文章编号:1006-2467(2020)06-0599-08 DOI: 10.16183/j.cnki.jsjtu.2020.99.006

计算周期序列 k-错线性复杂度的混合遗传算法

牛志华， 苑 璨， 孔得宇
(上海大学 计算机工程与科学学院,上海 200444)

摘 要: 周期序列的线性复杂度及其稳定性是序列密码评价的重要度量指标.k-错线性复杂度是线性复杂度稳定性的一个重要评价指标.然而,目前对于大部分周期序列(除周期为 2^n、p^n、$2p^n$ 外),尚无有效的算法求解其 k-错线性复杂度.因此,本文提出了一种混合的遗传算法来近似计算任意周期序列的 k-错线性复杂度.采用轮盘赌、最优保留策略、两点交叉和单点随机变异,并引入自适应算子来调整交叉概率和变异概率,以保证遗传算法的收敛性.通过并行计算适应度函数来提高算法的效率,同时与模拟退火算法相结合,加速算法收敛并避免早熟.结果表明:当 $k<8$ 且周期小于 256 时,k-错线性复杂度的实验值仅比精确值高 8%.

关键词: 密码学; 周期序列; 线性复杂度; k-错线性复杂度; 遗传算法
中图分类号: TN 918.1 **文献标志码**: A

A Hybrid Genetic Algorithm for Computing the k-Error Linear Complexity of Periodic Sequences

NIU Zhihua, *YUAN Can*, *KONG Deyu*
(School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China)

Abstract: The linear complexity of periodic sequences and its stability are important metrics for the evaluation in stream cipher. The k-error linear complexity is an important evaluation index for the stability of linear complexity. However, at present, it is difficult to compute the k-error linear complexity of the period sequences (except for 2^n、p^n、$2p^n$). Therefore, a hybrid genetic algorithm is proposed to approximate the k-error linear complexity of arbitrary periodic sequences by adopting the roulette wheel and elitist reserved strategy, the two-point crossover and simple random mutation, and by introducing adaptive operators to adjust the crossover and mutation probabilities to ensure the convergence of the genetic algorithm. The efficiency of the algorithm is improved by using the parallel computing fitness function. Simultaneously, by combining with the simulated annealing algorithm, it increases the convergence speed and avoids the premature convergence. The results show that the experiment value of k-error linear complexity is only 8% higher than the exact value when $k<8$ and the period is less than 256.

Key words: cryptography; periodic sequences; linear complexity; k-error linear complexity; genetic algorithm

---

收稿日期:2018-07-07
基金项目:国家重点研发计划项目(2016YFB1000600,2016YFB1000601,2016YFB10006011),国家自然科学基金重点项目(61936001),上海市自然科学基金项目(16ZR1411200,17ZR1409800,19ZR1417700),科技部广东省部省联动项目(2018B010113001),国家自然科学基金项目(61572309)
作者简介:牛志华(1976-),女,山西省晋中市人,副教授,主要研究方向为序列密码.
电话(Tel.):021-66135387;E-mail:zhniu@shu.edu.cn.

周期序列 s 的 k-错线性复杂度定义为:当改变 s 的1个周期中至多 k 位之后,得到的所有序列的线性复杂度中的最小值.一个良好的序列需具有较大的线性复杂度和 k-错线性复杂度.具有低线性复杂度的序列在密码学上是弱的,因为通过使用 Berlekamp-Massey (BM)算法[1],获得任何两倍于其线性复杂度的连续比特,就可以有效地恢复整个序列.同样,具有低的 k-错线性复杂度的序列也较弱,因为如果找到相应的线性递归关系,该序列就容易受到攻击.

针对序列的线性复杂度及其稳定性,即序列线性复杂度和 k-错线性复杂度指标,国内外学者做了大量研究.对于 2^n-周期的序列, Games 等[2]提出了计算序列线性复杂度的快速算法,其时间复杂度远低于通用的 BM 算法,文献[3-4]基于 Games-Chan 算法,研究了序列 k-错线性复杂度的分布,而文献[5]针对固定不变的 k-错线性复杂度,研究了周期序列的分布.对于 p^n-周期的序列,文献[6-7]研究了序列的 k-错线性复杂度谱,文献[8]提出了计算错误序列的算法.对于 $2p^n$-周期的序列,文献[9]提出了计算序列 k-错线性复杂度的算法.更多关于线性复杂度和 k-错线性复杂度的研究成果见文献[10-15].目前,有很多算法可以计算周期为 2^n、p^n、$2p^n$ 的序列的 k-错线性复杂度,但没有算法可以计算其他任意周期序列的 k-错线性复杂度.因此,设计一个能计算任意周期序列的 k-错线性复杂度的通用算法非常关键.

k-错线性复杂度的本质是求线性复杂度最小值的优化问题,而遗传算法很适合求解优化问题. Alecu 等[16]首次将遗传算法应用到二元序列复杂度的初步探索中,利用遗传算法来近似计算周期为 2^n 的二元序列的 k-错线性复杂度,能够计算周期为32的二元序列的5-错线性复杂度,且实验结果比准确值平均高19.5%.

本文设计了一种混合遗传算法来计算二元有限域 GF(2)上给定序列的 k-错线性复杂度的近似值.采用二进制编码、轮盘赌选择和最优保留策略、两点交叉、单点随机变异、自适应调整交叉和变异概率,并行计算每代种群个体的适应度,以提高算法效率,并对每代的最佳个体进行模拟退火避免收敛于局部最优解.使用本文改进的遗传算法,计算周期为256的二元序列的8-错线性复杂度,实验值仅比准确值高8%.因此,在 k 值较小且可以接受微小误差的情况下,即可使用本文的混合遗传算法计算任意周期序列的 k-错线性复杂度.

1 可行性分析

本节首先介绍线性复杂度和 k-错线性复杂度,然后分析使用遗传算法计算序列的 k-错线性复杂度的可行性.

定义1 设 $s=\{s_0,s_1,\cdots\}$ 为 q 元有限域 GF(q) 上的序列,若对任意的 i,有 $s_i=s_{i+N}$,则 s 是周期为 N 的序列.如果 GF(q) 上的周期序列 s 满足 $s_j+c_1s_{j-1}+\cdots+c_Ls_{j-L}=0$ ($j\geqslant L$),其中 L 是正整数,$c_1,c_2,\cdots,c_L$ 在 GF(q) 中,则称 s 是一个 L 阶线性递归序列,满足上式的最小的正整数 L 称为该递归序列的线性复杂度,记为 LC(s).

周期序列的线性复杂度是序列密码的重要评价指标.然而,线性复杂度很高并不一定能保证序列的安全,即当改变这些序列周期的少数几位时,其线性复杂度大幅下降.因此,学者们提出了线性复杂度稳定性的评价指标,即 k-错线性复杂度.

定义2 周期为 N 的序列 s 的 k-错线性复杂度为,当改变 s 的一个周期中至多 k($0\leqslant k<N$) 位之后,得到的所有序列的线性复杂度中的最小值,记为 $\mathrm{LC}_k(s)$,即

$$\mathrm{LC}_k(s)=\min\{\mathrm{LC}(s+e)\mid w_{\mathrm{H}}(e)\leqslant k\} \quad (1)$$

式中:e 为周期为 N 的序列,常被称为错误序列、错误类型;$w_{\mathrm{H}}(e)$ 为序列 e 的1个周期中不为0的元素个数.显然,至少存在1个错误序列 e,使得 $\mathrm{LC}(s+e)=\mathrm{LC}_k(s)$,称使得 $(s+e)$ 的线性复杂度等于 k-错线性复杂度的错误序列 e 为 k-错线性复杂度对应的错误序列.

序列的 k-错线性复杂度,即所有的错误序列 e 加上原序列 s 的线性复杂度 LC($s+e$) 的最小值是1个典型的求最小值的单目标优化问题.而求解优化问题是遗传算法的1个非常重要且擅长的领域,问题的关键为确定"优"的标准.设计相应的适应度函数,使序列 $(s+e)$ 的线性复杂度越小时,适应度越高,序列越优秀.对于一般序列,可以用 BM 算法计算序列 $(s+e)$ 的线性复杂度;对于特殊周期的序列,如果有计算线性复杂度的算法,可以用其定义适应度函数来判断 $(s+e)$ 的适应性.因此,采用遗传算法近似计算序列的 k-错线性复杂度在理论上是可行的.

2 标准遗传算法计算 k-错线性复杂度时出现的问题

遗传算法是基于达尔文的进化论和孟德尔的群体遗传学说,模拟自然界的生物进化过程的一种随

机搜索和全局优化算法[16]. 它针对一个种群，按照适者生存的原理，按照适应度大小挑选个体，并进行交叉和变异操作，产生新一代种群，新种群比前代种群更适应环境，如此反复进行直到满足优化准则或者达到最大遗传代数，该过程得到的最优结果即为问题的近似最优解. 本节介绍使用的标准遗传算法和其中出现的问题.

算法 1　标准遗传算法(GA)

(1) 初始化：序列 s 的第一个周期 $s^N=\{s_0,s_1,\cdots,s_{N-1}\}$，$k$，PS，MAXGEN，POP(0)，GEN=0；

(2) 计算 POP(0)中每个个体的适应度；

(3) 判断 GEN >MAXGEN 是否成立，若成立则转(9)；

(4) 使用轮盘赌从 POP(GEN)中选择出 POP(GEN+1)；

(5) 在 POP(GEN+1)中按照交叉概率 P_c 执行单点交叉；

(6) 在 POP(GEN+1)中按照变异概率 P_m 执行单点随机变异；

(7) 计算 POP(GEN+1)中每个个体的适应度；

(8) GEN=GEN+1，转(3)；

(9) 输出序列 s 的近似 k-错线性复杂度 $LC_k(s)$，结束.

其中：PS 为种群大小；MAXGEN 为最大生成代数；POP(i)为第 i 代种群；GEN 为变量；P_c 为[0,1]区间的一个值，表示交叉概率；P_m 为[0,1]区间的一个值，表示变异概率. 在上述算法中，PS = 400，MAXGEN = 500，$P_c=0.6$，$P_m=0.03$.

然而，使用算法 1 来计算 k-错线性复杂度时，存在以下几个问题：

(1) 针对不同的优化问题，需要反复实验来确定 P_c 和 P_m，并且很难找到适应于每个问题的最佳值；

(2) 伴随着 N 的增加，编码长度也在不断增加，导致算法效率急剧下降；

(3) 实验表明，标准遗传算法容易快速收敛于局部最优解.

3　基于遗传算法的计算任意周期二元序列的 k-错线性复杂度的改进算法

针对标准遗传算法中出现的如何确定 P_c 和 P_m 的最佳值问题，本文采用自适应方法来调整 P_c 和 P_m 的值，在保证群体多样性的同时，保证遗传算法的收敛性. 对于算法效率的问题，采用并行计算来提高算法的性能，使 N、k 增加时，能够减少计算时间，提高效率. 将遗传算法和模拟退火算法相结合，提高算法的全局和局部搜索能力，加速收敛并避免早熟，解决标准遗传算法常获得局部最优解的问题.

3.1　编码

遗传算法通常采用二进制编码，浮点数编码和符号编码. 本文研究的输入序列 s 在 GF(2) 上，对染色体使用二进制编码较为方便. 二进制编码将问题的可能解映射成有限个二进制符号组成的符号串，即定义 1 个染色体为任何可能的错误模式：$e\in GF(2)^n$，$w_H(e)\leqslant k$. 最佳的染色体是在改变原始序列的最多 k 位之后，引起线性复杂度下降最大的染色体.

3.2　适应度函数

遗传算法在进化过程中以适应度函数作为依据进行优胜劣汰进行选择，基本不利用外部信息，因此适应度函数的选择直接影响到遗传算法的收敛速度和能否找到最优解.

根据 k-错线性复杂度的定义，序列的 k-错线性复杂度为其改变至多 k 位能得到线性复杂度的最小值. 因此，将目标序列 s 与错误序列 e 相加得到改变后的序列($s+e$)，计算其线性复杂度，该值越小，说明 e 的适应度越高. 定义适应度函数为

$$f(e)=C_{\max}-LC(s+e) \tag{2}$$

式中：$C_{\max}$ 是序列 s 可能出现的最大线性复杂度的值，用序列 s 的周期 N 代替. 本节采用 BM 算法(算法 2)计算序列($s+e$)的线性复杂度.

BM 算法可用于计算任意域上的周期序列的线性复杂度和特征多项式[1]. 根据 BM 算法，如果 1 个序列的线性复杂度为 LC(s)，那么只需知道该序列的任意长度为 2LC(s)的连续比特即可恢复出全部的序列.

算法 2　BM 算法

(1) 初始化：$s^N=\{s_0,s_1,\cdots,s_{N-1}\}$，$P(X)=1$，$P'(X)=1$，LC = 0，$m=-1$，$d'=1$，$t=0$；

(2) 判断 $t>N-1$ 是否成立，若成立，则转(9)；

(3) $d=s_t+\sum_{i=1}^{LC}p_is_{t-i}$；

(4) 判断 $d\neq 0$ 是否成立，若成立则转(6)；

(5) $t=t+1$，转(2)；

(6) $T(X)=P(X)$，$P(X)=P(X)-d(d')^{-1}P'(X)X^{t-m}$；

(7) 判断 $2LC>t$ 是否成立，若成立则转(5)；

(8) $LC=t+1-LC$，$m=t$，$P'(X)=T(X)$，$d'=d$，转(5)；

(9) 输出 LC 和 $P(X)$,结束.

其中:s^N 为序列 s 的一个周期;LC 为 s 的线性复杂度;$P(X)$ 为特征多项式;其他的变量均为辅助的中间变量.

3.3 选择、交叉、变异算子

选择算子模拟自然界中优胜劣汰、适者生存的自然法则,其目的在于为下一代保留适应度高的优秀基因,使得种群进化朝着最优解的方向进行,加速收敛速度,同时也要保证种群的多样性.所以我们使用轮盘赌和最优保存策略相结合的选择算子.

交叉算子模拟了自然界生物体的基因重组,体现了信息交换的思想.通过两个染色体的交叉组合,来产生新的优良个体,从而搜索空间中新的点.本文使用两点交叉,因为在单点交叉中,染色体末端的良好基因总是被交换,不利于保留优良基因.

若只有选择和交叉算子,遗传算法就无法在初始基因组合以外的空间进行搜索,进化过程很容易陷入局部最优解而导致进化过程终止,不能得到全局最优解.而变异算子是模拟生物在自然遗传环境中由各种偶然因素引起基因突变的过程,以一个很小的变异概率随机地改变遗传基因(染色体的符号串的某一位)的值.本文使用基本位变异算子,提高种群的多样性,防止早熟.

3.4 自适应算子

标准遗传算法中,交叉概率 P_c 和变异概率 P_m 在进化过程中固定不变,而它们是影响遗传算法性能的关键,直接影响算法的收敛性.如果 P_c 太大,种群虽然更容易产生新个体,但是优良个体在种群中保留率也会降低;如果 P_c 太小,搜索效率会下降甚至停滞不前.如果 P_m 太大,遗传算法成为纯粹的随机搜索算法;如果 P_m 太小,则不易产生新的个体结构.

为了提高遗传算法的收敛性,达到全局最优解,采用 Srinivas 等提出的自适应遗传算法[17],使 P_c 和 P_m 的值随着适应度的变化而自动改变.当种群的整体适应度趋于一致或者趋于局部最优时,增大 P_c 和 P_m;当种群的整体适应度比较分散时,降低 P_c 和 P_m.同时,对于适应度高于平均值的个体,降低 P_c 和 P_m,保证其进入下一代;对于适应度低于平均值的个体,提高 P_c 和 P_m,以将其淘汰.此操作使得自适应的 P_c 和 P_m 能够提供相对于某个解的最佳 P_c 和 P_m.因此,自适应遗传算法在保持种群多样性的同时,也保证了遗传算法的收敛性.

P_c 和 P_m 按照如下公式进行计算:

$$P_c = \begin{cases} P_{c1} - \dfrac{(P_{c1}-P_{c2})(f'-f_{avg})}{f_{max}-f_{avg}}, & f' \geqslant f_{avg} \\ P_{c1}, & f' < f_{avg} \end{cases} \tag{3}$$

$$P_m = \begin{cases} P_{m1} - \dfrac{(P_{m1}-P_{m2})(f_{max}-f)}{f_{max}-f_{avg}}, & f \geqslant f_{avg} \\ P_{m1}, & f < f_{avg} \end{cases} \tag{4}$$

式中:$P_{c1}=0.6$, $P_{c2}=0.3$, $P_{m1}=0.1$, $P_{m2}=0.01$;f 为需要变异个体的适应度;f' 为两个需要交叉操作的个体中适应度较大的个体适应度;f_{avg} 为种群的平均适应度;f_{max} 为种群中的最大适应度.

3.5 并行算子

初步实验说明,当 N 较大时,标准遗传算法的效率会下降.由于算法的运行时间主要在于个体适应度的计算,而遗传算法的特性在于个体之间相互独立,个体的适应度可并行计算,划分的不同种群之间也可以相互独立的进行演化,所以使用并行遗传算法来加速搜索过程,以减少计算时间.

使用 OpenMP 实现并行遗传算法[18].程序开始执行时,仅为 1 个单线程程序,当需要并行计算适应度函数时,会形成 1 个线程组,多个线程并行执行计算适应度函数的代码.最后,到所有线程都执行完并行代码后,回归主线程继续执行至结束.此操作不仅降低了并行程序设计的难度,还提高了算法的收敛速度.

3.6 模拟退火算子

标准遗传算法通常容易产生早熟现象、局部搜索能力较差等问题,而模拟退火算法具有较强的局部搜索能力,并能使搜索过程避免陷入局部最优解[19].所以在遗传算法中引入模拟退火算子,提高算法的全局和局部搜索能力,加速收敛并避免早熟.

模拟退火(SA)是一种基于 Monte Carlo 迭代法的启发式随机搜索算法.SA 来源于对固体物质的退火降温过程中的热平衡问题的模拟和随机搜索优化问题,以找到全局最优解或近似全局最优解[20].在搜索最优解的过程中,SA 除了可以接受最优解外,还有 1 个随机接受准则(Metropolis 准则),可以有限度地接受恶化解,并且接受恶化解的概率慢慢趋向于 0,使得算法有可能从局部最优中跳出,找到全局最优解,保证算法的收敛.

将遗传算法和模拟退火算法相结合,对每一代最优个体执行模拟退火操作.步骤如下:

(1) 初始化遗传算法,初始化退火温度 $T_0 = \mathrm{LC}(s)$.

(2) 对种群进行交叉、变异等操作，选出本代最优个体.

(3) 在温度 T_K 下执行操作：产生新的可行解 e'(e' 是 e 相应的目标函数值)；计算新解的评价函数 $f(e')$ 和旧解的评价函数 $f(e)$ 的差值：$\Delta f = f(e') - f(e)$；依照概率 $\min\{1, \exp(-\Delta f/T_K)\} > \text{random}[0,1]$ 接收新解，random[0,1] 是 1 个 [0,1] 区间内的随机数；按一定的方式，缓慢降低温度，定义降温函数为 $T_{K+1} = \alpha T_K$，$K = K+1$(α 为降温系数，为 1 个小于 1 的正数).

(4) 执行最优保留算子，若满足收敛条件，则结束；否则转(2).

3.7　混合遗传算法

使用轮盘赌和最优保留、两点交叉代替单点交叉、单点随机变异、自适应交叉和变异概率，对每代的最佳个体进行模拟退火以保证不陷入局部最优解，并行编程以提高效率.

算法 3　混合遗传算法(IGA)

(1) 初始化：$s^N = \{s_0, s_1, s_2, \cdots, s_{N-1}\}$，$k$，PS，MAXGEN，POP(0)，GEN=0；

(2) 根据式(2)并行计算 POP(0)中每个个体的适应度；

(3) 判断 GEN>MAXGEN 是否成立，若成立则转(11)；

(4) 使用轮盘赌从 POP(GEN)中选择出 POP(GEN+1)；

(5) 在 POP(GEN+1)中按照式(3)得到的自适应交叉概率 P_c 执行两点交叉；

(6) 在 POP(GEN+1)中按照式(4)得到的自适应变异概率 P_m 执行单点随机变异；

(7) 根据式(2)并行计算 POP(GEN+1)中每个个体的适应度；

(8) 最优选择；

(9) 对本代最优个体执行模拟退火算子；

(10) GEN=GEN+1，转(3)；

(11) 输出 $LC_k(s)$，结束.

4　实验与分析

本节首先使用混合遗传算法(算法 3)与标准遗传算法(算法 1)对周期为 2^n 的序列进行数值实验，并对两个算法进行重复实验，对比两个算法的结果，说明混合遗传算法比标准遗传算法在误差率和算法效率方面更优秀. 然后展示 2^n 周期序列的 k-错线性复杂度实验值和准确值的对比图，说明混合遗传算法的误差率较低，并挑选几个任意周期序列的 k-错线性复杂度实验图说明本文设计的混合遗传算法可以计算非特殊周期序列的 k-错线性复杂度.

实验设定 PS=200，MAXGEN=200 以及 PS=400，MAXGEN=500，使用混合遗传算法 IGA(算法 3)，针对周期 2^n 的不同取值进行重复实验，分别计算序列的 6-错线性复杂度，统计算法的准确率和运行时间，结果如表 1 所示. 表中：A 为实验值比准确值高出的百分比，即误差率；T 为算法的运行时间.

表 1　PS = 200，MAXGEN = 200 **和** PS = 400，MAXGEN = 500 **时，混合遗传算法的结果**

Tab. 1　**Results of IGA at** PS = 200，MAXGEN = 200 **and** PS = 400，MAXGEN = 500

N	k	PS	MAXGEN	A_{IGA}/%	T_{IGA}/s
64	6	200	200	0.68	1
64	6	400	500	0.52	6
128	6	200	200	0.66	3
128	6	400	500	0.57	16
256	6	200	200	0.55	10
256	6	400	500	0.12	50

对于不同周期序列的 6-错线性复杂度，当 PS=400，MAXGEN=500 时，尽管运行时间长，但误差率均比 PS=200，MAXGEN=200 时低. 在尽量保证算法准确度的情况下，下文实验选取 PS=400，MAXGEN=500 的组合参数，针对周期 2^n 和 k 的不同取值进行重复实验，每组实验随机生成 400 个序列，统计算法的准确率和运行时间，结果如表 2 和表 3 所示. 其中 IGA 为混合遗传算法(算法 3)的结果，GA 表示标准遗传算法(算法 1)的结果.

表 2　**混合遗传算法计算** $k = 6, 7, 8$ **的结果**

Tab. 2　**Results of IGA at** $k = 6, 7, 8$

N	k	PS	MAXGEN	A_{IGA}/%	T_{IGA}/s
64	6	400	500	0.52	6
64	7	400	500	6.13	6
64	8	400	500	7.88	6
128	6	400	500	0.57	16
128	7	400	500	1.94	18
128	8	400	500	2.26	16
256	6	400	500	0.12	50
256	7	400	500	0.33	52
256	8	400	500	1.48	50

表 3 标准遗传算法计算 $k=6,7,8$ 的结果

Tab. 3 Results of GA at $k=6,7,8$

N	k	PS	MAXGEN	P_c	P_m	$A_{GA}/\%$	T_{GA}/s
64	6	400	500	0.6	0.03	1.72	24
64	7	400	500	0.6	0.03	7.21	27
64	8	400	500	0.6	0.03	10.44	26
128	6	400	500	0.6	0.03	1.36	66
128	7	400	500	0.6	0.03	3.42	67
128	8	400	500	0.6	0.03	5.17	67
256	6	400	500	0.6	0.03	0.82	205
256	7	400	500	0.6	0.03	1.68	210
256	8	400	500	0.6	0.03	2.82	209

标准遗传算法中，P_c 和 P_m 设置为固定值，本文通过多次实验对比，发现 $P_c=0.6$，$P_m=0.03$ 时，实验结果更准确，故后文的实验使用此参数组合.

在 IGA 实验中，采用 4 个线程并行计算的运行时间均比采用 GA 所需的时间更短，效率更高. 在程序运行中，使用 BM 算法计算每个个体的适应度占据了大部分时间，而每个个体之间是相互独立的，在计算适应度时能够并行执行，极大地减少了运行时间. 在硬件环境支持的情况下，线程数越多，运行时间越短，效率越高. 当 PS = 200，MAXGEN = 200 时，计算 k-错线性复杂度，64 位的序列通常只需要 1 s，128 位的序列需要 3 s，256 位的序列需要 10 s. 随着种群规模和迭代代数的增加，PS=400，MAXGEN=500，虽然运行时间在增加，但算法的误差率在不断减小.

当 $k<8$ 时，IGA 有较小的误差率，$N=64$ 时，误差率小于 8%；$N=128$ 时，误差率小于 3%；$N=256$ 时，误差率小于 2%. 当 $k>8$ 时，误差率变大，但经常使用传统算法研究的多是 1-错，2-错线性复杂度，而对于 8-错及更大的 k-错线性复杂度，其研究难度较大，同时当 k 值很大时，k-错线性复杂度的研究意义不大. 实验证明，k 值较小时，混合遗传算法的效率更高，误差率更低，实验结果更加接近准确值.

下面分别展示不同的 2^n 周期序列的 k-错线性复杂度的实验值和准确值的对比图. 图 1～3 分别为 $n=6,7,8$ 时，即 $N=64,128,256$ 时，3 个序列 $s1$、$s2$、$s3$ 的 k-错线性复杂度情况. 可以看出，当 $k<8$ 时，本文混合遗传算法的误差率较低，实验值与准确值很接近，当 k 较大时，算法慢慢收敛，实验值与准确值的误差变大.

$s1=\{01100011110011010101001111011100110011101111000101100011111100000\}$，$N=64$

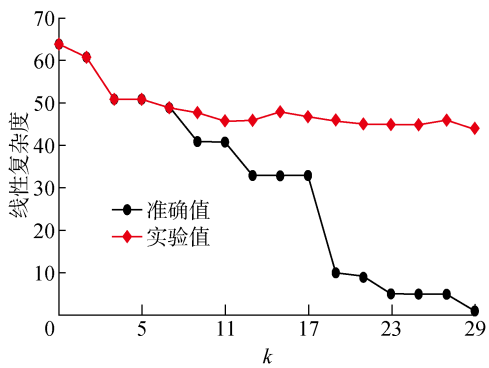

图 1 序列 $s1$ 的 k-错线性复杂度实验值和准确值对比

Fig. 1 Comparison of experimental and accurate values of k-error linear complexity of sequence $s1$

$s2=\{11100100101111000110110000001101011010111111011011011111101110011010100101110010101110001100010001010000001110001101000000101001 1\}$，$N=128$

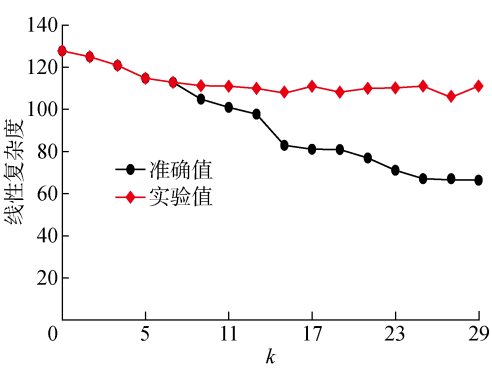

图 2 序列 $s2$ 的 k-错线性复杂度实验值和准确值对比

Fig. 2 Comparison of experimental and accurate values of k-error linear complexity of sequence $s2$

$s3$ = {0101101101101000001001110101011011001011010000011101011110010000010110110111011011101011100111001101010101001001011110110111010010100010110000000011000000111111010001111110011000011100110101110101011110011000100100110010100111111000010000110001011111000011010}, $N=256$

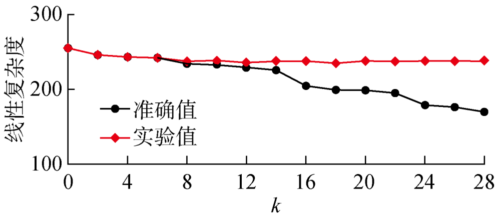

图 3　序列 $s3$ 的 k-错线性复杂度实验值和准确值对比

Fig. 3　Comparison of experimental and accurate values of k-error linear complexity of sequence $s3$

对比文献[16]的实验结果，周期为 32 的二元序列的 5-错线性复杂度的实验结果比准确值平均高 19.5%. 而本文算法可以计算周期为 256 的二元序列的 8-错线性复杂度，其实验值仅比准确值高 8%. 因此，本文算法不仅使可计算的 N、k 增加，还提高了算法的准确性和效率.

前文的实验之所以采用 $N=64, 128, 256$，是因为我们可以计算这些周期序列的准确的 k-错线性复杂度，从而可以对比实验结果，计算误差率，说明算法准确性的提高. 而我们的算法可以计算任意周期的二元序列的 k-错线性复杂度，所以下面给出两条任意周期序列的 k-错线性复杂度的实验图. 图 4 和 5 分别为 $N=200$ 的序列 $s4$ 和周期 $N=500$ 的序列 $s5$ 的 k-错线性复杂度的实验图.

$s4$ = {01000111110011000010001011011101100000111001101011110100000010110001100111100011001010111001111101101101111001110001010001110110110001101110101111011100101110100001010101001100010100011111001000001110}, $N=200$

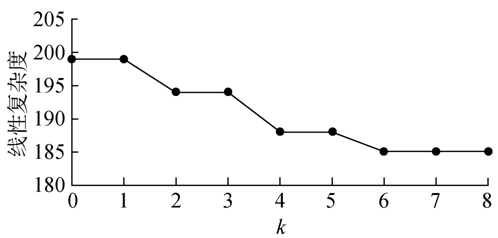

图 4　序列 $s4$ 的 k-错线性复杂度实验值

Fig. 4　Experimental values of k-error linear complexity of sequence $s4$

$s5$ = {1010101100100001111011001010000000001010110001011001110001010001111001010111000101010110000000101110100000000111110011001101100011001000000001100111111000000000101000100101110111001010101010111111011000000011100010000110110100100100000110010111000101000010000110110110110111110101111000000101101001101000001001011111000000000000011010110110011001101101001111101000100010001011111110110010000101010111000111101001101110000011011000101111101001000010101110001000000011111110010001000101101011100000111010010101100}, $N=500$

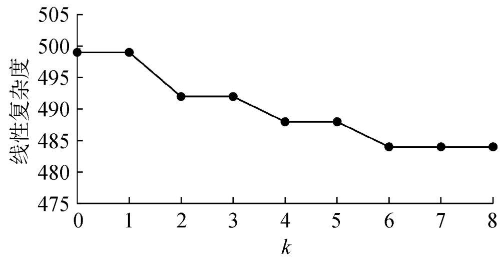

图 5　序列 $s5$ 的 k-错线性复杂度实验值

Fig. 5　Experimental values of k-error linear complexity of sequence $s5$

从图 4 和 5 可以看出，非 2^n 周期序列的 k-错线性复杂度实验值变化规律和 2^n 周期序列的 k-错线性复杂度实验值变化规律类似. 即在 k 值较小且可以接受微小误差的情况下，可以计算任意周期序列的 k-错线性复杂度.

5　结语

使用遗传算法计算在有限域上给定周期序列的 k-错线性复杂度时，针对标准遗传算法所存在的问题，对其进行改进，设计了一种混合的遗传算法. 为了保证遗传算法的收敛性，加速寻优并避免陷入局部最优解，使用轮盘赌和精英选择、自适应调整交叉和变异概率，进行两点交叉和单点随机变异操作，并行计算每代个体的适应度，并对每代的最优个体进行模拟退火操作. 结果表明，当 k 较小时，混合遗传算法的误差率小于 8%，同时提高了遗传算法的寻优性能和运行效率. 因此，在 k 值较小且可以接受微小误差的情况下，可以使用本文设计的混合遗传算法来计算任意周期序列的 k-错线性复杂度.

参考文献：

[1]　MASSEY J. Shift-register synthesis and BCH decoding[J]. **IEEE Transaction on Information Theory**,

1969, 15(1): 122-127.

[2] GAMES R, CHAN A. A fast algorithm for determining the complexity of a binary sequence with period 2^n [J]. **IEEE Transaction on Information Theory**, 1983, 29(1): 144-146.

[3] KE P H, CHANG Z L. On the error linear complexity spectrum of binary sequences with period of power of two[J]. **Chinese Journal of Electronics**, 2015, 24(2): 366-372.

[4] ZHOU J, LIU W. The k-error linear complexity distribution for 2^n-periodic binary sequences[J]. **Designs Codes and Cryptography**, 2014, 73(1): 55-75.

[5] PAN W, BAO Z, LIN D, *et al*. The distribution of 2^n-periodic binary sequences with fixed k-error linear complexity[C]// **International Conference on Information Security Practice and Experience**. Zhangjiajie, China: Springer, 2016: 13-36.

[6] TANG M, ZHU S. On the error linear complexity spectrum of p^n-periodic binary sequences[J]. **Applicable Algebra in Engineering Communication and Computing**, 2013, 24(6): 497-505.

[7] LI F L, ZHU S, HU H, *et al*. Determining the k-error joint linear complexity spectrum for a binary multisequence with period p^n [J]. **Cryptography and Communications**, 2016, 8(4): 513-523.

[8] TANG M. An algorithm for computing the error sequence of p^n-periodic binary sequences[J]. **Applicable Algebra in Engineering, Communication and Computing**, 2014, 25(3): 197-212.

[9] ZHOU J. On the k-error linear complexity of sequences with period $2p^n$ over GF (q) [J]. **Designs Codes and Cryptography**, 2011, 58(3): 279-296.

[10] YU F W, SU M, WANG G, *et al*. Error decomposition algorithm for approximating the k-error linear complexity of periodic sequences[C]// **Trustcom/BigDataSE/ISPA**. Tianjin, China: IEEE, 2016: 505-510.

[11] NIU Z, CHEN Z, DU X. Linear complexity problems of level sequences of Euler quotients and their related binary sequences[J]. **Science China Information Sciences**, 2016, 59(3): 1-12.

[12] LIU L F, YANG X Y, DU X N, *et al*. On the linear complexity of new generalized cyclotomic binary sequences of order two and period pqr[J]. **Tsinghua Science and Technology**, 2016, 21(3): 295-301.

[13] ZHAO C, MA W, YAN T, *et al*. Linear complexity of least significant bit of polynomial quotients[J]. **Chinese Journal of Electronics**, 2017, 26(3): 573-578.

[14] CHEN Z, NIU Z, WU C. On the k-error linear complexity of binary sequences derived from polynomial quotients [J]. **Science China Information Sciences**, 2015, 58(9): 1-15.

[15] LIU L, YANG X, DU X, *et al*. On the k-error linear complexity of generalised cyclotomic sequences [J]. **International Journal of High Performance Computing and Networking**, 2016, 9(5/6): 394-400.

[16] ALECU A, SALAGEAN A. A genetic algorithm for computing the k-error linear complexity of cryptographic sequences [C]// **IEEE Congress on Evolutionary Computation**. Singapore: IEEE, 2007: 3569-3576.

[17] SRINIVAS M, PATNAIK L. Adaptive probabilities of crossover and mutation in genetic algorithms[J]. **IEEE Transactions on Systems Man and Cybernetics**, 1994, 24(4): 656-667.

[18] HERDA M. Parallel genetic algorithm for capacitated p-median problem [J]. **Procedia Engineering**, 2017, 192: 313-317.

[19] LAN S, LIN W. Genetic algorithm optimization research based on simulated annealing [C]// **International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing**. Shanghai, China: IEEE, 2016: 491-494.

[20] PENG Y, LUO X, WEI W. A new fuzzy adaptive simulated annealing genetic algorithm and its convergence analysis and convergence rate estimation[J]. **International Journal of Control Automation and Systems**, 2014, 12(3): 670-679.

（本文编辑：陈晓燕）