Optimization of the iterative decoding algorithms for irregular block codes

Jovan Milojkovic
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
mj205018p@student.etf.bg.ac.rs

Srdjan Brkić
Tannera Technologies LLC
Belgrade, Serbia
brka05@gmail.com

Predrag Ivaniš
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
predrag.ivanis@etf.bg.ac.rs

Bane Vasic
Department of ECE
University of Arizona
Tucson, USA
vasic@ece.arizona.edu

2024 32nd Telecommunications Forum (TELFOR) | 979-8-3503-9106-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/TELFOR63250.2024.10819061

Abstract — **In this paper, we analyze the performances of iterative decoders for linear block codes. In particular, we consider two modifications of the gradient-descent bit flipping (GDBF) algorithm with momentum, where multiple component decoders with different momentum values are concatenated to improve the decoder performance. The learning parameters of the component decoders are obtained by using a Genetic algorithm based on the database of the uncorrectable error patterns of the previous decoder. We present three optimization strategies and provide a comparison with the state-of-the-art decoders. The numerical results are presented on short Bose-Chaudhuri-Hocquenghem (BCH) codes and the channel with additive white Gaussian noise (AWGN).**

Keywords— Iterative decoders, genetic algorithm, GDBF decoder, error correction codes, optimization, BCH codes.

I. INTRODUCTION

Contemporary communication systems' reliability is usually provided by powerful error correction codes. In the fifth-generation standard for broadband cellular networks (5G NR) [1], low-density parity-check (LDPC) codes are applied for error correction. In the European standards for satellite digital video broadcasting (DVB-S2 and DVB-S2X) [2], longer LDPC codes are concatenated with Bose–Chaudhuri–Hocquenghem (BCH) codes, providing capacity-approaching error correction capabilities.

The popularity of LDPC codes is mostly related to low-complexity iterative decoding algorithms, which operate on a bipartite graph. The state-of-the-art LDPC decoders are usually based on the belief propagation (BP) algorithm that is capable of achieving the maximum likelihood (ML) bound in the case when the short cycles are absent in the bipartite graph, which has regular or optimized irregular row and column weight distributions [3]. Recent research results indicate that further reduction in decoding complexity can be obtained if algorithms based on the gradient descent bit flipping are applied [4-7].

This research was supported by the Science Fund of the Republic of Serbia, under grant No. 7750284 (Hybrid Integrated Satellite and Terrestrial Access Network - hi-STAR). This work was also supported by the Serbian Ministry of Science, Technological Development and Innovation. Bane Vasić acknowledges support by the NSF under grants CIF-2106189, CCF-2100013, ECCS/CCSS-2027844, ECCS/CCSS-2052751, and in part by the CoQREATE program under grant ERC-1941583, Bane Vasić has disclosed an outside interest in his startup company, Codelucida to The University of Arizona. Conflicts of interest resulting from this interest are being managed by The University of Arizona in accordance with its policies.

Although both LDPC and BCH codes belong to the class of linear block codes, achieving the ML bound for BCH codes is not simple. Classic hard-decision BCH decoders have modest correction capabilities. Soft-decision decoders, which use channel measurements in the decoding process, usually improve performance at the price of huge complexity [8].

An interesting approach for decoding of various BCH codes using the BP algorithm was presented in [9,10]. Recently, it has been shown that the neural BP (NBP) decoder, as a generalization of BP obtained by adding learnable weights to messages passed between nodes of the bipartite graph, can be applied in decoding BCH codes [11-13].

We proposed the adaptive diversity gradient-descent bit-flipping (AD-GDBF) decoder for LDPC codes [14], outperforming BP-based decoders for binary symmetric channels. This decoder was realized as a concatenation of multiple GDBF decoders with momentum (GDBF-w/M) [7]. In this approach, the momentum values and energy function coefficients were optimized for every particular LDPC code, based on the database of uncorrectable error patterns. The generalized AD-GDBF (gAD-GDBF) algorithm was proposed in our recent paper [15], where this approach was extended to BCH codes and the channel with additive white Gaussian noise (AWGN). The gAD-GDBF algorithm has been shown to outperform the BP and NBP decoders for the analyzed BCH codes. Furthermore, the proposed decoder is less complex than the state-of-the-art decoders or its improvements.

In this paper, we analyze the simplified version of AD-GDBF algorithm, and we try to identify a simple optimization strategy that should provide the best performance in the low latency scenario.

II. INTRODUCTION

A. GDBF algorithm for linear block codes

Let $H_{m\times n}$ be the parity check matrix of a linear block code (n,k). With $h_{j,i}$ let us depict the element of the matrix in j-th row and i-th column. Each row of the parity check matrix represents a parity check equation c_j, and each column of the parity check matrix represents one variable bit v_i. The number of ones in i-th column of the parity check matrix represents the degree of the variable bit v_i. With $P(v_i) = \{j|h_{ji} = 1\}$ let us depict the set of indices that shows in which

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

Authorized licensed use limited to: University of Arizona. Downloaded on February 10,2025 at 18:36:51 UTC from IEEE Xplore. Restrictions apply.

parity check equations variable bit v_i is present. Also, it can be seen that $|P(v_i)|$ is the degree of the variable bit v_i. Similarly, with $Q(c_j) = \{i|h_{ji} = 1\}$ let us depict the set of indices that show which variable bits are coupled with parity check equation c_j.

We observe the bipolar codeword $x = (x_i|i = 1 \dots n)$, where $x_i \in \{1, -1\}$, that is being transmitted over the AWGN channel. The channel harms the transmitted codeword and outputs a vector $y = (y_i|i = 1 \dots n)$, where $y_i \in \mathcal{R}$. It can be seen that for every parity check equation c_j, for a transmitted bipolar codeword x, the next statement must be satisfied $\forall j$, $\prod_{i\in Q(c_j)} x_i = 1$.

The gradient descent method for decoding of LDPC codes is presented in [4], and the same algorithm can be applied to any linear block code. Here, the algorithm tries to optimize an objective function, defined in [4, Eq. (5)]. The algorithm does this by associating energy to variable bits and flips those variable bits, which have energy below some threshold. This is done in iterations, where a group of bits, or one bit, is flipped in each iteration. Let us depict the current iteration of the algorithm as l, and with L_{max} let us depict the maximum number of iterations for the algorithm.

The corresponding energy function can be defined as in [15], where for the l-th iteration and for the i-th variable bit, we calculate

$$E_i^l = w_{1,i} y_i \dot{x}_i^{l-1} + w_{2,i} \sum_{j\in P(v_i)} \prod_{o\in Q(c_j)} \dot{x}_o^{l-1} + m_{l_i}, \quad (1)$$

where $\dot{x}_i^{l-1}$ is the estimated codeword in the $l-1$ iteration, $\dot{x}_i^0 = sign(y_i)$, and i=1, 2,…, n. Weight factors $w_{1,i}$ and $w_{2,i}$ are learnable weights, and m_{l_i} is the momentum factor defined as in [15].

It can be seen that every variable bit can have different weight factors. In our work, we consider only the case when a group of variable bits which have the same degree has the same weight factors, i.e. $w_{1,i} = w_{1,|P(v_i)|}$ and $w_{2,i} = w_{2,|P(v_i)|}$. It is interesting to notice that the energy function used in the GDBF-w/m algorithm (introduced in the paper [7]) can be obtained as a special case of Eq. (1) if we set $w_{1,i} = w_1$ and $w_{2,i} = 1$, $\forall i$. Furthermore, Eq. (1) reduces to the energy function from the GDBF algorithm (presented in the paper [4]) if $w_{1,i} = 1$, $w_{2,i} = 1$, and when the momentum does not exist.

B. Learning of the component decoders

The decoder parameters are optimized for the specific linear block code and the channel conditions. Before the decoder design, the parity check matrix $\boldsymbol{H}$ of the linear block code is specified, and we fixed the value of E_b/N_0 (denoting the energy per bit divided by the power spectrum density of the AWGN channel).

At the beginning of the learning process, a simple GDBF algorithm is used to decode the linear block code specified with matrix H. Monte Carlo simulation is performed in not more than L_1 iterations (L_k denotes the maximum number of iterations for the k-th decoder), to collect the database of uncorrectable error patterns for the given E_b/N_0. For the first decoder, the set of learnable parameters in the decoder is specified (for the algorithm presented in the previous section, these parameters are $w_{1,i}$, $w_{2,i}$, and m_{l_i}, for i=1, 2,…, n), and possible values of these parameters have to be specified. An optimization algorithm is used to choose the optimal values of these parameters from the predefined sets, based on the previously created database of uncorrectable patterns.

When the first decoder in the chain is specified, some error patterns in the previous database must be removed if this decoder can correct them. After L_2 iterations, Monte Carlo simulation is started to identify a new set of uncorrectable error patterns that should be used to extend the previous database. The updated database will be used for the optimization of the second decoder in the chain. The procedure described in this paragraph is repeated for the next decoders in the chain.

C) The application of the genetic algorithm

The Genetic Algorithm (GA) is an algorithm that can be used for optimization problems. It is a population-based algorithm where every solution to the problem is represented by a chromosome [16]. Each solution of the problem has its fitness value, and the higher fitness values correspond to better optimization. In the first generation, the chromosomes are chosen randomly, and individuals are chosen in a selection phase. The best individuals are transferred to the next generation (elitism). The chosen individuals begin a combination process in which their genes are combined to generate a new solution to the problem. After the mutation phase, a new population is created. After several generations, we terminate the algorithm and use the best chromosome as the solution to the given problem.

In our case, the chromosomes represent parameters for one decoder, i.e., one solution to the problem. By GA for one decoder and some initial number of iterations, we learn weights for energy function and momentum values. In order to use the GA for optimization, learnable weights, and momentum are represented by bits. The values of weights $w_{1,|P(v_i)|}$ and $w_{2,|P(v_i)|}$ are quantized, and those values are used for their representation. The weights should not be zeros, so the counting starts with one. For one weight, we get some integer value from the bits, add one, and then divide that value with some normalizing factor and obtain the value for that one weight. A different approach is applied for the momentum with the length L'. Before optimization, a matrix $M_{2^u,L'}$, where 2^u is the number of momentums used in optimization, is constructed. The construction consists of making all possible momentums with predefined values. The number of unique momentums is less than 2^u. Since the number of unique momentums is less than 2^u, we fill the rest of the elements of M with already used momentums starting from momentum at the first row of the matrix M. Then we use u bits to select one momentum vector, one row of the matrix M, which will be used in a decoder solution.

III. Numerical Results

This paper will present the numerical results for a short right-regular BCH code with n=63 and k=36 (given in [17]). The codewords are transmitted through the AWGN channel, and a real-valued channel measurement vector $\boldsymbol{y}$ appears at the decoder's input. Iterative decoding based on the algorithm described in Section II is performed for different strategies of the parameters' adaptation.

Authorized licensed use limited to: University of Arizona. Downloaded on February 10,2025 at 18:36:51 UTC from IEEE Xplore. Restrictions apply.

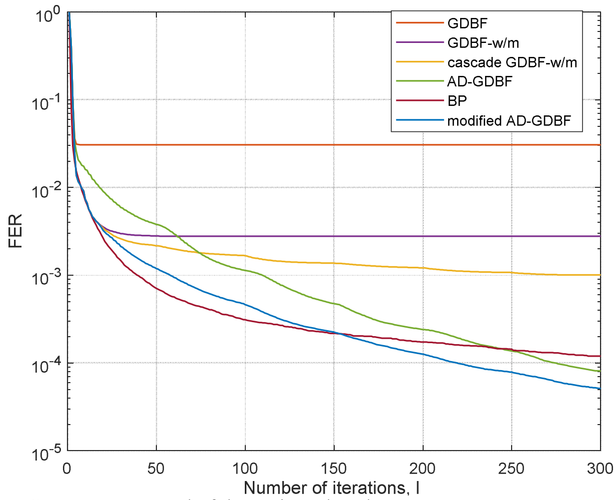

Fig. 1. Convergence speed of the various decoders.

We first show how frame error rate (FER) depends on the parameter L_{max}. The corresponding numerical results for various decoders and E_b/N_0=7 dB are presented in Figure 1. It is obvious that the GDBF algorithm (without momentum) only corrects errors in the first few iterations. Even if the momentum is optimized in the GDBF-w/M, the performance improvement diminishes after 50 iterations.

Further, we consider a concatenation of the GDBF-w/M decoders, where the component decoders have different momentum vectors $\boldsymbol{m}$ and energy function coefficients $\boldsymbol{w}_1$. In this scenario, L_1=15, L_2=30, L_k=50 for k=3,4,..,6, and L_7=55. The next decoder learns the optimum values of $\boldsymbol{m}$, $\boldsymbol{w}_1$ using the genetic algorithm applied to the database of uncorrectable error patterns of the previous decoder in the chain. Although a certain improvement is visible, the performance is not comparable with the BP algorithm for the same number of iterations.

The same learning procedure can be repeated when the parameters $\boldsymbol{m}$, $\boldsymbol{w}_1$, and $\boldsymbol{w}_2$ are optimized, with the modification that the same weighting coefficients are determined for the group of the variable bits with the same weight. This approach is a simplified version of the AD-GDBF algorithm [14], which considers the code irregularity and L_k=50, $\forall k$. This decoder outperforms the BP algorithm for L_{max}>240. However, it is interesting that the GDBF-w/M algorithm without concatenation of decoders outperforms the AD-GDBF algorithm for L_{max}<60.

Therefore, we propose the modification of the AD-GDBF, on the basis of the hybrid approach. This approach is described as follows. The first decoder in the chain is the GDBF-w/M decoder, with optimized parameters $\boldsymbol{m}$ and $\boldsymbol{w}_1$, where these parameters are optimized based on the database of uncorrectable error patterns of the GDBF decoder. The remaining decoders in the concatenation are optimized as in the AD-GDBF algorithm. The maximum number of iterations for every component decoder is determined to avoid the saturation of FER. This approach results in performance comparable with BP for any value of L_{max}, as presented in Fig. 1. This is an important conclusion, as it is known that the AD-GDBF is a less complex algorithm when compared to BP [14].

The first decoder in the chain has the simplified energy function (denoted by EF1), with the same values of the $\boldsymbol{w}_1$ for all variable nodes, and we do not take into account $\boldsymbol{w}_2$ (i.e. $w_{2,i} = 1, \forall i$). Although this represents a special case of the general energy function, given in Eq. (1) and denoted by EF2, the GA does not find this optimal solution if the number of optimization parameters is too large. In Fig. 2, fitness factors for various generations are presented when we apply 40 chromosomes. If EF2 with more learnable parameters is used, the fitness factor increases even after 150 generations. If EF1 is used, the fitness factor is usually smaller, but the optimal solution is achieved after not more than 30 generations.

The results presented in Fig. 2 also indicate that the performance improvement due to the use of EF2 is smaller for the decoder at the beginning of the chain. This is why EF1 can be used in the first decoder without sacrificing the error performance and reducing the optimization space. Even if EF2 is used, fitness factors generally decrease if the decoder is closer to the end of the chain. About 70% of the errors remaining after the first decoding stage are corrected by the second decoder. Only 33% of the remaining error patterns from the previous decoder are corrected by the seventh decoder in the chain.

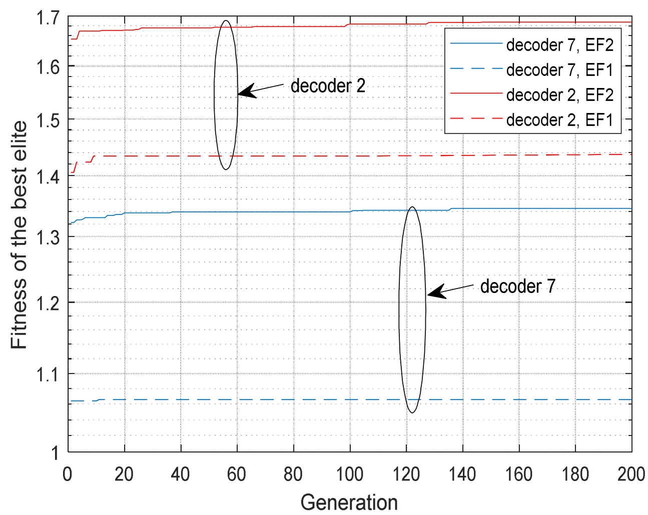

Fig. 2. Achieved fitness rate for the two approaches.

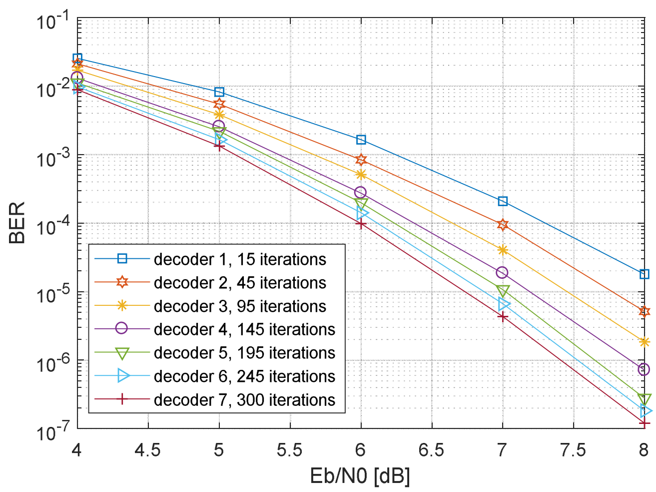

Fig. 3. Performance of the component decoders.

Authorized licensed use limited to: University of Arizona. Downloaded on February 10,2025 at 18:36:51 UTC from IEEE Xplore. Restrictions apply.

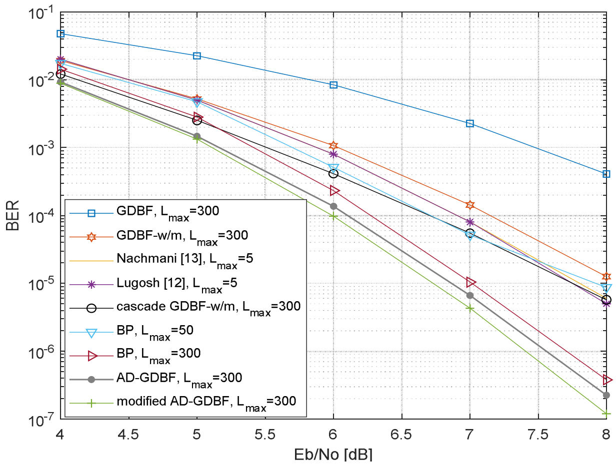

Fig. 4. Performance of the various decoding algorithms.

Fig. 3 presents the performance improvement after applying each of the seven component decoders. The bit error rate (BER) for the first decoder corresponds to the GDBF/w/m [7], where the momentum vector $\boldsymbol{m}$ and coefficient $\boldsymbol{w}_1$ are optimally chosen. The concatenation of the decoders reduces the BER, but the improvement decreases for every additional component decoder.

Finally, in Figure 4, we present BER as a function of E_b/N_0. It is obvious that the AD-GDBF decoder is far superior to the GDBF and GDBF-w/M decoders. If the adaptation of the momentum in 300 iterations (as proposed in [14]) is combined with the energy function EF1 (as proposed in [7]), the achieved performance of the corresponding cascade GDBF-w/M algorithm is comparable with that of the BP algorithm with 50 iterations.

The AD-GDBF decoder with optimized weighting coefficients that take into account the irregularity of the BCH code, with the proposed modification, outperforms the BP decoder with coding gain close to 0.25 dB for the same number of iterations (L_{max}=300). Furthermore, we can see that the proposed decoder outperforms the NBP and neural offset min-sum (NOMS) decoders, designed for L_{max}=5 iterations. Decoders obtained by GA and used parity check matrix are given in [17].

IV. Conclusion

In this paper, we have analyzed the performance of BCH codes when iterative decoding algorithms are applied. We have used the main idea from AD-GDBF, and the importance of the decoder parameters is investigated. Furthermore, we have proposed one modification of the optimization procedure, which reduces the problem space and, by doing so, improves the decoder performance. The modification is based on combining two energy functions in a decoder, one with a smaller problem space and another with a bigger one. It can be seen that the given hybrid approach can improve the performance of the overall decoder in Figures 1 and 4 when compared to other approaches, where the synergy of the decoders is also implemented, but only one energy function is used.

For fuIn true research, we open several questions in this paper. It will be interesting to investigate if there is any other optimization algorithm besides GA that should yield better results. Also, we will try to identify another method to further reduce the problem space so that AD-GDBF yields better decoders with fewer generations.

References

[1] 3rd Generation Partnership Project. Technical Specification Group Radio Access Network; NR; Multiplexing and Channel Coding (Release 16), 3GPP TS 38.212 V16.5.0 (2021-03); 3GPP: Valbonne, France, 2021. [Google Scholar]

[2] ETSI Digital Video Broadcasting (DVB). Second Generation Framing Structure, Channel Coding and Modulation Systems for Broadcasting, Interactive Services, News Gathering and other Broadband Satellite Applications; Part 2: DVB-S2 Extensions (DVB-S2X), ETSI EN 302307-2 V.1.1.1 (2014-10); ETSI: Sophia Antipolis, France, 2014.

[3] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 599-618, Feb 2001.

[4] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, “Gradient descent bit flipping algorithms for decoding LDPC codes,” IEEE Trans. Commun., vol. 58, no. 6, pp. 1610–1614, June 2010.

[5] O. A. Rasheed, P. Ivanis, and B. Vasic, “Fault-tolerant probabilistic gradient-descent bit flipping decoder,” IEEE Commun. Letters, vol. 18, no. 9, pp. 1487–1490, Sep. 2014.

[6] B. Vasić, P. Ivaniš, D. Declercq, and K. LeTrung, “Approaching maximum likelihood performance of LDPC codes by stochastic resonance in noisy iterative decoders,” in Proc. Inf. Theory Appl. Workshop (ITA2016), Feb. 2016, pp. 1–9.

[7] V. Savin, “Gradient descent bit-flipping decoding with momentum,” in Proc. 2021 11th Inter. Symp. on Topics in Coding (ISTC), Montréal, Canada, Aug. 2021, pp. 1-5.

[8] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," IEEE Trans. Inf. Th., vol. 18, no. 1, pp. 170-182, January 1972.

[9] J. Jiang and K. R. Narayanan, "Iterative Soft-Input Soft-Output Decoding of Reed–Solomon Codes by Adapting the Parity-Check Matrix," IEEE Trans. Inf. Th., vol. 52, no. 8, pp. 3746-3756, Aug. 2006,

[10] M. Baldi, F. Chiaraluce, “A Simple Scheme for Belief Propagation Decoding of BCH and RS Codes in Multimedia Transmissions”, International Journal of Digital Multimedia Broadcasting, Vol. 2008, Article ID 957846, 12 p.

[11] E. Nachmani, Y. Be’ery, and D. Burshtein, “Learning to decode linear codes using deep learning,” in Proc. 54th Annu. Allerton Conf. Commun., Control, Comput, Allerton, IL, USA, Sep. 2016.

[12] L. Lugosch and W. J. Gross, "Neural offset min-sum decoding," 2017 IEEE Inter. Symp. on Inf. Th. (ISIT), Aachen, Germany, 2017, pp. 1361-1365

[13] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be’ery, “Deep learning methods for improved decoding of linear codes,” IEEE J. Sel. Topics Signal Process., vol. 12, no. 1, pp. 119–131, Jan. 2018.

[14] S. Brkic, P. Ivanis, and B. Vasic, “Adaptive gradient descent bit-flipping diversity decoding,” IEEE Commun. Letters, vol. 26, no. 10, pp. 2257–2261, Oct. 2022.

[15] J. Milojković, S. Brkic, P. Ivaniš and B. Vasić, "Learning to Decode Linear Block Codes using Adaptive Gradient-Descent Bit-Flipping," in Proc 2023 12th Inter. Symp. on Topics in Coding (ISTC), Brest, France, ž2023, pp. 1-5.

[16] S. Mirjalili, Evolutionary Algorithms and Neural Networks, Theory and Applications, Springer, 2019.

[17] Used decoders database with parity check matrix. [Online]. Available: https://github.com/jovan94/TELFOR2024 .

Authorized licensed use limited to: University of Arizona. Downloaded on February 10,2025 at 18:36:51 UTC from IEEE Xplore. Restrictions apply.