链滴

服务性能过低?

作者：linker
原文链接：https://ld246.com/article/1526481874002
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

关于性能是很多新手程序员最关心的话题.

然而性能是最不重要的,直到有性能问题.

决定一个服务的性能,最关键是架构和算法.

合适的架构,把费时的操作放在 异步的Job中,或者以P2P的方式化解,或者用客户端的计算能力;

合适的算法,以O(1)替代O(N),或者以直接用GPU来计算;

这都是很好的办法.

如果这些都不没有解决问题,那么围观的方法才应该被考虑.

下面我们来看一幅图:

Table 2.2 Example Time Scale of System Latencies

Event	Latency	Scaled
1 CPU cycle	0.3 ns	1 s
Level 1 cache access	0.9 ns	3 s
Level 2 cache access	2.8 ns	9 s
Level 3 cache access	12.9 ns	43 s
Main memory access (DRAM, from CPU)	120 ns	6 min
Solid-state disk I/O (flash memory)	50–150 μs	2–6 days
Rotational disk I/O	1–10 ms	1–12 months
Internet: San Francisco to New York	40 ms	4 years
Internet: San Francisco to United Kingdom	81 ms	8 years
Internet: San Francisco to Australia	183 ms	19 years
TCP packet retransmit	1–3 s	105–317 years
OS virtualization system reboot	4 s	423 years
SCSI command time-out	30 s	3 millennia
Hardware (HW) virtualization system reboot	40 s	4 millennia
Physical system reboot	5 m	32 millennia

现代的CPU已经非常复杂,关于性能需要记住的几点是:

- 内存操作很慢,最好是不要用间接方式操作数据(例如map[string]*struct xxx),直接复制小结构体会好.
- 系统调用非常非常慢,而且CPU核心越多系统调用越慢.一次系统调用往往会导致TLB命中率暴降,当前ore的L1 Cache完全被刷新.
- SSD有时候比系统调用快.
- 网络IO很慢,进程内的缓存远快于远程缓存.
- 缓存一致性协议非常非常重要. 在多个线程间不合适的共享一个结构体会导致灾难性的性能下降.例用一个Go的chan在100个goroutine间共享,并高频率的读写.这会导致缓存一致性协议频繁的刷新不同ore的不同级别的Cache,性能会非常低.
- 原子操作在CPU核心很多(>60)且高冲突的情况下,并不快.
- 线程切换非常非常的慢. 一次线程切换往往会导致缓存的命中率暴降.

原文链接：服务性能过低?