

机群系统上长序列最大串联重复识别并行算法

陆向艳, 钟 诚

(广西大学 计算机与电子信息学院, 广西 南宁 530004)

摘 要: 采用适当的划分机制, 将序列的后缀划分为若干组, 在并行机群中独立对每组进行序列最大串联重复识别, 从而得到完整序列的最大串联重复, 给出一种减少了处理时间并降低了空间消耗的序列最大串联重复识别并行算法. 实验结果表明: 该并行算法具有良好的适应性和可扩展性.

关键词: 最大串联重复; 后缀数组; 划分; 并行算法; 机群系统

中图分类号: TP338.6

文献标识码: A

文章编号: 1000-7180(2010)08-0186-04

Parallel Algorithm for Long Sequences Maximal Tandem Repeats on the Cluster Computing Systems

LU Xiang-yan, ZHONG Cheng

(School of Computer and Electronics and Information, Guangxi University, Nanning 530004, China)

Abstract: The sequence suffix is partitioned into several parts by using appropriate policy, each partition is processed independently on the cluster systems to produce a temporary result, the final maximal tandem repeats set is achieved from the union of all temporary results, and a parallel algorithm for long sequences maximal tandem repeats recognition is presented based on partitioning suffix array. The parallel algorithm can decrease processing time and space consumption. The experimental results also show that the algorithm is flexible and scalable.

Key words: maximal tandem repeats; suffix array; partitioning; parallel algorithm; cluster computing systems

1 引言

重复序列识别是生物信息学中分析基因组序列的主要手段之一, 可分为串联重复序列和散在重复序列两大类. 串联重复序列是指 1~100 个碱基左右的核心重复单位, 以头尾相串联的方式重复多次所组成的重复序列, 它广泛存在于真核生物和一些原核生物的基因组中. 通过识别串联重复序列可以发现基因组的进化规则和许多疾病的遗传规律. 目前, 每天都有大量的基因组数据提交到 GeneBank 等数据库上, 其规模逐年成倍增长, 因此设计高效的串联重复序列识别算法非常重要. 大多数生物序列分析软件包中都有重复序列识别工具^[1-2], 但这些工具只能识别少量特定的最大串联重复序列, 并且相应

算法的描述很少在文献中报道. 文献[3-5]研究了最大串联重复识别的串行算法. 然而, 很少看到有在更实际的并行计算环境下设计的最大串联重复并行算法的研究成果. 针对减少长序列最大串联重复识别的时间和空间消耗问题, 本文在文献[5]的算法的基础上, 采取并行划分后缀方法, 给出一个在机群系统上实现的序列最大串联重复识别并行算法并进行实验性能测试.

2 相关概念和算法简介

2.1 相关概念

设 Σ 是一个含有 m 个字母的有限字母表 (在 DNA 序列分析中, 字母表 Σ 仅含有四个字母: a, g, c, t 或 A, G, C, T). 若 $S = a_1a_2 \cdots a_n, a_i \in \Sigma (1 \leq$

$i \leq n$), 则称 S 是 Σ 上的一个字符串(即序列), $|S|$ 为序列长度. 对于 S , 若存在一个整数 p , 使 $a_i = a_{i+p}$, ($1 \leq i < i+p \leq n$), 则整数 p 称为 S 的一个周期. 定义指数 $e = |S|/p$, 当 $e \geq 2$ 时, S 就是一个串联重复. 假设 $S[i \cdots j]$ 是序列的一个子序列 ($1 \leq i < j \leq n$), 若 $S[i \cdots j]$ 是最小周期为 p 的一个串联重复, 则当不等式 $S[i-1] \neq S[i-1+p]$ 成立时, 即 $S[i \cdots j]$ 不能向左边扩展时, $S[i \cdots j]$ 就是左最大重复; 当不等式 $S[j+1] \neq S[j+1-p]$ 成立时, 即 $S[i \cdots j]$ 不能向右边扩展时, $S[i \cdots j]$ 是右最大重复. 当一个串联重复既是左最大重复又是右最大重复时, 即为一个最大串联重复.

2.2 基于后缀数组的最大串联重复识别顺序算法的分析

后缀数组是一个字符串所有后缀的字典序索引, 是后缀树的一种简化表示. 后缀树和后缀数组是序列分析中两种非常有效的数据结构.

定义 1 对于字符串 $S = S[0 \cdots n] = t_0 t_1 \cdots t_n$, 令 $S_i = S[i \cdots n] = t_i t_{i+1} \cdots t_n$, $i \in [0, n]$, S_i 称为其第 i 个后缀. S 的后缀数组 $SA[0, n]$ 为 $[0, n]$ 之间的一个序列, 满足 $S_{SA[0]} < S_{SA[1]} < \cdots < S_{SA[n]}$.

定义 2 令 $lcp(i)$ 为后缀数组中第 i 个后缀及第 $i-1$ 个后缀之间的最大公共前缀的长度 ($1 \leq i \leq n$), 则 LCP 称为最大公共前缀列, 推广到任意的 i 与 j 有:

$$lcp(i, j) = \min\{lcp[i+1], lcp[i+2], \dots, lcp[j]\}, (1 \leq i < j \leq n)$$

通过考察后缀数组以及它同 LCP 列的关系, 利用文献[5]中的引理 1~3 就可以确定串联重复的位置. 文献[5]中基于后缀数组的最大串联重复识别算法描述如下:

算法 1 基于后缀数组的最大串联重复识别算法^[5]

输入: DNA 序列 S , 最小长度 ml , 最小周期 $minlen$, 最小重复次数 $mintime$.

输出: 序列 T 中所有长度 $\geq ml$ 的最大串联重复.

Begin

Step1: 计算序列 S 的后缀数组 $SA[i]$ 和 LCP 列 $lcp(i)$, ($0 \leq i \leq n$).

Step2: 利用标记数组 $Tonken[i]$ 将 lcp 数组中满足 $lcp[i] < (ml+1)/2$ 的元素标记为 1, 其余为 0. 标记为 0 的元素 $lcp[j]$ 满足 $lcp[j] \geq (ml +$

$1)/2 > ml/2$.

Step3: 对数组 $Tonken[i]$ 中连续为 0 的 k ($0 \leq k < n$) 个元素对应的 $lcp[i]$, $lcp[i+1]$, $\dots$, $lcp[i+k-1]$, 对任意的 j ($i < j \leq i+k-1$), 计算 $lcp(i, j)$, 利用文献[5]中的引理 1~3 计算是否存在一个起始于位置 $position = \min\{SA[i], SA[j]\}$ 的右最大重复. 若存在这样的右最大重复, 则判断其是否属于左最大重复, 若是, 则该重复为要找的最大串联重复, 其中 $d = |SA[j] - SA[i]|$ 为重复周期长度, $l = d + lcp(i, j)$ 为串长度. 将 $(position, d, l)$ 存入最大串联重复集合 MTR 中.

Step4: 判断 MTR 中的元素是否满足最小周期和最小重复次数的设置, 如满足, 则返回结果.

End.

算法 1 利用后缀数组 $SA[i]$ 及其衍生产品——最大公共前缀列 LCP 列 $lcp(i)$ 的特点来实现. 文献[6]和文献[7]分别给出了 $SA[i]$ 与 LCP 列的线性时间构造方法. 与文献[3]所提出的基于后缀树的最大串联重复识别算法相比, 算法 1 时间复杂度与之基本相同, 均为 $O(n)$; 内存空间需求上算法 1 是 $5n$, 文献[3]是 $13n$, 因此算法 1 在空间消耗上比文献[3]优越. 然而算法 1 虽然在空间消耗上有所降低, 但是对于长序列 (100MB 以上), 其空间需求仍然巨大, 甚至会超过计算机系统内存容量限制, 而且其时间消耗也较大.

3 长序列后缀数组的划分与序列最大串联重复的并行识别

在计算生物学中, 为了系统地研究全基因组中的重复序列, 需要处理的数据量往往成百上千 MB 以上, 仅单个序列最大长度也可达到 100MB 以上. 对于如此庞大的数据处理, 需要解决时间消耗和空间消耗的问题. 这就是串联重复序列识别算法的挑战. 对于长度为 n 的序列, 建立后缀树的空间消耗一般需要 $20n \sim 40n$, 建立后缀数组也需 $3n \sim 5n$, 因此对于长序列的处理往往会超过内存规模. 为了能够实现长序列的最大串联重复识别, 需要将序列划分为若干子序列, 对各子序列分别识别最大串联重复, 将各子序列最大串联重复集合并为原始序列的最大串联重复集合.

3.1 长序列后缀数组的划分

由文献[5]的定义 1、2 可知, 最大串联重复出现在其最大公共前缀长度为 l ($l > 0$) 的第 $i \sim j$ ($0 < i < j \leq n$) 个后缀中, 即满足 $lcp(i, j) = l$ ($l > 0$)

的第 $i \sim j$ 个后缀中, 这些后缀在后缀数组中其下标是连续的段组. 例如序列 $S = \text{"TCACACACATG"}$, 其后缀数组及对应的后缀如表 1 所示. 则在其后缀数组 SA 中, 存在 4 个下标连续的段组 $[9 \cdots 10]$, $[0 \cdots 3]$, $[4 \cdots 7]$, $[8 \cdots 8]$, 各组对应的后缀分别是 $\{\text{"TCACACACATG"}, \text{"TG"}\}$, $\{\text{"CACACACATG"}, \text{"CACACATG"}, \text{"CACATG"}, \text{"CATG"}\}$, $\{\text{"ACACACATG"}, \text{"ACACATG"}, \text{"ACATG"}, \text{"ATG"}\}$, $\{\text{"G"}\}$, 这些分组其前 l 个字母相同, 其中第 2、3 组各存在 1 个最大串联重复: "CACACACA" , "ACACAC" .

表 1 序列 "TCACACACATG" 的后缀数组

SA	位置	对应位置的后缀
9	0	TCACACACATG
4	1	CACACACATG
0	2	ACACACATG
5	3	CACACATG
1	4	ACACATG
6	5	CACATG
2	6	ACATG
7	7	CATG
3	8	ATG
10	9	TG
8	10	G

根据以上分析得出一种识别最大串联重复识别的序列划分方法: 设 S 是 Σ 上的一个序列, M 是 S 的所有后缀构成的集合, 将 M 按照首字符分为 $|\Sigma|$ 组, 各组内部后缀首字符相同, 在每组内求最大串联重复, 各组最大串联重复的并集即为 S 的最大串联重复集合.

对于长序列的划分, 以上划分方法可以推广如下: 首先按照首字母分组, 由于各分组中后缀在后缀数组中的下标是连续的段组, 因此可在各组内再按后缀的第二个字母进行第二次划分, 划分后的分组其后缀数组中的下标仍然是连续的段组, 以此类推, 直到划分后的各组长度符合要求.

3.2 机群系统上长序列最大串联重复识别并行算法的设计与分析

在机群系统上, 设计实现长序列最大串联重复识别并行算法的基本思想是: 第一阶段是将序列 S 按照 3.1 小节的方法进行划分. 第二阶段是各节点并行计算在自己负责段组内的最大串联重复, 最后将执行结果返回.

算法 2 机群系统上长序列最大串联重复识别

并行算法

输入: 序列 S , 最小长度 ml , 最小周期 $minlen$, 最小重复次数 $mintime$.

输出: 序列 S 中所有长度 $\geq ml$ 的最大串联重复.

Begin

第一阶段(序列划分阶段):

(1) 主处理器将序列 S 的是全部后缀首先按照首字符分为 4 组, 若每组规模超过内存规模或机群节点数 $p > 4(p = 2m)$, 则每组内再按照 S 的第二个字母进行划分, 以此类推. 将这些组均匀分配并分别播送到各从处理机(划分完毕后主处理机可充当从处理机);

第二阶段(最大串联重复识别阶段):

(2) 从处理机 PE_j 接收目标分组集合 K_j ;

(3) 从处理机 PE_j 顺序对 K_j 的各个分组利用算法 1 求出其相应最大串联重复;

(4) 从处理机 PE_j 将所识别的最大串联重复集合返回给主处理机;

End.

假设算法 2 将长度为 n 的序列 T 的后缀数组划分为 $k(k \geq p)$ 组, 则最大串联识别的计算时间是 $O((n/k) * (k/p)) = O(n/p)$. 主从处理器间的通信仅发生在计算开始之前和计算结束之后, 并行传输的通信时间是 $O(n/k)$. 内存消耗是 $5n/k$.

4 实验

实验平台采用 2 台 PC 构造并行机群, 网络速度是 100Mb/s, 节点配置均为是 Intel Pentium IV 2.66GHz、内存 512MB 兼容机. 各节点操作系统为 Windows XP, 采用 MPI 和 VC++ 进行编程. 使用 NCBI 数据库中大小不等的 DNA 序列作为测试对象测试文中算法的运行时间和加速比. 如表 2、表 3 所示.

表 2 串联重复串最小长度为 4 的并行算法性能

序列编号	序列长度	并行执行时间(s)	串行执行时间(s)	加速比
DQ198799.1	8108	0.20	0.25	1.22
AY584531.1	12061	0.61	0.78	1.34
AP006270.1	113220	5.28	5.63	1.51
NC_010368.1	284332	5.98	9.57	1.63
BA000002.3	691920	25.91	43.79	1.69

实验结果表明: (1) 随着序列长度的增加, 算法加速比获得相应增大, 这是因为并行算法在执行过

程中,随着序列长度的增大,主从处理机之间所需的通信开销在减少,算法总的处理时间(通信时间和计算时间)在降低;(2)算法执行时间随着串联重复串最小长度的增加而减少,原因是如果串联重复串最小长度越小,那么算法搜索空间就越大,搜索时间也就越长^[8].

表3 串联重复串最小长度为2的并行算法性能

序列编号	序列长度	并行执行 时间(s)	串行执行 时间(s)	加速比
DQ198799.1	8108	0.22	0.26	1.18
AY584531.1	12061	0.70	0.85	1.21
AP006270.1	113220	5.69	7.92	1.39
NC_010368.1	284332	22.17	33.75	1.52
BA000002.3	691920	49.36	78.79	1.60

5 结束语

文中给出的并行算法将序列后缀划分为若干组,可以对每组独立地进行长序列最大串联重复识别,从而得到完整序列的最大串联重复.算法分析与实验结果表明该方法具有很好的适应性和可扩展性,可以提高长序列有最大串联重复识别处理时间,并有效解决内存容量受限制问题.

参考文献:

- [1] Benson G. Tandem repeats finder, a program to analyze DNA sequences [J]. Nucleic Acid Research, 1998(27): 573-580.

- [2] Sonnhammer E L L, Durbin R. A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis[J]. Gene, 1995, 167(1/2): GC1-GC10.
- [3] Gusfield D, Stoye J. Linear time algorithms for finding and representing all the tandem repeats in a string[R]. Dept. of Computer Science, University of California, Davis, 1998.
- [4] Parisi V, Fonzo VD, Aluffi-Pentini F. String: finding tandem repeats in DNA sequences [J]. Bioinformatics, 2003, 19(14): 1733-1738.
- [5] 王晓敏, 王正志. 基于后缀列的基因序列最大串联重复查找技术[J]. 生物信息学, 2006(2): 72-75.
- [6] Karkkainen J, Sanders P. Simple linear work suffix array construction[C]// Proceedings of Automata, LINGA-GS and Programming. Berlin: Springer-Verlag, 2003: 943-955.
- [7] Kasai T, Lee G. Linear-time longest-common-prefix computation in suffix arrays and its applications [C]// Proc. of Annual Symposium on Combinatorial Pattern Matching. Berlin: Springer-Verlag, 2001: 181-192.
- [8] 钟诚, 陈国良. PRAM 和 LARPBS 模型上的近似串匹配并行算法[J]. 软件学报, 2004, 15(2): 159-169.

作者简介:

陆向艳 女, (1973-), 硕士, 讲师. 研究方向为计算机网络与并行分布计算.

钟 诚 男, (1964-), 博士, 教授, CCF 高级会员. 研究方向为计算机网络与并行分布计算.

(上接第 185 页)

- [2] Canfora G, Penta M D. Testing services and services-centric systems: challenges and opportunities[C]// IT Pro March. Italy, 2006: 10-17.
- [3] Yonez HH Liu, Li ZJ, Zhu J, et al. Business process regression testing [C]// ICSOC 2007. Austria: Vienna, 2007.
- [4] Cesare Pautasso. JOpera: an agile environment for web service composition with visual unit testing and refactoring [C]// VL/HCC'05. USA: Dallas, 2005.
- [5] Ginige J A, Sirinivasan U, Ginige A. A mechanism for efficient management of changes in BPEL based business processes: an algebraic methodology [C]// ICEBE'06. China: Beijing, 2006.
- [6] 金若凡, 孙海龙, 刘旭东, 等. SOArTester: 一个基于精简用例的组合服务自动化测试系统[C]// NASAC'09. 中

国: 沈阳, 2009.

- [7] Rapps S, E J Weyuker. Selecting software test data using data flow information[J]. IEEE Transactions on Software Engineering, 1985, 11(4): 367-375.

作者简介:

蒋俊彦 男, (1980-), 硕士研究生, 工程师. 研究方向为服务计算、服务测试.

李先贤 男, (1969-), 教授, 博士生导师. 研究方向为服务计算、可信计算、信息安全.

李建欣 男, (1979-), 博士, 讲师. 研究方向为服务计算、可信计算、信息安全.

吴 茜 女, (1987-), 硕士研究生. 研究方向为服务计算、服务测试.