上海人工智能实验室
Shanghai Artificial Intelligence Laboratory

智能缓存（Learned Cache）

陈 榕

Shanghai AI Laboratory

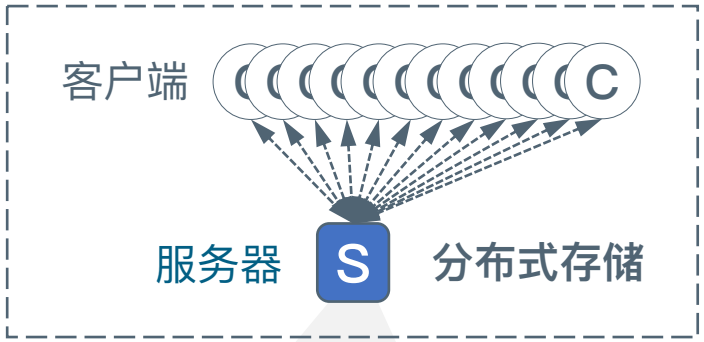
2021年6月15日

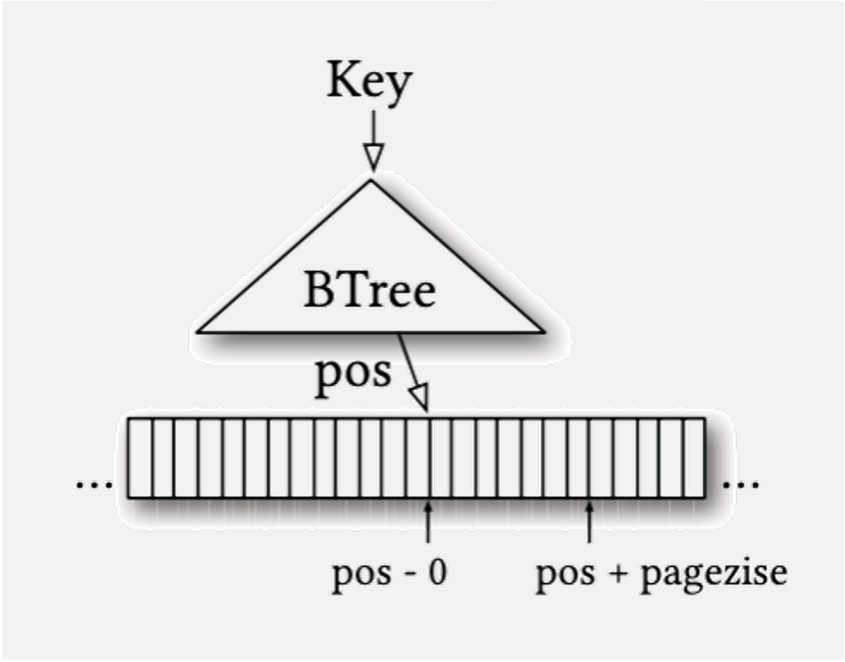
合作者：魏星达、陈海波等

键值存储系统

键值存储（Key-Value Store, KVS）

- 大量分布式应用的存储基础
  - 事务处理/OLTP、数据分析/OLAP
  - 电子商务、商务智能/BI
  - 机器学习、神经网络平台
- 键值存储模型
  - **Model** (key) → pos　// e.g., B$^+$Tree
  - **KVS** (pos) → value

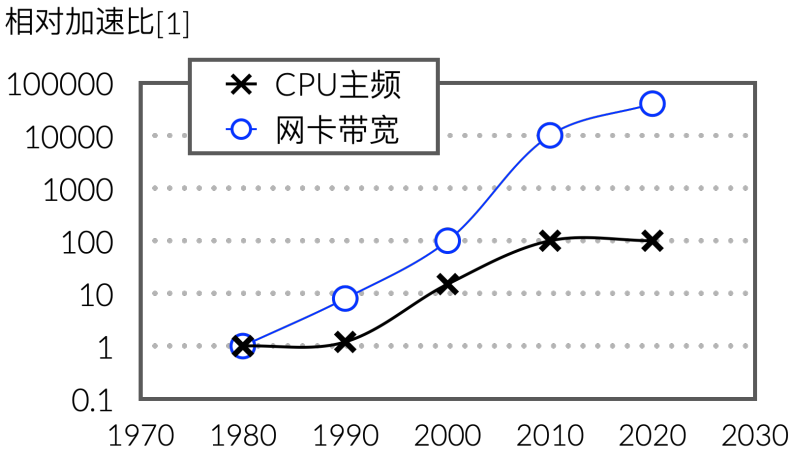




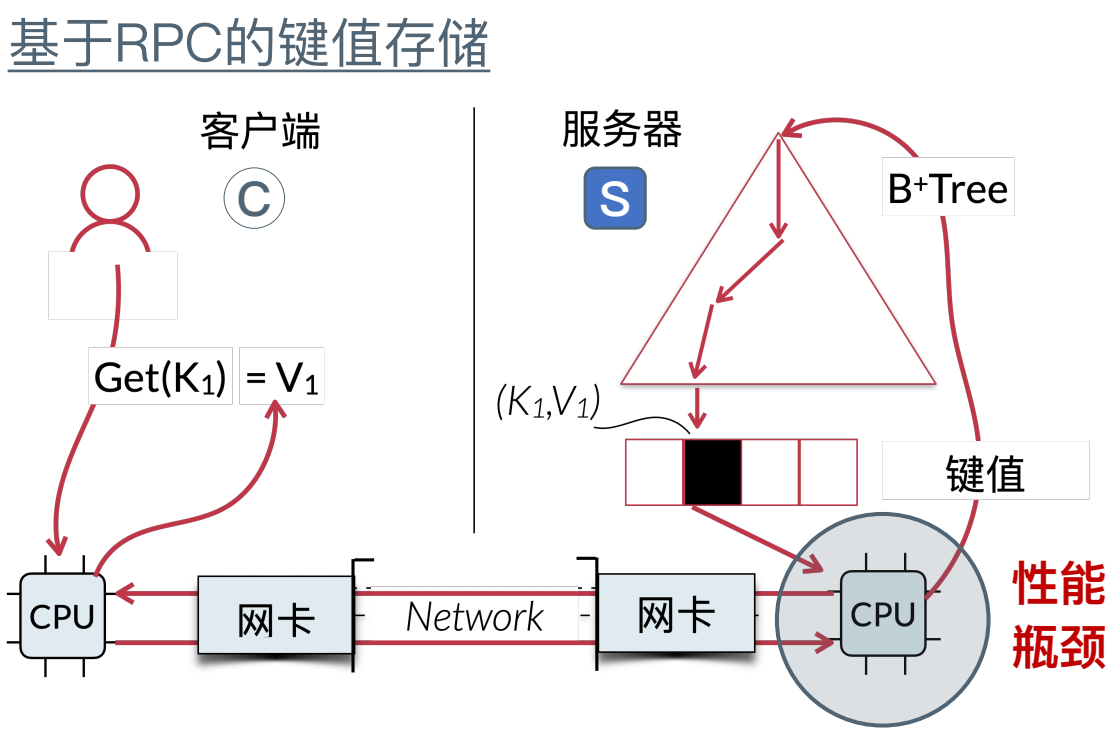
基于RPC的分布式键值存储

传统基于RPC的键值存储设计无法充分利用RDMA网络的高性能

- 利用Two-sided RDMA实现高性能RPC
- 服务器CPU会成为键值存储的性能瓶颈
- CPU频率增长落后于网卡带宽增长



来自 StRoM @Eurosys'20



基于One-sided RDMA的分布式键值存储

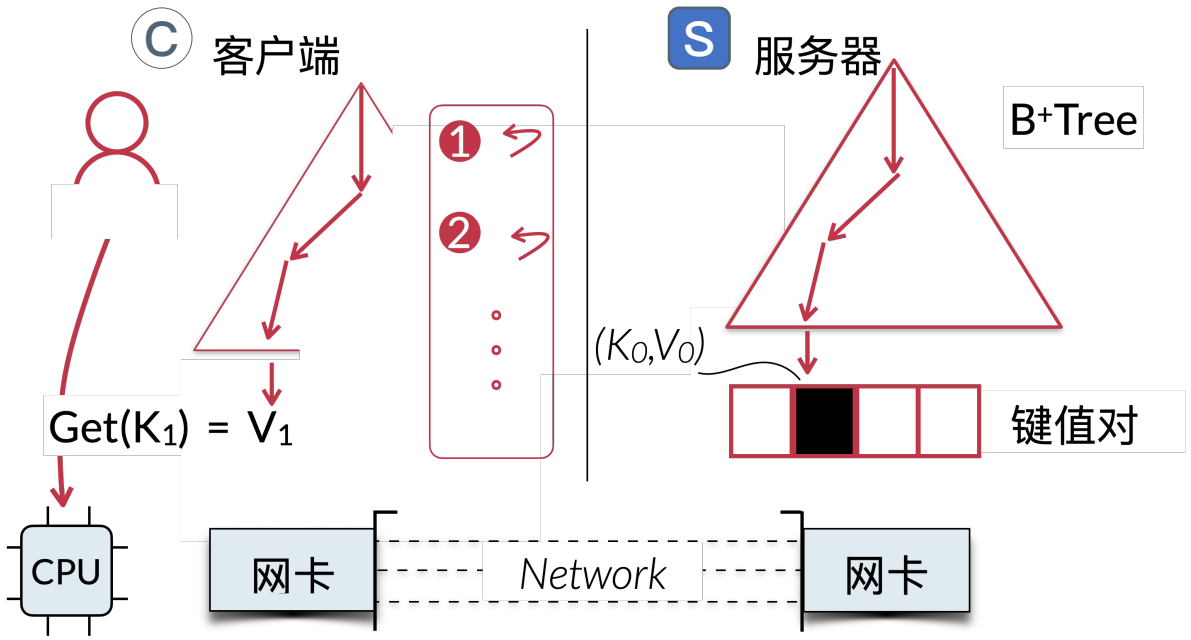
使用One-sided RDMA遍历B树索引会造成大量额外网络操作

- One-sided RDMA 语义有限：READ、WRITE
- 单个RDMA操作只能读取一层B树节点

分布式存储性能直接受限于网络操作数

- **1次**键值操作需要$O(\log(n))$**次**网络操作
- 如，100M键值数据，1次Get需要7次RDMA

基于One-sided RDMA的键值存储

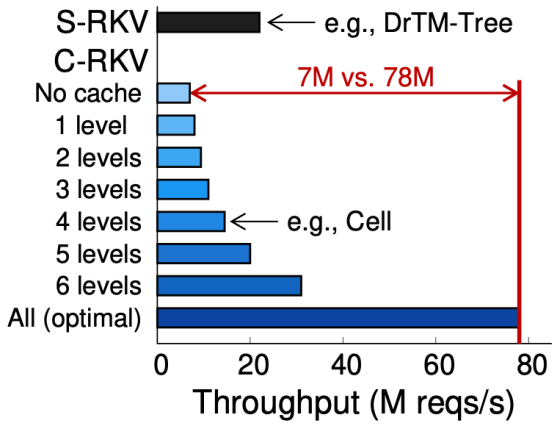


索引缓存优化（Index Caching）

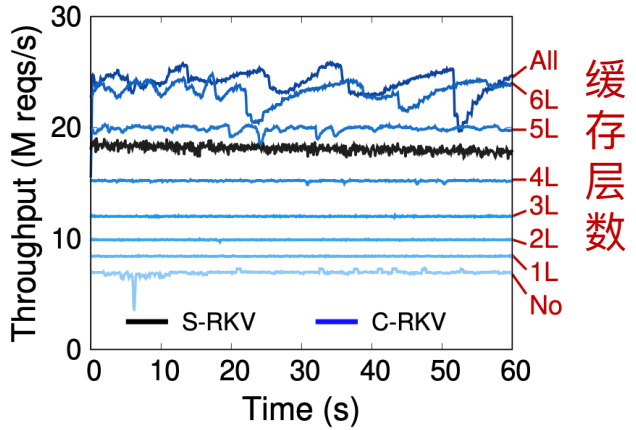
索引缓存优化【DrTM[SOSP15] CELL[ATC16] FaRMv2[SIGMOD19]】

- 客户度缓存（同构）索引，降低遍历索引开销
- B树不适合索引缓存：
  - 整树缓存占有客户端内存巨大（100M键值需要~650MB）
  - 动态场景（插入/删除）造成缓存频繁失效，性能颠簸严重

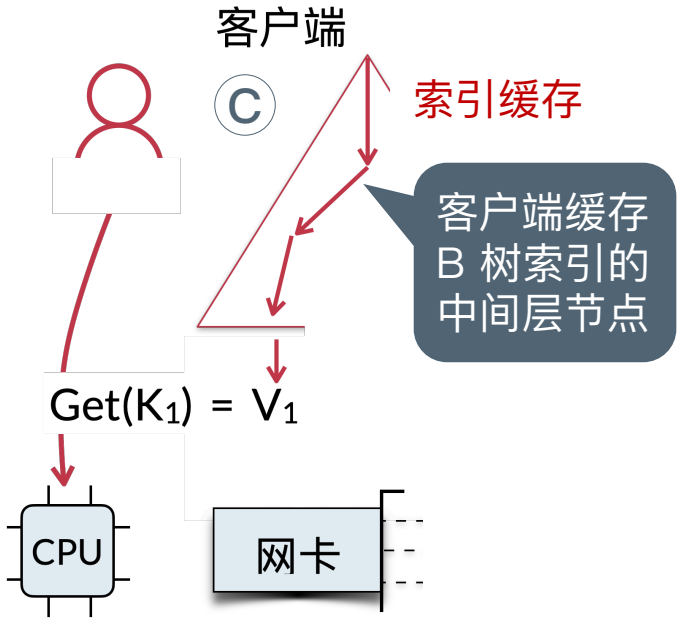
静态负载



动态负载



基于One-sided RDMA的键值存储



现有基于RDMA的键值存储设计空间

① 基于Two-sided RDMA键值存储

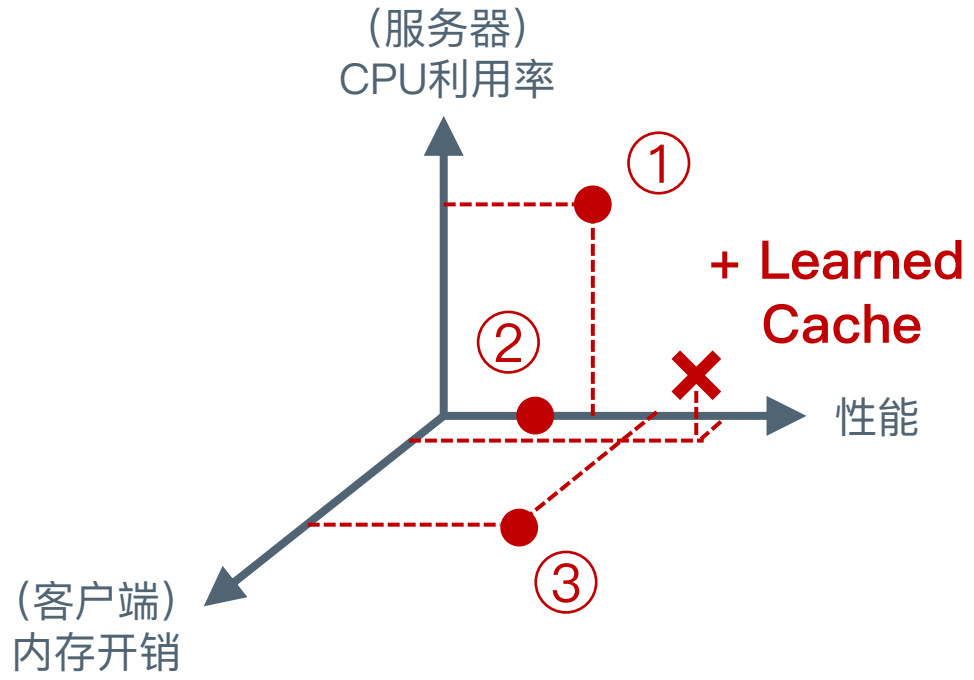
- 性能有限，且受限于服务器CPU

② 基于One-sided RDMA键值存储

- 性能有限，浪费RDMA网络资源

③ +（树结构）索引缓存优化

- 客户端内存开销大，动态负载性能颠簸



智能索引/Learned Cache：基于机器学习(ML)的新型索引缓存

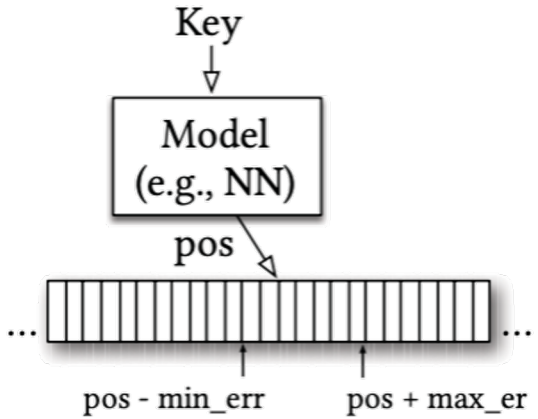
基于机器学习的智能索引（Learned Index）

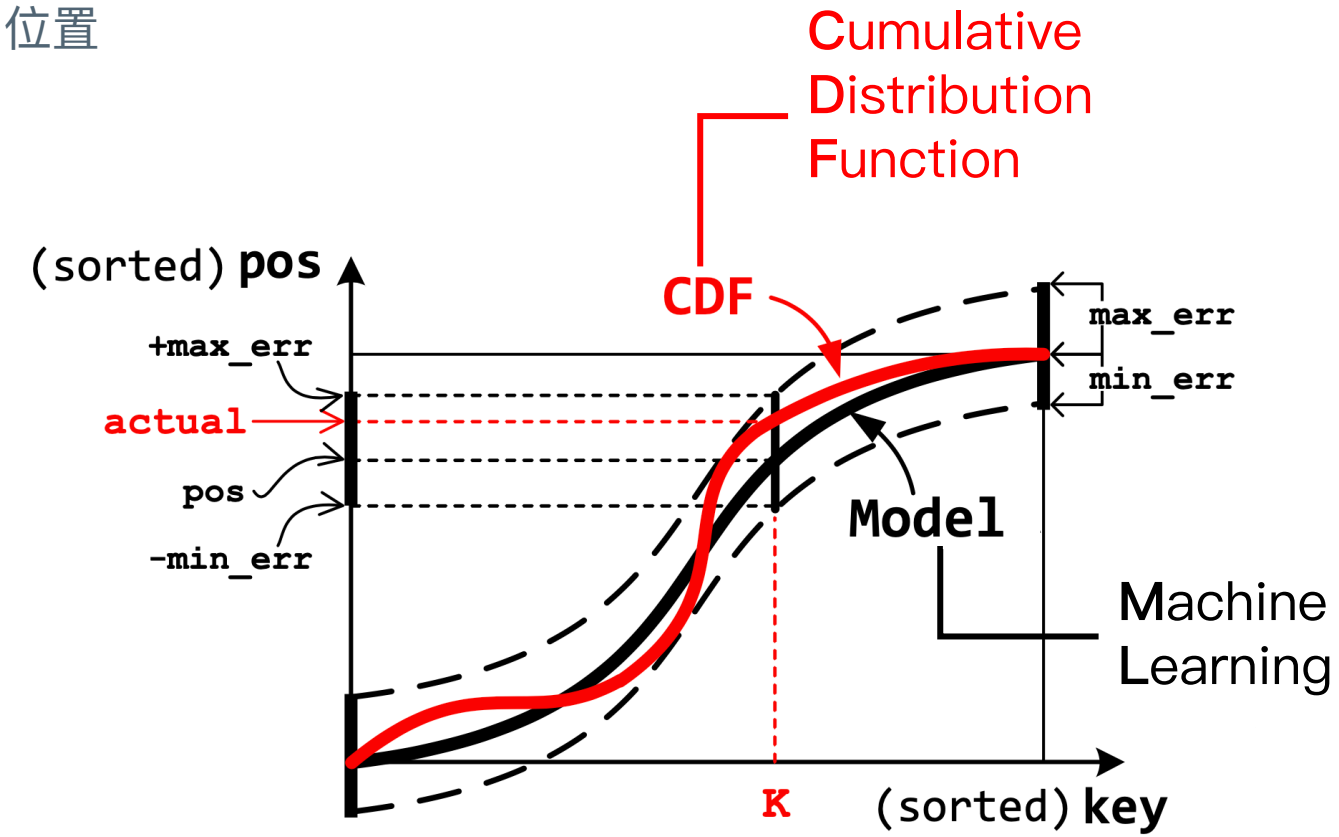
智能索引【Learned Index[SIGMOD18]】

- 使用机器学习模型（代替树结构）预测位置
  - MLmodel (key) → pos

    e.g., Linear Regression/LR

    Neural Nets/NN
- 训练ML模型逼近CDF(键–位置)

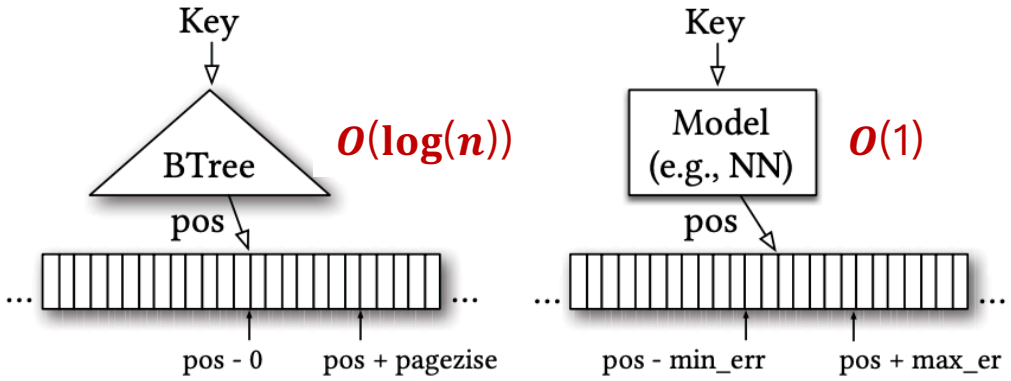




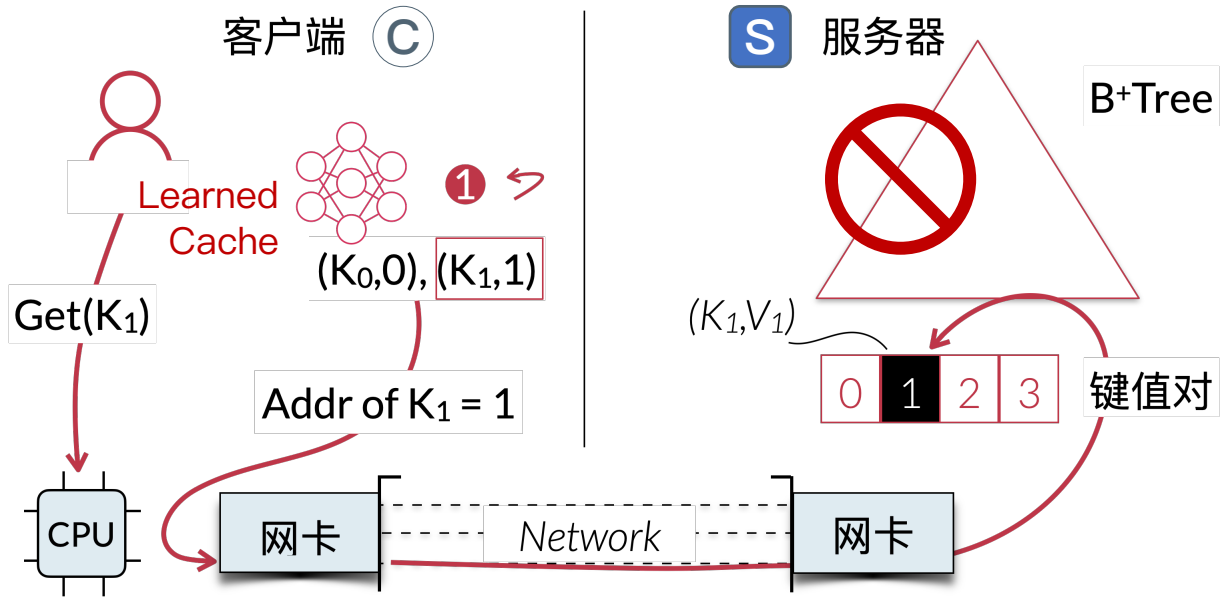
XStore：利用智能缓存加速RDMA键值存储

总体思路：使用机器学习模型构建索引缓存，“用计算换网络”

- ML模型缓存优点：占用空间极少，无需整树遍历
- RDMA网络操作：$O(\log(n)) \rightarrow O(1)$
- 低内存开销 + 高查询效率



基于智能缓存的RDMA键值存储



XStore：利用智能缓存加速RDMA键值存储

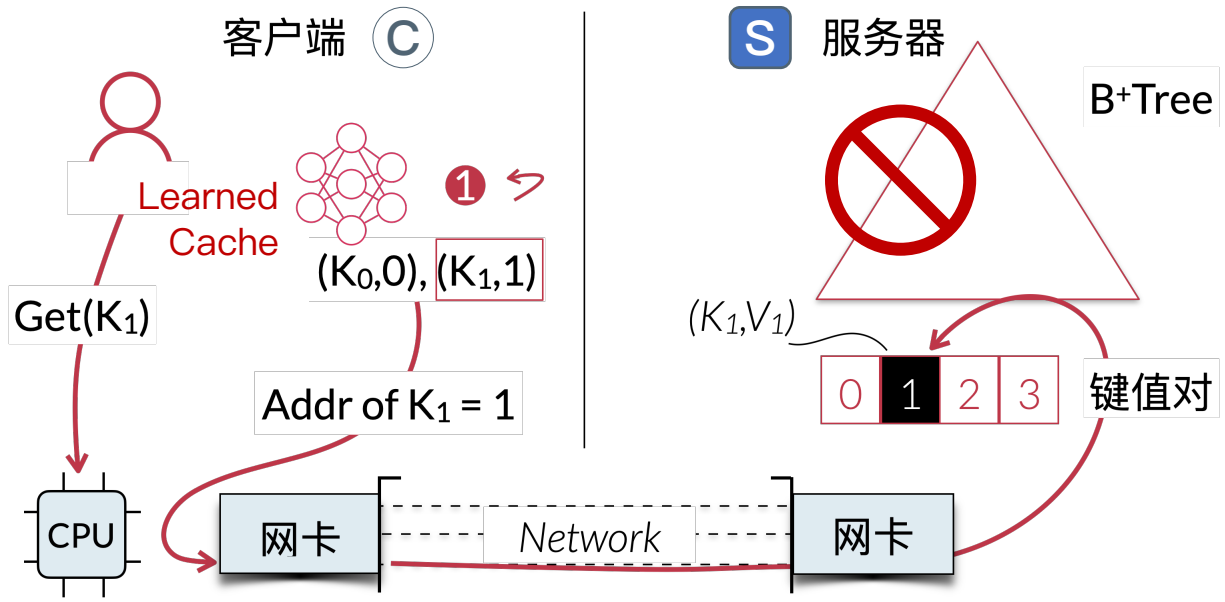
总体思路：使用机器学习模型构建（异构）索引缓存，“用计算换网络”

- ML模型缓存优点：占用空间极少，无需整树遍历
- RDMA网络操作：$O(\log(n)) \rightarrow O(1)$
- 低内存开销 + 高查询效率

技术挑战：智能缓存如何支持动态负载?

- 动态操作（插入）如何支持?
- 客户端智能缓存应如何更新?
- 机器学习模型如何重新训练?

基于智能缓存的RDMA键值存储



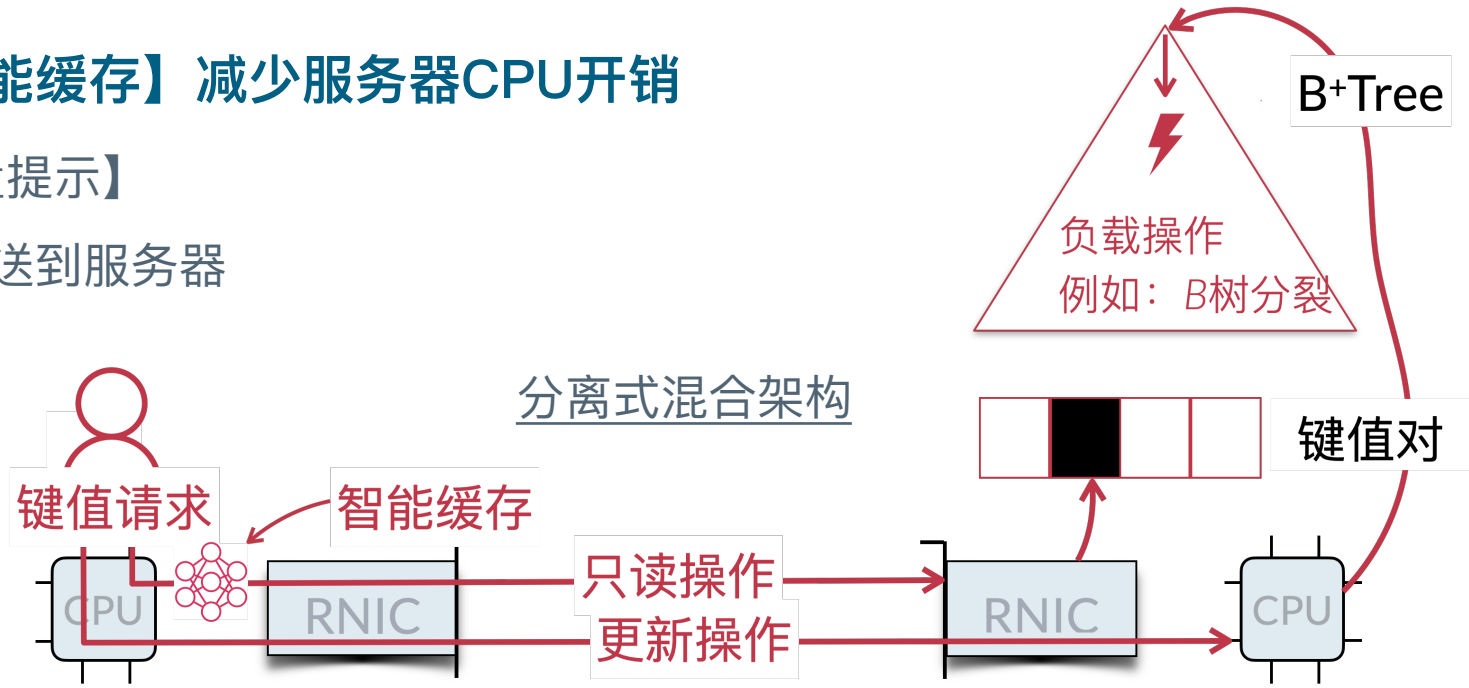
技术挑战一：动态操作（插入）如何支持？

分离式混合架构

- 只读操作：借助智能缓存，客户端使用1次One-sided RDMA完成
- 更新操作：使用1轮RPC操作（Two-sided RDMA），发送到服务器端完成

POSITION HINT优化：使用【智能缓存】减少服务器CPU开销

- 客户端利用智能缓存计算【位置提示】
- 【更新操作+位置提示】一起发送到服务器
- 提升动态负载性能~50%



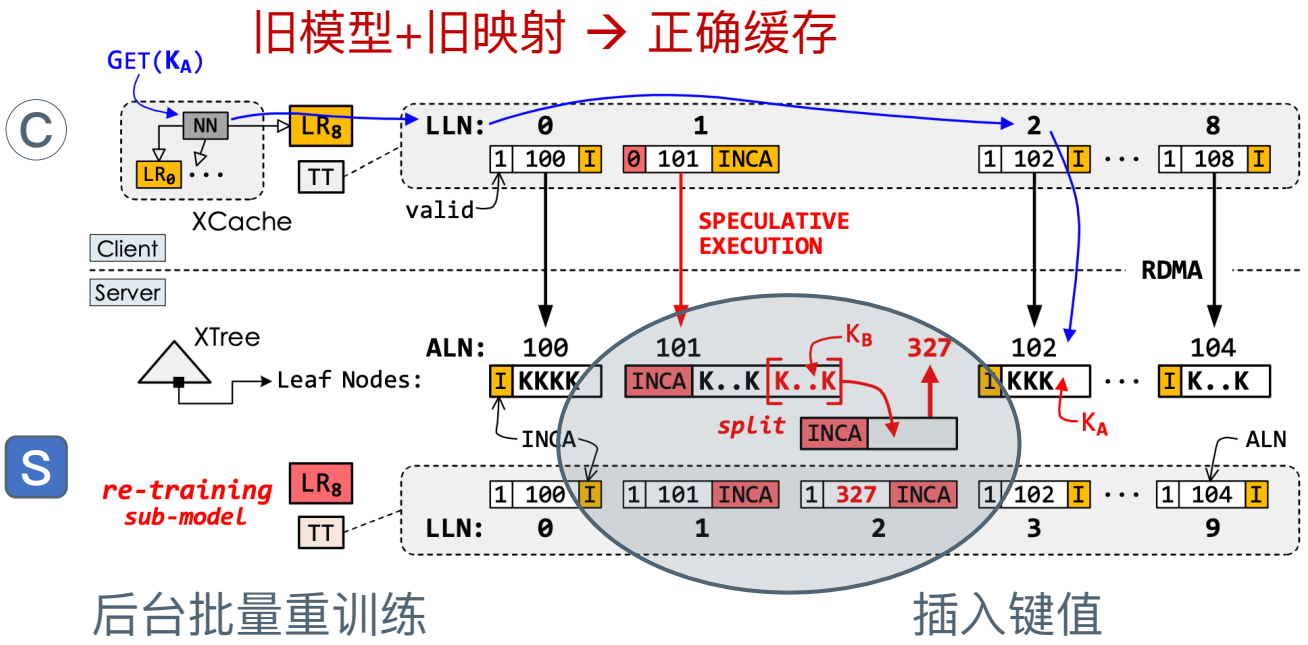
技术挑战二：客户端智能缓存应如何更新？

解耦【模型训练】和【缓存更新】，支持后台训练和延迟更新

- 引入地址映射表：逻辑地址 → 实际地址；ML模型：Map(Model(key)) → 实际地址
- 客户端：使用匹配的模型和映射表（即使不是最新的）可以获得正确缓存信息（实际地址）
- 边界情况：访问新插入键值对，退化成RPC实现

SPECULATIVE EXECUTION优化

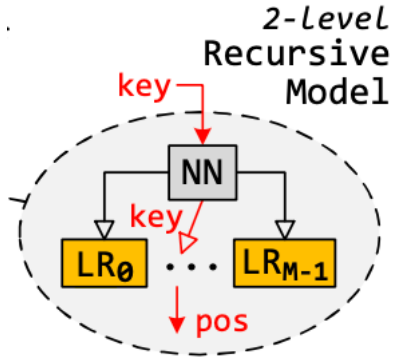
- 50%概率不增加额外开销
- 50%概率增加一次RDMA操作
- 提升动态负载性能～23%



技术挑战三：机器学习模型如何重新训练？

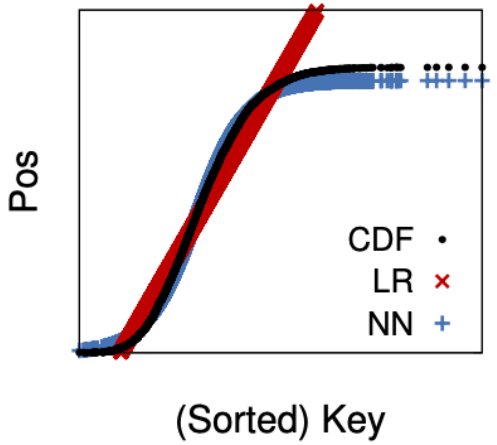
两层混合机器学习模型：支持局部重训练（partial retraining）

- 上层（Top-model）：高精度模型（Neural Nets，NN），减少传输量
- 下层（Sub-model）：低精度模型（Linear Regression，LR），快速重训练

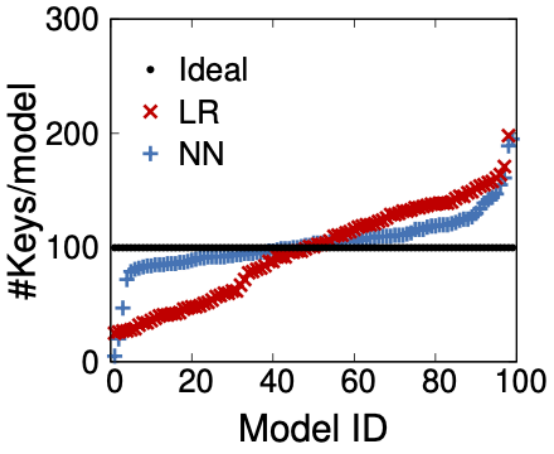


真实数据集（lognormal分布）

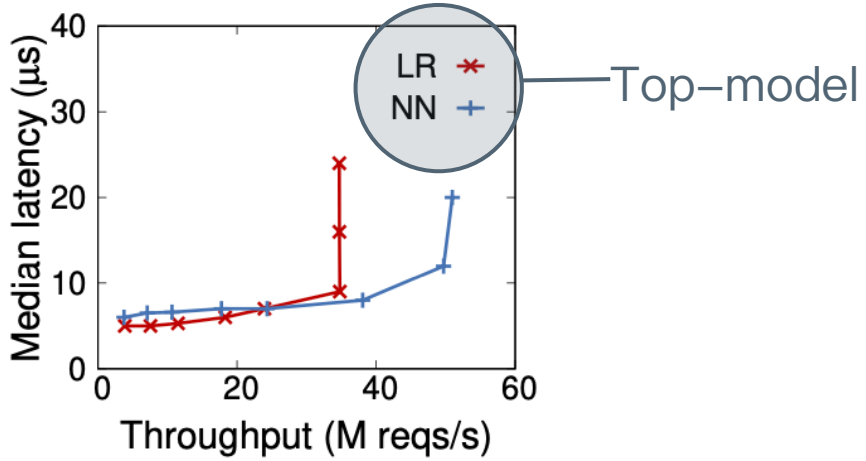
更精确



更平衡



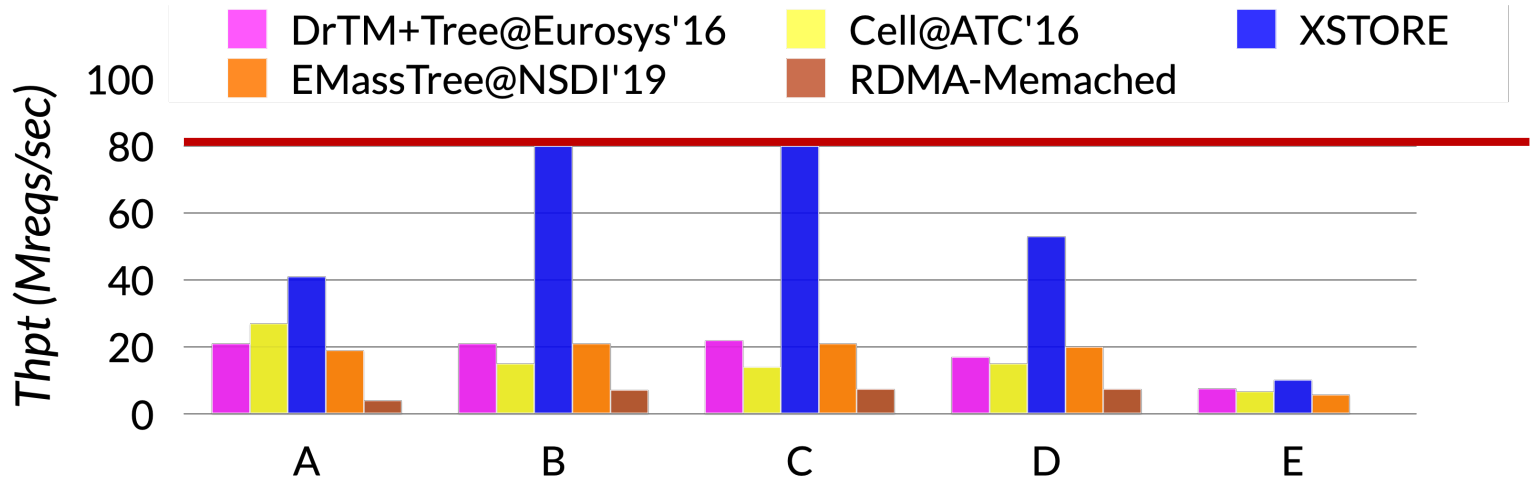
更高性能



基准性能测试

YCSB基准测试（A～E）：100M键值对（8B键+8B值），Uniform/Zipfan分布

- Two-sided RDMA设计：DrTM+Tree[EuroSys16] EMassTree[NSDI19] RDMA-Memcached
- One-sided RDMA设计：Cell[ATC16] XStore



Workload	A	B	C	D	E
R(S)/(U/I)	50:50	90:10	100:0	95:5	95:5

RDMA网卡性能极限

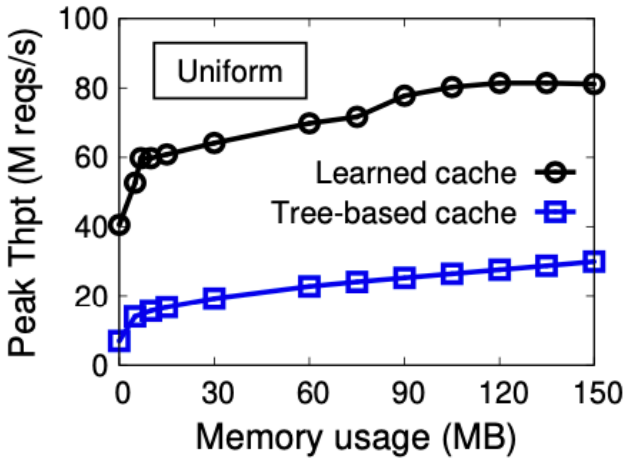
只读负载：80M reqs/s
性能提升3.7～5.9X

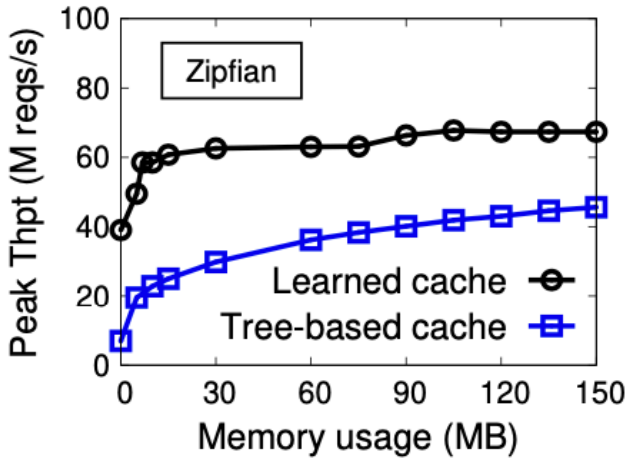
动态负载：53M reqs/s
性能提升2.7～3.5X

其他测试

客户端内存开销

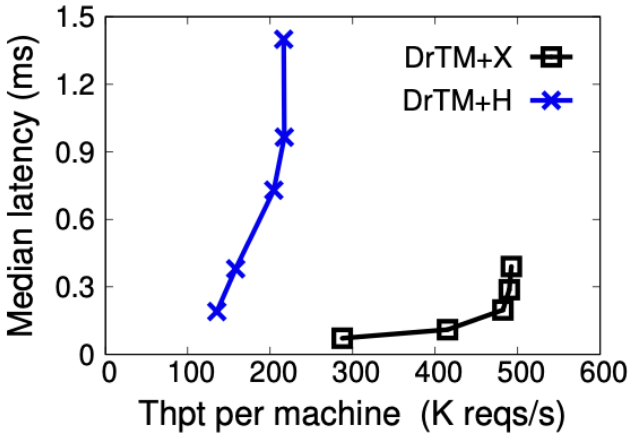
- ML模型大小：6.8MB
- 减少99%内存，性能下降20%
  （+1次 RDMA读取映射表）





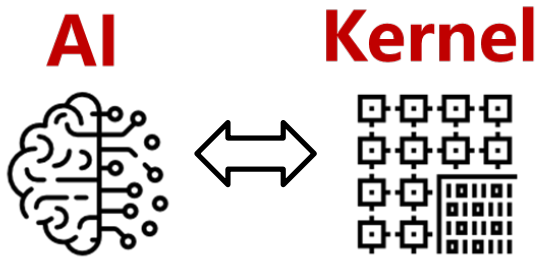
数据库性能提升

- vs. 基于RPC设计
- TPC-C基准测试
- ～2.3X性能提升



总结

人工智能为系统软件研究带来新机遇：人工智能赋能系统软件



尝试一：智能方法应用于分布式存储（Learned Cache）

- 虽优点明显、但挑战严峻
- 人工智能方法需要与系统软件方法紧密配合
- 成果发表在OSDI（操作系统领域旗舰会议），并获ACM Transactions on Storage邀稿
- 开源地址：https://github.com/SJTU-IPADS/xstore

尝试二：智能方法应用于并发控制协议（Learned Concurrency Control）

- 构建细粒度策略空间，选择并创造最优并发控制协议
- 参见OSDI论文“Polyjuice: High-Performance Transactions via Learned Concurrency Control”

感谢 提出宝贵意见

https://www.shlab.org.cn
Shanghai Artificial Intelligence Laboratory