ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.6, June 2007, pp.1298−1307
DOI: 10.1360/jos181298

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于彩色编码技术的基序发现算法*

王建新+, 黄元南, 陈建二

(中南大学 信息科学与工程学院,湖南 长沙 410083)

A Motif Finding Algorithm Based on Color Coding Technology

WANG Jian-Xin+, HUANG Yuan-Nan, CHEN Jian-Er

(College of Information Science and Engineering, Central South University, Changsha 410083, China)

+ Corresponding author: Phn: +86-731-8830212, Fax: +86-731-8830212, E-mail: jxwang@mail.csu.edu.cn

Wang JX, Huang YN, Chen JE. A motif finding algorithm based on color coding technology. *Journal of Software*, 2007,18(6):1298−1307. http://www.jos.org.cn/1000-9825/18/1298.htm

Abstract: Finding common pattern, motifs or signals, in a set of DNA sequences is an important problem in computational biology. Recently, some biologists extremely focus on the (*l*,*d*)-(*K*-*k*) motif finding problem when the number of sequences *K* is 20 and the number of sequences with instances *k* is 16, (*l*,*d*)-(20-16) problem for short. For solving this problem, this paper introduces a novel sample-driven algorithm (SDA), called color coding motif finding algorithm, CCMF for short. It uses color coding technology to converse a (*l*,*d*)-(20-16) problem to some (*l*,*d*)-(16-16) problems, then uses divide-and-conquer and branch-and-bound approaches to solve this (*l*,*d*)-(16-16) problem. Using the conversion process can reduce 4 845 combinations to 403 colorings, while increasing the running rate enormously. The experimental results on synthetic and real datasets show that the CCMF algorithm can accurately and efficiently find all (*l*,*d*)-(20-16) patterns and instances. Its comprehensive performances in finding motifs are superior to those of other existing algorithms. It is applicable for real biological purpose. The color coding technology can also be used to improve the performances of other similar (*l*,*d*)-(*K*-*k*) problems when *k* is less than *K*.

Key words: color coding technique; motif finding; coloring; (*l*,*d*)-(*K*-*k*) problem; algorithm optimization

摘 要: 从 DNA 序列中发现基序是生物计算中的一个重要问题,序列条数 *K*=20 包含基序用例的序列条数 *k*=16 的(*l*,*d*)-(*K*-*k*)问题(记作(*l*,*d*)-(20-16)问题)是目前生物学家十分关注的基序发现问题.针对该问题提出了一种基于彩色编码技术的 SDA(sample-driven algorithm)搜索算法——彩色编码基序搜索算法(color coding motif finding algorithm,简称 CCMF 算法).它利用彩色编码技术将该问题转化为(*l*,*d*)-(16-16)问题,再采用分治算法和分支定界法来求解.在解决将(*l*,*d*)-(20-16)问题转化为(*l*,*d*)-(16-16)问题时,CCMF 算法利用彩色编码技术将 4 845 个组合降低到 403 个着色,这将极大地提高算法的整体运行效率.使用模拟数据和生物数据进行测试的结果表

* Supported by the National Natural Science Foundation of China under Grant Nos.30370418, 90209008, 60302016, 60532050, 30500131 (国家自然科学基金); the Project for National Science Fund for Distinguished Young Scholars of China under Grant No.60225008 (国家杰出青年科学基金); the Joint Research Fund for Overseas Chinese Young Scholars under Grant No.30528027 (海外杰出青年研究基金); the Beijing Natural Science Foundation under Grant Nos.4051002, 4042024 (北京市自然科学基金)

Received 2006-06-26; Accepted 2006-08-29

明,CCMF 算法能够快速发现所有(*l*,*d*)-(20-16)问题的基序模型和基序用例,具有优于其他算法的综合性能评价,能够用于真实的基序发现问题.同时,通过修改着色方案,CCMF 算法可以用于求解一般的(*l*,*d*)-(*K*-*k*)问题,其中,*k*<*K*.

关键词: 彩色编码技术;基序发现;着色;(*l*,*d*)-(*K*-*k*)问题;算法优化

中图法分类号: TP301 文献标识码: A

在 DNA 序列中存在一些具有特定保守功能且长度最多为 30 个碱基的序列片段,如转录因子绑定位点.通常,人们称这些序列片段为基序(motif),也可称为模型(pattern)或信号(signal).由于在生物进化的过程中,DNA 序列中存在的基序用例中有部分碱基会发生替换的现象,因此,基序模型与基序用例之间存在一定的差异.基序发现(motif finding)问题是在 DNA 序列中识别基序模型和基序用例的过程,它是生物信息学的一个重要的研究内容.基序发现问题的解决,有助于了解序列的功能和阐明序列之间的进化关系,并且在识别 DNA 序列的调控信号方面有着重要的应用.大量的研究工作都集中在共表达基因[1,2]或直系同源基因[3,4]的上游区域中发现转录因子绑定位点.文献[5]给出了基序发现问题的描述,文献[6,7]中也将该问题称为植入基序问题(planted (*l*,*d*)-motif problem,简称 PMP),我们可以将基序问题定义为:

定义 **1**((***l***,***d***)基序问题). 假设存在一个长度为 *l* 的基序模型 *M*,在给定的 *K* 条长度为 *L* 的 DNA 样本序列中每条序列的不同位置上植入一个基序用例,每个基序用例与基序模型 *M* 之间有 *d* 个碱基发生了替换.(*l*,*d*)基序问题是在已知 *l* 和 *d* 的情况下,找出(*l*,*d*)基序模型 *M*,并从样本序列中找到所有的基序用例.

过去很多算法[1−4]是针对(*l*,*d*)基序问题设计的,对于基序模型 *M* 的长度 *l* 较小(≤20)的情况,这些算法能够得到较为令人满意的结果,但(*l*,*d*)基序问题的解决并不能很好地满足现实生物基序发现的要求.由于生物实验中存在的噪音和误差,使得样本序列中出现部分序列不包含基序用例或者包含多条序列的情况,生物学家经常无法采集到一组能够保证每条序列正好包含一个基序用例的样本序列[8],在所有样本序列的部分序列中寻找基序用例和基序模型成为研究的热点.同时,生物学家通常无法确定基序用例与基序模型之间的差异,即每个基序用例与基序模型之间发生替换的碱基个数 *d*.在很多情况下,他们只能知道这个差异的大致范围,因此,寻找至多 *d* 个碱基发生了替换而不是恰好 *d* 个碱基发生了替换的基序模型能够更好地满足实际基序发现问题的要求[6,7].因此,文献[6,7]将(*l*,*d*)基序问题进行了扩展,扩展的(*l*,*d*)-*k* 基序问题的定义描述如下:

定义 **2**(扩展的(***l***,***d***)-***k*** 基序问题). 假设存在一个长度为 *l* 的基序模型 *M*,在给定的 *K* 条长度为 *L* 的 DNA 样本序列中的至少 *k*(*k*≤*K*)条序列中植入基序用例,每个基序用例与基序模型 *M* 之间至多有 *d* 个碱基发生了替换.扩展的(*l*,*d*)-*k* 基序问题是在已知 *k*,*l* 和 *d* 的情况下,找出基序模型 *M*,并从样本序列中找到所有的基序用例.

本文主要研究在 *k*<*K* 的情况下扩展的(*l*,*d*)-*k* 基序问题.为了描述方便,我们将这个问题记为(*l*,*d*)-(*K*-*k*)问题.

本文第 1 节介绍(*l*,*d*)基序问题和(*l*,*d*)-(*K*-*k*)基序问题的模型.第 2 节概述基序问题模型的研究现状.第 3 节详细描述彩色编码基序搜索算法(CCMF 算法)的思想和实现细节.第 4 节是采用模拟数据和真实的生物数据对 CCMF 算法和其他算法进行测试与比较分析.第 5 节总结本文内容.

1 研究现状

已有的基序发现问题的算法可分为两类:基于模式的搜索算法(pattern-driven algorithm,简称 PDA)和基于样本的搜索算法(sample-driven algorithm,简称 SDA)[9].PDA 需要检验 4^l 个长度为 *l* 的序列片断(*l*-mer),对它们在序列样本中出现的用例计分,并找到得分最好的基序模型.理论上,寻找基序模型的最精确的方法是 PDA[9],敏感性好,可得到最优结果.PDA 的不足之处在于:当 *l* 比较大的时候,4^l 的搜索空间就变得很大,算法的执行效率迅速降低.SDA 的基本思想是:对样本序列出现的所有 *l*-mers,计算出它的满足一定条件的邻域片断集合,并在这个集合中搜索基序模型.SDA 的搜索空间至多为 $KLC_l^d 3^d$,其中,C_l^d 为从 *l*-mer 中选取 *d* 个碱基的组合数.显然,SDA 的搜索空间远远小于 PDA 的搜索空间,因此能够获得较高的搜索效率.目前较为有效的算法都是基于 SDA 的思想,只不过在减小搜索空间的策略上各不相同.然而,采用了启发式规则的搜索算法都存在不同程度的丢失

最优解的可能.

YMF[10],Oligo-analysis[1]和 MDscan[4]都是较好的求解 PDA 穷举搜索算法.YMF 需要输入基序的特征信息,且要求 $l\in[0,11]$,$d\le2$;Oligo-analysis 只能适用于 $l\le6$ 的基序搜索;MDscan 要求 $l\in[5,15]$,$d\le3$.因此,这些算法都不能解决 l 和 d 较长的情况.

2002 年,Eleazar 和 Pevzner 提出的 MITRA[11]是执行效率较好的 SDA 穷举搜索算法,它在样本中的所有 l-mers 的邻域中搜索基序模型和基序用例.虽然只要基序模型 M 存在,MITRA 就能够确保找到最佳的基序模型,但所需的计算时间远远多于其他算法,求解(15,4)需耗时约 300 秒.

一些采用了启发式规则的 SDA 算法,如 2000 年 Pevzner 和 Sze 提出采用图论方法的 WINNOWER 算法[5]、2002 年 Buhler 和 Tompa 提出的采用随机映射技术的 PROJECTION 算法[12]、2003 年 Alkes,Sriram 和 Pevzner 提出采用分支定界技术的 PatternBranching 算法[12]、2004 年 Sze 等人提出分治和分支定界算法[13]都能够较好地解决(15,4)问题.2004 年,Styczynski 等人提出的是一种在图中搜索所有最大团的算法,是目前已知求解 (l,d)-$(K$-$k)$问题准确性最高的算法[6].2005 年提出的 Voting 算法[7]采用投票计分机制识别基序,它能在不限定 k 的情况下找出基序用例个数最多的基序模型.

上述大部分算法是针对解决(l,d)基序问题而提出的,其中只有极少数算法能用于(l,d)-$(K$-$k)$问题的求解.文献[6]提出的算法可以解决(l,d)-$(K$-$k)$问题,但其运行时间很长,在实际中无法具体应用.

通常,生物学家用微阵列(micoarray)或者 CHIp-Chip(chromatin immunoprecipitation combined with microarray chip hybridization)技术进行实验,通过实验可以得到一组可能包含同一基序模型用例的样本序列[7].近期,人们对基序发现问题的研究主要是围绕(l,d)-$(K$-$k)$问题进行,其中,(l,d)-(20-16)问题更是目前生物学家十分关注的基序发现问题.本文针对(l,d)-(20-16)问题提出了一种基于彩色编码技术的 SDA 搜索算法——彩色编码基序搜索算法(color coding motif finding algorithm,简称 CCMF 算法).它利用彩色编码技术将(l,d)-(20-16)问题转化为(l,d)-(16-16)问题,再用分治法和分支定界算法来求解(l,d)-(16-16)问题.使用彩色编码技术的优势是:在保证正确找到基序模型的基础上,能够有效地减少搜索空间,提高执行效率.对算法的理论分析和实验测试表明,CCMF 能够完全地识别出样本序列中所有基序模型,并具有良好的执行效率,及理想的总体评价,并能应用在实际的模式发现问题中.

2 CCMF 算法

CCMF 算法的目标是在 20 条长度为 L 的样本序列中寻找所有(l,d)-(20-16)的基序模型和基序用例.与其他采用图论方法的算法[5,6,13]类似,CCMF 算法也是将寻找基序问题转化为在图中找团的问题进行求解.

2.1 以组合方式构造16分图

为了方便讨论,首先给出关于 K 分图(又称为 K 部图(K-partite graph))的定义[14]:

定义 **3**($\boldsymbol{K}$ 分图). 设 $G_K=(V,E)$为一个无向图,若 $V=\{V_{(1)},\ldots,V_{(K)}\}$,其中,$V_{(i)}$为第 i 个结点部集,使得 $V_{(1)}\cup\ldots\cup V_{(K)}=V$,$V_{(1)}\cap\ldots\cap V_{(K)}=\varnothing$,且 G_K 中的每条边的两个端点不属于同一个结点部集,则称 G_K 为 K 分图,记作 $G_K=(\{V_{(1)},\ldots,V_{(K)}\},E)$.

CCMF 算法首先根据给定的 20 条样本序列构造 20 分图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$.$G_{20}$ 的构造过程为:将样本序列中每个 l-mer 视为一个结点.第 i 条序列起始位置为 j 的 l-mer 记作 v_{ij}.每个 $V_{(i)}$中包含序列 i 上所有 $N=L-l+1$ 个结点,则 G_{20} 图中共有 $20\times N$ 个结点.$D_H(v_{ij},v_{pq}),i\ne p$ 表示点 v_{ij} 和 v_{pq} 之间的海明距离,若 $D_H(v_{ij},v_{pq})\le2d$,其中 $i<p$,则$(v_{ij},v_{pq})\in E$.

文献[15]中所提出的算法采用分治算法和分支定界法有效地解决了在图 G_{20} 中寻找 20 团的问题,但其无法求解(l,d)-(20-16)问题.通过简单的变换,该方法可以应用到求解(l,d)-(16-16)问题,所以 CCMF 算法首先考虑将(l,d)-(20-16)问题转换为(l,d)-(16-16)问题,也就是说,需要将图 G_{20} 转换成图 G_{16},从而采用分治算法和分支定界法解决在图 G_{16} 中寻找 16 团的问题.

下面给出两个组合数学中关于组合的定义[15]:

定义 **4**((***K*-*k***)组合). 从 K 分图 $G_K=(\{V_{(1)},\ldots,V_{(K)}\},E)$的 K 个不同的结点部集中取 k 个结点部集合并成一个结点集 $V'=\{V'_{(1)},\ldots,V'_{(k)}\}$($k\le K$),称为从 K 分图 G_K 的结点集中取出 k 个结点部集的一个组合,记作(K-k)组合.

定义 **5**((***K*-*k***)组合数). 从 K 分图 $G_K=(\{V_{(1)},\ldots,V_{(K)}\},E)$的 K 个不同的结点部集中取出 k 个结点部集的所有不同组合的个数,称为从 K 分图 G_K 的结点集中取出 k 个结点部集的组合数,记作 C_K^k.

在 20 分图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$中任意选取 16 个结点部集 $V_{(i)}$可以构成一个 16 分图 G_{16}.明显可知,采用组合方式将存在 $C_K^k=C_{20}^{16}=4845$ 个不同的 16 分图 G_{16}.如果试图在每个 16 分图 G_{16} 中去寻找 16 团,将需要进行 4 845 次采用分治算法和分支定界算法求解在 16 分图 G_{16} 中寻找 16 团的问题的算法,这需要很长的运行时间.这个用组合方式求解(l,d)-(20-16)问题的算法称为组合基序搜索算法(combination motif finding algorithm,简称 CBMF 算法).本文将采用彩色编码方案来降低(20-16)组合数,并在保证覆盖所有(20-16)组合的条件下提高算法的执行效率.

2.2 以着色方式构造16分图

1995 年,Alon 等人提出的彩色编码方案[16]是通过散列公式 f 将 $Q=\{1,\ldots,K\}$完全映射到 $W=\{1,\ldots,k\}$的方案,使得 W 中每个元素 x 都能映射到 Q 的不同 $f(x)$上,称 f 为 W 的完全散列公式.若通过 f 对 Q 中每个元素着色,那么通过 f 映射后 W 中没有相同颜色的元素.

文献[16,17]提出采用彩色编码方案可求解 k-PATH,LOG PATH 等 NP 难的问题.目前,这一技术已被应用在蛋白质调控网格[18,19]和子图匹配[17,19]等生物计算问题中.本文采用彩色编码方案将 20 分图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$完全映射到 16 分图 G_{16}.

下面给出两个关于着色的定义:

定义 **6**((***K*-*k***)着色). 用 k 种颜色$\{c_1,\ldots,c_k\}$对 K 分图 $G_K=(\{V_{(1)},\ldots,V_{(K)}\},E)$的结点集进行着色($k\le K$),每种颜色至少使用了$\left\lfloor\frac{K}{k}\right\rfloor$次,每种颜色最多使用了$\left\lceil\frac{K}{k}\right\rceil$次,着色后,结点部集对应的颜色为$\{vc_1,\ldots,vc_K\}$,$vc_i\in\{c_1,\ldots,c_k\}$,$i\in[1,K]$,称为对 K 分图进行 k 着色,记作(K-k)着色,称 k 为 G_K 的色数,称$\{vc_1,\ldots,vc_K\}$为一个(K-k)着色.

定义 **7**(覆盖). 若用一个(K-k)着色$\{vc_1,...,vc_K\}$对 K 分图 $G_K=(\{V_{(1)},\ldots,V_{(K)}\},E)$进行着色后,从$\{V_{(1)},\ldots,V_{(K)}\}$中任选 k 个部集组成一个(K-k)组合$\{V'_{(1)},\ldots,V'_{(k)}\}$,若其中结点部集所着颜色互不相同,则称($K$-$k$)着色$\{vc_1,\ldots,vc_K\}$覆盖($K$-$k$)组合$\{V'_{(1)},\ldots,V'_{(k)}\}$.

定义 **8**((***K*-*k***)着色方案). 对 K 分图进行 k 着色的一个可能覆盖所有(K-k)组合的(K-k)着色集合($k\le K$),称为对 K 分图进行 k 着色的着色方案,简称为(K-k)着色方案,记作 $Colorings_k^K=\{VC_1,\ldots,VC_{mc}\}$,其中,$VC_i$ 为第 i 个(K-k)着色, $i\in[1,mc]$,mc 为着色数,即 $mc=|Colorings_k^K|$.

本文需要设计可覆盖所有(20-16)组合的(20-16)着色方案.彩色编码方案设计的基本原则是采用尽可能少的着色覆盖所有组合.

为了更好地说明(20-16)着色方案的设计方法,给出以下结论:

命题 **1**. 对 20 分图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$进行 16 着色,必定有一种颜色至少着 2 个结点部集,也必定有一种颜色最多着 1 个结点部集.

证明:根据以下鸽笼原理(pigeonhole principle)的推论可轻易证明:若用 n 种颜色将 m 个结点部集着色,那么必定存在一种颜色至少着了$\left\lceil\frac{m}{n}\right\rceil$个结点部集,必定存在一种颜色最多着了$\left\lfloor\frac{m}{n}\right\rfloor$个结点部集.在(20-16)着色方案中,$n$=16,$m$=20,则明显可知,用 16 种颜色$\{c_1,\ldots,c_{16}\}$对 20 个结点部集$\{V_{(1)},\ldots,V_{(20)}\}$进行着色(简称(20-16)着色),必定有一种颜色至少着了 2 个结点部集,也必定有一种颜色最多着了 1 个结点部集. □

对 20 分图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$进行 16 着色,就是求集合 $Colorings_{16}^{20}$,其目标是使得集合中的着色尽量少,即使得着色数 mc 尽量少.

CCMF 算法在着色前将相邻的 2 个结点部集划分为 1 块,20 个结点部集共有 10 块$\{B_1,\ldots,B_{10}\}$,其中,

$B_i=(V_{(2i-1)},V_{(2i)})$.

命题 **2**. 对图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$进行 16 着色后,一定有 4 个结点部集的颜色与其他结点部集的颜色有重复.

证明:由定义 6 容易得证. □

命题 **3**. 将图 $G_{20}=(\{V_{(1)},\ldots,V_{(20)}\},E)$分块后进行 16 着色,4 个着重复颜色的结点部集在块集合$\{B_1,\ldots,B_{10}\}$中的位置有且仅有以下 3 种情况:

(1) 4 个结点部集属于不同的块中;

(2) 2 个结点部集在同一块中,其他 2 个结点部集分别属于其他不同的块中;

(3) 2 个结点部集在同一块中,其他 2 个结点部集属于另一块中.

证明:假设依据命题 3 的 3 种情况产生集合 $Colorings_{16}^{20}$,其中,C_1,C_2,C_3 分别为这 3 种情况产生的着色集合.由于上述 3 种情况中包含重复颜色的结点部集分布方式相互不兼容,显然,C_1,C_2,C_3 构成 $Colorings_{16}^{20}$ 的一个划分,即 $C_1\cup C_2\cup C_3= Colorings_{16}^{20}$,且 $C_1\cap C_2=C_1\cap C_3=C_2\cap C_3=\varnothing$,命题 3 得证. □

图 1 中假设被重复使用的颜色为 a,b,c 和 d.CCMF 算法中产生 $Colorings_{16}^{20}$ 集合的过程包括以下 3 个步骤:

第 1 步.从$\{B_1,\ldots,B_{10}\}$中任选 4 个块,用$\{c_1,\ldots,c_{16}\}$种颜色分别对这 4 个块着色,使每个块中的 2 个结点部集着相同的颜色,如图 1 中的第 1 步所示;用未用的 12 种颜色分别对未着色的 12 个结点部集进行着色;这一步产生的所有着色构成 C_1 集合;

第 2 步.从$\{B_1,\ldots,B_{10}\}$中任选 2 个块,用$\{c_1,\ldots,c_{16}\}$中的 2 种颜色分别对这 2 个块着色,使每个块中的 2 个结点部集着相同的颜色;将未着色的 8 个块分成 4 组,在其中选取 1 组,用 2 个未用的颜色对该组中的 2 个块的结点部集着色,使得同一块中的结点部集着不同颜色,如图 1 中的第 2 步所示;用未用的 12 种颜色分别对未着色的 12 个结点部集进行着色;这一步产生的所有着色构成 C_2 集合;

第 3 步.将相邻的 2 个块划分一组,10 个块共有 5 个组$\{Z_1,Z_2,Z_3,Z_4,Z_5\}$,其中,$Z_i=(B_{(2i-1)},B_{(2i)}),i\in[1,5]$,并分成以下两步:

(3.1) 从$\{Z_1,Z_2,Z_3,Z_4,Z_5\}$中任选 2 组,用$\{c_1,\ldots,c_{16}\}$中的 2 种颜色对其中一组着色,使得该组内同一块中的结点部集着不同颜色;用未使用的 2 种颜色对另外一组着色,使得该组内同一块中的结点部集着不同颜色,如图 1 中的第 3 步(3.1)所示;用未使用的 12 种颜色分别对未着色的 12 个结点部集进行着色.这一步产生的所有着色构成 $C_{(3.1)}$集合;

(3.2) 将$\{Z_1,Z_2,Z_3,Z_4,Z_5\}$构造成 3 个集合$\{Z_1,Z_2\},\{Z_3,Z_4\}$和$\{Z_5,Z_1\}$,对这 3 个集合中的任意一个,用$\{c_1,\ldots,c_{16}\}$的 4 种颜色进行着色,使得集合中每个组的结点部集的颜色均不相同,如图 1 中的第 3 步(3.2)所示;用未使用的 12 种颜色分别对未着色的 12 个结点部集进行着色.这一步产生的所有着色构成 $C_{(3.2)}$集合.

综合步骤(3.1)和(3.2)产生的着色,集合 $C_3=C_{(3.1)}\cup C_{(3.2)}$.

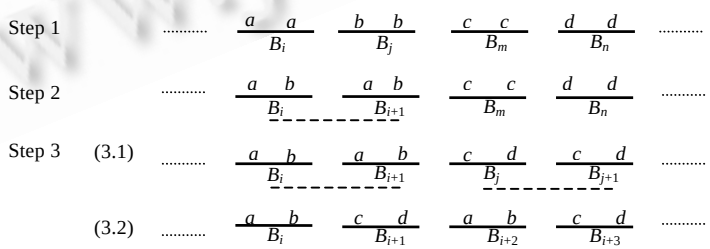

Fig.1 The steps to obtain $Colorings_{16}^{20}$

图 1 产生 $Colorings_{16}^{20}$ 集合中着色示意图

定理 **1**. CCMF 算法所产生的 $Colorings_{16}^{20}$ 集合中共包含 403 个(20-16)着色.

证明:依据产生 $Colorings_{16}^{20}$ 集合的过程可知:由第 1 步操作所产生的集合 C_1 中的着色数为$|C_1|=C_{10}^4=210$ 个;由第 2 步操作所产生的集合 C_2 中的着色数为$|C_2|=C_{10}^2\times C_4^1=180$ 个;第 3 步分两个小步完成:由(3.1)步操作所产生的集合 $C_{(3.1)}$中的着色数为$|C_{(3.1)}|=C_5^2=10$ 个;由(3.2)步操作所产生的集合 $C_{(3.2)}$中的着色数为$|C_{(3.2)}|=3$ 个,那么,集合 $Colorings_{16}^{20}$ 的着色数 mc=210+180+10+3=403 个. □

定理 **2**. CCMF 算法所产生的 $Colorings_{16}^{20}$ 集合中的 403 个(20-16)着色可完全覆盖 4 845 个(20-16)组合.

证明:依据命题 3 的 3 种情况以及产生 $Colorings_{16}^{20}$ 集合的过程可知:第 1 种情况中重复颜色的结点部集块分布有 $C_{10}^4=210$ 种子情况,其中每种可能性还包括$(C_2^1)^4=16$ 种在该情况下进一步选择元素的方法.集合 C_1 中的每个着色恰好可以覆盖其中一种相应的子情况,即可覆盖$(C_2^1)^4=16$ 个组合,则集合 C_1 共可覆盖 210×16=3360 个组合.相应地,第 2 种情况中有 $C_{10}^2\times C_4^1=180$ 种子情况,其中每种包括 $C_2^1\times(C_2^1)^2=8$ 种在该情况下进一步选择结点部集的方法,则集合 C_2 中的每个着色可覆盖 8 个组合,则集合 C_2 共可覆盖 180×8=1440 个组合.而第 3 种情况中每种可能性中包含了 4 种在该情况下进一步选择结点部集的方法;集合 $C_{(3.1)}$中的每个着色可覆盖$(C_2^1)\times(C_2^1)=4$ 个组合,则集合 $C_{(3.1)}$可覆盖 10×4=40 个组合;集合 $C_{(3.2)}$中的每个着色也可覆盖 4 个组合,但它所覆盖的组合均与 $C_{(3.1)}$中 2 个组合重合.此外,集合$\{Z_5,Z_1\}$的着色所覆盖的其他 2 个组合中还有一个与集合$\{Z_1,Z_2\}$的着色所覆盖的组合重合,所以,去除这些重合的情况后,集合 $C_{(3.2)}$可覆盖 3×2−1=5 个组合.因此,集合 C_3 共可覆盖 40+5=45 个组合.

注意到,C_1 和 C_2 中的着色覆盖的组合是没有重合的,C_3 的重合情况也已分析完毕;同时,由命题 3 可知,C_1,C_2 和 C_3 构成 $Colorings_{16}^{20}$ 的一个划分.综上分析,用这 403 个(20-16)着色能覆盖 3360+1440+45=4845 个组合. □

明显可知,用 $Colorings_{16}^{20}$ 集合中的 403 个(20-16)着色对图 G_{20} 进行着色,并将着相同颜色的结点部集合并为一个结点部集后,可以构成 403 个不同的 16 分图 G_{16}.采用上述着色方法后,CCMF 算法将只需要进行 403 次采用分治算法和分支定界法求解在 16 分图 G_{16} 中寻找 16 团的问题,相对没有采用着色方法前的 4 845 次在原始的 20 分图 G_{20} 中寻找 16 团的算法(CBMF 算法),能极大地提高其运行速度.

2.3 基序模型搜索

CCMF 算法利用上面设计好的 $Colorings_{16}^{20}$ 中的一个(20-16)着色对 G_{20} 进行着色,并将着相同颜色的结点部集合并后产生 16 分图 $G'_{16}=(\{V'_{(1)},\ldots,V'_{(16)}\},E)$;然后,用分治和分支定界算法[20]在每个 16 分图 G'_{16} 中去寻找 16 团.明显可知,由于利用 $Colorings_{16}^{20}$ 的着色将 G_{20} 转换为 G'_{16} 的过程中没有改变结点和边,所以,G_{20} 与 G'_{16} 中的 16 团是一致的.

G_{20} 的 16 团可表示 20 条序列中有 16 条序列包含候选基序用例.找到候选基序用例后,需要对其进行检验并识别基序模型,以保证每个基序用例与基序模型之间不超过 d 个不匹配的碱基,这需要求解这些候选基序用例的最近子串问题.2003 年,Lanctor 等人已经证明了判断一个序列集合中是否存在最近串的问题是 NP 难的[20].CCMF 算法采用基序用例的最近串来表示基序模型,因此,算法使用文献[16]中求最近串的算法从基序用例中检验并恢复基序模型.

由于在 20 条序列中的同一个基序模型的用例可能不只 16 个,且同一条序列中可能出现多个同一个基序模型的用例.此外,每个用例的长度也不一定都是一致的,有个别用例的长度可能超过 l 个碱基.因此,从 G_{20} 找到的所有 16 团之间存在一定的重叠性.为了消除这种重叠性,CCMF 算法在搜索所有 16 团的工作完成后需进行最后一步 *Merge-Motif*()过程,借鉴文献[6]中求最大基序的方法将搜索到的基序模型相同的用例归并在一起.

CCMF 算法的描述见算法 1.

算法 **1**. Algorithm $CCMF(S_{20})$.

Input: A set of 20 sequences $S_{20}=\{x_1,\ldots,x_{20}\}$;

Output: All motif patterns, instances and their positions in S_{20}.

$G_{20}\leftarrow Generate\text{-}Graph(S_{20})$;

$Colorings_{16}^{20}\leftarrow Build\text{-}coloring(G_{20},16)$;

For i=1 to mc

G'_{16}←coloring G_{20} with ith coloring from $Colorings_{16}^{20}$;

$Cset$←*Find-Motif*(G'_{16});

Return *Merge-Motif*(*Cset*)

CCMF 算法中构造图 G_{20} 的过程 *Generate-Graph*()的时间复杂度为 $O(KL)$,其中 K=20;产生 $Colorings_{16}^{20}$ 集合的过程 *Build-coloring*()、用 $Colorings_{16}^{20}$ 中的一个着色对 G_{20} 着色和基序合并过程 *Merge-Motif*()的时间复杂度都为 $O(1)$;*Find-Motif*()过程中使用分治和分支定界算法的时间复杂度大致为 $O(k0\times k(PL)^{k0})$[13],其中,$k0$=2 为分治法的分治系数,P 为通过分支定界剪枝规则的概率,k=16;*Find-Motif*()过程中检验和恢复基序模型的算法的时间复杂度为 $O(kL+kd\times d^d)$[14].那么,*Find-Motif*()过程的时间复杂度为 $O(k0\times k(PL)^{k0}+kL+kd\times d^d)$.CCMF 算法的时间复杂度为 $O(KL+mc\times[1+k0\times k(PL)^{k0}+kL+kd\times d^d])$,其中,$mc$=403.

由以上分析可知,CCMF 算法的运行时间主要取决于 *Find-Motif*()过程的运行时间和运行次数 mc.CCMF 算法采用了文献[13,21]中的相关技术优化了 *Find-Motif*()过程的运行时间,并主要利用彩色编码技术有效降低了 *Find-Motif*()过程的运行次数 mc,从而提高了整个算法的运行效率.

3 算法测试

我们用 C++语言实现了 CCMF 算法和 CBMF 算法,同时使用模拟数据和生物数据对 CCMF 算法和其他算法进行了测试.所有的测试在 3.0GHZ 的 Intel Xeon DP 处理器,4G 内存的机器上运行,使用 Red Hat 9.0 的操作系统.

3.1 模拟数据

测试中使用的(l,d)-(20-16)问题的模拟数据生成过程[5]是:首先随机生成 20 条长度 L 均为 600 的一组样本序列,随机产生一个长度为 l 的基序模型 M,在样本序列中随机选择 16 条序列,在每条选出的序列上随机选择位置 i($i\in[1,L-l]$),用一个基序用例替换从 i 开始的 l 个碱基,其中,每个基序用例是在基序模型 M 中随机选择 j($j\le d$)个碱基用{A,C,T,G}中任意一个碱基进行随机替换后得到.

为了比较CCMF算法的性能,在同样的环境下使用同样的测试数据对目前求解(l,d)基序问题效果最好的两种算法 PatternBranching 和 PROJECTION 以及可求解(l,d)-(K-k)问题的 Gemoda 和 Voting 进行了测试,其中,Voting[7]是求解忽略参数 k 的基序问题的算法,Gemoda 是文献[6]算法的实现,也是目前求解(l,d)-(K-k)问题正确性最高的算法.测试使用的程序均由相关文献的作者提供.Voting 使用的版本是 2005 年 5 月 20 日公布的 0.2 版.运行时,各算法参数的设置由原文提供或参照作者意见.

各种算法都要指定基序模型的长度 l 和基序用例允许发生替换的碱基个数 d 的值.PatternBranching 不能指定 k,可指定搜索基序模型个数 Num(≤5),PROJECTION 中可指定 k 和 Num,算法的运行时间将随 Num 成倍增长.为了便于比较,这两种算法的 Num 都设为 1,PROJECTION 中 k 设为 16,Voting 中忽略参数 k,默认情况下程序给出 Num(≤50)个 k 值最大的基序模型.实验发现,Num 的设定对执行时间影响不大,Gemoda 可以指定不同相似矩阵(如 PAM250,Blosumn62 等).根据作者的建议,对于模拟测试数据,我们使用 dna_idmat 的相似矩阵,同时,为了提高 Gemoda 的执行效率,同时根据原文作者的意见,测试中还设定在相似矩阵中每次采样 1 000 个点,CCMF 和 CBMF 需要指定 l,d 和 k 的值,其中 k 设为 16.

测试中除了 Gemoda 的(14,4)和(17,5)测试外,其他测试都使用了 20 组模拟数据进行实验.表 1 给出了各种算法 20 次测试的成功率,即 20 组模拟数据的测试结果中正确预测的实验次数的次数.其中,正确预测是指算法预测的基序用例与植入的基序模型相容,即基序模型与基序用例之间发生替换的碱基个数小于或等于 d[5].表 1 中“−”表示由于单次测试所需时间过长,仅进行了一组模拟数据的测试.表 2 是各种算法的平均运行时间的比较.有下划线的数据为单组测试的运行时间.

Table 1 Successful rate comparison of CCMF and other algorithms

表 **1** CCMF 和其他算法的成功率比较

(*l*,*d*)	PatternBranching	PROJECTION	Voting	Gemoda	CBMF	CCMF
(10,2)	0/20	0/20	15/20	20/20	20/20	20/20
(11,2)	0/20	12/20	19/20	20/20	20/20	20/20
(12,3)	0/20	0/20	12/20	20/20	20/20	20/20
(13,3)	0/20	10/20	18/20	20/20	20/20	20/20
(14,4)	0/20	1/20	10/20	—	20/20	20/20
(15,4)	0/20	0/20	19/20	20/20	20/20	20/20
(17,5)	0/20	3/20	1/20	—	20/20	20/20

Table 2 Average running times comparison of CCMF and other algorithms (s)

表 **2** CCMF 和其他算法的平均运行时间比较 (单位:秒)

(*l*,*d*)	PatternBranching	PROJECTION	Voting	Gemoda	CBMF	CCMF
(10,2)	11.77	54.60	0.54	275.33	463.28	158.74
(11,2)	16.47	11.96	1.15	28.89	276.28	47.18
(12,3)	13.93	2.48	10.72	19396.64	8439.56	3292.98
(13,3)	19.40	45.61	19.47	336.33	521.44	149.20
(14,4)	14.43	97.06	148.96	>1000 hours	156889.59	28179.22
(15,4)	20.03	124.60	283.62	10282.44	5943.15	2036.03
(17,5)	19.17	84.57	69.29	946440.54	71105.14	9673.71

PatternBranching 是假设样本序列中每条序列中均包含一个基序用例.我们从测试结果中发现,每个 PatternBranching 的结果中均给出了 20 个基序用例,且每条序列恰好包含一个用例,每次实验的结果中都包含了与基序模型 M 不相容的伪用例.虽然 PatternBranching 的执行效率最高,但它只能适用于(*l*,*d*)基序问题的求解,而无法解决(*l*,*d*)-(20-16)问题.

PROJRCTION 假设样本序列中有 k 条序列中包含基序用例,且每条序列中仅包含一个基序用例.测试结果中,PROJRCTION 找出基序模型 M 是在样本序列中 16 条序列中每条序列出现一个基序用例的基序模型.测试结果表明,对所有(*l*,*d*)-(20-16)问题模型,这种算法预测的正确性都不高.

在 Voting 的结果中,我们仅保留基序用例个数大于 16 的基序模型来评价算法性能.从测试结果可知,对大多数 Voting 的执行效率较高,对多数(*l*,*d*)-(20-16)问题执行效果较好;对于 l,d 较大的情况(如 l=17,d=5 的问题),Voting 采用了类似随机映射的启发式技术来提高执行效率,这对算法的正确性有较大影响.可见,上述 3 种算法都无法保证找到所有的基序模型和基序用例.

Gemoda 直接从图 G_{20} 中搜索最大团,测试证明,它能找到所有(*l*,*d*)-(20-16)问题的基序用例,测试的正确性很高,但测试结果中没有给出这些基序用例对应的基序模型.而且 Gemoda 的测试一般都很耗时,特别是求解(14,4)-(20-16)问题的运行时间太长,仅一组测试数据的运行时间就超过 1 000 小时还没有给出结果.

与 Gemoda 一样,CCMF 和 CBMF 也能找到(*l*,*d*)-(20-16)问题的所有基序用例.这更进一步证明,CCMF 和 CBMF 是完全搜索的策略.相对于 Gemoda,CCMF 算法和 CBMF 算法还给出了对应的基序模型,且运行效率较高.

CBMF 和 CCMF 也是采用图论中方法在 20 分图 G_{20} 中搜索团来求解(*l*,*d*)-(20-16)问题,因此,图 G_{20} 中边的稠密程度是影响执行速度最关键的因素.图 G_{20} 中,边越密,求解的时间越长,求解难度也就越大.CBMF 利用 4 845 个组合,重复执行 4 845 次从图 G_{20} 中选择 16 个结点部集,在这些部集组成的不同 16 分子图中,用分治算法和分支定界法搜索 16 团;CCMF 利用 $Colorings_{16}^{20}$ 中的 403 个着色,重复执行 403 次着色,并将每次着色后同色的结点部集合并为同一个结点部集,由此构成图 G'_{16},再利用分治算法和分支定界法搜索 16 团.可见,CBMF 重复执行分治算法和分支定界法的次数是 CCMF 的 4845/403≈12 倍.表 2 的测试结果表明,CCMF 的执行效率并没有比 CBMF 提高 12 倍.这是由于 CBMF 每次只对图 G_{20} 中的子图重复执行,子图中只包含所选的 16 个结点部集中的结点,而 CCMF 每次都是将图 G_{20} 转化为 G'_{16},图 G'_{16} 中的结点数并没有减少.尽管如此,由表 2 可知,对所有的测试数据,CCMF 的执行效率与 CBMF 相比都有明显的提高,特别是对求解难度大的(14,4)和(17,5)问题,其执行效率提高得更为明显,这充分证明了 CCMF 的高效性.

3.2 生物数据

为了验证 CCMF 算法的实用性,本文选择了 4 组已知基序模型和基序用例的真实生物序列对 CCMF 算法进行测试.这 4 组数据包括 E.coli 的 CRP 绑定位点的序列集合[22],这是最早用于基序发现问题算法测试的数据之一,其中每条序列长度为 105 个碱基.另外 3 组数据来自酵母的转录因子的数据库 SCPD[23].SCPD 库中给出了每个转录因子的模型,绑定位点的位置以及包含这些转录因子的序列.本文从 SCPD 库选取了 3 组有代表性的包含不同转录因子的序列样本.这 3 组数据分别选取了已知绑定位点的 20 条基因上游区域附近的长度为 600 的样本序列片段,在所选的 20 条序列中至少有 16 条序列中包含已知的信号位点.测试中使用的参数是根据已知的基序模型和用例选定的.

表 3 给出了测试中使用的参数和测试的结果,粗体部分表示与参考基序模型一致的部分.

Table 3 Experimental results of CCMF on real biological data

表 **3** CCMF 算法的生物数据测试结果

Name	(*l*,*d*)	Reference motif pattern	CCMF
CRP	(20,8)	*TGTGANNNNGNTCAC*	*TTT**TGTGAAATGGGTCAC**AT* *TT**TGTGAACTAGTTCAC**ACT* ***TGTGAAAAAGTTGAC**ATTTT*
*RAP*1,*EBF*1	(9,1)	[*A*/*G*][*A*/*C*]*ACCCA*	*A**AAA**ACCA**A* ***AAACC**AA**AA*
TATA,TBP	(9,1)	*TATA*[*A*/*T*]*A*[*A*/*T*]	*A**TATATAA**A* ***TATATAA**AA*
UASPHR	(8,1)	*CTTCCT*	*TT**CTTC**TT*

Note: *N* means [*A*/*C*/*T*/*G*].

由表 3 可知,CCMF 算法在 CRP 样本中发现了 3 个基序模型,实际上,它们是同一基序模型的不同位移,且都包含了参考模型;对其他 3 组生物数据,CCMF 算法都能发现与参考模型基本相符的模型,且所找到的基序用例的位置与 SCPD 中标志的转录因子的绑定位点的位置一致.因此可以认为,CCMF 算法所发现的模型是可信的.CCMF 算法可应用于真实的基序发现问题.

4 结 论

为了解决(*l*,*d*)-(20-16)问题,本文提出了一种新颖的 SDA 完全搜索算法:彩色编码基序搜索算法(CCMF 算法).CCMF 算法突破传统 SDA 算法只适用于求解每条序列均包含基序用例的(*l*,*d*)基序问题的限制,可有效地求解(*l*,*d*)-(20-16)问题.CCMF 算法采用了分治和分支定界算法等相关技术,降低每次基序寻找的运行时间,并主要利用彩色编码技术有效降低了运行次数,极大地提高了整个算法运行速度.对模拟数据和生物数据的测试表明,CCMF 能够快速且完全地找出数据中包含的所有基序模型和基序用例,具有优于其他算法的综合评价,能够应用于实际的基序发现问题.通过修改着色方案,CCMF 算法还可以用于求解一般的(*l*,*d*)-(*K*-*k*)问题.

致谢 在此,感谢 Pro. Buhler J.,Tompa M.,Styczynski M.P.,Chin F.Y.L.和 Leung H.C.M.,他们慷慨地提供了 PatternBranching,PROJECTION,Gemoda 和 Voting 算法的源代码,并对实验过程中算法的运行参数设置等问题给予许多宝贵意见.

References:

[1] Helden JV, Andre B, Collado-Vides J. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. Journal of Molecular Biology, 1998,281:827−842.

[2] Liu X, Brutlag DL, Liu J. Bioprospector: Discovering conserved DNA motifs in upstream regulatory regions of co-expressed gene. In: Altman RB, eds. Proc. of the 6th Pacific Symposium on Biocomputing. Word Scientific Publish Company, 2001. 127−138.

[3] Blanchette M, Schwikowski B, Tompa M. Algorithms for phylogenetic footprinting. Journal of Computational Biology, 2002,9(2): 211−223.

[4] Liu XS, Brutlag DL, Liu JS. An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. Nature Biotechnology, 2002,20(8):835−839.

[5] Pevzner P, Sze S. Combinatorial approaches to finding subtle signals in DNA sequences. In: Bourne PE, Gribskov M, Altman RB, *et al*., eds. Proc. of the 8th Int'l Conf. on Intelligent Systems for Molecular Biology (ISMB 2000). San Diego: AAAI Press, 2000. 269−278.

[6] Styczynski MP, Jensen KL, Rigoutsos I, Stephanopoulos GN. An extension and novel solution to the (*l*,*d*)-motif challenge problem. Genome Informatics, 2004,15:63−71.

[7] Chin FYL, Leung HCM. An efficient algorithm for the extended (*l*,*d*)-motif problem with unknown number of binding sites. In: Bourbakis NG, ed. Proc. of IEEE 5th Symp. on Bioinformatics and Bioengineering (BIBE 2005). Minneapolis: IEEE Computer Society, 2005. 11−18.

[8] Brazma A, Jonassen I, Eidhammer I, Gilbert D. Approaches to the automatic discovery of patterns in biosequences. Journal of Computational Biology, 1998,5(2):279−305.

[9] Buhler J, Tompa M. Finding motifs using random projections. Journal of Computational Biology, 2002,9(2):225−242.

[10] Sinha S, Tompa M. Discovery of novel transcription factor binding sites by statistical overrepresentation. Nucleic Acids Research, 2002,30(24):5549−5560.

[11] Eskin E, Pevzner PA. Finding composite regulatory patterns in DNA sequences. Bioinformatics, 2002,18:354−363.

[12] Price A, Ramabhadran S, Pevzner PA. Finding subtle motifs by branching from sample strings. Bioinformatics, 2003,SII:149−155.

[13] Sze SH, Lu S, Chen J. Integrating sample-driven and pattern-driven approaches in motif finding. In: LNCS/LN Bioinformatics (WABI 2004), 2004. 438−449.

[14] Gross JL, Yellen JL. Handbook of Graph Theory and Applications. CRC Press, 2003. 12−43.

[15] From MathWorld−A wolfram Web resource. 2006. http://mathworld.wolfram.com/Combination.html

[16] Alon N, Yuster R, Zwick U. Color-Coding. Journal of the ACM, 1995,42(4):844−856.

[17] Chen J, Lu S, Sze SH, Zhang F. Improved algorithms for path, matching, and packing problems. In: Gabow HN, ed. Proc. of the 18th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA'2007). 2007. 298−307.

[18] Scott J, Ideker T, Karp R, Sharan R. Efficient algorithms for detecting signaling pathways in protein interaction networks. Research in Computational Molecular Biology (RECOME 2005). 2005. 1−13.

[19] Chen J, Kanj I, Meng J, Xia G, Zhang F. On the effective enumerability of NP problems. In: Bodlaender HL, Langston MA, eds. Proc. of the 2nd Int'l Workshop on Parameterized and Exact Computation (IWPEC 2006). Zürich: Springer-Verlag, 2006. 215−226.

[20] Lanctot JK, Li M, Ma B, Wang SJ, Zhang LX. Distinguishing string selection problems. Information and Computation, 2003,185(1): 41−55.

[21] Gramm J, Niedermeier R, Rossmanith P. Exact solutions for closest string and related problems. In: LNCS 2223 (ISAAC 2001), 2001. 441−453.

[22] Stormo GD, Hartzell III GW. Identifying protein-binding sites from unaligned DNA fragments. In: JSTOR, ed. Proc. of the National Academy of Sciences of the United States of America. National Academy of Sciences, 1989,86(4):1183−1187.

[23] Zhu J, Zhang M. SCPD: A promoter database of the yeast Saccha-romyces cerevisiae. Bioinformatics, 1999,15:563−577. http://cgsigma.cshl.org/jian/

王建新(1969－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为生物信息学,网络优化理论.

陈建二(1954－),男,博士,教授,博士生导师,主要研究领域为生物信息学,计算机理论,计算复杂性及优化,计算机网络优化算法,计算机图形理论与算法.

黄元南(1979－),女,博士生,主要研究领域为生物信息学,计算机理论.