Warehouse-Scale Computers

邵恩
高性能计算机研究中心

9/5/2024

Outline

- 仓库规模计算(Warehouse-Scale Computing)和数据并行
  - Warehouse-scale computers (WSCs)
  - 请求和数据级并行
  - Scale Up :扩展节点上的计算资源
  - Scale Out:通过WSC的调度程序架构扩展

Warehouse-scale computers (WSCs)

- 用途广泛：提供互联网服务(services)
  - Search, social networking, online maps, video sharing, online shopping, email, cloud computing, etc.
- 区别1：与高性能计算（HPC）"Clusters"的区别
  - 具有更高性能的处理器和网络
  - 强调线程级并行(thread-level parallelism)，WSC强调请求级并行 (request-level parallelism)
- 区别2：与数据中心的区别
  - 数据中心强调虚拟机(virtual machines)和硬件异构性，以便为不同的客户提供服务

WSC 的特征

- 功耗压力：计入运营成本
  - 电力消耗是设计系统时的主要约束，而不是次要约束
- 可扩展：规模及其机会和问题
  - 由于WSC需要大量购买设备，因此可以对系统进行定制
- 利用率低：
  - Computing efficiently at low utilization

WSC的效率和成本

- 地理分布：Location of WSC
  - 近互联网骨干网，电力成本，房产税率，地震，洪水和飓风的风险低

Amazon Sites

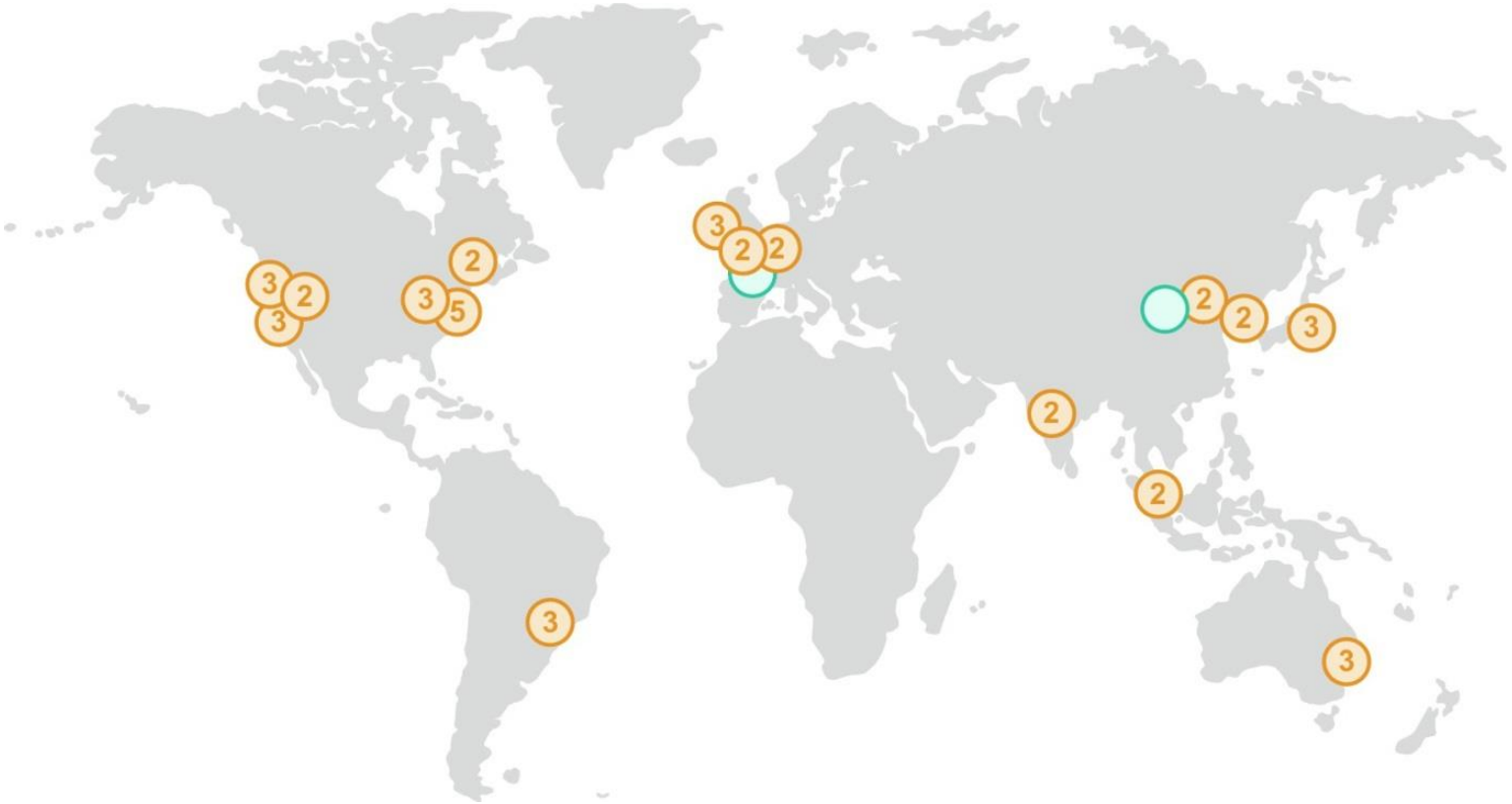


Figure 6.18 In 2017 AWS had 16 sites ("regions"), with two more opening soon. Most sites have two to three *availability zones*, which are located nearby but are unlikely to be affected by the same natural disaster or power outage, if one were to occur. (The number of availability zones are listed inside each circle on the map.) These 16 sites or regions collectively have 42 availability zones. Each availability zone has one or more WSCs. https://aws.amazon.com/about-aws/global-infrastructure/.

Google Sites

Figure 6.19 In 2017 Google had 15 sites. In the Americas: Berkeley County, South Carolina; Council Bluffs, Iowa; Douglas County, Georgia; Jackson County, Alabama; Lenoir, North Carolina; Mayes County, Oklahoma; Montgomery County, Tennessee; Quilicura, Chile; and The Dalles, Oregon. In Asia: Changhua County, Taiwan; Singapore. In Europe: Dublin, Ireland; Eemshaven, Netherlands; Hamina, Finland; St. Ghislain, Belgium. https://www.google.com/about/datacenters/inside/locations/.

Microsoft Sites

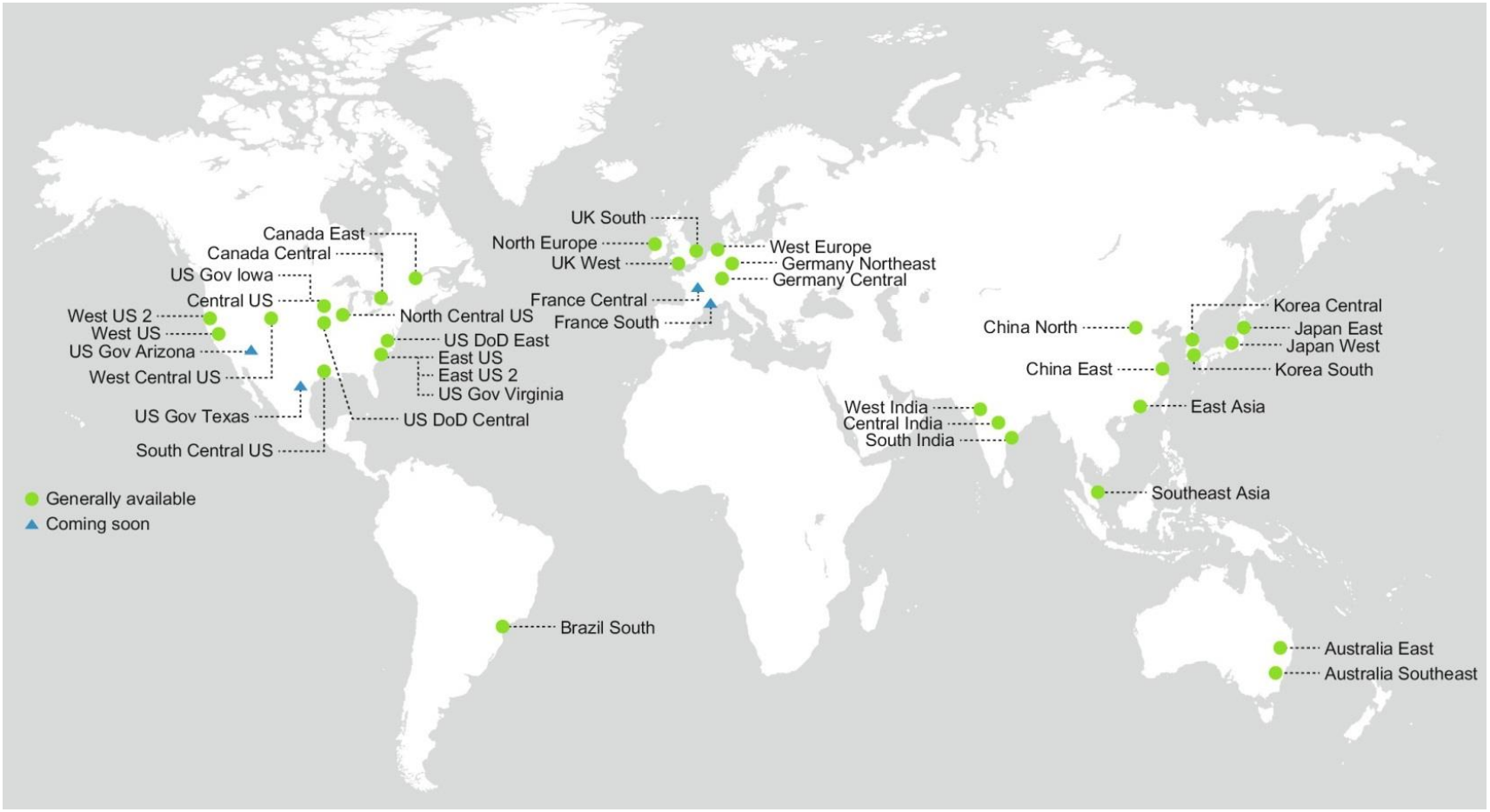


Figure 6.20 In 2017 Microsoft had 34 sites, with four more opening soon. https://azure.microsoft.com/en-us/regions/.

电力损耗

High Voltage Utility Distribution

115kv

Sub-station

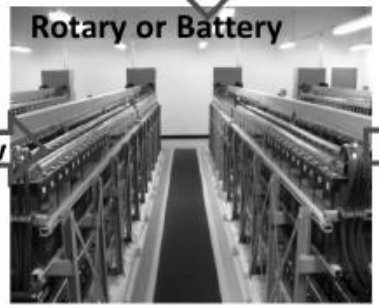
0.3% loss
99.7% efficient

13.2kv

Generators

13.2kv

UPS:
Rotary or Battery

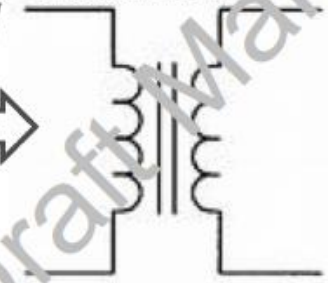


6% loss
94% efficient, ~97% available

13.2kv

UPS & Gen often on 480v

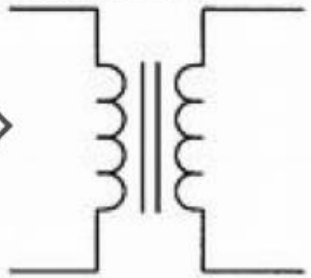
Transformers



2% loss
98% efficient

480V

Transformers



2% loss
98% efficient

208V

~1% loss in switch gear & conductors

IT Load (servers, storage, Net, …)

11% lost in distribution
.997*.94*.98*.98*.99 = 89%

制冷

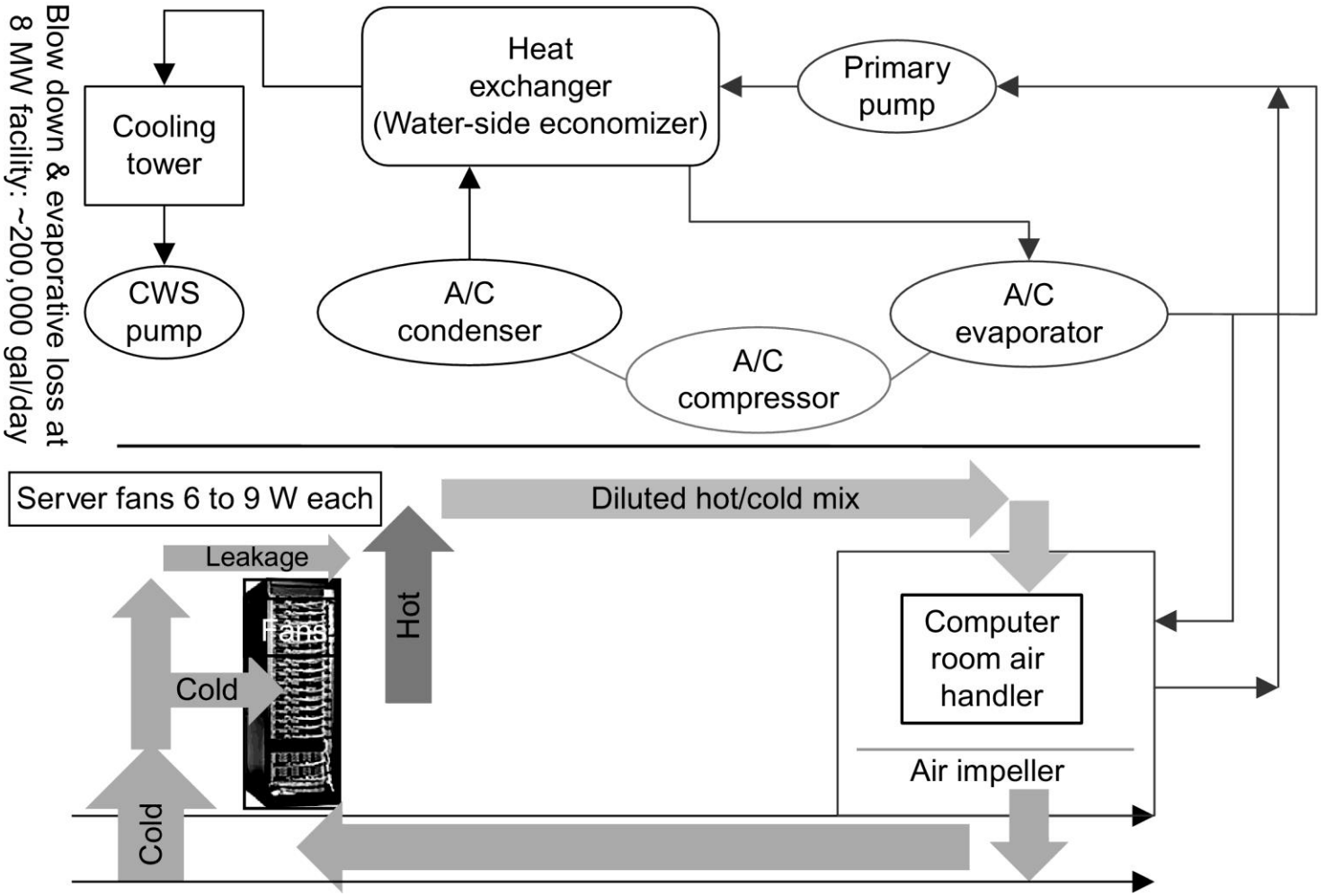


Figure 6.9 Mechanical design for cooling systems. CWS stands for circulating water system. From Hamilton, J., 2010. Cloud computing economies of scale. In: Paper Presented at the AWS Workshop on Genomics and Cloud Computing, June 8, 2010, Seattle, WA. http://mvdirona.com/jrh/TalksAndPapers/JamesHamilton_GenomicsCloud20100608.pdf.

WSC的基础设施和成本

- 冷却系统也使用水（蒸发和溢出）
  - 例如，每天70,000至200,000加仑，用于8 MW设施
- 电力成本分解：
  - 冷水机：IT设备使用电力的30-50%
  - 空调：IT电力的10-20%，主要是风扇
- 服务上限：WSC能支持多少台服务器？
  - Each server:
    - "铭牌功率额定值"给出最大功耗
    - 要获得实际值，需要在实际工作负载下测量功率

WSC的基础设施和成本

- 服务容量上限：确定最大服务器容量
- 部件功耗占比：
  - 处理器：42%
  - DRAM：12%
  - 磁盘：14%
  - 网络：5%
  - 冷却：15%
  - 电力开销：8%
  - 杂项：4%

用于非计算行为

电力利用效率Power Utilization Effectiveness (PEU)

- 电力利用效率（PEU） = 设施总电力/IT设备电力（越低越好）

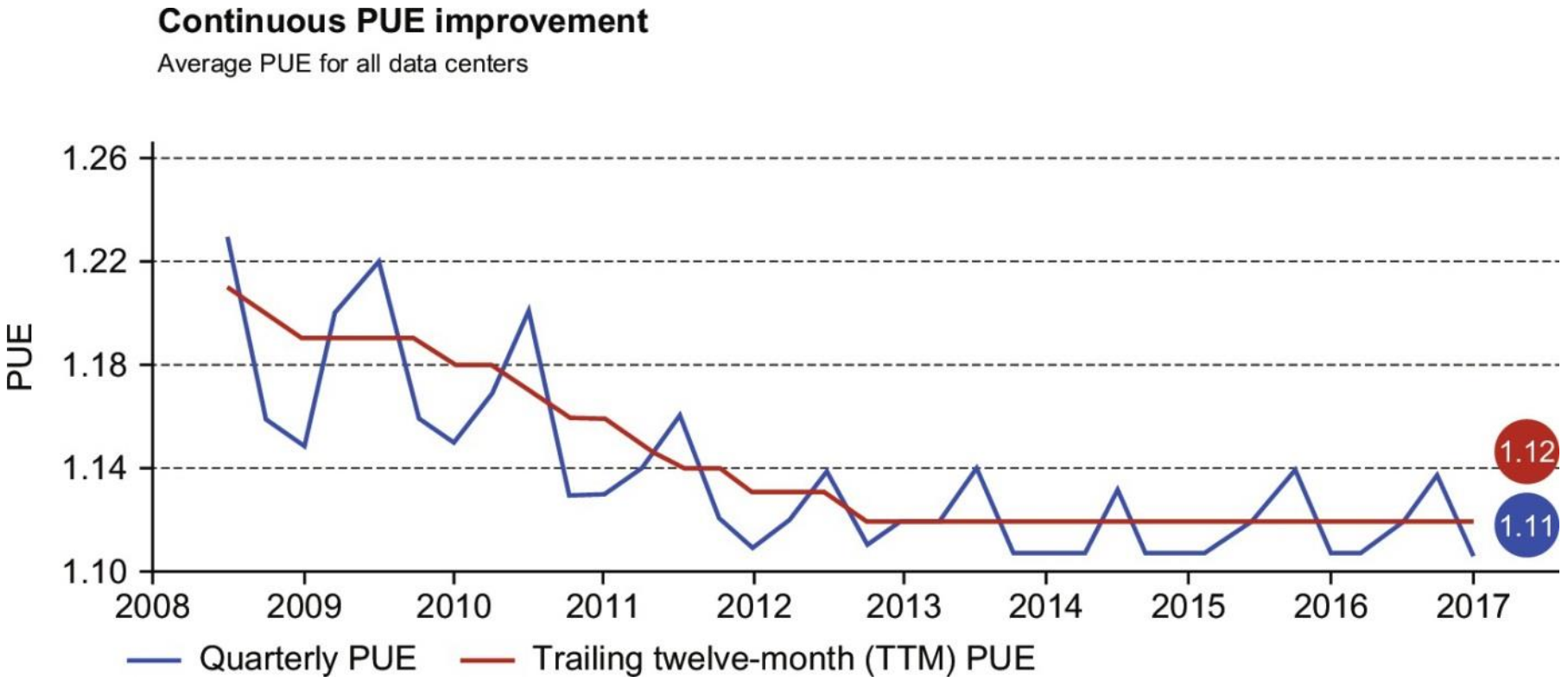


Figure 6.11 2008年至2017年间15个Google WSC的平均电力利用效率（PUE）。峰值线是季度平均PUE，较直线是过去12个月的平均PUE。对于2016年第4季度，平均值分别为1.11和1.12。

性能,延迟

- 延迟敏感：延迟是重要的指标，因为用户可见
- MS Bing研究：随着响应时间的增加，用户将减少搜索使用的次数
- 服务级目标（SLO）/服务级协议（SLA）
  - 例如，99%的请求低于100毫秒

Server delay (ms)	Increased time to next click (ms)	Queries/ user	Any clicks/ user	User satisfaction	Revenue/ user
50	–	–	–	–	–
200	500	–	–0.3%	–0.4%	–
500	1200	–	–1.0%	–0.9%	–1.2%
1000	1900	–0.7%	–1.9%	–1.6%	–2.8%
2000	3100	–1.8%	–4.4%	–3.8%	–4.3%

异常情况

出现次数（1年内）

Approx. number events in 1st year	Cause	Consequence
1 or 2	Power utility failures	Lose power to whole WSC; doesn't bring down WSC if UPS and generators work (generators work about 99% of time).
4	Cluster upgrades	Planned outage to upgrade infrastructure, many times for evolving networking needs such as recabling, to switch firmware upgrades, and so on. There are about nine planned cluster outages for every unplanned outage.
1000s	Hard-drive failures	2%–10% annual disk failure rate (Pinheiro et al., 2007)
	Slow disks	Still operate, but run 10× to 20× more slowly
	Bad memories	One uncorrectable DRAM error per year (Schroeder et al., 2009)
	Misconfigured machines	Configuration led to ~30% of service disruptions (Barroso and HÖlzle, 2009)
	Flaky machines	1% of servers reboot more than once a week (Barroso and HÖlzle, 2009)
5000	Individual server crashes	Machine reboot; typically takes about 5 min (caused by problems in software or hardware).

硬件故障

软件服务故障

Figure 6.1 List of outages and anomalies with the approximate frequencies of occurrences in the first year of a new cluster of 2400 servers. We label what Google calls a cluster an *array*; see Figure 6.5. Based on Barroso, L.A., 2010. Warehouse Scale Computing [keynote address]. In: Proceedings of ACM SIGMOD, June 8–10, 2010, Indianapolis, IN.

CPU资源利用率偏低

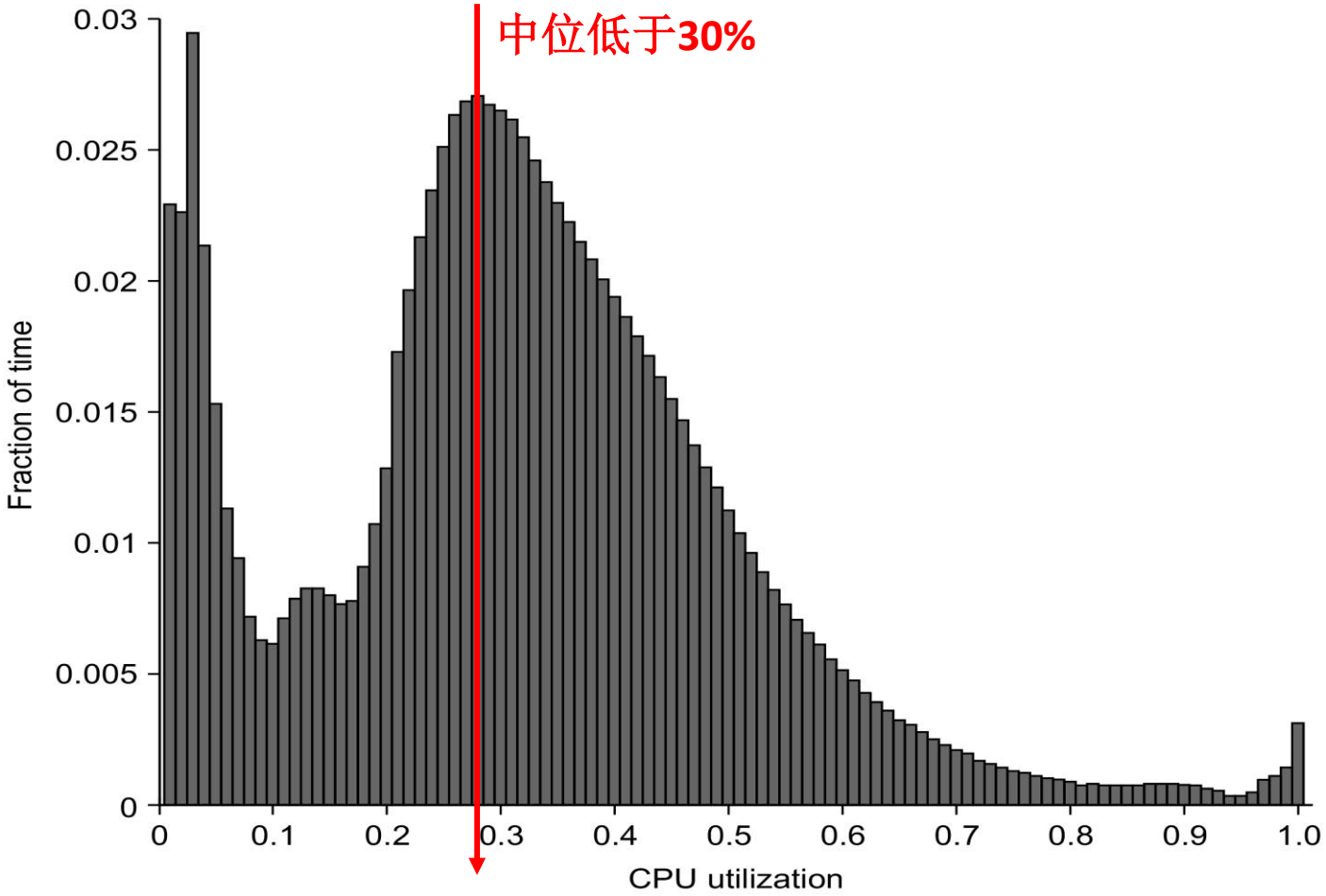


Figure 6.3 Average CPU utilization of more than 5000 servers during a 6-month period at Google. Servers are rarely completely idle or fully utilized, instead operating most of the time at between 10% and 50% of their maximum utilization. The third column from the right in Figure 6.4 calculates percentages plus or minus 5% to come up with the weightings; thus 1.2% for the 90% row means that 1.2% of servers were between 85% and 95% utilized.
From Figure 1 in Barroso, L.A., Hölzle, U., 2007. The case for energy-proportional computing. IEEE Comput. 40 (12), 33–37.

Google WSC Rack

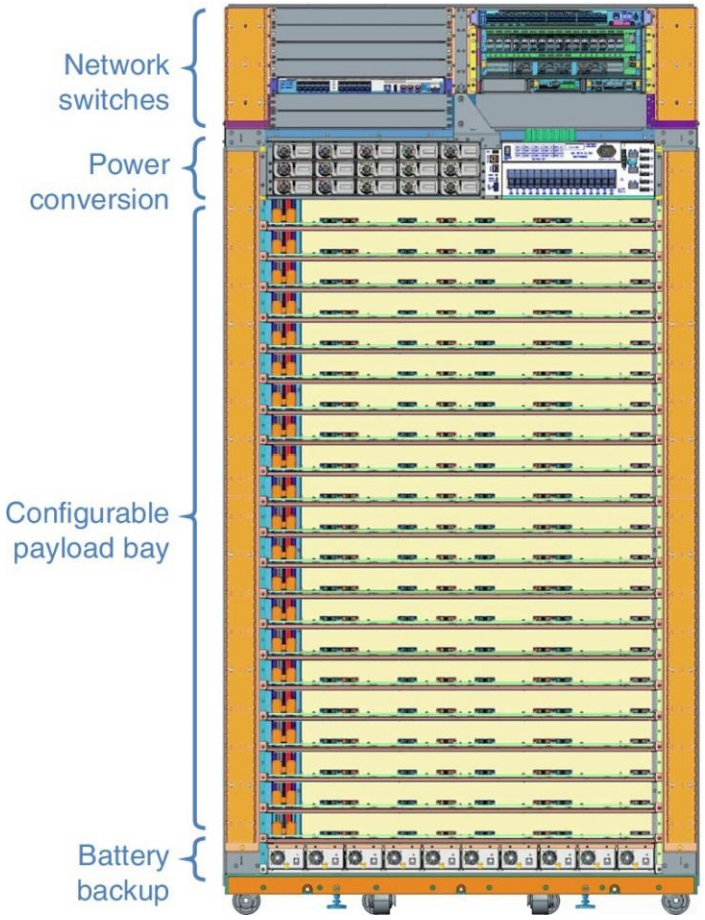


Figure 6.30 A Google rack for its WSC. Its dimensions are about 7 ft high, 4 ft wide, and 2 ft deep (2 m × 1.2 m × 0.5 m). The Top of Rack switches are indeed at the top of this rack. Next comes the power converter that converts from 240 V AC to 48 V DC for the servers in the rack using a bus bar at the back of the rack. Next is the 20 slots (depending on the height of the server) that can be configured for the various types of servers that can be placed in the rack. Up to four servers can be placed per tray. At the bottom of the rack are high-efficiency distributed modular DC uninterruptible power supply (UPS) batteries.

Rack阵列

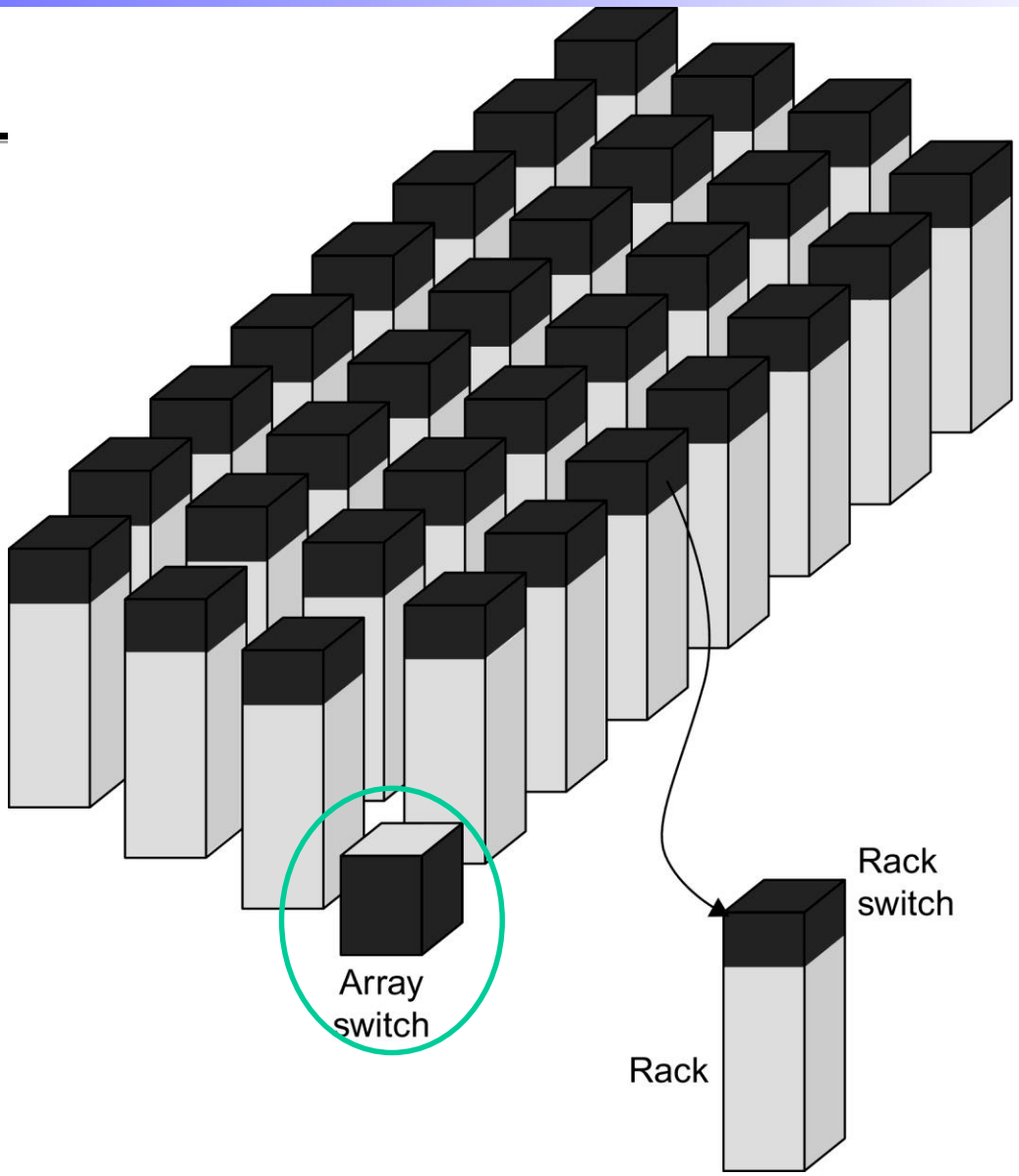


Figure 6.5 Hierarchy of switches in a WSC. Based on Figure 1.1 in Barroso, L.A., Clidaras, J., Hölzle, U., 2013. The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synth. Lect. Comput. Architect. 8 (3), 1–154.

更早期的WCS 网络结构

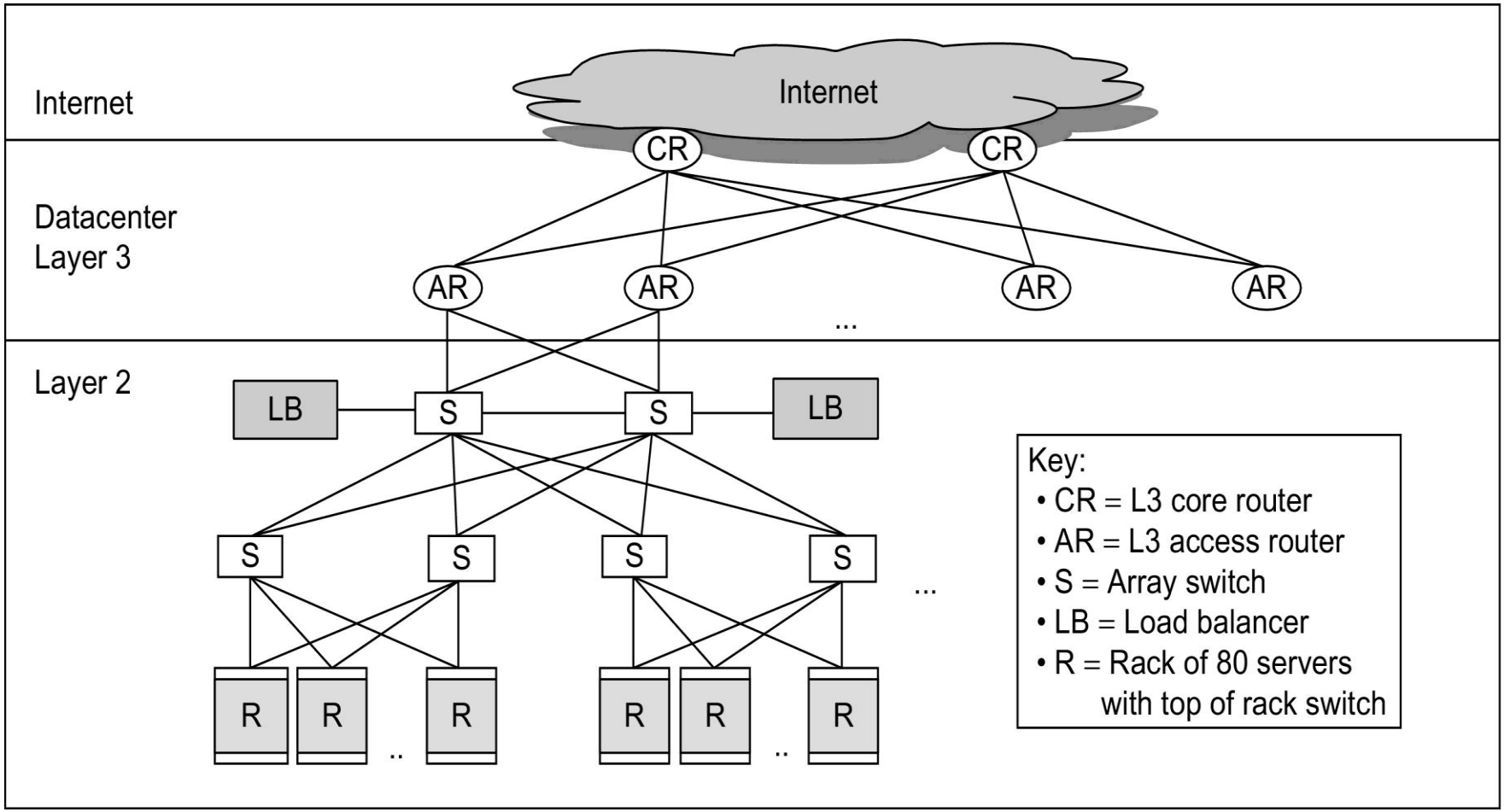


Figure 6.8 A Layer 3 network used to link arrays together and to the Internet (Greenberg et al., 2009). A load balancer monitors how busy a set of servers is and directs traffic to the less loaded ones to try to keep the servers approximately equally utilized. Another option is to use a separate *border router* to connect the Internet to the data center Layer 3 switches. As we will see in Section 6.6, many modern WSCs have abandoned the conventional layered networking stack of traditional switches.

Array Switch——扩展性瓶颈

- 连接机架阵列的交换机（Array switch）
  - Array switch 应具有 10 倍的机架交换机等分带宽（bisection bandwidth）
  - 交换机的成本随着端口数量n，以n^2的方式增长
  - 通常使用内容可寻址存储芯片和 FPGA

Clos 网络结构

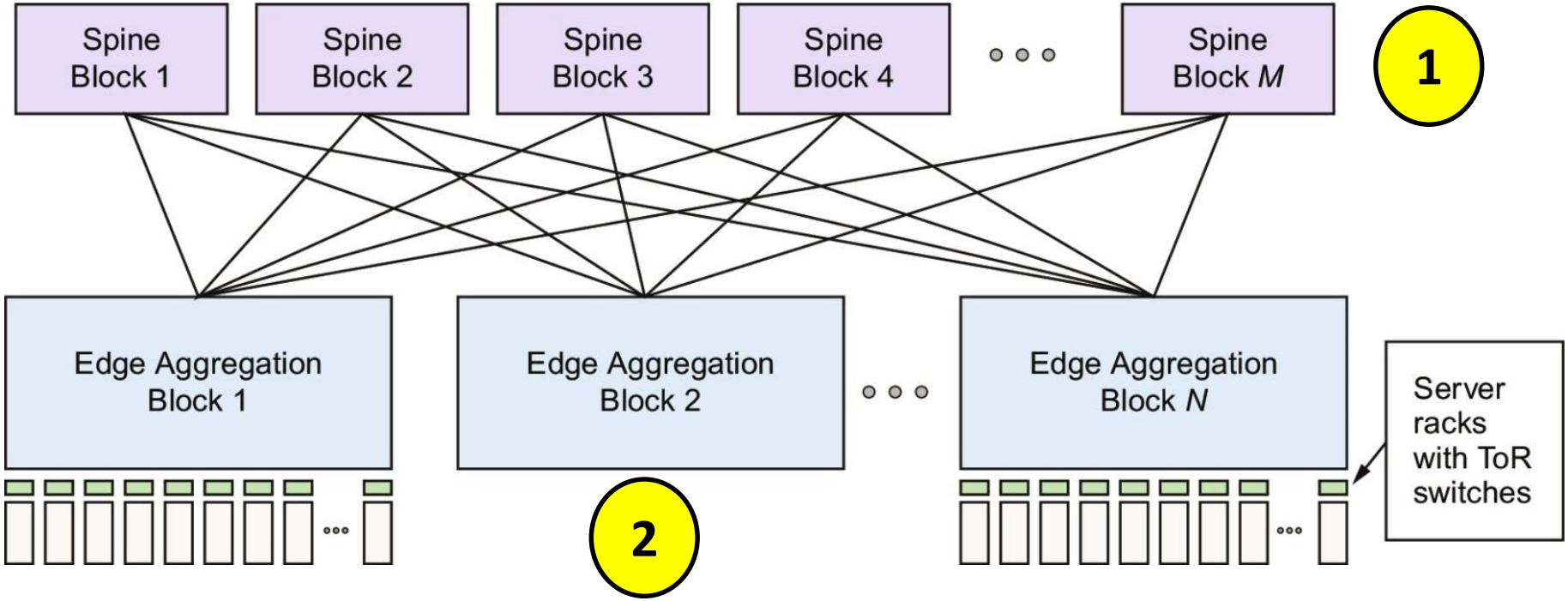


Figure 6.31 A Clos network has three logical stages containing crossbar switches: ingress, middle, and egress. Each input to the ingress stage can go through any of the middle stages to be routed to any output of the egress stage. In this figure, the middle stages are the *M* Spine Blocks, and the ingress and egress stages are in the *N* Edge Activation Blocks. Figure 6.22 shows the changes in the Spine Blocks and the Edge Aggregation Blocks over many generations of Clos networks in Google WSCs.

Google Jupiter Clos Network

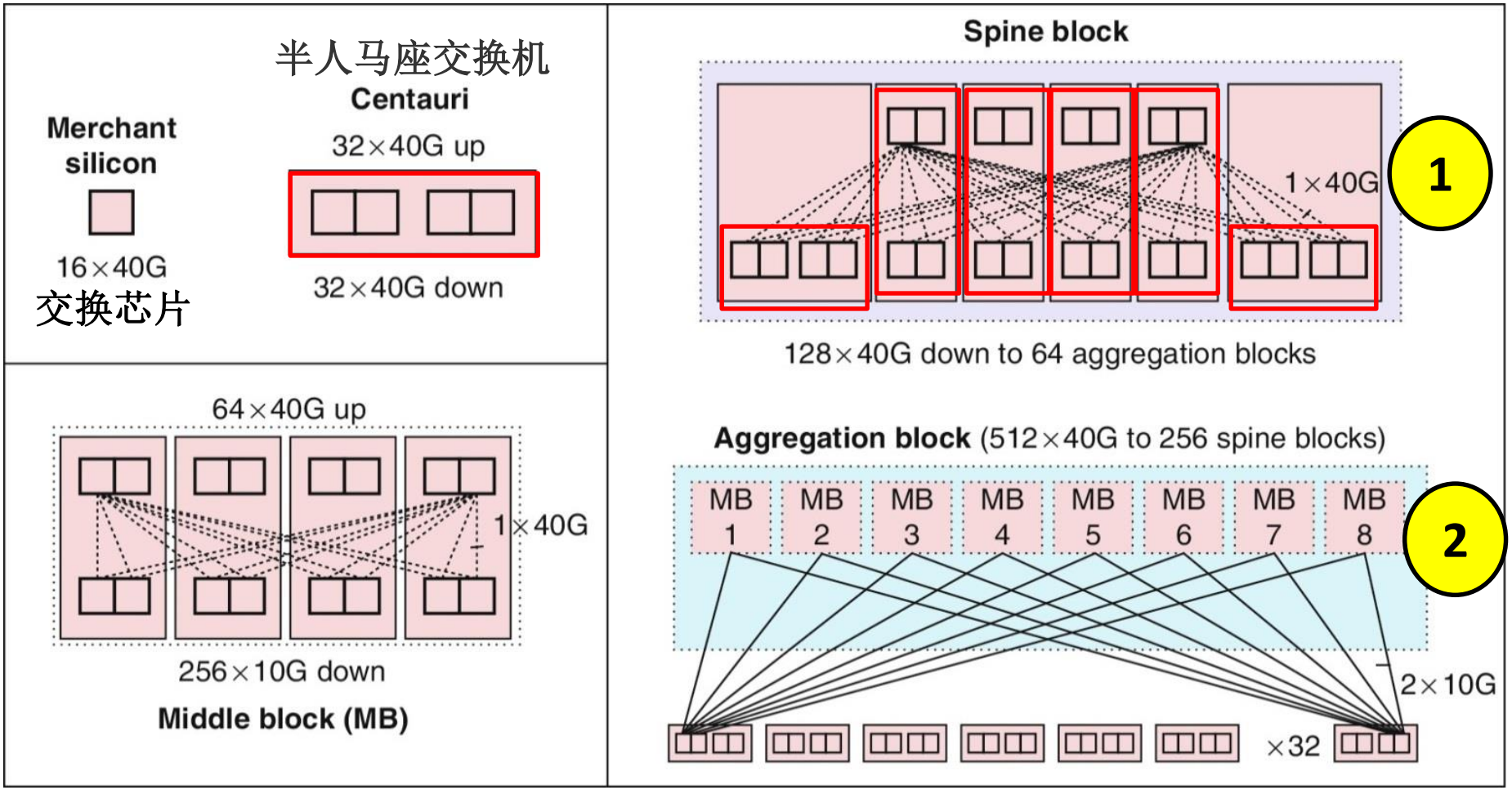
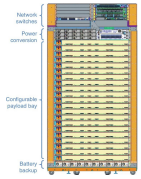


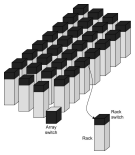
Figure 6.32 Building blocks of the Jupiter Clos network.

WSC 内存层次结构

- 服务器可以以访问 NUMA 节点的模式，访问其他服务器上的 DRAM 和磁盘

	Local	Rack	Array
DRAM latency (μs)	0.1	300	500
Flash latency (μs)	100	400	600
Disk latency (μs)	10,000	11,000	12,000
DRAM bandwidth (MB/s)	20,000	100	10
Flash bandwidth (MB/s)	1000	100	10
Disk bandwidth (MB/s)	200	100	10
DRAM capacity (GB)	16	1024	31,200
Flash capacity (GB)	128	20,000	600,000
Disk capacity (GB)	2000	160,000	4,800,000





存储设备的选型

- 使用服务器内部的磁盘
  - WSC一般常使用本地磁盘 （local disks）
- 通过 Infiniband 的网络扩展存储
  - Google 文件系统 (GFS) 扩展本地磁盘
  - 至少维护三个副本

WSC成本开销

- 建造成本：Capital expenditures (CAPEX)
  - Cost to build a WSC
- 运营成本：Operational expenditures (OPEX)
  - Cost to operate a WSC

Outline

- 仓库规模计算(Warehouse-Scale Computing)和数据并行
  - Warehouse-scale computers (WSCs)
  - 请求和数据级并行
  - Scale Up :扩展节点上的计算资源
  - Scale Out:通过WSC的调度程序架构扩展

26

新兴的计算系统体系结构设计

Today's Lecture

- 并行请求
  分配给服务器节点
  例如，搜索"Garcia"
- 并行线程
  分配给计算核心
  例如，Lookup
- 并行指令
  >1 指令同时执行
  例如，5 条流水线指令
- 并行数据
  >1 数据项同时处理
  例如，Add of 4 pairs of words
- 硬件
  All gates @ one time

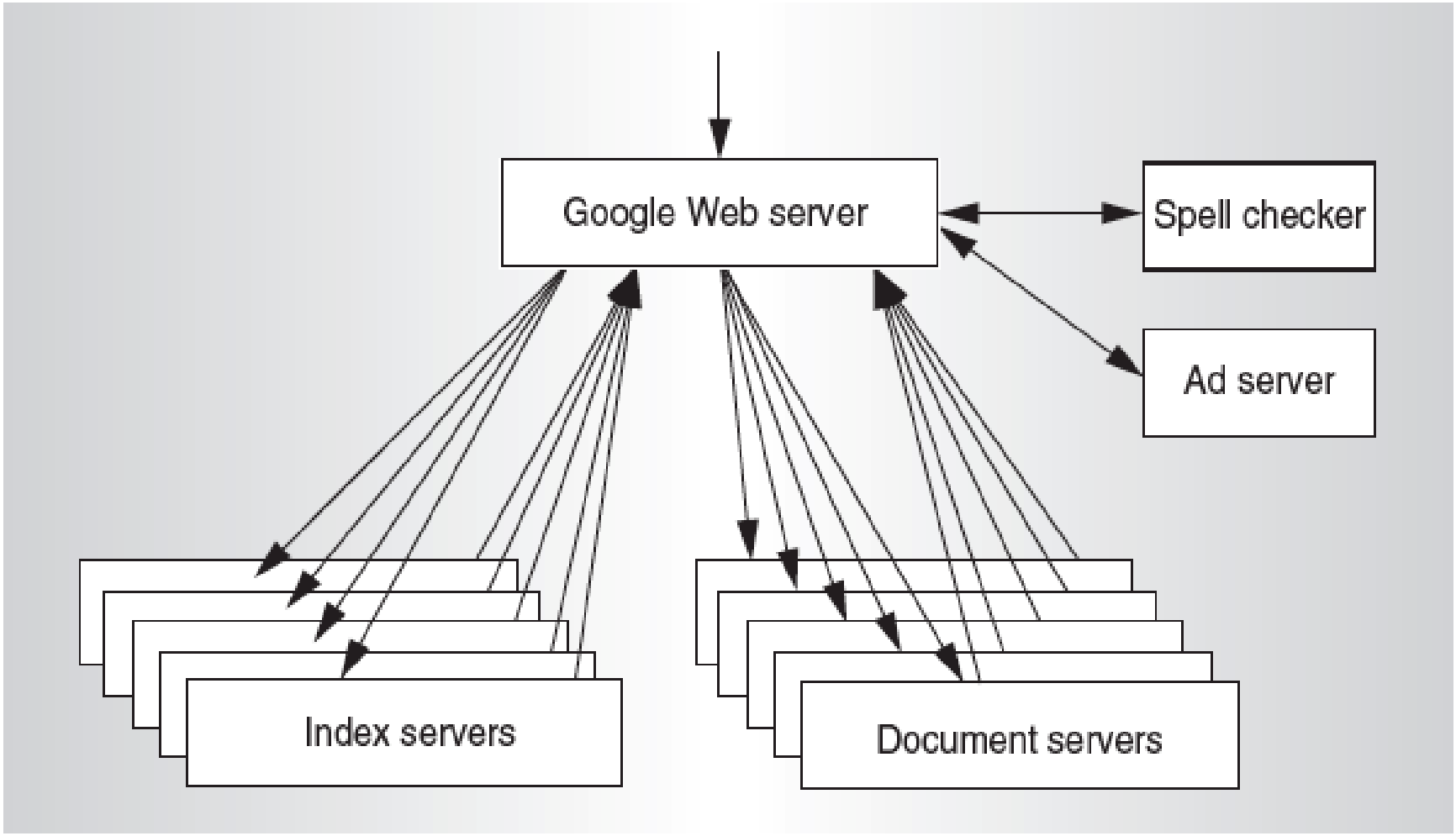
Software | *Hardware*

Harness Parallelism & Achieve High Performance

请求级并行Request-Level Parallelism (RLP)

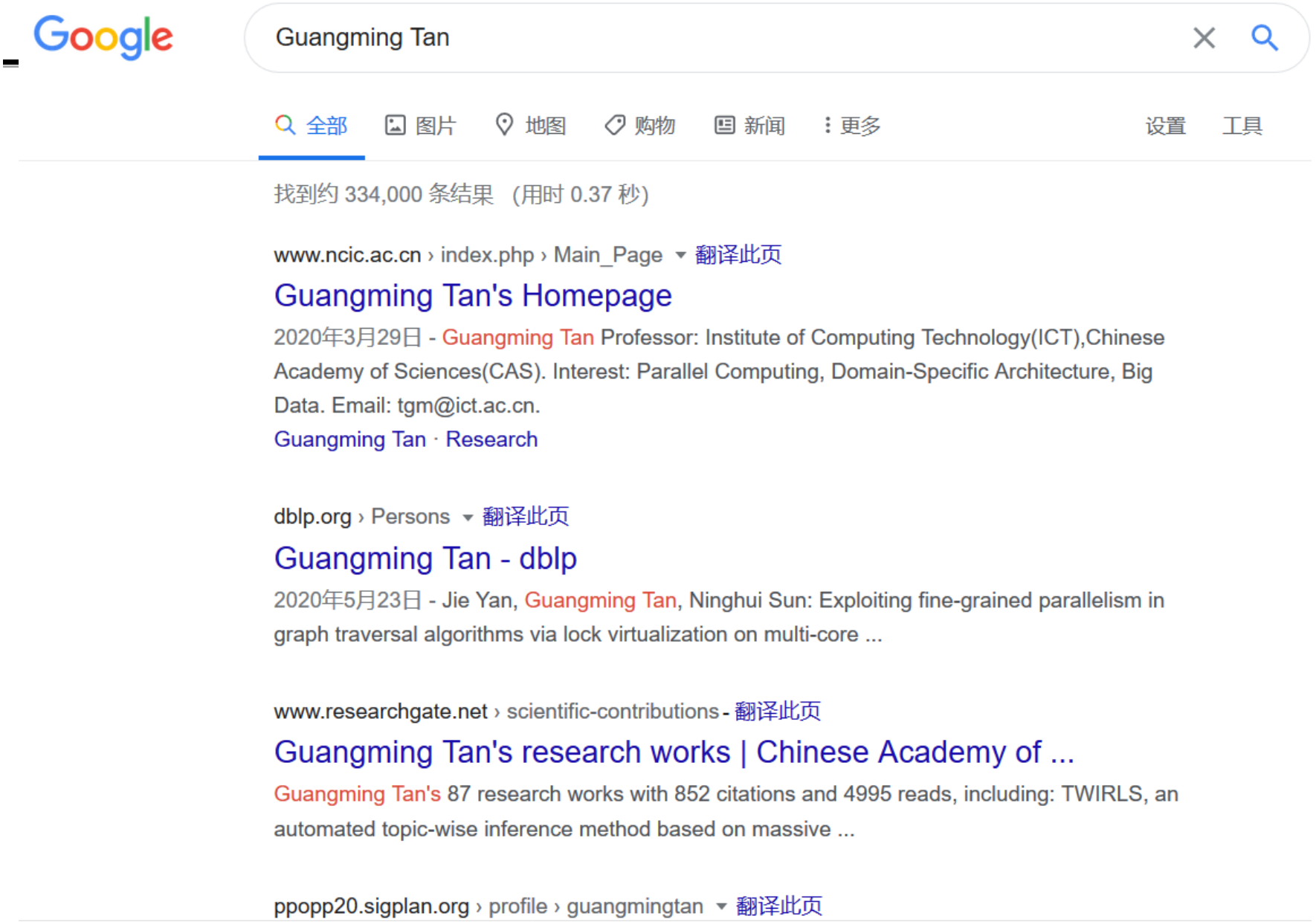
- 每秒处理数百或数千个请求
  - 批处理模型：不是你的笔记本电脑或手机，而是互联网服务，如网络搜索、社交网络……
  - 作业间无关：此类请求在很大程度上是独立的
    - 通常涉及以读取数据库为主
    - 很少涉及严格的读写数据共享或跨请求同步
- 计算在请求内和不同请求之间轻松分区

谷歌查询服务架构


Google Web server
Spell checker
Ad server
Index servers
Document servers

剖析一下网络搜索的过程

- Google "Guangming Tan"


Google
Guangming Tan
全部
图片
地图
购物
新闻
更多
设置
工具
找到约 334,000 条结果 (用时 0.37 秒)
www.ncic.ac.cn › index.php › Main_Page 翻译此页
Guangming Tan's Homepage
2020年3月29日 - Guangming Tan Professor: Institute of Computing Technology(ICT),Chinese Academy of Sciences(CAS). Interest: Parallel Computing, Domain-Specific Architecture, Big Data. Email: tgm@ict.ac.cn.
Guangming Tan · Research
dblp.org › Persons 翻译此页
Guangming Tan - dblp
2020年5月23日 - Jie Yan, Guangming Tan, Ninghui Sun: Exploiting fine-grained parallelism in graph traversal algorithms via lock virtualization on multi-core ...
www.researchgate.net › scientific-contributions - 翻译此页
Guangming Tan's research works | Chinese Academy of ...
Guangming Tan's 87 research works with 852 citations and 4995 reads, including: TWIRLS, an automated topic-wise inference method based on massive ...
ppopp20.sigplan.org › profile › guangmingtan 翻译此页

剖析一下网络搜索的过程 (1 of 3)

- Google "Guangming Tan"
  - 快照：直接请求"最近的"Google Warehouse Scale Computer
  - 负载均衡：前端负载均衡器将请求定向到 WSC 中的多个阵列（服务器集群）之一
  - 机柜内调度：所选择的机柜内，选择其中一个谷歌网络服务器（GWS），来处理请求并组成响应页面
  - 节点间通信：GWS 与索引服务器通信以查找包含搜索词"Guangming "、"Tan"的文档，并使用搜索位置
  - 结果返回：返回具有相关性分数的文档列表

剖析一下网络搜索的过程 (2 of 3)

- 夹带私货：In parallel,
  - 广告系统Ad system: 针对搜索词推送广告
  - 配图：获取各种适合搜索词条的图像
- ID结果分辨：使用docids（文档ID）访问索引文档
- 页面组成：构成页面
  - 网页快照：生成文档摘录，按照上下文中的关键字，以相关性分数排序
  - 链接：赞助商链接（顶部）和广告（两侧）

剖析一下网络搜索的过程 (3 of 3)

- 实施策略
  - 多副本：制作许多数据副本（又名"副本"）
  - 副本均衡：跨副本的负载均衡请求
- 索引和文件的冗余副本
  - 指令并行优化：增加请求级并行的机会
  - 容错：使系统更能容忍故障

数据级并行 (DLP)

- 2 个类型
  1. 数组相加：内存中的大量数据可以并行操作（例如，将 2 个数组相加）
  2. 文本查找：磁盘中的大量数据可以并行操作（例如，搜索文档）
- 通过MapReduce就可以在多个服务器和磁盘上进行 数据级并行(DLP)

Outline

- 仓库规模计算(Warehouse-Scale Computing)和数据并行
  - Warehouse-scale computers (WSCs)
  - 请求和数据级并行
  - Scale Up :扩展节点上的计算资源
  - Scale Out:通过WSC的调度程序架构扩展

36

如何解决同时处理大量数据？

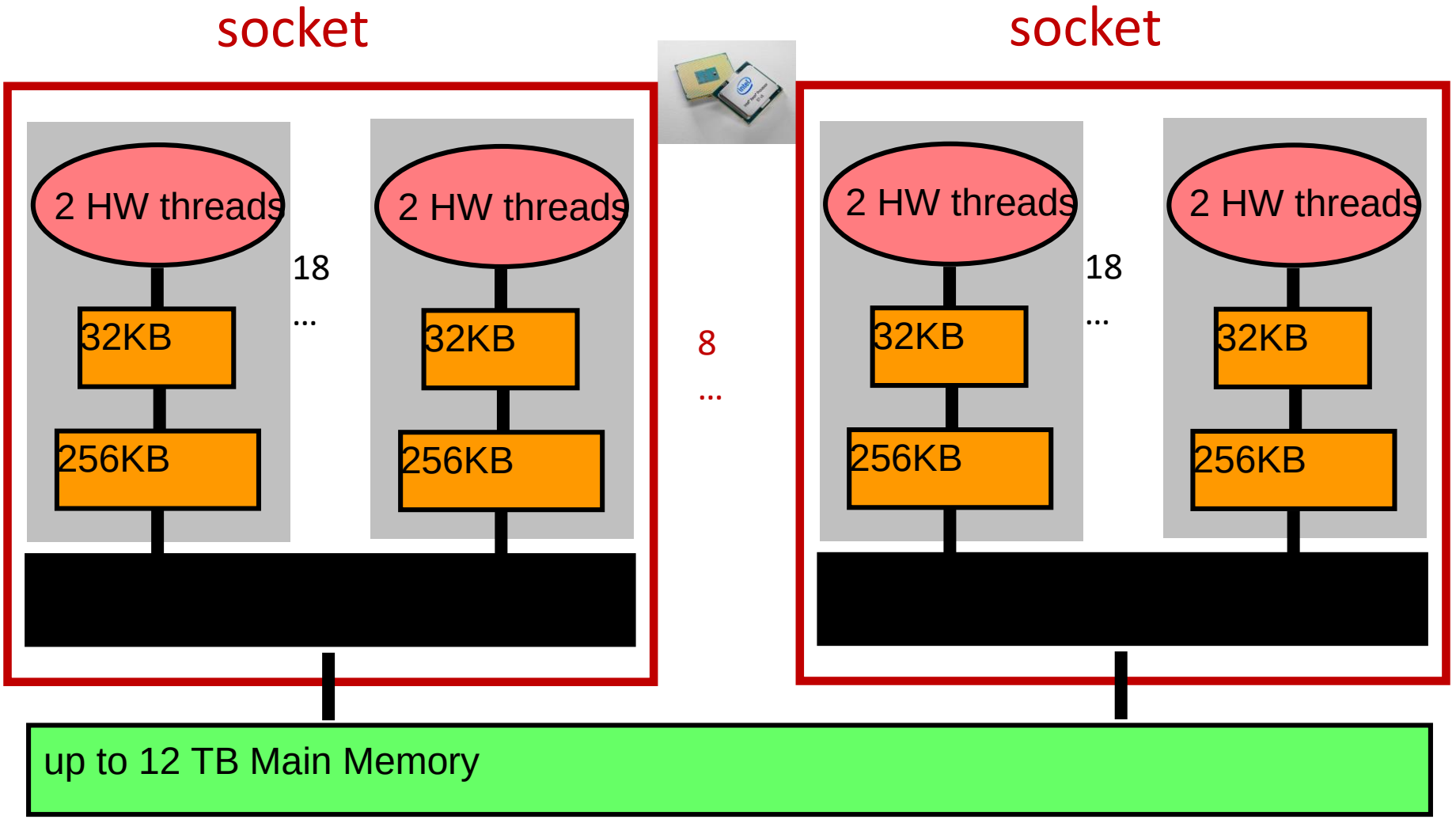
- **Scale Up 纵向扩展**
  - 通过并行处理和更快的内存/存储，扩大节点上的计算资源
- **Scale Out 横向扩展**
  - 增加计算节点的数量

Why **Scale Up** when you can **Scale Out?**

- 大数据的大部分焦点都集中在横向扩展（**Scale Out**）
  - Hadoop, etc

- 适用性不同：但是如果数据适合多核的内存，那么性能通常会提高一个数量级
  - 众核优势：Graph（multicore）比 Hadoop（cluster）快 1000 倍
  - 大内存节点：Multicores + 1-12 TB 内存：适合大多数图形分析问题！

Multicore: 144-core Xeon Haswell E7-v3



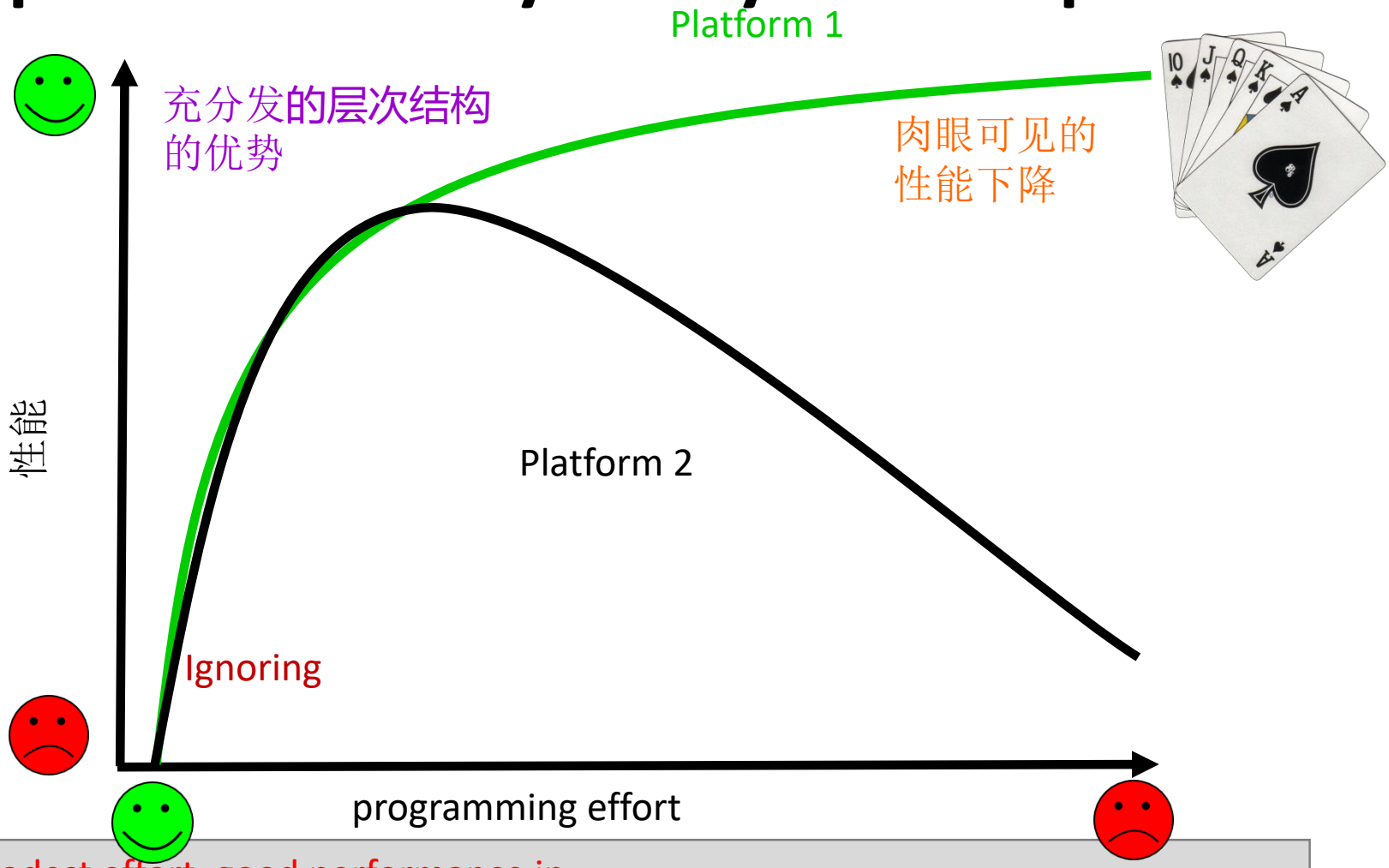
Attach: Hard Drives & Flash Devices

层次化趋势

- 计算性能分层：良好的性能/能耗，需要有效地利用层次结构
- 层次多维化：层次化变得越来越多
  - 更多计算核心
  - 更多级别的cache
  - 新的内存/存储技术
    - 闪存/SSD、新兴的 PCM
    - 各个层次结构间性能的差别——不能只看层次结构的最后一层

寻找层级架构的“甜区”

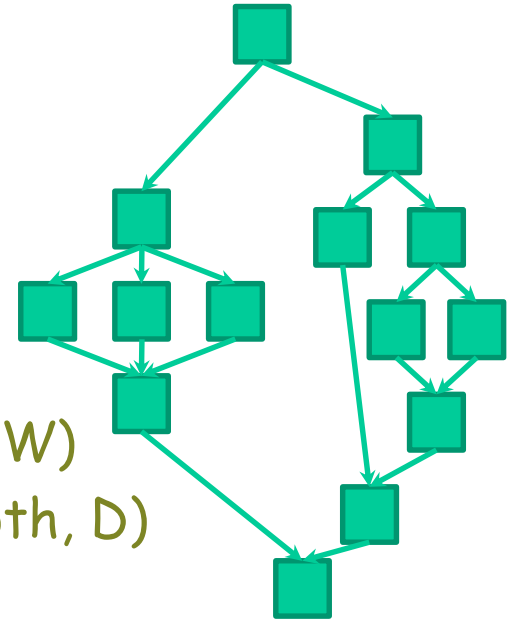
Hi-Spade: Hierarchy-Savvy Sweet Spot



Goals: Modest effort, good performance in practice, robust, strong theoretical foundation

程序分析Program-centric Analysis

- 从对程序的业务流程分析开始：动态有向无环图(DAG)
-



- 以程序为中心的指标
  - 处理单元数：Number of operations (Work, W)
  - 关键路径分析：Length of Critical Path (Depth, D)
  - 数据复用：Data reuse patterns (Locality)

目标：以程序为中心的指标 + 智能线程调度程序
在许多平台上提供可证明的良好性能

Outline

- 仓库规模计算(Warehouse-Scale Computing)和数据并行
  - Warehouse-scale computers (WSCs)
  - 请求和数据级并行
  - Scale Up :扩展节点上的计算资源
  - Scale Out:通过WSC的调度程序架构扩展

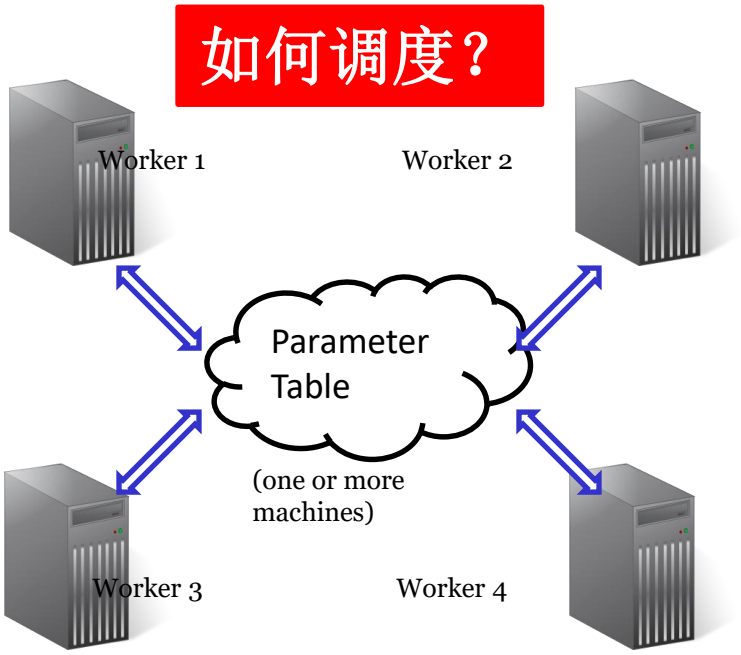
43

如何解决同时处理大量数据？

- **Scale Up 纵向扩展**
  - 通过并行处理和更快的内存/存储，扩大节点上的计算资源
- **Scale Out 横向扩展**
  - 增加计算节点的数量

分布式机器学习的参数服务器

- 使所有机器都可以方便地访问全局模型参数
- 可轻松转换单机并行 ML 算法
  - 分布式共享内存："Distributed shared memory" programming style
  - 取代集中化内存管理：Replace local memory access with PS access



Single Machine Parallel

```
UpdateVar(i) {
  old = y[i]
  delta = f(old)
  y[i] += delta
}
```

Distributed with PS

```
UpdateVar(i) {
  old = PS.read(y,i)
  delta = f(old)
  PS.inc(y,i,delta)
}
```

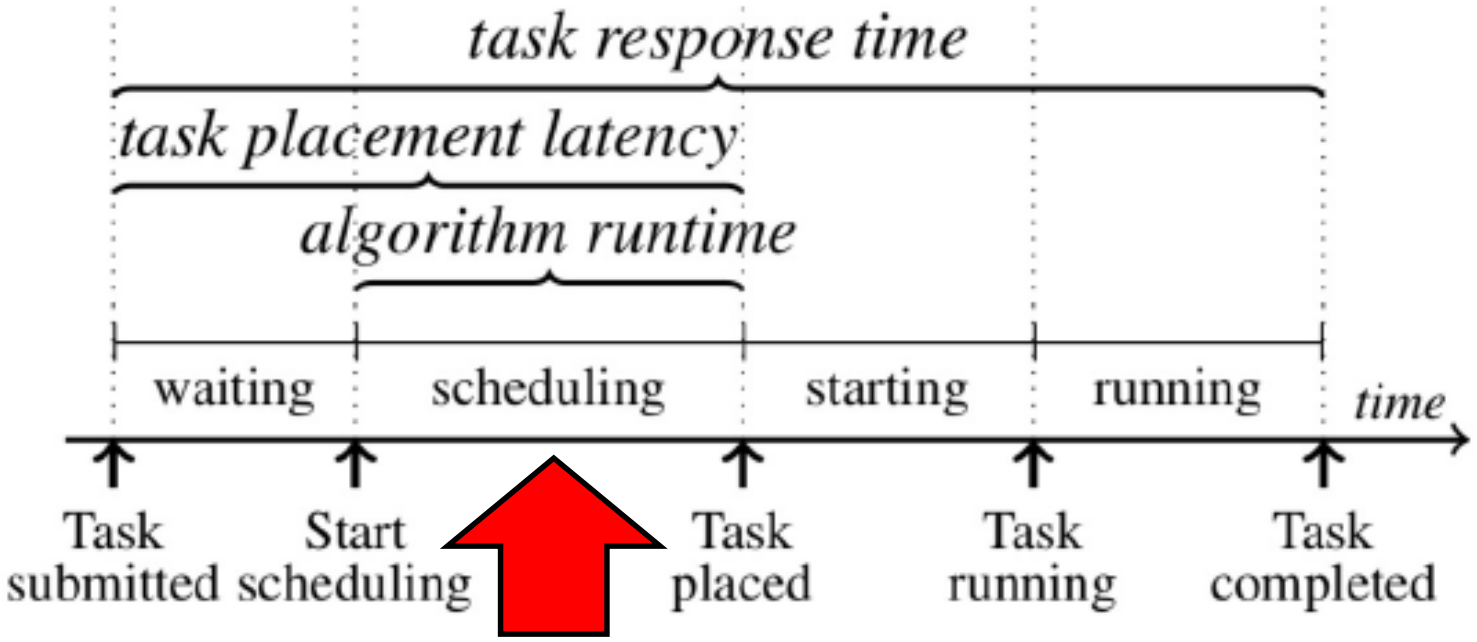
† Ahmed et al. (WSDM'12), Power and Li (OSDI'10)

WSC scheduler

- 调度任务
  - - 用户之间，优先为哪些用户分配资源
  - - 任务之间，优先为哪些任务分配资源
- 调度问题
  - - WHAT：按什么原则分配计算资源
    - 任务调度算法（资源位置）
  - - WHEN：何时分配计算资源
    - 任务调度时机（优先级）
  - - HOW：如何分配计算资源数量
    - 任务调度过程（资源数量）

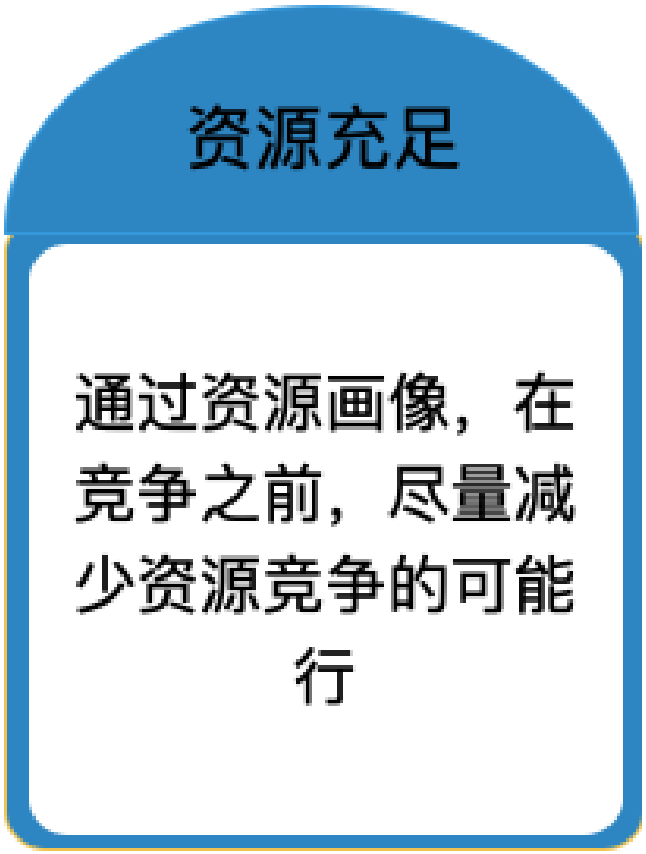
WSC scheduler（2）

- 任务运行生命周期
  - - 提交等待、任务调度、任务启动、任务运行
  - - 调度质量直接决定整个任务生命周期长短。

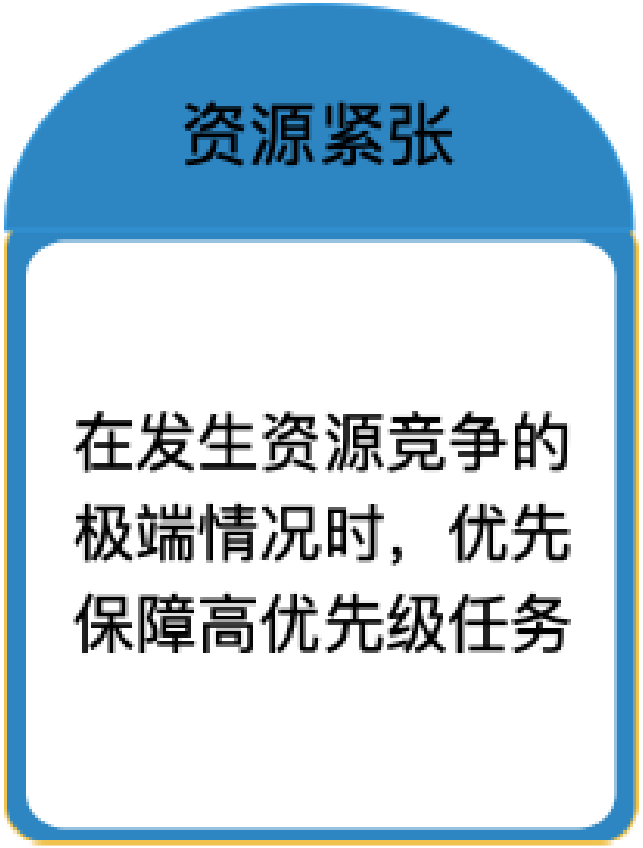


WSC scheduler（3）

- 调度原则：减少竞争，优先满足高级别任务



减少任务调度时间
提高资源利用率



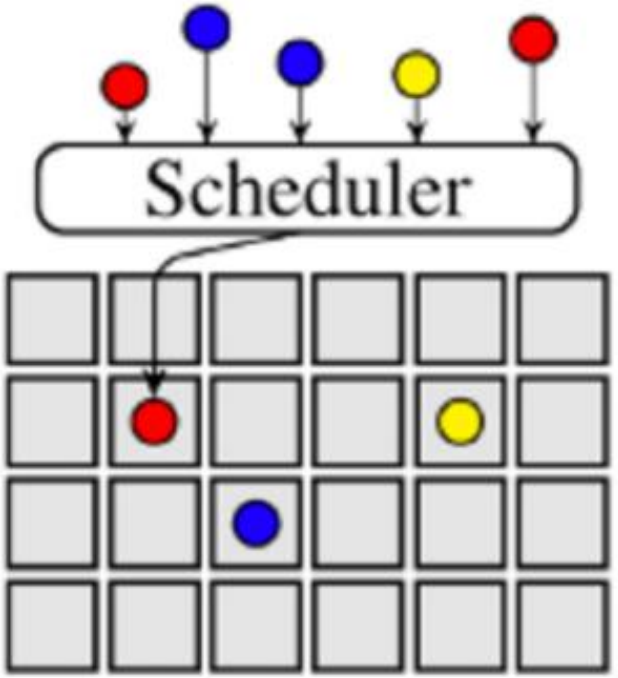
减少作业JCT时间
负载QoS保证

WSC scheduler's bottleneck

- 高性能平台的异构性和不同应用任务的混合共享计算大大增加了调度问题的难度
  - 不同应用存在不同指标要求（如吞吐量 、延迟）
  - 计算任务对于异构资源的需求（有的是CPU程序，有的是GPU）差异
  - 不同应用到来时间和对于不同资源需求量的不同
  - 真实场景存在不同节点计算能力差异、网络带宽有限及异常宕机等问题

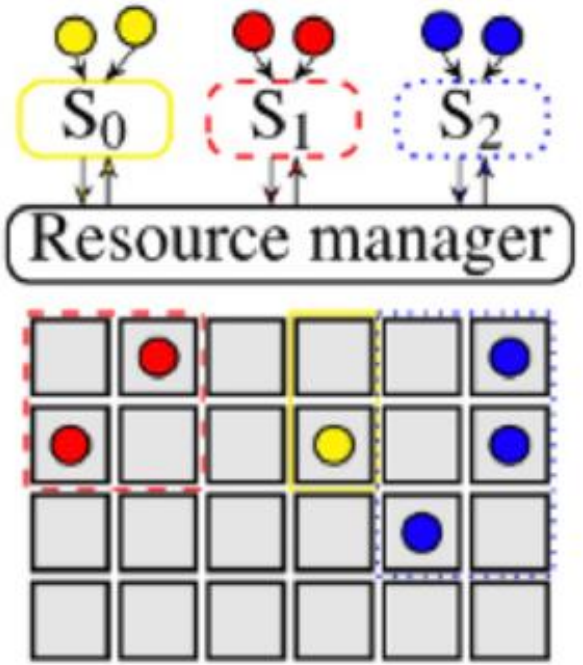
Evolution: 中心化调度框架

- 特点
  - - 单一调度进程，负责整个计算平台的任务调度
  - - 所有工作负载都有一个调度器处理
  - - 所有任务都通过相同的调度逻辑来处理
  - - 例如：SLURM、Borg、Hadoop和Kubernetes等
- 评论
  - - 单一调度器为满足不同应用不同需求，大大增加调度逻辑和部署难度
  - - 调度器本身具有扩展性瓶颈（N->1）



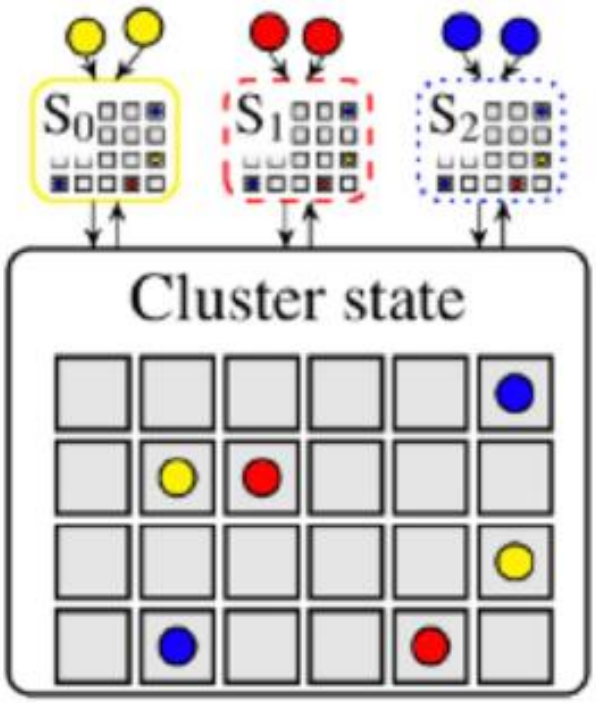
Evolution:两级调度框架

- 特点
  - - 将资源管理和任务调度分开
  - - 允许根据不同的应用做不同的调度逻辑 (调度策略)
  - - 例如：Mesos、YARN
- 效果
  - - 高优先级抢占会变得很难实现
  - -调度器无法看到运行任务间的干扰，对性能的影响



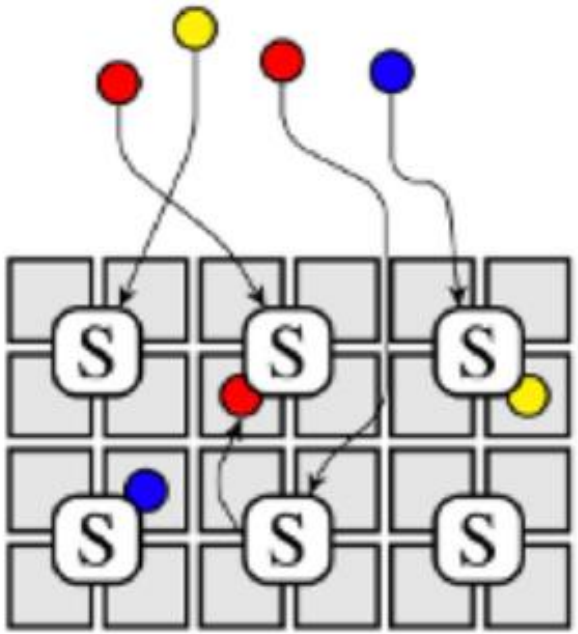
Evolution:共享状态调度框架

- 特点
  - - 每一个调度器拥有一个集群状态副本
  - - 独立维护本地集群状态副本，一旦有变化，发布事务更新整个集群状态
  - - 例如：Google的Omgea，Microsoft的Apolio，Hashicorp的Nomad
- 效果
  - - 必须工作在有稳定状态信息的情况下
  - - 集群资源竞争很高时，调度器性能会下降（碰撞问题）



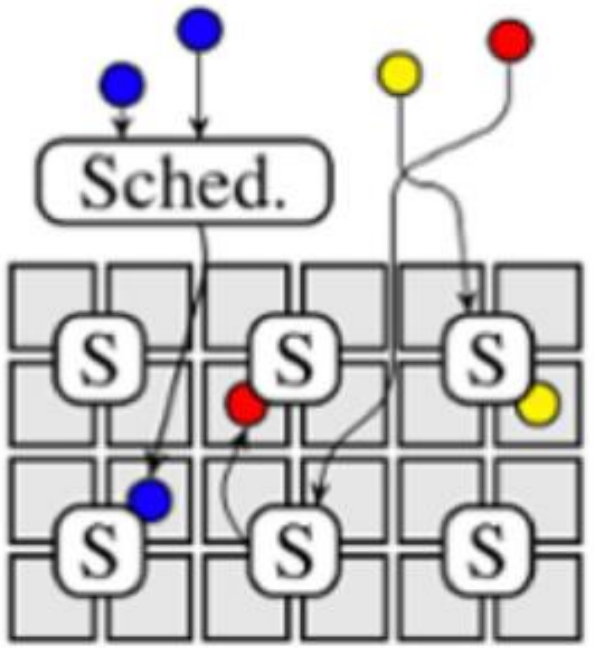
Evolution:全分布式调度框架

- 特点
  - - 去中心化，多个调度器彼此无任何协调
  - - 任务可以提交任何一个调度器，且调度器可将任务下发到任何一个计算节点
  - - 例如：Sparrow
- 效果
  - -调度服从统计和随机分布，无中央控制
  - - 调度快速，代价低
  - - 由于缺乏中央控制，无法保证优先级调度和公平策略调度



Evolution:混合式模式的调度框架

- 特点
  - - 结合中心化调度和共享状态的设计
  - - 两条调度：分布式调度负责非常短的任务或低优先级的任务，中心式调度负责剩余的负载
  - - 例如：Tarcil, Mercury, Hawk
- 评论
  - - 克服全分布式调度框架的不足



调度系统框架对比

	Framework	Architecture	Resource granularity	Multi-scheduler	Pluggable logic	Priority preemption	Re-scheduling	Oversubscription	Resource estimation	Avoid interference
OPEN	Kubernetes	monolithic	multi-dimensional	N[v1.2, DD, Issue]	Y[DD]	N[Issue]	N[Issue]	Y[DD]	N	N
	Swarm	monolithic	multi-dimensional	N	N	N[Issue]	N	N	N	N
	YARN	monolithic/ two-level	RAM/CPU slots	Y	N[*app-lvl. only*]	N[JIRA]	N	N[JIRA]	N	N
	Mesos	two-level	multi-dimensional	Y	Y[*framework-lvl.*]	N[JIRA]	N	Y[v0.23, Doc]	N	N
	Nomad	shared-state	multi-dimensional	Y	Y	N[Issue]	N[Issue]	N[Issue]	N	N
	Sparrow	fully-distributed	fixed slots	Y	N	N	N	N	N	N
CLOSED	Borg	monolithic[7]	multi-dimensional	N[7]	N[7]	Y	Y	Y	Y	N
	Omega	shared-state	multi-dimensional	Y	Y	Y	Y	Y	Y	N
	Apollo	shared-state	multi-dimensional	Y	Y	Y	Y	N	N	N

Figure 2: Architectural classifaction and feature matrix of widely-used orchestration frameworks, compared to closed-source systems.

End
Q/A

邵恩
高性能计算机研究中心