年 2 月出生于陕西省咸阳市.现为成都信息工程大学软件工程学院硕士研究生.主要研究方向为人工智能数据库.
E-mail: 923749851@qq.com

乔少杰 男,1981 年 10 月出生于山东省招远市.博士,现为成都信息工程大学软件工程学院教授.主要研究方向为人工智能数据库、时空数据库、机器学习.中国电子学会会员编号:E190157673M.
E-mail: sjqiao@cuit.edu.cn