高等计算机体系结构，2020年春季

作业6： 预取和并行

主讲教师：栾钟治

助讲教师：杨海龙；助教：孙庆骁

作业下发时间：2020年6月05日

作业回收时间：2020年6月19日

# 预取 I 20分

假如你是一位架构师，正在为你的机器设计预取引擎。你先在机器上使用跨度预取器执行了A和B两个应用。

应用A:

uint8\_t a[1000];

sum = 0;

for (i = 0; i < 1000; i += 4)

{

sum += a[i];

}

应用B:

uint8\_t a[1000];

sum = 0;

for (i = 1; i < 1000; i \*= 4)

{

sum += a[i];

}

i 和 sum在寄存器中，数组a在内存中，一个cache块大小为4个字节。

(a) 使用跨度预取器，应用A和B的预取精度和覆盖率分别是多少？这个跨度预取器检测两次连续访存的跨度，从当前访问的cache块按照这个跨度预取下一个cache块。

**答：应用A可以看到执行了1000/4=250次，第一次去内存取数组，而cache块大小为4字节32位，数组访问每次间隔为4，数组存储单位为无符号的8位，刚好每次访存时候都是一个cache块，因此第一次和第二次会访存形成跨度为4的预取模式。访问序列为a[0]，a[4]， a[8]，…，a[996]，预取序列从a[8]，…，a[996]，a[1000]（访问a[996]时候会预取a[1000]），因此有249个块被预取，248个块被使用，所以预取精度为248/249=99.6%，cache块缺失250次，预取消除块缺失有248次，因此覆盖率为248/250=99.20%**

**应用B访问为a[1]，a[4]， a[16]， a[64]， a[256]，由于访存的跨度不是常数，所以所有预取都没命中，因此覆盖率和命中率都为0。**

(b) 请分别为应用A和B建议能获得更好的精度和覆盖率的预取器

i) 应用A

**答：预取器总是预取下一个块的方式，这样的话a[4]这个块在a[0]访问的时候就会被预取，从而使得预取精度提高到为249/250，覆盖率为249/250。**

ii)应用B

**答：目前没有特别好的预取方式能够提升应用B这种翻倍访问的方式，大多数的预取器比如跨度、流、下一块都没办很好的预取应用B访问的块，没办法提供一种合适的访存模式。**

# 预取II 25分

你跟你的同学一起设计一个预取器，这台机器使用单核、 L1 和 L2 cache 以及DRAM 内存。我们需要分析不同的预取器和可能的tradeoff。

在本题中，我们要计算预取器在达到稳定状态后的预取精度、覆盖率和带宽开销，所以，**所有计算都不包括最开始的6次请求，这6次请求作为预取器的训练集。**

(a)你首先设计一个跨度预取器，观察最后三次cache块请求，如果最后三次请求的跨度是常数，预取器将会使用这一跨度预取下一个cache块。

你执行了一个应用，它具有如下的访存模式 (这些是cache块地址):

A，A+1，A+2，A+7，A+8，A+9，A+14，A+15，A+16，A+21，A+22，A+23，A+28，A+29，A+30...

假设这个模式持续了很长时间。

计算你的跨度预取器对于这个应用的精度和覆盖率

**答：预取器的规则是假如最后三次请求的跨度是常数，那么就预取这个跨度的下一块，那么以**A+14，A+15，A+16**为例子，预取器发现符合规则预取A+17但是没有命中，所以预取精度和覆盖率为0。**

(b) 你的同学设计了一个新的预取器，当有一个cache块访问时，预取紧接着的N个cache块

(i) 如果用他的预取器执行你刚刚执行过的应用，预取覆盖率和精度分别是66.67%和50%，N是多少?

**答：N=2。**

**当N=1时候，预取覆盖率为66.67%，精度为66.67%不正确。当N=2时候，访问块14，预取器预取块15,16，访问块15时候预取16,17，访问块16预取17,18。因此预取序列中覆盖访问序列的覆盖率为66.67%，而预取的精度为50%，只有一半的预取命中。**

(ii) 假如我们将带宽开销定义为：有预取器时所有cache块的请求数/没有预取器时所有cache块的请求数，那么你同学的预取器在执行刚才那个应用时的带宽开销是多少？

**答：首先这里假设预取器的重复请求是不在计数中的，因此按照上一题的分析，宽带开销=5/3。例如访问14，预取15,16,17,18。**

(c) 你的同学希望改进他的预取器对于刚才那个应用的覆盖率，他可以容忍带宽开销最多两倍。请问他能做到吗？为什么可以/不可以？

**答：不可以。想要获得更高的覆盖率必须要让预取跨过每3组连续的中间跳过的块，比如预取访问A+14,A+15,A+16，假如想要让A+14和A+21被预取，那么至少要预取5个，那么带宽开销为(4+3)/3=2.33>2，增加的带宽开销超过2倍。**

(d) 对于上面的应用，如果想获得100%的覆盖率，N最小得是多少？这个时候的带宽开销是多少?

**答：N最小为5，带宽开销为7/3=2.33**

# Cache 一致性 10分

(a) MESI cache一致性协议比MSI 协议好在哪里?

**答：MESI cache一致性协议比MSI协议增加了一个状态E(Exclusive独占)，表示的是cache中拷贝的独占的，并且是干净的。假如总线发生读操作时候，没有其他的cache有拷贝，那么块就会转移到E状态。假如总线发生写操作时候，如果cache状态为独占状态，那就可以省去没必要的更新广播，节省带宽。**

(b) 你想要利用MESI置无效协议设计一个基于目录的cache一致性系统，在特定的工作负载下，系统表现得很糟糕，经过仔细的分析，你发现有4个节点持续的发出对某个cache块的置无效请求，这什么情况?

**答：四个节点持续对cache访问，产生写竞争，形成ping-pong效应，互相抢占置无效请求。**

(c) 如何解决这一问题?

**答：假如竞争是真共享导致的，那么要修改代码减少竞争的出现，或者使用基于更新的一致性原则。**

**假如竞争是假共享导致的，可以通过优化编译器或者重写代码消除竞争。**

# 一致性协议 15分

假设有一个多处理器系统，系统有512个处理器，每个处理器有1MB的私有写回方式的cache，每个cache块64字节，主存大小为1GB。

(a)如果我们基于MESI cache 一致性协议设计了监听总线，需要多少状态位才能够实现这个一致性协议? 这些状态位放在哪?

**答：每个cache块需要2个状态位来表示MESI cache 一致性协议，总共有512\*(220÷26)=223个cache块，因此总共需要223×2=224位。**

**状态位存在私有cache的标签存储中，即tag域处。**

(b) 如果用基于目录的cache一致性协议替换，需要多少状态位? 这些状态位在哪?

**答：对于内存中的每一个内存块，需要在目录中存储总共512+1=513位，512位标识内存块被哪个处理器cache所拥有，1位用于标识该块是否唯一被处理器cache所有，因此需要230÷26×513=224 ×513位存储。其次仍然需要处理器私有cache中的224位存储MESI状态位。**

**因此总共需要224+513×224位。**

**目录状态位：存储在cache目录表中。MESI状态位：储存在私有cache中。**

(c) 对于这个系统，你会选择哪一个协议? 为什么?

**答：选择基于目录的cache一致性协议。过多的处理器使用总线监听机制会导致总线的负担加重，降低CPU性能，减少正常通信的带宽，因此基于目录的系统更加合适，比较适用于多处理器的系统。**

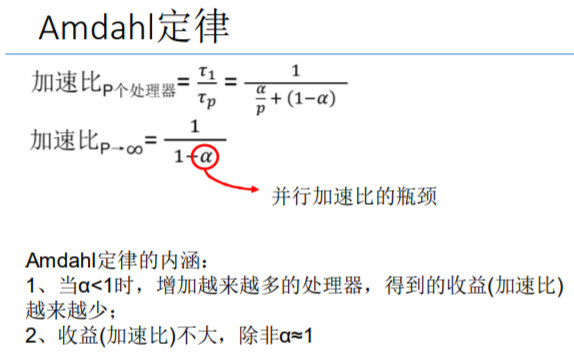
# 并行加速比 30分

假如你是一家公司的程序猿，你被要求并行化一个老程序以使它能够在现代多核处理器上跑得更快。

(a) 你并行化了这个程序，然后发现它对于单线程版本的加速比相比于处理器个数的增加而言相差很多。你发现在每个核的数据cache中有大量的cache无效存在，什么样的程序行为导致了这种现象? (请用10个字左右简要说明)

**答：由于共享数据冲突导致的cache ping-pong效应。**

(b)你修改了程序以解决这个性能问题，然后你发现程序在每个并行计算之后的一个单个线程都会更新一个全局状态，因此导致性能的下降。你的程序有90%的工作是并行的(按照处理器个数X秒计算得出)，另有10%的工作是串行的，并行部分是完美的并行。如果多核处理器核数无限，程序的最大加速比可以到多少?

**答：我们由****可以得到：**

**当有n个处理器时，加速比(n)=1/(1-a+a/n)其中a为并行计算部分所占比例。**

**由于 n→∞， 加速比(n)=1/(0.1+0.9/∞)→10。**

(c) 如果要获得4倍的加速比，应该有多少处理器?

**答：加速比(n)=1/(1-a+a/n)=4，a=0.9，则n=6。**

(d) 为了使你改写的程序更高效，公司决定设计一款专门的异构处理器。这款处理器由一个大核(执行代码更快，但是占据的片上面积更大)和多个小核(执行代码更慢，但是消耗面积更小)共享处理器的片上空间。

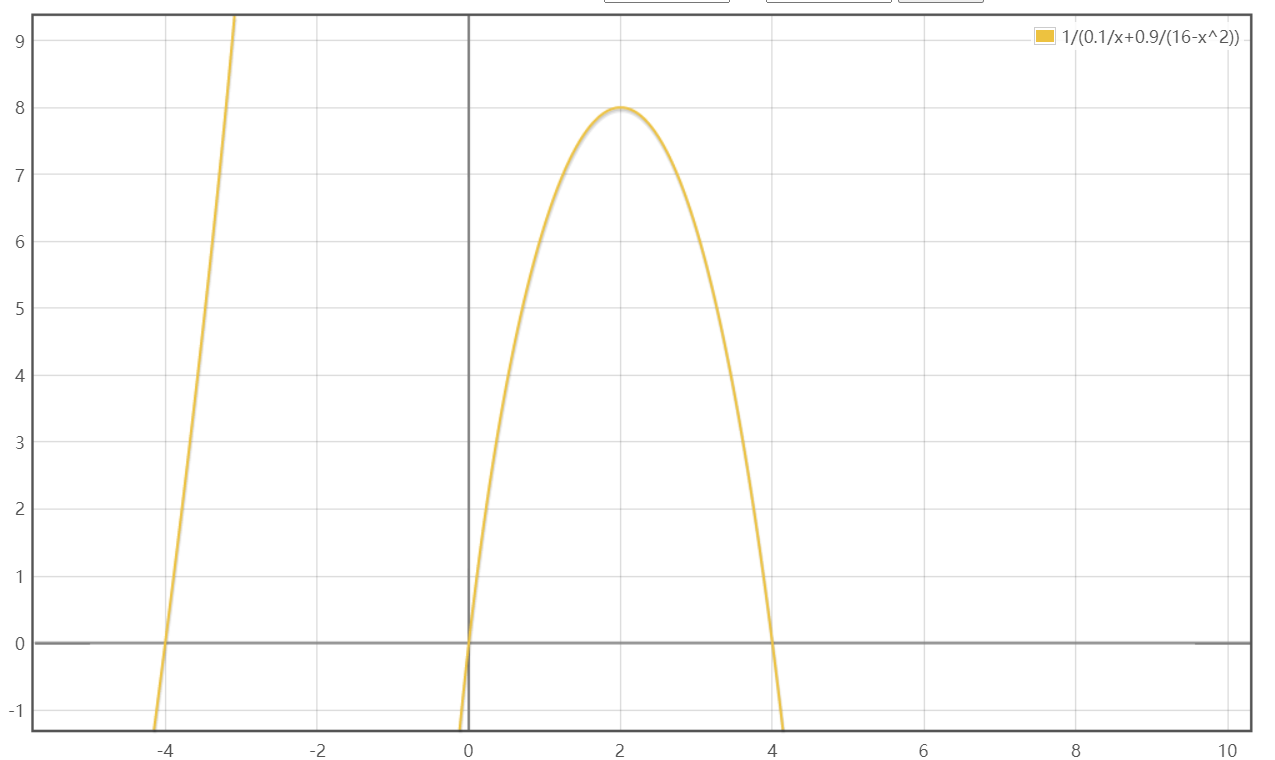
你的程序并行部分的所有线程将只会在小核上执行；程序的串行部分将有一个线程执行在大核上。核的性能(执行速度)与它的面积的平方根成正比。

假设芯片面积有16个单元可用，一个小核至少占用1个单元，大核可以占用任意数量的单元。同时假设没有被大核使用的面积会被小核填满。

(i) 如果想让你的程序获得可能的最快执行速度，大核需要多大?

**答：假设大核面积为x2，小核的面积为16- x2，因为核的性能(执行速度)与它的面积的平方根成正比，那么总加速比为：1/(串行部分加速比+并行部分加速比)**

**简单通过绘图可以得到当x>=0时候，当x=2时，加速比取最大为8。大核面积为4，小核面积为12。**



(ii) 如果所有16个单元全部拿来用做小核，这个处理器就变成了同构的多核处理器，对于你的程序而言，它的加速比是多少? 假设串行部分跑在一个小核上，并行部分跑在所有16个小核上。

**答：1/(0.1+0.9/16)=6.4**

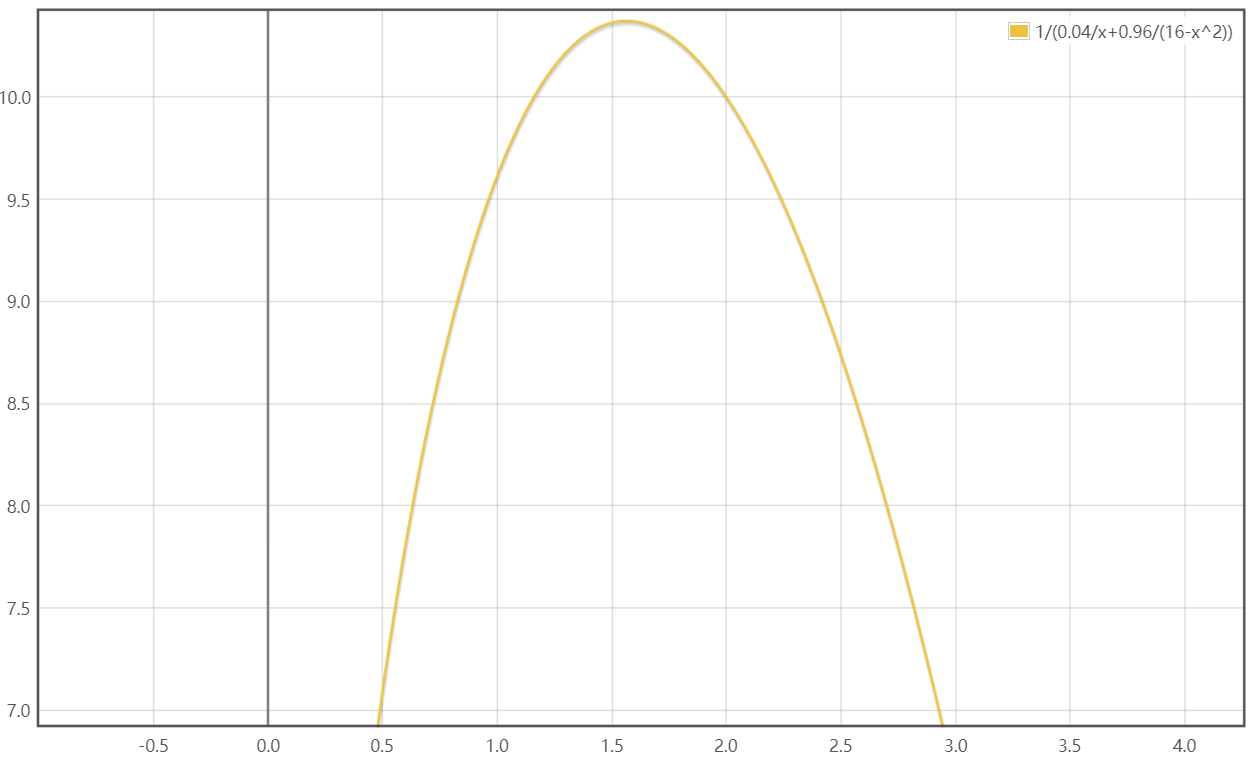
(iii) 在串行部分是10%的情况下，使用异构多核(大小核)处理器是有意义的吗? 为什么是/不是?

**答：是。从上面的计算可以看出，使用大核对串行部分获得的加速比超过不使用大核纯并行计算的加速比。**

(e) 现在你继续优化了你的程序，使得串行部分仅占4%(剩下96%是并行部分)。

(i) 这个时候大核应该有多大(占多少单元)?

**答：按照上面的计算公式**



**大核的面积要求为整数，可以看到x=1是小于x=2的加速比的，所以大核占4个单元。**

(ii) 大核这么大的时候加速比是多少?

**答：加速比为10**

(iii) 假如此时我们采用16个小核的同构多核处理器，你的程序的加速比是多少(假设串行部分跑在一个小核上，并行部分跑在所有16个小核上)?

**答：1/(0.04+0.96/16)=10**

(iv) 在串行部分是4%的情况下，使用异构多核(大小核)处理器还是有意义的吗? 为什么是/不是?

**答：不是，因为异构和同构的加速比一样，没有带来任何提升，而因为异构多核处理器设计更为复杂，所以从成本来看设计异构没有意义。**