文章编号: 1001-0920(2018)11-1990-07 DOI: 10.13195/j.kzyjc.2017.0788

基于多策略人工蜂群的多序列比对算法

匡芳君[1,2†], 张思扬[1], 刘传才[2]

(1. 温州商学院 信息工程学院, 浙江 温州 325035; 2. 南京理工大学 计算机科学与工程学院, 南京 210094)

摘 要: 多序列比对是生物信息学中最重要和最具挑战性的任务之一. 基于多序列比对是NP完全组合优化问题, 引入Tent混沌初始化种群策略、不同蜂种的邻域搜索策略和锦标赛选择策略等, 提出一种基于多策略人工蜂群的多序列比对算法. 该算法应用Tent混沌初始化种群策略以使初始个体多样化并获取较好初始解; 针对不同蜂种的特性设计不同的邻域搜索策略以平衡算法的全局探索和局部开发能力. 同时引入序列比对的蜜源编码方法以适应多序列比对的离散性. 实验结果表明, 所提出算法的鲁棒性较强, 能获取较好的比对性能和生物特性.

关键词: 人工蜂群算法; 多策略; Tent混沌初始化; 邻域搜索; 多序列比对

中图分类号: TP18 **文献标志码:** A

Multiple sequence alignment algorithm based on multi-strategy artificial bee colony

KUANG Fang-jun[1,2†], *ZHANG Si-yang*[1], *LIU Chuan-cai*[2]

(1. School of Information Engineering, Wenzhou Business College, Wenzhou 325035, China; 2. School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract: Multiple sequence alignment(MSA), known as NP-complete combinatorial optimization problem, is one of the most important and challenging tasks in bioinformatics. A multi-strategy artificial bee colony(MS-ABC) algorithm is proposed for MSA, which is composed of multiple strategies, such as the Tent chaotic initialization population strategy, different neighborhood search strategies and tournament selection strategy. In the MSA-ABC algorithm, the Tent chaotic initialization population strategy is presented to diversify the initial individuals and to obtain good initial solutions. Then, the different neighborhood search strategies for different bee species are designed to balance the global exploration and the local exploitation. Moreover, the food source encoding method is used to adapt discreteness of MSA. The experimental results demonstrate that the proposed algorithm is more robust and can obtain better alignment quality and biological characteristics.

Keywords: artificial bee colony; multi-strategy; Tent chaotic initialization; neighborhood search; multiple sequence alignment

0 引 言

多序列比对(Multiple sequence alignment, MSA)是生物信息学研究的热点之一, 通过多序列比对可以挖掘生物序列中的结构、进化和功能等信息, MSA也是一个NP完全组合优化难题[1]. 自然启发式算法作为一类适用于求解NP难题的优化算法, 具有精度高、对度量标准不敏感等优势, 吸引着大量研究人员, 其研究成果颇丰. 如Gupta等[2]提出了一种基于遗传算法的多序列比对方法(MSA-GA); Gao[3]提出了一种基于惯性权重粒子群优化算法(MS-PSO); Tsvetanov等[4]提出了一种基于改进蚁群算法的多序列比对方法(MS-ACO); Öztürk等[5]提出了一种新的人工蜂群算法的多序列比对方法(ABC-Aligner); Zhu等[6]提出了一种基于分解的多目标进化算法(MOMSA)以解决多序列比对问题, 初始种群通过使用空位插入操作生成, 进化操作利用遗传操作的交叉和变异算子实现; Liu等[7]提出了一种基于隐马尔可夫模型和后验概率分配函数的多序列比对算法(MSAProbs); Rani等[8]提出了基于遗传和人工蜂群的混合算法(GA-ABC)和多目标细菌觅食优化算法(MO-BFO), 并应

收稿日期: 2017-06-20; **修回日期:** 2017-08-22.

基金项目: 国家自然科学基金项目(61373063, 61233011, 61402227).

责任编委: 巩敦卫.

作者简介: 匡芳君(1976–), 女, 教授, 博士, 从事群智能与多目标优化、模式识别、生物信息学及其应用等研究; 刘传才(1963–), 男, 教授, 博士生导师, 从事智能计算、计算机视觉、图像处理及应用等研究.

†通讯作者. E-mail: kfjztb@126.com

用于多序列比对,但在工作中主要集中于MO-BFO算法; Sun等[9]提出了一种基于量子粒子群优化和隐马尔可夫模型的多序列比对算法; Rubio-Largo等[10]结合多目标和进化启发式算法,提出了一种基于混合蛙跳算法的多序列比对方法; Zambrano-Vega等[11]进行了基于多目标启发式算法的多序列比对方法的比较研究,并对这些启发式方法进行了综述和比较.

人工蜂群算法(Artificial bee colony, ABC)是由Karaboga于2005年提出的群智能算法[12],具有全局探索能力强、控制参数少、收敛速度快和鲁棒性强等优势,被广泛应用于求解复杂优化问题[13],但当其接近全局最优解时,种群多样性减少,搜索速度变慢,甚至易陷入局部最优,因此也衍生了很多改进算法和应用[14-16]. 本文在研究多序列比对算法作为多目标优化问题的基础上,在人工蜂群算法中引入Tent混沌初始化种群策略、不同蜂种的邻域搜索策略和锦标赛选择策略等,提出了一种基于多策略人工蜂群的多序列比对算法.

1　多序列比对优化模型

多序列比对主要反映给定序列之间的进化关系. 目的是使参与比对的序列尽可能相似(或距离最小),即相同残基的位点位于同一列. 多序列比对问题描述如下[6]: 假设参与比对的序列组 $S = (S_1, S_2, \cdots, S_i, S_N)$ 含有 N 条序列,每条序列长度分别为 $l_1, l_2, \cdots, l_N$,序列中字符由4种核苷酸或20种氨基酸组成. 通过插入或删除空位找到一个多序列比对 S',使对齐的字符数最多. 多序列比对 S' 的约束条件为: 1) S' 中各条比对后的序列长度相等; 2) 若删除 S' 中的第 i 行所有空位,则 $S'_i = S_i$; 3) 使多序列比对 S' 的评分 $\mathrm{Score}(S')$ 最大.

多序列比对常用目标函数主要有: WSP权重计分函数,隐Markov模型, COFFEE计分函数等[6]. 本文采用WSP函数,其评分函数 $\mathrm{Score}(S')$ 表示为

$$\mathrm{Score}(S') = \sum_{l=1}^{L} S'_l. \tag{1}$$

$$S'_l = \sum_{i=1}^{N-1} \sum_{k=i+1}^{N} w_{ij} p(A_i, A_j). \tag{2}$$

$$p(A_i, A_j) = \begin{cases} 0,\ A_i = A_j = '-'; \\ -g_{\mathrm{open}},\ (A_i = '-' \text{ and } A_j \neq '-') \text{ or} \\ (A_i \neq '-' \text{ and } A_j = '-') \text{ and is opening}; \\ -g_{\mathrm{extend}},\ (A_i = '-' \text{ and } A_j \neq '-') \text{ or} \\ (A_i \neq '-' \text{ and } A_j = '-') \text{ and is extension}; \\ \mathrm{Blosum}(A_i, A_j),\ A_i \neq '-' \text{ and } A_j \neq '-'. \end{cases} \tag{3}$$

其中: L 为多序列比对 S' 的长度; S'_l 为多序列比对 S' 第 l 列的代价; N 为序列比对组中序列的条数; w_{ij} 为序列 s_i 和 s_j 的权重,取决于序列间的距离,本文利用编辑距离(Levenshtein distance, LD)来计算 w_{ij},令 $w_{ij} = 1 - \dfrac{\mathrm{LD}(s_i, s_j)}{\max(|s_i|, |s_j|)}$,其中 $\max(|s_i|, |s_j|)$ 为两条序列 s_i 和 s_j 长度的最大值; $p(A_i, A_j)$ 为序列中残基 A_i 和 A_j 的评分; $g_{\mathrm{open}}, g_{\mathrm{extend}}$ 分别为单位空位开放罚分和单位空位扩展罚分; $\mathrm{Blosum}(A_i, A_j)$ 为根据残基片段替换矩阵(Blocks substitution matrix)计算残基匹配评分[6]. 使序列间匹配的残基对个数最大,而空位罚分最小是多序列比对追求的目标,故多序列比对多目标优化函数表示如下:

$$\begin{aligned} &\max\text{imize } F(S') = (f_1(S'), f_2(S'))^{\mathrm{T}}, \\ &f_1(S') = \mathrm{Score}(S'), \\ &f_2(S') = -g(S'), \\ &S' \in \Omega. \end{aligned} \tag{4}$$

其中: $g(S') = n_1 \times g_{\mathrm{open}} + n_2 \times g_{\mathrm{extend}}$ 为对齐序列组 S' 的空位罚分, n_1, n_2 分别为空位开放的数量和空位扩展的数量; Ω 为多序列比对空间. 本文选用Blosum 62替换矩阵, $g_{\mathrm{open}} = 6, g_{\mathrm{extend}} = 0.85$.

例1　假设参与比对的序列组为 S,通过序列比对后得到的对齐序列比对为 S',结果如图1所示.

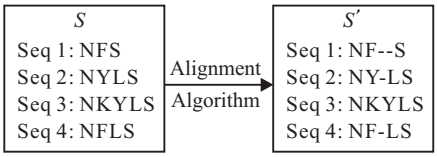

图 1　多序列比对实例

1) 计算序列组 S 中每一对序列间的权重 w_{ij},即 $w_{12} = 1 - \dfrac{\mathrm{LD}(s_1, s_2)}{\max(|s_1|, |s_2|)} = 1 - \dfrac{2}{4} = 0.5$. 同理, $w_{13} = 0.4, w_{14} = 0.75, w_{23} = 0.8, w_{24} = 0.75, w_{34} = 0.6$.

2) 按式(3)和(4)计算序列 S' 每一列代价值,即

$$\begin{aligned} &S'_1 = \\ &w_{12} \times p(S'_{11}, S'_{21}) + w_{13} \times p(S'_{11}, S'_{31}) + \\ &w_{14} \times p(S'_{11}, S'_{41}) + w_{23} \times p(S'_{21}, S'_{31}) + \\ &w_{24} \times p(S'_{21}, S'_{41}) + w_{34} \times p(S'_{31}, S'_{41}) = \\ &(0.5 \times \mathrm{Blosum}(N, N)) + (0.4 \times \mathrm{Blosum}(N, N)) + \\ &(0.75 \times \mathrm{Blosum}(N, N)) + (0.8 \times \mathrm{Blosum}(N, N)) + \\ &(0.75 \times \mathrm{Blosum}(N, N)) + (0.6 \times \mathrm{Blosum}(N, N)) = \\ &22.8. \end{aligned}$$

同理, $S'_2 = 3.45, S'_3 = -10.8, S'_4 = 7.197\,5, S'_5 = 15.2$.

3) 按式(1)计算对齐序列 S' 的评分 $\mathrm{Score}(S') = 37.847\,5$.

4) 计算对齐序列组S'的空位罚分$g(S') = n_1 \times g_{\text{open}} + n_2 \times g_{\text{extend}} = 18.85$. 因此,此对齐序列组$S'$的目标函数值为$f_1 = 37.847\,5, f_2 = -18.85$.

2 多策略人工蜂群算法

为更好地均衡ABC算法的全局探索能力与局部开采能力之间的关系,本文利用Tent混沌初始化种群,多种邻域搜索和锦标赛选择等策略,提出多策略人工蜂群算法(Multi-strategy ABC, MS-ABC).

2.1 ABC算法

ABC算法[11]的蜂群分为引领蜂(Employed bee)、跟随蜂(Oonlooked bee)和侦察蜂(Scout bee). 可行解的质量或适应度用蜜源的丰富程度表示,问题的最优解用最大适应度的蜜源表示. 优化问题的可行解通过式(5)随机产生蜜源的位置来表示. 引领蜂与蜜源一一对应,引领蜂的数量与跟随蜂数量相等,且等于蜂群总数的一半.

$$X_i^j(0) = X_{\min}^j + R \times (X_{\max}^j - X_{\min}^j). \tag{5}$$

其中: $X_i^j(0)$为初始化时第i个蜜源的第j维向量; $i = 1, 2, \cdots, N, j = 1, 2, \cdots, D$, N为蜜源数量, D为问题空间的维数; $X_{\max}^j, X_{\min}^j$分别为第j维向量的最大值和最小值; R为[0,1]间的随机数. 蜜源产生后就分派给引领蜂. 为了能根据旧蜜源位置X_i^j产生新蜜源位置V_i^j, ABC算法的基本搜索策略[11]为

$$V_i^j(t) = X_i^j(t) + \Phi(X_i^j(t) - X_k^j(t)). \tag{6}$$

其中: $i = 1, 2, \cdots, N, j = 1, 2, \cdots, D, k = 1, 2, \cdots, N$为随机选择的下标,且$k \neq i$; Φ为$[-1, 1]$之间的随机数.

如果一个蜜源位置循环蜜源搜索限定次数Limit后仍不能更新,则该蜜源处的引领蜂变成侦察蜂,并且该侦察蜂按式(5)搜索策略在解空间中产生新蜜源位置来替换原蜜源位置.

2.2 Tent混沌初始化种群策略

为了使产生的初始个体尽可能均匀分布,并且让种群保持多样性,考虑到Tent映射比Logistic映射具有更好的收敛速度和遍历均匀性,采用Tent混沌初始化种群[15]产生蜜源位置,既能保留算法初始化的随机性,又能提高种群多样性,从而择优产生蜜源位置. Tent映射通过贝努利移位变换后可以表示为$x_{t+1} = (2x_t) \bmod 1$,计算机进行Tent映射运算时,小数部分的二进制数进行无符号左移[15].

2.3 锦标赛选择策略

ABC算法的跟随蜂如果利用比例选择策略进行蜜源的选择,则可能导致较差蜜源不能得到较好地更新,使算法种群多样性减少,从而导致算法后期个体适应度在种群进化时趋于一致,进而使算法不易跳出局部最优. 因此,本文采用锦标赛选择策略[15]选择蜜源. 该策略是基于局部竞争机制的选择过程,只根据适应度值的相对值来选择蜜源,从而避免超级个体对算法的影响. 锦标赛选择概率为

$$P_i(t) = \frac{c_i(t)}{\sum_{i=1}^{N} c_i(t)}, \tag{7}$$

其中c_i为每个个体的得分.

2.4 多邻域搜索策略

在求解复杂优化问题时,如何平衡全局探索与局部开发能力是群智能算法提升性能的关键[16]. ABC算法前期全局探索能力强,收敛速度快,后期局部开采能力弱,收敛速度慢. 同时考虑到多序列比对是一个特殊的离散问题,式(5)、(6)不能直接用于多序列比对问题. 因此,本文将根据蜂群的种类设计不同的邻域搜索策略,以克服基本人工蜂群算法易于陷入局部最优解的缺陷.

2.4.1 引领蜂邻域搜索

在引领蜂邻域搜索时,既考虑个体E_i,又考虑整个种群迄今为止搜索到的最优引领蜂(蜜源)E_{best},对任一引领蜂E_i和当前最优引领蜂E_{best}通过单点交叉操作产生新的引领蜂V_i,即在蜜源E_{best}选取较好交叉位块,以替代E_i中相应的序列位块,若不足序列长度,则在中间补足空位. 假设当前最优蜜源E_{best}为

S_1': GARFIE- LDTHE- - - - FAST- - - CAT

S_2': GARFI-ELDTHEVERYFAST- - - CAT

S_3': GARFI- ELDTHE- - - -LASTFAT CAT

S_4': - - - - - - - - - THE - - - - FA- T - - - CAT

引领蜂E_i为

S_1': GA-RFIELD- -THE- - - FAST- CAT - -

S_2': GA-RFIELD- -THEVERYFAST– CAT

S_3': G- -ARFIELDT-HEL- -AST-FAT CAT

S_4': - - - -THEFAT- - - - - - - - - - CAT- - - -

则对E_i和E_{best}通过单点交叉操作后产生新的引领蜂V_i为

S_1': GA-RFIELD- -THE- - - FAST- - - CAT

S_2': GA-RFIELD- -THEVERYFAST- -CAT

S_3': G- -ARFIELDT-HEL- -AST- FATCAT

S_4': - - - -THEFAT- - - - - - - - - - - - - - CAT

2.4.2 跟随蜂邻域搜索

跟随蜂根据锦标赛选择操作选中引领蜂E_i作为蜜源时,跟随蜂在蜜源邻域位置通过变异操作产生一个新的蜜源位置V_i,为了提高算法精度,引入一个新

的变异操作,其过程如下:

1) 移动空位组: 对各条序列随机选取一个空位组,对该空位组向左或向右移动一个残基位.

例如,假设选中的引领蜂E_i如下:

S_1': GA- -RFIELD- -THE- - -FAST- - - CAT

S_2': GA- -RFIELD- -THEVERYFAST- -CAT

S_3': G- - -ARFIELD-THE- - -LASTFATCAT

S_4': - - - - -THEFA- - - - - - - - - - - - - - CAT

假设随机选中第一个空位组,向右移动一个残基位后得到的V_i如下:

S_1': GAR- -FIELD- -THE- - -FAST- - - CAT

S_2': GAR- -FIELD- -THEVERYFAST- -CAT

S_3': GA- - -RFIELD-THE- - -LASTFATCAT

S_4': T- - - - -HEFA- - - - - - - - - - - - - - CAT

2) 合并空位组: 先随机选择一条序列,接着在此序列中随机选择一组空位,并将它与最近的空位组合并. 假设随机选择的是第 2 条序列第 1 个空位组,则合并后V_i的结果如下:

S_1': GAR- FIELD- -THE - - -FAST - - - CAT

S_2': GARFIELDG- - - THEVERYFAST- -CAT

S_3': GA- -RFIELD-THE - - - LASTFATCAT

S_4': T - - - - HEFAT - - - - - - - - - - - - - - CAT

3) 划分空位组: 先随机选择一条序列,接着在此序列中随机选择一组空位,并将此空位组划分成两个大小相等的空位组. 假设随机选择的是第 1 条序列第 2 个空位组,划分后V_i的结果如下:

S_1': GAR- FIEL- -D- -THE - - -FAST - - - CAT

S_2': GARFIELD- - -THEVERYFAST- -CAT

S_3': GA- -RFIELD -THE- - - LASTFATCAT

S_4': T - - - -HEFAT - - - - - - - - - - - - - - CAT

4) 删除全为空位的列: 由于序列中不允许存在全为空位的列,应将全为空位的列删除,例如删除 3) 中全为空位的列后产生新蜜源V_i为

S_1': GAR- FIEL-DTHE - - -FAST - - - CAT

S_2': GARFIELD- -THEVERYFAST- - CAT

S_3': GA- -RFIELDTHE - - - LASTFATCAT

S_4': T - - - - HEFAT - - - - - - - - - - - - - CAT

2.4.3　侦察蜂邻域搜索

对于经过Limit都没有更新的蜜源,利用Kalign 2比对算法对该蜜源中局部块重新比对产生新的蜜源V_i替代原蜜源,其过程如下:

1) 随机选取序列中某个块,块大小为序列长度的 5 %～25 %之间. 假设蜜源选中块如阴影部分

S_1': GAR- FI EL-DTH E- - -FAST- - - CAT

S_2': GARFIE LDT- -H EVERYFAST- - CAT

S_3': GA- RFI -ELDTH E- - - LASTFATCAT

S_4': T - - - - H EFAT - - - - - - - - - - - - CAT

2) 删除被选中块的所有空位,结果如下:

S_1'中的块: ELDTH

S_2'中的块: LDTH

S_3'中的块: ELDTH

S_4'中的块: EFAT

3) 利用Kalign 2比对算法对删除空位后的块重新生成对齐序列比对,结果如下:

S_1'中的块: ELDTH

S_2'中的块: - LDTH

S_3'中的块: ELDTH

S_4'中的块: EFAT -

4) 最后将 3) 中产生的序列比对结果替代原序列中的选中块,得到一个新的蜜源V_i,即

S_1': GAR- FI ELDTH E - - - FAST- - - CAT

S_2': GARFIE - LDTH EVERYFAST- - CAT

S_3': GA- RFI ELDTH E - - - LASTFATCAT

S_4': T- - - - H EFAT - - - - - - - - - - - CAT

3　基于多策略人工蜂群的多序列比对算法

3.1　编码设计

利用MS-ABC算法解决多序列比对问题时,采用式(4)作为目标函数,并需要解决以下 4 个问题:

1) 初始化.

假设有N条序列参与比对,且各序列长度依次为$l_1, l_2, \cdots, l_N$. 在种群初始化时,随机产生$l_{\max}$与$1.2\,l_{\max}$间的随机数L作为每个个体的长度,其中$l_{\max} = \max(l_1, l_2, \cdots, l_N)$.

2) 个体编码.

由于多序列比对问题具有特殊性,本文采用二维矩阵对个体进行编码. 假设有N条序列参与比对,各序列长度依次为$l_1, l_2, \cdots, l_N$,比对后序列长度为L,则根据多序列比对模型,每条序列需要插入的空位数分别为$L - l_i$,其中$1 \leqslant i \leqslant N$. 因此,随机生成$N$个 1 到$L - l_i$随机排列的一维矩阵$a_i$. 则某个蜜源的编码$\gamma = [a_1, a_2, \cdots, a_N]$. 假设有两条序列$S_1 = YDETPIKASER, S_2 = YDEILYQSKRF$参与比对,则可能的蜜源编码$\gamma = [a_1, a_2]$,其中$a_1 = [2, 8, 11, 13], a_2 = [2, 4, 10]$.

3) 与蜜源编码相对应的比对结果.

在编码实现时,因矩阵存储的是插入空位的位置,在输入比对结果时,需将编码转化成对应比对,将比对结果以矩阵形式存储. 其中矩阵的行是插入空位后的各条序列,如序列$S_2 = YDEILYQSKRF$,插入空位位置为$a_2 = [2, 4, 10]$,则插入空位后序列为

$S_2 = Y - D - EILYQ - SKRF.$

4) 混沌映射产生混沌序列.

利用Tent映射产生的混沌序列表示插入空位的位置,故应对映射后的序列做取整操作,并判断欲插入的空位位置是否已存在. 若已存在,则在剩余位置中任选一位置插入,否则直接插入到该位置.

3.2 算法流程

基于MS-ABC多序列比对算法步骤描述如下:

Step 1: 设置多序列比对算法的参数包括种群规模SN、蜜源(引领蜂)的个数为$M = \mathrm{SN}/2$、最大迭代次数$G_{\max}$、蜜源开采限制次数Limit等参数,设置当前迭代次数$\mathrm{iter} = 0$, 设置同一蜜源的开采标志向量$\mathrm{trial}(i) = 0$.

Step 2: 根据“2.2 Tent混沌初始化策略”和“3.1 编码设计”产生初始蜜源插入空位的位置,初始比对的长度L在$l_{\max}$与$1.2\,l_{\max}$之间,从而形成初始化种群,并保证其对应的序列比对结果中不存在只由空位“-”组成的列.

Step 3: 对于每个个体S_i,根据式(4)计算目标函数$f_1(S_i)$和$f_2(S_i)$. 对适应度排序,取前M个个体作为蜜源位置,令$F_1 = (f_1(S_1), f_1(S_2), \cdots, f_1(S_L))$, $F_2 = (f_2(S_1), f_2(S_2), \cdots, f_2(S_N))$,则可以得到$F = (\max(F_1), \max(F_2))$为目标理想值,并记录种群当前最优位置$S_{\mathrm{best}}$.

Step 4: 引领蜂i按2.3.1节引领蜂邻域搜索策略在蜜源附近搜索产生新的序列比对V_i,并按式(4)计算适应度值$F(V_i)$.

Step 5: 如果$F(V_i) > F(S_i)$,则$S_i = V_i$,否则保持原始序列比对S_i不变.

Step 6: 根据式(7)锦标赛选择策略计算选择概率$P_i(t)$.

Step 7: 跟随蜂根据$P_i(t)$选择蜜源,按2.3.2节跟随蜂邻域搜索策略在蜜源附近搜索产生新的蜜源V_i,并按式(4)计算其适应度值$F(V_i)$.

Step 8: 如果$F(V_i) > F(S_i)$,则$S_i = V_i$,$\mathrm{trial}(i) = 0$;否则保持原始序列S_i不变,$\mathrm{trial}(i) = \mathrm{trial}(i) + 1$.

Step 9: 若$\mathrm{trial}(i) \geqslant \mathrm{Limit}$,则引领蜂$i$放弃当前蜜源变为侦察蜂,然后侦察蜂按照2.3.3节侦察蜂邻域搜索策略在解空间搜索产生新蜜源V_i替代原蜜源.

Step 10: 记录当前所有最优蜜源,保存迄今为止的最优序列比对S_{best}.

Step 11: 更新迭代次数$\mathrm{iter} = \mathrm{iter} + 1$,如果$\mathrm{iter} \geqslant G_{\max}$,则输出最优序列比对结果,否则返回Step 4.

MS-ABC多序列比对算法的流程如图2所示.

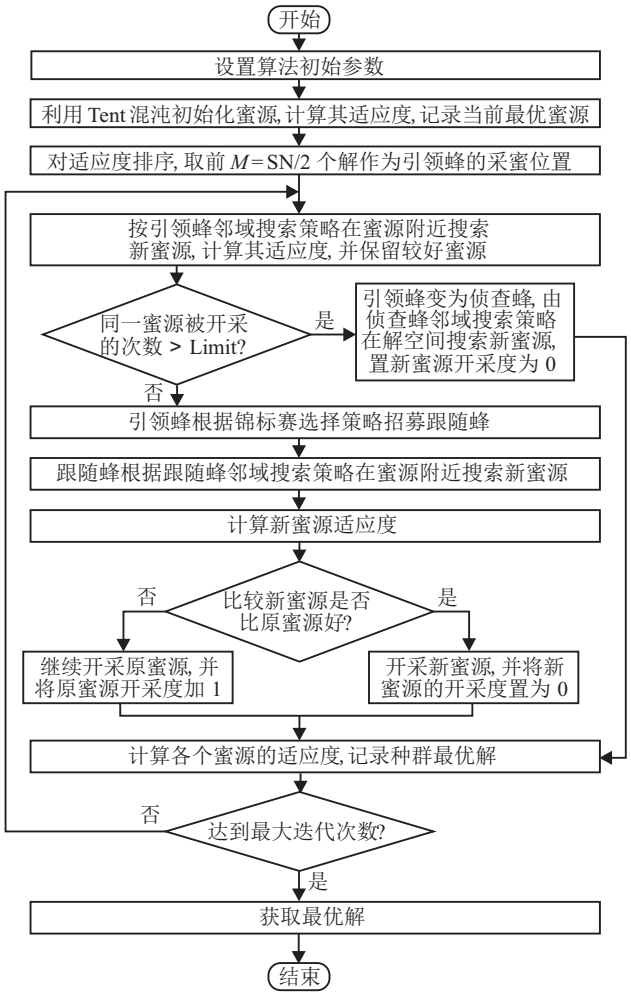

图 2 基于多策略人工蜂群的多序列比对算法流程

4 实验结果与分析

4.1 实验数据

为验证MS-ABC算法解决多序列比对的性能和有效性,采用BAliBASE 3.0数据库[17],它是基于三维重叠的标准比对库,包含6 255条序列,218个测试实例,被划分为6个序列集合: BB11, BB12, BB20, BB30, BB40, BB50,每组集合代表不同生物特性,且包含的序列相似度也不同. BB11集合中包含38个测试实例,序列间相似度小于20%; BB12集合包含44个测试实例,序列间相似度在20%到40%之间; BB20集合包含41个测试实例,序列间相似度大于40%; BB30集合包含30个测试实例,序列间相似度大于40%; BB40集合包含49个测试实例,序列间相似度大于20%; BB50集合包含16个测试实例,序列间相似度大于20%.

4.2 实验结果与分析

为了验证本文所提算法MS-ABC的性能,分别与MSA-GA[2]、MS-PSO[3]、MS-ACO[4]、ABC-Aligner[5]、MOMSA[6]、MSAProbs[7]、MO-BFO[8]、MUSCLE[16]、Clustal Ω[18]等多序列比对算法进行比较实验. 采用

表 1　各算法解决多序列比对的SP平均值及相应标准差

算法	SP						
	BB11(38)	BB12(44)	BB20(41)	BB30(30)	BB40(49)	BB50(16)	Overall(218)
MS-ABC	**0.728 4±0.108 1**	0.951 6±0.011 2	**0.934 5±0.003 8**	**0.887 3±0.013 3**	**0.930 8±0.007 3**	0.899 6±0.011 5	**0.890 7±0.010 8**
MSA-GA	0.462 8±0.201 2	0.838 9±0.120 5	0.814 7±0.019 6	0.788 6±0.082 4	0.804 5±0.105 2	0.774 6±0.062 4	0.761 5±0.093 8
MSA-PSO	0.617 5±0.153 4	0.926 8±0.101 8	0.906 9±0.098 6	0.839 8±0.121 6	0.906 3±0.100 8	0.834 5±0.078 2	0.846 7±0.158 5
MSA-ACO	0.605 1±0.167 8	0.913 4±0.093 6	0.900 9±0.128 4	0.824 6±0.098 9	0.881 3±0.105 2	0.823 4±0.120 5	0.824 7±0.094 8
ABC-Aligner	0.695 0±0.094 5	0.942 5±0.065 6	0.924 7±0.030 2	0.853 2±0.107 9	0.922 6±0.086 4	0.888 6±0.015 8	0.875 9±0.100 8
MOMSA	0.567 8±0.124 7	0.866 5±0.098 4	0.892 5±0.008 3	0.759 5±0.183 0	0.824 1±0.009 8	0.765 8±0.104 3	0.754 6±0.158 4
MSAProbs	0.682 1±0.100 5	**0.956 4±0.068 7**	0.928 7±0.009 8	0.865 7±0.031 4	0.923 6±0.080 5	**0.901 6±0.019 8**	0.875 4±0.024 5
MO-BFO	0.635 4±0.105 8	0.934 8±0.089 6	0.902 4±0.127 8	0.843 5±0.086 7	0.894 5±0.100 2	0.846 7±0.084 6	0.834 9±0.088 7
MUSCLE	0.682 6	0.944 7	0.928 4	0.875 5	0.925 4	0.894 4	0.875 2
Clustal Ω	0.590 1	0.906 0	0.911 6	0.862 4	0.901 0	0.862 0	0.838 9

表 2　各算法解决多序列比对的TC平均值及相应标准差

算法	TC						
	BB11(38)	BB12(44)	BB20(41)	BB30(30)	BB40(49)	BB50(16)	Overall(218)
MS-ABC	**0.591 2±0.125 4**	**0.876 9±0.030 2**	**0.515 7±0.070 5**	**0.640 5±0.098 6**	**0.658 4±0.018 9**	**0.622 5±0.034 7**	**0.650 6±0.086 5**
MSA-GA	0.316 7±0.302 3	0.635 8±0.195 1	0.245 8±0.115 8	0.608 1±0.190 4	0.358 7±0.208 7	0.298 9±0.097 4	0.356 4±0.129 3
MSA-PSO	0.384 6±0.193 5	0.836 8±0.090 6	0.356 4±0.187 6	0.489 2±0.107 5	0.583 4±0.097 3	0.509 5±0.100 3	0.548 9±0.182 1
MSA-ACO	0.368 5±0.173 4	0.803 1±0.078 3	0.398 7±0.163 2	0.486 4±0.077 3	0.508 1±0.085 6	0.439 8±0.072 3	0.516 4±0.098 4
ABC-Aligner	0.486 4±0.112 3	0.870 2±0.092 7	0.446 9±0.127 6	0.568 7±0.128 5	0.625 8±0.112 4	0.558 4±0.099 8	0.584 6±0.124 0
MOMSA	0.389 5±0.140 5	0.814 3±0.112 9	0.363 8±0.119 2	0.384 1±0.098 2	0.534 6±0.009 8	0.514 6±0.187 0	0.561 8±0.101 4
MSAProbs	0.484 2±0.127 1	0.871 5±0.098 5	0.469 8±0.094 8	0.611 8±0.100 9	0.610 9±0.010 9	0.611 2±0.069 5	0.605 8±0.095 3
MO-BFO	0.432 1±0.208 2	0.814 8±0.100 9	0.401 3±0.131 1	0.501 3±0.210 6	0.545 9±0.094 5	0.486 7±0.185 4	0.578 4±0.152 8
MUSCLE	0.440 9	0.861 9	0.479 4	0.622 7	0.603 9	0.593 1	0.600 3
Clustal Ω	0.362 2	0.793 8	0.452 9	0.579 1	0.582 6	0.537 4	0.551 3

多序列比对评估指标: 残基对评分和(Sum of pairs score, SP), TC度量(Total column, TC). SP指的是算法测试得到的多序列比对与标准参考比对对齐的残基对的比例, 而TC是指所有序列中正确对齐列的比例.

所有算法在CPU主频为4 GHz, 显卡为NVidia GeForce GTX TITAN X(12 G), 内存为16 G, 操作系统为Windows 10 Enterprise 64位的PC机上运行, 在Matlab R2014 b和JAVA平台编码实现. 所有进化算法的种群规模都取SN = 100, 其中: ABC算法种群包含引领蜂50个, 跟随蜂50个; PSO算法种群包含粒子100个; ACO中种群包含100只蚂蚁; GA中种群包含100个染色体. 所有算法最大迭代次数$G_{\max} = 5\,000$, 所有ABC算法的蜜源开采限制次数Limit = 25. 所有进化算法在固定迭代次数下, 通过求解同一组序列的比对, 利用算法运行30次所得的平均SP值和TC值来考察算法的性能. 实验独立运行30次后, 获得测试结果中的SP平均值及其相应的标准差比较如表1所示, TC平均值及其相应的标准差比较如表2所示, 较好的算法结果在表中用粗体表示.

表1和表2分别表示10种算法对多序列比对的SP平均值及其标准差和TC平均值及其标准差的求解. 综合表1、表2分析可以看出, MS-ABC算法除在BB12和BB50序列比对组中的性能比MSAProbs算法的SP平均值性能稍低外, 其他情况都具有较明显优越性. 同时, 不管序列比对组的序列长度、相似度差别多大, MS-ABC算法都能得到较好、标准差较小的序列比对, 且对所有序列组比对时也能获得较好比对效果, 这表明MS-ABC算法鲁棒性和稳定性较好, 而且对BB12和BB50序列比对结果比Clustal Ω和MUSCLE专业软件得到的结果也要好, 特别是对218个测试实例中的6 255条序列比对时, MS-ABC算法获得的性能也相对而言较好, 其主要原因是MS-ABC算法在邻域搜索过程中综合考虑了全局探索能力和局部开采能力的均衡. 对于序列之间的相似性小于20 %的测试集BB11, MS-ABC算法得到SP平均值和TC平均值虽然比其他非专业比对软件的效果好, 但分别只有0.728 4和0.591 2, 主要是序列间的相似度对序列比对结果的影响较大, 因此, 数据集BB11对于MS-ABC算法仍然是一个挑战.

表3表示的是10种比对算法对BAliBASE 3.0数据库中218个测试实例的运行时间的比较. 由表3可知, Clustal Ω和MUSCLE序列比对算法的运行时间比其他启发式算法的运行时间要短, 分别是9′16″和13′29″, 主要是因为Clustal Ω和MUSCLE算法是用于多序列比对的高度专业化的软件, 它在比对序列之前, 根据序列间进化距离创建进化树; MOMSA算法

表3 各算法对BAliBASE 3.0数据库中218个测试实例的运行时间 h:m:s

算法	MSA-ABC	MSA-GA	MSA-PSO	MSA-ACO	ABC-Aligner	MOMSA	MSAProbs	MO-BFO	MUSCLE	Clustal Ω
Runtime	01:59:23	01:47:34	02:58:52	02:46:25	03:06:42	23:37:56	03:28:21	02:48:17	00:13:29	**00:09:16**

最慢,主要是因为MOMSA算法没有考虑对参数进行优化,使用了默认参数,而其他启发式序列比对算法对参数进行了优化;MS-ABC算法的运行速度相对而言稍快,下一步工作将采用多CPU或GPU优化算法,以提高算法的运行速度.

5 结 论

本文应用多策略人工蜂群算法优化多序列比对模型目标函数,提出了基于多策略人工蜂群的多序列比对算法. 算法特点主要有:1) 利用Tent混沌初始化策略提高算法种群的多样性;2) 针对不同蜂种特性设计不同的邻域搜索策略以平衡算法的全局探索与局部开发能力,从而使算法有效跳出局部最优,并快速搜索最优解;3) 引领蜂使用锦标赛选择策略招募跟随蜂,在一定程度上避免了超级个体对算法的影响;4) 引入蜜源编码方法以适应多序列比对的离散性;5) 利用多策略人工蜂群算法优化多序列比对多目标优化模型,有利于提供更多有意义的多序列比对供用户选择. 实验结果表明,所提出算法不仅能有效地求解多序列比对问题,还能获得良好的比对结果. 因此,该方法可为解决多序列比对问题提供新思路,也可为其他相关研究提供新的解决方案. 但随着比对序列数量的海量增加,本文算法也会遇到运行速度慢、内存不足和CPU处理效能等瓶颈. 另外,算法对于相似度小的数据集比对精度仍需提高. 下一步重点考虑如何利用多CPU或GPU和Spark平台搭建并行计算集群来提高算法的并行优化和扩展性,并结合隐马尔可夫模型和群智能混合算法研究多序列比对模型中的多目标优化问题,以提高序列比对质量.

参考文献(References)

[1] Wang Y X, Wang Z H. Introduction to bioinformatics: Algorithms and applications for high performance computing[M]. Beijing: Tsinghua University Press, 2011.

[2] Gupta R, Pankaj A, Soni A K. MSA-GA: Multiple sequence alignment tool based on genetic approach[J]. Int J of Soft Computing and Software Engineering, 2013, 3(8): 1-11.

[3] Gao Y X. A multiple sequence alignment algorithm based on inertia weights particle swarm optimization[J]. J of Bionanoscience, 2014, 8(5): 400-404.

[4] Tsvetanov S, Ivanova D, Zografov B. Ant colony optimization applied for multiple sequence alignment[J]. Biomath Communications, 2015, 2(1): 800-806.

[5] Öztürk C, Aslan S. A new artificial bee colony algorithm to solve the multiple sequence alignment problem[J]. Int J of Data Mining and Bioinformatics, 2016, 14(4): 332-353.

[6] Zhu H Z, He Z S, Jia Y Y. A novel approach to multiple sequence alignment using multi-objective evolutionary algorithm based on decomposition[J]. IEEE J of Biomedical and Health Informatics, 2016, 20(2): 717−727.

[7] Liu Y C, Schmidt B, Maskell D L. MSAProbs: Multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities[J]. Bioinformaics, 2010, 26(16): 1958-1964.

[8] Rani R R, Ramyachitra D. Multiple sequence alignment using multi-objective based bacterial foraging optimization algorithm[J]. Biosystems, 2016, 150(10): 177-189.

[9] Sun J, Wu X J, Fang W, et al. Multiple sequence alignment using the hidden Markov model trained by an improved quantum-behaved particle swarm optimization[J]. Information Sciences, 2012, 182(1): 93-114.

[10] Rubio-Largo A, Vega-Rodriguez M A, Gonzalez-Alvarez D L. A hybrid multiobjective memetic metaheuristic for multiple sequence alignment[J]. IEEE Trans on Evolutionary Computation, 2016, 20(4): 499-514.

[11] Zambrano-Vega C, Nebro A J, Durillo J J, et al. Multiple sequence alignment with multiobjective metaheuristics: A comparative study[J]. Int J of Intelligent Systems, 2017, 32(2): 843-861.

[12] Karaboga D. An idea based on honey bee swarm for numerical optimization[R]. Technical Report-TR06, Erciyes University, Kayseri/Turkey, 2005.

[13] Öztürk C, Hancer E, Karaboga D. Dynamic clustering with improved binary artificial bee colony algorithm[J]. Applied Soft Computing, 2015, 28(3): 69-80.

[14] Mao M X, Duan Q C. Modified artificial bee colony algorithm with self-adaptive extended memory[J]. Cybernetics and Systems: An Int J, 2016, 47(7): 585-601.

[15] Kuang F J, Jin Z, Xu W H, et al. Hybridization algorithm of Tent chaos artificial bee colony and particle swarm optimization[J]. Control and Decision, 2015, 30(5): 839-847.

[16] Wang Z G, Wang M G. Multi-search strategy of artificial bee colony algorithm based on symbolic function[J]. Control and Decision, 2016, 31(11): 2037-2044.

[17] Edgar R C. BENCH: A collection of protein sequence alignment benchmarks including BALIBASE v3, PREFAB v4, OXBENCH, and SABRE[EB/OL]. http://www.drive5.com/bench, 2016.

[18] Multiple Sequence Alignment tools[EB/OL]. [2017-02-18]. http://www.ebi.ac.uk/Tools/msa/.

(责任编辑:闫 妍)