**并行计算 作业**

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 王少博 | 学号 | 181110315 |
| 总分 | 10 | 实际得分 |  |
| 作业内容（问题，思路，程序，结果，过程中遇到问题的解决方法） | | | |
| **问题1**：  a. 假设一个共享内存系统使用窥探缓存一致性（snooping cache coherence）和写回（write-back）机制，core 0 的缓存中有变量 x，并执行赋值语句x=5，假设 core 1 的缓存中没有 x，在core 0 更新 x 后，core 1 尝试执行y=x，y 的值是什么？为什么？  b. 假设上面的共享内存系统使用基于目录的缓存一致性（Directory-based cache coherence），y 的值会是什么？为什么？  c. 如何解决上面出现的问题？  **解答**：首先对于问题的条件进行分析。   1. 窥探缓存一致性的定义：1)所有的core 共享总线2)总线上传输的任何信号都可以被连接到总线的core“看到”3)当core 0 更新存储在其缓存中的x 的副本时，向总线广播此信息4)如果 core 1“窥探”总线，将看到 x 已被更新，可以将其 x 的副本标记为无效5)实际的广播只通知其他core 包含x 的cache line 已经更新; 2. 写回（Write-back）：当写入 Cache 时，不马上写入主存，而是将cache 中更新的数据标记为 dirty ，当该 cache line 被从主存中读取的新的cache line 所替换时，将该dirty line 写入主存 3. 基于目录的缓存一致性: 1)使用称为“目录”的数据结构来存储每条cache line的状态   2)当一个变量被更新时，会查询目录，并且在其缓存中包含该变量的cache line将失效3)目录需要额外存储空间，但是更新缓存变量时，只需要与存储该变量的core联络4)使用称为“目录”的数据结构来存储每条cache line 的状态5)当一个变量被更新时，会查询目录，并且在其缓存中包含该变量的cache line 将失效 6)目录需要额外存储空间，但是更新缓存变量时，只需要与存储该变量的core 联络  解答如下：   1. y的值应该是x的初始值或者5。因为所有的core共享总线，总线上传输的任何信号都可以被连接到总线的core“看到”，当core0更新了x的值，之后，会向总线广播此信息，此时core0中的x已经标记为已修改（Modified）缓存段，已修改缓存段如果被丢弃或标记为失效，那么先要把它的内容回写到内存中——这和回写模式下常规的脏（dirty）段处理方式一样，所以当core1尝试执行y=x时，如果core0已经成功将x写回（write-back）了，y的值就是5，否则就是x的初始值。 2. 答：y等于x的初始值或者5。因为core0在执行赋值语句x=5之后，由于采用写回机制，并没有立即将x写入磁盘中，而且将主目录x=5中存储的这个cache line标记为dirty。但是由于主目录更新存在时延，有可能在主目录没有更新的时候core1就执行了y=x这个赋值语句，所以y可以等于x的初始值或者5。 3. 更新协议结合两者的优势。基于目录的Cache一致性协议。同时使用cache和目录结构。问题a可以采用窥探缓存一致性（snooping cache coherence）和直写（write through）来解决共享内存一致性问题，直写可以保证数据都被及时地写到内存中。问题b可以采用加入内存屏障的方法。   **问题2**  假设一个串行程序的运行时间为：Tserial = n²（单位为毫秒），这个程序的并行版本运行时间为：Tparallel = n²/p + log2(p)，编写一个程序获得在n和p为不同值时，并行版本的加速和效率（n = 10, 20, 40,……,320 和 p = 1, 2, 4, ……, 128）。当固定n，p增加时，加速和效率如何变化？当固定p，n增加时，加速和效率如何变化？  **解答**  回顾一下两个定义。   1. 加速：   令则  因此当固定时，当时候y随着p单调递增,当时候随着单调递减。即当时候随着单调递减,当时候随着单调递增;  显然，当固定时，随着的增大而增大且趋于   1. 效率：   显然，当固定的时候,随着的增大而减小。当固定的时候，随着的增大而增大 | | | |
| 教师评价 | | | |
|  | | | |