高等计算机体系结构，2019 年春季

作业 6： 预取和并行（参考答案）

主讲教师：栾钟治

助讲教师：杨海龙；助教：许崇杨，左佩璇

作业下发时间：2019 年 5 月 20 日

作业回收时间：2019 年 6 月 03 日

1 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。

参考答案：
应用 A 的预取精度是 248/249，覆盖率是 248/250。
应用 A 访问 a[0], a[4], a[8], … a[996]，有 1000/4 = 250 次访存。前两次访问 a[0]和 a[4] 不命中，之后，预取器学习到跨度是 4 并开始预取 a[8], a[12], a[16]等等直到 a[1000] (访问 a[996]时 a[1000] 被预取，虽然并没有被用到)。统计结果，249 个 cache 块被预取，248 个被使用。
因此，预取精度为 248/249，覆盖率为 248/250。

应用 B 的预取精度为 0，覆盖率为 0。

应用 B 访问 a[1], a[4], a[16], a[64]和 a[256]，有 5 次访存。然而，由于这些访问的跨度不是常数，因为数组索引是 4 的倍数，而不是增或者减一个常数。因此，跨度预取器无法预取到被访问的 cache 块，使得预测精度和覆盖率均为 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

参考答案：
下一块预取器总是预取下一个 cache 块，因此，有 a[4] 的 cache 块也会被预取，则预取精度提高到 249/250 并且覆盖率仍保持 249/250。

ii)应用 B

参考答案：
大多数普通的预取器比如跨度、流、下一块等都无法提升应用 B 的预取精度，因为无法为这些预取器提供一个合适的访存模式。某些采用预执行方法的预取，比如 runahead 执行或者双核执行可能能够改进应用 B 的预取精度。

2 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

参考答案：
0%，0%。
每三个一组的请求之后，预取按照检测到的跨度触发，但是预取到的块总是无用的；需要的请求不会被这个预取器的预取覆盖。

(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?

参考答案：
N= 2。
比如在访问块 14 之后，预取器预取块 15 和 16，访问 15 之后预取 16 (与已经发射的预取整合) 和 17，访问 16 之后预取 17 和 18。因此，每三个需要的访问中的两个被覆盖(66.7%)，并且预取的数据一半是有用的 (50%)。

(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？

参考答案：
5/3。
对于每一组连续三个访问的 cache 块，有两个额外的块被预取。比如，取 cache 块 14, 15 和 16，块 17 和 18 也会被预取。

(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？

参考答案：
不可以。
要获得更好的覆盖率，预取器必须能够跨过上一组 3 个请求取到下一组中的请求，因为上一组的三个请求都已经在前一次预取到了。比如，访问 A+14, A+15 和 A+16，它们都已经被预取了，要想提高覆盖率，需要预取到 A+21 (下一组三个跨度请求中的第一个)，但是，这需要预取 A+16 和 A+21 之间 的四个块(A+17, A+18, A+19, A+20)，增加的带宽开销超过两倍。

(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?

参考答案：
N= 5 (这样， A+16 预取 A+21， A+21 预取 A+22, A+23 等等)；
带宽开销是 7/3。

3 Cache 一致性 10 分

(a) MESI cache 一致性协议比 MSI 协议好在哪里?

参考答案：
允许 cache/处理器写一个位置(唯一的干净拷贝)而不需要通知其它的处理器/cache。

(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?

参考答案：
4 个节点间发生 cache 块乒乓现象。

(c) 如何解决这一问题?

参考答案：
如果现象是由真共享导致的，重写代码减少共享或者使用同步原语减少通信。如果搞不定，可以考虑使用基于更新的一致性协议。
如果是由伪共享导致的，通过编译器或重写代码改变数据的布局，以消除伪共享。

4 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?

参考答案：
2^{24} 位。
总共有 2^{23} 个 cache 块 (cache 有 2^{20} 字节，2^{9} 个 cache，因此有 2^{29} 字节 在私有 cache 中；用每个 cache 块 2^{6} 字节去除)，每块需要 2 位表示状态 (M, E, S 或 I)，因此需要 2^{24} 位。
这些位存在于私有 cache 的标签存储中。

(b) 如果用基于目录的 cache 一致性协议（像我们课堂上讲的例子那样）替换，需要多少状态位? 这些状态位在哪?

参考答案：
2^{24} x 513 + 2^{24} 位
仍然需要在私有 cache 中的 2^{24} 个 MESI 状态位，然后，必须计算目录存储空间。共有 2^{24} cache 块在主存中，每个块需要对应每个处理器 1 位外加 1 个独占位(每个块 513 位)。因此目录中一共需要 2^{24} x 513 位。
目录位: 存在于 cache 目录；MESI 状态位: 存在于私有 cache。

(c) 对于这个系统，你会选择哪一个协议? 为什么?

参考答案：
目录。
总线无法扩展到 512 个处理器，目录可以。
虽然基于总线的监听系统对于存储的需求要低很多，但是总线无法提供足够的带宽来维持 512 个处理器的需要。因此，基于目录的系统(用可扩展的互连网络构建)更合适。

5 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)

参考答案：
由数据共享导致的 cache 乒乓

(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?

参考答案：
10。
根据 Amdahl 定律: 对于 n 个处理器，加速比(n) = 1/(0.1 + 0.9/n)
由于 n→∞， 加速比(n)→10。

(c) 如果要获得 4 倍的加速比，应该有多少处理器?

参考答案：
6。
由 加速比(n) = 4:
$4 = 1/(0.1 + 0.9/n) \rightarrow 0.25 = 0.1 + 0.9/n \rightarrow 0.15 = 0.9/n$， $\therefore n = 6$

(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?

参考答案：
4 个单元。
如果给定大核的尺寸是 n^2，则大核在串行段的加速比是 n，16-n^2 个小核实现并行段的并行化。这样加速比= $1/(0.1/ n + 0.9/(16- n^2))$。为了最大化加速比，需要最小化分母。对于 n=1，分母是 0.16；n=2，分母是 0.125；n=3，分母是 0.1619 。因此 n=2 是最佳的，所以大核占据 $n^2 = 4$ 个单元。

(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。

参考答案：
6.4
加速比= $1/(0.10 + 0.90/16) = 6.4$

(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?

参考答案：
是。
因为串行部分足够大，使得大核在串行部分获得的加速比超过了由于大核带来的并行吞吐的下降。

(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)?

参考答案：
4 个单元
跟之前的题类似，加速比= $1/(0.04/ n + 0.96/(16- n^2))$。最小化分母以最大化加速比。n=1，分母是 0.104；n=2，分母是 0.1；n=3，分母是 0.1504。因此大核占 4 个单元。

(ii) 大核这么大的时候加速比是多少?

参考答案：

10
加速比= 1/(0.04/2 + 0.96/12) = 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)?

参考答案：
10
加速比= 1/(0.04/1 + 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?

参考答案：
不是。
异构系统如果无法提供比同构系统更多的性能收益，由于它比同构系统设计复杂得多，所以将是没有意义的。