PRAM 和 LARPBS 模型上的近似串匹配并行算法*

钟 诚+, 陈国良

(中国科学技术大学 计算机科学与技术系,安徽 合肥 230027)
(国家高性能计算中心,安徽 合肥 230027)

Parallel Algorithms for Approximate String Matching on PRAM and LARPBS

ZHONG Cheng+, CHEN Guo-Liang

(Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)
(National High-Performance Computing Center, Hefei 230027, China)
+ Corresponding author: Phn: +86-771-3272685, E-mail: chzhong6517@sina.com, ciugxcn@163.com, http://www.ustc.edu.cn
Received 2002-12-05; Accepted 2003-09-26

Zhong C, Chen GL. Parallel algorithms for approximate string matching on PRAM and LARPBS. *Journal of Software*, 2004,15(2):159~169.
http://www.jos.org.cn/1000-9825/15/159.htm

Abstract: Approximate string matching technique has been widely applied to many fields such as network information retrieval, digital library, pattern recognition, text mining, IP routing searching, network intrusion detection, bioinformatics, and computing in musicology. The two concise parallel dynamic programming algorithms for approximate string matching with k-differences on CREW-PRAM (parallel random access machine with concurrent read and exclusive write) are presented by parallel computing the edition distance matrix D in the wave-front approach and by parallel computing the edition distance matrix D along the diagonal and horizontal directions, respectively. The first algorithm requires $O(n)$ time and obtains a linear speedup by $(m+1)$ processors. The second algorithm needs $O(n/\alpha+m)$ time in use of $\alpha(m+1)$ processors, where $1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$. The simulation experiment shows that both of the algorithms achieve a good speedup. Based on the divide-and-conquer strategy and split-merge approach, the two communication-efficient and scalable parallel algorithms for approximate string matching with k-mismatches on linear arrays with reconfigurable pipelined bus system (LARPBS) are presented by reconfiguring dynamically the optical bus system, applying neatly the optical message-passing technique, and computing in parallel the prefix sums. The first algorithm requires $O(m)$ time by n processors and the second one has $O(1)$ time in use of nm processors.

Key words: approximate string matching; parallel algorithm; CREW-PRAM (parallel random access machine with

* Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1998030403 (国家重点基础研究发展规划(973)); the National High-Tech Research and Development Plan of China under Grant No.2001AA111041 (国家高技术研究发展计划(863))

作者简介: 钟诚(1964－),男,广西桂平人,博士,教授,主要研究领域为并行算法,网络信息安全,生物信息处理算法;陈国良(1938－),男,教授,博士生导师,中国科学院院士,主要研究领域为并行算法,并行计算.

concurrent read and exclusive write); reconfigurable optical bus system; edition distance; Hamming distance

摘 要: 近似串匹配技术在网络信息搜索、数字图书馆、模式识别、文本挖掘、IP 路由查找、网络入侵检测、生物信息学、音乐研究计算等领域具有广泛的应用.基于 CREW-PRAM(parallel random access machine with concurrent read and exclusive write)模型,采用波前式并行推进的方法直接计算编辑距离矩阵 D,设计了一个允许 k-差别的近似串匹配动态规划并行算法,该算法使用$(m+1)$个处理器,时间复杂度为 $O(n)$,算法理论上达到线性加速;采取水平和斜向双并行计算编辑距离矩阵 D 的方法,设计了一个使用$\alpha(m+1)$个处理器和 $O(n/\alpha+m)$时间的、可伸缩的、允许 k-差别的近似串匹配动态规划并行算法,$1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$.基于分治策略,通过灵活拆分总线和合并子总线动态重构光总线系统,并充分利用光总线的消息播送技术和并行计算前缀和的方法,实现了汉明距离的并行计算,设计了两个基于 LARPBS(linear arrays with reconfigurable pipelined bus system)模型的通信高效、可扩放的允许 k-误配的近似串匹配并行算法,其中一个算法使用 n 个处理器,时间为 $O(m)$;另一个为常数时间算法,使用 mn 个处理器.

关键词: 近似串匹配;并行算法;CREW-PRAM(parallel random access machine with concurrent read and exclusive write);可重构光总线系统;编辑距离;汉明距离

中图法分类号: TP301 **文献标识码**: A

近似串匹配问题分为允许 k-差别的近似串匹配和允许 k-误配的近似串匹配两种,前者通过计算模式与正文子串之间的编辑距离来实现,后者采取计算模式与正文子串之间的汉明距离来完成.Navarro 于 2001 年发表了一篇优秀的综述文章,对目前取得的顺序的近似串匹配算法的研究成果进行了全面的评述[1].

关于允许 k-差别的近似串匹配并行算法的研究,Navarro 和 Baeza-Yates 在文献[2]中指出,由于编辑距离矩阵当前元素 D_{ij} 值的计算依赖于其前驱元素 $D_{i-1,j-1}$,$D_{i-1,j}$ 和 $D_{i,j-1}$ 的值,所以人们认为编辑距离矩阵 D 元素值的计算难以直接并行化.文献[3]指出,Landau 和 Vishkin 发现,由 Ukkonen 定义的与编辑距离矩阵 D 等效的矩阵 L 的元素的计算可以并行化.基于 CRCW-PRAM(parallel random access machine with concurrent read and exclusive write)模型,Landau 和 Vishkin 使用 $O(n)$处理器和 $O(k+\log m)$时间,设计了一个离线的允许 k-差别的近似串匹配动态规划并行算法,其中 n 为正文长度,m 为模式长度,$m<n$,k 为任意给定的正整数,$0\le k<m$.Landau 和 Vishkin 的并行算法需要使用 $O(n)$处理器、花费 $O(\log n)$时间对正文进行预处理以建立后缀树,这一预处理过程十分复杂,执行代价 $O(n\log n)$较大,而且后缀树实际占用存储空间为正文规模 n 的 12~70 倍[2].1993 年,Jiang 和 Wright 仍使用矩阵 L,设计了一个运行在 CRCW-PRAM 模型上、时间复杂度为 $O(k)$、允许 k-差别的近似串匹配动态规划并行算法[4].该算法虽然无须预处理正文,但使用了 $O(nm)$规模的处理器,因此,算法的并行执行代价 $O(knm)$太大.1997 年,Lee 和 Ercal 在可重构网孔机器上设计了基于矩阵 L 的允许 k-差别的近似串匹配动态规划并行算法.此算法虽然获得了 $O(k)$的执行时间,但是也使用了规模为 $O(nm)$的处理器,其执行代价仍然是较大的 $O(knm)$[3].

对于允许 k-误配的近似串匹配并行算法,文献[5]在假定机器字长为 $\log n$ 的前提下,基于 CRCW-PRAM 模型,使用 n 个处理器和 $O(\log n)$时间,设计了一个允许 k-误配的近似串匹配并行算法.因为 n 的值一般都很大,所以此算法的执行代价 $O(n\log n)$较大,当 $\log n>m$ 时,执行代价大于 $O(nm)$.文献[6]通过构造一种余数系统,在二维可重构网孔机器上设计了一个使用 $O(m\log|\Sigma|)\times O(n(\log|\Sigma|+\log^2 k/\log\log k))$处理器和 $O(1)$时间的允许 k-误配的近似串匹配并行算法,该算法的执行代价也大于 $O(mn)$.文献[3]在假定$(n-m+1)\ge m\ge 16$ 的前提下,在处理器规模为$(n-m+1)\times m\times(n-m+1)$的三维可重构网孔机器上设计了一个 $O(1)$时间的允许 k-误配的近似串匹配并行算法,该算法的代价 $O(mn^2)$很大.文献[7]则利用数据流并行机器的思想,在脉动阵列机器上设计了一个允许 k-误配的近似串匹配并行处理硬件结构,此硬件算法使用 $O(dm)$处理器、花费 $O(n/d+\alpha)$时间,执行代价为 $O(nm+\alpha dm)$,正整数 d 表示流水线并行度,$0\le\alpha<m$.

本文第 1 节采用波前式并行推进的方法直接计算编辑距离矩阵 D,设计一个基于 CREW-PRAM 模型的允许 k-差别的近似串匹配动态规划并行算法;然后基于水平和斜向双并行直接计算编辑距离矩阵 D 的方法,设计

一个可伸缩的、并行度更高的允许 k-差别的近似串匹配动态规划算法;最后,给出这两个算法的仿真实验结果.第 2 节基于分治策略,通过灵活拆分总线和合并子总线动态重构光总线系统,并充分利用光总线的消息播送技术和并行计算前缀和的方法,实现汉明距离的并行计算,设计两个基于 LARPBS(linear arrays with reconfigurable pipelined bus system)模型的通信高效、可扩放的允许 k-误配的近似串匹配并行算法.第 3 节对全文进行总结.

1 CREW-PRAM 模型上的允许 k-差别的近似串匹配并行算法

1.1 编辑距离与允许k-差别的近似串匹配问题

定义 1. 任意给定两个串 X 和 Y,那么串 X 和 Y 之间的编辑距离 $D(X,Y)$定义为使用如下 3 种编辑操作将串 X 转换成串 Y 所需的最少的编辑操作次数:① 从串 X(或者 Y)中删除一个符号(字符);② 向串 X(或者 Y)插入一个符号;③ 用另一个符号替换串 X(或者 Y)中指定的某个符号.

性质 1. 两个串 X 和 Y 之间的编辑距离 $D(X,Y)$具有如下性质:① 非负性:$D(X,Y)\geq 0$,$D(X,Y)=0$ 当且仅当 $X=Y$;② 对称性:$D(X,Y)=D(Y,X)$;③ 三角不等式:设 Z 为另一个串,则 $D(X,Y)\leq D(X,Z)+D(Z,Y)$.

定义 2. 对于任意给定的一个长度为 n 的串 $X[1,\ldots,n]$,称 $X[i,\ldots,j]$为串 X 的子串,特别地,子串 $X[1,\ldots,i]$称为串 X 的前缀,而子串 $X[j,\ldots,n]$称为串 X 的后缀,其中 $1\leq i,j\leq n$.

定义 3. 所谓允许 k-差别的近似串匹配问题是指,对于任意给定的一个长度为 m 的称为模式的串 $P[1,\ldots,m]$和一个长度为 n 的称为正文的串 $T[1,\ldots,n]$,$m<n$,并任意给定一个正整数 k,$0\leq k<m$,寻找出编辑距离小于等于 k 的模式 P 在正文 T 中所有匹配出现的终止位置 j,$1\leq j\leq n$.

假设构成模式串和正文串的可能的符号(字符)均来自有限字典表 Σ,字典表的大小记为 $\sigma=|\Sigma|$,Σ^* 表示由 Σ 中的符号所构成的任意一个有限长度的串,而 Σ^l 表示由 Σ 中的符号所构成的任意一个长度为 l 的串,$l\geq 0$.这样,允许 k-差别的近似串匹配问题就可以由下面定义的函数 f 来描述,并通过计算函数 f 的值来获得近似串匹配的解:

$$f:\Sigma^m\times\Sigma^n\times k\rightarrow\{0,1\}^{n-m+k+1},$$

其中 $f(P,T,k)=c_{m-k}c_{m-k+1}\ldots c_n$,对于 $m-k\leq j\leq n$ 且 $1\leq i\leq j$,有

$$c_j=\begin{cases}1, & \text{若} D(P,T[i,\ldots,j])\leq k\\ 0, & \text{否则}\end{cases}.$$

对于任意给定的模式 $P[1,\ldots,m]$和正文 $T[1,\ldots,n]$,$m<n$,以及任意给定的正整数 k,$0\leq k<m$,如何寻找出允许 k-差别的模式 P 在正文 T 中所有匹配出现的终止位置 j 呢?基于动态规划算法的思想是,构造一个规模为 $(m+1)\times(n+1)$的编辑距离矩阵 D 来实现允许 k-差别的近似串匹配.编辑距离矩阵 D 的计算满足下面的递推方程:

$$D[i,j]=\begin{cases}0, & i=0\\ i, & j=0\\ \min\{D[i,j-1]+1,D[i-1,j]+1,D[i-1,j-1]+c\}, & i\neq 0\text{且}j\neq 0\end{cases}\tag{1}$$

其中

$$c=\begin{cases}0, & \text{若} P[i]=T[j]\\ 1, & \text{否则}\end{cases}.$$

$D[i,j]$表示将模式前缀 $P[1,\ldots,i]$转换成正文子串 $T[l,\ldots,j]$($1\leq l\leq j$)所需的编辑操作次数.允许 k-差别的近似串匹配动态规划顺序算法的时间复杂度为 $O(nm)$,空间复杂度为 $n+3m+4$.

1.2 波前式并行计算编辑距离矩阵D的允许k-差别的近似串匹配动态规划算法

设 CREW-PRAM 并行系统共有 $m+1$ 个处理器,分别记为 PR_0~PR_m,处理器之间通过访问共享存储器进行通信.模式和正文均存储在并行系统的共享存储器中.

从第 1.1 节可知,由于 $D[i,j]$的值严格依赖于 $D[i-1,j-1]$,$D[i-1,j]$和 $D[i,j-1]$的值,所以 $D[i,j]$的值的计算呈顺序性.如何开拓 $D[i,j]$计算的并行性呢?让我们从整体的角度考查如图 1 所示的编辑距离矩阵 D 的计算过程.

对于图 1 中第 l 条“反对角线”上的任意一个矩阵元素 $D[i,j]$,当计算 $D[i,j]$的值时,如果它需要引用的 $D[i-1,j-1]$,$D[i-1,j]$和 $D[i,j-1]$的值已知,那么可以同时计算此条“反对角线”上所有矩阵元素 $D[i,j]$的值.显然,开始时

第 1 条“反对角线”上有一个矩阵元素初值 $D[0,0]$,第 2 条“反对角线”上有两个矩阵元素初值 $D[1,0]$和 $D[0,1]$.因此,计算第 3 条“反对角线”上的任意一个矩阵元素 $D[i,j]$所需引用的 $D[i-1,j-1]$,$D[i-1,j]$和 $D[i,j-1]$的值全部已知,可以并行地计算此条“反对角线”上的所有矩阵元素 $D[i,j]$的值.依此类推,沿着矩阵的右下角方向以波前式并行推进的方法计算编辑距离.当正在计算第 l 条“反对角线”上的所有矩阵元素 $D[i,j]$的值时,由于第 $l-2$ 条和第 $l-1$ 条“反对角线”上的矩阵元素的值已知,所以第 l 条“反对角线”上的所有矩阵元素 $D[i,j]$的值可以直接并行计算.当并行计算出最后一条“反对角线”上所有矩阵元素的值时,长度为 m 的模式和长度为 n 的正文之间的所有编辑距离的值已全部计算好.

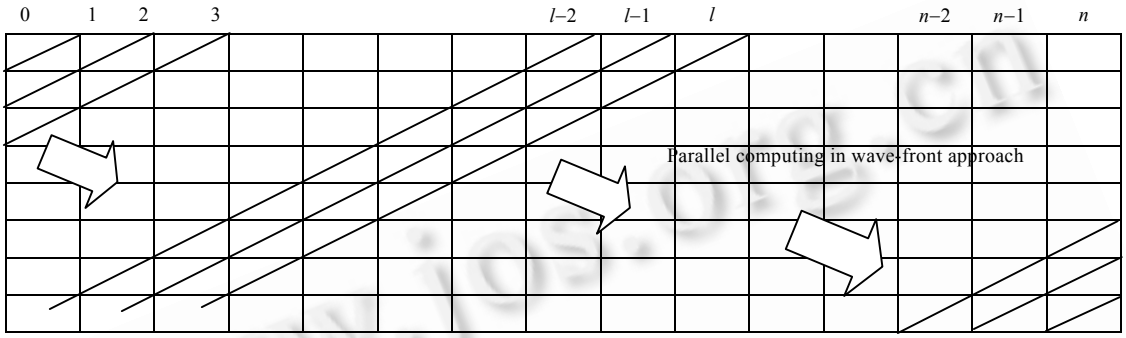


Fig.1 Implementing the approximate string matching with k-differences by parallel computing the edition distance matrix D in wave-front approach

图 1 波前式并行推进直接计算编辑距离矩阵 D 实现允许 k-差别的近似串匹配

引理 1. 编辑距离矩阵 D 的“反对角线”的数目为 $n+m+1$.

证明:包括边界的第 0 行和第 0 列在内,编辑距离矩阵 D 共有 $m+1$ 行和 $n+1$ 列元素.矩阵中每列都对应 1 条“反对角线”,每行也都对应 1 条“反对角线”,其中有 1 条“反对角线”被第 n 列和第 0 行交叉共享,因此,编辑距离矩阵 D 的“反对角线”的数目为 $n+m+1$. □

引理 2. 编辑距离矩阵 D 中任一条“反对角线”上的元素数目最多为 $m+1$.

证明:对于编辑距离矩阵 D 中任意的一条“反对角线”,因为它在矩阵的每一行最多包含(穿过)一个元素,而矩阵 D 总共有 $m+1$ 行,所以编辑距离矩阵 D 中任一条“反对角线”上的元素数目最多为 $m+1$. □

计算编辑距离矩阵 D 的另一个问题是如何降低算法对存储空间的需求.为此,引入规模分别为 $m+1$ 的 3 个向量 L,PL 和 PP,用于存储矩阵 D 中当前正在计算的第 l 条“反对角线”上元素的值、第 l 条“反对角线”的前驱——第 $l-1$ 条“反对角线”上元素的值,以及第 $l-1$ 条“反对角线”的前驱——第 $l-2$ 条“反对角线”上元素的值.

考查如图 1 所示的矩阵中第 l 条“反对角线”上元素 L_{i-1},L_i 和 L_{i+1} 的编辑距离的并行计算.处理器 PR_i 为了计算 L_i 的值,需要读取 PP_{i-1} 或者 PL_{i-1} 和 PL_i 的值;同样,PR_{i-1} 为了计算 L_{i-1} 的值,需要读取 PP_{i-2} 或者 PL_{i-2} 和 PL_{i-1} 的值;而 PR_{i+1} 为了计算 L_{i+1} 的值,需要读取 PP_i 或者 PL_i 和 PL_{i+1} 的值.因此,算法存在多个处理器需要同时读取同一数据的情形.对于 CREW-PRAM 模型,用 $O(1)$时间可解决并发读冲突问题.

定理 1. 对于 CREW-PRAM 计算模型,使用$(m+1)$个处理器,波前式并行推进直接计算编辑距离 D 的允许 k-差别的近似串匹配动态规划算法所需时间为 $O(n)$,空间复杂度为 $n+4m+5$,执行代价为 $O(nm)$,算法理论上达到线性加速.

证明:若 $m=1$,则算法退化为顺序算法.对于 $m>1$ 的情形,算法并行地初始化向量 L,PL 和 PP 所需的时间为 $O(1)$.根据前面的分析,对于 CREW-PRAM 模型,并行计算第 l 条“反对角线”上各编辑距离元素值的操作可以在 $O(1)$时间内完成.并行更新向量 L,PL 和 PP 所需的时间也为 $O(1)$.算法共迭代 $n+m-1$ 次.因此,算法的执行时间为 $O(n+m)$.由于算法使用 $O(m)$个处理器,所以执行代价是 $O(nm)$,加速比为 $O(m)$,执行效率为 $O(1)$,这表明算法在执行过程中各处理器几乎全部是活跃的,算法理论上达到线性加速.算法所需的空间为 $n+4m+5$. □

1.3 水平和斜向双并行计算编辑距离矩阵D的允许k-差别的近似串匹配动态规划算法

如果系统的处理器数目比模式长度 m 要大,那么仅运用波前式并行计算编辑距离矩阵 D 的方法去求解允

许 k-差别的近似串匹配问题,就没有充分发挥所有处理器的作用,因此需要进一步挖掘算法的并行性.

假设系统有$\alpha(m+1)$个处理器 PR_1~$PR_{\alpha(m+1)}$,α为正整数且$1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$.为方便讨论起见,假设$\alpha$能整除 n.应用分治策略,将长度为 n 的正文串 $T[1,\ldots,n]$划分成长度分别为 n/α的α个正文段:$T[(i-1)\times n/\alpha+1,\ldots,i\times n/\alpha]$,$1\le i\le\alpha$.对于此$\alpha$段正文,分别构造$\alpha$个编辑距离子矩阵 D_i,$1\le i\le\alpha$.D_i 的规模分别为$(m+1)\times(n/\alpha+m)$,$1\le i\le\alpha-1$;最后一个编辑距离子矩阵 D_α的规模为$(m+1)\times(n/\alpha+1)$.将处理器 $PR_{(i-1)(m+1)+1}$~$PR_{i\times(m+1)}$映射分配给编辑距离子矩阵 D_i,$1\le i\le\alpha$.编辑距离子矩阵的结构如图 2 所示.

		$T[(i-1)n/\alpha+1]$	…	$T[i\times n/\alpha-1]$	$T[i\times n/\alpha]$	…	$T[i\times n/\alpha+m-1]$
	0	0	…	0	0	…	0
$P[1]$	1						
…	…						
$P[j]$	j						
…	…						
$P[m]$	m						

Fig.2 Edition distance sub-matrix D_i

图 2 编辑距离子矩阵 D_i

因为各编辑距离子矩阵具有相对独立性,所以这些编辑距离子矩阵可以并行计算——从水平方向看,并行计算α个编辑距离子矩阵;对于每个编辑距离子矩阵,则按波前式并行推进的方法——斜向并行计算编辑距离子矩阵 D_i,$1\le i\le\alpha$.这样,构成一种水平和斜向双并行计算编辑距离矩阵 D 的格局,如图 3 所示.

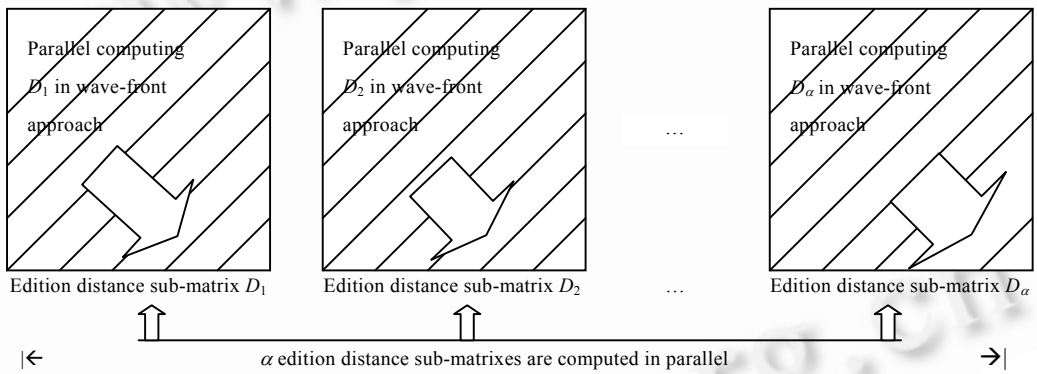


Fig.3 Parallel computing edition distance matrix D along the diagonal and horizontal directions

图 3 水平和斜向双并行计算编辑距离矩阵 D

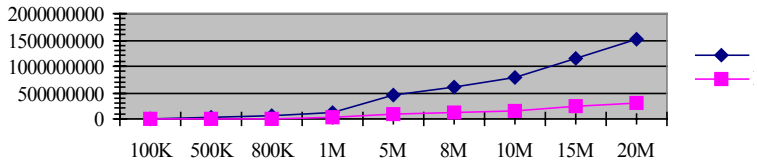
定理 2. 对于 CREW-PRAM 模型,水平和斜向双并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配动态规划算法所需时间为 $O(n/\alpha+m)$,使用$\alpha(m+1)$规模的处理器,执行代价为 $O(nm)$,$1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$.当处理器数目为 n 时,算法时间复杂度为 $O(m)$,理论上达到线性加速.

证明:当并行比较正文和模式字符时,多个处理器需要同时读取模式的字符,对于 CREW-PRAM 模型,这种并发读冲突问题可在 $O(1)$时间内解决.对于某一段正文的匹配比较,算法使用$(m+1)$个处理器按波前式并行推进的方法计算编辑距离.由定理 1 可知,每段正文的匹配比较工作所需时间为 $O(n/\alpha+m)$.因为共有$\alpha(m+1)$个处理器,所以,可以并行地对这α段正文进行编辑距离计算和模式匹配比较.因此,水平和斜向双并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配算法所需时间为 $O(n/\alpha+m)$.因为$1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$,所以算法的执行代价为 $O(nm+\alpha m^2)=O(nm)$.特别地,当处理器数目为 n 时,算法的时间复杂度为 $O(m)$,加速比为 $O(n)$. □

1.4 算法的仿真实验

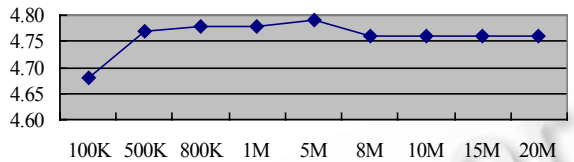
采用并行 MULTIPASCAL 软件系统[8]编程在 MS DOS 平台上仿真 SM-MIMD 并行机器,实现波前式并行计算编辑距离矩阵 D 以及水平和斜向双并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配算法.波前式并

行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配动态规划算法的仿真实验结果如图 4(a)~(d)所示.



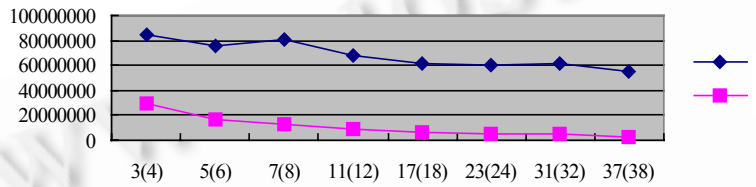
(a) The vertical coordinate denotes the execution time (ms), the horizontal coordinate denotes length n of text, m=5, the number of processors is 6, k=1

(a) 纵轴为执行时间(ms),横轴为正文规模 n,m=5,处理器数为 6,k=1



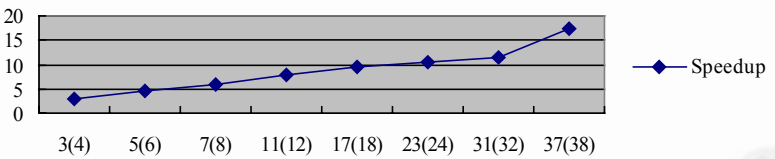
(b) The vertical coordinate denotes the speedup, the horizontal coordinate denotes length n of text, m=5, the number of processors is 6, k=1

(b) 坐标纵轴表示加速比,横轴为正文规模 n,m=5,处理器数为 6,k=1



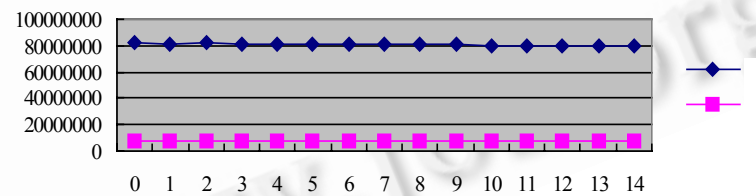
(c) The vertical coordinate denotes the execution time (ms), the horizontal coordinate denotes the length of pattern, n=1MB, k=1

(c) 纵轴为执行时间(ms),横轴为模式长度 m,n=1MB,k=1



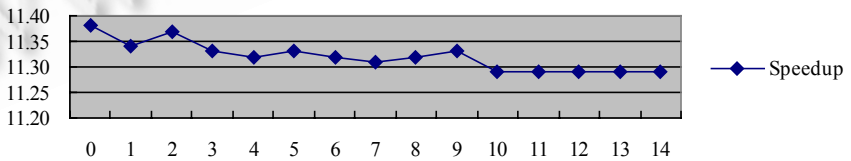
(d) The vertical coordinate denotes the speedup, the horizontal coordinate denotes length m of pattern, n=1MB, k=1

(d) 纵轴为加速比,横轴为模式长度 m,n=1MB,k=1



(e) The vertical coordinate denotes the execution time (ms), the horizontal coordinate denotes the value of k, n=1MB, m=15, the number of processors is 16

(e) 纵轴表示执行时间(ms),横轴为 k 值,n=1MB,m=15 处理器数为 16



(f) The vertical coordinate denotes the speedup, the horizontal coordinate denotes the value of k, n=1MB, m=15, the number of processors is 16

(f) 纵轴表示加速比,横轴为 k 值,n=1MB,m=15,处理器数为 16

Fig.4 The result of the simulation experiment for parallel approximate string matching algorithm with k-differences by parallel computing edition distance matrix D in wave-front approach

图 4 波前式并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配算法仿真实验结果

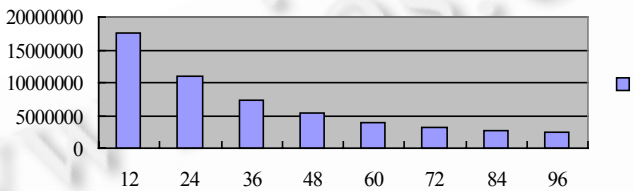
从图 4(a)可以看到,当模式长度固定为 5(处理器数目为 6)时,无论是串行执行时间还是并行执行时间,都随

着正文规模 n 的增大而相应地增加,这与理论分析是一致的.当 n 增大到一定规模时,所有处理器几乎都在有效地工作.图 4(b)表明,无论正文规模 n 如何,并行算法均呈亚线性加速趋势,且加速比相差不超过±0.11.

图 4(c)表明,若固定正文规模,则当模式规模(处理器数)增大时,求解问题的时间也相应减少.此结果与增加处理器可以加快匹配速度的期望相吻合.图 4(d)表明,加速比平均约等于 0.5,与理论分析的线性加速有一定的距离.究其原因是受仿真实验条件的限制使得正文规模 n 相对较小,而系统生成并发进程、同步进程和解决并发读所花费的时间在并行算法求解问题总时间中所占比例较高.另一方面,当模式长度为 31(处理器数为 32)时,加速相对较小,这是因为此时近似匹配成功的机会较多(输出匹配位置占据较多时间).这说明,并行算法实际的运行时间和加速在量的意义上与模式和正文内容的构成有一定的关系.

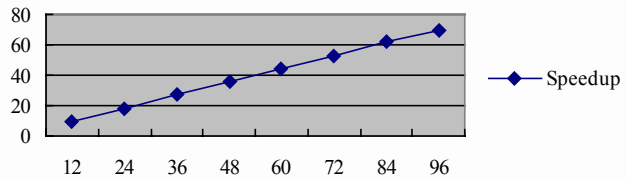
从图 4(e)和图 4(f)可以看到,若固定正文规模为 1MB、模式长度为 15(处理器数为 16),则无论是串行执行时间还是并行执行时间几乎都与错误阈值 k 无关,这与理论分析完全一致.此外,k 值的大小对加速的影响也非常有限.

水平和斜向双并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配动态规划算法的仿真实验结果如图 5 所示,其中正文规模 n 固定为 2MB,模式长度 m=11,错误阈值 k=1,处理器规模从 12 增长到 96.



(a) The vertical coordinate denotes the execution time (ms), the horizontal coordinate denotes the number of processors, n=2MB, m=11, k=1

(a) 纵轴为并行算法的执行时间(ms),横轴为处理器数目,n=2MB,m=11,k=1



(b) The vertical coordinate denotes the speedup, the horizontal coordinate denotes the number of processors, n=2MB, m=11, k=1

(b) 纵轴为并行算法的加速比,横轴为处理器数目,n=2MB,m=11,k=1

Fig.5 The result of the simulation experiment for parallel approximate string matching algorithm with k-differences by parallel computing edition distance matrix D along the diagonal and horizontal directions

图 5 水平和斜向双并行计算编辑距离矩阵 D 的允许 k-差别的近似串匹配并行算法仿真实验结果

从图 5(a)可以看到,当正文规模和模式长度固定时,增加处理器数目可以显著地减少并行算法的执行时间;另一方面,并行算法所需的执行时间下降的速度开始时比较快,然后逐步减缓,这是由于当处理器数目不断增加时,分配给相应处理器所处理的正文段的字符数目越来越少.这表明,系统可用的处理器数目越多,分配给每个处理器所处理的正文段就越短,从而使得并行算法有效地发挥所有处理器作用的时间也就越短.因此,当正文规模和模式长度固定时,一方面,增加系统可用处理器数目可以显著减少并行算法的执行时间;另一方面,当处理器数目增加到一定规模时,并行算法执行时间下降的速度逐步减缓.

图 5(b)表明,当正文规模和模式长度固定而处理器数目增加时,并行算法能够保持亚线性加速的趋势.

2 LARPBS 系统上的允许 k-误配的近似串匹配并行算法

2.1 允许k-误配的近似串匹配问题

定义 4. 对于任意的两个符号(字符)$a,b\in\Sigma$,定义布尔函数 fb 如下:

$$fb(a,b)=\begin{cases}1, & \text{若}a\neq b\\ 0, & \text{否则}\end{cases}.$$

设串 $X=X[1,\ldots,n], Y=Y[1,\ldots,n]\in\Sigma^*$,则 X 和 Y 之间的汉明距离(Hamming distance)定义为

$$ham(X,Y)=\sum_{i=1}^{n} fb(X[i],Y[i]).$$

定义 5. 设模式 $P=P[1,\ldots,m]$,正文 $T=T[1,\ldots,n]\in\Sigma^*$,$m<n$,任意给定整数 k,$0\le k<m$,所谓允许 k-误配的近似串匹配问题是指,寻找出所有满足条件 $ham(P,T_i)\le k$ 的模式在正文中匹配的起始位置 i,其中 $1\le i\le n-m+1$,子串 $T_i=T[i,\ldots,i+m-1]$.

允许 k-误配的近似串匹配问题的求解可转变成求解如下函数:

$$fp:\Sigma^m\times\Sigma^n\times k\to\{0,1\}^{n-m+1},$$

其中 $fp(P,T,k)=c_1c_2\ldots c_{n-m+1}$,对于 $1\le i\le n-m+1$,有

$$c_i=\begin{cases}1, & \text{若} ham(P,T_i)\le k\\ 0, & \text{否则}\end{cases}.$$

2.2 LARPBS模型及其基本数据移动操作

可重构流水线总线系统(LARPBS)模型使用光总线连接处理器,并使用光波在处理器之间传输消息[9,10].利用光波传送消息的优点是:高速、光信号单向传输、单位长度的可预测的传输延迟,后两个特性保证 LARPBS 系统能够以流水线方式同步并发存取光总线.上述优点加上总线结构高效的广播和多播能力,使得 LARPBS 计算模型十分适合于通信操作密集的应用.可重构光总线系统 LARPBS 的结构图从略,参见文献[9,10].

LARPBS 系统除了具有极强的通信能力以外,可以在 1 个总线周期内任意地将一个流水线光总线阵列处理器系统动态地重构成 l 个独立的子系统,$1\le l\le n$,这些子系统可以独立地进行不同的计算而相互不受干涉.LARPBS 系统的计算由一系列全局通信和局部计算步骤组成,其计算复杂度采用总线周期数和算术操作次数来度量.

引理 3[9,10]. LARPBS 系统总线上每个处理器将 1 个数据项发送给另一个处理器的操作称为点对点通信,它可以在 1 个总线周期内完成.

引理 4[9,10]. LARPBS 系统的源处理器将其局部寄存器的数据值发送到总线上所有 n 个处理器的操作称为广播,广播操作可以在 1 个总线周期内完成.

引理 5[9,10]. LARPBS 系统的源处理器将其局部寄存器的数据值发送到总线上多个处理器的操作称为一对多广播(多播).每个目标处理器在 1 个总线周期内可以接收源处理器发送来的数据.

引理 6[9,10]. 假设 n 个数据分布在 LARPBS 系统总线上的 n 个处理器中,每个处理器保存 1 个数据.并假设活跃的(active)数据元素为 s 个,$1\le s\le n$.所谓活跃元素依据其局部变量的值来标识.拥有活跃元素的处理器称为活跃处理器.将活跃元素迁移到处理器 PR_{n-s}~PR_{n-1} 的操作称为压缩(compression),压缩操作可以在 $O(1)$总线周期内完成.

引理 7[9,10]. 对于 n 个处理器的 LARPBS 系统,每个处理器 PR_i 保存 1 个二进制值 v_i, $0\le i\le n-1$.求 $psum_i=\sum_{j=0}^{i}v_j$ 称为计算二进制值前缀和, $0\le i\le n-1$.计算二进制值前缀和的操作可以在 $O(1)$总线周期内完成.

2.3 LARPBS模型上的允许k-误配的近似串匹配并行算法

设正文 $T[0,\ldots,n-1]$存储在 LARPBS 系统上的 n 个处理器中,其中处理器 PR_i 存储正文字符 $T[i]$,$0\le i\le n-1$.此外,处理器 PR_i 包含用于存储汉明距离的数组元素 $S[i]$和布尔数组元素 $B[i]$,$0\le i\le n-1$.初始时,模式 $P[0,\ldots,m-1]$存储在处理器 PR_0~PR_{m-1} 中,即 PR_j 存储 $P[j]$,$0\le j\le m-1$.为方便讨论,假设 m 能够整除 n.在 LARPBS 模型上设计允许 k-误配的近似串匹配并行算法的思想是,基于分治策略,灵活拆分总线和合并子总线动态重构光总线系统,并充分利用光总线的消息播送技术和并行计算前缀和的方法,实现汉明距离的并行计算.算法描述如下:

Algorithm. Parallel approximate string matching with k-mismatches on LARPBS with n processors.

输入:$P[0,\ldots,m-1]$, $T[0,\ldots,n-1]$和 k;

输出:近似匹配起始位置 i,$0\le i\le n-m$.

Begin

(1) l=0,处理器 PR_0 将 k 广播给总线上所有的处理器.

(2) 长度为 n 的总线系统 LARPBS 并发地执行多播操作,其中处理器 PR_j 将模式字符 $P[j]$播送给处理器 $PR_{i\times m+j+l}$,$1\le i\le(n/m-1)$,$0\le j\le m-1$.

(3) 将长度为 n 的总线系统重构成 n/m 个长度为 m 的子总线系统,分别记为 LARPBS-i,$1\le i\le n/m$.

(4) 各子总线系统LARPBS-i 上所有处理器并行地比较其存储器中的正文字符 $T[(i-1)\times m+j]$和模式字符 $P[j]$,若 $T[(i-1)\times m+j]=P[j]$,则置 $B[(i-1)\times m+j]=0$,否则置 $B[(i-1)\times m+j]=1$,$0\le j\le m-1$,$1\le i\le n/m$.

(5) 各子总线系统 LARPBS-i 并行地求其上 m 个二进制值 $B[(i-1)\times m]$~$B[i\times m-1]$的前缀和 $psum_i$,然后执行点对点通信操作,将 $psum_i$ 发送到处理器 $PR_{(i-1)\times m+l}$ 的 $S[(i-1)\times m+l]$中,$1\le i\le n/m$.

(6) $l=l+1$,若 $l\le m-1$,则将 n 个处理器重构成一个长度为 n 的总线系统 LARPBS,然后各处理器并行地执行点对点通信操作(其中,若 $i+l\le n-1$,则处理器 PR_{i+l} 将其存储器中的 $T[i+l]$发送至 PR_i 的 $T[i]$中,$0\le i\le n-l-1$),然后转步骤(3);否则执行步骤(7).

(7) 重构长度为 n 的总线系统 LARPBS,处理器 PR_i 比较 $S[i]$和 k 的大小,如果 $S[i]\le k$,那么输出匹配位置 i,$0\le i\le n-1$.

End.

定理 3. 对于 n 个处理器的 LARPBS 系统,允许 k-误配的近似串匹配并行算法的时间复杂度为 $O(m)$,执行代价为 $O(mn)$.

证明:由引理 5 可知,多播操作可以在 1 个总线周期内完成,因此,算法步骤(2)并发进行多播通信所需时间为 $O(1)$.步骤(3)的子总线系统重构工作可以在 1 个总线周期内完成.显然,步骤(4)的并行比较操作所需时间为 $O(1)$.由引理 7 可知,求任意 n 个二进制值的前缀和可在 1 个总线周期内完成,并且由引理 3 可知,点对点通信操作也可以在 1 个总线周期内完成,因此,步骤(5)的并行操作仅需 $O(1)$时间.同样,算法步骤(6)的总线系统重构操作和并发点对点通信操作所需时间为 $O(1)$.算法步骤(3)~(6)共迭代 m 次.算法步骤(7)的并行比较和输出操作可以在常数时间内完成.

因此,对于 n 个处理器的 LARPBS 系统,允许 k-误配的近似串匹配并行算法的时间复杂度为 $O(m)$,执行代价为 $O(mn)$. □

现在研究设计基于 LARPBS 系统的常数时间复杂度的允许 k-误配的近似串匹配并行算法.假设 LARPBS 系统共有 $n\times m$ 个处理器,将长度为 $n\times m$ 的一个总线系统逻辑上看成是由 m 个长度为 n 的子总线系统组成,并将处理器分别编号为 PR_i,$0\le i\le nm-1$.初始时,正文存储在处理器 PR_i 的 $T[i]$中,$0\le i\le n-1$;而模式存储在处理器 PR_j 的 $P[j]$中,$0\le j\le m-1$.$S[0,\ldots,nm-1]$为中间数组,而数组 $PO[0,\ldots,n-1]$则用于保存可能的正文匹配起始位置.并假设特殊字符"#" $\notin\sum$.算法形式描述如下.

Algorithm. Parallel approximate string matching with k-mismatches on LARPBS with nm processors.

输入:$P[0,\ldots,m-1]$, $T[0,\ldots,n-1]$和 k;

输出:近似匹配起始位置 i,$0\le i\le n-m$.

Begin

(1) 长度为 nm 的总线系统 LARPBS 上各处理器 PR_i 执行操作 $S[i]=-1$,$0\le i\le n\times m-1$.

(2) 长度为 nm 的总线系统 LARPBS 执行广播操作将 k 播送给所有处理器.

(3) 长度为 nm 的总线系统 LARPBS 并发执行多播操作,其中处理器 PR_i 将正文字符 $T[i]$播送给处理器 $PR_{j\times n+i}$ 的 $T[j\times n+i]$,$0\le i\le n-1$,$0\le j\le m-1$.

(4) 长度为 nm 的总线系统LARPBS并发执行点对点通信操作,其中处理器 $PR_{j\times(n+1)+i}$ 将字符 $T[j\times(n+1)+i]$和正文起始位置$(j+i)$分别发送给处理器 $PR_{j\times n+i}$ 的 $T[j\times n+i]$和 $S[j\times n+i]$,$0\le i\le n-j-1$,$0\le j\le m-1$.

(5) 将长度为 nm 的总线系统 LARPBS 重构为 m 个长度 n 的子总线系统 LARPBS-j,$0\le j\le m-1$.m 个子总线系统并发执行多播操作,其中处理器 $PR_{j\times n}$ 判断 j 是否大于 0,若是,$PR_{j\times n}$ 则将特殊字符"#"发送给处理器 $PR_{(j+1)\times n-j},\ldots,PR_{(j+1)\times n-1}$ 对应的 $T[(j+1)\times n-j],\ldots,T[(j+1)\times n-1]$;否则,$PR_{j\times n}$ 执行空操作,$0\le j\le m-1$.

(6) 重构一个长度为 nm 的总线系统 LARPBS,并发执行多播操作,其中处理器 PR_j 将模式字符 $P[j]$播送给处理器 $PR_{i\times m+j}$ 的 $P[i\times m+j]$,$0\le j\le m-1$,$1\le i\le n-1$.

(7) 将长度为 nm 的总线系统重构成 n 个长度为 m 的子总线系统,分别记为 LARPBS-i,$1\le i\le n$.各子总线系统 LARPBS-i 上所有处理器并行比较其存储器中的正文字符 $T[(i-1)\times m+j]$和模式字符 $P[j]$,若 $T[(i-1)\times m+j]=P[j]$,则置 $B[(i-1)\times m+j]=0$;否则,置 $B[(i-1)\times m+j]=1$,$0\le j\le m-1$,$1\le i\le n$.

(8) 各子总线系统 LARPBS-i 并行求其上 m 个二进制值 $B[(i-1)\times m]$~$B[i\times m-1]$的前缀和 ps_i,ps_i 保存在处理器 $PR_{(i-1)\times m}$ 中,$1\le i\le n$.

(9) 重构一个长度为 nm 的总线系统,并发执行点对点通信操作,其中处理器 $PR_{j\times n+i\times m}$ 将 $S[j\times n+i\times m]$发送给处理器 $PR_{j+i\times m}$ 的 $PO[j+i\times m]$,$1\le j\le m-1$,$0\le i\le n/m-1$.

(10) 对于长度为 nm 的总线系统,并发执行点对点通信操作,其中处理器 $PR_{(i-1)\times m}$ 将 ps_i 发送给处理器 PR_{i-1} 的 p_s_{i-1},$1\le i\le n$.

(11) 激活长度为 nm 的总线系统上 n 个处理器 PR_{i-1},$1\le i\le n$;处理器 PR_{i-1} 比较汉明距离 p_s_{i-1} 和 k 的值,若 $p\_s_{i-1}\le k$,则输出匹配起始位置 $PO[i-1]$,$1\le i\le n$.

End.

定理 4. 对于 nm 个处理器的 LARPBS 系统,允许 k-误配的近似串匹配并行算法可在常数时间内完成.

证明:步骤(1)的并行赋初值操作所需时间为 $O(1)$.由引理 4 可知,步骤(2)的广播操作可在 1 个总线周期内完成.由引理 5 可知,步骤(3)的多播操作也可以在 1 个总线周期内完成.由引理 3 可知,步骤(4)并发执行点对点通信操作所需的时间为 $O(1)$.在算法的步骤(5),子总线系统重构工作可在常数时间内完成,而且其并发多播操作也可以在 1 个总线周期内完成.同样,算法步骤(6)的总线系统重构工作及其并发多播操作也可以在 $O(1)$时间内完成.对于算法步骤(7),一方面,其子总线系统重构工作可在 1 个总线周期内完成,另一方面,其并行比较操作所需时间为 $O(1)$.由引理 7 可知,步骤(8)的并发求二进制值前缀和的操作可在 1 个总线周期内完成.算法步骤(9)的总线系统重构工作及其点对点通信操作可在 $O(1)$时间内完成.算法步骤(10)并发执行点对点通信的操作可在常数时间内完成.算法步骤(11)的并行匹配比较和输出操作所需的时间为 $O(1)$.

因此,对于 nm 个处理器的 LARPBS 系统,允许 k-误配的近似串匹配并行算法所需时间为 $O(1)$,执行代价为 $O(nm)$.

3 结 语

基于 CREW-PRAM 模型,首先采用波前式并行推进的方法直接计算编辑距离矩阵 D,设计了一个允许 k-差别的近似串匹配动态规划算法,该算法使用$(m+1)$个处理器,时间复杂度为 $O(n)$,理论上达到了线性加速;然后,采取水平和斜向双并行计算编辑距离矩阵 D 的方法,设计了一个使用$\alpha(m+1)$个处理器和 $O(n/\alpha+m)$时间的可伸缩的允许 k-差别的近似串匹配动态规划并行算法,$1<\alpha\le\left\lceil\frac{n}{m+1}\right\rceil$,当处理器数目为 n 时,算法的时间复杂度为 $O(m)$,理论上达到线性加速.仿真实验结果表明,这两个算法均获得了良好的加速.所设计的算法简明、易于实现,适合于正文和模式在线输入的近似串匹配应用.

基于分治策略,通过灵活拆分总线和合并子总线动态重构光总线系统,并充分利用光总线的消息播送技术和并行计算前缀和的方法,实现汉明距离的并行计算,设计了两个通信高效的、允许 k-误配的近似串匹配并行算法,其中一个算法使用 n 个处理器,时间复杂度为 $O(m)$,另一个为常数时间算法,使用 mn 个处理器.这两个算法的执行代价均为 $O(nm)$.可重构光总线系统可以方便地扩展处理器的规模,使得它适合于不同规模问题的有效求解,因此,所给出的基于 LARPBS 模型的允许 k-误配的近似串匹配并行算法是可扩放的.

References:

[1] Navarro G. A guided tour to approximate string matching. ACM Computing Surveys, 2001,33(1):31~88.

[2] Navarro G, Baeza-Yates R. A hybrid indexing method for approximate string matching. Journal of Discrete Algorithms, 2000,1(1): 21~49.

[3] Lee H-C, Ercal F. RMESH algorithms for parallel string matching. In: Proc. of the 3rd Int'l. Symp. on Parallel Architectures, Algorithms, and Networks (I-SPAN'97). Los Alamitos: IEEE Computer Society Press, 1997. 223~226. http://ieeexplore.ieee.org/xpl/tocresult.jsp?isNumber=13955&page=2.

[4] Jiang Y, Wright AH. O(*k*) parallel algorithms for approximate string matching. Journal of Neural Parallel and Scientific Computation, 1993,1:443~452.

[5] Amir A, Landau GM. Fast parallel and serial multidimensional approximate array matching. Theoretical Computer Science, 1991, 81(1):97~115.

[6] Alan AB, Mei A. A residue number system on reconfigurable mesh with applications to prefix sums and approximate string matching. IEEE Trans. on Parallel and Distributed Systems, 2000,11(11):1186~1199.

[7] Park JH, George KM. Efficient parallel hardware algorithms for string matching. Microprocessors and Microsystems, 1999,23(3): 155~168.

[8] Lester B. The Art of Parallel Programming. Englewood Cliffs: Prentice Hall, 1993.

[9] Pan Y, Li Y, Li J, LI K, Zheng SQ. Efficient parallel algorithms for distance maps of 2-D binary images using an optical bus. IEEE Trans. on Systems, Man, and Cybernetics—Part A: Systems and Humans, 2002,32(2):228~236.

[10] Han Y, Pan Y, Shen H. Sublogarithmic deterministic selection on arrays with a reconfigurable optical bus. IEEE Trans. on Computers, 2002,51(6):702~706.

IEEE/WIC/ACM 2004 网上智能与智能 Agent 技术联合国际会议(WI/IAT 2004)

征 文 通 知

由 IEEE Computer Society、国际网上智能协会(Web Intelligence Consortium, WIC),、Association for Computing Machinery(ACM)、国家自然科学基金委员会、中国计算机学会和北京工业大学联合主办的 2004 IEEE/WIC/ACM 2004 网上智能与智能 Agent 技术联合国际会议(WI/IAT 2004)定于 2004 年 9 月 20 日~24 日在北京举行。本次会议热诚欢迎广大学术界和工业界的研究者踊跃投稿。同时，本次会议将设立成果展示会，届时将演示和宣传优秀的科研成果产品。

一．征文主题

• WI 2004 会议征文主题包括(但不限于)：智能电子技术；智能人网交互；知识网格和网格智能；Semantics 和 Ontology 工程；社会网络和社会智能；Web Agent；Web 信息过滤和检索；Web 挖掘；Web 安全和集成。

• IAT 2004 会议征文主题（但不限于）：Agent 系统建模和方法学；自组织知识信息 Agent；自组织计算(AOC)；分布式智能；分布式问题求解；可学习和自适应性 Agent。

二．征文要求

论文必须未公开发表，并用英文书写。本次会议只接受电子文档（PDF，PostScript 或 Ms-Word）。有关论文格式和投稿要求，详见会议网址。

三．论文出版

会议由 IEEE Computer Society Press 出版正式论文集，优秀论文将收录于 Web Intelligence and Agent System: An International Journal 和 Annual Review of Intelligent Informatics 中。

四．重要日期

论文截止日期：2004 年 4 月 4 日　　录用通知日期：2004 年 6 月 10 日

Workshop/Tutorial 申请截止日期：2004 年 6 月 10 日　　照排版论文截止日期：2004 年 7 月 5 日

Workshop/Tutorial 日期：2004 年 9 月 20 日　　会议日期：2004 年 9 月 21 日~24 日

五．会议主页：http://www.maebashi-it.org/WI04，http://www.maebashi-it.org/IAT04