第20卷 第7期 计 算 机 学 报
1997年7月 CHINESE J. COMPUTERS

一类递推方程的一个新的优化向量并行算法

胡 玥 高庆狮
(北京科技大学智能、语言与计算机科学研究所 北京 100083)
刘志勇
(国家自然科学基金委员会 北京 100083)

摘 要 本文提出一种新的解 Kogge 和 Stone 所定义的一类递推方程的优化的并行算法，当采用 p 台处理机，对规模为 N 的一类递推方程求解时，该算法的加速比为 $O(p)$，其中 $1\leq p\leq N^{1-\varepsilon}$，$\varepsilon$ 是一个任意小的正数. 与已有的并行算法相比，该算法具有效率高，适用范围广的优点. 该算法可以在 EREW PRAM 模型机上实现，也可以在具有素数内存系统的流水线向量处理机上实现.

关键词 并行算法，优化算法，一阶线性递推方程，递推倍增算法.
分类号：TP301

A NEW OPTIMAL VECTOR PARALLEL ALGORITHM FOR THE SOLUTION OF A GENERAL CLASS OF RECURRENCE EQUATIONS

HU Yue GAO Qingshi
(*Institute of Intelligence, Language and Computer Science, Beijing University of Science and Technology, Beijing* 100083)
LIU Zhiyong
(*National Natural Science Foundation of China, Beijing* 100083)

Abstract This paper proposes a new optimal vector parallel algorithm for the solution of a general class of recurrence equations defined by Kogge and Stone. The speedup of this algorithm in $O(p)$, when $1\leq p\leq N^{1-\varepsilon}$, where p is the number of processors, and ε is a given positive constant. Compared with previous algorithms, it has higher efficiency and the range of p is wide. This algorithm can be implemented by EREW PRAM machine or pipeline vector computers with prime memory system.

Keywords Parallel algorithm, optimal algorithm, first-order linear recurrence equation, recursive doubling algorithm.

本文1995-11-29收到，修改文1997-01-27收到. 本课题得到国家863高科技基金及国家自然科学基金的资助.胡 玥，讲师，获硕士学位，从事并行算法和机器翻译研究工作. 高庆狮，中国科学院院士，从事体系结构、并行算法、自然语言和智能研究工作. 刘志勇，研究员，获博士学位，从事并行处理、人工智能研究工作.

1 引　言

Kogge 和 Stone 在文献[1]中给出了一类递推方程一般形式的如下定义:

$$\begin{cases} x_0 = b_0 \\ x_i = f(b_i,\ g(a_i,\ x_{i-1})) \qquad i = 1,2,\cdots,N-1 \end{cases}$$

且函数 f,g 满足下列条件:

(1) $f(x,f(y,z)) = f(f(x,y),z)$,

(2) $g(x,f(y,z)) = f(g(x,y),g(x,z))$,

(3) 存在一个函数 $h(x,y)$,使得 $g(x,g(y,z)) = g(h(x,y),z)$.

其中,a_i,b_i 为常数,$i = 0,1,2,\cdots,N-1$.

例如:

$$\begin{cases} x_0 = b_0 \\ x_i = b_i x_{i-1}^{a_i}, \quad i = 1,2,\cdots,N-1 \end{cases}$$

$$\begin{cases} \vec{x}_0 = \vec{b}_0 \\ \vec{x}_i = A\ \vec{x}_{i-1} \end{cases}$$

其中:A 为数组,$\vec{x}_i$,$\vec{b}_0$ 为向量.

该类方程在科学与工程计算中具有广泛的应用,得到了广泛的研究.二十多年来,人们提出了许多解决线性递推方程的并行算法[1~15],其中有一些是对处理机台数有限制的,还有一些是对处理机台数不加限制的.

Kogge 和 Stone[1] 提出一种递推倍增(Recursive Doubling)并行算法(RD)来计算上述定义的一类递推方程.当处理机台数 p 等于数据总数 N 或大于 $N/2$ 时,该算法是速度最快的算法.其计算时间为 $O(\log_2 N)$,加速比为 $O(p/\log_2 N)$,效率为 $O(p/\log_2 N)/p = O(1/\log_2 N)$.

Gao[3] 提出了一种纵-横(V-H)并行算法来计算与文献[1]相同的一类递推方程.当 $N = p$ 时,其计算时间 $T = 5(p-1)$;当 $N = mn$ 且 $m,n < p$ 时(其中,m,n 是正整数),$T = 3(n-1) + 2(m-1)$;当 $N = C(p-1)+1$ 时,$T = 5C(p-1)$.其中,C 是任意的正整数.它是一个优化并行算法,其加速比为 $O(p)$,效率为 $O(p)/p = O(1)$,也就是效率为常数.但是,其适用范围为 $1 \leqslant p \leqslant O(N^{\frac{1}{2}})$.

Chen,Kuck 和 Sameh[7] 提出了一个解决带状线性方程组的并行算法,T 大约等于 $(2m^2 n/p) + (3mn/p) + (m^2+m+1)\log(p/4m)$.在 $m=1$ 情况下,T 大约等于 $(5n/p) + 3\log_2 p - 6$.

在 Gajski[9] 的工作基础上,Mayer 和 Podrazik[14] 提出一个一阶线性递推方程(文献[1]所定义的一类递推方程的一个特例)的并行算法,当 $(N-1)/(p-1)$ 是整数时,T 为 $5(N-1)/(p+1)$.

Gao 和 Liu 在文献[15]中提出一种优化的并行算法.使用 p 台处理机,该算法可在 $O(N/p)$ 步内求解此类递推方程,在广泛的范围内($p < N \leqslant p^2$),该算法的加速比可为 $O(p)$,效率可为 $O(1)$.

在本文中我们提出一种新的优化并行算法(OVP).该算法是建立在 V-H 算法[3] 和 RD 算法[1] 基础上的一种优化并行算法.其适用范围为 $1 \leqslant p \leqslant N^{1-\varepsilon}$($\varepsilon$ 为大于 0 的任意正数).其实际计算速度比并行算法(KDOP)[15] 快.

本算法能在 EREM PEAM 模型机上实现. 也能在实际的 SIMD 机(如一般的具有重叠传输的内存系统的流水线向量机)上实现. 下面我们将给出关于该递推问题的定理和算法(它基于 V-H 算法和 RD 算法),最后对算法的效率和实现进行讨论.

2 一类递推方程的有效解法定理

在文献[15]中,我们给出一个该类递推方程有效解法的定理. 该定理保障此类方程递推求解方法的正确性. 本文将提出的算法也要使用该定理,本节我们先引述该定理,其证明可参阅文献[15].

设一类递推方程一般形式定义如下:

$$\begin{cases} x_0 = b_0 \\ x_i = f(b_i, g(a_i, x_{i-1})), \quad i = 1,2,\cdots,N-1 \end{cases}$$

且函数 f,g 满足下列条件:

(1) $f(x,f(y,z)) = f(f(x,y),z)$,

(2) $g(x,f(y,z)) = f(g(x,y),g(x,z))$,

(3) 存在一个函数 $h(x,y)$ 使得 $g(x,g(y,z)) = g(h(x,y),z)$ 且 $h(x,h(y,z)) = h(h(x,y),z)$.

其中,$a_i\{i = 1,2,\cdots,N-1\}$,$b_i\{i = 0,1,\cdots,N-1\}$ 为常数,则有如下定理成立.

定理 1.

设 $P(i,i) = a_i$, $i = 1,2,\cdots,N-1$,

$P(i,i+j) = h(a_{i+j},P(i,i+j-1))$, $i = 1,2,\cdots,N-1$; $j = 0,1,\cdots,N-1$,

$P(0,0) = I$, I 为 h 的幺元素,

$Q(i,i) = b_i$, $i = 0,1,\cdots,N-1$,

$Q(i,i+j) = f(b_{i+j}, g(a_{i+j}, Q(i,i+j-1)))$, $i,j = 0,1,\cdots,N-1$,

且 $h(x,I) = x$,

则(1) $P(i,j) = h(P(k,j),P(i,k-1))$, $0 < i \leqslant k \leqslant j \leqslant N-1$

(2) $Q(i,j) = f(Q(k,j),g(P(k,j),Q(i,k-1)))$, $0 \leqslant i \leqslant k \leqslant j \leqslant N-1$

(3) $x_i = Q(0,i)$, $i = 0,1,\cdots,N-1$

3 一类递推方程的优化向量并行算法(OVP)

本节给出一个解该类递推方程的并行方法. 该算法综合运用递推倍增法(RD)的思想和纵横加工法(V-H)的思想.

在下述 OVP 算法中,我们假设处理机台数为 p. 递推方程规模为 N,而 $N = mk^2$,$p = N/k$,且设 p 为 2 的完全方次($p = 2^L$),$m = k^r$, 其中,r,k,L 均为正整数.

算法共分三步:将任务分为 mk 段,每段的尺寸为 k.

(1) mk 段同时串行计算,求得各段的相对值.

(2) 对(1)求得的各段的相对端点值,应用递推倍增法(RD),求得各段的最终端点值.

(3) 各段利用(2)中求得的上段的端点值作为初始值同时串行计算,求得最终结果.

具体算法如下:

OVP 算法.

```
Begin
  1. for t := 1 to k − 1 do
       for i := 0 to p − 1 do
       begin
         P(ik,ik + t) := h(P(ik + t,ik + t),P(ik,ik + t − 1))
         Q(ik,ik + t) := f(Q(ik + t,ik + t),g(P(ik + t,ik + t),Q(ik,ik + t − 1)))
       end          /...i 为并行,t 为串行 /
  2. for t := 1 to L do
       for i := 0 to 2^(L−t) − 1 do
         for j := 2^(t−1) to 2^t − 1 do
         begin
           P(i2^t k,(i2^t + j + 1)k − 1) :=
                h(P((i2^t + 2^(t−1))k,(i2^t + j + 1)k − 1),P(i2^t k,(i2^t + 2^(t−1))k − 1))
           Q(i2^t k,(i2^t + j + 1)k − 1) := f(Q((i2^t + 2^(t−1))k,(i2^t + j + 1)k − 1),
                g(P((i2^t + 2^(t−1))k,(i2^t + j + 1)k − 1),Q(i2^t k,(i2^t + 2^(t−1))k − 1)))
           end          /...i 和 j 一起为并行,t 为串行 /
  3. for t := 0 to k − 2 do
       for i := 1 to p do
         Q(0,ik + t) = f(Q(ik − t,ik + t),g(P(ik + t,ik + t),Q(0,ik + t − 1)))
                       /...i 为并行,t 为串行 /
End
```

由于第一节中所定义的递推方程是对一大类递推方程的形式描述[1,15],所以计算函数 f, g,h 所用的时间是随着具体应用问题有所不同的(通常为加、乘、取方次等运算). 为形式上的分析方便起见,这里我们假设 f,g,h 的执行时间均为一个计算步. 算法中第一个和第三个 for 语句可在 $5(k-1)$ 步完成,而第二个 for 语句可在 $3L = 3\log_2 p$ 步完成. 从而该算法可在 $5(k-1) + 3\log_2 p$ 步内完成. 其时间复杂度为 $O(k) = O(N/p)$.

4　比较与应用

1. 几种方法的数量级对比(见表 1)

设 $N = n^k$,其中,k 是任意正整常数,n 是正整数.

表 1

	串行(S)	倍增(RD)	纵横(V-H)	K 维(KDOP)	OVP
p	1	$O(n^k)$	$O(n^{k/2})$	$O(n^{k-1})$	$O(n^{k-1})$
T_p	$O(n^k)$	$O(\log_2 n)$	$O(n^{k/2})$	$O(n)$	$O(n)$
S_p	1	$O(n^k/\log_2 n)$	$O(n^{k/2})$	$O(n^{k-1})$	$O(n^{k-1})$
E_p	1	$O(1/\log_2 n)$	$O(1)$	$O(1)$	$O(1)$

其中,T_p 为时间,p 为台数,S_p 为加速比,E_p 为效率

2. OVP 算法与 KDOP 算法具体运行效率比较

OVP 算法与KDOP 算法是不同的算法，虽然它们的效率 E_p 是同数量级的，但是它们的实际运行效率不同. 具体比较如下：

设 $N=n^c, p=n^{c-1}$，其中，c 是任意正整常数，n 是正整数. OVP的运行时间为 T_1，KDOP 的运行时间为 T_2，f,g 的计算时间均按一个计算单位计算，则：

$$T_1=5(n-1)+3(c-1)\log_2 n \tag{1}$$

$$T_2=3c(n-1) \tag{2}$$

由式(1)，(2) 可得 $T_2>T_1$ 的条件为：

$$(k\geqslant 3)\ \&\ (n>4)\ \text{或}\ (k>3)\ \&\ (n\geqslant 4)$$

因为 $T_2>T_1 \Leftrightarrow 3cn-3c>5n+3(c-1)\log_2 n-5$

$$\Leftrightarrow (c-2)n+(1/3)(n-1)>(c-1)\log_2 n+c-2$$

$$\Leftrightarrow (c-2)(n-1)+(1/3)(n-1)>(c-1)\log_2 n$$

从而可以得出：

OVP 算法与KDOP 算法是不同的算法，虽然它们的效率 E_p 数量级相同，但在 $(k>3\ \&$ $(n\geqslant 4)$ 或 $(k\geqslant 3)\ \&\ (n>4)$ 的条件下 OVP 算法的速度比 KDOP 算法的速度要快.

3. 应用

关于这一类递推方程的应用问题在文献[1]中已经讨论. 例如：

(1) $g(x,y)=xy,\ f(x,y)=x+y,\ h(xy)=xy$

$$\begin{cases} x_0=b_0 \\ x_i=b_i+a_i x_{i-1}, \quad i=1,2,\cdots,N-1 \end{cases}$$

(2) $g(x,y)=y^x,\ f(x,y)=xy,\ h(x,y)=xy$

$$\begin{cases} x_0=b_0 \\ x_i=b_i x_{i-1}^{a_i}, \quad i=1,2,\cdots,N-1 \end{cases}$$

(3) $g(A,\vec{y})=A\vec{y}, f(\vec{y},z)=z,\ h(A,B)=AB$

$$\begin{cases} \vec{x}_0=\vec{b}_0 \\ \vec{x}_i=A\vec{x}_{i-1},\ i=1,2,\cdots,N-1 \end{cases}$$

其中：A,B 为矩阵，$\vec{x}_i,\vec{b},\vec{y},\vec{z}$ 为向量，$i=1,2,\cdots,N-1$.

5 结 论

本文给出一种解一类递推方程的一个新的优化向量并行算法，该算法综合运用递推倍增法(RD) 的思想和纵横加工法(V-H) 的思想，在计算的不同阶段对任务进行不同的划分，从而对有数据依赖的任务进行并行处理. 该算法的加速比为 $O(p)$，(其中 $1\leqslant p\leqslant N^{1-\varepsilon}, \varepsilon=$ $(r+2)$ 可以是一个任意小的正数). 与已有的并行算法相比，该算法具有效率高，适用范围广的优点. 该算法可以在 EREW PRAM 模型机上实现，也可以在具有素数内存系统的流水线向量处理机上实现.

参 考 文 献

1 Kogge P M, Stone H S. A parallel algorithm for the efficient solution of a general class of recurrence equation. IEEE

Trans Computers, 1973, C-22(8):786－792

2 Kogge P M. Parallel solution of recurrence problems. *IBM J Res & Dev*, 1974,18(1):138－148

3 Gao Q S. Another parallel algorithm for the efficient solution of a class of recursive computation. *Computer Application and Applied Mathematics*,1974,1(8):11－15

4 Chen S C, Kuck D J. Time and parallel processor bounds for linear recurrence system. *IEEE Trans Computers*, 1975, C-24(7):701－717

5 Kuck D J. Parallel processing of ordinary programs. *Advances in Computers*, 1976, 15:119－179

6 Sameh A H, Brent R P. Solving triangular systems on a parallel computer. *SIAM J Number Anal*, 1977,14(6):1101－1113

7 Chen S C, Kuck D J, Sameh A H. Practical parallel band triangular system solvers. *ACM Trans Math Software*, 1978, 4(3):270－277

8 Ladner R E, Fischer M J. Parallel prefix computation. *J ACM*,1980, 27(4):831－838

9 Gajski D D. An algorithm for solving linear recurrence systems on parallel and pelined machines. *IEEE Trans Computers*, 1981, C-30(5):190－206

10 高庆狮. 向量巨型机. 北京:中国出版社,1984

11 Carlson D A, Sugla B. Time and processor efficient parallel algorithms for recurrence equations and elated problems. In: Proc International Conference on Parallel Processing, 1984, 310－314

12 Kruskal C P, Rudolph L, Snir M. The power of parallel prefix. *IEEE Trans Computers*,1985,C-34(10):965－968

13 Snir M. Depth-size trade-offs for parallel prefix computation. *J Algorithms*,1986, 7(2):185－201

14 Meyer G G L, Podrazik L J. A parallel first-order linear recurrence solver. *J Parallel and Distributed Computing*, 1987, 4(3):117－132

15 Gao Q S,Liu Z Y. K-dimensional optimal parallel algorithm for the solution of a general class of recurrence equations. *J Computer Science and Technology*,1995, 10(5):417－424