DNA片段拼接中重复序列算法研究*)

王 磊 张祖平 陈建二
(中南大学信息科学与工程学院 长沙 410083)

摘 要 本文主要研究DNA片断拼接中重复序列信息识别算法。包含大量重复信息的DNA序列,其重构是大规模DNA片段拼接所面临的实际困难之一。针对目前大多数拼接算法对于重复段的处理采用效率较低的反复迭代算法的特点,提出了基于k-mer子串的重复段分析方法,充分考虑了拼接中可能的分割点,设计与分析了识别重复序列并提高序列一致性的高效算法。

关键词 生物信息学,片段拼接,重复片断,k-mer子串

Algorithms of Repeats in DNA Fragment Assembly

WANG Lei ZHANG Zu-Ping CHEN Jian-Er
(School of Information Science & Engineering, Central South University, Changsha 410083)

Abstract The paper mainly studies repeats analysis in DNA fragment assembly. One of the practical difficulties that remain in large-scale DNA fragment assembly is the correct reconstruction of DNA sequences that contain repeats. Aiming at the characteristic of iteratively dealing with repeats by most fragment assemblers which are very inefficient, the paper not only proposes a general method of kmer-based repeat analysis that fully considers all possible break points in the assembly, but also designs and analyses high-performance algorithms to move repeats as well as improve overall consistency.

Keywords Bioinformatics, Fragment assembly, Repeat, k-mer substring

1 引言

为了适应复杂生物(例如人等)大规模、长片断的DNA序列测定的需要,当前世界范围内的主要测序中心以及重要的测序工程都普遍采用了鸟枪(Shotgun)测序法。它根据目前可以用测序仪直接测出序列的长度水平,将较长的DNA序列的多条克隆随机打断成很短的片断,再通过测序仪精确地将这些小的片断序列一一测出,最后根据这些小片断序列间的重叠关系用计算机将它们进行拼接,以期得到目标序列的一个或多个较长的连续段[2]。

在DNA片段拼接中,包含重复序列信息的片段(简称重复片段)的正确匹配是拼接shotgun序列的实质性困难之一[8]。Shotgun集合中的序列片断拼接成目标序列的主要难点在于重复序列的大量出现(例如在人类基因组中含有50%以上的重复序列)。重复序列的存在将会产生大量错误的重叠,最终导致结果的严重偏差。目前已经出现很多用于shotgun片段拼接的工具,比较流行的有Phrap、CAP3、TIGER、Celera assembler等[3,4,6]。在处理重复片段时,都是采用对大量的片段数据进行反复迭代的方法,此间还需要加入很多人工的经验分析和干预,一定程度上增加了拼接所花费的时间[1,8],降低了机器的使用效率。

在拼接前的预处理工作(即将重复序列屏蔽起来)对于提高序列拼接的精确度和减少错误率显得很重要。现今流行的重复序列屏蔽的拼接方法有很多,譬如REPS[6]等。本文提出了一种新的方法来分析和处理重复序列,即采用基于k-mer子串快速确定重复序列的位置。这种方法适用于任何遵循如下几个步骤的序列拼接方法的预处理中:①比较片断集合中的所有片断,获得可能存在的重叠部分(Overlap);②建立所有片断间的相互组合关系(Layout),以片断为顶点,重叠的片断相互连接;③构建组合关系上的全长序列(Consensus)。

本文针对目前大多数拼接程序对于重复段的处理采用效率较低的反复迭代方法的特点,提出了基于k-mer子串的重复段分析方法,充分考虑了拼接中可能的分割点,并且设计与分析了识别重复序列并提高序列一致性的算法。

2 改进的重复片段分析方法

DNA序列和每一个片段序列都可以看作是字符集$\{A, C, T, G\}$上的字符串,每个长为k的字符串称为k-mer子串;对于随机打断产生的片段称之为insert,它的两端依靠目前的测序技术大约可以直接测出600bp长的序列,称为read;通过片段间相互重叠关系产生的连续片断集合称为contig。通过拼接产生的序列单元称为scaffold(支架)或supercontig,在最好的情况下,它近似等同于原始的基因序列。

2.1 算法的主要思想

从海量的序列中寻找重复序列(repeat),传统的方法有两种:一种是与已有重复段序列库进行比较判定。因为目前已经确定的重复段序列不全,无法识别所有的重复序列,实际应

*)国家自然科学基金重点项目(60433020)、国家留学基金资助。王 磊 研究生,研究方向:算法理论、大型数据库系统;张祖平 副教授,博士生,主要研究领域为算法理论、计算机网络与优化及大型数据库系统;陈建二 博士,教授,博士生导师、长江学者特聘教授,主要研究领域为计算机网络、计算机优化和计算机图形学等。

用效果不是很好;第二种是应用序列比对[7]。因为不知道重复序列是在哪一个位点开始,又是在哪个位点结束,所以用序列比对的方法耗费的计算时间很长,每两条基因序列耗费时间大约为 $O(L^2)$(L 为序列的长度)。由于在生物基因序列中,L 非常大,这样的时间复杂度显然不能满足实际需要。为了节约计算时间,就需要能够快速识别重复序列的方法。其实质也就是快速识别重复序列可能出现的位点。有了这种位置信息,再来寻找重复序列就容易得多。本着这个思想,利用基于 k-mer 子串的重复段分析方法,能快速识别可能的重复序列位点,计算时间可以得到较大的减少。

算法可以描述如下:

Input:已经过测序的 Shotgun 片断。

Output:识别出的原来序列中的重复段序列和其在原序列中的位置。

1)对所有的 Shotgun 片段数据扫描,计算每个 k 子串在所有片断中出现的次数 D;

2)如果 $D \geqslant T$,则此 k 子串标记为 repeat,并且用图来直观表示,图中每个结点表示 scaffold,边表示在两个不同 scaffold 中都出现的重复序列;

3)对每个 repeat,确定它在 Shotgun 集合中序列上可能的位置;

4)根据 2)中的 repeat 可能位置,判别 repeat 是否真的是序列中的子串,我们采用了两种方法进行判断:

①根据重复段区域序列覆盖率异常高的特性,按照实际经验选取某个基准量,当序列覆盖率超过了这个值即认为该区域是真正的 repeat;

②因为重复段与多个不同 scaffold 相连的特性,可以根据这些 scaffold 之间的距离判断其是否是真的 repeat。

5)将片段集合中的序列上的 repeat 进行屏蔽。

下面就其中几个较为复杂的步骤进行介绍。

2.2 扫描基于 k-mer 子串的重复序列

在拼接中扫描所有 k-mer 子串,按照其字符顺序放在一个排好序的表中(按字符在字母表中的顺序排列,如果首字符相同,则按第二个字符排列,依此类推),方便快速查找在序列中多次出现子串的位置。计算每个 k-mer 子串在所有片断中出现的次数 D(我们称之为深度,Depth),如果 $D \geqslant T$(T 为认定某一 k-mer 子串为 repeat 的阈值,它的取值与原来基因序列的覆盖 C(Coverage)和重复序列在原来序列中出现的次数 N(Copy Number)有关,如果 $C=8$(即 8 重覆盖),N 至少为 2,那么 T 的值至少为 $C \times N=16$,则将此 k-mer 子串标记为 repeat。这些就是重复段序列,它们在原来的基因序列中存在多个精确的拷贝。

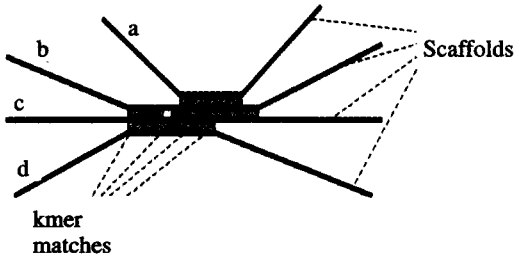
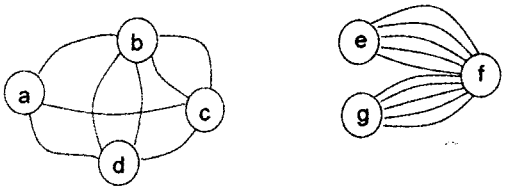


图 1 序列拼接中不同 scaffolds 间 k-mer 子串匹配

为此,我们建立如图 2a)来表示这种结构:用图 2a)中每一个节点表示一个支架 scaffold,节点之间的边通过相匹配的间隔区间(重复段)来表示。如图 2a)所示,它对应的是图 1 中不同 scaffold 间的匹配关系。



a) a)

图 2 a)对应于图 1 的连通图,其中图 1 的重叠区域对应于图中的边;

b)表示两个单模标本分别在 e,f,g 三个 scaffolds 中进行了拼接

通过对图中节点遍历,重复序列的结构就能标识出来。如果某个节点有许多到其它节点的连接(或者许多到同一个节点的连接,但是来自于不同的位点),就表示散布的重复序列;在这种情况下,重复段区域就可以根据图的连通性进行辨别,并可以按重复序列"家族"归类。然而,如果有些区域中一个节点只对应另外一个节点,并且它们对应的重复区域在整个支架中均匀分布,如图 2b)所示,那么这就可以看作是一个片段间的复制,或者是人工的拼接过程,在这个过程中,一个多态基因组的两个单模标本已经被分离地拼接在一起了。在拼接中遇到这种情况,我们需要将其排除,以提高序列一致性。

2.3 识别重复序列可能所在的位点

我们可以标识出潜在重复段区域边界,并且在拼接时不考虑通过这些区域的情况。当一个 read 能通过与 read x 和 read y 重叠向右(或者向左)延伸,但是 x 和 y 两者互相不重叠,那么我们就将这段区域标为潜在重复段区域。如果仅知某个 read r 和它右侧多个不同 read 重叠,我们不会把它与其中任何 read 合并而向右延伸,因为我们根本不知道 r 应该与哪个 read 合并。相反,如果 r 右端所有相邻的 read 都互相重叠的话,r 就能比较放心地与距离它最近的那条 read 合并。如图 3 所示,read r 与 read x 和 read y 重叠,但 x 和 y 互相不重叠,x 和 y 的最右端不同,那么图 3 中区域 R(从 x 和 y 中的某一点开始,包含整个 r)就是所说的潜在重复段区域。在拼接时应先排除这些标记的区域。

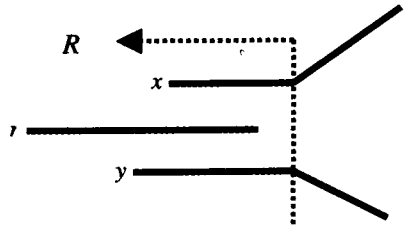


图 3 潜在重复段边界

在实践中我们发现,因为测序错误而非重复段也能导致这种情况的重叠发生,所以我们称之为潜在重复段边界。因此,怎么样区别潜在重复段边界是否真的是 repeat,又成为一个新的亟待解决的问题。

当拼接到潜在重复段区域边界时,其中一些是真正的重复段,也就是来自于原来序列中不同位点的相同序列合并在一起的区域。这样的重复段可以通过两种方式进行识别。第一种,在这些区域往往会有异常高的覆盖。这可以使用 log-

odds 比率(在一个 contig 中给定长度和密度并表示唯一序列的 read 与来自于一个重复段中至少两个拷贝的 read 间的比值)。

当 read 取样都是一致时,那么可以认为片段起点的出现满足泊松分布。

假设某个 unitig(唯一的 contig)有 N 个片段,整个基因组长为 L,对一个包含 k 个片段并且它的第一个片段起点和最后一个片断起点之间的距离为 P,那么在没有出现重复的 unitig 情况下,在 P 中可以看到 $k-1$ 个起点的概率是$[(P*N/L)^k/k!]\exp(-P*N/L)$;而当 contig 是两个重复段的拷贝互相重叠的情况下,概率是$[(2P*N/L)^k/k!]\exp(-2P*N/L)$。这两种情况取自然对数,得到一个统计量,称为 A statistic[1]:

$$P*N/L-(\log 2)*k$$

根据经验,当 A statistic 值不小于 10 时,可以认为此 unitig 不含重复段(序列在某一区域的覆盖越高,这个统计量的值越小)。

第二种,主要适用于大的 supercontig 或 contig 中的重复序列识别。容易知道,包含重复序列 contig 通常会与多个不同的非重叠 contig 相连,反映了与基因组中重复段相连的多个区域。按照下面的规则[5]可以将重复区域进行识别。

规则 1 令 d(A,B)表示 contig A 和 contig B 之间相隔的距离(用负数表示),ERR(A,B)表示距离的标准偏差。如果 $d(A,B)<-2000-4ERR(A,B)$,那么 contig A 和 contig B 被标记为 repeat。这是因为如果 A 和 B 相连,而它们估计的相重叠长度太大,我们有理由相信它们是重复段的两个拷贝。

规则 2 类似地,如果 contig A 分别与 contig B 和 contig C 相连,通过求 d(A,B)和 d(A,C),我们可以估算出 B 和 C 之间的缺口或重叠的长度 d(B,C),同时也能计算出 d(B,C)的标准偏差 ERR(B,C),如果 $d(B,C)<-2000-4ERR(B,C)$,那么将 contig A 标记为 repeat。

在拼接过程中,完全包含于其他 read 中的 read 我们称之为 sub-read,容易知道它们并不影响拼接的结果。因此在拼接时,利用 k-mer 子串的方法可以将其排除,有效地提高了拼接的速度。

2.4 方法的计算复杂性

为了便于讨论,定义下列符号:n 为要拼接的片段总数,l 为片段长度。方法的主要过程是对所有片段和 k-mer 子串进行扫描或查找。因为基因序列由 4 个字符组成,所以总的 k-mer 子串数最多为 4^k。每个 k-mer 子串要和每个片段的每个字符进行比较,因此经过估算可以得到其复杂性为 $O(n\cdot l\cdot 4^k)$。但考虑到最终 l 和 k 均有上界,故复杂性进一步简化为 $O(n)$。对于 DNA 基因序列而言,其长度 L 的值很大,而我们所获取的片段数目 n 不会超过这个值。在最坏情况下,n 和 L 在一个数量级上。因此,$O(n)$与先前的 $O(L^2)$相比,是一个相当理想的结果。

3 实验结果及分析

就以上内容,用计算机模拟进行了验证。所用的 DNA 序列来自 NISC 数据库(www. nisc. nih. gov),包括了 mouse 和 human 等几种不同长度基因组序列。由于计算机资源有限,只取了序列中的一部分(不超过 10MB)。为方便起见,将我们采用的方法称为 KmerRepeat,用它与目前应用比较广泛的重复段处理方法 RECON[10]得到的结果进行比较,结果如图 4 所示。

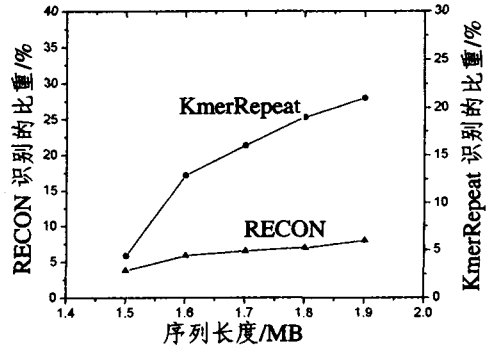


图 4 两种方法识别的重复序列结果比较

图 4 中的结果显示,KmerRepeat 识别重复段的能力明显强于 RECON,但其不足是不能识别出足够多的重复段(80%以上)。而且随着序列长度的增加,方法识别重复序列的能力并没有显著增加,因而实际应用效果并不明显。

其次,统计了 CPU 运行时间。为了方便比较,我们选取 human 基因组中的 5 段不同长度序列进行验证,运行时间如图 5 所示。

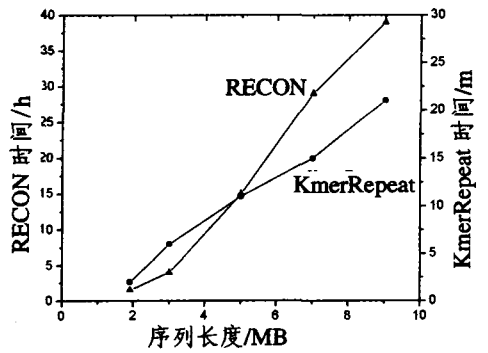


图 5 两种方法运行时间比较

所有实验均在 Intel Pentium4 2. 4GHz Processor,512MB Memory 的 IBM 机器上完成。

由图 5 可以看出,KmerRepeat 计算速度要快于 RECON,因而它特别适用于大的、重复序列较多的基因组,而且运行时间与序列的长度有明显的线性关系,这也进一步验证了时间复杂度的分析结果。

结束语 利用 DNA 片段的 k-mer 子串,可以以较少的处理代价很好地确定序列拼接中比较棘手的重复序列区域。首先将所有可能出现的重复段位置进行标记,针对这些区域再采用两种方式判定是否是真正的重复序列,即采用了双重过滤的办法。这样,较为完整地处理了原来基因序列中的重复序列,减少拼接时无谓的序列比对和重叠。另外,利用 k-mer 子串较好地排除了不影响拼接结果的 sub-read,提高了拼接效率,为以后序列的正确拼接打下了坚实的基础。同时经过分析算法的时间复杂度也由 $O(L^2)$提高到了 $O(n)$(L、n 分别表示基因序列的长度和要拼接的片段数目,一般情况下,$L>>n$;最坏情况下,L 与 n 在一个数量级上),从而大大提高了拼接算法的效率。

但存在的问题是还不能识别足够多的重复段序列(80%以上),精确度还不够高(false-positives 率即不是重复序列却被认为是重复序列的概率不超过 0. 1%),原因可能是由于我

(下转第 170 页)

4 M_{adder}编码的示例

在基于DNA自动机的二进制串行加法中，用到了3种限制性核酸内切酶，其中*FokI*的识别位点是5'-*GGATG*-3'，剪切位点是(9,13)；*BsmFI*的识别位点是5'-*GGGAC*-3'，剪切位点是(10,14)；*SmiI*的识别位点是5'-*ATTTAAAT*-3'，剪切位点是识别位点内部1/2位置。为了验证基于DNA自动机的二进制串行加法的可行性，我们在表3(a)给出了M_{adder}的所有符号的DNA编码，表3(b)给出了转移函数的DNA编码，表3(c)给出了检测分子的DNA编码。基于已有研究成果[4]，我们不难得到现有的生物技术可以实现基于M_{adder}DNA编码的二进制串行加法运算的结论。

表3(a) M_{adder}符号的DNA编码

(注：T是结束处理的终止符)

0	1	T
ACGAAG	ACGACA	GTATGA
TGCTTC	TGCTGT	CATACT

表3(b) M_{adder}转移函数的DNA编码

(注：编码中方框内数字代表满足条件的碱基序列，这些碱基与相邻序列连接后不含*FokI*识别位点)

转移函数	DNA编码
$Tran(S_0,0,0)=(S_0,0)$	ACGAAC [7] CATCCGGATG TGA TC GTAGGCCTAC ACT GCTT
$Tran(S_0,0,1)=(S_1,0)$	ACGAAG [7] CATCCGGATGTG TC GTAGGCCTACAC GCTG
$Tran(S_1,0,0)=(S_0,1)$	ACGAAG [7] CATCCGGATGTGA TC GTAGGCCTACACTCTTC
$Tran(S_1,0,1)=(S_1,1)$	GACA [9] CATCCGGATGTG GTAGGCCTACACGCTG
$Tran(S_0,1,0)=(S_0,1)$	GAAG [9] CATCCGGATGTGA GTAGGCCTACACTGCTT

表3(c) M_{adder}结果检测分子的DNA编码

(注：编码中方框内数字代表满足条件的碱基序列，这些碱基与相邻序列连接后不含*BsmFI*和*SmiI*识别位点)

终止状态	栈顶元素	DNA编码
S_1	0	ACGAGGTATGA TGGTCCC ATTTAAATGGGAC [10] TACT TC CATACT ACCAGGG TAAATTTACCCTG
S_1	1	GACA GTATGATGGTCCC ATTTAAATGGGAC [10] ATAC CTATCT ACCAGGG TAAATTTACCCTG
S_0	1	GACTCTATGATGGTCCC ATTTAAATGGGAC [10] ATAC CATACT ACCAGGGTAAATTTACCCTG
S_0	0	ACGAAGGTATGATG GTCCC ATTTAAAT GGGAC [10] ATAC TCCATACTAC CAGGGTAAATTTACCCTG

结束语 本文提出了一种基于DNA自动机的n位二进制串行进位加法的模型。该模型类似于电子计算机中加法的处理过程，操作数的编码规则简单；实现运算的生物操作在实验室简单可行，多位二进制数的加法仅仅是一位二进制数加法的简单循环。本文为DNA计算机中加法的实现提供了一种可行的模型，通过选择合适的编码，该模型可进一步推广到乘法、除法等其他算术运算，还可以推广到数的补码运算等，对研究DNA计算机的运算系统具有十分重要的意义。

参考文献

1 Guarnieri F, Fliss M, Bancroft C. Making DNA add [J]. Science, 1996, 273: 220～223

2 Guarnieri F, Bancroft C. Use of a horizontal chain reaction for DNA-based addition. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1999, 44: 105～111

3 Yurke B, et al. DNA implementation of addition in which the input strands are separate from the operator strands [J]. Biosystems, 1999, 52: 165～174

4 Benenson Y, et al. Programmable and autonomous computing machine made of biomolecules [J]. Nature, 2001, 414: 430～434

5 Li W-G, Ding Y-S, Huang Z-D, et al. Stack-type data structure for DNA-based computer. The 11th International Meeting on DNA Computing, London, Ontario, Canada, June, 2005 (Poster)

6 Ding Y-S, Shao S-H, Ren L-H. DNA Computing and Soft Computing. Beijing: Scientific Publishing House, 2002 (in Chinese)

7 许进，张雷. DNA计算机原理、进展及难点(Ⅰ)：生物计算系统及其在图论中的应用. 计算机学报，2003，26(1)：1～11

(上接第166页)

们要求必须是连续而且精确的k-mer子串，影响了算法的应用。

进一步的工作是通过非精确或非连续的k-mer子串以提高算法的灵敏度和研究基于k-mer子串的片段拼接算法及海量片段数据的大规模并行处理技术。

参考文献

1 Myers E W, Sutton G G, Dew LM, et al. A Whole-Genome Assembly of Drosophila [J]. Science, 2000, 287: 2196～2204

2 International Human Genome Sequencing Con- sortium. Initial Sequencing and Analysis of the Human Genome [J]. Nature, 2001, 409: 860～864

3 Kececioglu J D, Meyers E W. Combinatorial Al- gorithms for DNA Sequencing Assembly [J]. Algorithmica, 1995, 13: 7～15

4 Pevzner P A, Tang Haixu, Waterman M S. An Eul -erian Path Approach to DNA Fragment Assembly [J]. Proceedings of National Academy of Sciences, 2001, 98: 9487～9753

5 Batzoglou S, Jaffe D B, Stanley K, et al. ARACHNE: A Whole-Genome Shotgun Asse -mbler In: Proceedings of the Ninth Annual International Conference in Computational Mole- cular Biology (RECOMB), May 2005, Cam bridge, 2005. 177～189

6 Setuball J C, Werneck R F. A Program for Build- ing Contig Scaffolds in Double-barrelled Shotgun Genome Sequencing: [Institute of Computing Technical Report IC-01-05]. Unicamp, 2001

7 张博峰，王正华. DNA片段拼接中基于定长特征子串的重复序列信息屏蔽方法[J]. 国防科技大学学报，2002，24(6)：67～70

8 张春霆. 生物信息学的现状与展望[J]. 世界科技研究与发展，2000，22(6)：17～20

9 涂俐兰，王能超，等. 生物序列拼接及其算法[J]. 生命科学研究，2003，7(2)：79～80

10 Bao, Eddy. 2002. RECON documentation. http://www.genetics.wustl.edu/eddy/recon