Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 8, n. 1, 2021.

Trabalho apresentado no XL CNMAC, Evento Virtual - Co-organizado pela Universidade do Mato Grosso do Sul (UFMS).

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Análise da codificação de canal em comunicações ultra confiáveis para sistemas 5G

Leonardo Terças[1]
Centre for Wireless Communications (CWC), University of Oulu, Finland
Cintya W. O. Benedito[2]
Universidade Estadual Paulista (UNESP), Campus de São João da Boa Vista, SP
Carlos H. M. de Lima[3]
Centre for Wireless Communications (CWC), University of Oulu, Finland

Resumo. Visando sistemas de comunicação sem fio de quinta geração (5G), algumas técnicas de codificação de canal se tornaram candidatas para serem utilizadas em suas transmissões. O objetivo deste trabalho, é realizar a comparação de duas destas técnicas, os códigos convolucionais e os códigos polares, visando a aplicação em comunicações ultraconfiáveis para sistemas 5G.

Palavras-chave. Codificação de Canal, Códigos Convolucionais, Códigos Polares, Sistemas 5G.

1 Introdução

Codificação de canal é uma técnica onde bits de redundância são inseridos na mensagem de interesse, de modo que, durante a recepção da mesma, estes possam ser utilizados para detectar e corrigir erros. Esta técnica apresentada por Shannon em 1948, vem sendo alvo de diversas pesquisas, que buscam desenvolver e aprimorar bons códigos corretores de erros, para atingir a capacidade de canal prevista por Shannon, [7]. Assim, códigos com alto desempenho de codificação e decodificação são essenciais para os futuros sistemas de comunicação sem fio. Alguns códigos foram selecionados como candidatos para a comunicação sem fio de quinta geração, como os códigos convolucionais, os códigos turbo, os códigos LDPC (em inglês, *Low-density-parity-check*) e os códigos polares [9].

Neste trabalho iremos comparar o desempenho de técnicas de codificação de canal em cenários de comunicações ultra confiáveis (URC, em inglês *Ultra-Reliable Communication*), que é um modo de operação ainda não existente nos sistemas atuais, que prevê a sustentação de uma qualidade de serviço mínima durante quase 100% do período de conexão pois tais códigos apresentam grandes chances de compor a nova geração. Os códigos convolucionais e os códigos polares serão utilizados devido a baixa complexibilidade de implementação e o alto desempenho [8]. As simulações serão realizadas através de implementação computacional no software MATLAB, onde palavras de comprimento k serão codificadas, transmitidas por um canal AWGN (*Additive White Gaussian Noise*) e decodificadas por um receptor, a fim de avaliar a taxa de erro de bit (BER - *Bit Error Rate*), a latência de processamento, que é o tempo utilizado para codificar e decodificar uma mensagem de interesse, e a taxa de transferência de informação (*Throughput*), que é a quantidade de bits que são decodificados por segundo levando em consideração um ciclo de *clock* do processador [6].

[1]leonardo.tercas@oulu.fi
[2]cintya.benedito@unesp.br.
[3]carlos.lima@oulu.fi

2 Códigos Convolucionais e Códigos Polares

Um código convolucional utiliza memórias finitas no codificador para gerar uma redundância, [2]. Este código é definido através dos parâmetros (n, k, M), onde n é o número de bits produzidos na saída, k o número de bits inseridos na entrada e M a quantidade de memórias no codificador. Um código com estes parâmetros pode ser caracterizado por uma máquina de estados. Na Figura 1 podemos observar a máquina de estados de um código convolucional com parâmetros $(2, 1, 7)$, onde cada retângulo representa uma memória do codificador e este código possui apenas uma entrada e uma saída.

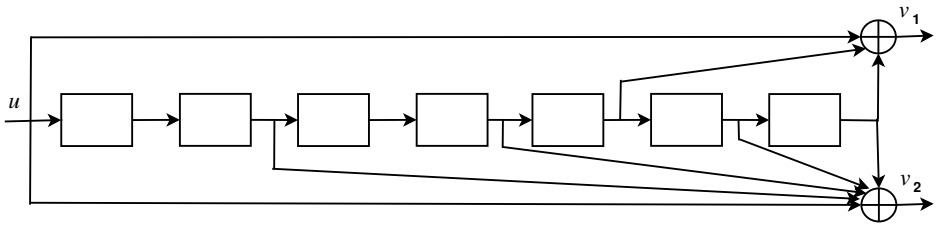


Figura 1: Máquina de estados código convolucional (2,1,7).

A codificação de um código convolucional pode ser realizada de várias formas, dentre elas o diagrama de treliça que permite observar, para cada instante t_n, todas as transições possíveis de estados. Cada uma das colunas deste diagrama representa todos os estados da máquina e suas conexões. Este diagrama pode ser utilizado tanto na codificação, quanto na decodificação. A decodificação mais utilizada para códigos convolucionais, é a baseada no algoritmo de Viterbi, que é uma técnica de decodificação por máxima verossimilhança (*Maximum Likelihood Decoding*).

Os códigos polares são códigos de bloco lineares que foram introduzidos por Arikan, [1]. Estes códigos são construídos a partir de uma estratégia conhecida como polarização de canal, onde cópias de uma canal discreto sem memória são combinados de modo que após divididos podem ser separados em canais bons e ruins, deste modo, bits de informação são enviados nos canais confiáveis (bons) e bits congelados nos canais ruidosos (ruins).

A partir da polarização de canal, obtém-se os canais por onde serão transmitidos os bits de informação e os bits congelados. Mas o congelamento dos bits pode ser realizado a partir de uma divisão do vetor de entrada u_1^N em duas partes: μ_A, que são os bits livres e μ_{A^C} que representa os bits congelados. Deste modo, temos que a codificação pode ser dada por:

$$x_1^N = u_A G_N(A) \bigoplus u_{A^C} G_N(A^c), \tag{1}$$

onde u é um vetor de comprimento $N = 2^n$, com $n > 0$, que inclui os bits de informação e os bits congelados. Para determinar a matriz geradora G_N são utilizadas algumas expressões algébricas, as quais são baseadas no produto de Kronecker da matriz F (ver [1]). Assim,

$$G_n = F^{\otimes n}, \tag{2}$$

onde $F^{\otimes n}$ é o n-ésimo produto de Kronecker da matriz F representada por:

$$F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{3}$$

Para a decodificação de códigos polares o algoritmo de Cancelamento Sucessivo (SC) é utilizado, [1]. Nessa técnica, o decodificador geralmente dispõe somente das informações sobre os valores e

DOI: 10.5540/03.2021.008.01.0360

posições dos bits congelados, u_{A^c} e A, respectivamente. Conforme é feita a decodificação do sinal, o vetor resultante $\hat{u}_1^N$ é referente aos vetores u_1^N. A razão de verossimilhança (*Log-Likelihood Ratio* - LLR) é dada por:

$$L_N^{(i)}(y_1^N, u_1^{i-1}) = \frac{W_N^i(y_1^N, u_1^{i-1}|u_i = 0)}{W_N^i(y_1^N, u_1^{i-1}|u_i = 1)}. \tag{4}$$

Considerando um decodificador de tamanho N, a equação anterior pode ser calculada a partir das relações recursivas f e g, calculados no domínio LLR, de entradas a e b, dadas por:

$$f(a,b) = \log\left(\frac{e^{a+b}+1}{e^a+e^b}\right) \tag{5}$$

e

$$g(a,b,s) = (-1)^s a + b, \tag{6}$$

onde s é a soma dos bits previamente decodificados que estão participando do nó g atual. A decodificação do sinal, ou seja, o vetor resultante $\hat{u}_1^N$ com os bits estimados, é dado por:

$$\hat{u}_1 = \begin{cases} u_i, & \text{se } i \notin A^c \\ 0, & \text{se } i \in A^c \text{ e } L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) \geq 1 \\ 1, & \text{se } i \in A^c \text{e } L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) < 1 \end{cases}. \tag{7}$$

Podemos utilizar também o decodificador por cancelamento sucessivo simplificado SSC. Este decodificador é uma melhoria do decodificador SC, pois sabendo a informação dos valores dos bits congelados, os nós que seriam necessários para determinar estes bits não são calculados. Deste modo, tem-se um ganho de processamento, pois recursos não são gastos desnecessariamente e há melhoria de latência, pois a mensagem de interesse é decodificada mais rapidamente, [5].

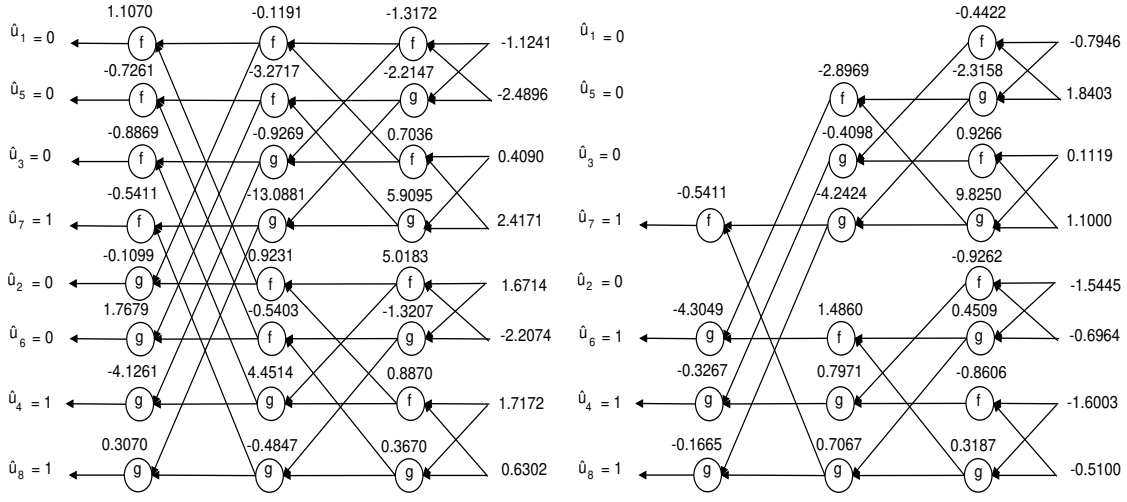
Exemplo 2.1. *Considere um código de tamanho $N = 8$ com taxa 1/2, transmitido por um canal AWGN com $E_b/N_0 = 1$ dB. As entradas (u_1, u_2, u_3, u_5) serão congeladas e serão transmitidos bits 0's. Supondo que na saída deste canal com ruído AWGN, foram recebidos os vetores $\boldsymbol{y}_1 = (-1.1241, 2.4896, 0.4090, 2.4171, 1.6714, -2.2074, 1.7172, 0.6302)$ e $\boldsymbol{y}_2 = (-0.7946, 1.8403, 0.1119, 1.1000, -1.5445, -0.6964, -1.6003, -0.5100)$. Aplicando as Equações 5 e 6, juntamente da relação (7), iremos realizar a decodificação do vetor $\boldsymbol{y}_1$ utlizando o decodificador SC e de $\boldsymbol{y}_2$ utilizando o decodificador SSC. Os valores encontrados em cada etapa da decodificação s nas Figuras 2(a) e 2(b).*

Uma análise importante do decodificador é determinar a quantidade de ciclos de *clocks* que o processador precisa para decodificar uma mensagem de interesse. Entende-se por *clock*, ou ciclo de *clock*, como sendo o número de ações que o processador consegue executar por segundo, como inicializar um programa, escrever algo na memória ou realizar alguns cálculos. Dessa forma, a decodificação deste código pode ser representada a partir de um gráfico de árvore, nesta representação cada bit estimado $\hat{\mathbf{u}}_i$ é representado por um nó. Temos dois tipos de nós $\mathcal{N}_0$ que corresponde aos bits congelados e $\mathcal{N}_1$ que correnspponde aos bits de informação. Em decodificadores SC, todos os nós são calculados, então podemos dizer que a quantidade de *clocks* que um decodificador utiliza para decodificar uma mensagem é dada por (ver [10]):

$$clock_{SC} = 2(N-1). \tag{8}$$

Já a quantidade de ciclos de *clock* necessários para decodificar um decodificador SSC é dada pela seguinte equação:

$$clock_{SSC} = 2(N-1) - \sum_i (2^{d_i+1} - 1) - \sum_j [(2^{d_j+1} - 1) - (d_j + 1)], \tag{9}$$



(a) Decodificação SC (b) Decodificação SSC

Figura 2: Exemplo Decodificadores

onde i e j, representam a quantidade de sub árvores formadas por nós $\mathcal{N}_0$ e $\mathcal{N}_1$, respectivamente, que estão contidas na árvore de decodificação, [10].

Uma outra medida importante de um código é o *throughput*, ou seja, a taxa de processamento de informação. Este parâmetro mede a quantidade de bits que são decodificados por segundo levando em consideração um ciclo de *clock* do processador. O *throughput* de um decodificador SC é dado pela seguinte equação:

$$T = \frac{k}{clock * t_p}, \tag{10}$$

onde T é dado em bits/s e t_p é o tempo necessário para que o processador codifique e decodifique uma mensagem de interesse, [5].

3 Análise dos Resultados

Realizamos a implementação dos codificadores e decodificadores dos códigos convolucionais e polares, com o objetivo de comparar o desempenho em cenários de comunicação ultraconfiável (URC) previstos nos sistemas 5G. Estes cenários possuem algumas métricas necessárias, como latência de 1 ms a 10 ms dependendo a aplicação e BER $\leqslant 10^{-6}$, [3, 4]. A latência prevista nestes cenários levam em consideração o tempo necessário para a transmissão fim a fim da mensagem, as simulações realizadas analisam somente o tempo de processamento que é o tempo em que as codificações tem influência, é válido ressaltar que este tempo de processamento depende das especificações da máquina onde as simulações foram realizadas. A máquina utilizada tem um processador AMD Ryzen 3 1200 Quad-Core Processor 3.10 GHz e 8 GB de RAM. Para simular a transmissão de pacotes curtos previsto neste tipo de cenário, adota-se o comprimento da palavra-código enviada de no máximo 512 bits, [8]. Assim, serão realizadas três análises: a comparação da BER, latência e *throughput* das duas codificações.

Para a simulação da BER, a mensagem $\hat{\mathbf{u}}$ que foi decodificada será comparada com a palavra-código $\mathbf{u}$ transmitida, para contabilizar, caso sejam diferentes, a ocorrência de erros de decodificação. Podemos observar na Figura 3 os resultados obtidos. Como previsto, comparando as

BER's encontradas, os códigos polares possuem um desempenho melhor que os códigos convolucionais, que melhora conforme o comprimento da palavra transmitida aumenta, atingindo a métrica de BER $\leqslant 10^{-6}$ com uma potência de transmissão cada vez menor. Já os códigos convolucionais possuem um desempenho similar, independente do comprimento da palavra e para os parâmetros utilizados não atingem a métrica esperada. Deste modo, na comparação da BER, a utilização dos códigos convolucionais não seriam interessantes para a transmissão de pacotes curtos em sistemas 5G no canal simulado.

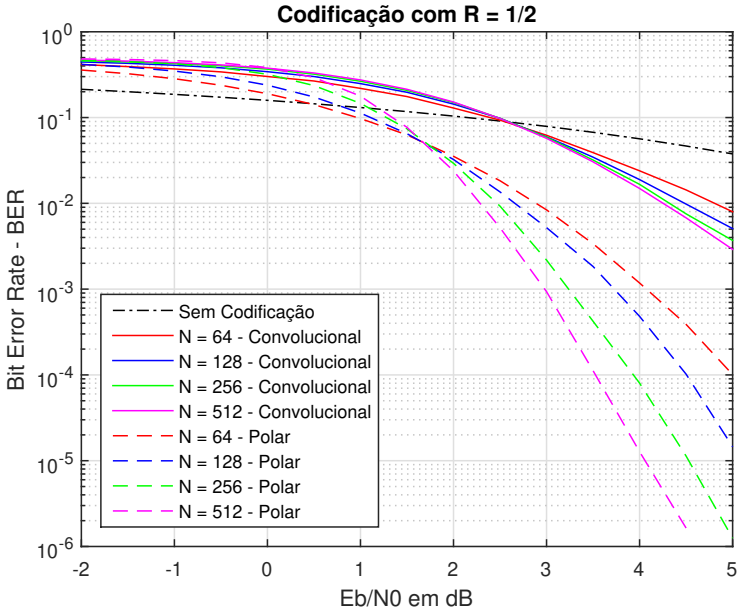
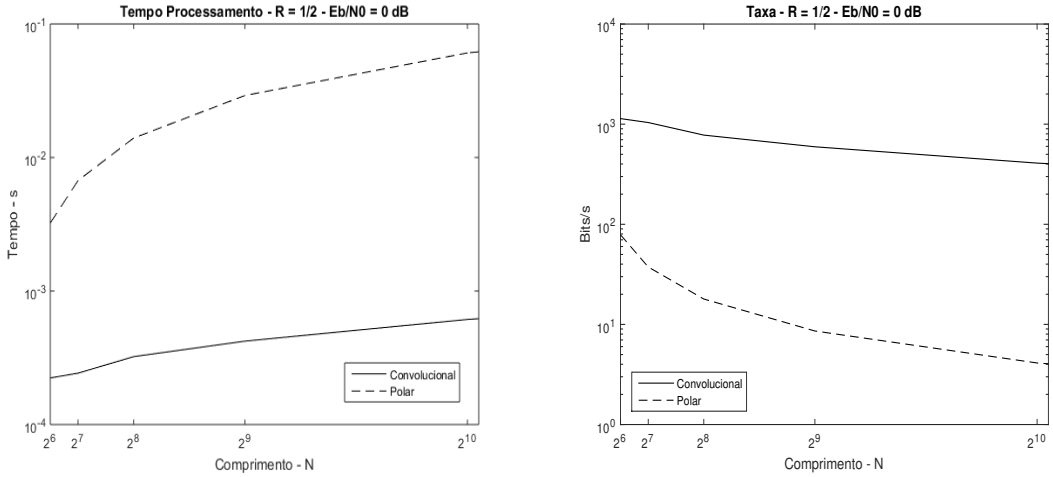


Figura 3: BER com taxa 1/2 para (N,k) = (64,32; 128,64; 256,128; 512,256)

A segunda análise feita, tem como objetivo avaliar a latência de processamento destes códigos. A simulação realizada contabiliza o tempo necessário para codificar e decodificar uma mensagem de interesse. Podemos observar na Figura 4(a) os resultados obtidos. Podemos observar que os códigos convolucionais possuem uma latência de processamento bem baixa, menor que 1 ms. Já os códigos polares possuem uma latência de processamento muito alta, o que não é interessante, pois somente o tempo de processamento já supera o limite de 10 ms do cenário. Vale ressaltar, que o maior tempo de processamento dos códigos polares, encontra-se na decodificação. Deste modo, os códigos polares, utilizando a decodificação SC, não é viável para estes cenários em estudo.

A terceira análise realizada foi com relação ao *throughput*, este parâmetro é muito importante, pois com ele é possível observar a quantidade de bits de informação que são decodificados por segundo pelo decodificador, [6]. Novas simulações foram realizadas, utilizando os dados coletados com a Figura 4(a) e a Equação 10, gerando os resultados apresentados na Figura 4(b). Podemos observar que os códigos convolucionais conseguem decodificar mais bits por segundo que os códigos polares, isto ocorre pelo fato da latência de processamento dos códigos convolucionais ser menor, assim estes códigos possuem mais tempo para processar bits de informação. Além disso, quanto maior a palavra enviada pior será o desempenho dos códigos polares, isto ocorre novamente pela latência de processamento destes códigos, pois com o aumento da palavra-código a latência também é maior. Se compararmos um comprimento de palavra de 64 com 256 bits, o *throughput* decaí quase pela metade.

DOI: 10.5540/03.2021.008.01.0360



(a) Latência de processamento

(b) *Throughput*

Figura 4: Latência e *Throughput* com taxa 1/2

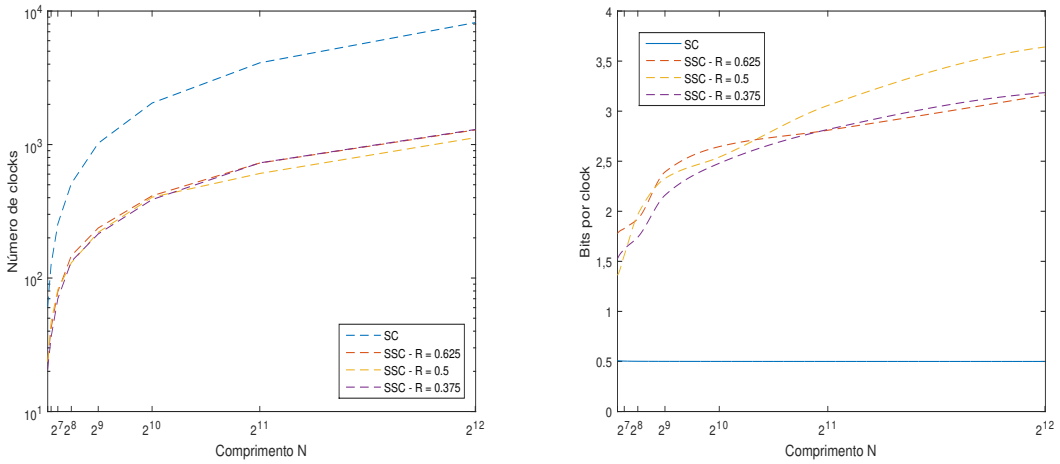
Um método para melhorar a latência dos códigos polares é a utilização de outro método de decodificação, visto que a decodificação SC demanda de maior tempo de processamento pela necessidade de efetuar todas as etapas da decodificação mesmo conhecendo a posição e os valores dos bits congelados. Assim, realizou-se uma comparação entre os dois decodificadores SC e SSC. Implementando as Equações 8 e 9 foi possível comparar a quantidade de *clocks* necessários para decodificar uma mesma mensagem de interesse e os resultados são apresentados na Figura 5(a) os resultados obtidos. Para comparar a quantidade de bits que são decodificados em média por ciclo de *clock* em cada decodificação, novas curvas foram geradas, como pode ser observado na Figura 5(b). Podemos observar que a decodificação SC possui uma quantidade média de bits que são decodificados por segundo constante, independente do comprimento da palavra este decodificador decodifica aproximadamente 0.5 bit por *clock*. Já os decodificadores SSC, possuem um desempenho superior e que melhora conforme o comprimento da palavra-código aumenta, pois neste caso, mais bits são decodificados em média por ciclo de *clock*. Além disso, o decodificador SSC com taxa 0.5 possui um desempenho melhor se comparado com as outras duas taxas simuladas, pois decodifica até meio bit a mais para comprimentos maiores de palavra.

4 Conclusões

Neste trabalho foram apresentados codificadores e decodificadores de códigos convolucionais e polares e, uma implementação computacional para avaliar o desempenho dos mesmos em cenários desejáveis para sistemas 5G. Os resultados mostram que os códigos polares possuem um bom desempenho em correção e detecção de erro, entretanto possuem alta latência, utilizando o decodificador por cancelamento sucessivo. Já os códigos convolucionais se sobressaem em latência, entretanto não atingem as métricas esperadas em correção e detecção de erro.

Agradecimentos

Os autores agradecem o apoio financeiro da FAPESP Processo 2018/26733-8.



(a) *Clocks* necessários para decodificar diferentes comprimentos de palavras

(b) Bits decodificados por *clock* para diferentes comprimento de palavra

Figura 5: Análises de *clock*

Referências

[1] Arikan, E. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels, *IEEE Transactions on Information Theory*, 16:3051–3073, 2009.

[2] Forney, G. Convolutional codes I: Algebraic structure, *IEEE Transactions on Information Theory*, 16:720-738, 1970.

[3] Lema, M. A., et al. Business case and technology analysis for 5G low latency applications. *IEEE Access*, 5:5917-5935, 2017.

[4] Popovski, P. Ultra-reliable communication in 5G wireless systems. *e1st International Conference on 5G for Ubiquitous Connectivity - IEEE*, 2014.

[5] Sarkis, G. and Gross, W. J. Increasing the throughput of polar decoders. *IEEE Communications Letters*, 17:4, 725-728, 2013.

[6] Sarkis, G. and Gross, W. J. Polar codes for data storage applications. *2013 International Conference on Computing, Networking and Communications - IEEE*, 2013.

[7] Shannon, C. E. A mathematical theory of communication. *The Bell System Technical Journal*, volume 27, no. 3, pages 379-423, 1948.

[8] Sybis, M., Wesolowski, K., Jayasinghe, K., Venkatasubramanian, V. and Vukadinovic, V. Channel coding for ultra-reliable low-latency communication in 5G systems. *2016 IEEE 84th Vehicular Technology Conference*, 1-5, 2016.

[9] Tahir, B., Schwarz, S. and Rupp, M. BER comparison between Convolutional, Turbo, LDPC, and Polar codes, *24th International Conference on Telecommunications - IEEE*, 1-7, 2017.

[10] Zhang, C. and Keshab, K. P. Latency analysis and architecture design of simplified SC polar decoders. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 61:2, 115-119, 2013.

