Aplicação de Algoritmos Genéticos no Problema de Busca de Conjuntos de Informação

José R. Hoffmann, Marcos M. Tenório, Emilio C. G. Wille e Walter Godoy Jr.

***Resumo*— A decodificação por conjuntos de informação para códigos corretores de erro reduz a complexidade e o tempo de processamento em comparação à decodificação por máxima verossimilhança, podendo apresentar os mesmos níveis de correção desta última. Na primeira abordagem uma coleção de conjuntos de informação é usada para gerar palavras código candidatas. O processo então seleciona como palavra decodificada aquela que está mais próxima da sequência recebida. O desempenho deste algoritmo depende diretamente do número de padrões de erro que a coleção de conjuntos de informação é capaz de cobrir. Não há nenhum procedimento construtivo conhecido para encontrar uma coleção otimizada. Este trabalho apresenta uma abordagem de busca de conjuntos de informação usando algoritmos genéticos. Com o uso desta abordagem foram encontradas coleções de conjuntos de informação com alta capacidade de correção em um curto espaço de tempo. Resultados de simulação computacional mostram que o desempenho do algoritmo conjuntos de informação com tais coleções otimizadas é quase idêntico ao desempenho de máxima verossimilhança.**

***Palavras-Chave*— Códigos corretores de erros, Decisão suave, Algoritmo conjuntos de informação, Algoritmos Genéticos.**

***Abstract*— Information set (IS) decoding of error correcting codes reduces the complexity and time processing in comparison to maximum-likelihood decoding (MLD), and may presents the same decoding performance. In the first approach a collection of information sets are used to generate candidate codewords. The procedure then selects as the decoded codeword the one which is closest to the received sequence. The performance of the IS algorithm depends on the number of error patterns that the collection of data sets can cover. There are no known constructive procedure for finding optimum collection of information sets. This paper presents an approach for obtaining optimum collections of information sets using genetic algorithms. Using this approach we found, in short time, collections of information sets with high error correcting capability. Results from computer simulation show that the performance of the IS algorithm with optimized collections is nearly identical to the maximum-likelihood decoding.**

***Keywords*— Error Correcting Codes, Soft-Decision Decoding, Information Set Decoding, Genetic Algorithms.**

I. INTRODUÇÃO

Códigos corretores de erros são ferramentas fundamentais usadas para transmitir informações digitais em canais não confiáveis [1]. Seja $C(n,k)$ um código de blocos binário com matriz geradora G e uma matriz de verificação de paridade H, e seja $\mathbf{c_i} \in GF(2)^n$ uma palavra código de C. O procedimento convencional de decodificação algébrica para todo código de blocos binário $C(n,k)$, com distância de Hamming mínima d_{Hmin}, possui capacidade de correção de erros $t = \lfloor (d_{Hmin}-1)/2 \rfloor$, onde $\lfloor x \rfloor$ denota o maior inteiro, menor ou igual a x. Suponha que a palavra código $\mathbf{c_i}$ é transmitida por um canal de comunicação (com ruído) e que a sequência $\mathbf{y} \in R^n$ é recebida. Seja $d_E(.,.)$ a distância Euclidiana. A tarefa de um processo de decodificação é encontrar uma palavra código $\mathbf{c_i}$ tal que $d_E(\mathbf{c_i},\mathbf{y}) \leq d_E(\mathbf{c_j},\mathbf{y})$, para todo $j \neq i$. Assim, a palavra código $\mathbf{c_i}$ é a palavra código decodificada.

A decodificação de máxima verossimilhança (*maximum likelihood decoding* - MLD) é uma abordagem ideal, mas o processo de decodificação geralmente é impraticável, especialmente quando a quantidade de palavras código é grande. Consequentemente, tem havido um grande número de pesquisas com o intuito de melhorar a eficiência da decodificação. Muitas destas pesquisas consideram a técnica de decodificação suave (*soft-decision decoding*) na qual a demodulação e a decodificação interagem usando informações de confiabilidade provenientes da demodulação no processo de decodificação [2], [3], [4], [5], [6], [7], [8].

A decodificação por conjuntos de informação (*information set (IS) decoding*) é uma técnica direta de decodificação, em que uma coleção de conjuntos de informação é usada para gerar palavras código candidatas. O processo então seleciona como palavra decodificada a que está mais próxima da sequência recebida [9]. Não há nenhum procedimento construtivo conhecido para encontrar a coleção otimizada de conjuntos de informação. É possível usar uma "busca inteligente" para construir o IS durante o processo de decodificação (por exemplo, utilizando informações disponíveis de confiabilidade), ou utilizar um conjunto "pré-determinado". Em [10] foi proposto um procedimento para a obtenção de um conjunto de cobertura que gera em seqüência um conjunto de $(n-k)$ padrões tal que cada novo padrão adiciona um número mínimo de padrões de erro anteriormente não cobertos. No entanto, esse algoritmo consome muito tempo computacional para a determinação de soluções. Boas soluções podem ser obtidas com menor tempo de processamento utilizando a abordagem baseada em algoritmos genéticos (AGs) proposta neste trabalho.

Este artigo está organizado da seguinte forma. A Seção II apresenta uma introdução à decodificação por conjuntos de informação e problemas de cobertura. A Seção III mostra uma abordagem baseada em algoritmos gulosos para encontrar conjuntos de informação, que foi originalmente proposta em [10]. A Seção IV fornece uma breve descrição sobre algoritmos genéticos e seus operadores, bem como explica o algoritmo proposto neste artigo para a determinação

J.R. Hoffmann, M.M. Tenório, E.C.G. Wille e W. Godoy Jr., Universidade Tecnológica Federal do Paraná. Av. Sete de Setembro 3165, 80230-901, Curitiba (PR), Brasil. E-mail: josericardo.hoffmann@gmail.com, mmtenorio@gmail.com, ewille@utfpr.edu.br, godoy@utfpr.edu.br

de conjuntos de informação. Na Seção V um conjunto de resultados numéricos é discutido. As conclusões são apresentadas na Seção VI.

II. Algoritmo Conjuntos de Informação

A decodificação por conjuntos de informação foi primeiramente sugerida por Prange [9] para decodificação de códigos cíclicos e tem sido extensivamente estudada e modificada. Em todas as suas formas de atuação, a decodificação por conjuntos de informação explora a redundância dos códigos. Um conjunto de informação é definido como qualquer conjunto de k posições de uma palavra código que pode ser especificado independentemente. Se não houverem erros nos símbolos contidos no IS, os símbolos restantes da palavra código recebida podem ser reconstruídos. Neste trabalho, um IS é representado por um vetor binário de comprimento n, também chamado de *máscara*, onde k bits são iguais a 1 e os outros são definidos como 0. Os bits não zeros do IS correspondem a colunas linearmente independente da matriz geradora G.

Uma coleção de conjuntos de informação que abranja todos os padrões de erro de um determinado tipo é definido como *cobertura*. Formalmente, a determinação de uma coleção de conjuntos de informação corresponde a um problema de cobertura (n, l, t) [11]. Dado um conjunto de n objetos, procura-se o número mínimo de subconjuntos de cardinalidade l, tal que qualquer subconjunto de cardinalidade t está contido em pelo menos um dos subconjuntos de cardinalidade l. No contexto de códigos corretores de erros, os subconjuntos de cardinalidade t são os padrões de erro, e os subconjuntos de cardinalidade l são as posições de paridade, tal que $l = n-k$. Uma t-tupla que é coberta por uma $(n-k)$-tupla e dita "capturada"pela k-tupla correspondente. As k-tuplas devem representar um conjunto de informação.

O algoritmo conjuntos de informação básico funciona da seguinte forma. Uma coleção de ISs é usada para gerar (a partir da sequência recebida) um conjunto correspondente de palavras código candidatas. O algoritmo seleciona a palavra código candidata que possui a melhor métrica. Este procedimento garante que, se a sequência recebida não contém erros em um dos conjuntos de informação, então a palavra código transmitida estará presente na coleção de palavras código candidatas.

III. Algoritmo Guloso para Busca de ISs

Em [10], os autores apresentaram um algoritmo guloso para encontrar uma coleção de conjuntos de informação que cobre todos os padrões de erro de um tipo particular. A geração da coleção otimizada é feita levando em consideração dois fatores:

1) Uma busca exaustiva é realizada a fim de obter a maior distância possível entre os conjuntos de informação escolhidos, usando a métrica da distância de Hamming.

2) A coleção de conjuntos de informação deve cobrir todos os padrões de erro de peso $(d_{Hmin} - 1)$ com o mínimo de sobreposições possíveis.

PROCEDIMENTO *Algoritmo Guloso*( );
(1) **INICIO**
(2) Cria lista com todos os padrões de erro;
(3) Cria lista com todas as máscaras válidas;
(4) $uncoveredErrorPatterns \leftarrow$ GenerateErrorPatterns(n, t);
(5) $unselectedMasks \leftarrow$ GenerateValidMasks(n, k, G);
(6) $selectedMasks \leftarrow$ empty;
(7) **ENQUANTO** $uncoveredErrorPatterns$ não está vazio **FAÇA**
(8) $m \leftarrow$ FindBestMask(unselectedMasks);
(9) selectedMasks.insert(m);
(10) selectedMasks.insert(m);
(11) uncoveredErrorPatterns.remove(ErrorsCoveredBy(m));
(12) **FIM ENQUANTO**
(13) Retorna lista otimizada;
(14) **FIM**;

Fig. 1. Algoritmo guloso para o problema de busca de IS´s.

O algoritmo desenvolvido em [10] é apresentado na Fig. 1. Ele se baseia nas quatro sub-rotinas seguintes:

GenerateErrorPatterns(n,t): cria uma lista com todos os padrões de erros corrigidos pelo código - todos as palavras de n bits com peso de Hamming = t.

GenerateValidMasks(n,k,G): cria uma lista com todos os conjuntos de informação de comprimento = n bits e peso de Hamming $\leq k$.

FindBestMask(máscaras): itera através da lista de máscaras, calculando o número de padrões de erro coberto por cada máscara e retorna a máscara que cobre o maior número de erros.

ErrorsCoveredBy(m): itera através de todos os padrões de erro não cobertos, retornando um subconjunto com todos os padrões cobertos pela máscara m.

Em cada iteração, uma máscara (o melhor conjunto de informação) é selecionada, e todos os padrões de erro cobertos por ela são removidos da lista de padrões ainda não cobertos. Quando o número de padrões não cobertos alcança 0, o algoritmo é finalizado. Como a máscara selecionada é a melhor disponível, a saída do algoritmo é naturalmente ordenada, começando com o conjunto de informação que cobre o maior número de erros.

IV. Algoritmos Genéticos

Os algoritmos genéticos (AGs) são heurísticas de otimização estocásticas, onde explorações no espaço de soluções são conduzidas imitando a genética de população estabelecida na teoria da evolução de Darwin. Os princípios básicos dos AGs foram estabelecidos de forma detalhada por Holland [12]. Para utilizar um algoritmo genético, é necessário representar uma solução para um problema como um genoma (ou *cromossomo*). Em um problema os parâmetros que estão sujeitos à otimização constituem o espaço fenótipo. Por outro lado, os operadores genéticos trabalham sobre aspectos matemáticos abstratos como uma sequência binária (por exemplo, os cromossomos), o espaço genótipo. *Seleção*, *operadores genéticos* e *substituição*, diretamente derivados dos mecanismos de evolução natural são aplicados a uma população de soluções, favorecem o nascimento e sobrevivência das melhores soluções.

```
PROCEDIMENTO Algoritmo Genético( );
(1)  INICIO
(2)     Insere parâmetros de projeto;
(3)     Inicializa população aleatoriamente;
(4)     Avalia o fitness de cada indivíduo;
(5)     ENQUANTO critério de parada não é satisfeito FAÇA
(6)         Aplica seleção de pais;
(7)         Aplica crossover com probabilidade p_c;
(8)         Aplica mutação;
(9)         Avalia o fitness dos filhos produzidos;
(10)        Aplica a estratégia de substituição;
(11)    FIM ENQUANTO
(12)    Retorna melhor indivíduo;
(13) FIM;
```

Fig. 2. Pseudo-código de um algoritmo genético.

A descrição em pseudo-código de um AG é mostrada na Fig. 2. A população compreende um grupo de N_I indivíduos (cromossomos), dos quais candidatos podem ser selecionados para a solução de um problema. Inicialmente, uma população é gerada aleatoriamente. Os valores de *fitness* de todos os cromossomos são avaliados através do cálculo da função objetivo em forma decodificada (fenótipo). Um grupo particular de cromossomos (pais) é selecionado à partir da população para gerar os filhos através das operações genéticas *crossover*) e (*mutação*. A aptidão dos filhos é avaliada de forma semelhante a de seus pais. Os indivíduos da população corrente são substituídos pelos seus descendentes, com base em uma estratégia de substituição. Tal ciclo do AG é repetido até que um critério de parada seja atingido (por exemplo, um número predefinido de gerações N_G é alcançado). Assim, ao longo deste processo de evolução simulada, o melhor cromossomo na população final pode se tornar uma solução altamente evoluída para o problema.

A. Aplicando AGs no problema de busca de ISs

Os parágrafos seguintes descrevem as técnicas que foram empregadas no AG para a busca de conjuntos de informação.

Esquema de codificação: Nesse trabalho, uma máscara é um vetor binário, de comprimento n, usado para representar um conjunto de informação. Assim, um cromossomo corresponde a uma lista de máscaras válidas com cardinalidade N, como na Fig. 3.

1	111111100000000
2	000000001111111
3	111000000001111
...	...
N	000011111110000

Fig. 3. Representação do cromossomo.

Avaliação do fitness: Seja ϕ_i o número de padrões de erro cobertos pela máscara na posição i (e não coberto por uma máscara em qualquer outra posição $j < i$) do cromossomo. Assim, o fitness de uma solução candidata corresponde à soma dos padrões de erros corrigidos, e pode ser avaliado pela seguinte equação:

$$fitness = \sum_{i=1}^{N} \phi_i \tag{1}$$

onde N representa o número total de máscaras.

Seleção de pais: A seleção dos pais emula o mecanismo natural de "sobrevivência do mais apto". Neste trabalho foi utilizado o método de *seleção torneio*, onde dois indivíduos são escolhidos aleatoriamente e aquele com o maior fitness (o que "vence o torneio") é usado como um pai. A seleção torneio é então repetida com uma nova seleção de dois indivíduos a fim de encontrar outro pai de características diferentes.

Operadores genéticos: O *crossover* é um operador de recombinação usado para produzir filhos. Neste trabalho é utilizado o *crossover de um ponto*. Dado dois pais, um ponto de quebra é selecionado aleatoriamente e as porções dos dois cromossomos além deste ponto são trocadas para formar novos filhos. A *mutação* é usada para evitar a convergência das soluções para ótimos locais de baixa qualidade. Neste proposta foram obtidos bons resultados usando um operador de mutação que simplesmente substitui uma máscara, de forma aleatória, para cada descendente produzido.

Estratégia de substituição: A fim de gerar uma nova população foi utilizada a *estratégia elitista* onde uma vez que a população de filhos é gerada, ela é mesclada com a população de pais e segue a seguinte regra: somente os melhores indivíduos presentes nas populações de pais e filhos entram na nova população.

Tamanho da população: A fim de apresentar boa eficiência na pesquisa é importante dimensionar corretamente a população do AG. No entanto, se a população é muito pequena, existe um grande risco de convergência para um ótimo local; nesse caso não se pode manter a diversidade da população que dirige o progresso de um algoritmo genético. Com o aumento do tamanho da população, o AG encontra melhores soluções de forma mais lenta; desta forma é mais difícil para o AG propagar boas combinações de genes e juntá-las ao longo de sua execução [13].

V. Resultados Numéricos

O desenvolvimento e a execução do algoritmo genético proposto foram realizados em Matlab, que facilita a operação com matrizes além de possuir bibliotecas para o desenvolvimento do AG. O computador usado para o trabalho foi um Intel Core 2 Quad 2,66Ghz e 3GB de memória RAM.

A. Conjuntos de informação encontrados

O algoritmo proposto foi usado para gerar conjuntos de informação otimizados para os seguintes códigos: código de bloco binário $C(15, 7)$, código de Golay $C(23, 12)$, e o código de Golay estendido $C(24, 12)$. A Tabela I mostra os resultados obtidos (além do tempo de processamento para o algoritmo guloso).

C(15,7): Para este código foram encontradas 43 máscaras que cobrem 1941 padrões de até 4 erros. O número desejado de máscaras pode ser aumentado e, assim, abranger uma cobertura maior de erros. Contudo quanto menor o número

TABELA I
RESULTADOS PARA OS CÓDIGOS $C(15,7)$, $C(23,12)$ E $C(24,12)$.

Código	C(15,7)	C(23,12)	C(24,12)
Número de conjuntos de informação (N)	43	676	1500
Padrões de erros corrigidos	1941	145499	1391040
Tempo de processamento (GA)	50,4s	2,6h	27h
Tempo de processamento (alg. guloso [10])	1,0s	3,0h	46h

de máscaras, maior é a velocidade de decodificação. Para este código o tempo de processamento foi de 50,4 segundos. Para uma população de $N_I = 100$ indivíduos, o algoritmo alcançou um ótimo global em 408 gerações com a probabilidade de crossover de $p_c = 100\%$. A Figura 4 apresenta a evolução do fitness em função do número de gerações.

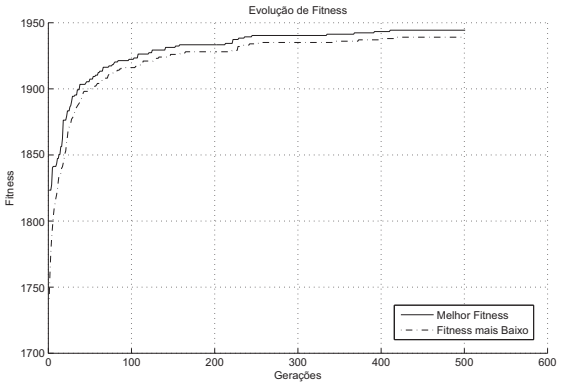

Fig. 4. Desempenho do AG para o código $C(15,7)$.

C(23,12): Para este código foram encontradas 676 máscaras que cobrem 145.499 padrões de até 6 erros. Um tempo maior de busca foi observado com uso algoritmo guloso, especialmente em Matlab, mas com o AG usando a mesma ferramenta, o resultado foi encontrado em 2,6 horas. A probabilidade de crossover foi mantida em $p_c = 100\%$ para uma população de $N_I = 50$ indivíduos, o algoritmo mostrou bons resultados em 51 gerações. Conforme mostra a Figura 5, diferente do caso $C(15,7)$, o AG trouxe uma solução ótima em um menor número de gerações (decorrente da menor dimensão da população).

C(24,12): O desempenho do AG para o código $C(24,12)$ foi relativamente semelhante ao $C(23,12)$. O conjunto de máscaras encontradas para este código totalizou 1500. Com uma população definida de $N_I = 50$ indivíduos, mantendo ainda a probabilidade de crossover $p_c = 100\%$, o algoritmo cobriu um total de 1.391.040 padrões de erros em 51 gerações. Como esperado, houve melhoria no tempo de processamento em relação ao algoritmo guloso. Observa-se que, para códigos pequenos, o algoritmo proposto em [10] utiliza menos tempo que a heurística proposta, porém para códigos mais longos (e de melhor desempenho) o tempo de busca acaba diminuindo.

B. Desempenho de decodificação

Esta seção apresenta o desempenho de decodificação do algoritmo IS considerando os conjuntos de informação

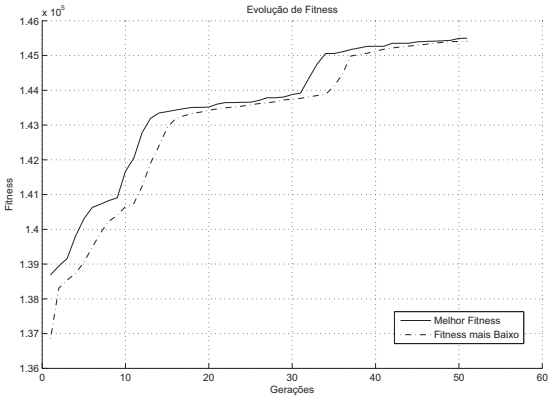

Fig. 5. Desempenho do AG para o código $C(23,12)$.

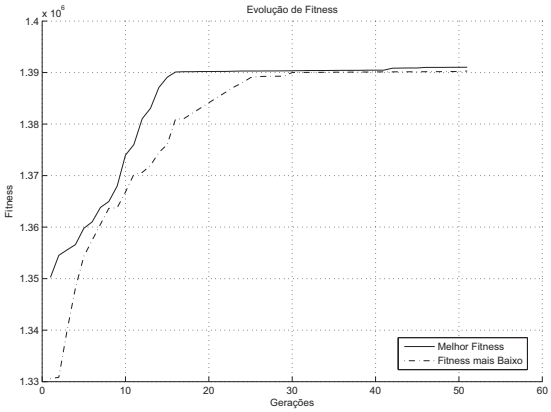

Fig. 6. Desempenho do AG para o código $C(24,12)$.

otimizados obtidos com uso da abordagem proposta neste artigo.

Realizou-se uma simulação computacional considerando sinalização ortogonal binária (BPSK) e demodulação coerente sobre um canal com Ruído Aditivo Gaussiano Branco (AWGN) de média nula. As simulações ocorreram até que a estimativa da taxa de erro de bit (BER) encontrasse uma largura relativa do intervalo de confiança igual a 10% com probabilidade ≥ 0.95. Estimou-se a largura do intervalo de confiança usando a abordagem de Monte Carlo [15], uma técnica padrão para avaliar os resultados simulados.

Interessa para este trabalho valores baixos/moderados para a razão energia de bit pela densidade de ruído (E_b/N_0), pois é sabido que para valores elevados de E_b/N_0 em desempenho de decodificação, para muitos algoritmos, é muito próximo da decodificação de máxima verossimilhança [14]. A Figura 7 mostra o desempenho de decodificação, em termos de BER versus E_b/N_0, para os códigos considerados neste trabalho[1]. A figura mostra também o desempenho para a modulação BPSK

[1]Resultados para o algoritmo MLD não são mostrados na figura, sendo praticamente idênticos ao IS.

sem o uso de codigos corretores de erro.

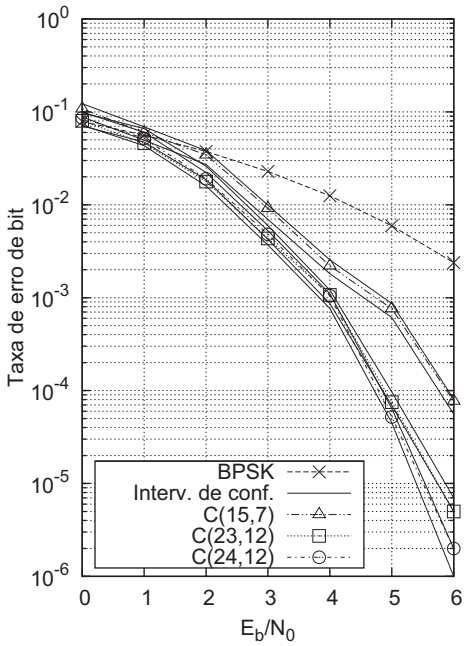

Fig. 7. Desempenho de decodificação.

VI. Conclusão

Este trabalho apresentou uma abordagem para busca de conjuntos de informação através de algoritmos genéticos. Formalmente, a determinação de uma coleção de conjuntos de informação corresponde a um problema de cobertura para o qual não existe um procedimento construtivo para determinar uma coleção otimizada. Com a aplicação dos algoritmos genéticos foram determinados conjuntos de informação com alta capacidade de correção em um curto espaço de tempo.

Para códigos pequenos, o algoritmo proposto em [10] utiliza menos tempo que a heurística proposta, porém para códigos mais longos (e de melhor desempenho) o tempo de busca acaba diminuindo. Os resultados de simulação computacional mostraram que o desempenho do algoritmo conjuntos de informação com as coleções otimizadas é quase idêntico ao desempenho de máxima verossimilhança.

Vale a pena relembrar que o algoritmo de máxima verossimilhança geralmente é impraticável, exigindo tempos de decodificação exagerados, especialmente para códigos longos. Embora o número de conjuntos de informação necessários para cobrir os padrões de erro pareça elevado, em especial para códigos longos, é interessante notar que a primeira metade dos conjuntos de informação encontrados cobrem aproximadamente 90% dos padrões de erro. Desta forma pode-se realizar uma implementação sub-ótima do algoritmo de decodificação, fazendo com que o tempo de decodificação seja modificado em função das restrições que possam se fazer presentes.

Referências

[1] R.H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, John Wiley & Sons, (2002).

[2] H.T. Moorthy, S. Lin e T. Kasami, *Soft-Decision Decoding of Binary Linear Block Codes Based on Iterative Search Algorithm*, In IEEE Transactions on Information Theory, Vol. IT-43, N. 3, 1030-1040, (1997).

[3] M.P.C. Fossorier, S. Lin e J. Snyders, *Reliability-Based Syndrome Decoding of Linear Block Codes*, In IEEE Transactions on Information Theory, Vol. IT-44, N. 1, 388-398, (1998).

[4] D. Chase, *A Class of Algorithms for Decoding Block Codes with Channel Measurement Information*, IEEE Transactions on Information Theory, IT-18, 170-182, (1972).

[5] L.B. Lavitin e C.R.P. Hartmann, *A New Approach to the General Minimum Distance Decoding Problem: The Zero Neighbors Algorithm*, IEEE Transactions on Information Theory, Vol. IT-31, N. 3, (1985).

[6] D.J. Taipale e M B. Pursley, *An improvement to generalized-minimum-distance-decoding*, IEEE Transactions on Information Theory, IT-37, 167-172, (1991).

[7] W. Godoy Jr. e E.C.G. Wille, *Proposal of sub-optimum decoding algorithm with a bound of Voronoi region* $V(c_0)$, In Computer Communications, Vol. 21, 736 -740, (1998).

[8] D.J. Barros, W. Godoy Jr. e E.C.G. Wille, *A new approach to the information set decoding algorithm*, In Computer Communications, Vol. 20, 302-308, (1997).

[9] E. Prange, *The use of information sets in decoding cyclic codes*, In IRE Transactions, Vol. IT-8, S5-S9, (1962).

[10] W. Godoy Jr, E.C.G. Wille e R.P. Jasinski. *A Simple Algorithm for Decoding of Binary Block Codes based on Information Sets*, In International Communications Conference (ICC2010), Cape Town, (2010).

[11] A.H. Chan e R.A. Games, *(n, k, t)-covering systems and error-trapping decoding*, In IEEE Transactions on Information Theory, Vol. IT-27, 643-646, (1981).

[12] J.H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, (1975).

[13] B.A. Julstrom, *Population Size in GAs for TSP*, In Proceedings of 2NWGA, Vaasa, Finlândia, (1996).

[14] J.G. Proakis, *Digital Communications*, 4TH Ed., McGraw-Hill, New York - NY, (2001).

[15] J.M. Hammersley, D.C. Handscomb, *Monte Carlo Methods*, Chapman and Hall, New York - NY, (1964).