

【电子与信息科学 / Electronics and Information Science】

极化码自适应信道译码算法

叶茂林, 谭晓青, 许丽卿, 吕善翔

暨南大学信息科学技术学院 / 网络空间安全学院, 广东广州 510632

摘要: 极化码是一种理论上可以达到香农极限的信道编码方法, 且有编译码复杂度低的优点, 是5G通讯中信道编码解决方案之一。串行抵消列表(successive cancellation list, SCL)译码是极化码目前最常用的译码方法, 但该方法的空间和时间复杂度都很高; 快速串行抵消(fast successive cancellation, Fast-SC)译码可有效降低译码复杂度, 但存在可靠性不高的问题。为兼顾极化码译码效率和可靠性, 提出一种自适应信道的预快速SCL(preFast-SCL)译码算法。该算法通过联合Fast-SC和SCL译码算法, 在译码开始时先使用Fast-SC算法快速得到一组译码结果, 并对其进行校验, 校验通过则作为结果输出, 不通过则再使用SCL译码保证可靠性。仿真结果表明, 随着信道条件变好, preFast-SCL译码算法复杂度逐步降低。在加性高斯白噪声信道下, preFast-SCL与SCL译码算法可靠性基本相同, 在信噪比为2.0 dB时, preFast-SCL的译码复杂度比SCL降低了45%, 且在更高的信噪比条件下, 时间性能增益更佳。

关键词: 通信传输技术; 极化码; 快速译码; 自适应译码; 联合译码; 时间性能增益

中图分类号: TN911.2 **文献标志码:** A **doi:** 10.3724/SP.J.1249.2022.05521

Adaptive channel decoding method for polar codes

YE Maolin, TAN Xiaoqing, XU Liqing, and LÜ Shanxiang

College of Information Science and Technology / College of Cyber Security, Jinan University,
Guangzhou 510632, Guangdong Province, P. R. China

Abstract: The polar code is a channel coding technology which can reach the Shannon limit in theory and has the advantage of low coding and decoding complexity. It is now one of the channel coding solutions in 5G communication. The successive cancellation list (SCL) decoding algorithm is the most commonly used decoding method for polar codes, but it has high memory and time complexity. Fast successive cancellation (Fast-SC) decoding algorithm can effectively reduce the decoding complexity, but it has the problem of low reliability. In order to take into account the decoding efficiency and reliability of polar codes, a preFast-SCL decoding algorithm for adaptive channels is proposed. The algorithm combines the advantages of Fast-SC and SCL decoding algorithms. At the beginning of decoding, the Fast-SC algorithm is used to quickly obtain a group of decoding results and verify them. If the verification is passed, it will be outputted as the result, but if not, SCL decoding will be used to ensure the reliability. Simulation results show that the complexity of preFast-SCL decoding algorithm decreases gradually with the improvement of channel conditions. In additive white Gaussian noise channel, the reliability of preFast-SCL is basically the same as that of SCL decoding algorithm. When the signal to noise ratio (SNR) is 2.0 dB, the decoding complexity of preFast-SCL is reduced by 45% compared with SCL, and the time performance gain is better under the condition of higher SNR.

Key words: communication transmission technology; polar codes; Fast-SC decoding; adaptive decoding; joint decoding; time performance gain

Received: 2021-06-15; **Accepted:** 2021-07-16; **Online (CNKI):** 2022-08-08

Foundation: National Natural Science Foundation of China (62032009); Basic and Applied basic research in Guangdong Province (2019B030302008)

Corresponding author: Professor TAN Xiaoqing. E-mail: ttanxq@jnu.edu.cn

Citation: YE Maolin, TAN Xiaoqing, XU Liqing, et al. Adaptive channel decoding method for polar codes [J]. Journal of Shenzhen University Science and Engineering, 2022, 39(5): 521-528. (in Chinese)



在二进制无记忆对称信道中,使用极化码的串行抵消(successive cancellation, SC)译码可令信道容量理论上达到香农极限^[1],因此极化码成为目前 5G 的编码标准之一,且在未来的多种无线通信应用场景下有巨大潜力^[2-3].然而,在编码长度有限情况下,SC 译码的性能低于低密度奇偶校验(low density parity check, LDPC)码^[4],需研究更可靠译码方法.

ARIKAN^[5]提出置信传播(belief propagation, BP)译码方法,利用置换因子图方式进行译码,迭代计算多次后,该方法性能高于 SC 译码方法. TAL 等^[6]提出连续抵消列表法译码,通过在译码时保留多条路径提高了算法的可靠性. NIU 等^[7]基于串行抵消列表(successive cancellation list, SCL)提出循环冗余校验辅助(cyclic redundancy check-aided SCL, CA-SCL)译码算法,利用循环冗余校验(cyclic redundancy check, CRC)检查 SCL 算法产生的译码分支,准确率比 BP 译码方法高^[8],但 SCL 算法复杂度高且译码复杂度不会随信道条件变化而变化,每次译码都会计算大量分支的值,显著的增加了译码过程的时间步数^[9],但在信道条件较好的时候,SC 算法有较好的性能,不必额外开分支使用 SCL 算法.在效率优化方面, SARKIS 等^[10]根据冻结位数量和分布特点将译码树分割出 4 种特殊节点类型,提出快速串行抵消(Fast-SC)译码算法. ANIF 等^[11]将 Fast-SC 中的特殊节点进行了更详细的分类,并给出了确定特殊节点方法,进一步降低了译码时延. CAVATASSI 等^[12]将 Fast-SC 译码算法拓展到多核情况,增加了 Fast-SC 译码的灵活度.然而, Fast-SC 译码搜索特殊节点返回中间码字的迭代过程均与 SC 译码相同,存在可靠性能偏低的问题.为兼顾极化码译码时间效率和准确率上的性能, LI 等^[13]提出自适应 SCL(adaptive-SCL)算法,通过动态选取保留路径数目,提高了译码效率,其实质为 SC 联合 SCL 算法译码.本研究基于联合译码的思想,提出 preFast-SCL 译码方法,进一步提高译码效率.该方法首先使用 Fast-SC 译码算法,快速得到一组译码结果,对得到的译码结果进行校验,通过校验则作为结果输出,不通过校验则继续迭代使用 SCL 译码算法保证译码算法的可靠性.仿真结果表明,新算法的正确率与 SCL 算法基本相同,在中高信噪比条件下可有效降低译码复杂度.

1 极化码译码算法

为便于描述极化码常见和本研究提出的译码方法,表 1 给出了部分参数说明.

表 1 参数说明

Table 1 Parameters description

参数	含义
$L_N^{(i)}(y_1^N, \hat{u}_1^{i-1})$	译码节点 i 的对数似然比
$L_N^{(i)}[l]$	第 l 分支的 $L_N^{(i)}(y_1^N, \hat{u}_1^{i-1})$
r_i	信道 i 接收到的信号
x_1^i	等价于集合 $(x_1, x_2, \dots, x_i)$
$\hat{u}_1^i$	信道序号 1 到 i 的译码结果
$\hat{u}_1^i[l]$	第 l 个分支中的 $\hat{u}_1^i$
β	特殊节点的译码结果, $\beta = (\beta_1, \beta_2, \dots, \beta_N)$

1.1 SC 译码算法

极化码的 SC 译码使用对数似然比(logarithm likelihood ratio, LLR)进行译码运算.对于长度 $N = 2^n (n \in \mathbf{N}^*)$ 的极化码,设 $\mathbf{u} = (u_1, u_2, \dots, u_N)$ 为发送信号. SC 译码算法先使用接收到的信号序列的对数似然比 $\mathbf{y}_1^N$ 计算 u_1 对应的估计值 $\hat{u}_1$,随后利用 $(\mathbf{y}_1^N, \hat{u}_1)$ 估计 $\hat{u}_2$,再用 $(\mathbf{y}_1^N, \hat{u}_1^2)$ 估计 $\hat{u}_3$,以此类推并最终得到估计的发送信号 $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2, \dots, \hat{u}_N)$.其中, $\hat{u}_1^i$ 为信道序号 1 到 i 的译码结果.信道序号值为 i 的节点的估计值为

$$\hat{u}_i = \begin{cases} h_i(L_N^{(i)}(y_1^N, \hat{u}_1^{i-1})), & i \in A \\ u_i, & i \in A^c \end{cases} \quad (1)$$

其中, $L_N^{(i)}(y_1^N, \hat{u}_1^{i-1})$ 为节点 i 的对数似然比,定义为

$$L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) \triangleq \ln \left(\frac{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|0)}{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|1)} \right) \quad (2)$$

这里, $W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|0)$ 和 $W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|1)$ 分别是已知 y_1^N 和 $\hat{u}_1^{i-1}$ 时 $\hat{u}_i$ 为 0 和 1 的条件概率. A 和 A^c 为信息信道编号集合和冻结信道编号集合,由收发端事先约定.当 $i \in A^c$ 时,译为双方约定的比特值,一般为 0;当 $i \in A$ 时,则通过判决函数

$$h_i(L_N^{(i)}(y_1^N, \hat{u}_1^{i-1})) = \begin{cases} 0, & L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) \geq 0 \\ 1, & L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) < 0 \end{cases} \quad (3)$$

得到节点 i 的译码结果 $\hat{u}_i$.

SC 译码算法结构如图 1. 其中,信道 i 接收到

的信号 r_i 转换为 LLR 值的转换公式为

$$y_i = \frac{2r_i}{\sigma} \quad (4)$$

这里, σ 为噪声方差. 图1中虚线和实现箭头分别为式(5)和式(6)的 f 操作和 g 操作; 每个节点代表1个 LLR 值, 最右侧节点的 LLR 值与接收端信道的 LLR 值相等, 既 $L_1^{(i)} = y_i$, 其余节点的 LLR 值可通过递归公式计算得到, 再使用式(1)估计对应节点的值, 则最左侧节点组成的序列即译码结果.

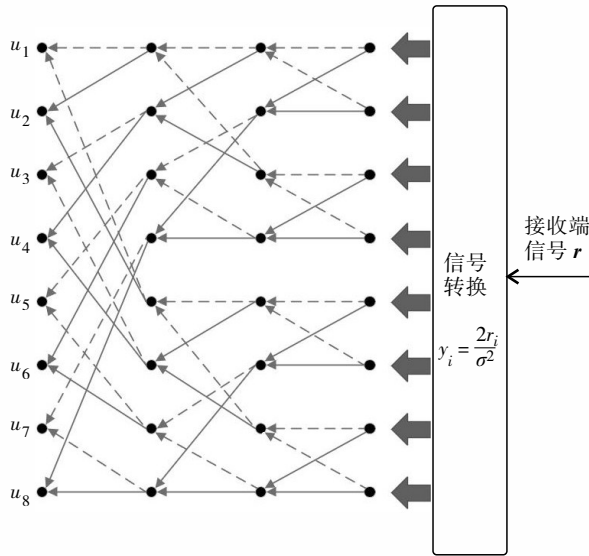


图1 SC译码算法结构

Fig. 1 Structure of SC decoding algorithm.

每个计算基本单元如图2所示. 基本单元运算可分为两部分:

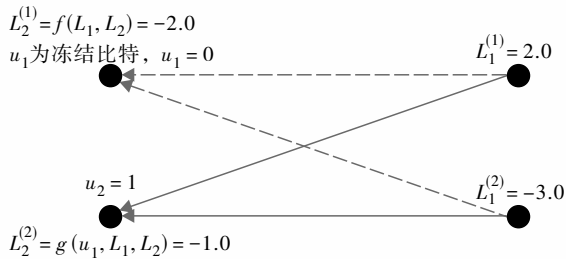


图2 SC译码基本单元

Fig. 2 Basic unit of SC decoding.

1) f 操作. 从右往左计算左上节点的 LLR 值, 并使用式(1)估计该节点的比特值为 u_1 .

$$f(L_1^{(1)}, L_1^{(2)}) = \ln \left(\frac{1 + \exp(L_1^{(1)} + L_1^{(2)})}{\exp(L_1^{(1)}) + \exp(L_1^{(2)})} \right) \quad (5)$$

其中, L_1 和 L_2 为图2中右边2个节点的 LLR 值.

2) g 操作. 采用式(6)计算左下节点的 LLR 值, 再使用式(1)估计该节点的比特值.

$$g(L_1^{(1)}, L_1^{(2)}, u_1) = (1 - 2u_1)L_1^{(1)} + L_1^{(2)} \quad (6)$$

为减少仿真计算的复杂度, 将式(2)简化^[14]为

$$f(a, b) = \text{sign}(a)\text{sign}(b)\min(|a|, |b|) \quad (7)$$

SC 译码算法迭代过程的伪代码见图3. 其中, $u_{1,e}^i$ 和 $u_{1,o}^i$ 分别为 u_1^i 中的偶数位和奇数位; 符号 $\oplus$ 表示模2加法; 冻结比特约定译码为0.

```

01  initialization
02   $\mathbf{u} = [0, 0, \dots, 0]$ 
03  for  $i = 1 : N$  do:
04       $a = L_{N/2}^{(i)}(y_1^{N/2}, u_{1,e}^{2i-2} \oplus u_{1,e}^{2i-2})$ 
05       $b = L_{N/2}^{(i)}(y_{1+N/2}^N, u_{1,e}^{2i-1})$ 
06       $L_N^{(2i-1)}(y_1^N, u_{1,e}^{2i-2}) = f(a, b)$ 
07       $L_N^{(2i)}(y_1^N, u_{1,e}^{2i-1}) = g(a, b, u_i)$ 
08       $u[i] = h_i(L_N^{(i)}) = \begin{cases} 0, & L_N^{(i)} \geq 0, & i \in A \\ 1, & L_N^{(i)} < 0, & i \in A \\ 0, & i \in A^c, \end{cases}$ 
09  end for
10  output  $\mathbf{u}$ 

```

图3 SC译码算法的伪代码

Fig. 3 Pseudocode of SC decoding algorithm.

1.2 SCL译码算法

SCL 译码在 SC 译码的基础上保留了 L 条具有最小路径度量(path metric, PM)值的路径, 增加了译码的可靠性. 对于长度为 $N = 2^n$ ($n \in \mathbf{N}^*$) 的极化码, SCL 译码中第 l 条 ($l = 1, 2, \dots, L$) 路径上的第 i 位 ($i = 1, 2, \dots, N$) 的路径度量为

$$\text{PM}_l^{(i)} = \sum_{j=1}^i \ln(1 + \exp(2\hat{u}_j[l]L_N^{(j)}[l] - 1)) \quad (8)$$

其中, $\hat{u}_j[l]$ 为第 l 个分支中第 j 个节点的译码比特值. 由式(1)可得路径 l 的 LLR 为

$$L_N^{(i)}[l] = \ln \left(\frac{P_N^{(i)}(y_1^N, u_{1,e}^{i-1}[l]|0)}{P_N^{(i)}(y_1^N, u_{1,e}^{i-1}[l]|1)} \right) \quad (9)$$

PM 值越小, 路径越可靠. 每个节点译码最多保留 L 条路径, 拓展路径条数超过 L 后将保留 L 条具有最小 PM 值的路径, 其余路径将被剪枝删除. 因此, 迭代完成后会得到 L 条分支, 再按照 PM 值从小到大排序, 取第1条分支作为结果并输出.

CA-SCL 是信道传输中差错检验的常见技术, 通过加入冗余校验位, 可更充分利用 SCL 的译码路径. CA-SCL 译码算法将译码路径按 PM 值从小到大排序, 再依次进行校验, 并取第1个通过路径作为译码结果输出, 若都不通过则译码失败. CA-SCL

算法的准确性比SCL算法有明显提高^[7].

2 自适应信道译码算法

传统的SCL译码虽然准确性高,但缺乏对信道条件的判断和利用,且在时间复杂度上仍有较大优化空间.本研究通过优化Fast-SC算法和探究保留路径 L 对SCL算法的影响,选择适当的保留路径数,译码时通过1次预快速译码检测信道情况,提高极化码译码效率.

2.1 Fast-SC译码算法及优化

Fast-SC算法将译码树分成若干种特殊节点,然后根据节点的长度,迭代计算出相应层数的对数似然比值.图4展示了4种特殊节点的结构,使用该结构时其准确性与SC译码算法基本相同^[15].特殊节点包含但不限于以下4种结构:

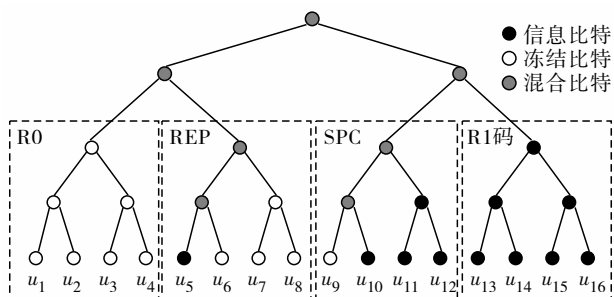


图4 SC快速译码特殊节点

Fig. 4 Special nodes of Fast-SC decoding.

1) 码率0(rate 0, R0), 所有信源比特全是冻结比特, 所有比特译码为0, 即 $\beta = (0, 0, \dots, 0)$.

2) 重复(repetition, Rep)码, 除最后一位外其他位全为冻结位. 记 $S = \sum_{i=1}^N y_i$, 若 $S \geq 0$, 则 $\beta = (0, 0, \dots, 0)$; 否则, $\beta = (1, 1, \dots, 1)$.

3) 单偶校验(single parity check, SPC)码, 除了 u_1 是冻结比特, 其余都是信息比特. 硬判决为

$$\beta_i = \frac{1 - \text{sign}\left(\ln\left(\frac{P(r_i|0)}{P(r_i|1)}\right)\right)}{2} = \frac{1 - \text{sign}(y_i)}{2} \quad (9)$$

当SPC译码结果 $\beta = (\beta_1, \beta_2, \dots, \beta_N)$ 的异或和为1, 即 $\bigoplus_{i=1}^N \beta_i = 1$ 时, 使用比特翻转思想^[16], 取 $p = \arg\left(\min_{1 \leq i \leq N} |y_i|\right)$, 进行 $\beta_p = \beta_p \oplus 1$ 操作, 得到的 β 为译码结果.

4) 码率1(rate 1, R1), 所有的信源比特都为

信息比特, 直接使用式(8)进行硬判决.

这4种节点都可由LLR序列 $(y_1, y_2, \dots, y_N)$ 直接估计极化码码字序列 $\beta = (\beta_1, \beta_2, \dots, \beta_N)$, 无需使用SC译码迭代至叶节点. 但是, R0节点的译码没有使用接收序列, 在优化后的Fast-SC译码算法中, 需先辨认特殊节点是否为R0节点, 若是, 则返回全0比特; 否则, 计算下一层LLR时再使用对应的译码规则.

表2对比了当 $N = 1024$ 时, SC、Fast-SC和优化Fast-SC译码算法中 f 函数执行次数和对应的误帧率(frame error rate, FER). 由表1可见, 优化Fast-SC译码比Fast-SC算法在效率上约有7%的提升.

表2 不同译码算法 f 函数执行效率和误帧率($N = 1024$)

Table 2 Comparison of f function execution efficiency of different algorithms ($N = 1024$)

码率	f 函数执行次数			FER
	SC算法	Fast-SC算法	优化Fast-SC算法	
0.250	5 120	2 652	2 460	2.7×10^{-3}
0.375	5 120	2 776	2 590	4.2×10^{-3}
0.500	5 120	2 744	2 576	1.1×10^{-2}
0.625	5 120	2 750	2 516	7.9×10^{-2}
0.750	5 120	2 748	2 512	8.6×10^{-1}

2.2 选择CA-SCL译码算法合适的分支数

若CA-SCL算法保留的路径数 L 太大, 会产生很大时延, 但若 L 太小, 则可选的译码器很少, 且算法误帧率高、可靠性低. 图5给出了CA-SCL算法中保留路径数对误帧率的影响. 其中, 图注中

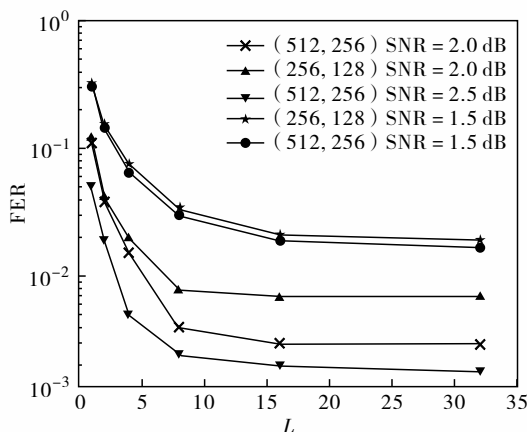


图5 CA-SCL译码算法保留路径数与误帧率的关系

Fig. 5 The relationship between reservation path and frame error rate of CA-SCL decoding algorithm.

括号内数值分别为码长和信息值; 信噪比(signal to noise ratio, SNR)分别为 1.5、2.0、2.5 dB. 由图 5 可见, 随着 L 的增大, 算法译码的准确性上升, 特别是在 $L = [2, 16]$ 时, FER 下降最快. 当 $L \geq 8$ 时, FER 的变化趋于平缓. 因此, 本研究取 $L = 8$

作为保留路径数.

2.3 preFast-SCL 译码算法

自适应信道的 preFast-SCL 译码算法流程和主要模块如图 6.

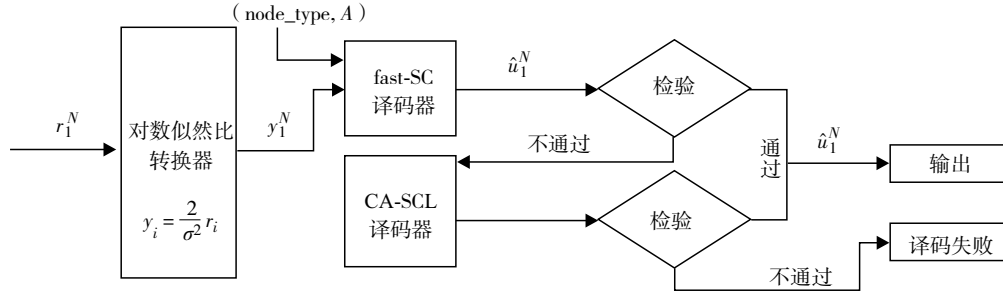


图6 preFast-SCL 译码流程图

Fig. 6 Flow chart of preFast-SCL.

得到对数似然比序列后, 配合信息位集合 A 和使用的特殊节点类型 $node_type$ 进行 Fast-SC 译码中, 得到译码序列 $\hat{u}_1^N$, 并对该结果进行 CRC 校验. 若通过校验, 则作为最终结果输出; 否则, 使用 CA-SCL 译码器进行译码, 提高译码结果可靠性. 当信道条件较好时候, Fast-SC 有较高通过率, 可大幅提升译码效率. 当 Fast-SC 译码结果出现差错, 校验不通过时, 仍有 CA-SCL 译码器兜底, 有效提升了译码结果的可靠性. preFast-SCL 译码算法的伪代码见图 7. 其中, $CRC(\hat{u}_1^N)$ 为 $\hat{u}_1^N$ 进行 CRC 校验, $\hat{u}_{1,L}^N$ 存储了所有 L 条路径的译码结果; $\hat{u}_{1,L}^N[l]$ 为第 l 条路径的译码结果.

下面将讨论的最大和平均时间复杂度, 以及空间复杂度的推导.

1) preFast-SCL 最大复杂度是 $O(LN \lg N)$

Fast-SC 算法的时间复杂度为 $O(N \lg N)$; CA-SCL 译码时间复杂度为 $O(N \lg N)$; CRC 校验时间复杂度 $O(m)$. 其中, L 为 CA-SCL 算法的保留路径数; m 为 CRC 校验位长度. 最差的情况是执行快速译码后, 检验失败, 而在执行 CA-SCL 译码算法时候, 检测到最后一条路径才结束进行输出, 此时有最大时间复杂度为

$$C_{\max} = O(L+1)m + O(N \lg N) + O(NL(n \lg n)) = O(LN \lg N) \quad (11)$$

2) 平均时间复杂度是 $O(1 + P_e(\gamma)L)N \lg N$

$P_e(\gamma)$ 是 SC 算法在信噪比 $E_b/N_0 = \gamma$ 时的误帧率. 其中, E_b 为平均比特符号能量; N_0 为噪声功率频谱. 算法总会执行 1 次 Fast-SC 译码算法, 但仅

```

01 initialization
02 nodeStruc = getNodeStruc(A, node_type)
03  $\hat{u}_1^N \leftarrow \text{Fast-SC}(\text{nodeStruc}, y_1^N)$ 
04 if  $CRC(\hat{u}_1^N) = \text{failure}$  then:
05    $(\hat{u}_{1,L}^N, PM_L) \leftarrow \text{CA-SCL}(y_1^N, A)$ 
06   Index  $\leftarrow \text{index of smallest } |PM_L|$ 
07   for  $i = 1:L$  do:
08      $l \leftarrow \text{Index}[i]$ 
09     if  $CRC(\hat{u}_1^N) = \text{success}$  then:
10        $\hat{u}_1^N \leftarrow \hat{u}_{1,L}^N[l]$ ; break;
11     end if
12   end for
13 end if
14 output  $\hat{u}_1^N$ 

```

图7 preFast-SCL 译码算法伪代码

Fig. 7 PreFast-SCL decoding algorithm pseudo code.

在不通过 CRC 校验情况下才执行 CA-SCL 算法, 因此, 平均时间复杂度为

$$\bar{C} = O((1 + P_e(\gamma)L)N \lg N) \quad (12)$$

其中, $P_e(\gamma)$ 为极化码误帧率满足 $P_e \leq O(2^{-N^\beta})$, β 为极化率^[17], 即码长越长, preFast-SCL 的译码时间复杂度比起 SCL 算法, 下降得明显.

3) preFast-SCL 算法的空间复杂度为 $O(LN)$

Fast-SC 译码算法与 SC 算法的空间复杂度相同, 只需 $O(N)$ 空间, 而 CA-SCL 算法使用懒复制(lazy copy)策略^[6], 只需 $O(LN)$ 空间. preFast-SCL 空间

复杂度为

$$S = O(\max(LN, N)) = O(LN) \quad (13)$$

3 系统仿真分析

3.1 参数设置

为探究 preFast-SCL 算法译码性能, 本研究在加性高斯噪声信道下进行仿真, 仿真参数设置如下: CA-SCL、adaptive-SCL^[13]和 preFast-SCL 算法使用的 CRC 生成多项式均为 $g(x) = x^{16} + x^{15} + x^2 + 1$, 采用二进制相移键控 (binary phase shift keying, BPSK) 调制方式, 极化码构造采用改进高斯估计法^[18], 码率为 0.5, BP 译码最大迭代数为 50 次。

3.2 仿真结果与分析

图 8 对比了 preFast-SCL、adaptive-SCL^[13]和 CA-SCL 译码算法在 N 分别为 512 和 1 024, 信息位分别为 256 和 512 时的性能。其中, 图注中括号内容表

示信息比特序列长度和码长, 如 (512, 1 024) 表示信息比特序列长度为 512, 码长为 1 024。

由图 8(a) 可见, 信道适应算法 preFast-SCL 和 adaptive-SCL 都不会造成 FER 性能损失, 可靠性与 CA-SCL 算法基本相等。由图 8(b) 可得, 在 $E_b/N_0 = 0 \sim 1.5$ dB 时, preFast-SCL 和 adaptive-SCL 算法的复杂度比 CA-SCL 算法更高, 但在 $E_b/N_0 > 1.5$ dB 时, preFast-SCL 和 adaptive-SCL 算法的时间复杂度比 CA-SCL 算法都有大幅降。其中, preFast-SCL 算法的时间性能优于 adaptive-SCL 算法。

图 9 对比了 preFast-SCL 译码算法在不同码长和信噪比条件下的 FER 和时间效率性能。由图 9 可见, 在低信噪比条件下, preFast-SCL 译码算法的长码译码速率低于短码译码速率。码长为 N 的码字, 单位比特译码时间的复杂度为 $O(\lg N)$, 即码字越长, 译码速率越低。但另一方面, 当相同信噪比

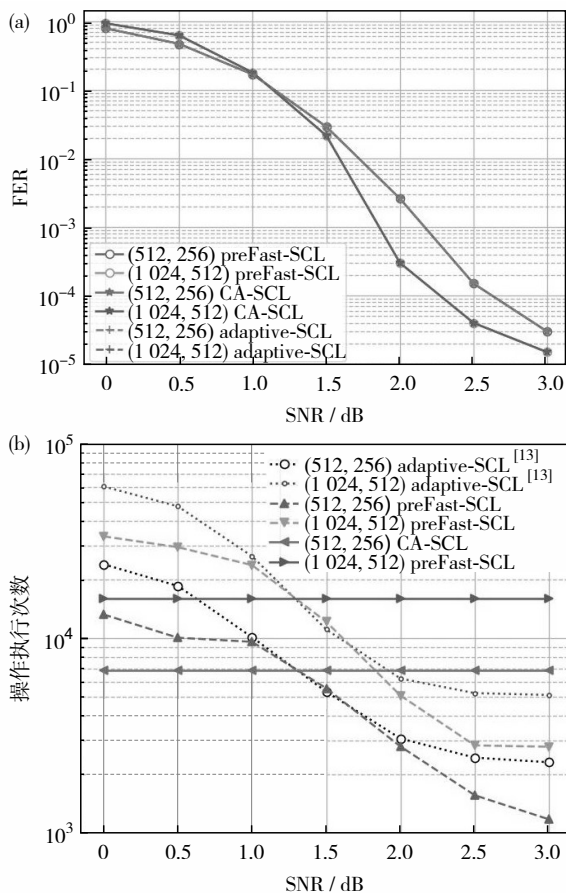


图 8 信道适应译码算法与 CA-SCL 性能对比

(a) 可靠性; (b) 耗时

Fig. 8 Performance comparison between channel adaptive decoding algorithm and CA-SCL for (a) reliability and (b) time consuming.

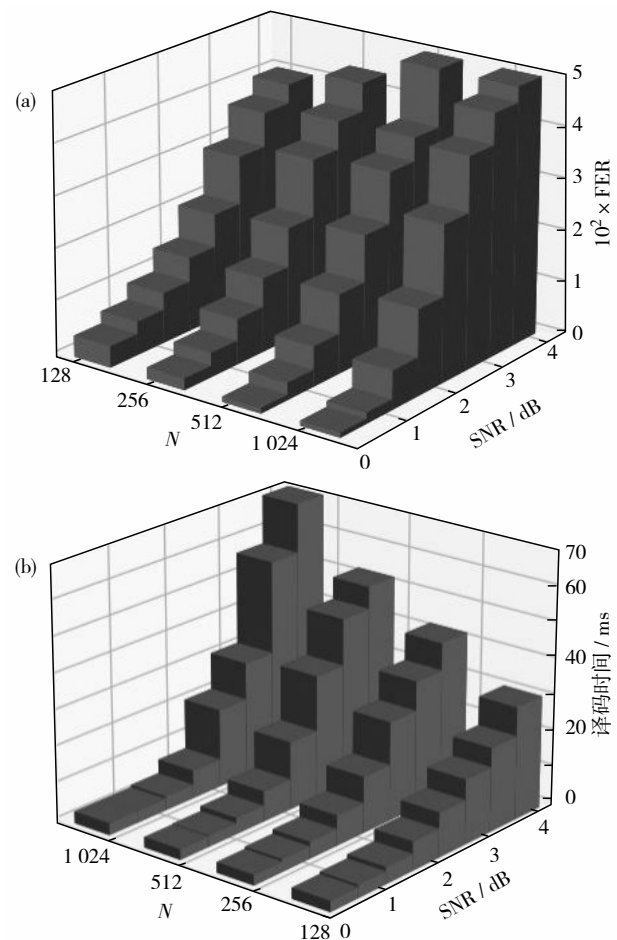


图 9 信噪比与码长对 preFast-SCL 性能影响

(a) 可靠性; (b) 译码耗时

Fig. 9 Influence of SNR and code length on the performance of preFast-SCL for (a) reliability and (b) time consuming.

时, 码长增长, 则码长较长的极化码的 $P_e(\gamma)$ 值会降低. 由式(12)可得, preFast-SCL译码算法的平均复杂度会下降, 随着信噪比提升, 长码译码的复杂度会有很大改善.

图10为preFast-SCL算法、BP算法和SC算法的在码长 N 分别为256、512和1 024时的可靠性仿真对比. 由图10可见, preFast-SCL算法比传统的SC和BP算法, 可靠性能上有较大提升.

图11给出了在码长 N 分别为512和1 024时, preFast-SCL、SC、CA-SCL和Fast-SC算法的复杂度比较. $N = 512$, $E_b/N_0 = 2.0$ dB时, preFast-SCL算法需执行的操作数约为CA-SCL算法的45%, 而在 $N = 1 024$, $E_b/N_0 = 2.0$ dB时, preFast-SCL算法需执行的操作数仅是CA-SCL算法的30%, 且该值随

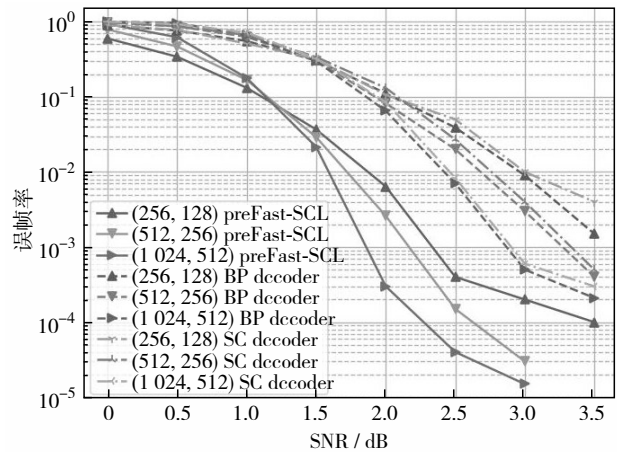


图10 PreFast-SC与传统算法可靠性对比

Fig. 10 Reliability comparison between preFast SC and traditional algorithms.

着信噪比和码长的增加仍会继续降低.

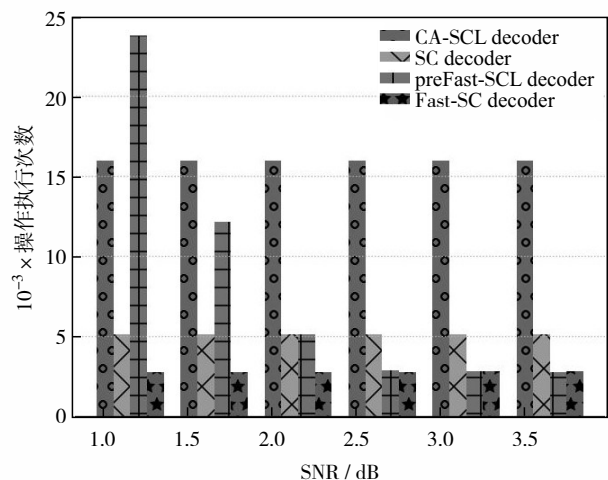
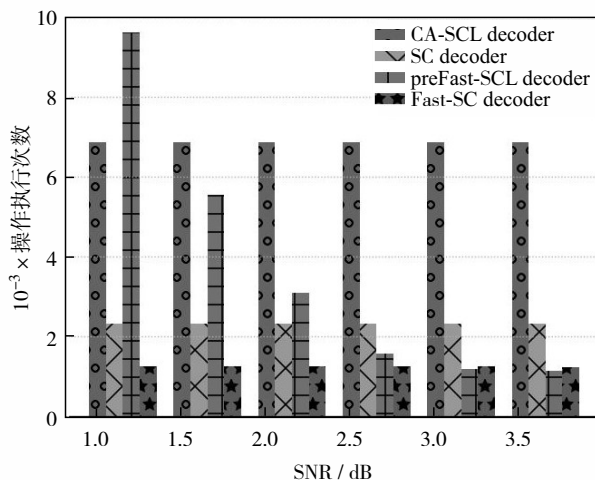


图11 PreFast-SCL与传统算法时间性能对比 (a) $N = 512$; (b) $N = 1 024$

Fig. 11 Time performance comparison between preFast-SCL and traditional algorithms for (a) $N = 512$ and (b) $N = 1 024$.

结 语

提出通过CRC的辅助对传统的Fast-SC译码算法进行了进一步优化, 并联合Fast-SC和CA-SCL译码算法, 提出preFast-SCL极化码译码方法. 仿真结果表明, 新算法保持了相对较低的误帧率, 而且可以在信道环境较好的时候, 有效的降低译码所需的时间.

在低信噪比($E_b/N_0 < 1.5$ dB)条件下, preFast-SCL算法在时间复杂度上的表现并不理想. 未来仍需将新算法跟其他同类算法做比较, 探寻提升preFast-SCL在低信噪比性能的新方法.

基金项目: 国家自然科学基金资助重点项目(62032009); 广东省基础与应用基础研究资助重大项目(2019B030302008)

作者简介: 叶茂林(1996—), 暨南大学硕士研究生. 研究方向: 极化码编译码及应用. E-mail: 673717994@qq.com

引 文: 叶茂林, 谭晓青, 许丽卿, 等. 极化码自适应信道译码算法[J]. 深圳大学学报理工版, 2022, 39(5): 521-528.

参考文献 / References:

- [1] ARIKAN E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels [J]. IEEE Transactions on Information Theory, 2009, 55(7): 3051-3073.
- [2] 周锋. 多系统GNSS非差非组合精密单点定位相关理论

<http://journal.szu.edu.cn>

- 论和方法研究[J]. 测绘学报, 2020, 49(7): 938.
- ZHOU Feng. Theory and methodology of multi-GNSS undifferenced and uncombined precise point positioning [J]. *Acta Geodaetica et Cartographica Sinica*, 2020, 49(7): 938. (in Chinese)
- [3] HONG Tao, TANG Tao, DONG Xinjie, et al. Future 5G mmwave TV service with fast list decoding of polar codes [J]. *IEEE Transactions on Broadcasting*, 2020, 66(2): 525-533.
- [4] TAHIR B, SCHWARZ S, RUPP M. BER comparison between convolutional, Turbo, LDPC, and polar codes [C]// *The 24th International Conference on Telecommunications (ICT)*. Limassol, Cyprus: IEEE, 2017: 1-7.
- [5] ARIKAN E. Polar codes: a pipelined implementation [C]// *Proceedings on the 4th International Symposium on Broad-band Communication*. Melaka, Malaysia [s. n.], 2010: 11-14.
- [6] TAL I, VARDY A. List decoding of polar codes [J]. *IEEE Transactions on Information Theory*, 2015, 61(5): 2213-2226.
- [7] NIU Kai, CHEN Kai. CRC-aided decoding of polar codes [J]. *IEEE Communications Letters*, 2012, 16(10): 1668-1671.
- [8] ELKELESH A, EBADA M, CAMMERER S, et al. Belief propagation list decoding of polar codes [J]. *IEEE Communications Letters*, 2018, 22(8): 1536-1539.
- [9] HASHEMI S A, CONDO C, GROSS W J. Fast simplified successive-cancellation list decoding of polar codes [C]// *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. San Francisco, USA: IEEE, 2017: 1-6.
- [10] SARKIS G, GROSS W J. Increasing the throughput of polar decoders [J]. *IEEE Communications Letters*, 2013, 17(4): 725-728.
- [11] HANIF M, ARDAKANI M. Fast successive-cancellation decoding of polar codes: Identification and decoding of new nodes [J]. *IEEE Communications Letters*, 2017, 21(11): 2360-2363.
- [12] CAVATASSI A, TONNELIER T, GROSS W J. Fast decoding of multi-kernel polar codes [C]// *IEEE Wireless Communications and Networking Conference*. Marrakesh, Morocco: IEEE, 2019: 1-6.
- [13] LI Bin, SHEN Hui, TSE D. An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check [J]. *IEEE Communications Letters*, 2012, 16(12): 2044-2047.
- [14] RYAN W E, LIN Shu. 信道编码: 经典与现代[M]. 白宝明, 马啸, 译. 北京: 电子工业出版社, 2017.
- RYAN W E, LIN Shu. *Channel codes: classical and modern* [M]. BAI Baoming, MA Xiao, trans. Beijing: Publishing House of Electronics Industry, 2017. (in Chinese)
- [15] 朱晓峰, 刘陕陕, 谭跃跃. 极化码编译码算法的研究其仿真[J]. *电子测量技术*, 2020(16): 82-86.
- ZHU Xiaofeng, LIU Shanshan, TAN Yueyue. Research on encoding and decoding algorithm of polarization code and its simulation [J]. *Electronic Measurement Technology*, 2020(16): 82-86. (in Chinese)
- [16] WANG Xiumin, WANG Ting, LI Jun, et al. Improved multiple bit-flipping fast-SSC decoding of polar codes [J]. *IEEE Access*, 2020, 8: 27851-27860.
- [17] BIOGLIO V, GABRY F, LAND I, et al. Multi-kernel polar codes: concept and design principles [J]. *IEEE Transactions on Communications*, 2020, 68(9): 5350-5362.
- [18] DAI Jincheng, NIU Kai, SI Zhongwei, et al. Evaluation and optimization of gaussian approximation for polar codes [J]. *Journal of the American Society for Information Science & Technology*, 2015, 51(1): 1-2.

【中文责编: 英子; 英文责编: 木柯】