UMA NOVA ESTRATÉGIA ARQ-HÍBRIDA PARA PROTOCOLOS EM REDES DE DADOS:
DECODIFICAÇÃO MONTE CARLO PARA CÓDIGOS DE BLOCO

H. MAGALHÃES DE OLIVEIRA e A. LORENA DE OLIVEIRA

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CODEC - Grupo de Comunicações DES/UFPE
C.P. 7800 - 50.711-970 - Recife - PE
Tel.: +(081) 2718210 FAX: +(081) 2718215 e-mail: 69HMO@upd.ufpe.br

Sumário

Este trabalho introduz uma nova classe de algoritmos probabilísticos de decodificação adequada para aplicações em protocolos de redes de dados a qual combina *inerentemente* ***as técnicas de correção FEC e ARQ. O algoritmo é geral, simples, bastante flexível e apresenta arquitetura paralela. Os compromissos entre a complexidade espacial/temporal e o desempenho são avaliados através de simulação empregando a técnica*** *importance sampling****. Resultados para alguns códigos como o BCH (31,15) e resíduos quadráticos QR(47,23) são apresentados.***

1. Introdução e Motivação

No processo de controle de erros em sistemas codificados, existem duas técnicas freqüentemente adotadas: A correção direta de erros (*Forward Error Correction* FEC) e a correção de erros com demanda de retransmissão (*Automatic Repeat Request* ARQ). Obviamente, cada uma delas apresenta vantagens e desvantagens particulares [1]. A técnica ARQ vem sendo extensivamente usada em redes de pacotes e outras redes de dados. Embora a estratégia de correção automática FEC seja bastante atrativa e concentre a maior parte da literatura, a maioria das aplicações práticas envolvendo protocolos de comunicações de dados empregam técnicas com demanda de retransmissão (e.g. protocolos BISYNC da IBM, HDLC CCITT X.25, RJE, SDLC, etc. [2]).
Uma séria desvantagem dos esquemas ARQ é relativa ao rendimento do protocolo (*throuhput efficiency*). A demanda de retransmissão no caso de deteção de erros pode ser substancialmente minimizada através da introdução da possibilidade de correção direta de erros, gerando esquemas híbridos. Assim, estes algoritmos funcionam essencialmente com correção automática FEC para padrões de erros facilmente corrigíveis, os quais ocorrem com elevada freqüência. Já no caso de padrões complexos (cuja ocorrência e mínima), as raras vezes em que o algoritmo mostra-se incapaz de corrigí-los, um erro é detetado e adota-se a estratégia ARQ. Desta forma, a eficiência global do sistema é aumentada sobremaneira [1,3].

Na codificação de canal, um dos maiores desafios consiste em encontrar algoritmos eficientes para a decodificação de códigos longos, i.e., com comprimento de bloco elevado. A decodificação de códigos lineares constitui um problema conjecturado ser de natureza inerentemente intratável; de fato, foi demonstrado [4] que a solução deste é um problema do tipo NP-completo [5]. A decodificação ótima no sentido de minimizar a taxa de erros por palavra ou símbolo-a-símbolo de códigos de bloco pode ser realizada por intermédio dos algoritmos de Bahl *et al*.[6], Hartman-Rudolph [7], ou decodificação por máxima verossimilhança via treliça [8,9]. No entanto, tais algoritmos são praticamente irrealizáveis para códigos de comprimento razoável. Para um código de blocos com parâmetros (n,k) e capacidade de correção t, existem algoritmos [10] cuja complexidade é certamente inferior a MIN $(k2^k,(n-k)2^{n-k})$.

Formalmente, supõe-se um decodificador modelado por L máquinas seqüenciais interconectadas: $\underline{S}=(S_1,S_2,...,S_L)$. Cada uma possui uma complexidade espacial $\underline{X}=(X_1,X_2,...,X_L)$ e uma complexidade temporal $\underline{T}=(T_1,T_2,...,T_L)$. O trabalho de cálculo W é definido pelo produto escalar desses vetores, i.e., $W:=\Sigma\ X_iT_i$. Globalmente, a complexidade no espaço é $X:=\Sigma\ X_i$ e a complexidade no tempo é $T:=MAX\ (T_i)$. O esforço computacional pode ser definido como o número máximo de operações elementares necessárias para efetuar a decodificação de uma única palavra. Sabe-se que um problema NP-completo pode ser resolvido em tempo polinomial somente se NP=P. Pode-se evitar esforços "inúteis" procurando algoritmos deste tipo. Neste caso, buscam-se, sejam algoritmos sub-ótimos, sejam meios probabilísticos - onde a complexidade é grande apenas no pior dos casos, mas muito pequena em quase todos os outros. Normalmente, a facilidade na decodificação é conseguida às custas de uma degradação aceitável no desempenho. Ainda que uma enorme quantidade de algoritmos determinísticos tenham sido propostos [10-14], é possível procurar soluções alternativas, considerando-se algoritmos probabilísticos. Em tais algoritmos (apenas os algoritmos determinísticos são do tipo não-polinomial NP!), o esforço computacional passa a ser variável, o que pode representar um inconveniente prático: o tempo de decodificação de uma palavra torna-se uma variável aleatória com

variância infinita, como é uma característica típica de algoritmos não-determinísticos (e.g., decodificação seqüencial para códigos convolucionais). Entretanto, impondo-se uma restrição adicional sobre a máxima complexidade de decodificação por palavra recebida, o decodificador pode tornar-se bastante atrativo e prático, as custas de uma degradação no desempenho.

O método de Monte Carlo, termo cunhado por *Von Neumann*, é usado para referir-se a geração de diferentes situações amostrais visando a resolução de problemas de díficil solução analítica ou de grande complexidade computacional e estudar o desempenho de procedimentos. É fato conhecido que as simulações em computador são técnicas poderosoas na abordagem de problemas complexos [15]. A idéia aqui é construir um decodificador baseado em técnicas de simulação, visando minimizar o esforço computacional médio, gerando um algoritmo cuja complexidade é bem inferior àquela do decodificador ótimo. Este trabalho destina-se à análise de uma nova classe de métodos de decodificação de códigos de blocos lineares binários os quais usam informação de natureza probabilística (canal de medida de informação) e são baseados no emprego de técnicas de Monte Carlo. Esta nova classe de algoritmos probabilísticos de decodificação envolve esquemas ARQ-Híbridos (tipo I ou tipo II) em protocolos de comunicações de dados. Trata-se de um novo processo de decodificação no qual a técnica FEC combina-se inerentemente com a estratégia ARQ, num sinergismo remarcável. O princípio do algoritmo lembra um método de otimização por gradientes estocásticos [15], simulando aleatoriamente padrões de erros mais prováveis segundo uma distribuição de probabilidade construída a partir de coeficientes de confiabilidade dos bits recebidos. Trata-se de um algoritmo não-determinístico que usa também informação probabilística (*decisão suave*). O decodificador introduzido é bastante simples e flexível, permitindo inúmeros compromissos entre a complexidade espacial ou temporal e o desempenho. Sua arquitetura é inerentemente paralela, fato que o torna atrativo em implementações práticas. Resultados de simulação de sistemas codificados são apresentados, possibilitando uma avaliação criteriosa dos compromissos entre desempenho e complexidade.

2. O algoritmo de decodificação Monte-Carlo

A estratégia adotada é uma solução inteligente: Os padrões de erros facilmente corrigíveis (os mais freqüentes) são corrigidos, enquanto que os padrões complicados (de rara ocorrência) levam o decodificador à solicitar retransmissão. A idéia essencial consiste em considerar "*soluções candidatas*" com base na geração de uma "vizinhanca estocástica" da palavra ruidosa recebida, se ela contém erros. Isto significa que há uma distribuição de probabilidade sobre todas as énuplas, concentrada em torno da palavra recebida e polarizada segundo informações fornecidas por um vetor de confiabilidades. Trata-se de uma ordenação dos padrões de erros, segundo a informação (*soft decision*) do canal. O procedimento de busca fundamenta-se no emprego de coeficientes de confiança condicional de Kiefer [16] como medida de confiabilidade dos bits recebidos. Em todo o processo são usados apenas geradores de números aleatórios com distribuição uniforme! A verificação se uma solução candidata deve ser adotada é feita através do teste da síndrome [1]. O peso de Hamming de uma énupla é o número de posições não nulas.

2.1 Versão Preliminar:

O algoritmo em sua versão simplificada consiste resumidamente em:

P1- Calcular o vetor recebido via decisão abrupta. Avaliar sua síndrome. Se a síndrome é nula, admite-se que não houve erro, senão acumular o número de tentativas processadas.

P2- Estimar os coeficientes de confiabilidade de cada dígito recebido. Atualizar os acumuladores de taxa de erro, probabilidade de sobrecarga e número médio de tentativas, respec., $P_E/P_{ovf}/E(ntry)$.

P3- Simular aleatoriamente um padrão de erros (e.g. peso de Hamming $\leq t$), para obter uma "solução candidata", segundo uma distribuição de probabilidade determinada pelos coeficientes de confiabilidade, gerando uma espécie de "gradiente estocástico".

P4- Se o número de soluções testadas excede um valor máximo previamente admitido (limite na complexidade), então sinalizar uma deteção de erro (*flag*), ativando a demanda de retransmissão. Incrementar o contador de sobrecarga (*overflow*) e retornar a P1.

P5- Recalcular a síndrome s para a palavra candidata. Se $s \neq 0$, retornar a P3, senão considerar concluída a decodificação, escolhendo a solução atual como palavra decodificada e retornar a P1.

Assim, se um padrão de erros não for detetado dentro de uma complexidade pré-estabelecida, o receptor rejeita a palavra recebida, sinaliza erro e requisita uma retransmissão. Usa-se um procedimento inteligente de tentativa-e-erro (Monte Carlo), empregando apenas a geração de números aleatórios com distribuição uniforme. Uma característica típica do tempo necessário para decodificar uma palavra é que sua variância pode ser infinita, tal como ocorre na decodificação seqüencial. Aparentemente a variável aleatória correspondente ao tempo para decodificar uma palavra segue uma distribuição de Pareto. A introdução de um número máximo de ensaios de palavras candidatas, *nsim*, além de limitar a complexidade máxima, aparece como uma solução natural em esquemas híbridos.

Denotando-se por V_n o espaço vetorial das énuplas binárias e d_H a distância de Hamming entre duas énuplas (o número de posições em que elas diferem), dada uma palavra recebida, define-se:

Definição: Diz-se que um conjunto de énuplas $B_t(\underline{r}):=\{\underline{x}\in V_n \ni d_H(\underline{x},\underline{r})\le t\}$ geradas aleatoriamente segundo uma distribuição $\{P(\underline{x}), \underline{x}\in B_t(\underline{r})\}$ constitui uma vizinhança estocástica de $\underline{r}$, se e somente se a probabilidade dos eventos $\Delta B_m(\underline{r}):=B_m(\underline{r})-B_{m-1}(\underline{r})$, $1<m\le t$ decresce monotonicamente com m.

A distribuição sobre a vizinhança $\{P(x)\}$, calculada a partir dos coeficientes de confiabilidade, "deforma" a vizinhança, indicando direções onde verossimilmente encontra-se a palavra código mais próxima. A maneira de gerar uma vizinhança da palavra recebida com base nas informações de verossimilhança resulta numa ordenação natural das configurações/padrões mais prováveis de erros. Isto é realizado com auxílio de um vetor de limiares *T*.

2.2 Estabelecimento dos limiares:

Dado o vetor de confiabilidades condicionais $\underline{R}=(R_1,R_2,...,R_n)$, $R_i\ge 0{,}5$, associado a um vetor recebido $\underline{r}=(r_1,r_2,...,r_n)$, consi-dera-se o seu vetor complementar $\underline{\Gamma}=(\Gamma_1,\Gamma_2,...,\Gamma_n)$, com $\Gamma_i:=1-R_i\le 0{,}5$. Os limiares normalizados $\underline{T}:=(T_1,T_2,...,T_n)$ são calculados de acordo com a relação:

$$T_i := T_{i-1} + \frac{\Gamma_i}{\sum_{k=1}^{n}\Gamma_k} \quad i=1,2,...,n \ \text{com}\ T_0 := 0. \qquad (1)$$

Simula-se uma variável aleatória z com distribuição uniforme U(0,1) e verifica-se em que intervalo a amostra caiu; isto corresponde à posição do padrão de erro a testar, i.e., Dado z, $\exists!j \ni T_j<z\le T_{j+1}$. Assuma que o padrão de erros a testar possui erro na j-ésima posição. Escolhido um j particular, as informações probabilísticas são atualizadas excluindo-se a j-ésima posição do vetor de confiabilidades. Isto é feito diretamente assumindo $R_j=1$ ($\Gamma_j=0$) e recalculando-se um novo vetor normalizado de limiares $\underline{T}$, no qual $T_j=0$. Encontra-se a palavra candidata corrigindo-se o padrão de erros e efetua-se o teste da síndrome. Após no máximo *n* passos, todos os padrões de peso de Hamming unitário foram testados em ordem decrescente de probabilidade de ocorrência. O número médio de passos é muito inferior ao valor máximo *n*. Se o teste envolve um padrão de erro com peso de Hamming superior à unidade, diversas variáveis aleatórias são geradas até compor um erro com o peso desejado.

Arquitetura paralela:

L máquinas são colocadas em paralelo, cada executando o algoritmo a partir de geradores de números aleatórios independentes. Quando uma delas é hábil para decodificar, envia uma sinal de *reset* para todas as demais. Assim, o compromisso entre a complexidade temporal/espacial é estabelecido de forma simples.

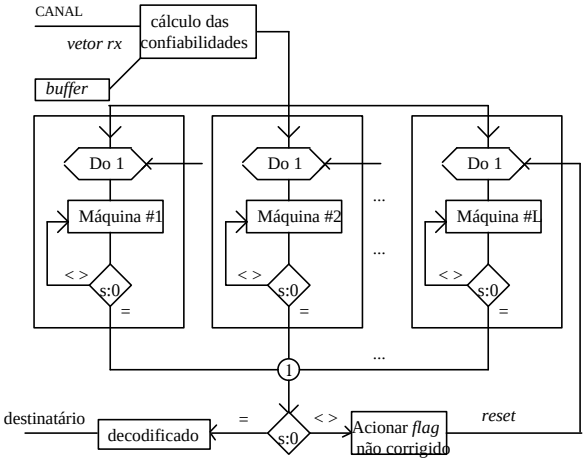

Figura 1: Implementação do algoritmo de decodificação Monte Carlo em Arquitetura paralela.

3. Resultados de simulações: Análise de desempenho e Complexidade

Algumas famílias de códigos algébricos são examinadas, como os códigos de resíduo quadrático e BCH. Numa análise preliminar, os parâmetros dos códigos testados foram BCH(31,15) e QR(47,23). Os códigos escolhidos apresentam taxas próximas a 1/2, região onde a complexidade dos decodificadores é máxima [8]. Com relação à simulação, para avaliar o desempenho do decodificador, transmite-se apenas a énupla toda nula 000...0, pois os códigos empregados são lineares. O requisito sobre a linearidade do código advém da exigência do teste da síndrome. Códigos cíclicos CRC (*Cyclic Redundancy Checking*) são ainda mais atrativos, por permitir um cálculo rápido da síndrome. Supondo um canal com ruído aditivo Gaussiano de potência média σ^2 e sinais polares ($\pm$A), o coeficiente de confiança condicional de Kiefer [16] da i-ésima amostra r_i, é expresso por

$$R_i=[1+\exp(2|r_i|A/\sigma^2)]^{-1}. \qquad (2)$$

Uma nova variável é introduzida: A probabilidade de sobrecarga no algoritmo, P_{ovf}. Claro que $P_{ovf}\to 0$ quando o número de tentativas é ilimitado. A medida que o número de tentativas permitidas cresce, a palavra decodificada tende *quase certamente* para a palavra código mais próxima da palavra recebida. O tempo necessário para adquirir uma nova palavra deve ser suficiente para realizar a simulação do número máximo de padrões a serem testados (decodificação em tempo real), estabelecendo um tempo máximo para a decodificação. O algoritmo falha na decodificação (*ndc* :=não decodificar corretamente) se este tempo não for suficiente para decodificar ou se houve um erro na decodificacão: $P(ndc):=P_E+P_{ovf}$. Se o número de tentativas ultrapassar o número máximo pré-estabelecido, o sistema deteta erro e requisita retransmissão, entrando no modo ARQ (*Stopt-and-wait*, *Go-Back-N*, *Selective repeat*...). A cota sobre o número de soluções candidatas a testar *nsim* é um limitante óbvio na complexidade do algoritmo.

O algoritmo foi implementado em linguagem Pascal, permitindo a análise do compromisso entre o desempenho e a complexidade do decodificador. Os resultados são expressos através de curvas de Probabilidade de erro por bloco *versus* relação sinal/ruído (SNR) do canal, para cada esquema de codificação. As curvas são validadas calculando-se os intervalos de confianca a 95% para os pontos obtidos na simulação [17]. As convergências são garantidas pela versão forte da lei dos grandes números.

A simulação Monte Carlo convencional é eficiente e viável na estimativa da taxa de erros para baixas relações sinal/ruído. Além de situações práticas exigirem taxas de erro substancialmente menores, as informações dos coeficientes de confiança em baixa SNR são freqüentemente menos representativas, dificultando o procedimento de decodificação. Já para valores moderados ou elevados de SNR, as técnicas usuais conduzem a um tempo de CPU proibitivo. Nos casos de interesse prático, a aplicabilidade do algoritmo se dá exatamente nesta faixa, fazendo-se necessário recorrer-se a *importance sampling* IS [18]. A ineficiência do método clássico provém do fato de que uma grande quantidade de amostras geradas não produzem erro (que é o evento importante). No IS, o o evento importante é gerado artificialmente por meio de uma distorção deliberada, chamada de polarização. No final da simulação, o contador de erros deve ser devidamente despolarizado. No caso mais comum, o que é feito é gerar artificialmente amostras numa relação sinal-ruído mais baixa (aumento da variância), onde aparecem mais erros. Uma outra estratégia mais eficiente, o IS melhorado (IIS), consiste em deslocar a densidade de probabilidade do ruído [19]. O canal gaussiano foi simulado com SNRs na faixa de 10 a 16dB empregando IIS. Os padrões de erros mais freqüentes são corrigidos com enorme rapidez, enquanto que os improváveis (com informações de confiabilidade menos representativas) exigem maior tempo ou acionam a demanda de retransmissão. Os valores do número médio de tentativas necessárias para decodificar uma palavra recebida, E(*ntry*), indicam a eficiência do método. Concluída a simulação, avaliam-se a probabilidade de erro, a probabilidade de sobrecarga no algoritmo (*overflow*) e a probabilidade de não decodificar corretamente, empregando estimadores de freqüência relativa.

Apresenta-se um estudo da sensibilidade do algoritmo em função da SNR, comprimento do bloco do CRC, capacidade de correção do código etc. Verifica-se o ganho obtido em desempenho (diminuição na taxa de erros) em termos do aumento de complexidade (*nsim*). Numa região de interesse, para 16dB, têm-se E(*ntry*)=2,46 (resp. 4,27), fixada a complexidade escolhendo-se *nsim*=500 (resp. 5.000). Uma relação típica entre o número médio de tentativas para decodificar um bloco E(*ntry*) e a relação sinal/ruído é $E(ntry)=cte\,SNR^{\gamma}$, e.g., $E(ntry)=10^{4,17}.SNR^{-2,33}$, para *nsim*=500 (Regressão linear com $r^2=0,992$,s=0,0093), com o código BCH. (vide Apêndice A).

4. Discussão e conclusões

O problema da decodificação de códigos lineares é inerentemente não tratável (pertence à classe NP-completa), i.e., conjectura-se a inexistência de algorit-mos determinísticos em tempo polinomial. Uma solução alternativa consiste em procurar algoritmos sub-ótimos e/ou algoritmos probabilísticos, cuja complexidade é polinomial em média estatística, tal como a classe de algoritmos aqui introduzidos. Algumas variantes podem ser propostas: Fixar diferentes valores para o número máximo de padrões testados em função do peso do vetor de erro; implementar a correção de padrões de 1,2,...,t erros em série; ou em paralelo; escolha de uma distribuição de probabilidade sobre a ocorrência de padrões com até t erros; etc. O custo para introduzir/adaptar esta técnica em esquemas ARQ convencionais existentes é muito baixo. A complexidade espacial e temporal é fixada pelo usuário/gerenciador da rede.

Há uma interessante interpretação geométrica para o procedimento de decodificação: Dada a palavra recebida ruidosa, o processo consiste em "atirar" num alvo de centro r e raio t de acordo com uma distribuição obtida pelas confiabilidades, até "acertar" numa palavra código. O algoritmo introduzido não tem sua capacidade de correção limitada a t erros, i.e., o procedimento não é limitado à capacidade algébrica e de um modo geral, podem ser eventualmente testados e corrigidos padrões até o raio de cobertura do código. O preço a pagar é novamente um aumento na complexidade. Uma vantagem adicional decorrente do emprego dos coeficientes de Kiefer é que o algoritmo permite a decodificação de informação proveniente de fontes com distribuição *a prior*i desconhecida [20]. A classe de algoritmos introduzida exige baixa complexidade computacional e tem aplicabilidade a qualquer código de bloco linear. O decodificador é extrememente flexível e apresenta um excelente compromisso complexidade × desempenho. A maior atratividade é o fato que o procedimento pode ser controlado adaptativamente, considerando o projeto de um esquema ARQ Híbrido adaptativo, cuja capacidade de correção varia em função das condições do canal e do tráfego. A capacidade de correção pode tornar-se adaptativa com o fluxo de informações no canal e/ou a taxa de erros, simplesmente variando-se o CRC usado na proteção, pois o algoritmo não é específico para nenhum código. O decodificador pode também ser melhorado através de um processo de aprendizado (*unsupervised learning*), onde o número de tentativas realizadas com insucesso (sem encontrar um padrão de erros aceitável) fornece informação sobre o peso do vetor de erro que ocorreu no canal. Além de desenvolver novas variantes para o decodificador, resta realizar um estudo comparativo da sensibilidade e eficiência destas diversas variantes em função da SNR, comprimento do bloco do CRC, capacidade de correção do código etc. Estes esquemas ARQ-Híbridos são recomendados principalmente em protocolos envolvendo redes de dados com altas taxas (Mbits/s), onde o ARQ convencional é inaceitável.

Apêndice A

Alguns resultados de simulação são apresentados na figura 2 através das curvas de desempenho do algoritmo para o código BCH (31,15). A curva P_{uncod} corresponde Probabilidade de erro no canal sem empregar a codificação, enquanto que as demais curvas exibem os valores da Probabilidade de falha no algoritmo (erro + sobrecarga) fixado o número máximo de tentativas permitido, *nsim*=500, 900, 1500, 5000 e $+\infty$. A curva de melhor desempenho corresponde exatamente à Probabilidade de erro por bloco obtida empregando um algoritmo de decodificação algébrica (Belekamp-Massey).

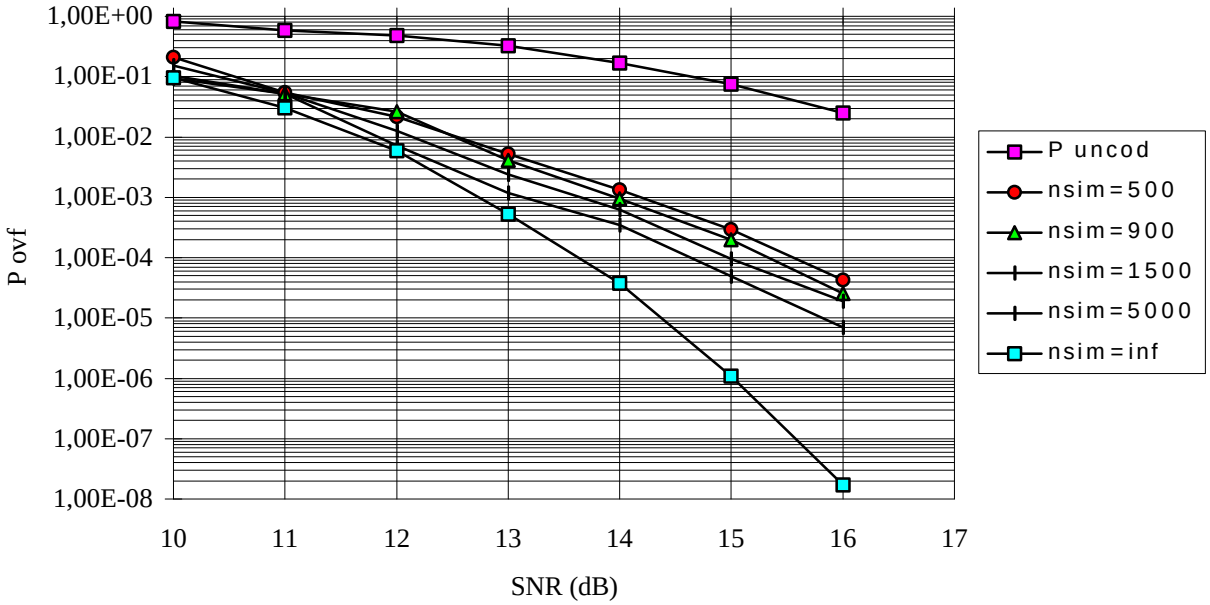

Figura 2. Probabilidade de não decodificar corretamente no algoritmo para vários níveis de complexidade.

As curvas da Figura 3 exibem os valores do número médio de tentativas (complexidade média) necessárias para decodificar uma palavra recebida do código BCH para diferentes valores de complexidade máxima do decodificador, usando o algoritmo Monte Carlo. Resultados para o código QR(47,23) foram qualitativamente similares.

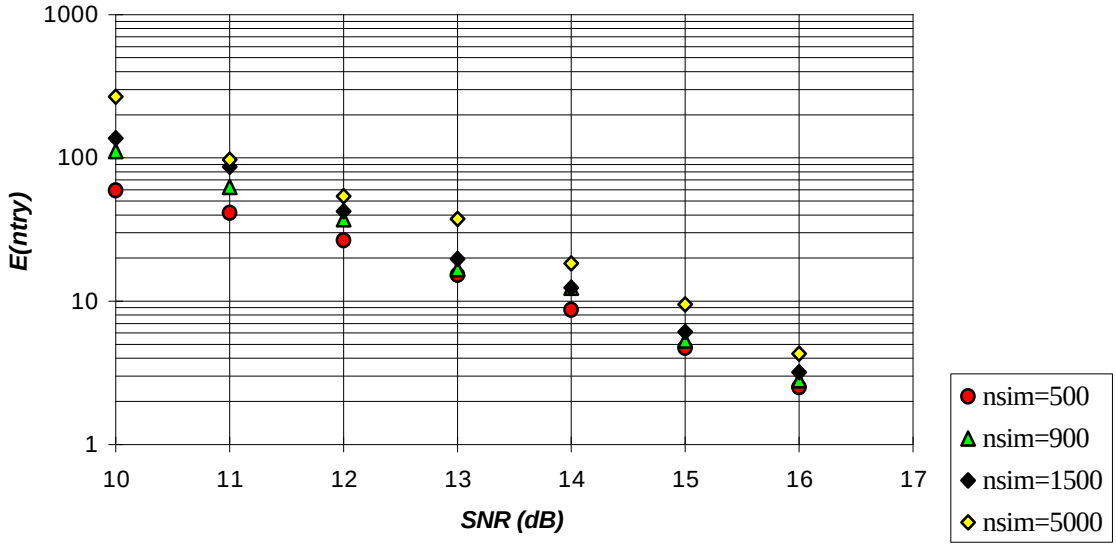

Figura 3. Número médio de tentativas para decodificar uma palavra recebida usando o algoritmo.

Agradecimentos

Este trabalho recebeu apoio parcial do *Conselho Nacional de Desenvolvimento Científico e Tecnológico* - CNPq.

Referências

[1] S. Lin *et al.*, "Automatic-Repeat-Request Error-Control Schemes", *IEEE Comm. Mag.*, **22**, p.5-17 Dec., 1984.

[2] H.M. de Lima e O.C.B. Duarte, "Um protocolo de retransmissão contínua para comunicação de dados a altas velocidades", *X Simpósio Brasileiro de Telecomunicações*, Brasília, pp.439-444, 1992.

[3] S.B. Wicker, "Adaptive Rate Coding Using Reed-Solomon Codes in a Hibrid-ARQ Protocol", ISSSE'89, *Int. Symp. on Signals, Systems and Electronics*, Erlang, FRG, pp.642-645, 1989.

[4] E.R. Berlekamp *et al.*, "On the Inherent Intractability of Certain Coding Problems", *IEEE Trans. Info. Theory*, **IT 24**, pp.384-386, May, 1978.

[5] M.R. Garey and D.S. Johnson, "*Computers and Intractability: A guide to the Theory of NP-Completeness*", San Francisco: W.H. Freeman & Co. 1979.

[6] L.R. Bahl *et al.*, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Trans. Info. Theory*, **IT 20**, pp.284-287, March, 1974.

[7] C.R.P. Hartmann and L.D. Rudolph, "An Optimum Symbol-by-Symbol Decoding rule for Linear Codes", *IEEE Trans. Info. Theory*, **IT 22**, pp.514-517, 1976.

[8] J.K. Wolf, "Efficient Maximum Likelihood Decoding of Linear Codes Using a Trellis", *IEEE Trans. Info. Theory*, **IT 24**, pp.76-80, Jan., 1978.

[9] G. Battail, "Décodage pondéré optimal des codes linéaires en blocs I. Emploi simplifié du diagramme du treillis" ,*Ann. Télécomm.*,**38**,n.11-12,pp.443-459, 1983.

[10] D.J. Muder, "Minimal Trellises for Block Codes", *IEEE Trans. Info. Theory*, **IT 34**, pp.1049-1053, Sept., 1988.

[11] A. Chase, "A Class of Algorithms for decoding Block Codes with Channel Measurement Information", *IEEE Trans. Info. Theory*, **IT 18**, pp.170-181, Jan., 1972.

[12] W. Godoy Jr. and D.S. Arantes, "Sub-optimum Soft-Decision Decoding of Block Codes Using the Zero-Neighbors Algorithm", *IEEE Int. Symp. on Info. Theory*, ISIT, Kobe, Japan, 1988.

[13] G. Battail, "We can Think of Good Codes and even Decode them", *Int. Coll. on Coding Theory and Applications*, Invited Conference at EUROCODE, Udine, Italy, Nov.,1992.

[14] D.J. de Barros *et al.*, "Um novo Limiar de parada na Decodificação", *XI Simpósio Brasileiro de Telecomunicações*, Set., Natal, RN, 1993.

[15] R.Y. Rubinstein, "*Simulation and the Monte Carlo Method*", NY: Wiley, 1981.

[16] J. Kiefer, "Conditional Confidence Statements and Confidence Estimators", *J. of The Amer. Statistical Assoc.*, JASA **72**,pp.789-827, Dec.,1977.

[17] T.H. Wonnacott and R.J. Wonnacott, "*Introdução à Estatística*", Rio de Janeiro: LTC,1980.

[18] M. Jeruchin, "Techniques for Estimating the Bit Error Rate in the Sinulation of Digital Communication Systems" , *IEEE J. Select. Areas in Comm.*, **SAC 2**,pp.153-170,Jan.,1984.

[19] D. Lu and K. Yao, "Improved Importance Sampling Technique for Efficient Simulation of Digiatl Communication Systems", *IEEE J. Select. Areas in Comm.*, **SAC 11**,pp.67-75,Aug, 1993.

[20] H.M. de Oliveira, "Empirical Bayes Adaptive Decoding for Sources with unknown Distribution", *IEEE Int. Symp. Info. Theory*, ISIT, p.1, Ann Arbor, Michigan,Oct., 1986.