计算技术研究所，微处理器中心

论文阅读报告　其4
The Datacenter as a Computer: An Introduction to the Design of Warehouse
第1章、第3章

谭弘泽
201828013229048

October 17, 2018

Contents

1 简介

在计算机网络已经建立数十年的现在，出现了很多热门的互联网服务和云计算。
很多年前如网游等一些需要计算量的应用常常采取服务器-客户端模式。但是随着云计算的普及，存储和计算能力的增长，由于方便更新迭代，方便管理等原因，逐渐都移向了服务器端计算。
这个趋势创造了一类新的计算系统——仓库规模计算机（WSC,warehouse-scale computer）。这些机器有着重量级的软件基础、数据仓库和硬件平台。程序是互联网服务，要跑在大量的机器上，并且包含很多子程序。

1.1 仓库规模计算机

当规模是唯一可分辨的特征的时候我们一般把这些机器统称为数据中心。而仓库规模计算机WSC是一种数据中心。传统的数据中心往往主持一些中小规模的服务，而这些主机硬件和软件往往是不同组织单元甚至不同公司的。
而WSC只属于一个公司，使用相对统一的硬件和系统软件平台，并且共享公共的系统管理层。最重要的是，WSC运行更小数量的超大的应用，而共同的资源管理基础设施允许极大的部署灵活性。
互联网应用必须达到高可用性，典型地会至少以99.99%正常运行时间为目标（四个9，大约是每年坏一个小时）。WSC要支持能容忍大数量的组成部件出错而不导致服务的性能和可用性。

1.2 规模上的成本效率

计算需求的增长主要被三个主要因素驱动:

1. 增加的服务热度会增加请求负载。
2. 问题规模在逐渐增长。
3. 即使吞吐量和数据量都不变，日益激烈的市场竞争也会要求更大。

成本效率也必须计算其他的明显的开销组成部分，包括主机设备容量，活动消耗（能源供给和能量花费），硬件，软件，管理人员，和修理。

1.3 不只是服务器的聚集

不能简单地把数据中心看作放在一起的一大堆服务器，在这些系统上运行的软件，不止会在单独的一台机器或者单独的一个机柜中运行，而会在数百甚至上千个独立的服务器上运行。这样的机器——大量的服务器聚簇本身，需要被考虑为一个整体的计算单元。
首先，巨大而迅速增长的工作负载会很难模拟。附加地，错误行为，功率能量考量在WSC中会有更加显著的影响。最后，WSC比起单独一个服务器或者一小组服务器又多了一层复杂性。这些附加的复杂性间接地来自于更深更异质的存储层次（本章后续讨论），更高的错误率（第7章），以及很可能更高的性能抖动（第二章）。
（本次论文/文章阅读并不会都读完。）

这本书的目标是向读者介绍这些新的设计空间，描述一些WSC的需要和特性，强调部分这一领域独特的重大挑战，并且分享部分我们在Google的设计、编程、和操作WSC的经验。

1.4 一个数据中心对比于几个数据中心

这本书中，我们把一个数据中心的层次当成计算机，即使有许多互联网应用会使用多个地点上相隔甚远的数据中心。
这么做的典型例子是为了容忍灾难发生，将非易变的数据做了多份备份。虽然有很多种图景可以选择，选择把一个数据中心的层次当成计算机的一个原因是，同数据中心和数据中心间通讯的连接质量有巨大的间隙。如果将来这个间隙能够显著的缩小的话，我们需要调整我们对机器边界的选择。

1.5 为何WSC可能跟你有关系

一个有40个服务器，每个含有四个8核双线程CPU的机柜，会有超过2000个硬件线程。作者预言在几年后这样WSC系统将会有很多组织用得起。
或许更重要的是，基础设备服务(IaaS,Infrastructure-as-a-Service) 云计算所涌现的热度保证了WSC能够让每一个人交钱就能用。作者相信他们建造这些系统的经验能够有助于理解在这样的下一代平台上的设计问题和编程挑战。

1.6 WSC的架构总览

WSC一代代的硬件实现可能很不一样。但是这些系统的架构组织上相对稳定，由于比较高层的总体架构提供了后续讨论的框架，至少作者认为还是比较有用的。

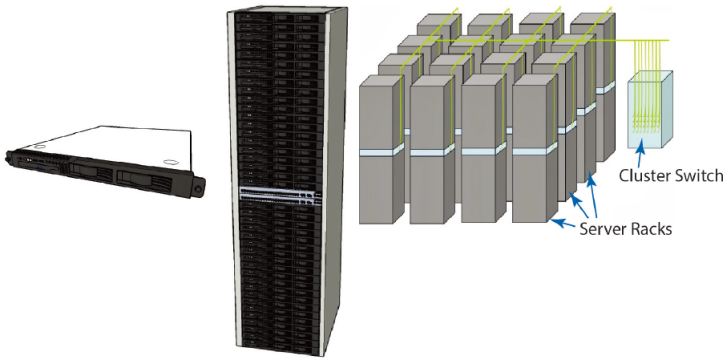


Figure 1.1: Sketch of the typical elements in warehouse-scale systems: 1U server (left), 7' rack with Ethernet switch (middle), and diagram of a small cluster with a cluster-level Ethernet switch/ router (right).

一组底端（low-end）服务器，会挂载在一个机柜中，并且用本地以太网交换机互联。
这些机柜级交换机，可以使用几或十几Gbps的连接，并且向上连接到一个或者多个聚簇级（数据中心级）以太网交换机。第二级交换可以连接上万级别的服务器。
多个处理用薄片可能用PCIe等I/O总线连接几个联网用薄片，作为包装。
一个薄片可能只有IU(44.45mm)，一个典型的机柜有42U高。
下图是Google WSC的一排服务器：

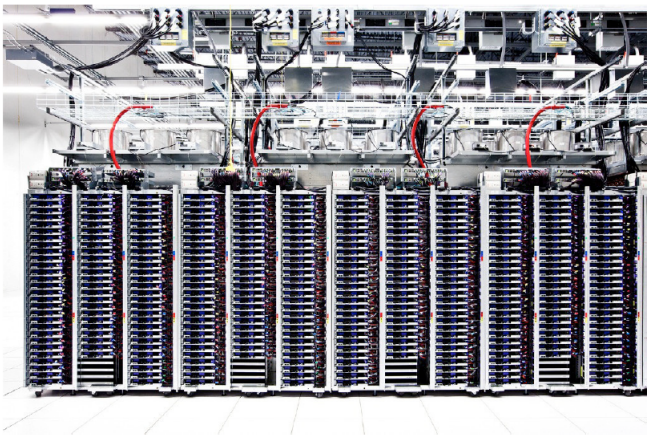


Figure 1.2: Picture of a row of servers in a Google WSC, 2012.

1.6.1 存储

磁盘驱动器和闪存设备会被连接到每个独立的服务器上并接受全局分布式文件系统的管理（如Google的GFS，Google File System），或者成为直接连接在聚簇级交换线缆上的网络连接存储设备（NAS,Network Attached Storage）。
两种实现方法间关于重复的模型有着根本性的不同。NAS通过重复和错误矫正来提供高可用性；而GFS等系统会用更多的带宽来完成写操作，并且由于有多个副本即使失去了一整个机柜都也保持数据可用。权衡更高的写消耗和低成本、高可用性和读取带宽是许多Google的早期负载的正确选择。
NAND Flash技术使得SSD(Solid State Drive)能够被用得起。而SSD提供了比硬盘高数倍的IO速率。

1.6.2 网络线缆

连接接口和网线是需要花钱的，和机器本身一样是成本的一部分。
我们可以选择增加更多的服务器和存储器来处理更多的副本，也可以选择增加更多的连接来提高带宽。这一点没有固定答案，不同的数据中心运营方可以各自权衡。
暂时地，我们将假设机柜内连接比机柜间连接便宜。

1.6.3 存储层次

下图显示了一个假想的WSC存储层次给程序员的视图。从Cache到RAM，到本地磁盘/闪存，到同机柜的内存和磁盘/闪存，到数据中心中其他的内存和磁盘/闪存，每一层都要慢一个数量级左右，延迟也大量增大。

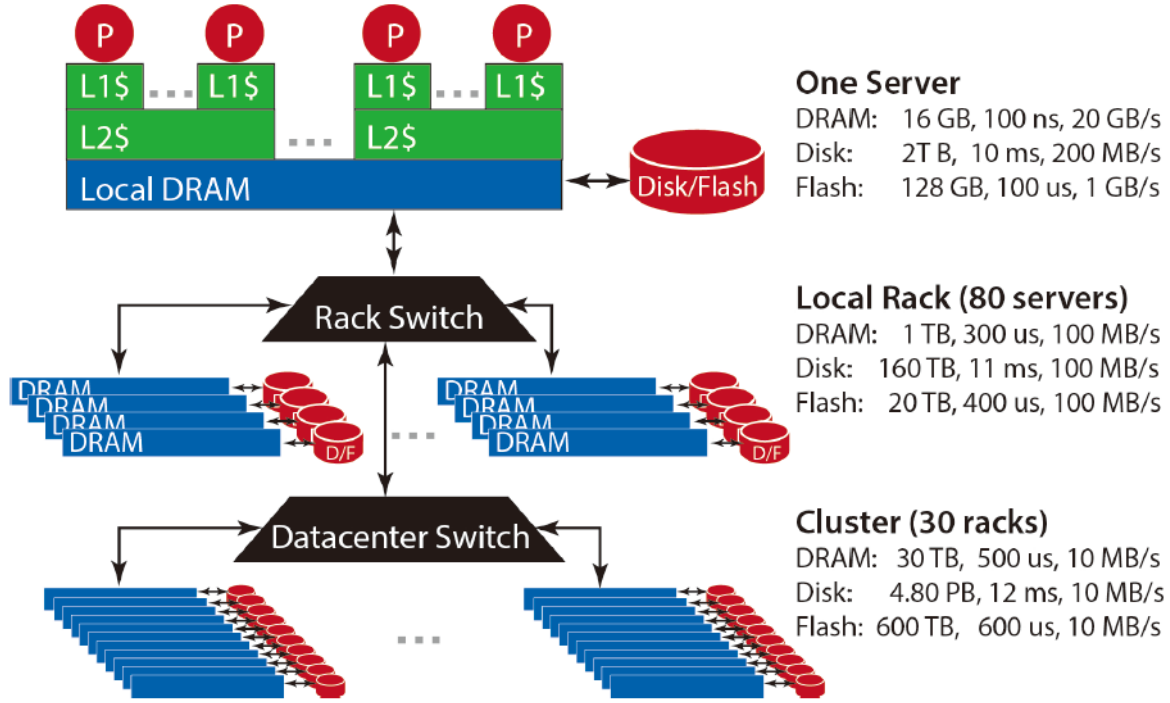


Figure 1.3: Storage hierarchy of a WSC.

只有在本地，磁盘才比闪存吞吐量大（单个任务）。经过交换机以后，由于网络连接速度受限，会引入新的吞吐量瓶颈。

1.6.4 量化延迟带宽和容量

此书把延迟、带宽和存储容量画在了一张图中，如下：

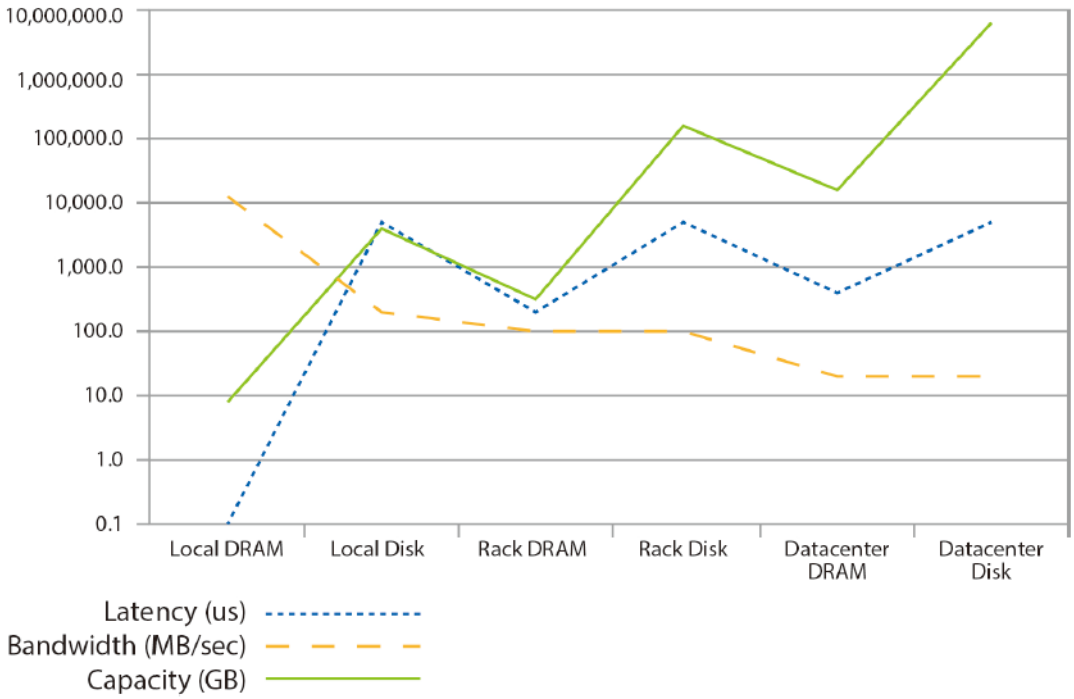


Figure 1.4: Latency, bandwidth, and capacity of a WSC.

延迟和容量几乎总是同涨同落的一套需要权衡的因素。而带宽总是层次上离处理器越近越高。
需要经过机柜交换机或者数据中心交换机的传输延迟收TCP/IP协议的延迟和带宽影响。
磁盘以典型的SATA接口的商业磁盘驱动器的延迟和传输速率来代表。
如果是大一些的应用，一个机柜无法容纳，那么程序员必须处理吞吐量限制带来的矛盾，而这将给编程带来很大的困难。

在WSC架构师的一个关键挑战就是去用一种成本效率较高的方式抹平这些这些矛盾。反过来，软件架构师的一个关键挑战是去建设机群的内部结构和服务来隐藏给应用开发者的大部分复杂性。
书中还就NAND Flash的性能和成本与DRAM和磁盘进行了对比（延迟和操作次数每秒两项都假设是一次4KB的数据传输），如下图：

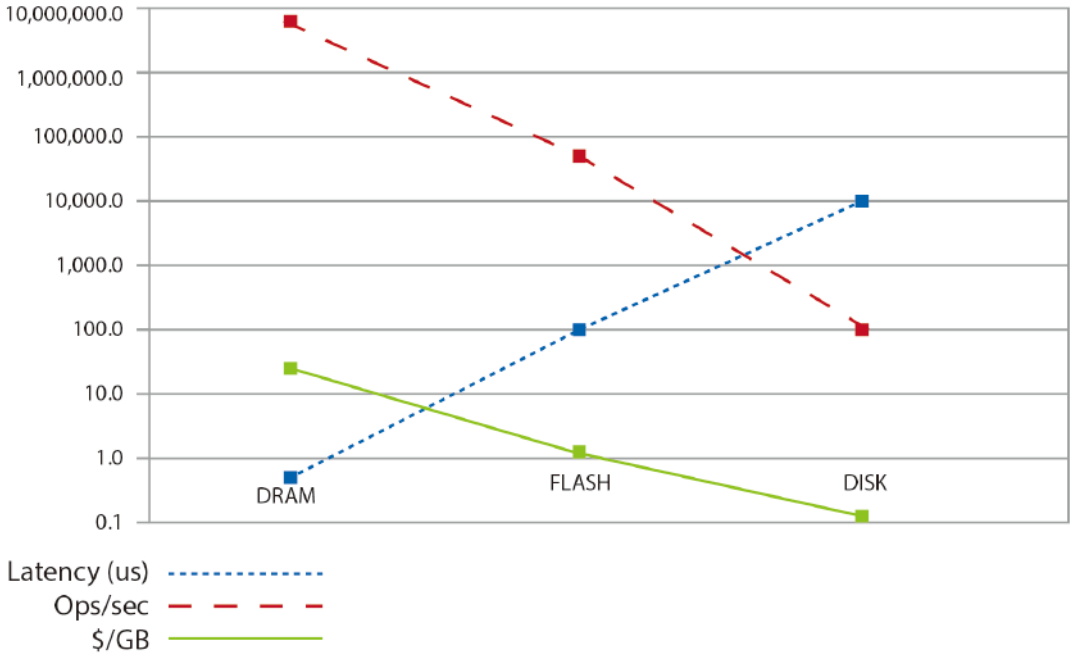


Figure 1.5: Performance and cost of NAND Flash with respect to DRAM and Disks (latency and Ops/sec assume a 4 KB data transfer).

NAND Flash在成本、延迟和吞吐量上几乎都是DRAM和磁盘的几何平均（三个点如图的对数坐标中几乎是一条直线）。
既然闪存会需求比WSC更高得多的带宽，闪存的性能已经高到把它有效地运用到分布式存储系统中能成为一个挑战。书中，后面将会在第3章讨论闪存的潜力和挑战。
暂时，要注意在最糟糕的情况下闪存的写能比读慢上若干个量级。

1.6.5 功率用途

关于能量和功率的使用，书中第5章将会讨论更多细节。下图通过把峰值功率的用途分解开，提供了一些关于能量在如何被现代IT设备使用的直觉。

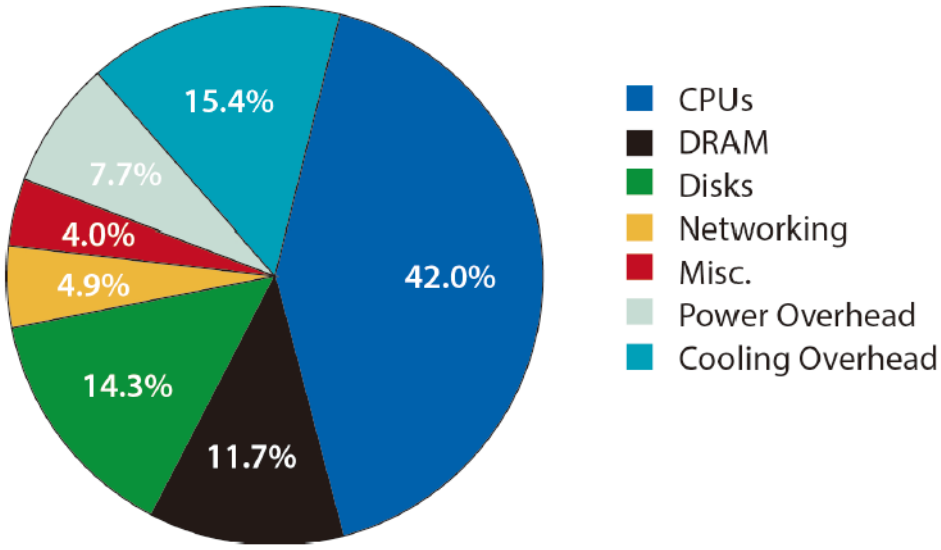


Figure 1.6: Approximate distribution of peak power usage by hardware subsystem in a modern data-center using late 2012 generation servers and a facility energy overhead of 30%. The figure assumes two-socket x86 servers, 16 DIMMs and 8 disk drives per server, and an average utilization of 80%.

这张图的数据来自Google 2007年的数据中心。作者提到此书上一版的时候存储系统和内存几乎是相当的，而变成现在的这个样子是很多原因的综合结果。

1. 成熟的温度管理系统能够让CPU运行在接近最大封装功率附近，导致每个CPU插槽有比较大的功耗。
2. 存储技术从功率饥渴的FBDIMM（Fully Buffered DIMM，Fully Buffered Dual-Inline-Memory-Modules）变迁到了有更好的功率管理系统的DDR3。
3. DRAM的电压从1.8V降低到了1.5V。
4. 现在的系统中CPU性能对DRAM每GB容量的比值变高了。

1.6.6 把握失败

WSC需要互联网服务能够容忍相对较高的组件出错率。例如，磁盘驱动能够把年化失败率提到比4%还高。不同的布置可能会有平均每年1.2次到每年16次的服务器级重启。应用要能对此做出正确反应。这个主题会在第2章（关于应用）和第7章（关于错误统计）展开讨论。

2 硬件建造块

WSC会很大程度上被建造块的选择所决定。拿什么东西搭，很大程度上决定了最后是什么东西。

2.1 成本高效的服务器硬件

写书的时候（不过到现在也没什么变化），低端服务器的机群是WSC系统偏好的建造块。
一般来说，由于价格波动和性能受测试程序（benchmark）的特性以及在测试程序中所做的努力的影响，很难有意义地衡量成本效率的对比。
作者认为比较有意思的讨论是关于低端服务器结点和极低端（Wimpy）服务器结点的，这章之后会讲。

2.1.1 大规模对称多处理器通讯效率的冲击

简单的面向处理器的成本效率分析并不会算上大规模对称多处理器（SMP）比起以商用线缆连接的低端服务器机群戏剧性地高的互联性能的好处。对称多处理器中的结点可以用100ns量级的延迟通讯，然而通常布置在服务器集群中的基于LAN（一般的网线接口）的网络会有超过100μs的延迟。
假设固定的本地计算时间是1ms，程序执行时间的关系式如下：

$$Excutiontime = 1ms + f * [100ns/\#nodes + 100\mu s * (1 - 1/\#nodes)]$$

其中，变量f表示每1ms工作单元中的全局访问次数。
这些全局访问有$1/\#nodes$的概率是SMP的内部访问，需要按DRAM的速度计算（100ns左右）；有$1 - 1/\#nodes$的概率要访问SMP的外部，需要按LAN的速度计算（100μs）。

作者表示，下图中的曲线有两个值得划重点的有趣的地方。
一者，在少量通讯的情形下，使用多个结点的集群导致的性能下降较少。而在中、大量通讯的情形下，这个惩罚会很重，但是最为剧烈的地方是在把一个节点一分为二那里。
在这个模型中，一个128处理器的SMP能比32个4处理器SMP的集群性能高出10倍以上。

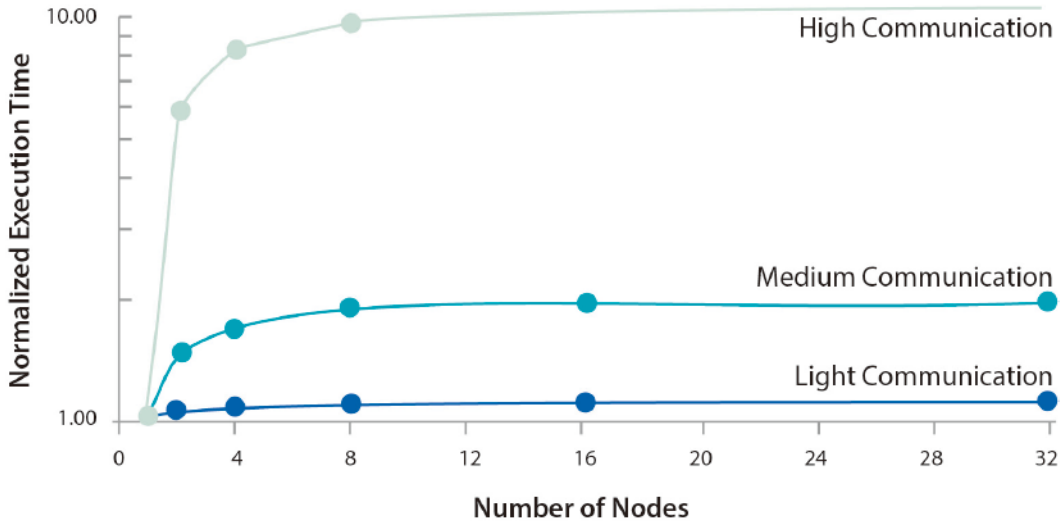


Figure 3.1: Execution time of parallel tasks as the number of SMP nodes increases for three levels of communication intensity. Execution time is normalized to the single node case and plotted in logarithmic scale.

根据定义，WSC系统会包含上千个处理器结点。作者比较了同样处理器个数下，集成128处理器SMP的性能和集成4处理器SMP的性能。

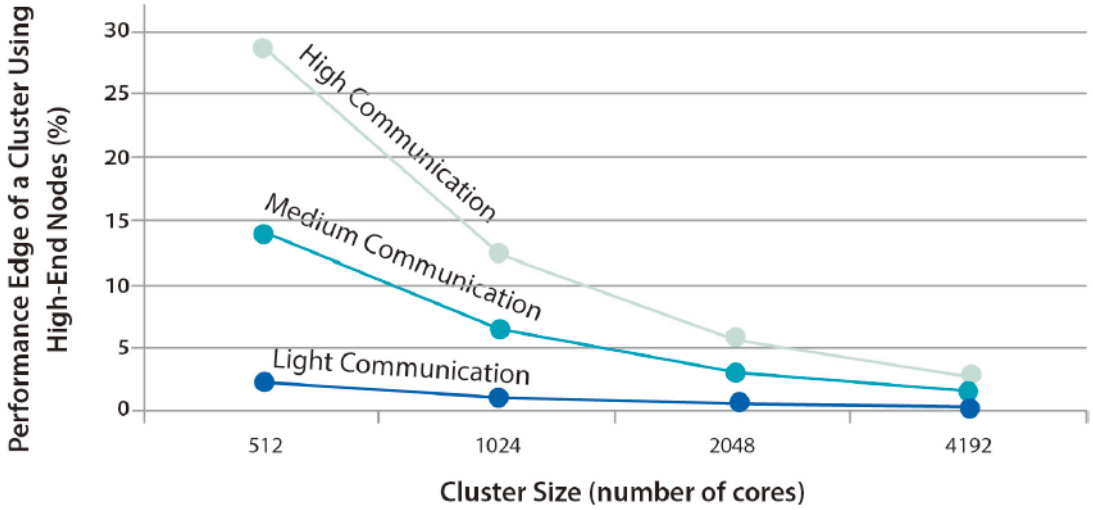


Figure 3.2: Performance advantage of a cluster built with large SMP server nodes (128-core SMP) over a cluster with the same number of processor cores built with low-end server nodes (four-core SMP), for clusters of varying size.

在结点个数较少的时候，128处理器SMP能够带来一些性能优势。但是随着处理器总数的增加，128处理器SMP能够带来的性能优势逐渐趋于0。而有着这样小的性能差距，多端处理器（高4到20倍）的高昂价格会使它成为非常没有吸引力的选项。
性能增强对于在一个结点内部的计算还是很重要的。不过一旦引入沉重的附加开销，高端服务器结点的成本效率在WSC中就没有什么竞争力了，因为它们是为小规模计算机准备的。

2.1.2 健壮的相比于无用的服务器

使用更小更慢的CPU的好处和使用中等规模通讯服务器来代替高端SMP的讨论很类似:

1. 同样数量的吞吐率下，更小的CPU能够便宜2-5倍。
2. 许多应用是内存或者IO受限的，结果更快的CPU并不能很好地适应更大的应用，而且还会进一步提高小CPU的价格优势。
3. 更慢的CPU更有能量效率；典型地，当CPU频率降低k倍的时候，CPU功率会按照$O(K_2)$降低。

然而使用非常低性能的核对WSC来说可能也是不具吸引力的，下面是书中总结的一些讨论。

1. 许多互联网应用都收到请求级和数据级并行的好处，这样的系统并不会免疫Amdahl定律。随着线程数增加，越来越难以降低串行和通讯，从而限制了加速。同样大量线程处理并行任务，响应时间的不确定性会变得影响很大。由于经常在请求完成之前要完成所有的并行任务，而这些任务中最长的响应时间会成为总体的响应时间，由此影响最终的服务延迟。
2. 网络系统的要求会随着较小的系统的数量的增加而增加，所以结点数不能太大。
3. 小服务器也会导致实用性降低。把任务分给服务器需要占用相应的空间，如果每个服务器的容量太小以至于装不小第二个应用（CPU、RAM资源受限）就会导致分配很困难。
4. 计算算法的并行度大得出奇，有时候由于数据被分割得太小还是会导致内禀地低效。比如启发式算法能看到的局部非常小，那么这个局部可能很难反映出整个全局的性质。

2.1.3 平衡性设计

应该注意三个重要的考虑:

1. 聪明的程序员能够把算法重新构造成更好地匹配较便宜的设计方案。这是一个软硬件共同设计的机会，但是注意不要搞出来编程过于复杂的机器。
2. 最有成本效率并且平衡的硬件配置可能是匹配多个工作负载所合成的资源需求而不是完美地匹配其中任何一个。
3. 可替代的资源会倾向于能被更有效地使用。

正确的设计点还取决于工作负载本身，因为数据尺寸和服务热度也起至关重要的作用。例如，一个数据尺寸很大又比较冷门的服务可能放到相对编译但是吞吐量低的磁盘中就可以；而非常高热度的服务如果数据规模足够小或者有明显的局部性能够使用在内存中的服务来获得好处。此外，在设计中可能还有必要考虑到设备需要更新换代等问题。

2.2 WSC存储

WSC工作负载所操作的数据大致分为两类：运行任务的用户的私有数据和和成为了共享分布式工作负载共享状态的一部分的数据。私有数据主要在本地并且管理比较简单，而共享数据就需要更加持久并且需要更加成熟的分布式存储系统。下面讨论WSC存储的主要特点。

2.2.1 无结构WSC存储

GFS把文件存储了多份用于恢复，并把分布式的存储抽象成了统一的文件系统便于管理。
早期会直接存储副本，但是后来为了节约空间使用Reed Solom编码节约空间。
快速地恢复的重要性在于长的恢复时间窗口会让没有被复制的块容易遭受数据损失。

2.2.2 结构化WSC存储

GFS和Colossus的简单文件抽象可能对于操作二进制大对象数据的系统是充分的但是应用开发者也需要WSC等价于数据库的功能，其中数据集可以有结构并且为简单的小规模更新或者复杂查询编号。如Google的BigTable和Amazon的Dynamo这样的结构化分布式存储系统被设计来满足这样的需求。
但是Google的一些应用开发者发现处理弱一致性模型以及BigTable的简单数据编排的局限性很不方便。如MegaStore和后来的Spanner等第二代结构化存储系统被开发出来来处理这些问题。MegaStore和Spanner就牺牲了一些效率来提供更简单的编程界面。
此外，还有Memcached、RAMCloud以及FAWN-KV等系统，提供了一些其他的特性。

2.2.3 存储和网络技术的相互作用

WSC的分布式存储系统的成功部分归功于数据中心网络结构的发展。Ananthanarayanan et al. 观察到网络和磁盘性能的间隙已经宽到磁盘局部性已经不再和数据中心内的计算相关了。这一观察使得基于磁盘的分布式存储统被极大地简化了。
Flash的出现给数据中心网络结构又带来了新的挑战。Flash的局部性仍然有影响。

2.3 WSC网络系统

如果每个服务器都有个能跟任意一个服务器交流，那么当我们希望提高带宽到2倍的时候，需要提升的是对分带宽而不仅仅是叶节点的对分带宽。
然而受限于交换机，加倍对分带宽是很困难的。单个交换机的性能已经找不到更高的了。
我们可以通过级联这些交换机芯片来构建更大的交换机，典型地弄成fat tree的形式或者CLOS（由Charles Clos提出）网络的形式，如下图：

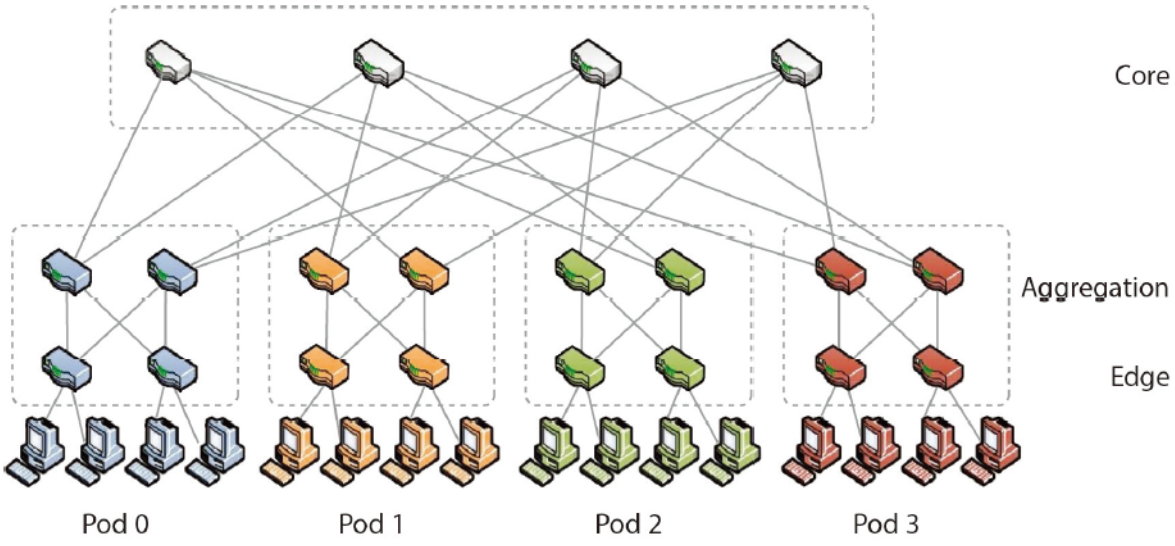
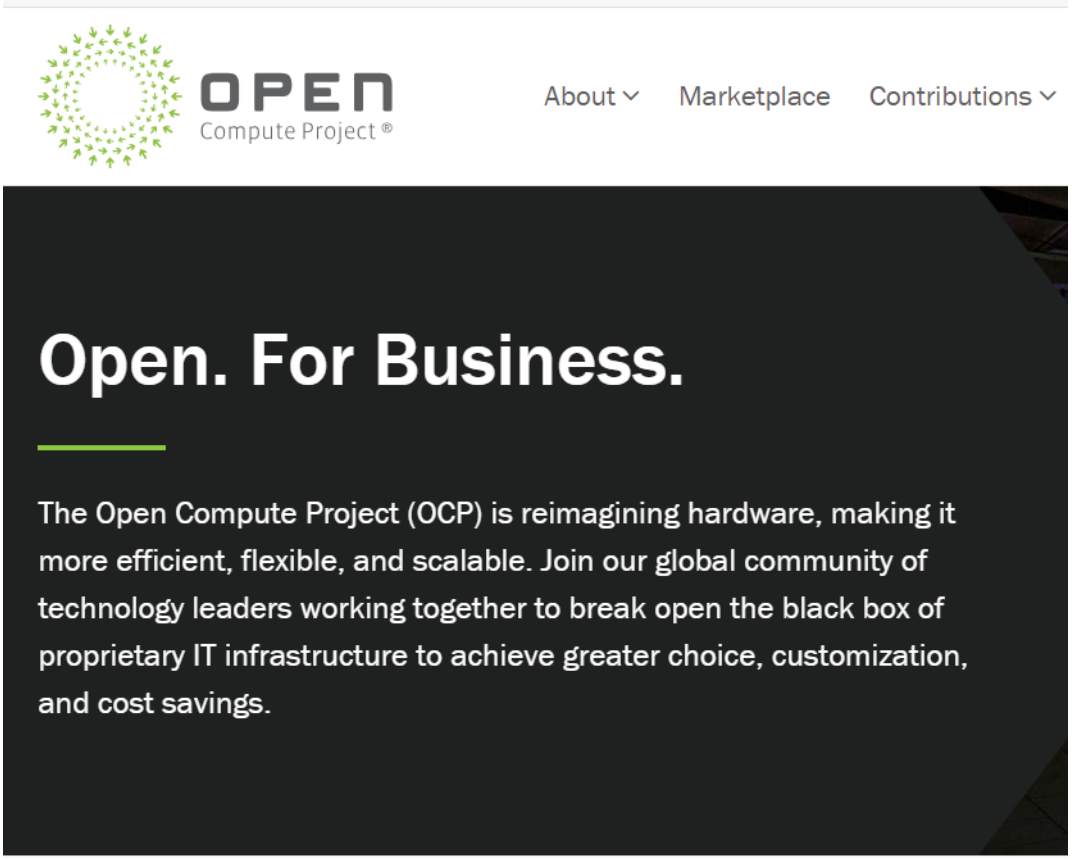


Figure 3.3: Sample three-stage fat tree topology. With appropriate scheduling, this tree can deliver the same throughput as a single-stage crossbar switch. (Image courtesy of Amin Vahdat.)

这样一个使用k端口的交换机的树能够使用$5k^2/4$个交换机支持$k^3/4$个服务器的满速吞吐，从而允许有上万个端口的网络。然而，注意到，这样做成本会显著地增加。增加端口会增加成本，如果连接线变长也会增加成本。一个100米的10Gbps的连接的光学组件能够轻易地消耗数百美元。
为了节约成本，WSC的实现者一般通过超额机柜顶交换机的网络来节约成本。一个机柜内部的服务器数量足够少，使用一个交换机互联就可以达到满速。对于一个完整的fat tree，每个交换机需要每一个服务器都有一个上行端口，但是我们可以降低一部分带宽来节约成本，比如每2个服务器才有一个上行端口等。
另外一个方式是一些特殊用途的网络可以进行特殊连接。存储器区域网络（SANDISK）连接服务器和磁盘，如FibreChannel等。

2.4 进一步阅读

书中推荐Hennessy和Patterson的计算机体系结构教科书第五版中（《Computer architecture. A quantitative approach. 5th》）有关于WSC的新材料。开放计算项目（Open Compute Project）(http://opencompute.org) 有许多WSC的硬件组件的详细标准



3 附录A-单词翻译对照表

英文	中文
ARPANET,Advanced Research Project Agency Network	阿帕网络
milestone	里程碑
ubiquity	普遍性
vendor	供应商
deployment	调度，部署
reign	统治，盛行
dedicated	专用的，献身的
dedicate	奉献
facility	设备，设施
graceful	优雅的
retrieval	检索，恢复
synonym	同义词，同一性
relentless	无情的，不间断的
relent	减弱，缓和
provisioning	供应（动名词）
provision	n.规章;vt.供应
miscellaneous	混杂的，多方面的
co-located	共置的
rack	齿条，架子，机柜
niche	壁龛
sheer	绝对的，透明的
render	致使，提出，给予
hypothetical	假设的，假想的
discrepancy	矛盾
architect	设计师
conversely	反过来，相反的
insight	洞察，直觉
networking	n.网络化，网络系统
brawny	顽强的，健壮的，严重肿胀的
wimpy	懦弱的，无用的，哭哭啼啼的
obsolete	废弃的，老式的
low-end	低端的，低档的
premium	保险费，额外费用；优质的，高昂的
qualitative	定性的，性质上的，质量的
fungible	可替代的
crawl	爬
blob,binary large object	二进制大对象
interplay	相互作用
relevant	有关的，中肯的，实质性的
oversubscribe	超额认购，超额
whereas	然而，鉴于，反之