分　数：	
评卷人：	

華中科技大學

研究生（数据中心技术）课程论文（报告）

题 目：分布式系统动态自动化管理综述

学　　号　M202173861

姓　　名　徐　勖

专　　业　电子信息

课程指导教师　施展　童薇

院（系、所）　计算机科学与技术

2021 年 1 月 6 日

分布式系统动态自动化管理综述

徐勖[1)]

[1)](华中科技大学计算机科学与技术学院，武汉 430074)

摘 要 为了提供高性能、高可用性及容灾备份，现代应用系统一般采取的都是分布式架构。分布式系统需要处理客户机工作负载、资源可用性和部署环境的波动，通常，这个负担由系统管理员承担，他们必须适当地配置系统，监视系统并解决问题。为了减少这种负担，动态自动化管理分布式系统是非常有发展价值的。本文主要从分布式系统的动态自动化管理的角度研究了三篇不同类型分布式系统动态自动化管理的论文。Abebe 等人提出了一种分布式数据库系统 MorphoSys，它可以根据工作负载动态选择和改变其物理设计，Taft 等人提出了一种新的分布式数据库系统 P-Store，它利用预测建模方法，在负载峰值出现之前对分布式数据库进行弹性重新配置，Herodotou 等人描述了一个通用框架，使用一组可插拔策略自动管理分布式文件系统中跨存储层自动管理数据。

关键词 数据中心；分布式系统；自动化管理

A Survey on Automatic Management for Distributed Systems

Xu Xu[1)]

[1)](School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract In order to provide high performance, high availability and disaster recovery backup, modern application systems generally adopt distributed architecture. Distributed systems need to deal with fluctuations in client workload, resource availability and deployment environment. Usually, this burden is borne by system administrators, who must properly configure the system, monitor the system and solve problems. In order to reduce this burden, dynamic automatic management of distributed system is of great development value. This paper mainly studies three papers on dynamic automation management of different types of distributed systems from the perspective of dynamic automation management of distributed systems. Abebe et al. Proposed a distributed database system MorphoSys, which can dynamically select and change its physical design according to the workload. Taft et al. Proposed a new distributed database system p-store, which uses the predictive modeling method to flexibly reconfigure the distributed database before the peak load. Herodotou et al. Described a general framework, Automatically manage data across storage tiers in a distributed file system using a set of pluggable policies.

Key words Data center; Distributed Systems; Automatic Management

1 引言

移动互联网的爆发伴随着分布式系统的突现，这样就对系统提出了各种各样的高标准，响应时间，性能，灾备，吞吐量等等。分布式是相对中心化而来，强调的是任务在多个物理隔离的节点上进行。中心化带来的主要问题是可靠性，若中心节点宕机则整个系统不可用。分布式除了解决部分中心化问题，也倾向于分散负载。

现代应用往往部署在分布式系统上，简单的来说，一个分布式系统是一组计算机系统一起工作，在终端用户看来，就像一台计算机在工作一样。这组一起工作的计算机拥有共享的状态，它们同时运行，独立机器的故障不会影响整个系统的正常运行。

现在有多种分布式架构的应用：分布式文件系统、分布式数据库系统、分布式计算等。

分布式系统最大的好处就是能够让用户横向地扩展系统。横向扩展是指通过增加更多的机器来提升整个系统的性能，而不是靠升级单台计算机的硬件。但是如果单台计算机的性能没有充分利用或者造成浪费是一件很不经济划算的事情。

高吞吐意味着系统可以同时承载大量的用户使用，这个吞吐量肯定是不可能用单台服务器解决的，需要多台服务器协作，才能达到所需要的吞吐量。而在多台服务器的协作中，如何才能有效地利用这些服务器，不使其中某一部分服务器称为瓶颈，从而影响整个系统的处理能力，这就是一个分布式系统在架构上需要仔细权衡的问题。

分布式系统需要处理客户机工作负载、资源可用性和部署环境的波动，通常，这个负担由系统管理员承担，他们必须适当地配置系统，监视系统并解决问题。为了减少这种负担，动态自动化管理分布式系统是非常有发展价值的。系统在部署时提取指标，以确定系统的执行情况，自适应调整系统的架构、处理方式，来优化对当前工作负载的处理。

2 原理和优势

本次主要调研了三篇文献，第一篇文章《MorphoSys: Automatic Physical Design Metamorphosis for Distributed Database Systems》是VLDB'21 的一篇文章。Abebe 等人在这篇文章中提出了一种分布式数据库系统 MorphoSys，它可以根据工作负载动态选择和改变其物理设计。MorphoSys 使用学习到的成本模型对所有的数据分区、复制和放置决策进行集成设计。它通过一种新颖的并发控制和更新传播方案，在面对设计更改时提供了高效的事务执行。实验对比表明 MorphoSys 通过有效和高效的物理设计提供了优秀的系统性能。第二篇文章《P-Store: An Elastic Database System with Predictive Provisioning》来源于 SIGMOD'19，在这篇文章中，Taft 等人提出了一种新的分布式数据库系统 P-Store，它利用预测建模方法，在负载峰值出现之前对分布式数据库进行弹性重新配置。第三篇文章《Automating Distributed Tiered Storage Management in Cluster Computing》发表于 VLDB'20，Herodotou 等人在文章中描述了一个通用框架，使用一组可插拔策略自动管理分布式文件系统（Distributed File Systems，DFSs）中跨存储层自动管理数据。这个策略采用轻量级梯度提升树 XGBoost 来学习工作负载如何访问文件，并使用该信息来决定何时将那些文件向上或向下移动。这个通用模型使用增量学习来动态地用新的文件访问来优化模型，允许分布式文件系统自然地调整和适应随着时间变化的工作负载。

这三篇文章总共介绍了两种不同的分布式系统——分布式数据库系统和分布式文件系统，均将机器学习与分布式系统结合，利用机器学习对工作负载进行和访问方式进行学习建模，实时学习，动态做出管理调度方案，是分布式系统可以更好地应对工作负载的变化，工作效率均优于静态架构或现有动态架构。

分布式系统动态自动化管理的原理为利用机器学习根据一些历史数据或者习惯数据对分布式系统未来访问方式，或访问量，或成本，或收益做出预测，根据这些预测做出综合决策，选择一种能最大化利益的操作对分布式系统架构进行调整改变。

分布式系统动态自动化管理可以能让系统在不了解未来工作负载的情况下实时生成和调整分布式架构，更好地去适应实时工作负载，减轻系统管理员负担，最大化提升系统的工作表现。

接下来，本文将就上述三篇文章进行详尽地分析。

3 MorphoSys: 分布式数据库系统的自动物理设计变形

3.1 概述

3.1.1 研究背景

现代数据库系统通过复制和分区这些数据来存储和管理大量数据，以将它们的事务处理分布在多个站点上。为分布式数据库选择的数据复制和分区方案构成了其物理设计。构建分布式物理设计包括做出以下关键决策(i)数据分区应该是什么，(ii)复制哪些数据分区。以及(iii)在哪个站点放置分区的 master 副本，哪些站点放置分区的 secondary 副本。这些决策会反过来决定事务执行的站点。

作者认为决定一个物理设计是困难的，因为每个选择都会在设计空间中做出权衡，而一个糟糕的选择会显著降低性能。当前大多数的设计决策通常是静态的，不能适应工作负载的变化，或者不能将多个设计选择（如数据复制和数据分区）组合在一起。

3.1.2 现有方法的不足之处

作者指出，现有的分布式数据库系统物理设计方法中常用的有两种，分别为数据分区 Partitioning 和数据复制 Repliocation。Partitioning 分散分区数据，以便事务在不同站点执行，从而将更新和读取负载分散到分布式系统中多个站点。Replication 将数据复制到多个站点，允许读事务在一个或多个数据副本上执行。

然而，这两种实现方式均有各自不同的局限性：数据复制中对于一个数据副本的更新会导致其他副本过期，而且对于分布式系统的并发更新时，副本存在不一致的风险；数据分区中一个事务如果对多个站点访问（尤其是更新）数据时，需要在执行事务的站点之间进行昂贵的同步和协调，以保证事务的原子性和一致性。在同一个站点上防止过多的数据分区（包括 master 副本和 secondary 副本）会在该站点上放置过多的负载，同时消耗更多的存储空间。随着内存数据库在减少延迟方面变得越来越普遍，有效地使用存储空间以降低成本与性能比变得非常重要。因此，动态数据复制可以明智地使用内存存储，有选择地决定复制什么，放置在哪里，而不是复制所有数据。除了在数据复制和分区时出现的事务协调和一致性问题外，还需要避免工作负载不平衡导致的热点。这种不平衡可以有两种类型：(i)当特定数据被频繁访问时导致托管数据的站点负载过大，以及(ii)当工作负载热点转移或出现时。这些类型的负载不平衡源于工作负载争夺物理计算资源（例如 CPU）或数据资源（例如锁）。

作者指出静态分布式物理设计存在热点、工作负载改变或当工作负载信息不可用于通知物理设计决策时，无法提供良好的性能。所以作者提出了 MorphoSys。

3.1.3 作者提出的改进

作者设计并构建了一个分布式数据库系统 MorphoSys，它可以自动决定如何划分数据，复制什么数据，以及在哪里放置这些分区和复制的数据。MorphoSys 从工作负载局部特征中动态学习分区应该死什么，以及在哪里放置 master 副本分区和 secondary 副本分区。这种动态使系统不必先验地了解工作负载访问模式，从而允许 MorphoSys 即时更改或变形分布式物理设计。此外，为了避免昂贵的多站点事务协调，MorphoSys 动态改变其物理设计以定位共同访问的数据，并保证单站点事务执行。

一旦 MorphoSys 开始执行，就不需要数据库管理员干预。MorphoSys 不断迭代地适应其物理设计，调整其复制和分区方案以满足工作负载。MorphoSys 使用基于工作负载观察的学习成本模型来决定如何以及何时更改其物理设计，并使用基于分区的并发控制和更新传播方案高效地动态执行设计更改。

3.2 设计

作者受原先物理设计方法动态复制、动态分区和动态调整主副本位置的启发，提出了三种动态调整方案相融合的方法。

与原动态复制相比，MorphoSys 在为每个数据分区做出复制决策时考虑了整个工作负载，并保证了单站点执行；与原动态分区相比，MorphoSys 通过使用动态物理设计操作符（改变主副本及其次要副本的位置）来保证单站点事务，在事务级别的粒度上迭代地更改分区，将每个事务都看作是调整其物理设计以适应工作负载的机会；MorphoSys 为了保证单站点执行采用了动态调整主副本位置，利用次要副本有效地更改主控，不需要完整的数据复制，动态地对数据分区以减少竞争的影响，MorphoSys 在事务到达时执行事务，以减少延迟。

MorphoSys 将所有这些物理设计决策以一种在线的方式整合在系统内部，使其能够实时根据工作

负载的变化做出调整。

MorphoSys 的分布式架构由存储数据和执行数据事务的数据站点组成。MorphoSys 以 OLTP 优化的面向行格式存储数据。每一行都表示一个由行 id 标识的数据项。MorphoSys 将数据项组合到分区中，每个数据项一次属于一个分区。对于每个分区，MorphoSys 决定存储分区的主副本的站点并懒惰复制这个分区（如果有次要副本的话）。这些决策中任何一个更改都会导致物理设计更改，MorphoSys 对这些决策进行操作，使用物理设计更改操作来动态更改系统的数据复制、分区和主副本位置方案。事务路由器将客户端向数据库系统提交的事务路由到数据站点时考虑当前系统的物理设计，MorphoSys 在客户端提交的事务中使用读和写集信息来计划和执行单站点执行所需的设计更改，并提高系统性能。

下图 1 为 MorphoSys 的架构，分区被标识为连续的行 id 范围，主副本用粗体表示，次要副本不是用粗体表示。图也说明了 MorphoSys 事务执行的概述。

事务路由器通过将事务路由到一个站点来确保单站点事务执行，该站点存储写集中包含数据项的分区的主副本，以及读集中包含数据项的分区的主副本或次要副本。如果没有站点满足这一需求，那么事务路由器通过动态地添加或删除分区的副本、更改分区的粒度或更改分区的主副本位置来动态地更改系统的物理设计。

数据站点必须有效地维护副本，支持更改数据项所属的分区，以及分区的主副本。数据站点还必须确保事务在出现物理设计更改时观察数据库的一致状态。

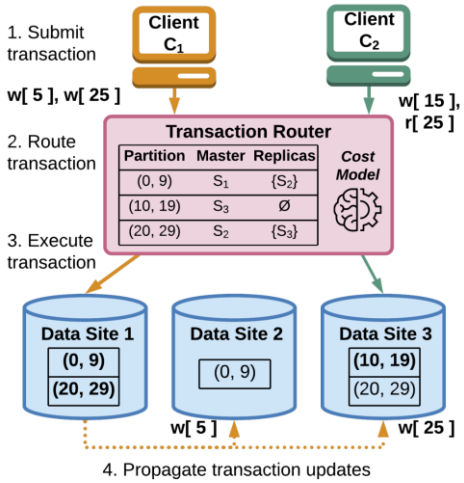
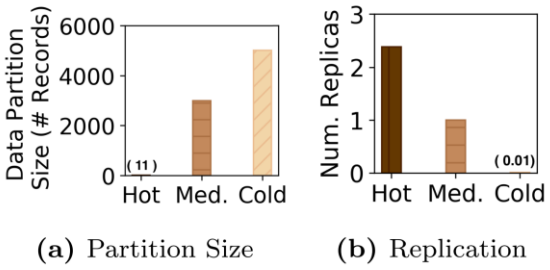


图 1 MorphoSys 系统架构

3.3 实验

为了评估效果，作者将 MorphoSys 与 DynaMast、ADR、Clay 三种种采用最先进的动态物理设计的分布式数据库系统和 Single-Master、Multi-Master、VoltDB 三种流行的静态物理设计的分布式数据库系统进行比较。为简洁起见，本文只选择部分实验结果进行介绍。在每个实验中，MorphoSys 从一个初始物理设计开始，该设计不包含任何次要副本，包含一个随机分布的主副本分区、一个未知的工作负载模型，并且必须从头学习它的成本模型。

首先，作者使用倾斜的 YCSB 工作负载测试系统的吞吐量。倾斜的工作负载会产生争用和负载不平衡，MorphoSys 可以使用动态物理设计更改来环节这两种情况。实验结果如图 2 所示。



(a) Partition Size (b) Replication

图 2 MorphoSys 的设计决策

通过实验结果可以看出，在图 2a 中，如果数据项位于访问频率最高的数据的前 10%，作者将其分类为 Hot；如果位于前 30%，则将其分类为 Medium；其余的数据为 Cold。如图 2a 所示，平均而言，MorphoSys 将 Hot 数据项分组到小分区中；相比之下，MorphoSys 将不经常访问即 Cold 数据项分组到大分区中。在图 2b 中显示了具有不同访问频率的数据项的复制因子。MorphoSys 复制频繁访问的数据项，避免复制不经常访问的数据项。

图 2 可说明 MorphoSys 将 Hot 分区也就是访问频繁和竞争激烈的数据分区重新动态拆分到更小的分区减少争用，将 Cold 分区合并为大分区；多次复制 Hot 分区去分散读请求，但是不复制 Cold 分区去节省存储空间。MorphoSys 可以根据实际的工作负载有效地选择物理设计。

从图 3 图 4 可看出 MorphoSys 提供了更好的工作性能。

作者还使用倾斜的 YCSB 工作负载诱导热点，并每 60 秒将倾斜的中心移到数据库的不同部分来测试了 MorphoSys 对工作负载变化的反应能力。实验结果如图 5 所示。

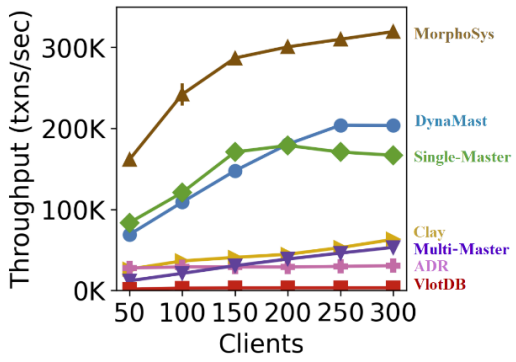


图 3 以读为主的倾斜 YCSB 实验结果

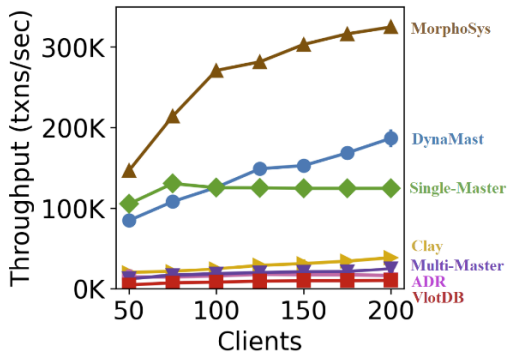


图 4 以写为主的倾斜 YCSB 实验结果

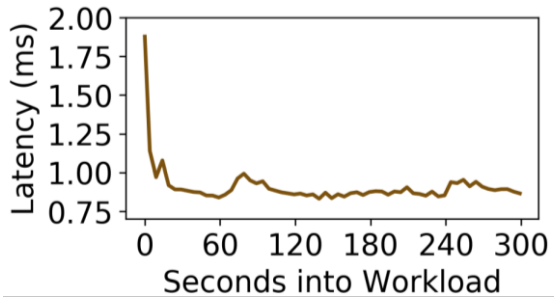


图 5 MorphoSys 对工作负载变化的自适应能力

这种具有挑战性的工作负载导致 MorphoSys 改变其物理设计，以减轻倾斜和负载不平衡的影响。图 5 显示了超过 5 分钟和 4 个工作负载转移的平均延迟。MorphoSys 在最初的 30 秒内学会了它的初始设计，此时 MorphoSys 达到了它的初始设计，即初始延迟减少了近 60%。当热点转移时，MorphoSys 的延迟从最小值增加最多 20%，平均在 20 秒内返回到最小延迟，因为 MorphoSys 快速分割分区，添加 Hot 分区的副本和更改主副本位置，以平衡负载和缓解热点问题。这些结果显示了 MorphoSys 快速适应工作负载变化的有效性，这得益于其学习过的工作负载和成本模型，以及低开销的设计更改。

4 P-Store: 具有预测性功能的弹性数据库系统

4.1 概述

4.1.1 研究背景

作者指出，当前 OLTP 数据库系统通常是静态配置的，有足够的能力应付高峰负荷。然而，对于许多 OLTP 应用程序来说，最大负载比最小负载大一个数量级，并且负载以每天重复的模式变化。因此，明智的做法是动态分配计算资源以满足需求。可以在检测到负载增加后主动分配资源。但这会给已经过载的系统增加额外的重新配置负担。在负载增加之前进行重新分配显然更可取。

4.1.2 现有方法存在的问题

作者指出现有的弹性数据库虽然已经显示了 DBMS 如何自动适应不可预知的工作负载变化，以满足其客户端的吞吐量和延迟需求。但是这些系统在重新配置过程中性能很差，因为只有在系统已经处于重负载的情况下才会触发重新配置。这样的性能问题使得弹性数据库系统对于需要高可用性和快速响应时间的公司来说是不可用的。

作者指出，如果更早地开始重新配置，就可以避免上述当前弹性数据库存在的问题，但这需要了解未来的工作负载。幸运的是，OLTP 工作负载通常遵循循环的、可预测的模式。

4.1.3 作者提出的改进

在这篇论文中，作者研究了 B2W Digital 的工作量，图 6 显示了三天内 B2W 的工作负载。总负载基本在一天的过程中遵循一个正弦波，此外，波峰和波谷之间的差别很大，可以看出峰值荷载约为低谷负载的 10 倍。像 B2W 这样的公司可以利用预测建模尽可能地准确使用需要的计算资源去管理工

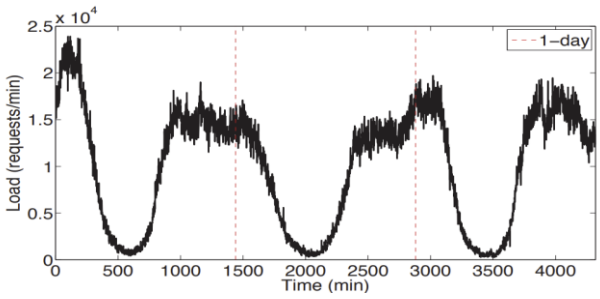


图 6 三天内 B2W 一个数据库上的负载

作负载，他们可以将数据库所需的平均服务器数量减少一半左右。

作者重点关注分布式、无共享 OLTP 数据库系统的工作负载预测和动态供应，这是以前的工作没有探索过的领域。研究了如何在系统负载超过系统容量前，通过主动重新配置数据库来实现节约成本。

作者提出了 P-Store——第一个使用最先进的时间序列预测技术来预测数据库未来负载的弹性 OLTP DBMS。它不会等待数据库超载，而是在系统仍然有足够的资源来迁移数据，并且并发地为客户机事务提供服务时主动重新配置数据库，而不会影响性能。

4.2 设计

4.2.1 动态规划算法

作者提出一种动态规划算法用于根据未来负载的预测确定何时以及如何重新配置数据库。

根据经验，预测模型需要三个参数：

(1) Q：每个服务器的目标吞吐量。用于确定为预计负载提供服务所需的服务器数量。

(2) $\hat{Q}$：每个服务器的最大吞吐量。如果负载唱过此阈值，则可能违反延迟约束。

(3) D：使用单个发送方-接收方线程对移动数据库中所有数据的最短时间，这样，重新配置对查询延迟没有明显影响。P-Store 调度的重新配置实际上会移动数据库的一个子集，其中包含并行线程，但是D被用作一个参数来计算重新配置需要多长时间，以便在预计的负载增加之前及时调度完成重新配置。我们假设D随数据库大小线性增加

作者给出假设：(1)负载预测是准确的，误差很小；(2)工作负载不会很快变化；(3)数据库大小没有快速变化；(4)工作负载分布在数据分区之间是大致一致的；(5)数据均匀地分布在数据分区上；以及(6)工作负载很少有分布式事务。

作者提出的预测算法通过规划从当前时刻到未来指定时间的一系列移动来适应不断变化的负载。

4.2.2 调度算法和重新配置模型

作者提出了一个执行重新配置的调度算法，以及一个在重新配置期间描述运行时间、成本和有效系统容量的模型。

在执行重新配置过程中，为了最小化成本，配置会尽可能晚地添加新服务器。在执行移动时，P-Store 使用三种可能的策略来最大化并行性，如图 7 所示。假设每个服务器有一个分区，在迁移过程中分配的服务器和有效容量。以D为单位的时间，使用单个线程迁移所有数据的时间。

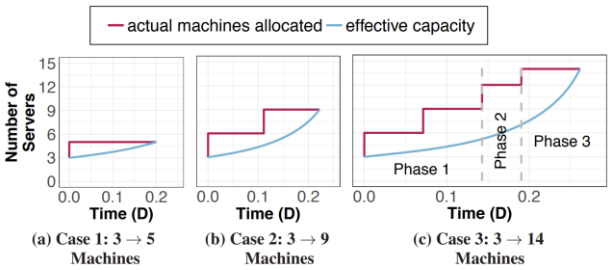


图 7 迁移过程中分配的服务器和有效容量

如果考虑数据库必须从B台机器改为A台机器的实际比例，重构时间为：

$$T(B,A)=\begin{cases}0, & \mathrm{B}=\mathrm{A}\\ \dfrac{D}{max_{||}}*\left(1-\dfrac{B}{A}\right), & \mathrm{B}<\mathrm{A}\\ \dfrac{D}{max_{||}}*\left(1-\dfrac{A}{B}\right), & \mathrm{B}>\mathrm{A}\end{cases}$$

成本取决于随时间分配的机器数量，因此作者将重新配置的成本定义为：

$$C(B,A)=T(B,A)*avg-mach-alloc(B,A)$$

作者定义均匀分布的N台机器的总容量为：

$$cap(N)=Q*N$$

4.2.3 负载时间序列预测

作者使用稀疏周期自回归预测模型 SPAR，捕获数据中的时间相关。SPAR 努力推断负载对长期周期模式和短期瞬态效应的依赖。作者在 SPAR 中基于$\mathrm{t}+\tau$时刻的周期信号，以及最近一段时间的负载与预期负载之间的偏移来建模$\mathrm{t}+\tau$时刻的负载。

经过实验评估，在不同的工作负载下，SPAR 通常产生最准确的预测（因为它捕获了数据中心的不同趋势）。

4.2.4 算法开源

P-Store 技术是通用的，可以应用于任何分区 DBMS，但是 P-Store 实现是基于开源的 H-Store 系统的。它使用 H-Store 的系统调用来获取系统聚合负载的测量值。数据迁移由 H-Store 的弹性子系统 Squall 管理。P-Store 系统结合了所有以前的时间序列预测技术，确定何时扩展，以及如何调度迁移。作者创建了一个“预测控制器”，它处理系统的在线监控和对实现这些技术的组件的调用。

P-Store 有一个活跃的学习系统，可以离线学

习，也可以随着时间的推移不断检测系统，并主动学习参数值。P-Store 的预测组件根据 H-Store 的聚合负载的当前测量值进行在线预测。

一旦 P-Store 开始运行，“预测控制器”就开始监视系统并测量负载。当有足够的负载可用时，它调用预测组件，返回未来负载预测的时间序列，控制器将数据传递给调度器，调度算法计算最佳的移动策略。

4.3 实验

作者为了评估 P-Store 性能，运行 B2W 基准测试，所有实验都是在一个 H-Store 数据库上进行的。

作者比较了几种不同的弹性方法的性能和资源利用率。作为比较的基准，作者在没有弹性的 H-Store 上运行基准程序。测试结果如图 8 所示。

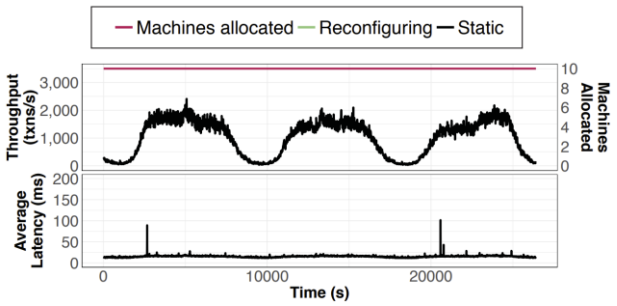


图 8 拥有 10 台机器的静态供应集群性能

图 8 表明平均延迟很低，在第一天和第三天只有两个小的峰值，这些峰值大概是由瞬态工作负载倾斜引起的。当服务器数量从 10 减少到 4 测试结果如图 9。

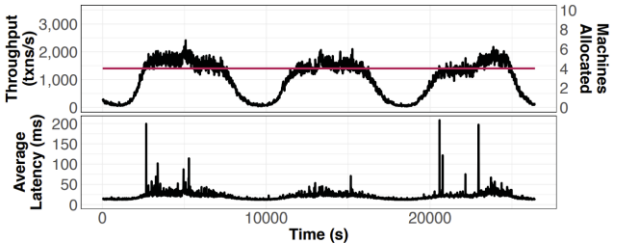


图 9 拥有 4 台机器的静态供应集群性能

高延迟事务的数量明显增加，像 B2W 这样的公司是不能容忍这些延迟峰值，因此要为峰值负载做好准备。作者还使用 E-Store 使用的弹性技术运行基准测试，测试结果如图 10 所示。

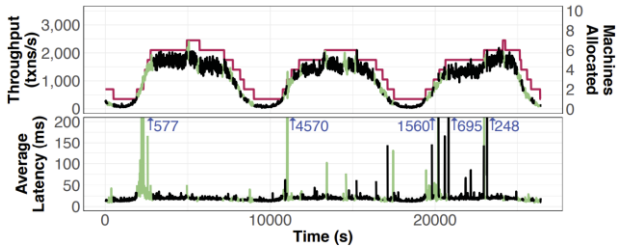


图 10 E-Store 性能

图 10 表明这种技术可以正确地对每天的负载变化做出反应，并根据需要重新配置系统，以满足需求，但是由于在峰值容量时进行重新配置，在每次负载增加时都会导致更高的延迟。图 11 显示了 P-Store 在 B2W 基准测试上的运行情况。

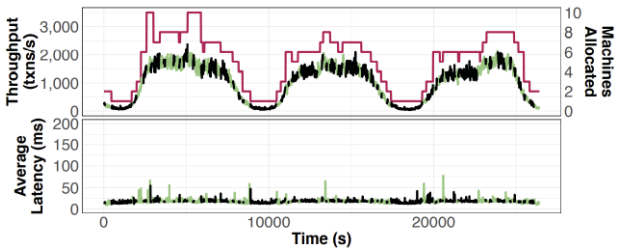


图 11 基于 SPAR 预测模型的 P-Store 的性能

与 E-Store 相比，延迟峰值要少要小得多，因为 P-Store 在负载增加之前重新配置了系统，并为瞬态负载变化和倾斜提供了更多的空间。

为简洁起见，作者进行的其他测试本文不再一一介绍。总的来说，在本文中，Taft 等人提出的 P-Store 通过主动伸缩而不是被动伸缩改进了现有的弹性数据库。

5 集群计算中的自动化分布式分层存储管理

5.1 概述

5.1.1 研究背景

数据密集型分析应用程序往往会在执行 I/O 时花费相当大的一部分执行时间。随着内存大小的增加和新的硬件技术（例如 NVRAM、SSD）导致了在分布式文件系统（以及一般的存储系统）中增加了多个存储层，数据在不同存储层之间移动的复杂性也会显著增加，使得系统提供的更高的 I/O 性能难以得到充分利用。

然而，用户现在在尝试优化其工作负载的同时，还要管理多个存储层及其上的数据，这增加了

复杂性。

三种常见分层存储系统的数据放置如图 12 所示。

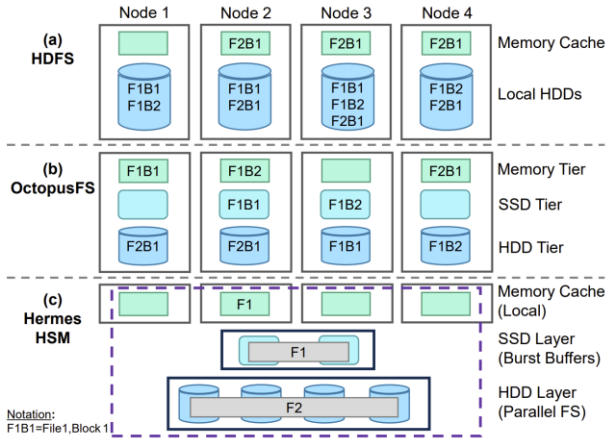


图 12 三种分层存储系统的数据放置

5.1.2 现有方法存在的问题

作者认为，现有的大多数系统如 Hadoop YARN、Spark 都向应用程序开发人员和数据分析人员公开用于数据缓存或移动的 API。然而，当缓存被填满时，在应用程序手动取消缓存某些文件之前，不会提供额外的缓存请求。OctopusFS 提供了一个放置策略，用于确定最初如何跨存储层存储数据，但缺少任何用于以后自动移动数据的功能其他系统如 Alluxio 何 GridGain 实现了在内存满时从内存中删除数据的基本策略，如 LRU。然而，众所周知，这些策略在大数据环境中性能不佳，因为它们最初是为从缓冲区缓存中移出固定大小的页面而设计的。此外，这些系统不提供任何缓存许可策略；也就是说，他们将在访问时将所有数据放在缓存中，而不考虑系统的当前状态、数据大小或任何工作负载模式。

总的来说，缺乏跨存储层的自动数据移动给用户或负责优化各种类型数据分析工作负载的系统管理员带来了巨大的负担。因此，识别重用的数据并将其保存在更高的层中可以带来显著的性能好处。此外，随着用户和工作的添加和删除，数据访问模式可能会随着时间的推移而变化，从而增加了准确和有效地适应这些变化的需求。因此分层存储系统必须包含自动化数据管理功能，以提高集群效率和应用程序性能。

5.1.3 作者提出的改进

作者认为分层存储系统必须包含自动化数据管理功能，以提高集群效率和应用程序性能，因此提出了一个分布式文件系统中自动化分层存储管理的通用框架。

具体来说，这个框架可以用于编排数据管理策略，以自适应地决定(i)何时以及哪些数据应该保留或移动到更高的存储层，以提高读性能，以及(ii)何时以及哪些数据应该移动到较低的存储层，以释放稀缺的资源。

此外，作者提出使用机器学习来跟踪和预测系统中的文件访问模式。特别是，作者使用了轻量级的梯度提升树来了解当前工作负载如何访问文件，并使用生成的模型来驱动自动文件系统策略。该方法使用增量学习，在模型可用时使用新文件访问动态优化模型，使模型能够自然地调整和适应随时间变化的工作负载。

作者的工作涉及分布式存储系统、分层存储管理解决方案和缓存，以及大量相关工作。跨层存储带来两个额外的好处：(1)它提高了总体 I/O 性能和读写操作的集群资源利用率；(2)它使更高层次的系统能够同时做出位置感知和层次感知的调度决策。

作者提出了一种全新的、完全自动化的、自适应的方法，在分布式文件和存储系统中进行分层存储管理。该框架可以在现有的分布式文件系统 OctopusFS（一种 HDFS 的向后兼容扩展的分布式数据库）中设置。

5.2 设计

作者为了研究在多个存储层之间存储和检索块副本的效果，使用 DFSIO 基准测试在四种分布式系统——Original HDFS、HDFS with Cache、OctopusFS 和 OctopusFS++这四个拥有 3 个存储层（内存、SSD 和 HDD）的 12 节点集群中读写 84GB 的数据。测试结果如图 13 图 14 所示。

测试结果展示了分布式文件系统中分层存储在提高 I/O 性能和更好的资源利用率方面的好处。同时也能看出一些关于静态数据放置决策的重要问题。首先，随着时间的推移，内存中充满了可能不再需要的文件，这阻止了更新或更重要的文件被存储在内存。另一方面，有些文件可能比其他文件被访问得更加频繁，因此移动或复制它们到内存中会更有益。文件访问模式会随着时间的推移而变化，因此文件系统必须能够调整其行为，以避免性能上的巨大变化。

作者根据数据在层之间移动的方式将数据移动分为两类，并制定了两个定义：副本降级 Replication downgrade 和副本升级 Replication

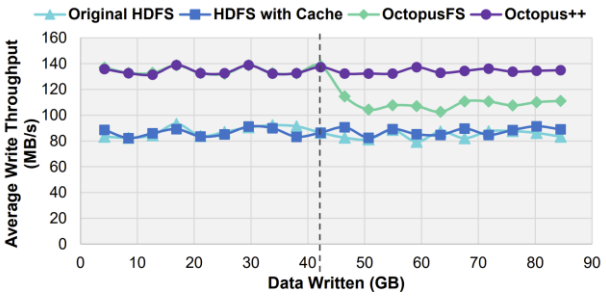


图 13 四种分布式系统每个节点平均写吞吐量

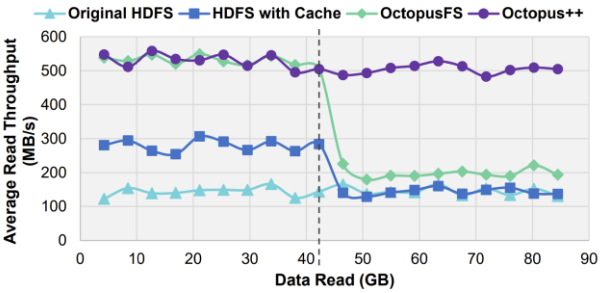


图 14 四种分布式系统每个节点平均读吞吐量

upgrade。副本降级是指(i)将文件副本从高存储层移动到低存储层，或(ii)删除文件副本的过程。副本升级是指(i)将文件副本从较低的存储层移动到较高的存储层，或(ii)创建新的文件副本。知道副本降级或升级所必须的四个重要决策点：

1. 什么时候开始降级（升级）进程
2. 降级（升级）哪个文件
3. 如何降级（升级）选定的文件
4. 何时停止降级（升级）进程

这 4 个决策点是对传统缓存管理以及分层存储管理 HSM 中分层和缓存的概括。

作者对访问模式建模时维护每个文件的最后 k 此访问时间，它与文件大小和创建时间一起构成了模型的训练数据。作者使用时间戳来生成时间增量，即(i)两个连续时间访问之间的时间差，(ii)最早的时间访问和创建时间，(iii)一个参考时间和最近的访问，(iv)一个参考时间和创建时间之间的时间差。参考时间是一个特定的时间点，用来区分被感知的“过去”和“未来”。“过去”表示文件被访问的时间和频率它被用来生成特征向量$\vec{x_i}$；“未来”显示文件是否会在给定的前瞻性类窗口中被重新访问，该窗口用于生成类标签y；如果文件在该窗口被访问，则$y=1$，否则$y=0$。最后将所有值缩放到 0 到 1 之间来规范化特征。在监督学习中，数据$D=\left((\vec{x_1}, y_1),(\vec{x_2}, y_2),\ldots,(\vec{x_n}, y_n)\right)$由特征向量$\vec{x_i}$和类值$y_i$，推断出模型$M \approx p(y|\vec{x})$。

作者选择了 XGBoost 作为学习模型，这是目前最先进的梯度提升树算法。在系统开始用模型 M 进行预测前需要确保 M 已经接受了足够的训练。XGBoost 模型将返回一个概率分数，表明文件 f 在接下来 w 分钟内被访问的可能性，使用这个概率分数来决定升级或降级哪个文件。作者为此生成两个不同的模型——一个用于升级策略一个用于降级策略，唯一区别在于窗口大小 w，升级策略是一个小窗口（例如 30 分钟），降级策略是一个较大的窗口（例如 6 小时）。

降级策略中，系统会在 T 层使用容量大于阈值（例如 90%）时开始主动对 T 层中文件进行降级，允许文件写入和文件降级之间有效的重叠。选择降级文件有多种策略（见表 1），将在实验中进行比较，此处不赘述。降级时需要选择一个较低的存储层来移动文件副本，作者采用了 OctopusFS 使用的数据放置策略，该策略通过解决多目标优化问题来做出决策。当存储层 T 的使用容量低于阈值（例如 85%），所有策略将停止降级过程，从而允许存储层 T 的一小部分容量被同时释放。降级算法如 Algorithm 1 所示。

表 1 降级策略

Acronym	Policy Name
LRU	Least Recently Used
LFU	Least Frequently Used
LRFU	Least Recently & Frequently Used
LIFE	LIFE (PACMan [5])
LFU-F	LFU-F (PACMan [5])
EXD	Exponential Decay (Big SQL [16])
XGB	XGBoost-based Modeling

Algorithm 1 Downgrade process outline

```
1: procedure DOWNGRADE(StorageTier fromTier)
2:   if policy.startDowngrade(fromTier) then
3:     repeat
4:       file = policy.selectFileToDowngrade(fromTier)
5:       toTier = policy.selectDowngradeTier(file, fromTier)
6:       downgradeFile(file, fromTier, toTier)
7:     until policy.stopDowngrade(fromTier)
```

与降级策略类似，在升级策略中，可用多种方法（即升级策略，见表 2）预测哪个文件即将被访问，即将被访问就要及时将文件升级，同样采用 OctopusFS 使用的数据方式策略选择一个更高的存储层，同时考虑容错、数据和负载均衡以及吞吐量最大化之间的权衡，知道不久的将来没有更多可能被访问的文件，或者知道计划升级的总大小超过一个阈值（例如 1GB）停止升级。升级算法如 Algorithm

2 所示。

表 2 升级策略

Acronym	Policy Name
OSA	On Single Access
LRFU	Least Recently & Frequently Used
EXD	Exponential Decay (Big SQL [16])
XGB	XGBoost-based Modeling

Algorithm 2 Upgrade process outline

```
1: procedure UPGRADE(StorageTier fromTier, File accessedFile)
2:   if policy.startUpgrade(fromTier, accessedFile) then
3:     repeat
4:       file = policy.selectFileToUpgrade(fromTier)
5:       toTier = policy.selectUpgradeTier(file, fromTier)
6:       upgradeFile(file, fromTier, toTier)
7:     until policy.stopUpgrade(fromTier)
```

5.3 实验

作者进行的评估时基于两个来自 Facebook 和 CMU 集群的真实世界生产跟踪的工作负载。在端到端评估中，作者在默认没有任何降级或升级策略 OctopusFS、使用 LRU 降级 OSA 升级 Octopus++和使用所有常见降级和升级策略（见表 1 和表 2）的 Octopus++。评估结果如图 15 所示。

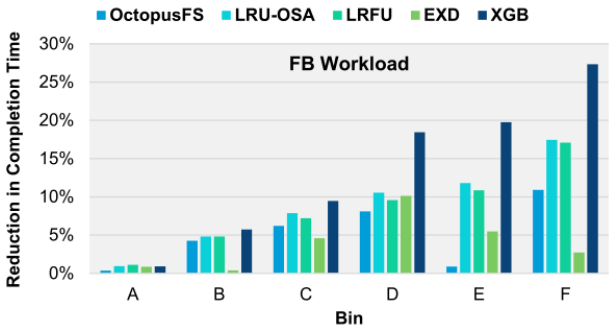


图 15 Facebook 工作负载在 HDFS 上完成时间减少的百分比

图 15 显示了与每个 bin 的两个工作负载的 HDFS 设置相比，任务完成时间减少的百分比。在数据比较小的任务（Bin A、Bin B）中只有很小的改善没因为与 CPU 处理和调度开销相比，花在 I/O 上的时间只是很小的一部分。EXD 在较大任务中表现较差，因为 EXD 在没有考虑文件大小的情况下，EXD 会在大文件被访问之前将其降级，从而导致从 SSD 或 HDD 中重新读取它们的数据。作者提出的 XGB 策略能够在所有作业提供最大的平均完成时间减少。在 CMU 工作负载上表现相似（只有少数因为 CMU 工作负载独有的特点导致在某种升降级策略下效率提升效果不佳），不再赘述。

总的来说，XGB 能够有效地学习不同的访问模式，并检测不同工作负载下作业之间的数据重用。

接下来，作者评估了随着时间的推移对模型进行累积训练的增量学习方法，与(i)每一小时对模型进行一次再训练，以及(ii)从第一个小时开始对数据进行一次一次性训练三种预测精度的表现情况。如图 16 所示。

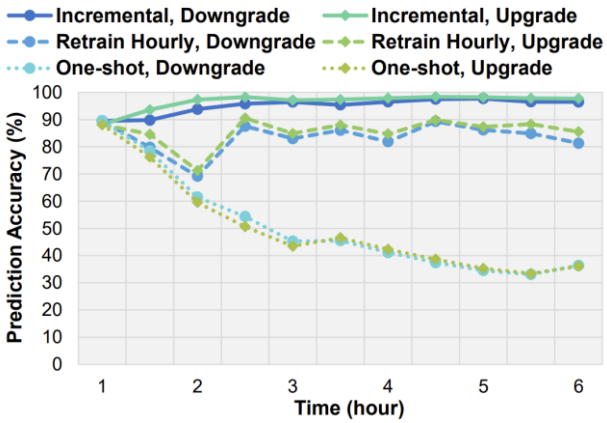


图 16 对 Facebook 工作负载采用增量学习、每小时再培训和一次性方法的预测精度

图 16 中显示了在 Facebook 工作负载上降级和升级策略的三种方法的预测精度如何随时间变化。尽管一次性学习可能在开始时准确率接近 90%，但随着时间的推移，随着工作量的变化，导致准确率显著下降到 40%一下。再训练方法显示出了一种震荡模式，每次训练后，准确率都在增加，但保持在 80%和 90%以内。相反，增量学习方法随着时间的推移会变得更好，有效地适应新的工作负载，并在整个实验过程中持续产生准确的预测（约 98%）

综上所述，Herodotou 等人提出的可用于分布式文件系统可用存储层之间自动移动数据的通用框架在工作负载性能和集群效率方面有明显的优势。

6 总结

分布式系统的出现正是为了应对大工作负载，但是工作负载的组成不是一成不变的，分布式系统根据工作负载及时调整自己的分布式架构和数据读取方式是非常重要的。系统自动化管理更能进一步减轻系统管理员的工作负担，而且系统通过机器学习获得的调整方案可能是管理员也无法得到的。

本次调研的三篇论文包含了两种分布式系统的动态自动化管理框架，都是开源可插拔的，具有普适性，所用的机器学习算法都能很好的为系统提供预测和帮助。

第一篇论文介绍了一个分布式数据库系统MorphoSys，这个系统可以自动修改其物理设计以提供卓越的性能。MorphoSys 使用学习到的成本模型做出全面的设计决策，并使用基于分区的并发控制和更新传播方案高效地动态执行设计更改。MorphoSys 能够在不了解工作负载的情况下实时生成和调整分布式物理设计。

第二篇论文提出了一种新的数据库系统 P-Store，它利用预测建模方法，在负载峰值出现之前对数据库进行弹性重新配置；描述了一种新的动态规划算法调度重新配置；提出了一个时间序列模型，可以准确地预测不同应用的负载。P-Store 通过主动伸缩而不是被动伸缩，改进了现有的数据库弹性工作。

第三篇论文描述了一个使用一组可插拔策略自动管理分布式文件系统中跨存储层数据的框架。作者提出的策略使用了轻量级的梯度提升树 XGBoost 来学习工作负载如何访问文件，并使用这些信息来决定哪些文件应该向上或向下移动。这些模型会根据文件访问模式的变化情况进行增量更新，因此能够随着时间的推移保持较高的预测精度。

各种分布式系统都有相通的地方，所以能够设想机器学习也运用到其他分布式系统如分布式管理系统中，或者本调研探讨的分布式分层文件系统自动化管理方案能否应用到分布式数据库中，这都是日后可以去研究的方向。

7 参考文献

[1] Abebe Michael, Brad Glasbergen, and Khuzaima Daudjee. MorphoSys: Automatic physical design metamorphosis for distributed database systems. Proceedings of the VLDB Endowment, 13-13, 2020: 3573-3587.

[2] Taft, Rebecca, et al. P-store: An elastic database system with predictive provisioning. Proceedings of the 2018 International Conference on Management of Data. 2018: 205-219.

[3] Herodotos Herodotou and Elena Kakoulli. Automating Distributed Tiered Storage Management in Cluster Computing. Proceedings of the VLDB Endowment, 13-1, 2019: 43-56.