科研信息化技术与应用
2013, 4(1): 41–48

· 技术 / TECHNOLOGY ·

海量结构化数据存储管理系统 OceanBase

阳振坤，杨传辉，李震

阿里巴巴，北京 100022

摘　要：数据库是现代社会十分关键的基础设施，一直以来，数据库的扩展以纵向模式 (Scale-up) 为主，数据库系统的容量和性能更多依赖于服务器单机硬件 (CPU/内存/磁盘/网络等) 的提升，难以满足当今信息化社会对海量结构化数据的高性能、低成本的存储和处理能力的需求。本文设计和实现了一个基于横向扩展 (Scale-out) 模式的数据库系统 OceanBase，该系统支持事务 (ACID) 和范围查询等关系数据库和 SQL 语言的主要功能分库分表不再需要，服务器可以在线地添加和移除。OceanBase 已用于阿里巴巴的多个线上系统，每天提供数十亿次的实时读写访问服务。

关键词：数据库；事务；分布式系统；扩展性

OceanBase——A Massive Structured Data Storage Management System

Yang Zhenkun, Yang Chuanhui, Li Zhen

Alibaba, Beijing 100022

Abstract: DBMS is a critical infrastructure for modern society. Performance improvement of DBMS has relied on CPU/memory/storage/network updates in a dedicated site (scale-up) and thus makes DBMS feeble to handle today's massive structured data with high performance and low cost. In this paper, we have built OceanBase, a semi-distributed shared-nothing system which supports key features of RDBMS and SQL, e.g., general purpose transaction (ACID), range query, etc. Sharding is obsolete and servers can be added/removed to/from OceanBase on-the-fly. OceanBase has been used by dozens of projects in the product system of Alibaba and offers billions of real-time read and write queries every day.

Keywords: Database; Transaction; Distributed System; Scalability

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

1 背景

毫无疑问，数据库是二十世纪人类科学技术最伟大的成就之一。数据库事务的原子性 (Atomicity)、一致性 (Consistency)、隔离性 (Isolation) 和持久性 (Durability) 对许多应用提供了关键的支撑。今天，如果数据库停止工作，那么人们不仅不能使用信用卡、不能订购飞机票和火车票、不能买卖股票证券，也无法使用网络、无法在商店购物，甚至无法打电话。经历半个世纪左右的发展，数据库系统日臻成熟，已成为当今信息社会最重要的基础设施之一。

可扩展性是几乎所有应用系统都面临的挑战，由于事务 (Transaction) 以及应用系统对响应时间的严格要求等因素，数据库系统 (指联机事务处理 OLTP 系统，下同) 基本都是单机系统，迄今为止数据库系统的扩展主要依赖于服务器单机硬件系统的提升 (小型机、中型机、大型机等)，即纵向扩展 (Scale-up) 模式。

然而，开放的互联网及其应用的发展速度远远超出了人们的想象，例如：

• 2008 年 7 月，Google 宣布索引了一万亿网页，假设平均每个网页 20KB，这些网页的总量达到 20PB；

• 2012 年 11 月 11 日，支付宝笔数达到 1.0580 亿，金额 191 亿，是 2011 年峰值 (11 月 11 日) 笔数 3 369 万、金额 52 亿的 3 倍多；

• 2012 年 11 月 9 日，淘宝收藏夹 (淘宝用户收藏的宝贝) 单表超过 100 亿条记录；

总体拥有成本 (TCO) 是纵向扩展 (Scale-up) 模式的另一个挑战。高端服务器以及对应的高端存储设备不仅价格昂贵，维护及服务成本也非常高，除了大银行、证券交易所、民航售票系统等少数机构外，很多机构如互联网公司等，无法承受这种高昂的成本。

与纵向扩展 (Scale-up) 模式相对应，横向扩展 (Scale-out) 模式通过增加服务器数量来满足业务对存储和计算能力的扩展需求。由于可以采用性价比好得多的主流服务器 (Commodity server)，横向扩展 (Scale-out) 模式不仅扩展能力强，性价比也非常高，这使得它在互联网以及云计算等领域获得了极大的成功。

本文研制完成了一个 share-nothing 的结构化数据存储管理系统 OceanBase。该系统具有关系数据库和 SQL 语言的主要功能，例如事务 (ACID)，范围查询等，并且不需要分库分表；基于横向扩展 (Scale-out) 模式，可以在线增加/减少机器，以扩充/缩减系统的容量和性能；消除了随机磁盘写，采用当前普及的固态盘 (SSD) 可获得很高的性能。

2 系统设计

2.1 系统概述

许多应用表格非常大，例如全国的身份证信息超过 14 亿条记录，银行一年的交易可能有几十亿条、几百亿条记录，等等。然而，这些大表每天修改 (包括增加/删除/更新等，下同) 的记录数一般并不多，例如平均每天出生和死亡的人口可能只有几万人，因为各种原因修改身份证信息的人毕竟是少数；银行每天的交易可能只有几千万条；这些修改操作，当前主流的 PC 服务器一台机器可能就可以存储和处理。OceanBase 系统正是基于这样的思路：

• 数据 = 基线数据 (Baseline data) + 修改增量 (Mutation incremental)；

• 基线数据 (Baseline data)：整个数据库数据在某个时刻 (例如凌晨 1:00) 的快照，内部按主键排序并水平切分 (称为 Tablet，对使用者透明) 后存储在多台“基线服务器”上，每个 tablet 有多个副本 (缺省 3 个) 并分布在不同的基线服务器上以防止机器故障导致数据不可用；

• 修改增量 (Mutation incremental)：整个系统从基线数据的快照时刻 (例如每天凌晨 1:00) 之后的修改操作 (包括增删改等，下同)，并作为一张或几张表 (“修改增量表”) 集中保存在一台“增量服务器”上。为了保证可靠性并进一步提高性能，增量服务器常常配置实时热备 (备机可提供读服务)；

• 通常每天在指定时刻 (例如凌晨 1:00) OceanBase 冻结当前的修改增量表，并同时开启新的修改增量表 (初始内容为空)，此后的修改写入新的修改增量表，

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

冻结的修改增量表则与当前的基线数据融合以产生新的基线数据，此过程即为“每日合并”。在此期间系统的读写事务照常进行；

• 读请求需要融合基线数据 (Baseline data) 与对应的实时修改增量 (Mutation incremental)；

如图 1 所示。该方案有如下的特点：

(1) 分布式的基线数据存储和处理提供了良好的扩展性，增加基线服务器即增加系统的容量并且提升系统的处理能力；

(2) 每个 tablet 有多个副本 (缺省 3 个) 并且分布在不同服务器上，整个系统虽然使用了普通的主流服务器 (Commodity server)，却能够提供足够高的可靠性和可用性 (参见 3.6“数据可靠性”)，不仅性价比更高，多个数据副本还可有效遏制访问热点的产生；

(3) 静态的基线数据避免了并发读写的冲突，省去了并发读写所需的访问控制，文件删除可直接通过过期回收机制来实现，多副本数据一致性则通过文件的校验和 (Checksum) 完成，简化了系统设计和实现；

(4) 集中式的修改避免了多点修改导致的冲突和不一致，也避免了分布式写事务并降低了写事务的响应时间，读写事务的实现与单机 RDBMS 系统类似；

(5) 修改增量通常在增量服务器的内存中，提高了读写事务的性能并缩短了响应时间；

(6) 摒弃了随机磁盘写，可充分发挥固态盘 (SSD) 良好的随机读性能，减少了服务器数量，不仅减少了服务器数量，单台服务器的功耗也降低了；

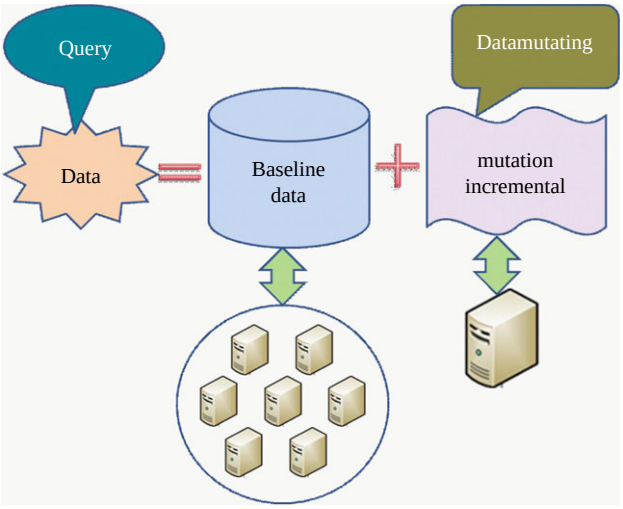

图1 OceanBase 系统原理图

Fig. 1 Principle of OceanBase

此方案可能的不足/风险：

(1) 每次查询需要获取基线数据和对应的修改增量 (通常是两次远程调用)，增加了事务的响应时间 (RT)：由于一个 OceanBase 机群通常部署在同一数据中心，因此累计增加的响应时间通常在1毫秒以内；

(2) 每次查询需要融合基线数据和对应的修改增量，增加了 CPU 的消耗：由于修改的记录通常只是整个数据中很少的一部分 (例如百分之几)，需要融合的数据也仅仅是很小的比例；

(3) 增量服务器导致单点故障：增量服务器通常配置为主备结构 (甚至一主多备)，主备之间实时同步，主机故障后，备机会实时升级为主机；

(4) 增量服务器成为容量瓶颈：修改增量累积到一定量后可以转存到固态盘或者分发到对应的基线服务器，避免了增量服务器的存储 (尤其是内存) 容量成为瓶颈；

(5) 增量服务器成为性能瓶颈：增量服务器是主备结构，主备之间实时同步，备机提供读服务，多网卡以及万兆网卡避免了增量服务器的网络带宽成为整个系统的瓶颈。由于修改增量数据通常在内存中，因此写事务能够达到很高的性能。不过，由于增量服务器主机是单一的写入点，因此整个系统仍然尽可能地降低其负载，比如由 MergeServer (参见 2.2“模块构成”) 对写事务进行预处理；

(6) 每日合并 (融合冻结的修改增量与现有基线数据以产生新的基线数据) 导致 CPU/磁盘 IO 的消耗，降低系统的处理能力，并增加事务的响应时间 (RT)：每日合并是低优先级任务，通常在数据库每天的访问低峰时段进行 (例如凌晨 1:00~5:00)，且一般是顺序读和顺序写，对应用的影响很小；如果配置主备机群 (参加 3.4“容灾”)，则两个机群的每日合并在时间上可以完全错开，从而彻底消除每日合并对应用的影响；

2.2 模块构成

OceanBase 由 SQL 客户端、UpdateServer、MergeServer、ChunkServer 和 RootServer 等几个模块组成，图 2 是 OceanBase 的一种典型部署。

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

• SQL 客户端

作为一个开源系统，OceanBase 支持开源的 MySQL 客户端，并支持 ODBC 和 JDBC 接口。

• UpdateServer 模块

UpdateServer 模块是 OceanBase 的修改增量管理模块，通常为主备结构并部署在两台或多台增量服务器上。主要功能是：

(1) 执行写事务的修改 (增删改等) 操作并持久化 redo log；

(2) 为读请求提供对应的修改增量；

(3) 根据配置或数据库管理员 (DBA) 的指令冻结现有的修改增量表并开启新的修改增量表；

(4) 通过 redo log 保持主备增量服务器以及主备机群数据的增量服务器的同步；

(5) (因宕机、其他异常或管理员) 重启后根据 redo log 重构修改增量表；

(6) 根据 RootServer 指令进行主备角色切换；

• MergeServer 模块

MergeServer 是 OceanBase 的用户接口模块和事务执行的协调者。主要功能是：

(1) 接收并解释用户的 SQL 请求，生成并优化 SQL 执行计划；

(2) 根据 SQL 执行计划向一个或多个 ChunkServer 发起读请求，和/或向 UpdateServer 发起写请求，并把结果返回用户；

• ChunkServer 模块

ChunkServer 模块是基线数据的管理模块。主要功能是：

(1) 管理以 tablet 为单元的数据库的基线数据；

(2) 提供基线数据以及融合基线数据与修改增量 (以产生实时数据) 的读访问服务；

(3) 根据 RootServer 的指令在 ChunkServer 之间迁移/复制基线数据 tablet；

(4) 根据 RootServer 的指令合并基线数据与冻结的修改增量并产生新的基线数据 (每日合并)；

(5) 分裂较大的 tablet，并根据 RootServer 的指令合并相邻且较小的 tablet；

• RootServer 模块

RootServer 是 OceanBase 系统的控制模块，通常一主一备并由外部系统进行监控和主备切换。主要功能是：

(1) 管理主备 UpdateServer，当主 UpdateServer 出现异常时选择并通知合适的备 UpdateServer 升级为主；

(2) 管理 MergeServer，对外部应用提供 MergeServer 列表以便应用连接 MergeServer 并执行数据库的查询修改等操作；

(3) 管理 ChunkServer，根据需要通知 ChunkServer 之间复制/迁移基线数据 tablet，比如当 tablet 副本数量不足，或者因为增加/减少机器/增删改数据等导致

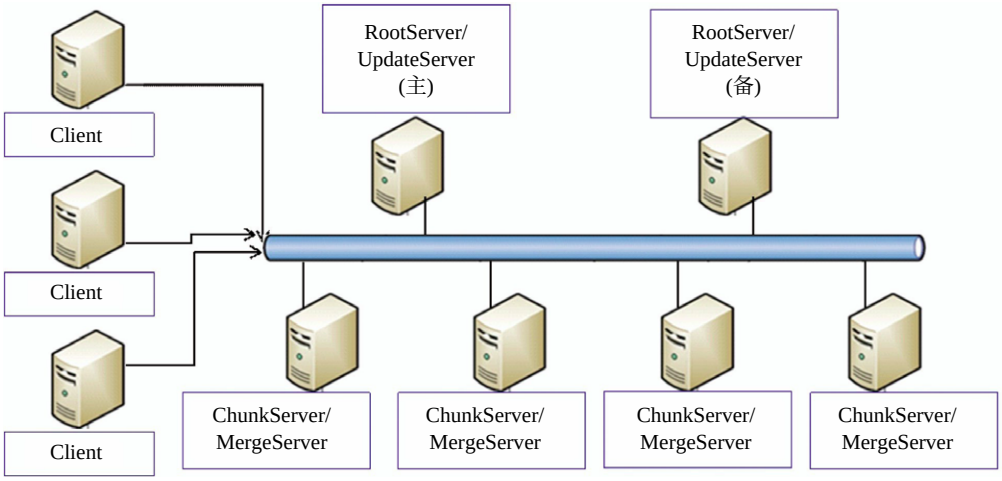

图2 **OceanBase** 系统架构图

Fig. 2 Infrasture of OceanBase

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

ChunkServer 之间负载不平衡；通知 ChunkServer 删除不需要的基线数据 tablet；

(4) 通知 ChunkServer 发起每日合并以便融合当前的基线数据与冻结的修改增量，从而产生新的基线数据；

(5) 实时同步主备 RootServer；

3 系统实现

3.1 读事务

MergeServer 接收到应用的读事务请求后的执行步骤：

(1) 解释接收到 SQL 请求，生成并优化执行计划；

(2) 向一个或多个 ChunkServer 发起查询指令，ChunkServer 融合基线数据与 UpdateServer 的修改增量并返回；

(3) 合并 ChunkServer 的查询结果并做必要的处理；

(4) 返回结果给应用。

3.2 写事务

MergeServer 接收到应用的写事务请求后的执行步骤：

(1) 解释接收到 SQL 请求，生成并优化执行计划；

(2) 对 SQL 执行计划进行预处理；

(3) 把预处理后的 SQL 执行计划发送给 UpdateServer，并由 UpdateServer 完成修改操作后返回 MergeServer；

(4) 返回结果给应用。

3.3 每日合并

因为采用修改增量的模式，所以 OceanBase 需要避免修改增量无限制地增长，因此 OceanBase 周期地把修改增量融合到当前基线数据以产生新的基线数据，并删除旧的修改增量和旧的基线数据，这个工作通常在数据库每天的低峰时段 (例如凌晨 1:00 ～ 5:00) 进行，所以称为“每日合并”。由于同一 tablet 的多个副本内容是相同的，因此有两种做法：

• 每个 tablet 选择一个副本与修改增量进行融合，完成后复制到相关的基线服务器；

• 每个 ChunkServer 独立对其所管理的 tablet 副本与修改增量进行融合；

OceanBase 选择了第二种方案，主要原因是：

(1) 基线服务器的网络带宽是有限的 (当前主流服务器配备 1 Gbps 网卡，给所有基线服务器配置多网卡或者万兆网卡会导致成本的增加)，以 100MB/s 速度复制 2TB 数据需要 20 000 秒；各个 ChunkServer 独立进行每日合并则网络只需传输从 UpdateServer 获得的修改增量，并且可以充分发挥每个 ChunkServer 多处理器/多核以及多磁盘的优势；

(2) 每个 tablet 选择一个副本进行合并然后分发到相关基线服务器时，需要处理被选择的 ChunkServer 可能因为宕机或其他异常导致合并失败或者复制失败等例外情形；而各个 ChunkServer 独立进行每日合并则无需关注这些异常 (RootServer 已有监控管理 ChunkServer 并处理 tablet 副本数异常的机制)。

每日合并也曾经带来了一些问题，比如刚开始的时候，所有 ChunkServer (每个 ChunkServer 多个每日合并线程) 几乎同时启动并立即向 UpdateServer 请求了大量的修改增量，这可能导致 UpdateServer 网络拥塞，这个问题通过让 ChunkServer 在某个时间范围内随机选择一个时刻开始每日合并而得到解决。另外，程序 bug 或其他异常可能导致 tablet 副本不一致，并且这种不一致还可能随着每日合并而扩散/放大，为了防止这种情况发生，ChunkServer 进行每日合并时会对新生成的 tablet 文件计算一个 checksum 并汇报给 RootServer，后者比较 tablet 多个副本的 checksum 并报告发现的异常。

3.4 容灾

每个 tablet 的多个副本分布在不同 ChunkServer 上以及 UpdateServer 的主备策略使得 OceanBase 系统具备了很高的可靠性 (参见 3.6“数据可靠性”)，然而，机房的电源、网络/交换机等故障仍然可能导致服务的中断，因此 OceanBase 还实现了跨机房容灾的功能，如图 3 所示 (此图没有显示 RootServer)：

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

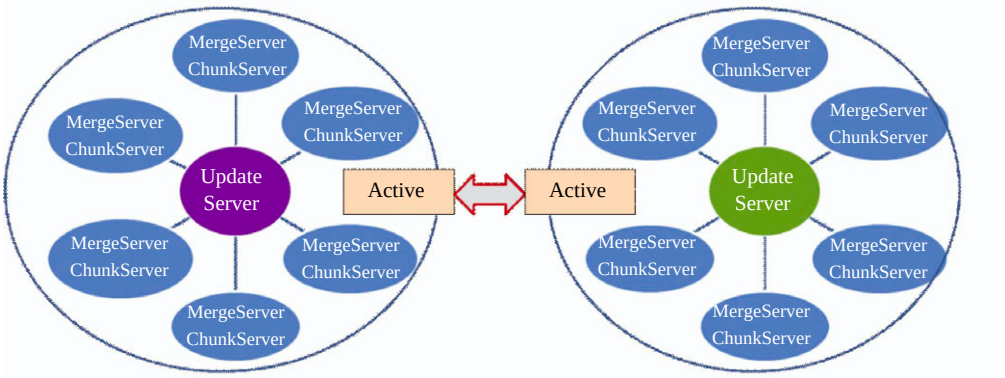

图3 OceanBase 主备机群

Fig. 3 Master and slave cluster of OceanBase

两个 (或多个) 机群中有一个是主机群，接受应用的写事务，其他机群是备机群，通过 redo log 保持与主机群的同步。如果两个机群之间的网络延时比较小(比如在同一城域内，网络延时可能在 1 毫秒以内)，那么备机群可以配置成与主机群实时同步并对外提供读服务；否则，主备机群可以配置成准实时同步，备机群的数据可能落后于主机群十几毫秒到几十毫秒等等。

容灾配置已经在阿里巴巴生产系统中部署并使用，相应的业务没有因为机房的电力或者网络交换机等的偶然故障而中断。此外，面对年底的“双十一”(11 月 11 日)、“双十二”(12 月 12 日) 大促带来的流量的陡增，备机群提供的读服务分担了主机群的压力，减少了一半左右服务器的数量。

3.5 固态盘

传统机械硬盘的随机 I/O 性能一般不超过 200 IOPS (每秒随机 I/O 次数) 且响应时间通常在 5~13 毫秒之间，固态盘 (SSD) 的随机读性能通常高于 20 000 IOPS 且响应时间小于 0.5 毫秒。按单位存储容量计算，固态盘单价大大高于机械硬盘，如果按单个 IOPS 计算，则机械硬盘的单价大大高于固态盘。鉴于数据库对存储容量的需求相对较小而对 IOPS 的需求很大，固态盘有很大的潜在优势。

但是，固态盘的写入以页单位，通常每个页 4KB 或者更大，在一个页内，不管写入多少内容都必须把整个页读出来，与写入内容合并，然后再把整个页写入，所以固态盘的随机写常常会产生很严重的写入放大 (Write amplification)；此外，固态盘是“先擦除再写入”模式，每个页只能写入一次，下次写入前必须先擦除，但擦除以条带 (Band) 为单位，条带 (Band) 通常很大，例如一个 300 GB 的固态盘的条带 (Band) 可能是 80 MB (与厂商/型号/规格等相关)，这使得固态盘在擦除前通常得把该条带 (Band) 上的全部有效数据都读出并在擦除后再次写入，而刚刚擦除过的空页则可能很快被后续的随机写消耗掉，这进一步加剧了固态盘的写入放大效应。因此，固态盘的随机写性能甚至不如机械硬盘，从而难以直接应用到现有数据库系统。

OceanBase 采用修改增量模式，规避了随机磁盘写，因此固态盘成为 OceanBase 极佳的存储装备 (Redo log，虽然是顺序追加写，由于每次写入数据量较小，仍然采用机械硬盘)。每日合并时，新的基线数据是大块顺序写入，这很好地避免了固态盘的写入放大。

擦写次数少是固态盘的另一个劣势，机械硬盘一般有 10 万次以上的擦写寿命，而固态盘通常只有 5 000~10 000 次擦写寿命。对于 OceanBase，一次每日合并最多把整个固态盘全部擦写一遍，对于擦写寿命为 5 000 次的固态盘，其使用寿命能达到 5 000 天或以上。

基于固态盘的 OceanBase 系统在阿里巴巴生产系统中发挥了重要作用，以淘宝的收藏夹为例，2010 年中，使用 32 台 SAS 盘服务器 (10 块 SAS 数据盘/服

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

务器)，系统只能提供 8 000QPS 的服务能力；2012 年 12 月，使用 80 台 SSD 服务器 (10 块 SSD 数据盘/服务器)，系统提供了 117 000QPS 的服务能力，即以 2.5 倍服务器数量，提供了 14.625 倍的数据库访问量。

3.6 数据可靠性

作为底层基础设施，数据库必须保证数据的可靠性。与经典 RDBMS 使用可靠性较高的高端服务器和高端存储设备相比，主流服务器的可靠性相对较低。OceanBase 使用这些主流服务器，数据可靠性如何？

假设机群内的服务器的规格和新旧状态相同，并假设 tablet 数据是均匀和随机分布的。当一台 ChunkServer 故障后，tablet 的副本数可能低于设定值，RootServer 将协调其余活着的 ChunkServer 进行 tablet 数据复制；如果其间又有 ChunkServer 宕机了，这时将有更多的 tablet 的副本树低于设定值，剩下的 ChunkServer 会继续复制；如果一些 ChunkServer 宕机导致某些 tablet 的所有副本都不能访问，那么数据就不可用了。

假设服务器的平均无故障时间是 T，每台 ChunkServer 上的数据量是 S，每个 ChunkServer 复制 tablet 占用的网络带宽是 W (预留网络带宽以便继续对外进行读写服务)，ChunkServer 数量是 N：

一台服务器故障后，余下服务器恢复数据所需时间是：$t(1) = S/(N-1)/W$；

两台服务器故障后，余下服务器恢复数据所需时间是：$t(2) = 2* S/(N-2)/W$；

单台服务器在时间 t 内故障的概率是：$a = t/T$；

N 台服务器在时间 t 内至少1台故障的概率是：

$f(N,1) = 1 - (1-a)^N$；

N 台服务器在时间 $t(1)$ 内至少 2 台故障的概率是：

$f(N,2) = 1 - (1-a(1))^N - N*a(1)*(1-a(1))^{N-1}$；

N 台服务器在时间 $t(2)$ 内至少 3 台故障的概率是：

$f(N,3) = 1 - (1-a(2))^N - N*a(2)*(1-a(2))^{N-1} - N*(N-1)/2!*a(2)^2*(1-a(2))^{N-2}$；

……

假设 T=25 000小时 (当前主流服务器通常有 5 万小时以上的平均无故障时间)；W=50MB/s (主流服务器通常配置千兆网络)；单机数据量 S=1TB，2TB，5TB 三种情形；服务器台数 N=5，25，125 和 500 四种情况，计算 3 台或以上服务器故障的概率 $f(N,3)$，结果如图 4 所示。

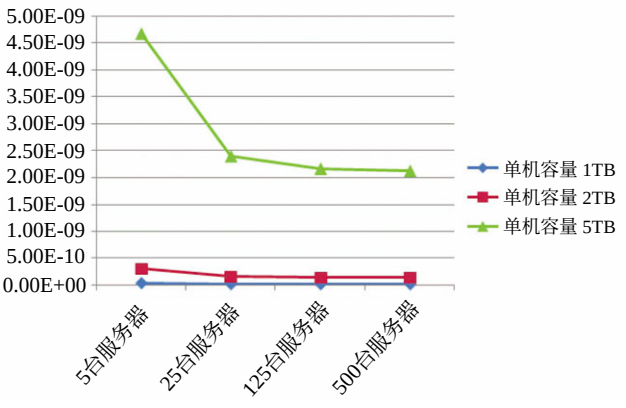

图4 三台或以上服务器故障的概率

Fig. 4 Possibility of failure of 3 or more servers

由此可知，尽管使用可靠性相对较低的普通服务器，如果数据为 3 个副本且不考虑服务器之外的其他因素，则数据不可用的几率是非常小的 (约 10^{-9}~10^{-11})。

4 相关工作和总结

Google Percolator [1] 和 Mcgastore [2] 是基于 Bigtable [3] 完成的 share nothing 分布式数据库，使用两阶段提交协议[4] (Two-phase commit protocol) 实现分布式事务，具有良好的扩展性并通过其扩展性实现了很高的整体事务性能 (Percolator 在 15 000 CPU 核上达到了 11 200 TPC-E tps)。Percolator/Megastre 通过 Bigtable 的单行事务功能实现了通用事务，数据访问需要经过 Bigtable 到 GFS[5] 再到 Linux 文件系统等环节，这些对它们的事务响应时间以及单机效率等产生了较大的影响。除了 GFS master 节点外，Percolator 机群中的节点是对等的。Google Spanner[6] 通过自定义的 True Tine API (基于全球定位系统 GPS 以及原子钟) 实现了全球范围内事务处理，并采用了新一代的 GPS 文件系统 Colossus。

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

OceanBase 机群中的节点是非对等的，除了控制节点 RootServer 之外，增量服务器 (UpdateServer) 用于实现修改操作和管理修改增量，而基线服务器 (ChunkServer) 则用于静态的基线数据的存储访问以及基线数据与增量数据的实时融合等。基线数据的分布式存储确保了系统容量和性能的扩展能力，集中化的修改操作则避免了分布式事务[4]，不仅简化了系统的架构和实现[7–8]，还提高了事务执行的效率、缩短了事务的响应时间。尽管采用了单一的增量服务器 (UpdateServer)，但由于数据库在一段时间内的修改操作总量有限，修改增量数据一般情况下得以常驻内容，此外由于备机可以提供读服务并而增量服务器可按需向基线服务器分发修改增量等，实际应用中增量服务器并没有成为整个系统的容量和性能瓶颈。

OceanBase 已经用于阿里巴巴的多个线上项目，每天提供超过十亿次的实时读写访问服务。最大的单表超过 100 亿条记录、3TB 数据。

参考文献

[1] Daniel Peng and Frank Dabek.Large-scale Incremental Processing Using Distributed Transactions and Notifications.Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation, USENIX (2010).

[2] Jason Baker, etc. Megastore: Providing Scalable, Highly Available Storage for Interactive Services, Proceedings of the Conference on Innovative Data system Research (CIDR) (2011), pp. 223–234.

[3] Fay Chang. etc. Bigtable: A Distributed Storage System for Structured Data. OSDI'06: Seventh Symposium on Operating System Design and Implementation, Seattle, WA, November, 2006.

[4] Irving L. Traiger, James N. Gray, Cesare A. Galtieri, Bruce G. Lindsay. Transactions and Consistency in Distributed Database Systems. ACM Transactions on Database Systems (TODS), Volume 7 Issue 3, Sept. 1982, Pages 323–342.

[5] Sanjay Ghemawat, Howard Gobioff, Shun-Tak Leung. The Google File System, 19th ACM Symposium on Operating Systems Principles, Lake George, NY, October, 2003.

[6] James C. Corbett, etc. Spanner: Google's Globally-Distributed Database, the Proceedings of OSDI'12: Tenth Symposium on Operating System Design and Implementation, October, 2012.

[7] David J. DeWitt, Jim Gray. Parallel Database Systems: The Future of High Performance Database Processing. Communications of the ACM, Vol. 36, No. 6, June 1992.

[8] Haran Boral, etc. Prototyping Bubba, A Highly Parallel Database System, IEEE Transactions On Knowledge And Data Engineering,Vol. 2, No. 1, March 1990.

收稿日期：2012 年 8 月 5 日

阳振坤：阿里巴巴高级研究员，博士。主要研究方向为海量数据库系统的研究与设计、云计算系统等。
E-mail: *zhengxiang@taobao.com*

杨传辉：阿里巴巴高级技术专家，硕士。主要研究方向分布式存储系统的研究与设计。
E-mail: *rizhao.ych@taobao.com*

李震：阿里巴巴资深经理，学士。主要研究方向为分布式数据存储，海量数据计算等。
E-mail: *chucai@taobao.com*

(C)1994-2021 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net