Bit-Flipping Algorithm for Joint Decoding of Multiple Correlated Sources transmitted over noisy channels.

Fernando Pujaico Rivera and Jaime Portugheis, *Member, IEEE*

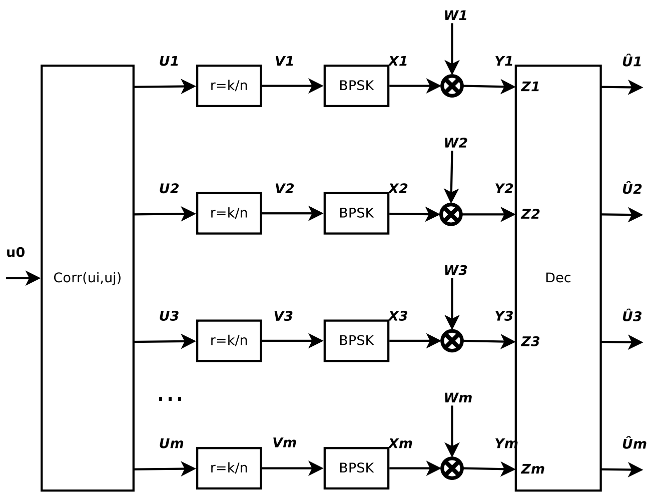

Figura 1. Diagrama de blocos do modelo geral de decodificação.

***Resumo*—This paper proposes a bit-flipping decoding algorithm suitable for decoding multiple correlated sources in a noisy envinroment. All sources are noisy versions of a single binary source and are transmitted through identical orthogonal binary symmetric channels (BSCs). The noisy versions are obtained by transmitting this single source through independent BSCs. The set of crossover probabilities of these BSCs can be chosen randomly. By fixing one of these probabilities to a value of approximately one, the decoding can be interpreted as decoding with multiple side information. Performance of the algorithm was obtained by computer simulation for systems up to 100 sources. All sources were encoded independently with short LDGM codes. For decoding with side information, the algorithm approaches a lower bound on performance. In conjuction with a fusion decision rule, the algorithm can be applied to a large scale sensor network modeled by the binary CEO problem.**

***Index Terms*—LDGM codes, hard-decision decoding, BF decoding algorithm.**

I. Introduction

II. System Model

A Figura 1 mostra o modelo de codificação e decodificação de fontes conjuntas.

As fontes u_i foram geradas usando uma fonte primaria binaria u_0 com $p(u_0 = 1) = 0.5$ e fontes secundarias e_i, $i \in \{1, ..., m\}$. Onde $p(e_i = 1) = p_i$

$$u_i = u_0 \otimes e_i \tag{1}$$

Manuscript received XXXX XX, 2010; revised XXXXX XX, 2013. Department of Communications, Faculty of Electrical and Computer Engineering. State University of Campinas (UNICAMP), Campinas-SP, Brazil.

E-mails: {fpujaico,jaime}@decom.fee.unicamp.br

III. Bit-Flipping Decoding

Existem muitos algoritmos simples para realizar uma decodificação da informação que percorre um canal com ruido. Estes algoritmos podem usar uma decisão suave (“soft”) ou abrupta (“hard”). Se diz que um algoritmo de decodificação realiza uma decisão suave se utiliza na sua predição dados que são números reais, pelo contrario se diz que o algoritmo de decodificação realiza uma decisão abrupta quando usa dados inteiros como dados de entrada.

Aqui se trabalhará sobre algoritmos de decodificação para códigos de verificação de paridade de baixa densidade ou também chamado $LDPC$, do inglês “Low Density Parity Check”.

Como pode-se ver na Figura 1 se tem m canais ruidosos com vetores Y_i na sua saída, ou com sua contraparte abrupta Z_i. Pode-se realizar com estes vetores uma decodificação para obter uma estimado $\hat{U}_i$ de U_i, baseando-se só na redundância acrescenta por cada codificador fonte-canal de taxa r (a esto se chamará decodificação independente), ou pode-se usar a informação redundante acrescentada pelo conhecimento da informação ruidosa recebida das outras $m-1$ fontes u_i (esto se chamará decodificação conjunta).

A. Algoritmos de decodificação independente

Entre os algoritmos de decodificação independente temos.

1) Algoritmo Parallel Hard Bit-Flipping: Ela trabalha sobre um vetor de entrada binário Z e tem como regra de geração de confiabilidade, E_{Ij},

$$E_{Ij} = \sum_{l \in \mathcal{M}(j)} (2s_l - 1), \tag{2}$$

O critério de troca dos bit errados é igual ao algoritmo BF. Troca-se todos os bits com um $E_{Ij} \geq \delta$. Para a implementação atual usou-se como δ o maior valor de E_{Ij}. O algoritmo III-A1 detalha todo o procedimento para a decodificação $PHBF$.

Decodificação Parallel Hard Bit-Flipping.

1) Inicia-se com $i = 0$ e o vetor de decisão abrupta $Z^{(i)} = (z_1^{(i)}, \cdots, z_j^{(i)}, \cdots, z_n^{(i)}) = Z$.
2) Calcula-se a síndrome $S = H^t Z^{(i)}$ (mod 2). Se todos os valores s_l com $l = \{0, \cdots, L-1\}$ são zero, então pára a decodificação e se retorna o vetor $Z^{(i)}$.

3) Para $j = 0, \cdots, n-1$, avalia-se a função de flipping E_{Ij} em (2).
4) Identifica-se o conjunto $\{j^*\}$, onde $j^* = \arg\max_j E_{Ij}$.
5) Troca-se de valor todos os bits $z_j^{(i)}$ onde $j \in \{j^*\}$, e faz-se $Z^{(i+1)} = Z^{(i)}$.
6) Se o máximo número de iterações não é atingido, faz-se $i = i + 1$ e vai-se ao passo 2. Caso contrário, se detêm a decodificação e se retorna Z.

A confiabilidade de cada bit z_j é calculada somando a quantidade de bits de verificação de paridade s_l que ligados a z_j indiquem a existência de um erro, e diminuindo a quantidade de bits de verificação de paridade s_l que ligados ao bit z_j não indiquem a existência de erro. "*Muito parecido a uma votação, onde são contabilizados os votos em contra e os votos a favor que anulam um voto em contra, ao final o conjunto de bits que tenham mais votos em contra serão trocados*".

B. Algoritmos de decodificação conjunta

O algoritmo de decodificação conjunta aqui apresentado é uma modificação aos algoritmos de decodificação independente. Em geral a modificação será trocar o uso da confiabilidade independente E_{Ij} do bit z_j por uma confiabilidade total $E_{Tj} = E_{Ij} + \beta E_{Cj}$. Onde β é um fator de ponderação e E_{Cj} é a confiabilidade conjunta para cada bit z_j do vetor recebido Z. Esta confiabilidade conjunta será calculada usando os dados dos bits recebidos dos outros $m-1$ canais e ponderado em função à correlação como eles. Assim o caso da decodificação independente é um caso especifico da decodificação conjunta quanto $\beta = 0$.

Para fazer a decodificação conjunta se definirá para cada um dos m canais de informação a confiabilidade ${E_T}_j^i$, onde i indica a que canal corresponde a confiabilidade do bit recebido z_j^i.

1) Algoritmo Distributed Sources Parallel Hard Bit-Flipping: O algoritmo "Distributed Sources Parallel Hard Bit-Flipping" (DS-PHBF) usa como confiabilidade

$$ {E_T}_j^i = {E_I}_j^i + \lfloor \beta {E_C}_j^i \rfloor. \quad (3)$$

Onde $\lfloor b \rfloor$ é o máximo inteiro de b, e β é um fator de ponderação entre a confiabilidade independente e a confiabilidade conjunta. Para calcular ${E_C}_j^i$ é necessário conhecer

$$ L^i = \sum_{a=1|(a\neq i)}^{m} 1, \quad (4)$$

dado que

$$ {E_C}_j^i = \sum_{a=1|(a\neq i, z_j^i \neq z_j^a)}^{m} \frac{{E_I}_j^a \ Corr(i,a)}{L^i} - \sum_{a=1|(a\neq i, z_j^i = z_j^a)}^{m} \frac{{E_I}_j^a \ Corr(i,a)}{L^i} \quad (5)$$

IV. Numerical Results

O desempenho do algoritmo $DS-PHBF$ se vê na Figura **??**. Nesta figura temos 10 fontes binarias correlacionadas u_i com $i \in \{1, 2, \ldots, 10\}$ e $H(u_i) = 1.0$.

As fontes foram criadas seguindo una distribuição uniforme entre 0.01 e 0.99 para os valores $p_i \in \{$ 0.01, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.91 $\}$ no modelo de geração de fontes explicado na seção II. As 10 fontes foram codificadas com um código $LDGM$ como um valor de $k = 204$ bits de informação e $n = 306$ bits codificados. A proteção para os bits de informação dentro do vetor codificado é de $X = 5$. Se aplicaram a todas as fontes os algoritmos $PHBF$ e $DS-PHBF$.

Na Figura 2 se vê desenhado com "$*$" o desempenho dos algoritmos $PHBF$, como é natural a decodificação de todas as fontes tem o mesmo desempenho dado que usam o mesmo algoritmo e a mesma matriz de verificação de paridade. O desempenho dos algoritmos $DS-PHBF$ se vê desenhado com "$-$", como pode-se ver todos os canais tiveram uma apreciável melhora no seu desempenho, nenhum dos canais pioraram seu desempenho. A media do desempenho do algoritmo esta desenhado com "□". O desempenho dos algoritmos $DS-PHBF$ em media mostraram ser ligeiramente melhor do desempenho dos algoritmos $PHBF$. Todos os algoritmos usaram $IT = 15$ iterações como máximo antes de desistir de corrigir os erros. O limite inferior do desempenho dos códigos $LDGM$ [**?**] esta desenhado com "O". O desempenho num canal não codificado esta desenhado com uma linha pontuada "$-.-$".

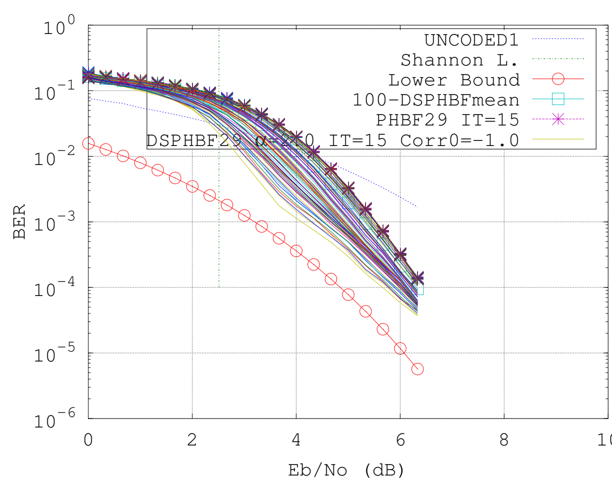

Figura 2. Gráfico do desempenho da decodificação de 100 fontes u_i correlacionadas com $K = 204$ e $N = 306$ com códigos LDGM $X = 5$ e $Y = 10$.

V. Conclusions

Referências

[1] R. G. Gallager, *Low density parity check codes*. MIT press, Cambridge, 1963.
[2] M. Sipser and D. Spielman, "Expander codes," IEEE Trans. Inform. Theory, vol. 42, pp. 1710-1722, Nov. 1996.
[3] Y. Kou, S. Lin and M. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," IEEE Trans. Inform. Theory, vol. 47, pp. 2711-2736, Nov. 2001.
[4] D. J. C. MacKay. Encyclopedia of Sparse Graph Codes [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html
[5] J. Garcia-Frias and W. Zhong, "Approaching Shannon performance by iterative decoding of linear codes with low-density generator matrix," IEEE Commun. Lett., vol. 7, pp. 266-268, June 2003.