并行算法设计与 MPI 编程实例: 编号问题

张林波

2010 年 4 月 22 日

1. 问题描述

假设通信器 `MPI_COMM_WORLD` 中包含 p 个进程，每个进程中有一个长度为 `n`、取值为非负整数的数组 `obj[]`。进一步，假定同一进程中 `obj[]` 数组的分量值互不相等，但不同进程中的 `obj[]` 数组的分量值则可能相等。

在实际应用中，`obj[]` 代表一组分布存储在各个进程中的对象的编号，同一个对象可以同时存在于多个进程中，其在不同进程中的编号必须一样。例如，并行有限元程序中，`obj[]` 可以是顶点、边、面或单元的全局编号。

试设计解决下述计算问题的并行算法和 MPI 并行程序。

问题一: 统计数目。统计所有进程中所包含的不同对象数 `N`。

问题二: 指定属主。为每个对象指定一个唯一的属主进程 (owner process)。

问题三: 重新编号。重新对所有对象进行编号，编号范围为 `0`,...,`N-1`。

2. 算法

(1) 收集算法：收集所有对象到进程 0。
优点：实现简单。
缺点：负载不平衡，并且可能超出进程 0 的最大内存。

(2) 循环算法：将所有进程按进程号首尾相连排成一个有向环 (最后一个进程与首个进程相连)，让 `obj[]` 在进程环上依次轮转同时进行比较、处理。
缺点：通信量大，且需要 p 步才能完成。

(3) 分配算法：给定一个从 `obj[]` 的值域到 $\{0,\ldots,p-1\}$ 的映射 map，各进程将 `obj[i]` 发送到进程 map(`obj[i]`) 进行比较、处理。可以通过不同 map 的选择来优化通信和负载平衡。作为一个特例，如果取 $\mathrm{map}(i)\equiv 0$，则该算法等同于收集算法。

3. 程序实例

下述三个程序是算法三的 MPI 程序实例，其中取 $\mathrm{map}(i) = i\,\%\,p$。三个程序运行时均需要一个命令行参数给出 `n` 值，例如“`mpirun -np 3 numbering-count 10`”。

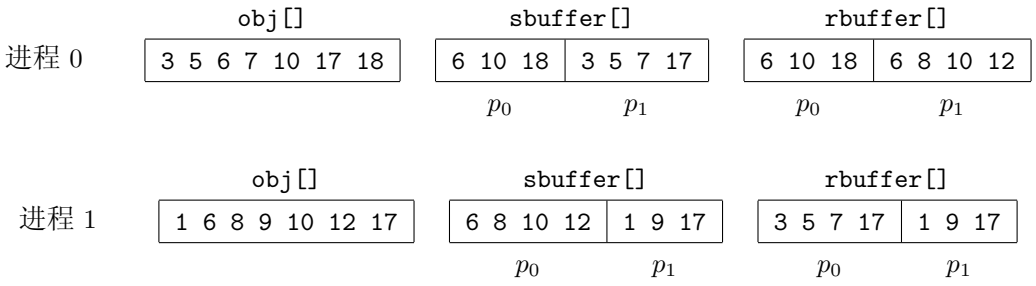

图 1 通信示例 (两个进程，$\mathrm{map}(i) = i\%p$)

程序一: 统计对象数。

`ftp://ftp.cc.ac.cn/pub/home/zlb/PCcourse/numbering-count.c`

程序二: 指定属主并统计对象数。

`ftp://ftp.cc.ac.cn/pub/home/zlb/PCcourse/numbering-owner.c`

程序三: 指定属主、统计对象数并对对象重新编号。

`ftp://ftp.cc.ac.cn/pub/home/zlb/PCcourse/numbering-renumber.c`

4. 思考

这里所介绍的分配算法中的通信一般情况下是“稠密”的，即每个进程需要与所有其它进程通信，通信复杂度为 $O(p)$，通信量为 $O(n)$。但对一些特殊情况，通过精心选取 map 函数，可以使得通信是“稀疏”的，即每个进程只与 $O(1)$ 个其它进程间有通信，并且通信量 $\ll n$。

例如：在并行有限元计算中，可以将有限元网格按单元划分为子网格，每个进程存储一个子网格。假设 `obj[]` 数组存储子网格中顶点的全局编号，其中，每个顶点可以存在于一至多个进程中，并且这些进程中的一个被指定为其属主。如果在自适应网格局部粗化过程中删除了部分顶点，需要统计剩下的顶点数目并且对它们重新进行编号，则可以采用这里介绍的分配算法，并取 $\mathrm{map}(i)$ 为顶点 i 的属主进程号，只要子网格的划分是合理的，则通信复杂度通常为 $O(1)$，通信量通常 $\ll n$。

5. 习题

(1) 试用循环算法解决问题二，并比较计算效率。

(2) 当如果去掉“同一进程中 `obj[]` 数组的分量值互不相等”的条件，程序一、程序二和程序三各需做哪些改动?

(3) 改进属主选择策略，以求达到更好的负载平衡效果 (属于各进程的对象数尽量相等)。

(4) 假设 `obj[]` 数组的值域为 $\{0, \ldots, M-1\}$。试改变程序三中 map 的定义，使得重新编号后新编号保持老编号的顺序。