浙江工业大学

本科毕业设计外文翻译

（2013 届）

论文题目　RAMClouds：可扩展的高性能内存存储

作者姓名	陈佳鹏
指导教师	陈　波
学科 (专业)	软件工程
所在学院	计算机科学与技术学院
提交日期	2013 年 06 月

RAMClouds：可扩展的高性能内存存储

摘要： 在线存储的磁盘的方法正成为越来越严重的问题：他们的规模不能很好的满足大型 Web 应用程序，对磁盘容量的改善已经远远超过了在访问延迟和带宽的改善。本文提出一种关于数据中心存储新的方法：RAMCloud，信息完全保存在 DRAM 和大型系统所建立的聚集成千上万的商品服务器的主存储器上。我们认为，RAMClouds 可以提供基于磁盘的系统、100-1000 倍低访问延迟和吞吐量的持久的和可用的存储。低延迟和大规模的结合，提供了一种新的数据密集型应用程序。

关键字： DRAM

一、引言

四十年来，磁盘是存储计算机系统中在线信息的主要手段。在此期间磁盘技术经历了巨大的改进，它一直被更高级别的存储系统利用，如文件系统和关系数据库。但是，磁盘的性能并没有像它的容量那样迅速改善，为了满足大型的 Web 应用程序的需要，开发人员发现越来越难测量基于磁盘的系统。很多人都提出了新的基于磁盘的存储方法来解决这个问题，另外的人建议用 闪存设备更换磁盘。相比之下，我们认为，解决的办法是将磁盘中的主要核心数据转移到随机存取存储器上，磁盘变成备份/归档的角色。

在本文中，我们认为，一类被称为 RAMCloud 的新存储将为许多的未来的应用提供存储基底。一个 RAMCloud 将所有的信息存储在商品服务器的主存储器中，使用数百或数千个这样的服务器创建一个大型存储系统。因为任何时候所有的数据都是在 DRAM，RAMCloud 可提供基于磁盘的系统、100-1000 倍低访问延时和吞吐量的持久的和可用的存储。虽然个别的内存是不稳定的，RAMCloud 可以使用复制和备份技术确保数据的耐用性和可用性，这等同于基于磁盘的系统。

我们认为 RAMClouds 将在三个方面从根本上改变存储行业的面貌。第一，通过省去了许多的可扩展性问题，这些问题削弱了当今开发人员的工作效率，它们简化了大型的 Web 应用程序的开发。第二，极低的延时将提供更丰富的查询模型，提供了一种新的数据密集型应用程序。第三，RAMClouds 将提供需要“云计算”的可扩展存储基底和其他数据中心的应用程序。一个 RAMCloud 可以支持一个大型应用程序或许多小应用程序，并允许小型应用程序快速的扩展到大型应用程序，无需为开发人员增加额外的复杂性。

二、RAMCloud 概念

RAMClouds 是最有可能被用于在含有大量服务器的数据中心，粗略地划分分为两类：应用服务器，用来实现应用程序的逻辑，如生成 Web 页面或执行业务规

则;存储服务器,提供长期共享存储的应用程序服务器。传统以来,存储由文件或关系型数据库组成。但在近年来,为了提高可扩展性,开发出了各种新的存储机制,如 Bigtable 和 memcached。每个数据中心通常支持许多应用,从范围从只有一个应用程序的服务器小应用到千上万的专用应用程序和存储服务器组成的大型应用。

RAMCloud 代表了在系统中组织存储服务器一种新的方式。有两个关键属性用来区分 RAMCloud 和其他存储系统。首先,任何时刻所有的信息都保存在 DRAM 中。RAMCloud 不是像 memcached 那样的缓存,且数据不是存储在一个 I/O 设备上。DRAM 是数据永久的家,磁盘仅用于备份。其次,RAMCloud 必须自动扩展以支持数以千计的存储服务器,应用程序看到的是一个单一的存储系统,和存储服务器的实际数目不相关。

存储在 RAMCloud 上的信息必须像存储在磁盘上那样的耐用。例如,单个储存服务器的出错不应该造成数据丢失或几秒钟的系统失效。第四章第二节讨论实现这种级别的持久性和有效性的系统所需要的技术。

表 2.1 总结了现在 RAMCloud 的配置,此配置假设每台服务器有 64GB 的内存,也就是现在最经济有效范围内最大的内存(随着内存大小的增加,价格会有显著的增长)。这个配置提供了 1000 台服务器共 64TB 的储存空间,每 GB60 美元。算上附加的服务器,现在有可能构建容量大约 500TB 的 RAMClouds。在 5 到 10 年内,假设 DRAM 的技术会一直有进步,那么就有可能以小于每 GB5 美元构建容量达 1 至 10PB 的 RAMClouds。

表 2.1: RAMClouds 参考配置

# 服务器	1000
每台服务器的容量	64GB
总容量	64TB
服务器总开销	4000,000$
每 GB 开销	60$
总吞吐量	10^9

三、动机

3.1 应用程序扩展性

RAMClouds 产生的两个动机:应用程序、技术。从应用程序的立场来看,几十年来,关系型数据库被拿来做储存系统,但是关系型数据库无法适应现在大规模应用程序这个级别所需要的数据规模。实质上,每个流行的 Web 应用程序也已经意识到单个关系型数据库已经无法满足吞吐量的要求。随着网站的膨胀,它必须要经历一系列大量的修改,每一个系列都会引进专案技术来扩展其存储系统,

如分割数据到多个数据库中。这些技术在一段时间内有了成效，但是当网站到达一个新级别的规模时，扩展性的问题又出现了，那就需要更特殊更有目的的技术了。

例如，到 2009 年 8 月，Facebook 的储存系统已经包含了 4000 个 MySQL 服务器。数据分布在实例之间，实例之间的一致性明确地由 Facebook 应用程序的代码管理。即使这样，这些数据库服务器也不能满足 Facebook 吞吐量的要求，所以 Facebook 同时附加了 2000 台 memcached 服务器，这些服务器把最近使用的查询结果以 Key-Value 的形式存储在主内存中。不幸的是，memcached 和 MySQL 服务器之间的一致性必须由软件管理，这增加了应用程序的复杂性。

3.2 技术趋势

RAMClouds 的第二动机来源于磁盘技术的演化（见表 3.1 ）。在过去的 25 年中，磁盘容量增加了至少 10000 倍，而且有可能会在未来继续增加。不幸的是，尽管这样，磁盘上信息的访问率提高的更加慢：大数据块之间的传输率只提高了 50 倍，而寻道时间和旋转等待时间只提高了 1/2。

表 3.1: 25 年前和现在的硬盘技术对比

	1980 年代中期	2009 年	改善
硬盘容量	30 MB	500 GB	16667x
最大传输率	2 MB/s	100 MB/s	50x
延时（寻道 + 旋转）	20 ms	10 ms	2x
容量/带宽（大块）	15 s	5000 s	333x
容量/带宽（小块）	600 s	58 days	8333x

3.3 缓存

从历史上看，高速缓存被视为解决磁盘延时问题的答案：如果大多数访问集中在磁盘块一个小的子集，在 DRAM 中保存最频繁访问的块就能实现高性能。在理想情况下，带有缓存的系统可以用磁盘级别的成本提供类似 DRAM 级别的性能。

但是，由于越来越大的数据需要被存在 DRAM 中，表 3.1 的趋势淡化了缓存所带来的好处。而且一些新 Web 引用程序（例如 Facebook）靠着数据之间复杂的联系几乎脱离了本地化。到 2009 年 8 月的时候，Facebook 大约 25% 的线上数据被保存在 memcached 服务器的主内存中，提供了 96.5% 的命中率。当算上数据库服务器上的缓存时，存储系统所用的总内存大约等于总数据的 75%。因此，RAMCloud 只会为 Facebook 提高大约 1/3 的内存使用率。

3.4 延时是否重要？

RAMClouds 和其他存储系统相比，最独特的方面在于极低的延时。一方面，5 到 10μs 延时可能并不显得那么重要，而且很难说现如今的 Web 应用程序需要这种级别的延时。但是，我们相信低延时是 RAMClouds 特点里面最重要的。

现如今的 Web 应用程序已经开始以复杂的方式使用数据并且和延时问题做斗争。例如，当 Facebook 收到一条 HTTP 请求时，应用程序服务器在内部将其均分成 130 条请求，然后组合出结果页面。这些请求必须按顺序被处理，因为之后的请求需要之前的请求所产生的结果。这些内部请求所积累的延时是限制响应给用户的时间的一个因素之一，所以大多数开发人员的功夫花费在最小化请求的数量和大小上。Amazon 发表了相似的结果，使用 100 到 200 的请求去生成每个页面的 HTML 代码。

四、问题研究

4.1 低延时 RPC

虽然已经在专门的网络实现小于 10μs 的延时，如 Infiniband 和 Myrinet，大多数现有的数据中心使用的网络基础设施基于以太网/IP/TCP，典型的往返过程的远程调用需要 300 到 500μs，但是这样做的话在某些级别上需要新的设备。今天，最大的障碍是网络交换机的延时。一个大型的数据中心可能有一个三层式交换结构，所以每一个数据包穿过五个交换机，因为它往上移动穿过交换层后回落到目的地。如今每个典型的交换机造成 10 微秒的延时，在每个方向上造成至少 100 μs 的交换延时。更新的 10GB 交换机（例如 Arista 7100S）使用通过路由的办法，并且要求至多 1 μs 的延时，但是这样还是会造成每个方向上总共大约 5 μs 网络延时。为了达到 RPC 时间小于 10 μs 的要求，需要对交换延时做附加改进。

4.2 持久性和可用性

为了 RAMClouds 得到广泛的应用，他们必须提供高层次的持久性和可用性（至少和如今的基于磁盘的系统一样好）。这意味着一台服务器的崩溃不会造成数据丢失或几秒钟内影响系统的可用性。RAMClouds 必须还提供稳定的保护，防止电源停电，可能需要跨数据中心复制。我们假设对数据持久性的任何保证在存储服务器响应写请求的时候生效。

在 DRAM 中为每个对象保存一个副本是持久性可以采取的一种途径，但是在每个服务器的本地磁盘上备份新数据可以作为每个写操作的一部分。然而，这

种做法是不可取的，因为它写入延时累加到磁盘延时上，如果服务器崩溃，数据还是不可用。至少必须讲数据复制到多台服务器机子上。

五、RAMCloud 缺点

RAMClouds 最明显的缺点是每比特位的高额费用和高能源消耗。由于这些指标，纯粹基于磁盘的系统是 RAMCloud 的 50-100 倍，基于闪存的存储系统是 RAMCloud 的 5-10 倍。和基于磁盘和闪存的系统，RAMCloud 系在数据中心中需要更多的地面空间。因此，如果应用程序需要廉价地存储大量的数据且具有相对低的访问速率，RAMCloud 不是最好的解决方案。

六、相关工作

几十年来，在存储系统中，DRAM 的作用一直在稳步增加，许多 RAMCloud 想法已经扩展到其他系统中。例如，在 20 世纪 80 年代中期，有许多与数据库中存储有关的实验研究都完全在主存储器中进行。但是，主内存中的数据库没有被广泛采用，也许因为他们的能力有限。对 DRAM 存储系统的延时优化产生的效益已经在其他项目中得到论证，例如 Rio Vista。

七、总结

在未来，技术的发展趋势和应用的需求将决定越来越多的在线数据会被保存在 DRAM 内。在本文中，我们认为，对于许多应用程序而言最好的长期 解决方案可能是一个激进的方法，所有的数据在任何时间都保存在 DRAM 内。RAMClouds 最重要的两个方面：极低的延迟，聚集大量的商业型服务器的资源的能力。两者合一，这些方面使得 RAMClouds 能扩展到满足最大的 Web 应用程序的要求。此外，低延时能丰富更多的查询模型，这些模型会简化应用程序的开发且诞生新种类的应用程序。最后，这种途径使得 RAMClouds 能为需要灵活的可扩展的存储系统的云计算平台提供了吸引人的基底。

参考文献

[1] Andersen D G, Franklin J, Kaminsky M, et al. FAWN: A fast array of wimpy nodes[C]. Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, 2009:1--14.

[2] Arista Networks 7100 Series Switches[C]. .

[3] Fox A, Griffith R, et al. Above the clouds: A Berkeley view of cloud computing[J]. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS, 2009, 28.

[4] Chang F, Dean J, Ghemawat S, et al. Bigtable: A distributed storage system for structured data[J]. ACM Transactions on Computer Systems (TOCS), 2008, 26(2):4.

[5] Chun B N, Mainwaring A M, Culler D E. Virtual network transport protocols for Myrinet[J]. Micro, IEEE, 1998, 18(1):53--63.

[6] Cooper B F, Ramakrishnan R, Srivastava U, et al. PNUTS: Yahoo!'s hosted data serving platform[J]. Proceedings of the VLDB Endowment, 2008, 1(2):1277--1288.

[7] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1):107--113.

[8] DeCandia G, Hastorun D, Jampani M, et al. Dynamo: amazon's highly available key-value store[C]. ACM SIGOPS Operating Systems Review, 2007，41:205--220.

[9] DeWitt D J, Katz R H, Olken F, et al. Implementation techniques for main memory database systems[M]. Vol. 14.[S.l.]: ACM, 1984.

[10] Dittia Z. Integrated hardware/software design of a high performance network interface[M]. [S.l.]: Washington University, 2001.

[11] Garcia-Molina H, Salem K. Main memory database systems: An overview[J]. Knowledge and Data Engineering, IEEE Transactions on, 1992, 4(6):509--516.

[12] Gray J, Putzolu F. The 5 minute rule for trading memory for disc accesses and the 10 byte rule for trading memory for CPU time[C]. ACM SIGMOD Record, 1987，16:395--398.

[13] Kallman R, Kimura H, Natkins J, et al. H-store: a high-performance, distributed main memory transaction processing system[J]. Proceedings of the VLDB Endowment, 2008, 1(2):1496--1499.

[14] Lowell D E, Chen P M. Free transactions with rio vista[C]. ACM SIGOPS Operating Systems Review, 1997，31:92--101.

[15] http://memcached.org. Memcached: a distributed memory object caching system[缺文献类型标志代码].

[16] Patterson D A, Gibson G, Katz R H. A case for redundant arrays of inexpensive disks (RAID) [M]. Vol. 17.[S.l.]: ACM, 1988.

[17] Ramakrishnan R, Gehrke J, Gehrke J. Database management systems[M]. Vol. 3.[S.l.]: McGraw-Hill, 2003.

[18] Haerder T, Reuter A. Principles of transaction-oriented database recovery[J]. ACM Computing Surveys (CSUR), 1983, 15(4):287--317.

[19] Robbins S. RAM is the new disk...[缺文献类型标志代码].

[20] Rosenblum M, Ousterhout J K. The design and implementation of a log-structured file system[J]. ACM Transactions on Computer Systems (TOCS), 1992, 10(1):26--52.

[21] Schroeder B, Pinheiro E, Weber W D. DRAM errors in the wild: a large-scale field study[C]. Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems, 2009:193--204.

[22] Stonebraker M, Madden S, Abadi D J, et al. The end of an architectural era:(it's time for a complete rewrite)[C]. Proceedings of the 33rd international conference on Very large data bases, 2007:1150--1160.