并行计算 HW1

PB18111697 王章瀚

2021 年 4 月 4 日

1.

改写求最大值问题的并行算法, 要求不使用数组 M.

不妨设 $A \in R^n$ 为待求最大值的数组, 则算法如下:

Algorithm 1 Parallel-Max

1: **for** i=1 to n **par- do** ▷ 初始化

2: $\quad B[i][1] = 1$

3: **for** i=1 to n **par- do** ▷ 运算

4: $\quad$ **for** j=1 to n **par- do**

5: $\quad\quad$ **if** proc-id == P_{ij} **then**

6: $\quad\quad\quad B[i][1] = \begin{cases} 0 & A_i < A_j \&\& B[i][0] == 1 \\ \text{空操作} & else \end{cases}$ ▷ **此处认为能够原子地竞争判断并写入**

7: **for** i=1 to n **par- do** ▷ 返回结果

8: $\quad$ **if** $B[i][1] == 1$ **then**

9: $\quad\quad$ **return** i

主要想法是仅使用 B 数组的第一列, 这里要求第六行能够**原子地竞争判断并写入**

2.

(课本5.6). 在 APRAM 模型上设计算法时, 应尽量使各处理器内的局部计算时间和读写时间大致于同步时间 B 相当. 当在 APRAM 上计算 n 个数的和时, 可以借用 B 叉树求和的办法. 假定有 p 个处理器计算 n 个数的和, 此时每个处理器上分配 n/p 个数, 各处理器先求出自身的局和, 然后从共享存储器中读取它的 B 个孩子的局和, 累加后置入指定的共享存储单元 SM 中, 最后根处理器所计算的和即为全和. 算法 5.4 示出了 APRAM 上的求和算法.

Algorithm 2 Parallel-Sum

Input: n 个待求和的数
Output: 总和在共享存储单元 SM 中
各处理器求 n/p 各数的局和, 并写入 SM 中
Barrier
for $k = \lceil \log_B(p(B-1)+1) \rceil - 2$ downto 0 **do**
    for all $P_i, 0 \leq i \leq p-1$ **do**
        if P_i 在第 k 级 **then**
            P_i 计算其 B 个孩子的局和并与自身局和相加, 然后将结果写入 SM 中
            Barrier

① 试用 APRAM 模型之参数, 写出算法的时间复杂度函数表达式
② 试解释 Barrier 语句的作用

①

下面按步骤给出时间复杂度, 然后求和. 其中全局读写时间为 d

(1). 各个处理器求 n/p 个数的局和, 并进行一次全局读写, 需要 $O(n/p + d)$

(2). Barrier 需要 $Barrier(p) \in O(d \log p)$

(3). 往后每级需要读取并计算 B 个孩子的局和, 共 $O(Bd)$, 而后同步需要 $O(d \log p)$共有 $\lceil \log_B(p(B-1)+1) \rceil - 1$ 级. 因此这一步总共是

$$O([\lceil \log_B(p(B-1)+1) \rceil - 1](Bd + d \log p))$$

所以总的时间复杂度应为

$$\begin{aligned} & O(n/p + d + d \log p + [\lceil \log_B(p(B-1)+1) \rceil - 1](B + \log p)d) \\ &= O(n/p + \lceil \log_B(p(B-1)+1) \rceil (B + \log p)d) \end{aligned}$$

②

经过 Barrier 同步之后才能确保每个处理器已经计算完局和, 这样 SM 中才有对应的局和值, 才能去使用.

3

①

按步骤如下:

- 显然, 这个算法有 $1+\lceil\log_d(p(d-1)+1)\rceil-1=\lceil\log_d(p(d-1)+1)\rceil$ 次通信开销,
- 而 (1) 处局和需要 $\Theta(n/p)$, 并且最底层每个处理器应上播 1 份数据, 然后需要一个路障同步, 需 $g+L$ 时间, 这里 g 是带宽倒数, L 是路障同步时间.
- (3) 中需要接收 d 个孩子消息, 并且求和, 发给父节点, 需要 $\Theta(d+g+L)$. 这样的操作经历了 $\lceil\log_d(p(d-1)+1)\rceil-1$ 次, 最后一次不用再发送

因此总共需要

$$\begin{aligned}&\Theta(n/p+g+L+[\lceil\log_d(p(d-1)+1)\rceil-1]\cdot(d+g+L))\\&\qquad=\Theta(n/p+[\lceil\log_d(p(d-1)+1)\rceil]\cdot(d+g+L))\end{aligned}$$

②

根据上述对时间复杂度的估计, 可以看出来, 在固定处理器数和消息发送和同步的时间的情况下, 可以近似最小化 $f(d)=n/p+g+L+[\lceil\log_d(p(d-1)+1)\rceil-1]\cdot(d+g+L)$ 来得到 d.
但若考虑到处理器数及网络情况等, 则最好应能够确保处理器得到充分利用且尽量避免网络拥挤.