软件学报专刊gt003

支持虚拟机动态迁移的自适应SSD缓存系统

作者：唐震、吴恒、王伟、魏峻、黄涛

软件工程技术研发中心

中国科学院软件研究所

昆明@2017/11/4

研究背景：固态盘在存储系统中具有重要地位

- 固态盘（Solid State Disk, **SSD**）是一种新型存储介质

根据Gartner报告[1]：

- SSD迎来**爆发式增长**（2015同比增长51%），到2017年，SSD市场将比2014年增长5倍
- 众多主流存储服务提供商推出了基于SSD的存储解决方案
- SSD成为存储系统中**重要的组成部分**

[1] Gartner, *Magic Quadrant for Solid State Arrays*. 2015.

研究背景：固态盘在存储系统中具有重要地位

- 固态盘与机械硬盘（Hard Disk Drive, HDD）相比

优势：

- 优异的随机读写速率（100~1000倍）
- 快速的顺序读写速率（5~10倍）

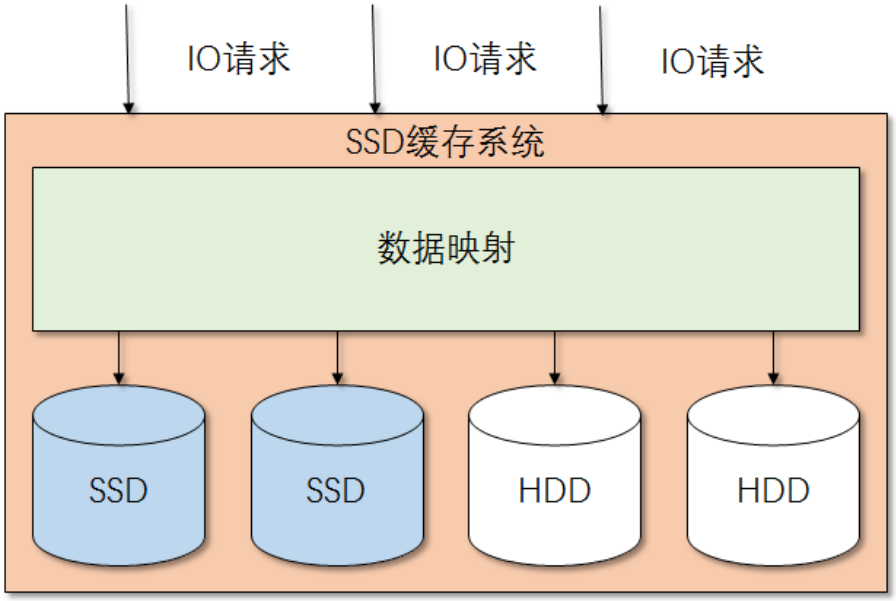
劣势：

- 单位价格较高，每GB价格约为HDD的3~10倍
- 闪存的擦写次数有限，SSD寿命相对较短

V.S.

研究背景：虚拟化环境下SSD缓存系统应用模式

- SSD的一种典型应用场景是作为HDD缓存，构成SSD缓存系统，兼顾了性能与价格，在虚拟化环境下得到广泛应用
- 主流虚拟化厂商都提供了支持
  - Amazon EC2、Microsoft Azure、阿里云



- SSD缓存系统由少量SSD作为大量HDD的读写缓存（通常为1:10左右）
- 数据映射模块负责维护SSD缓存数据
- 虚拟机的IO请求会先到达数据映射模块，再确定访问SSD或HDD

研究背景：存储资源需要得到合理调配

- SSD缓存系统中每秒I/O操作数（IOPS）和带宽是**稀缺资源**
  - IOPS和带宽直接反映了IO性能
  - 云服务提供商对不同性能的磁盘进行了差异化标价，需要合理规划IOPS和带宽资源
- 云环境下，IOPS和带宽对应用性能影响至关重要
  - 大数据处理类应用：Hadoop是**带宽**敏感的；基于Hadoop生态的上层应用（如HBase等）是IOPS敏感的；
  - 分布式协同类应用：ZooKeeper等应用是IOPS敏感的；
  - 数据库集群：MySQL，MongoDB等是**带宽和IOPS**敏感的；
  - ……
- SSD缓存资源需要得到**合理调配**

研究背景：SSD资源调度至关重要

- 然而，SSD的容量规划和IOPS、带宽使用的公平性之间存在矛盾
  - SSD缓存是先来先服务的
  - 与SSD缓存容量相比，IOPS和带宽较难保证公平性
  - 当工作负载对SSD服务能力的需求超过其供给时，会产生资源争用，极大影响性能
- 现有工作通常从缓存命中率[1]和工作集[2]角度评价SSD资源管理的效果，实现多目标的SSD缓存管理，最终以容量作为调整目标

[1] Koller R, Mashtizadeh A J, Rangaswami R. Centaur: Host-Side SSD Caching for Storage Performance Control; proceedings of the Autonomic Computing (ICAC), 2015 IEEE International Conference on, F 7-10 July 2015, 2015 [C].

[2] Arteaga D, Cabrera J, Xu J, et al. CloudCache: on-demand flash cache management for Cloud Computing [M]. Proceedings of the 14th Usenix Conference on File and Storage Technologies. Santa Clara, CA; USENIX Association. 2016: 355-69.

面临挑战：需要全局综合考虑SSD服务能力

- SSD缓存同样需要从存储介质角度考虑服务能力，以IOPS和带宽作为调整目标
  - SSD的读写IOPS和带宽存在上限
  - 在满足Miss率或缓存大小的前提下，这些虚拟机仍然可能会对IOPS和带宽资源产生争用，此时增加虚拟机的缓存容量收益较小
  - 这一资源争用无法通过调整SSD资源分配解决，需要借助虚拟机迁移来缓解
  - 现有工作未能充分考虑到SSD作为存储介质具有的天然限制，进行调度时难以权衡各关键因素
- 亟需一种以SSD服务能力（IOPS和带宽）为导向，以虚拟机迁移为手段的SSD资源调度方法

面临挑战：需要综合考虑云应用特点

- 云应用是虚拟化环境下的主要服务模式，云应用对IO的需求是以对IOPS和带宽的需求体现的。
- 云应用本身对虚拟机放置存在一定的倾向性，并会反映到IOPS和带宽的需求上来。例如
  - **大数据处理集群中的节点需要频繁访问其他节点的数据**
  - **分布式云存储集群中的节点需要满足副本和容错需求**
- 现有工作[1][2]在进行SSD缓存资源管理时未从云应用视角考虑
- 亟需一种综合考虑云应用特点、需求和虚拟机放置倾向性的SSD资源调度方法

[1] Shamma M, Meyer D T, Wires J, et al. Capo: recapitulating storage for virtual desktops [M]. Proceedings of the 9th USENIX conference on File and stroage technologies. San Jose, California; USENIX Association. 2011: 3-.

[2] Mesnier M, Chen F, Luo T, et al. Differentiated storage services [M]. Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. Cascais, Portugal; ACM. 2011: 57-70.

面临挑战：虚拟机迁移是重量级操作

- 虚拟机动态迁移的开销较大，需要合理规划
  - 迁移需要占用物理主机的IO资源和网络资源，降低虚拟机性能
  - 对被迁移虚拟机而言，存在短暂的服务质量降低
- 不合理的迁移顺序有可能会引入不必要的迁移
- 不合理的迁移频率会引发性能问题
  - 频率过低达不到调整的效果，无法实现性能提升
  - 频率过高会带来巨大的性能开销
- 亟需一种能够进行虚拟机优化放置并合理规划动态迁移的SSD资源管理方法

基于虚拟机动态迁移的SSD缓存调度

- 问题
  - 由SSD缓存管理系统触发的SSD负载均衡，保障虚拟机共享IOPS和带宽资源的公平性
- 主要场景
  - 大数据处理应用（Hadoop）和分布式协同应用（ZooKeeper）
- 主要贡献
  - 应用特点和需求的刻画（**挖掘虚拟机间关联**）
  - 自适应闭环（**应对云应用的拓扑和负载变化**）
  - 以SSD缓存服务能力为导向的虚拟机动态迁移（**解决SSD的服务能力争用问题**）
  - 优化迁移顺序和迁移时机（**降低迁移开销**）

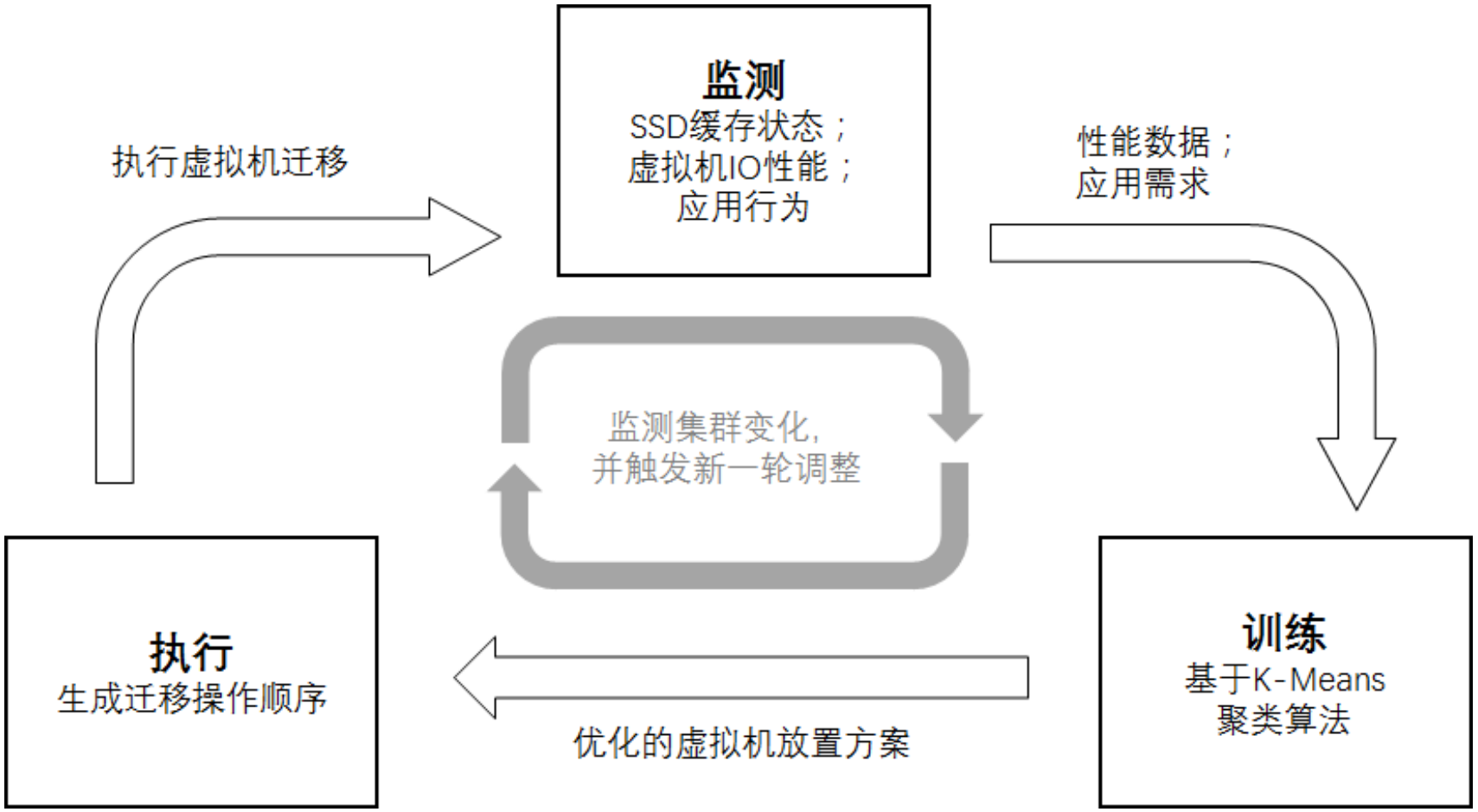
应用特点和需求的刻画

- 从SSD缓存出发，以应用的特点和需求指导虚拟机迁移
  - SSD缓存需求：满足应用对缓存容量、带宽和IOPS的需求
  - SSD缓存负载均衡：满足SSD缓存服务能力的约束
  - 应用关联和虚拟机放置倾向性
- 将虚拟机放置倾向性归结到两个维度的需求
  - 隔离需求：倾向于将虚拟机放置在不同的Hypervisor上
    - 满足副本或热备等需求
  - 内聚需求：倾向于将虚拟机放置在同一Hypervisor上
    - 获得快速的虚拟机间通讯，满足数据本地性需求

SSD缓存服务能力导向的虚拟机动态迁移

- 自适应闭环
  - 持续监控虚拟机集群
  - 基于聚类的方法，计算最优虚拟机放置方案
  - 计算最优迁移顺序，执行迁移
- 触发新一轮调整的时机
  - 监测到虚拟机集群规模变化
  - 监测到应用拓扑变化
  - 监测到应用行为模式改变
- 基于聚类的放置方案计算

SSD缓存服务能力导向的虚拟机动态迁移

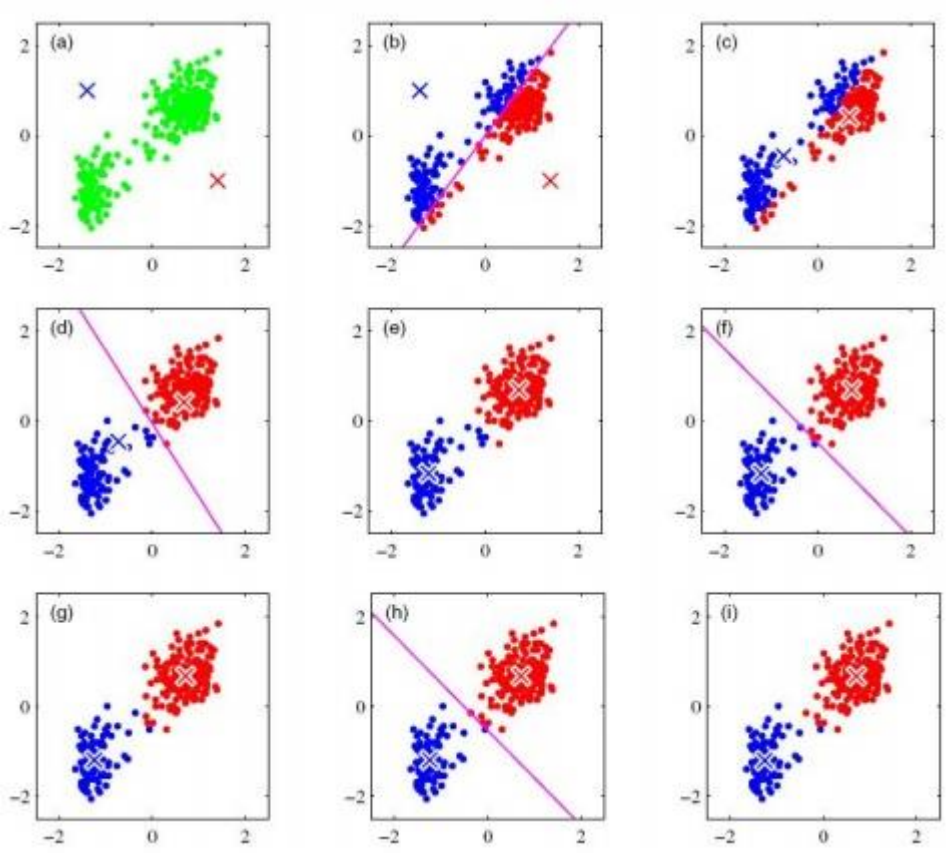


计算虚拟机放置方案

- 计算放置方案存在**状态空间爆炸**的问题，适合采用机器学习的方法解决，平衡**优化效果**和**执行效率**
- 考虑到虚拟机的IO访问和放置倾向**存在共性**，采用**K-Means算法**
- 按照三个原则进行聚类
  - SSD缓存负载均衡：满足虚拟机的缓存需求，同时满足SSD的约束，最大化资源利用率
  - 隔离：满足虚拟机分开放置的需求
  - 内聚：满足虚拟机共同放置的需求

计算虚拟机放置方案

- K-Means聚类算法
  - 经过多次迭代，根据距离实现聚类



计算虚拟机放置方案

- 特性选择
  - 虚拟机IO性能：CPU处理IO的时间；磁盘读写带宽和IOPS
  - SSD缓存特性：容量；利用率；读写操作数；Miss率
  - 网络特征：TCP连接数；网络带宽
  - 内聚程度：通过虚拟机间的网络和IO带宽计算得到
  - 隔离程度：基于默认策略，配合基于启发式规则的先验知识来确定
- 针对聚类算法的优化
  - 在进行初始质心选择时，选取所有应用中虚拟机隔离程度最高的最多X台虚拟机(X为Hypervisor数量)
  - 在进行迭代的聚类操作时，额外考察了每个聚类中的虚拟机的IO负载以及对缓存容量，SSD带宽以及IOPS的需求

计算虚拟机放置方案

- 启发式规则：目前关注ZooKeeper和Hadoop
  - ZooKeeper（同样适用于基于选举的应用）
    - Leader和Follower倾向于分开
    - Follower倾向于均匀分布
    - 保证单一Hypervisor失效时剩余存活的节点仍然满足Quorum
  - Hadoop（同样适用于其他大数据处理类应用）
    - HDFS的NameNode和备份NameNode倾向于分开
    - DataNode倾向于均匀分布
    - 计算节点倾向于与HDFS放置在同一Hypervisor上，保证数据本地性

迁移顺序计算

- 聚类结果并不包含聚类到物理机的映射，需要先映射到当前分配方案
- 当前分配方案P_0
  - 一个虚拟机放置的集合
  - $P_0 = \{H_1, H_2, \dots, H_n\}$
  - $H_i = \{VM_1, VM_2, \dots, VM_n\}$
- 聚类结果C
  - $C = \{C_1, C_2, \dots, C_n\}$
  - $C_i = \{VM_1, VM_2, \dots, VM_n\}$
- 映射算法主要计算了集合间的最大匹配（计算交集最大的映射方案）

迁移顺序计算

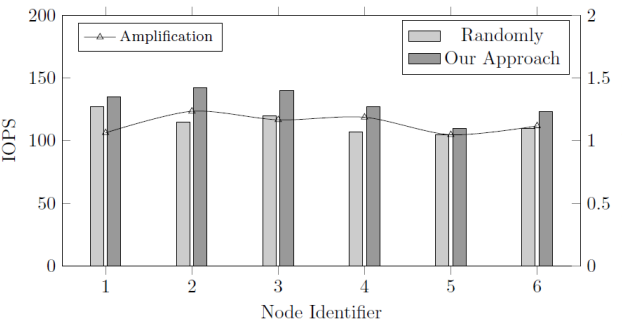
- 得到最大匹配之后计算每个Hypervisor上的变换顺序
  - $P_0 = \{H_1, H_2, H_3\}; H_1 = \{VM_1, VM_2, VM_3\}; H_2 = \{VM_4, VM_5, VM_6\}; H_3 = \{VM_7, VM_8, VM_9\}$
  - $C = \{C_1, C_2, C_3\}; C_1 = \{VM_1, VM_4, VM_5\}; C_2 = \{VM_2, VM_3, VM_7\}; C_3 = \{VM_6, VM_8, VM_9\}$
  - $MaxMapping = \{(H_1, C_2), (H_2, C_1), (H_3, C_3)\}$
  - $Op(1) = \{Out(1), In(7)\}; Op(2) = \{Out(6), In(1)\}; Op(3) = \{Out(7), In(6)\}$
- 得到操作顺序之后匹配In和Out操作，得到虚拟机迁移方案
  - $Migration = \{H_1.VM_1 \rightarrow H_2, H_2.VM_6 \rightarrow H_3, H_3.VM_7 \rightarrow H_1\}$
- 最终确定迁移顺序的原则是降低对**服务质量**的影响

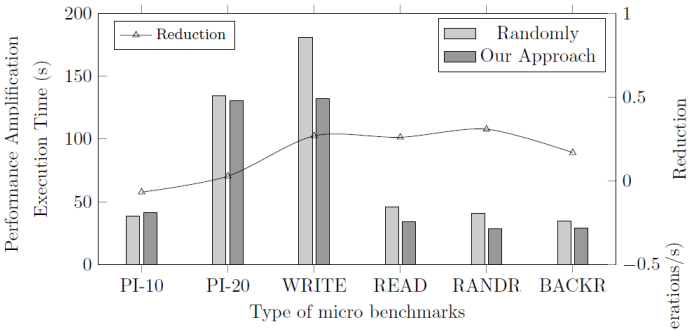
实验设计

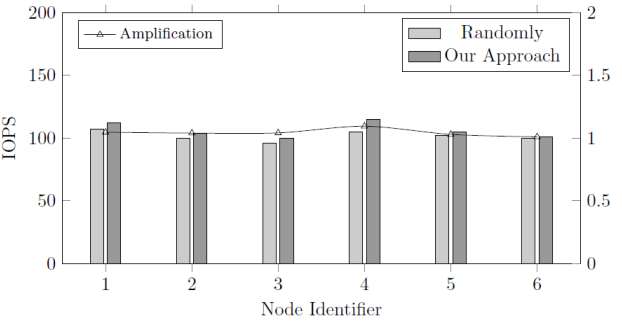
- 实验环境：
  - 软件环境：Xen 4.6
  - 硬件环境：每台Hypervisor配置有Intel Xeon E5-2620 CPU；32GB内存；1块Intel SSD 535 240GB；连接到后端共享存储
  - 虚拟机配置：2个虚拟CPU；2GB内存；基于SSD缓存的磁盘
- 实验场景：IO基准测试、Hadoop和ZooKeeper集群
  - 目标：验证支持虚拟机动态迁移的SSD缓存系统在几类场景下的性能提升和容错能力提升
  - IO基准测试不具有应用内关联和放置倾向性；Hadoop应用更倾向于数据本地性；ZooKeeper应用更倾向于容错
- 选取Hadoop和ZooKeeper的基准测试
  - Hadoop自带的TestDFSIO基准测试
  - ZooKeeper作者开发的SmokeTest异步IO基准测试
- 对比随机迁移方法

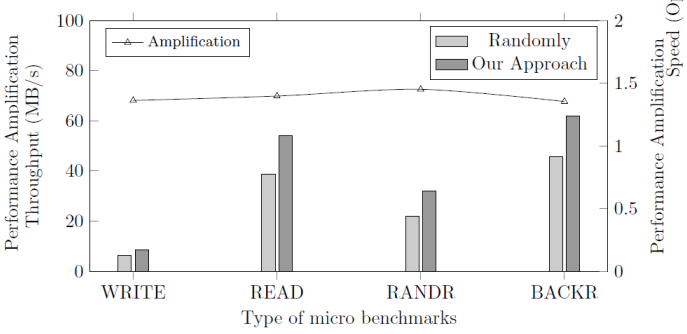
实验结果

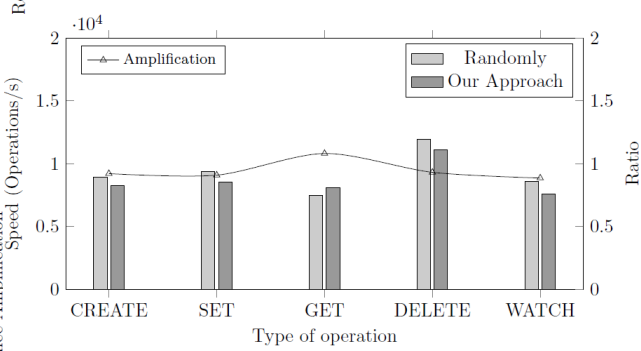
- 与随机迁移方法相比：
  - IO基准测试场景：读密集负载性能平均提升15%；写密集负载性能平均提升6%
  - Hadoop IO基准测试场景：执行时间平均降低25%；吞吐率平均提升39%
  - ZooKeeper 异步IO基准测试场景：保障容错的前提下，平均性能损失小于5%











小结

- 问题：云应用导向的，SSD缓存服务能力驱动的虚拟机在线迁移
- 挑战
  - 感知云应用的特点和对虚拟机放置的需求
  - 满足SSD服务能力约束
- 方法
  - 基于自适应闭环实现对应用特性监测
  - 基于聚类算法计算最优虚拟机放置方案
  - 计算最优虚拟机迁移顺序
- 评价：优化的虚拟机放置方案相比随机迁移方法
  - IO基准测试读密集负载性能平均提升15%；写密集负载性能平均提升6%；Hadoop IO基准测试执行时间平均降低了25%，吞吐率平均提升了39%；ZooKeeper基准测试性能损失在5%之内

谢谢

请各位老师批评指正！

tangzhen12@otcaix.iscas.ac.cn