基于改进蚁群算法的 DNA双序列比对

李方洁, 刘希玉, 陈 洁

(山东师范大学管理与经济学院, 山东 济南 250014)

[摘要] 序列比对是生物信息学中基本的信息处理方法之一. DNA序列比对分为多序列比对和双序列比对两种. 本文首先分析了经典蚁群算法在双序列比对中的应用, 然后针对经典蚁群算法收敛速度慢和陷入局部最优值等缺点进行改进, 最后通过实验证明改进的智能蚁群算法在收敛速度和最优值方面都有较大的改进.

[关键词] 蚁群算法, 双序列比对, 信息素

[中图分类号] TP301.6 [文献标识码] A [文章编号] 1001-4616(2010)04-0148-05

DNA Pair-Wise Sequence Alignment Based on an Improved Ant Colony Algorithm

Li Fangjie, Liu Xiyu, Chen Jie

(School of Management and Economy, Shandong Normal University, Jinan 250014)

Abstract Sequence alignment is a basic information processing approach. DNA sequence alignment has multiple sequence alignment and pair-wise sequence alignment. This paper analyzed pair-wise sequence alignment based on standard ant colony algorithm firstly. Secondly, in order to accelerate the convergence and avoid the stagnation, this paper proposed an improved ant colony algorithm. Lastly, it proved that both convergence rate and global optimum have much better results.

Key words ant colony algorithm, pair-wise sequence alignment, pheromone

序列比对(Sequence Alignment)是生物信息学中最基本的信息处理方法之一. 生物信息学是利用计算机、网络等工具, 采用数学和信息科学的理论和方法, 主要研究生物大分子, 比如蛋白质、核酸与 DNA等, 包括它们的序列结构和功能两个方面. 序列比对是指通过对生物序列进行比较, 来发现序列之间的相似性, 并辨别它们之间的产异性, 从而来发现生物序列中的功能、结构和进化等信息.

DNA序列比对分为双序列比对和多序列比对两种. Needleman和 Wunsch于 1970年提出了 NW算法, 基于 1个二维矩阵 $\boldsymbol{F}(n, m)$, 利用最短距离的求解方法, 找出最佳匹配路径[1]. Smith和 Waterman于 1981年提出了 Smith-Waterman算法, 是一种用来查找并比较具有相似性的动态规划算法, 在识别局部相似性时, 具有较高的灵敏度[2]. Pearson和 Lipman于 1985年提出了 Fasta算法, 该算法将查询序列中的所有字符都变成 Hash表, 然后再在数据库搜索时查询这个 Hash表, 以检索出可能的匹配. Altschul于 1990年提出了 Blast算法, 基于片段匹配算法和统计模型来找出目的序列和数据库之间的最佳局部对比结果. 以上提到的 4种算法都是用于解决 DNA双序列问题, 4种算法在算法复杂度、鲁棒性和速度方面上都具有明显的优缺点.

1 DNA双序列比对

定义 对于给定的一组 DNA序列 $S = (S_1, S_2, \cdots, S_k)$, $|S_i|$表示为某一条 DNA序列 $S_i (i = 1, 2$

收稿日期: 2009-06-10

基金项目: 山东省信息产业发展专项基金(2008R00038).

通讯联系人: 刘希玉, 博士, 教授, 研究方向: 计算智能、数据挖掘等. E-mail: xyliu@sdnu.edu.cn

…, k) 的长度. 序列 S_i 中每个序列元素由给定的字母表 Σ 中的一个字符表示, DNA 序列的字母表 $\Sigma = \{A, G, C, T\}$, 由 4 种核苷酸 A、G、C、T 组成.

在进行 DNA 序列比对时, 需要进行替代、删除与插入等操作. 空格 "–" 代表删除或插入操作时所产生的空位. 在进行空位 "–" 插入或删除时, 要求不存在全为空位的一列, 即 $\forall j, \exists i, m_{ij} \neq$ '–'. 其中 DNA 序列比对是一个 $k \times l$ 的矩阵 $\boldsymbol{M} = (m_{ij})_{k \times l}$. x, y 表示序列比对时第一条序列和第二条序列中的元素, 对于 x, $y \in \Sigma \cup \{-\}$, 定义 $\sigma(x, y)$ 为计分函数, 表示为 x, y 比较时的得分, 最简单的计分函数为:

$$\sigma(x, y) = \begin{cases} 2 (x = y \in \Sigma), \\ -1 (x \neq y \in \Sigma), \\ -2 (x = \text{'–'} \text{ or } y = \text{'–'}). \end{cases} \quad (1)$$

在公式 1 中, 两个序列元素 x 与 y 进行比对, 若 2 个元素相同并且同时属于字母表, 则得分为 2; 若 2 个元素不相等并且同时属于字母表, 则得分为 –1; 若 2 个元素中有任意一个为空位, 则得分为 –2.

序列 S 和 T 形成的一个比对 A, 序列 S', T' 是插入空位 "–" 后的序列 S 和 T, 则必须满足 $|S'| = |T'|$, 并且序列 S'、T' 去掉空位之后就是序列 S 和 T. 序列 S'、T' 形成的比对 A 的得分表示为 $\text{score}(A) = \sum_{i=1}^{s'} \sigma(S'[i], T'[j])$. 比对的得分越高, 表示两组序列的相似性越高. 而在进行 DNA 双序列比对时, 得分最高的比对是最好的比对算法.

2 蚁群算法在 DNA 双序列比对中的应用

2.1 蚁群算法的介绍

蚁群算法 (Ant Colony Algorithm), 又称蚂蚁算法, 是根据真实的蚁群集体活动而仿真设计出来的, 是一种群体智能进化算法, 是由意大利学者 Macro Dorigo 等人提出的一种随机搜索算法[4]. 它是继遗传算法、模拟退火算法、神经网络等算法之后的又一种启发式搜索算法.

蚁群算法最早是应用在 TSP 问题上, 取得了很好的效果, 后来与多种优化算法结合, 得到了广泛应用, 应用领域有: 二次分配问题、图像处理、车辆疏导、电网分配、网络路由、图形着色等问题. 蚁群算法的主要优点为鲁棒性强、分布式并行计算、易于实现并且易于与其他算法的结合. 但是它也有一些缺点限制了其应用范围, 例如: 搜索时间较长、收敛速度将慢、容易陷入局部最优值等. 近年来, 许多学者对蚁群算法进行了改进, 主要体现在减少搜索时间、加快收敛速度或是如何跳出局部最优值, 并扩大其应用领域.

2.2 经典蚁群双序列比对模型

对于 2 条 DNA 序列 S 和 T, 若依据蚁群双序列比对模型 (Ant Colony Pairwise Alignment Model) 生成的比对结果矩阵如下图 1:

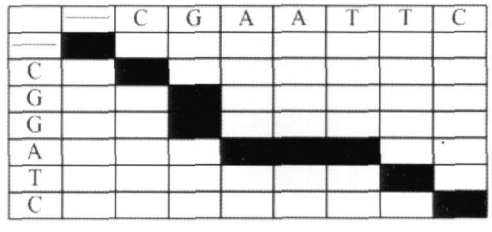

图 1 序列比对结果矩阵

Fig.1 Pair-wise sequence alignment matrix

其中序列 S = CGGATC, 序列 T = CGAATTC

则由上图可知, 加入空格之后, 得出的比对结果为:

S' = CGGA – – TC,

T' = CG–A A T TC.

主要步骤如下:

1) 根据输入的 2 个 DNA 序列和公式 1 产生匹配矩阵, 其中正数为原值, 负数取倒数的绝对值, 即相等的为 2, 不相等且无空格的为 1, 有空格的为 0.5. 例如, 若 2 个序列元素不相等并且不为空格, 根据公式 1, 得分为 –1, 则匹配矩阵中取倒数的绝对值, 故取 1;

2) 分配每个蚂蚁由起点开始进行搜索, 直到终点为止. 如果搜索循环次数达到最大循环数或是连续 N 次最优值没有改变, 则算法结束;

3) 蚂蚁在搜索过程中, 有右下方、下方和右方 3 个方向进行选择路径. 首先产生 1 个随机数, 如果这个随机数小于某个固定值 P_0, 则选择右下方移动; 如果这个随机数大于该固定值, 则需要计算 3 个方向的转移概率 P_k, 并根据轮盘赌法[5]在 3 个方向中决定 1 个方向进行移动. 在这一步中, 需要记录每只蚂蚁的转移路径和比对分数;

4) 当蚂蚁走到最右方时,规定蚂蚁只能向下方移动;当蚂蚁走到最下方时,规定蚂蚁只能向右方移动.在移动过程中,如果蚂蚁向下方移动,则在 T 序列的相应位置加入一个空位“-”;如果蚂蚁向右方移动,则在 S 序列的相应位置加入一个空位“-”;

5) 当所有蚂蚁到达终点时,则规定一轮搜索完毕.根据路径和得分来更新信息素矩阵,并记录最高分和平均分;

6) 循环进入第(2)步.

3　改进的智能蚁群算法在 DNA 序列中的应用

3.1　改进的智能蚁群算法的策略

3.1.1　转移概率的计算

在经典蚁群算法中,转移概率的计算是由信息素矩阵、字符匹配矩阵和方向权值 3个因素来决定的.在改进的智能蚁群算法中对于信息素矩阵与字符匹配矩阵进行了改进.

改进算法中,信息素矩阵定义为 $\boldsymbol{A}(i,j,k)$,其中 i表示序列 T的位置,j表示序列 S的位置,k表示蚂蚁选择的 3个位置,有右方、下方和右下方.$\boldsymbol{A}(i,j,k)$ 表示蚂蚁走过的边上的信息素.

经典蚁群算法中,在第一步进行字符匹配矩阵的构建.而在改进蚁群算法中,在蚂蚁选择路径的同时构建字符矩阵.若蚂蚁向下走,在匹配矩阵的值 $D_1 = 0.5$;若蚂蚁向右走,在匹配矩阵的值为 $D_2 = 0.5$;若蚂蚁向右下方走,并且 2个字符相等,则在匹配矩阵的值为 $D_3 = 2$,否则 $D_3 = 1$.这种匹配矩阵的构建方法可以提高蚁群算法的精确性.

d_k 为方向权值,用来控制蚂蚁偏离对角线的参数,防止插入过多的空格.如果蚂蚁当前的位置为 (i,j),d_1表示向下的方向权值,d_2表示向右的方向权值,d_3表示对角线的方向权值.如果 i与 j之间差的绝对值小于 H,则不需要调整 3个方向权值的大小.如果 i比 j大,i与 j之间的差大于 H,则证明蚂蚁向下走的太多,则增加对角线和向右走的方向权值.如果 j比 i大且 j与 i之间的差大于 H,则证明蚂蚁向右走的太多,则增加对角线和向下走的方向权值.

改进算法中转移概率的公式 2为:

$$P_k = \frac{\boldsymbol{A}(i,j,k)^{\wedge a} + D_k^{\wedge b} + d_k^{\wedge c}}{\sum_{k=1}^{3}\boldsymbol{A}(i,j,k)^{\wedge a} + D_k^{\wedge b} + d_k^{\wedge c}}.$$

3.1.2　动态更新 P_0

在经典蚁群算法中,蚂蚁搜索过程中,先产生一个随机数,如果这个随机数小于某个固定值 P_0,则选择右下方移动;否则根据转移概率和轮盘赌选择方向.改进的智能蚁群算法中,将 P_0 改成智能变化的值,公式 3为:

$$P_0(t) = \begin{cases} p_1(0 < t \leqslant T_1), \\ p_2(T_1 < t \leqslant T_2), \\ p_3(T_2 < t \leqslant T_3). \end{cases}$$

在循环次数小于等于常数 T_1 时,将 P_0 定义为比较大的值,让蚂蚁倾向选择比对分数比较大的右下方向,减少运算时间;随着计算时间增加,当循环次数小于等于常数 T_2 时,将 P_0 定义为原经典蚁群算法中的 0.3;当循环次数大于常数 T_2时,将 P_0定义为比较小的值,让蚂蚁倾向于随机选择路线,可以防止进入局部最优.

3.1.3　信息素智能更新

针对经典蚁群算法,改进的智能蚁群算法增加了对最大值路径的信息素更新策略.信息素更新策略的公式为:

$\boldsymbol{A}(i,j,k) = (1-\rho)\boldsymbol{A}(i,j,k) + \Delta A$($\Delta A$ 为信息素的增量)

$$\Delta A = \begin{cases} Q_1 * \text{point}(n) & (\text{point} \geqslant 0), \\ Q_1 * (1 + \text{point}(n)/\text{maxmisu}) & (\text{point} < 0), \\ Q_2 * \text{point}(n) & (\text{point} = \text{maxpoint}). \end{cases}$$

其中, xinX isu表示需找路径过程中蚂蚁产生的信息素, Q_1, Q_2 表示信息素更新时的倍数. 例如, 当第 n 只蚂蚁的分数等于当前所有循环的最大值, 则它所走过的路径上的信息素更新为 0. 2倍的信息素. 这样可以使蚂蚁尽快地找到最大值, 并且不容易进入局部最优值.

3. 2 改进的智能蚁群算法的实验分析

对两组数据进行了实验, 第一组数据是 DNA 序列 S = a c g c t c g c a g g a c a c t t t t t c a g a t c t a t t g a c t t c a g g a c g g c t g c t a a t a c; DNA 序列 T = a c c t c g c a g g a g a c t t t t t a c a g a a t c t a t g a c t t c a g a c g g c a g c t a t t a c; 其中序列 S 的长度为 53, 序列 T 的长度为 52. 第二组数据是从 NCBI序列库中取得的两组 DNA 序列, 序列号分别为: AB023276和 AB023278, 序列长度均为 457.

相关参数如下: 初始信息素 = 5, 方向权值 $d_1 = 2$, $d_2 = 2$, $d_3 = 3$, 防止蚂蚁偏离对角线的参数 $H = 5$, 最大循环次数 NCmax = 100, 蚂蚁数 $m = 100$, 信息挥发程度 $p = 0.05$, $q_0 = 0.3$, $p_1 = 0.6$, $p_2 = 0.35$, $p_3 = 0.2$, $a = 5$, $b = 3$, $c = 2$, $T_1 = 50$, $T_2 = 79$, $T_3 = 99$, $Q_1 = 0.1$, $Q_2 = 0.2$

第一组的计算结果如图 2

从图 1可以看出, 基于经典蚁群算法和改进蚁群算法得到的最优值都是 81, 但是改进蚁群算法在第 19次达到了最优值 81, 并保持 81分不变; 经典蚁群算法在第 23次达到最优值 81, 可以看出改进蚁群算法可以提高算法的收敛速度, 减少计算时间.

第二组的计算结果如图 3

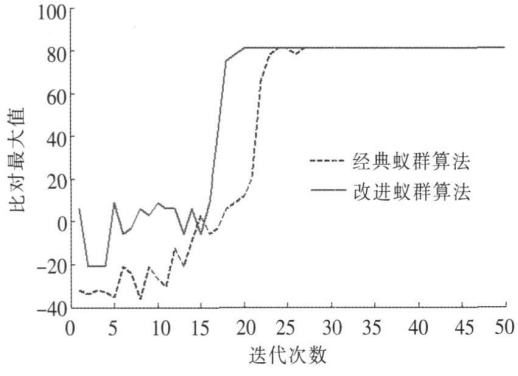

图 2 第一组数据计算结果

Fig.2 Results for sequence alignment (Case 1)

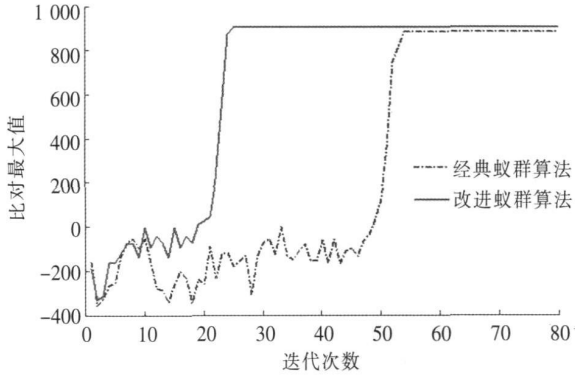

图 3 第二组数据计算结果

Fig.3 Results for sequence alignment (Case 2)

从图 3可以看出, 基于经典蚁群算法在第 59次得到的最优值 881, 并保持不变; 基于改进蚁群算法, 在第 20次得到的最优值 902, 并保持不变. 保持最优值不变的 10次迭代中, 基于经典蚁群算法的 10次平均数为 828. 89, 基于改进蚁群算法的 10次平均数为 862. 从以上比较可以看出, 改进蚁群算法在对于比较长的双序列进行分析时, 可以避免进行局部最优值, 并能在较短的时间内得到最优值, 具有很大的提高, 并且在平均值上面也有很大的改进.

针对第二组数据, 分别利用经典蚁群算法和改进蚁群算法进行了 20次试验, 得到的结果如表 1. 可以看出, 基于经典蚁群算法并不是每次都能取得最优值 881, 而基于改进蚁群算法总是可以取得最优值 902. 则表明基于改进蚁群算法的稳定性较高, 并且运算的平均数要明显优于经典蚁群算法.

表 1 各算法比对结果分析

Table 1 Performance of the algorithm in sequence alignment

	经典蚁群算法	改进蚁群算法
最优解	881	902
最低值	826	902
平均数	849	902

4 总结

经典蚁群算法在对短序列进行比对时, 具有优势, 但是对于较长的序列容易陷入局部最优值且收敛速度较慢. 本文提出的改进的智能蚁群算法, 通过实验证明, 改进蚁群算法在加快收敛速度和避免进入局部最优值等方面都具有显著的改进, 并在解决长序列比对具有显著的优势.

[参考文献]

[1] Needleman S B, Wunsch C D. A general method applicable to the search for similarities in the amino acid sequences of two proteins[J]. Journal of Molecular Biology, 1970(48): 443-453.
[2] Smith T F, Waterman M S. Identification of common molecular sequences[J]. Journal of Molecular Biology, 1981(147): 195-197.
[3] Ye Yuzhen, Adam Godzik. Multiple flexible structure alignment using patial order graphs[J]. Bioinformatics, 2005, 21(10): 2362-2369.
[4] Dorigo M. Optinization learning and natural algorithm[D]. Italy: Politecnico di Milano, 1992.
[5] 王小平, 曹立明. 遗传算法-理论应用和软件实现[M]. 西安: 西安交通大学出版社, 2002.
[6] 段海滨. 蚁群算法原理及其应用[M]. 北京: 科学出版社, 2005: 144-148.
[7] Chen Yixin, Pan Yi, Chen Juan, et al. Multiple sequence alignment by ant colony optimization and divide-and-conquer[C] // Brelin: Proc of ICCS, 2006: 646-653.
[8] Jangam S R, Chakraborti N. A novel method for alignment of two nucleic acid sequences using ant colony optimization and genetic algorithms[J]. Applied Soft Computing, 2007, 7(3): 1121-1130.
[9] 梁栋, 霍红卫. 自适应蚁群算法在序列比对中的应用[J]. 计算机仿真, 2005, 22(1): 100-102.
[10] Stefan Schroedl. An improved search algorithm for optimal multiple sequence alignment[J]. Journal of Artificial Intelligence Research, 2005, 23(5): 587-623.

[责任编辑: 顾晓天]

(上接第 147页)

[参考文献]

[1] 刘开瑛. 中文文本自动分词和标注[M]. 北京: 商务印书馆, 2000: 162-166.
[2] 张虎, 郑家恒. 基于分类的汉语语料库词性标注一致性检查[J]. 计算机工程, 2008, 34(8): 90-92.
[3] 周强. 规则和统计相结合的汉语词类标注方法[J]. 中文信息学报, 1995, 9(3): 1-10.
[4] 白栓虎. 汉语词切分及词性自动标注一体化方法[J]. 中文信息, 1996(2): 46-48.
[5] 刘群, 张华平, 俞鸿魁, 等. 基于层叠隐马模型的汉语词法分析[J]. 计算机研究与发展, 2003, 41(8): 1421-1428.
[6] 钱揖丽, 郑家恒. 汉语语料词性标注自动校对方法的研究[J]. 中文信息学报, 2003, 18(2): 33-35.
[7] 邓乃扬, 田英杰. 支持向量机——理论、算法与拓展[M]. 北京: 科学出版社, 2009: 79-111.
[8] Lafferty J, McCallum A, Pereira F. Conditional random fields: probabilistic models for segmenting and labeling sequence data[C]//Proceedings of the 18th ICML. San Francisco: Mogan Koufmann, 2001: 282-289.
[9] 丁德鑫, 曲维光, 徐涛, 等. 基于 CRF模型的组合型歧义消解研究[J]. 南京师范大学学报: 工程技术版, 2008, 8(4): 73-76.
[10] Adwait Ratnaparkhi. A simple introduction to Maximum Entropy Models for natural language process[R]. Philadelphia: University of Pennsylvania, Tech Rep: IRCS-97-08, 1997.
[11] 俞士汶, 段慧明, 朱学锋, 等. 北京大学现代汉语语料库基本加工规范[J]. 中文信息学报, 2002, 16(5): 49-64.
[12] 俞士汶, 段慧明, 朱学锋, 等. 北京大学现代汉语语料库基本加工规范(续)[J]. 中文信息学报, 2002, 16(6): 59-63.
[13] 郭永辉, 吴保民, 王炳锡. 一种用于词性标注的相关投票融合策略[J]. 中文信息学报, 2007, 21(2): 9-13.
[14] 姜维, 关毅, 王晓龙. 基于条件随机域的词性标注模型[J]. 计算机工程与应用, 2006, 21: 13-16.

[责任编辑: 顾晓天]