

基于分布式信源编码的微生物基因组序列压缩算法

陈 旻¹, 王开云^{2*}

(1. 云南警官学院 信息网络安全学院, 云南 昆明 650223; 2. 昆明学院 学报编辑部, 云南 昆明 650214)

摘要:提出一种基于分布式信源编码的微生物基因组序列压缩算法,用于改进微生物基因组序列压缩效率不高的现状. 首先将微生物基因组序列映射为两条二进制序列并映射为两幅二值图像以便使用更多的信源相关性. 然后构建分布式信源编码来同时传输两个二值图像. 同时,为保证边信息传输的高效,优化 Context 加权方法被用于边信息压缩. 实验结果表明,与现有微生物基因组序列压缩算法相比较,该算法能获得更高的压缩效率,并且保持较合理的运算复杂度.

关键词:分布式信源编码;微生物基因组序列压缩;边信息;Context 加权;希尔伯特空间填充

中图分类号: TN919.81 **文献标识码:** A **文章编号:** 1674-5639(2015)06-0106-06

DOI: 10.14091/j.cnki.kmxyxb.2015.06.027

A Novel Microbial Genome Sequence Compression Algorithm Based on Distributed Source Coding

CHEN Min¹, WANG Kai-yun^{2*}

(1. Department of the Information Security, Yunnan Police Officer Academy, Yunnan Kunming 650223, China;

2. Editorial Department of Journal, Kunming University, Yunnan Kunming 650214, China)

Abstract: A novel microbial genome sequence compression algorithm based on the distributed source coding technology is proposed to enhance the compression efficiency. First, the microbial genome sequence is mapped into two binary sequence and into two binary images to use more source correlation and then to construct the distributed source coding to transmit two binary image. At the same time, in order to guarantee the high efficiency of side information, the optimized context weighting is used in compressing the side information. The experiments results indicate that compared with the present microbial genome sequence compression algorithm, the new algorithm can obtain better compression efficiency and maintain reasonable complexity in operation.

Key words: distributed source coding; microbial genome sequence compression; side information; Context weighting; Hilbert spacing filling

第二代基因组测序技术使得获得的基因组序列数据呈指数级增长,也对存储这些数据提出了更高的要求^[1]. 在近 20 a 的研究中,前人基本上给出了三大类针对基因组序列的压缩算法. 一类是基于字典的压缩,另一类是使用无损熵编码进行压缩的方法,第三类是最近几年研究的热点,基于 Reference 的基因组压缩序列.

最早关于基因组压缩的研究可以追溯到 Grumbach 的工作. 文献[2]中,一种基于 LZ77 的字典压缩方法 BioCompress 用于对基因组序列进行压缩. 该算法首先检测重复序列并与字典中存在的序列进行比

对,若序列存在于字典中,则使用斐波那契编码对字典的条目索引进行编码,若没有匹配到,则每个碱基编码为 2 bit. BioCompress 及其改进算法^[3-4]给出了使用字典压缩算法对基因组序列进行编码的基本框架. 后续算法也只是针对没有匹配到的重复序列和非重复序列的编码方法进行改进. 例如文献[3]中,对于非重复序列采用二阶马尔科夫模型来对其进行描述,然后使用熵编码对非重复序列进行压缩. 而在文献[4]中,基于 Context 建模熵编码技术被用于压缩非重复序列和那些没有在字典中匹配到的序列. 这是 Context 建模熵编码第 1 次被应用于基因组序列压缩,但

收稿日期:2015-11-01

基金项目:国家自然科学基金资助项目(61062005);云南省自然科学基金青年基金资助项目(2013FD042).

作者简介:陈旻(1982—),男,云南昆明人,副教授,博士,IEEE 高级会员,主要从事信息传输理论和物联网通信技术研究.

* 通讯作者:王开云(1962—),男,云南昆明人,高级实验师,主要从事信息传输技术研究, E-mail: kmwky@sina.cn.

在该算法中,Context 建模技术只是字典压缩的一种有效补充.事实上,采用基于字典的压缩方式,从整体上来说,并不能够保证基因组序列获得较高的压缩效率.因为字典压缩依赖于字典,而随着字典的规模越来越大,对字典条目索引进行编码所需的码长也将逐渐增加,则会导致压缩效率降低.另一方面,为提高压缩效率,可以对字典进行适当初始化.但如此一来,就不能忽略对字典本身进行编码的代价.

此外,目前对于基因组序列压缩,一大类称为Reference 的压缩方法以较高压缩效率吸引着研究人员.该类算法的主要依据为物种基因组序列中存在重复序列片段.在压缩过程中,首先给定一条参考序列,待编码序列与参考序列进行比对,只传输或存储两条序列之间的差异.而其压缩效率提升的关键依赖于重复序列比对的命中率.命中率高,差异小,则压缩效率较高.例如文献[5]中,人类基因组序列的压缩效率可达到近80倍.然而,不论何种压缩算法,对哺乳动物基因组序列进行压缩时压缩效率通常较高.这是因为在这些物种的基因组序列中,重复序列存在的比例较高.以人类基因组为例,重复序列比例可达到近90%以上^[6].这也就保证了字典命中率或者比对命中率较高.然而,对微生物基因组序列而言,重复序列的占比要低得多,目前检测到的最高重复序列占比只有22%左右.这样的重复序列比例不足以发挥字典类或Reference 类压缩算法的优势.正因为如此,在前人关于微生物(包括细菌)基因组序列压缩研究中^[6-10],多采用Context 建模熵编码技术.

另一方面,因为物联网技术的发展,越来越多的嵌入式设备被应用于各个领域.在遗传生物学领域,手持式测序设备逐渐普及.然而,对于嵌入式设备来说,其最大的挑战在于能量.过高的运算量,过多的数据交换,都会消耗本来就珍贵的电能.于是,一种弱编码、强解码的编码方法成为新的研究热点.这种编码称为分布式信源编码^[11-12].此类编码保证编码端以尽可能简单的方式对信源进行编码传输,然后利用信道编码的方式传输信息,而在解码端部署运算量较高的解码算法来实现信源的重构.分布式信源编码的提出其实很早,但真正获得突破和应用是随着文献[11]的发表.同样,其首次被应用于基因组序列压缩是在文献[12]中报道.然而,该方法其实是Reference 类算法的补充,并不能真正算作是单纯的分布式信源编码基因组序列压缩算法.在该算法中,首先判断局部子

序列是否匹配到Reference,如果匹配则传输其Hash值,否则才使用分布式信源编码进行压缩.事实上,该文献主要针对人类基因组序列进行研究,且其关注的是编码时间的提高.就压缩效率而言,该算法远不如Reference 类算法,甚至不如一些熵编码压缩算法.对微生物基因组,Reference 类并不能发挥效果.

我们受其启发,结合之前采用的二维映射方法^[13],提出一种纯粹基于分布式信源编码的微生物基因组序列压缩算法.同时,为了尽可能提高效率,优化Context 加权^[14-15]也同样应用于新的微生物基因组序列压缩算法.在下文中,我们将详细论述该算法.

1 方法简述

上文对分布式信源编码研究现状及其在基因组序列压缩中的应用研究进行了讨论.在此,我们下面将对其进行详细分析.

文献[12]的工作开创了分布式信源编码应用于基因组序列压缩的新局面.然而,其毕竟只是对现有的Reference 比对类方法进行了改进,其应用域仅仅局限于那些没有被匹配到的独立碱基压缩.然而,这样的思想其实弱化了编码碱基间的相关性,并不能充分发挥分布式信源编码的优势.但文献[12]提出了一项建设性意见,即将Reference 与待编码序列均当作需要传递的信源序列,然后使用分布式信源编码进行压缩.该思想是一种极大的创新,而且其实验结果表明,对一大类重复序列较高的基因组序列来说,压缩效果较理想.然而,对微生物基因组序列,正如上文所分析的,相邻碱基间的相关性以及重复序列较少,则不论怎样改进编解码器,都很难获得较理想的压缩结果.不妨设 X 和 Y 分别代表待编码微生物基因组序列和Reference.分布式信源编码的原理描述为:传递信源的联合熵为 $H(X, Y)$.根据信息论可知:

$$H(X, Y) = H(X) + H(Y|X). \quad (1)$$

如果信源序列 X 已传递,则对信源序列的编码则无需按照熵值 $H(Y)$ 进行传递,只需传递 $H(Y|X)$ 即可.在此情况下,只要获得极小的条件熵 $H(Y|X)$ 估计,即可对 Y 实现压缩.此时, $H(X)$ 称为边信息.然而遗憾的是,分布式信源编码为了减小编码端的运算量,两个信源不允许通信.也就是说,信源 Y 在编码过程中并不知道信源 X 究竟能提供多少信息量来降低熵值,这是它不同于条件熵编码之处.换言之,只有在编码前期确定两个信源的相关性,才有可能获得较好压缩效果.其

实,这种说法也不准确.折中的说法是:如果参与编码的两个信源强相关,则就算在编码过程中不通信,在解码端也同样可以获得理想的压缩效果^[11].

因此,对两条微生物基因组序列,其本身的相关性就不强,直接采用分布式信源编码并不能获得理想的结果.一种直观的想法是只使用一条序列,然后分别送两个编码端.但如果单纯的直接传送,相当于将一条相同的序列重复传递两次,这不仅不会获得压缩,反而增加编码 bit.为了解决此矛盾,同时又能保证充分利用碱基间的相关性.我们沿着文献[13]的研究思路,将一条基因组序列映射到二维,然后再进行压缩.为了获得两条编码序列,我们按照以下原则进行映射.

首先,按照以下映射规则构建两条二进制序列:

$$\begin{cases} A \rightarrow A, \\ T \rightarrow C \end{cases} \quad (2)$$

如此一来,一条基因组序列被映射为只有 A 和 G 的序列.同时,为了保证解码,需要另一条二元素序列来表示一个碱基是被映射为自己还是另一个相对碱基,即:

$$a = \begin{cases} 0, & \text{if } A \rightarrow A, \\ 1, & \text{if } A \rightarrow T. \end{cases} \quad (3)$$

通过式(2)和(3),一条微生物基因组序列被映射为两条二进制序列,而且两条序列间还存在相关性,因而适合采用分布式信源编码进行压缩.在下一节中,我们将详细讨论算法的细节.

2 算法细节

2.1 二维映射

在文献[13]中,我们已经详细论述了使用希尔伯特空间填充曲线将基因组序列映射到二维图像,然后进行压缩的方法.在本文中,我们同样将两条二进制序列映射为两幅二值图像.其实,分布式信源编码并没有对信源的维度作出限制.只不过映射到二维可以使用更多的信源间相关性.同样的,按照(4)式,可以很方便地得到映射后的二值矩阵:

$$\begin{cases} \begin{bmatrix} H_{2^k} & 4^k E_{2^k} + H_{2^k} \\ 4^{(k+1)} E_{2^k} - H_{2^k} & (3 \times 4^k + 1) E_{2^k} - (H_{2^k})^T \end{bmatrix} \\ k \text{ 为偶数,} \\ \begin{bmatrix} H_{2^k} & (4^{k+1} + 1) E_{2^k} - H_{2^k} \\ 4^k E_{2^k} + H_{2^k}^T & (3 \times 4^{k+1}) E_{2^k} - (H_{2^k})^T \end{bmatrix} \\ k \text{ 为基数.} \end{cases} \quad (4)$$

值得注意的是,在文献[13]中,我们讨论了映射方法引入的无效编码区.无效编码区是采用二维

映射时必须考虑的问题.然而,在使用分布式信源编码进行压缩时,无效编码区并不会对压缩结果产生影响.原因是,在分布式信源编码中,作为边信息的一条序列按照正常方式(可以使用任何现有的编码方法)进行压缩.当边信息知道时,其二映射图像的拓扑结构也就随之知道,即什么位置是无效编码区即可得知.此时,在编码或传输另一条序列时,也就不需要传输无效编码区的符号.因此,无效编码区符号造成的编码代价也就不需要考虑.

借助二维映射方法,我们不仅能够将一条基因组序列映射为两条序列,还能够利用其相关性构建分布式信源编码压缩方法.

2.2 分布式信源编码方法

在获得两条二进制序列后,我们构建分布式信源编码器以实现压缩.分布式信源编码要求同时传递的两条序列之间不能通信,以降低编码端的运算复杂度.在本文中,我们同样采用带边信息的分布式信源编码器作为基因组序列的压缩器.对于纠错码,我们使用文献[12]提出的 LDPC 码来实现.

编码端:分布式信源编码要求编码端以尽可能简单的方式进行编码.设映射得到的两条序列分别为 X 和 Y .按照(1)式的含义,序列 X 称为边信息.需要考虑的是如何传输 $H(Y|X)$.根据文献[11~12]的研究,设有一个分组码 (n, k) ,其码字 C_n 可以由其信息位 C_k 与生成矩阵 G 运算得到,即:

$$C_N = \vec{G} \cdot C_k. \quad (5)$$

解码端:在解码时,当收到码字 C_n 时,可以运算得到其信息 C_k .此时,码率为 k/n .这就意味着,生成矩阵必须收发双方都知道.一旦信源包含的可能的符号种类较多,则生成矩阵 G 的规模将变得巨大,甚至超出现在的存储运算能力.根据前人研究,事实上并不需要直接传输生成矩阵,可以只传递其伴随式 S 即可.当收到码字 C_n 时,利用伴随式,并结合最小汉明距离原则,可以很方便地解码出信息位 C_k .这就意味着,只需要传递伴随式即可实现解码.而对 (n, k) 分组码而言,伴随式耗费的比特数为 k .于是,使用 k 比特对 n 比特信息量进行编码,以实现压缩.

然而,对映射到二进制的两个序列而言,其有效信息为 1 bit,如果再构建纠错码,则无形中增加了描述代价,且并不能获得相应的压缩效率.一种可行的方法是结合基因组序列结构特征进行考虑.根据遗传生物学知识可以知道,3 个碱基构成蛋白质.3 位

ATGC 4 种符号共有 64 种组合,两条二进制序列中,3 位的组合共有 8 种,两条合起来共 64 种.然而,实际应用中并没有 64 种蛋白质.设可能出现的蛋白质数量为 m ,则需要的碱基个数为 $\log_4 m < \log_4 64$.于是,如果按三碱基为一组,则可将其看作是一个 $(3, \log_4 m)$ 的分组码.同样的,将其映射后,一条二进制序列可以看做是一个 $(3, \log_2 \sqrt{m})$ 的分组码.理论上,其可实现的压缩比(或码率)为:

$$\eta = \frac{\log_2 \sqrt{m}}{3}, m < 64. \quad (6)$$

对于这样的分组码来说,不论是使用 CRC 还是 LDPC,均能够较好的获得其伴随式,从而保证分布式信源编码的实现.由于篇幅限制,本文并不给出编解码端的设计示意图,但采用的是编解码方法,可参考文献[12]工作.

2.3 边信息的压缩

在以上分析的基础上,为了进一步提高压缩效率,可以考虑在传输边信息时,也对其进行压缩.事实上,分布式信源编码并没有对边信息采用何种编码方法做出限制.由于在本文工作中,边信息是一个二进制信源,对其压缩,可以考虑使用基于 Context 建模熵编码技术进行.为了充分利用信源相关性,也同样可以考虑使用文献[14]中提出的优化 Context 加权方法.在本文中,使用的 Context 模板如下表 1 所示.

表 1 本文使用的 Context 模板

模型	阶数	一维时条件位置
Context model 1	5	XXXXX?
Context model 2	8	XXXXXXXXX?
Context model 3	3	XOOXX?

其中,符号 X 代表其在一维状态下选为条件位的符号,? 代表当前编码符号,0 代表该位置的符号不用作条件.

由此可知,构建 3 个 Context 模型,然后使用优化 Context 加权的方式获得编码分布.权值的优化仍然采用最小描述长度准则,借助最小二乘法来获得.其基本方法由式子(7)~(8)给出.

$$L = w_1 L_1 + w_2 L_2 + Q, \quad (7)$$

其中, L_1 和 L_2 分别为参与加权的条件概率分布对应的描述长度, w_i 为权值, Q 为加权代价.描述长度由对应分布的计数向量获得,其计算由(8)式给出:

$$L_1 = V_1 \log V_1 - n_0 \log n_0 - n_1 \log n_1 - n_2 \log n_2 - n_3 \log n_3 - \frac{1}{2} \log \frac{V_1}{n_0 n_1 n_2 n_3} + \sigma, \quad (8)$$

其中, V_1, n_i 表示计数向量中包含的计数总数和取值为 i 的符号的数量.在获得(7)式的基础上,权值 w_i 由求解方程(9)获得:

$$\begin{cases} \frac{\partial f(W)}{\partial w_i} = 0, i = 1, 2, \dots, N, \\ \sum_{i=1}^N w_i = 1. \end{cases} \quad (9)$$

这样一来,边信息即可实现压缩传输,从而提高整个算法的压缩效率.

3 实际应用中的考虑

在给出算法的基本框架后,在实际应用中,仍然需要考虑一些细节.主要包括:

1) 边信息序列的选取.在分布式信源编码的理论框架中,并没有明确指出哪一条序列应当作为边信息.换言之,对我们算法中得到的两条二进制信源来说, X 和 Y 均可作为边信息序列.为了提高压缩比,且我们的算法并不真正应用于嵌入式设备,所以,并没有必要真正降低编码端的运算复杂度.因此,可以在编码端事先检测两条序列的统计熵,即描述复杂度.然后使用 1 个 bit 确定究竟选取哪条序列作为边信息传输给解码端.值得注意的是,这 1 个 bit 的代价是值得的.因为只要熵值降低一点,对大量碱基来说,就意味着整个码长会降低一个可观的码长数.

2) 分布式信源编码在传输数据过程中,要求两条序列在不通信的情况下实时并传.然而,我们在算法中并不需要如此做.而是完全可以先把边信息序列传输,然后再传递伴随式.这同样是因为我们在本文工作中,并不专注于设计基于嵌入式或物联网传输的压缩方法,也就无需考虑设备消耗的问题.而且,两条二进制信源的互相关满足对称(二进制信源序列特点).所以,两条序列是否需要同时传输,并不影响最终压缩结果.但这并不意味着我们的算法不能用于嵌入式设备.只不过是为了保证边信息的高压缩比,才采用分顺序传输的方式进行.

综上所述,本文算法可以概括为以下 7 个步骤:

步骤 1 首先使用碱基映射方法和希尔伯特空间填充矩阵将一条微生物基因组序列映射为两条二进制序列;

步骤 2 分别计算两条二进制序列的熵值,选取熵值低的一条作为边信息,并使用 1 bit 告诉解码端;

步骤 3 使用优化 Context 加权熵编码对边信息序列进行压缩传输;

步骤 4 按照每 3 个碱基为 1 个码字的方式计算剩余序列的伴随式,并传输伴随式;

步骤 5 解码端根据伴随式重构二进制序列;

步骤 6 解码端根据收到的边信息和重构的二进制序列,并结合逆希尔伯特空间填充矩阵,恢复出两条二进制序列;

步骤 7 根据两条序列,最终恢复出原始的基因组序列。

4 实验

为验证本文算法,我们将基于分布式信源编码的基因组序列压缩算法应用于微生物基因组。我们用于实验的微生物基因组序列来源于 NCBI^[16]。这

些序列包括:

1) HEHCMVCG, 人类巨细胞病毒株 AD169 完整基因 DNA 序列;

2) CHMPXX, 地钱叶绿素基因 DNA 序列;

3) CHNTXX, 烟草叶绿素基因 DNA 序列;

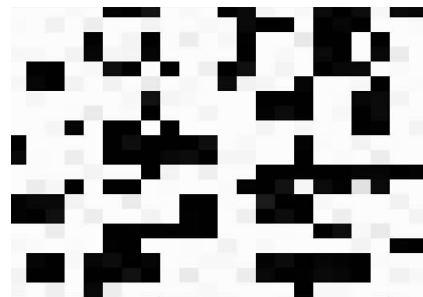
4) VACCG, 牛痘病毒完整基因 DNA 序列。

在我们的实验中,分布式信源编码器采用文献[12]建议的 LDPC 码生成器。映射方法采用碱基二值化和希尔伯特空间填充矩阵法实现。对于边信息序列,则采用我们前期研究的优化 Context 加权。首先完成映射。

按照二值化和希尔伯特空间填充矩阵方法,可以得到两条二值序列及其对应的二维图像。如图 1 所示。



(a) 碱基二值化图



(b) 映射二值图

图1 碱基序列VACCG映射后的两幅二值图像

注意,图 1 实际上为二值图像,为了看得清楚,我们将其做了灰度修改,以保证其清晰度。

此时,我们需要根据两幅二值图像的灰度直方图来计算其对应统计分布的熵值,以此判断那副二值图(或二进制序列)作为边信息优先传输。

确定后,我们使用熵编码传输边信息。对我们使用的优化加权,并排除那些导致加权代价过高的碱基,从而提高压缩效率。对参与实验的 4 条微生物碱基,其选中的边信息压缩结果如表 2 所示。

其中,边信息编号表示究竟是哪幅二值图像被选取作为边信息。编号 1 表示由公式(2)映射得到的二值序列,编号 2 表示由公式(3)映射得到的二值序列。

表 2 4 条序列边信息编码效率

基因组序列	边信息编号	压缩值/(bit · 符号 ⁻¹)
HEHCMVCG	1	0.91 bpp
CHMPXX	2	0.75 bpp
CHNTXX	2	0.74 bpp
VACCG	1	0.79 bpp

之后,使用分布式信源编码器对 4 条碱基进行编码,得到的结果如下表 3 所示。为了对比压缩效果,我们将本实验的压缩结果与现有的针对微生物基因组序列的压缩算法结果进行比对,这些方法的结果来源于文献[15]。参与比对的算法有:BioC^[2], GenC^[3], DNAP^[4], GeMNL^[5], XM^[15]。评价压缩效率的指标为编码的单位 bit 数,即 bit/碱基。

表 3 整体压缩结果对比

序列	BioC	GenC	DNAC	DNAP	GeMNL	XM	本文
HEHCMVCG	1.848 0	1.847 0	1.849 2	1.834 6	1.842 0	1.842 6	1.623 3
CHMPXX	1.684 8	1.673 0	1.671 6	1.660 2	1.661 7	1.657 7	1.433 8
CHNTXX	1.617 2	1.614 6	1.612 7	1.610 3	1.610 1	1.606 8	1.418 9
VACCG	1.761 4	1.761 4	1.758 0	1.758 3	1.764 4	1.764 9	1.424 1

从表3中不难看出,本文提出的算法要优于现有的任何一种基因组序列压缩算法.当然,我们并没有与现在的Reference类压缩算法进行比较,这是因为参与我们实验的基因组序列均为微生物类基因组.对于此类基因组序列,Reference类算法并不能获得比参与对比的其他压缩算法更好的效果.同时,我们给出本文算法与其他算法运行的时间比较.为了公平起见,仅对比能够在我们计算机上运行的算法,包括:BioC, XM. 其余算法因为没有找到相应的可执行程序,只能找到发表于其他论文上的运行时间,但由于运行环境不同,运行时间会存在不同,故此不参与比较.本文运行平台为:Inter i5 处理器 2.6 GHz, RAM 4 G. 运行时间如表4所示.

表4 算法运行时间比较

算法	编码器时间/s	解码器时间/s	总时间/s
BioC	—	—	4.7
XM	—	—	7.3
本文	1.3	4.4	5.7

很明显,就算使用了分布式信源编码,但本文算法仍然优于其他算法.就算其无法达到字典类算法的时间复杂度,但我们算法的运算需求仍然处于可以接受的范围.当然,我们并没有将边信息选取的消耗时间计算进去.这是因为熵值只是一种可选择性的尝试.在后续研究中我们将进一步研究更便捷的判别方法.

综上所述,本文提出的基于分布式信源编码的微生物基因组序列压缩算法是可行的.

5 结论

本文基于分布式信源编码,构建了一种新颖的基于分布式信源编码的微生物基因组序列压缩方法.在前人研究基础上,我们通过二维映射的方法将一条基因组序列映射为两条二进制序列以保证分布式信源编码的实现.实验结果表明,本文提出的算法确实能够较好地提高微生物基因组序列的压缩效率,且运算时间要大大低于现有的熵编码类微生物基因组序列压缩方法,达到最初设计目标.

[参考文献]

[1] GRUMBACH S, TAHI F. Compression of DNA sequences [C]//Proc Data Compression Conference. Snowbird; IEEE Computer Society, 1993:340–350.
[2] GRUMBACH S, TAHI F. A new challenge for compression algorithms: Genetic sequences [J]. Information Processing &

Management, 1994, 30(6):866–875.

- [3] RIVALS E, DELAHAYE J P, DAUCHET M, et al. A guaranteed compression scheme for repetitive DNA sequences [C]//Proc Data Compression Conference. Snowbird; IEEE Computer Society, 1996:453–471.
[4] CHEN X, KWONG S, LI M. A compression algorithm for DNA sequences and its applications in genome comparison [C]//Proceedings of the Fourth Annual International Conference on Computational Molecular Biology. New York: NY, 2000:107–116.
[5] CHEN X, LI M, MA B, et al. DNA compress: Fast and effective DNA sequence compression [J]. Bioinformatics, 2002, 18(2):1696–1698.
[6] BEHZADI B, FESSANT F L. DNA compression challenge revisited: A dynamic programming approach [J]. Combinatorial Pattern Matching, 2005, 353:190–200.
[7] MATSUMOTO T, SADAKANE K, IMAI H. Biological sequence compression algorithms [J]. Genome Informatics, 2000, 11:43–52.
[8] TABUS I, KORODI G, RISSANEN J. DNA sequence compression using the normalized maximum likelihood model for discrete regression [C]//Proc Data Compression Conference. Snowbird; IEEE Computer Society, 2003:253–263.
[9] KORODI G, TABUS I. An efficient normalized maximum likelihood algorithm for DNA sequence compression [J]. ACM Trans Inf Syst, 2005, 23(1):3–34.
[10] SOLIMAN T H A. A lossless compression algorithm for DNA sequence [J]. J Bioinformatics Research and Application, 2009, 5(6):593–602.
[11] PRADHAN S S, RAMCHANDRAN K. Distributed source coding using syndromes (DISCUS): Design and construction [J]. IEEE Trans Inform Theor, 2003, 49(3):626–643.
[12] WANG Shuang, JIANG Xiao-qian, CHI Li-juan, et al. Genome sequence compression with distributed source coding [C]//Proc Data Compression Conference. Snowbird; IEEE Computer Society, 2013:525–572.
[13] 陈旻, 王开云, 吴建国, 等. 基于希尔伯特分形的基因组序列压缩算法 [J]. 昆明学院学报, 2014, 36(6):42–46.
[14] 陈旻, 王开云, 贾学明, 等. 基于加权 Context 建模的 DNA 序列压缩算法 [J]. 昆明学院学报, 2014, 36(3):81–84.
[15] CAOM D, DIX T I, ALLISON L, et al. A simple statistical algorithm for biological sequence compression [C]//Proc Data Compression Conference. Snowbird; IEEE Computer Society, 2007:43–52.
[16] NCBI. Center for biotechnology information [EB/OL]. [2014-03-25]. <http://www.ncbi.nih.gov/genomes/Bacteria/>.