并行程序设计第2次作业

林陈冉

2017年3月11日

分析　设 $x=\{x_1,x_2,\cdots,x_n\}, y=\{y_1,y_2,\cdots,y_n\}\in\mathbb{R}^n$, x 和 y 的内积为

$$\langle x,y\rangle=\sum_{k=1}^{n}x_ky_k$$

需要执行 n 次乘法和 $n-1$ 次加法.

这个过程中数据彼此依赖度很小, 改成并行计算时, 设有 p 个处理器, 可以把 x , y 拆成 p 个向量, 求出内积后再求和, 即

$$x^i=\{x_{f(i)},x_{f(i)+1},\cdots,x_{f(i+1)-1}\},\quad y^i=\{y_{f(i)},y_{f(i)+1},\cdots,y_{f(i+1)-1}\},\quad i\in\{1,2,\cdots,p\}$$

$$\langle x,y\rangle=\sum_{i=1}^{p}\langle x^i,y^i\rangle=\sum_{i=1}^{p}\sum_{k=f(i)}^{f(i+1)-1}x_ky_k$$

其中 f 是一个映射, 它把 $\{1,2,\cdots,n\}$ 给分成 p 份. 考虑到各处理器运算能力接近, 可以简单进行等分, 即

$$f(i)=\begin{cases}(i-1)\lceil\frac{n}{p}\rceil & ,i=1,2,\cdots,p\\ n & ,i=p+1\end{cases}$$

计算　串行程序运行用时

$$T_1=n+n-1=2n-1$$

当有 p 个处理器

$$T_p=2\lceil\frac{n}{p}\rceil-1+\log_2 p$$

那么加速比为

$$S_p=\frac{T_1}{T_p}=\frac{2n-1}{2\lceil\frac{n}{p}\rceil-1+\log_2 p}$$

效率

$$E_p=\frac{S_p}{p}=\frac{2n-1}{2p\lceil\frac{n}{p}\rceil-p+p\log_2 p}$$

现在我们想保证效率为定值 E , 记 $K=\frac{E}{1-E}$

$$T_0-\frac{T_1}{K}=0\quad\Rightarrow\quad F(p)=2p\lceil\frac{n}{p}\rceil-p+p\log_2 p-(1+K)(2n-1)=0$$

这个方程并不容易解, 先考虑 $p\lceil\frac{n}{p}\rceil$

$$n \le p\lceil\frac{n}{p}\rceil < n + p$$

那么可以得到估计 $F_{up}(p) = -p + p\log_2 p + 1 - (1+K)(2n-1)$, $F_{down}(p) = p + p\log_2 p + 1 - (1+K)(2n-1)$, 且有关系

$$F_{up}(p) \le F(p) < F_{down}(p)$$

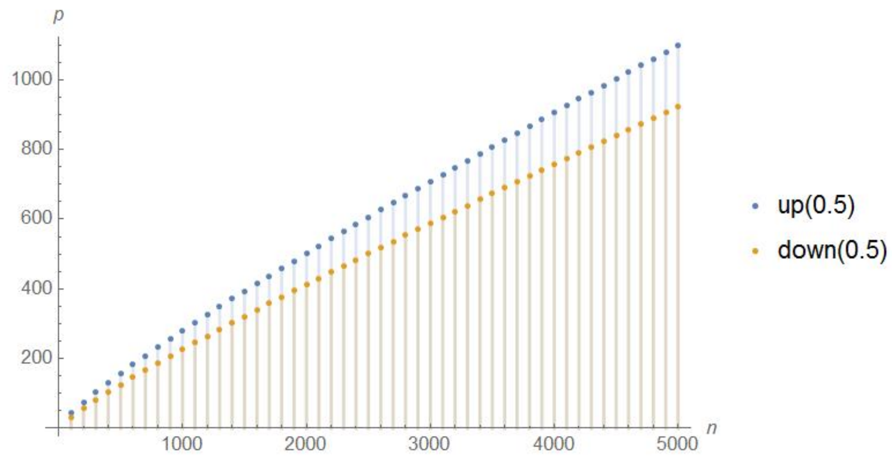

图 1: 固定效率0.5时$n-p$关系图

容易证明, 函数 $F_{up}(p)$, $F_{down}(p)$ 都是递减的. 那么令它们等于 0 就可以给出 p 的上下界. 图??即固定效率 $E_p = 0.5$ 时的估计, 散点曲线 $up(0.5)$ 是 $F_{up}(p) = 0$ 的解, 散点曲线 $down(0.5)$ 是 $F_{down}(p) = 0$ 的解. 可以看出, 存在明显的线性关系.

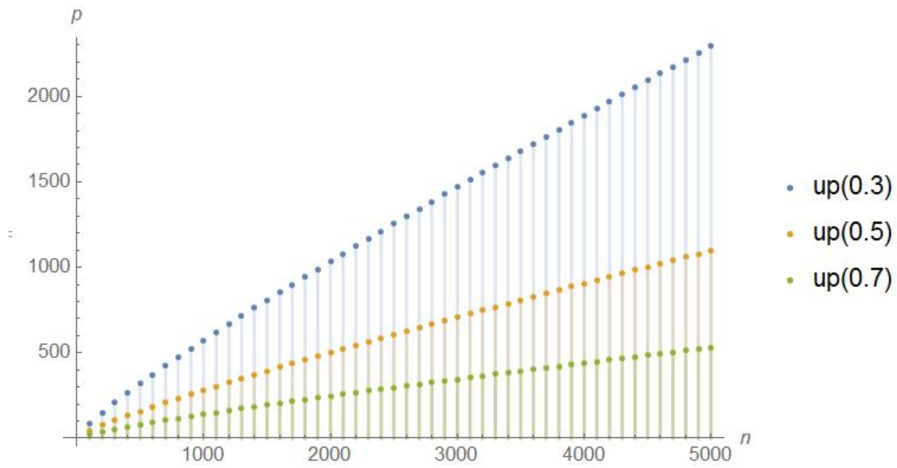

图 2: 不同效率时$n-p$关系图

图??显示了不同效率下的表现, 都取 p 的上估计(以下各图都是如此), 即 $F_{up}(p) = 0$ 的解, 散点曲线 $up(0.3)$ 表示效率 $E_p = 0.3$ 时的处理器个数 , 其他曲线同理. 可以看出, 若想保证效率, 需要用更少的处理器.

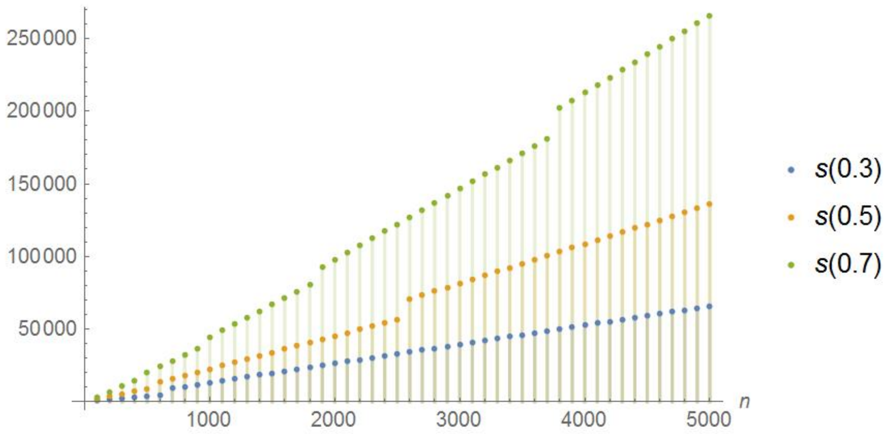

图 3: 不同效率时$n - S_p$关系图

图??显示了不同效率下的加速比, 散点曲线 $s(0.3)$ 表示效率 $E_p = 0.3$ 时的加速比，其他曲线同理. 可以看出, 保证效率少用处理器后, 计算速度变慢.

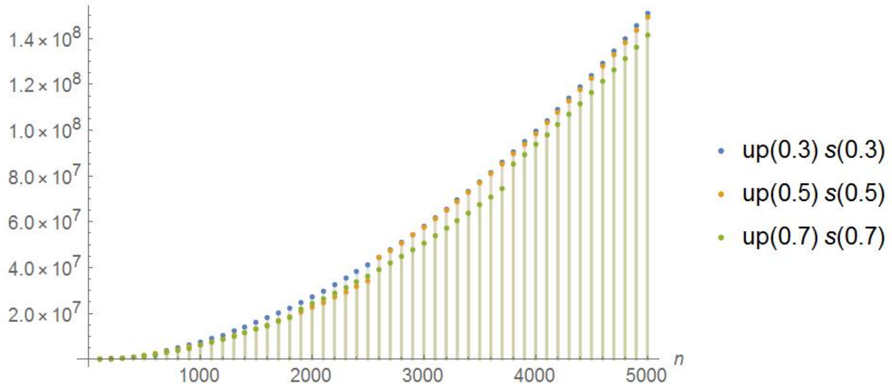

图 4: 不同效率时$n - p \times S_p$关系图

图??显示了不同效率下的加速比与处理器个数的乘积, 散点曲线 $up(0.3)s(0.3)$ 表示效率 $E_p = 0.3$ 时的加速比与处理器个数的乘积，其他曲线同理. 可以看出, 无论保证效率为多少, 加速比与处理器个数的乘积是不变的.