UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA DE CAMPINAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CÓDIGOS BALANCEADOS PARA TRANSMISSÃO DIGITAL POR FIBRA ÓPTICA

JOÃO MARÇOS TRAVASSOS ROMANO
Orientador: HELIO WALDMAN

Este exemplar corresponde à redação final da Tese defendida por João Marcos Travassos Romano e aprovada pela Comissão Julgadora em 27/06/1984.

Helio Waldman

Tese apresentada à Faculdade de Engenharia da Universidade Estadual de Campinas - UNICAMP - como parte dos requisitos exigidos para a obtenção do título de MESTRE EM CIÊNCIAS.

Aos meus pais,
à Maria Inês e
à Miriam,
com carinho.

PRÓLOGO

Este trabalho originou-se devido à conveniência de se fazer um estudo aprofundado sobre codificação de linha, tendo em vista a aplicação num futuro sistema de transmissão digital por fibra óptica em 140 Mbits/s.

Nossa idéia inicial era estudar tanto os códigos balanceados como os não-balanceados, analisando aspectos tais como sincronizabilidade, recuperação de relógio, complexidade e conformação espectral.

Das consultas bibliográficas, podia-se perceber que as classes de código mais apropriadas para a taxa de transmissão desejada eram as do 5B-6B e do 7B-8B. Essa escolha é feita em função da complexidade e excesso de velocidade requeridas pelo sistema. Não havia porém um critério bem estabelecido para, dentro de uma dada classe, decidir-se sobre uma determinada tabela de codificação.

Começamos então a analisar melhor os dois códigos citados, procurando uma tabela de codificação que otimizasse a sincronizabilidade do sistema. Esse estudo, que contitui o capítulo III da tese, ganhou uma extensão razoável quando passamos a pesquisar a sincronização por ocorrência de violações na lei do código, criando um modelamento que reduzia o problema a um processo de Markov. Por causa disso, fomos abandonando a idéia de estudar também os códigos não-balanceados.

Através dos programas desenvolvidos sobre a teoria do capítulo III, pode-se testar a sincronizabilidade de um código (n-1)B-nB, seja por violação na lei do código, como por ocorrência de palavras proibidas. No capítulo IV mostramos alguns resultados obtidos, que nos permitem sugerir algumas tabelas consideradas boas, dando-se mais ênfase ao código 5B-6B, pois foi o que nos pareceu mais indicado para a transmissão em 140 mb/s. Os programas podem ser utilizadas também para códi -

gos mB-nB, onde $m \neq n-1$, desde que sejam feitas algumas modificações pouco relevantes.

A leitura do artigo de Rousseau [14] despertou a idéia de um estudo teórico mais profundo sobre a utilidade de um código mB-nB, de acordo com os compromissos impostos sobre a complexidade, o balanceamento e o excesso de velocidade. Esse estudo, apresentado no capítulo II, acabou por produzir um resultado bastante original, fornecendo uma base teórica mais elegante para o nosso trabalho.

Por fim, resta-me dizer que ao longo de um trabalho de tese, dependemos da colaboração de muitas pessoas, de forma que me reconheço incapaz de esboçar aqui um agradecimento suficientemente justo, abrangendo a todos que participaram com seu incentivo e amizade. Existem porém alguns que tiveram uma influência direta, aos quais me cabe agradecer especialmente.

Primeiramente, ao Prof. Helio Waldman, pela oportunidade de realizar esta tese sob sua orientação, extremamente dedicada e criativa.

Também ao amigo Engº Paulo T. Hosoe, pelo acompanhamento desse trabalho, principalmente em sua fase inicial e aos demais companheiros do grupo de Comunicações Ópticas, pela ajuda e sugestões.

À Secretária Olga R. Morales e aos desenhistas Luís Pasquini e Édson Lima, pela competência e boa vontade demonstradas nos trabalhos de datilografia e desenho, feitos em condições um tanto urgentes.

Ao amigo Edson De Pieri e outros que ajudaram nos problemas computacionais e a todos os professores e colegas do DEE que se mostraram receptivos a dúvidas e discussões, colaborando não só com esse trabalho mas com a própria sobrevivência, em meio a tantas dificuldades, do verdadeiro espírito universitário.

Campinas, Junho de 1984.

ÍNDICE

CAPÍTULO I - INTRODUÇÃO AO PROBLEMA DA CODIFICAÇÃO DE LINHA .. 001

CAPÍTULO II - CÓDIGOS BALANCEADOS ÚTEIS 007

II.1 - Introdução 007

II.2 - O Balanceamento 008

II.3 - A Estrutura dos Códigos mB-nB 010

II.4 - O Máximo Desbalanço P e o Mínimo Excesso de Velocidade 012

II.5 - Limites Fundamentais de Eficiência 018

II.6 - As Tabelas de Cálculo de R (n,d_{max},D) 023

II.7 - Os Códigos Úteis e sua Variedade 052

II.8 - Conclusão 054

APÊNDICE II.A - LIMITES DE EFICIÊNCIA PREVISTOS PELA TEORIA DA INFORMAÇÃO 057

CAPÍTULO III - SINCRONIZABILIDADE DE BLOCOS EM CÓDIGOS mB-nB . 060

III.1 - Introdução 060

III.2 - Sincronização por Palavras Proibidas 062

III.2.a - Cálculos de $P_V(s=1)$ 062

III.2.b - Modelo Geral de Obtenção de $P_V(s)$ 071

III.3 - Sincronização por Violação da Lei do Código . 077

III.3.a - Diagramas de Deteção de Transições Proibidas 077

III.3.b - Fluxogramas de Estados de Observação 081

III.3.c - Fluxogramas de Macro-Estados 092

III.4 - Violação devido a erros de linha 096

III.5 - Os Canais de Serviço 098

APÊNDICE III.A - O PROGRAMA CODJM.FOT 102

APÊNDICE III.B - FLUXOGRAMAS DE ESTADOS DE OBSERVAÇÃO PARA DEFASAGENS s > 1 109

APÊNDICE III.C - O PROGRAMA BOLJM.FOT 121

CAPÍTULO IV - CONSIDERAÇÕES FINAIS 132

IV.1 - Resultados Obtidos sobre a Sincronizabilidade. 132

IV.2 - Os Tempos de Retenção e de Recuperação 140

IV.3 - Conclusão 141

BIBLIOGRAFIA 143

CAPÍTULO I

INTRODUÇÃO AO PROBLEMA DA CODIFICAÇÃO DE LINHA

Num sistema de codificação digital em banda base, seja via cabo metálico ou fibra óptica, é comum se fazer um processo de codificação do sinal digital gerado pela fonte, antes dele ser enviado à linha de transmissão.

Esta codificação se faz necessária pois o sinal na saída da fonte não possui certas características convenientes ao sistema de transmissão. Dentre elas podemos citar a conformação espectral e a facilidade apresentada para a recuperação de relógio nos regeneradores.

Antes de qualquer codificação, a sequência gerada possui um conteúdo espectral significativo nas baixas frequências, isto não é recomendável devido ao acoplamento A.C. que, em sistemas via cabo, é efetuado por transformadores dispostos entre a linha e os repetidores. Em receptores ópticos, a presença de capacitores no estágio detector e entre os estágios amplificadores também provocam esse corte nas baixas frequências, que ocorre em cerca de 0,1 a 1% da taxa de transmissão.

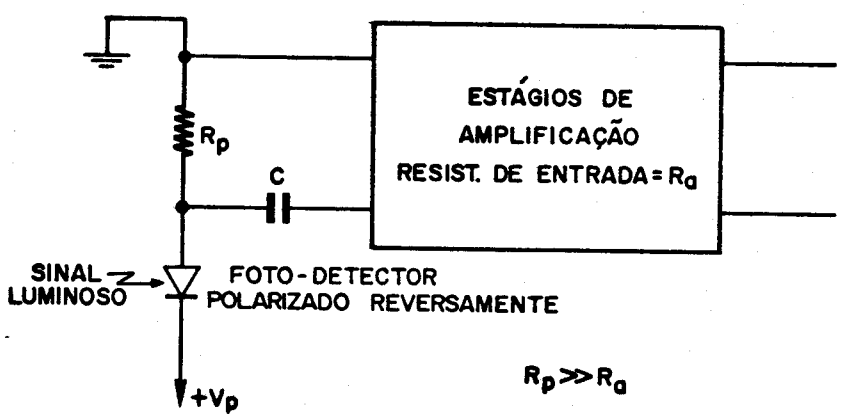


Fig. I.1 - Esquema simples de um conjunto detector-amplificador utilizando diodo PIN.

O acoplamento A.C. causa o corte do conteúdo em torno de f=0, dando origem às flutuações do nível de base ("baseline wander ou "d.c. wander") que, sobrepondo-se ao sinal nos circuitos equalizadores, reduzem a margem contra ruído na detecção dos bits. Para minimizar este fenômeno, deve-se codificar o sinal de modo a dar-lhe uma conformação espectral adequada, ou seja, com pouco conteúdo nas baixas.

A recuperação de relógio é feita através da própria sequência transmitida. Se o seu espectro possuir uma raia em $1/T_0$, onde T_0 é a taxa de símbolos, basta extrair esta senóide, utilizando-se um filtro de faixa suficientemente estreita e, a partir dela, obter a onda de relógio. Caso o sinal não apresente raia na frequência da taxa de símbolos, ela poderá ser criada através de um processamento não-linear.

Entretanto, este procedimento é prejudicado se houver longas sequências de dígitos nulos o que, no caso de um sinal de voz, por exemplo, é bastante possível. Necessita-se portanto, de uma codificação que garanta uma densidade de transições de pulso suficientemente alta.

Quando a transmissão se faz por cabo metálico, esses problemas são em geral solucionados pelo uso de códigos pseudo-ternários tais como o AMI, que possui um espectro com pouco conteúdo nas baixas frequências, e o HDB-3, que segue a regra do AMI com algumas modificações, no sentido de evitar longas sequências de zero na linha. Existem ainda outras soluções como o emprego de sistemas de resposta parcial, códigos de bloco com o MS-43, etc..

Em geral, a codificação introduz uma redundância no sinal a ser transmitido; essa redundância pode ser, posteriormente, utilizada para veicular informações adicionais.

Para a introdução dessa redundância, teremos que aumentar o número de níveis do sinal a ser transmitido ou elevar a sua taxa de símbolos na linha. Haverá também uma deterioração da margem de ruído nos regeneradores e uma maior complexidade no circuito devido ao aparecimento do conjunto codificador-decodi-

ficador, embora devamos notar que esta complexidade aparecerá apenas nos terminais transmissor e receptor e não nos repetidores.

Além das vantagens já mencionadas da codificação de linha, a existência de redundância no sinal irá nos permitir ainda uma monitoração do desempenho do sistema através de medidas da taxa de erros de linha, a possibilidade de se obter canais de serviço e a capacidade do sistema auto sincronizar-se.

Em se tratando de comunicações ópticas, algumas particularidades nos levam a adotar soluções diferentes das já citadas para transmissão via cabo.

De imediato, notamos que códigos como o AMI ou o HDB-3 , assim como sistemas de resposta parcial, não podem ser utilizados, pois necessitam de presença de pulsos negativos e positivos na linha, o que não é possível para o caso de sinal luminoso modulado em intensidade.

A princípio, poderíamos pensar no uso de códigos multiníveis, evitando, dessa forma, um aumento na taxa de símbolos transmitida. Isto seria conveniente pois a fibra óptica, como todo canal real de transmissão, apresenta atenuação e dispersão dos pulsos de energia, as quais serão fatores limitantes à velocidade do sistema. Cabe-nos porém dizer que esta limitação é consideravelmente inferior à que verificamos em canais metálicos, o que torna a comunicação por fibra óptica mais atrativa para sistemas de maior capacidade.

A atenuação na fibra foi sensivelmente reduzida em pouco espaço de tempo, passando de cerca da 1000 dB/km em 1965 a cerca de 2 dB/km para λ = 0,85 nm e 0,3 dB/km para λ = 1,3 nm em 1980. As perdas principais se dão devido aos fenômenos de absorção (conversão de energia eletromagnética em calor) e espalhamento (conversão da luz guiada em não-guiada).

A dispersão pode ser modal, devido à diferença do atraso de vários modos de propagação na fibra, ou material, resultante da diferença de velocidades de propagação dos diversos cumprimentos de onda presentes.

A dispersão modal só ocorre em fibras multimodo, sendo

que, nas que possuem índice de refração uniforme em seu núcleo (fibras de índice em degrau), ela chega a restringir a largura de faixa a algumas dezenas de MHz.km. Para reduzir este fenômeno, usa-se fibras de índice de refração graduado, permitindo-se uma largura de faixa de alguns GHz x km.

A dispersão material é comum a todas as fibras e depende da fonte de luz empregada. Se usamos diodo LED ela se torna mais significativa, enquanto que, com uma fonte de luz mais coerente como o LASER, ela se reduzirá sensivelmente.

Teremos portanto, em virtude da dispersão, uma restrição sobre a taxa de símbolos, ou seja, o aumento dessa taxa forçará uma diminuição do espaçamento entre os repetidores. A dispersão predominante será material ou modal, conforme a fonte de luz e o tipo de fibra utilizados.

O uso de mais de dois níveis nos afastaria esse problema pois evitaria o aumento da taxa inicial de símbolos que, no caso de 140 mb/s, já é relativamente alta.

Existem porém algumas desvantagens que tornam pouco recomendável a utilização de códigos multi-níveis. Em primeiro lugar, a linearidade de banda base foi demonstrada por Personick {5} para uma fonte incoerente conectada a uma fibra multimodo, apenas para o caso de sinal binário. Sendo assim, se o sinal transmitido possuir mais do que dois níveis, torna-se duvidosa a aplicação das técnicas usuais de equalização no receptor.

Além disso, como o ruído quântico no foto-diodo é correlato ao sinal, teremos que espaçar seus níveis e os respectivos limiares de decisão de maneira não-uniforme, mantendo-se, deste modo, a mesma probabilidade de erro, seja pela troca de um nível qualquer b_i por um outro b_{i+1} como pela troca de b_{i+1} por b_i {8}.

Por esses motivos, julgamos mais conveniente para altas velocidades como 34 Mb/s e 140 Mb/s, o emprego de códigos binários e sobre estes restringir nosso estudo.

A classe dos códigos binários pode ser dividido em dois grupos básicos: os códigos binários balanceados e os não-balanceados. Os códigos balanceados resolvem, por sua própria nature

za, os possíveis problemas do "dc wander" e da recuperação de relógio, pois possuem baixo conteúdo espectral em torno de f = 0 e alta densidade de transições de pulso.

Nos não-balanceados, entretanto, a conformação espectral é inadequada, sendo necessário um embaralhamento do sinal no sentido de garantir um número suficiente de transições de pulso para uma eficaz recuperação de relógio. Este embaralhamento também reduz o valor de pico das flutuações do nível de base, mas não melhora o seu valor RMS.

Dentre os códigos não-balanceados, o mais importante é o chamado código de verificação de paridade.

Neste tipo de código, para cada m bits de informação, são inseridos n, bits de verificação de paridade. Dessa forma, pode se construir diferentes formatos de quadro. Os n bits de controle podem ser agrupados depois dos de informação como também distribuídos em posições fixas ao longo do quadro, preservando- se assim de um surto de erros de linha.

Da mesma maneira como os n bits adicionais podem ser relativos ao controle de todos os m bits de informação, podemos também dividí-los em p palavras, cada uma verificando a paridade de p intervalos em que o bloco de informação seria também dividido.

Pode-se acrescentar ainda, além dos bits de paridade, informações como canais de serviço, alarme remoto, etc., aumentando-se assim o número de bits de controle, e portanto, a redundância do código. Para que estes bits de controle sejam identificados, será necessário um circuito de alinhamento de quadro na recepção.

A principal vantagem desse tipo de código é que a redundância introduzida requer um aumento percentual bem pequeno da taxa de símbolos.

Uma codificação dessa espécie, com m=17 e n=1, já foi adotada num sistema experimental da transmissão óptica em 140Mb/s. {10}.

Todavia, é intenção do presente trabalho deter-se apenas

nos códigos balanceados, mais concretamente os códigos de bloco tipo mB-nB. Com este fim, descrevemos no próximo capítulo um estudo teórico mais aprofundado sobre sua natureza e propriedades.

CAPÍTULO II

CÓDIGOS BALANCEADOS ÚTEIS

II.1 - INTRODUÇÃO

Os códigos do tipo mB-nB são códigos de bloco binários onde blocos de m bits adjacentes dos dados são codificados em palavras-códigos de n bits, sendo n > m. Existe portanto, com esta codificação, um aumento na taxa de símbolos transmitida, que é multiplicada pela razão n/m.

Nos sistemas de alta velocidade, como os de 140 Mb/s, a necessidade de preservar faixa recomenda o uso de baixas relações n/m. Por outro lado, a conveniência de manter a complexidade dos codificadores tão pequena quanto possível faz com que se evite valores muito grandes (acima de dez) para n e m. É possível entretanto que a evolução tecnológica nos permita, futuramente, utilizar valores maiores.

O objetivo deste capítulo é estabelecer os compromissos existentes entre velocidade, complexidade e balanceamento para este tipo de código . Antes porém de iniciar esse estudo, é oportuno definir os seguintes parâmetros.

Seja $\underline{a}_i = (a_{i1}, a_{i2}, \ldots, a_{in})$ a i-ésima palavra-código emitida pelo codificador, onde $a_{ij} \in \{0,1\}$. A disparidade D(i) associada à palavra $\underline{a}_i$ é dada por

$$D(i) = \sum_{j=1}^{n} (2a_{ij}-1) \qquad (1a)$$

A Soma Digital Corrida Terminal ou disparidade acumulada até a k-ésima palavra código é definida por:

$$SDCT(K) = SDCT(0) + \sum_{i=1}^{K} [D(i) - \langle D(i) \rangle] \qquad (1b)$$

onde SDCT(0) é uma constante arbitrária e $\langle D(i) \rangle$ é a disparidade média dentre todas as palavras-código empregadas na codificação.

A Soma Digital Corrida até o $[n(k-1)+\ell]$-ésimo bit transmitido será a seguinte:

$$SDC = SDCT(K-1) + \sum_{j=1}^{\ell} (2\,a_{kj}-1) - \langle D(i)\rangle \times \delta_{\ell n} \quad (1c)$$

onde

$$\delta_{\ell n} = \begin{cases} 0 \text{ p/ } \ell \neq n \\ 1 \text{ p/ } \ell = n \end{cases} \qquad \text{e} \qquad 1 \leq \ell \leq n$$

II.2 - O BALANCEAMENTO

Um sinal é dito balanceado se sua soma digital corrida está sempre confinada entre dois valores finitos, o que resulta no anulamento de seu espectro na frequência zero {4} . Por esta razão, o uso de códigos balanceados minimiza a flutuação de nível de base.

Para mostrar o relacionamento dessa flutuação com a soma digital corrida, vamos supor um sinal binário de taxa f_0 submetido a um filtro passa-altas com corte em f_0/N, dado pela expressão abaixo:

$$H(f) = 1 - \text{sinc}\,\frac{Nf}{f_0} \qquad (2)$$

onde

$$\text{sinc } x = \frac{\text{sen } \pi x}{\pi x}$$

Sendo x(t) o sinal em questão e y(t) o resultado da filtragem, teremos:

$$y(t) = x(t) * \left[\delta(t) - \frac{1}{NT}\ \text{ret}\,\frac{t}{NT}\right] \qquad (3)$$

onde $T = \frac{1}{f_0}$ e $\text{ret } x = \begin{cases} 1 & \text{p/ } |x| < \frac{1}{2} \\ 0 & \text{p/ } |x| > \frac{1}{2} \end{cases}$

Tratando-se de um sinal NRZ, temos

$$x(t) = \sum_{n=-\infty}^{\infty} an\ \text{ret}\left(\frac{t-nT}{T}\right), \quad an \in \{0,1\} \qquad (4)$$

e

$$y(t) = x(t) - \frac{1}{NT} \int_{-NT/2}^{NT/2} x(t)\,dt \qquad (5)$$

Para N ímpar, temos no centro do k-ésimo pulso

$$y(KT) = x(KT) - \frac{1}{N} \sum_{n=K-(N-1)/2}^{K+(N-1)/2} an =$$

$$x(KT) - \frac{1}{2} + \frac{n_1 - n_0}{2N} \qquad (6a)$$

onde

n_1 = número de marcas na somatória

n_0 = número de zeros na somatória

Para N par, o desbalanço (n_1-n_0) surge no fim de cada pulso:

$$y\,[(k+1/2)T] = x\,[(k+1/2)T] - \frac{1}{N} \sum_{n=K-N/2+1}^{K+N/2} an =$$

$$= x\,[(K+1/2)T] - \frac{1}{2} + \frac{n_1 - n_0}{2N} \qquad (6b)$$

Em ambos os casos, observamos que a flutuação do nível da base, dada pelo último termo, é diretamente proporcional à diferença entre o número de marcas e zeros na sequência de comprimento N. Esse desbalanço será dado pela variação da SDC do começo ao fim da sequência.

Em resumo, se codificamos o sinal de maneira a confinar sua SDC entre dois valores finitos, anulamos o seu espectro em f=0. Além disso, quanto menor for o intervalo dinâmico da variação da SDC, menor será o conteúdo espectral em torno da frequência zero, reduzindo-se assim a flutuação de nível de base.

Vamos supor, por exemplo, N=500, o que corresponde a uma frequência de corte nas baixas igual a 0,2% da taxa de símbolos. Então, se quisermos limitar a flutuação de nível de base a 1% do pico dos pulsos, será necessário fazer o intervalo dinâmico da SDC, que chamaremos P, menor ou igual a cinco. Para N= 1000, a mesma limitação pode ser satisfeita com $P \leq 10$.

Por fim, resta-nos observar que existem, em sistemas de transmissão óptica, projetos de equalizadores em que se integra o sinal na frente do receptor logo após a sua detecção {7} . Esta é mais uma razão que torna desejável o balanceamento dos si

nais pois, com esta integração, haveria uma perda de (20 log P) dB na faixa dinâmica do pré-amplificador (10 log P em termos de potência óptica).

Para que a SDC seja confinada, é suficiente confinar os seus valores terminais. Isto será garantido pela própria estrutura dos códigos mB-nB, como veremos a seguir.

II.3 - A ESTRUTURA DOS CÓDIGOS mB-nB

O confinamento da SDCT pode ser feito com o emprego de um ou dois alfabetos de codificação.

Se n for par, haverá $\binom{n}{n/2}$ palavras de comprimento n e disparidade nula. Se esse número for maior ou igual a 2^m, será possível garantir o balanceamento utilizando-se apenas um alfabeto, constituído somente por palavras de disparidade zero.

Fora da condição acima colocada, o balanceamento exige a adoção de dois alfabetos. Quando a SDCT estiver acima de um certo limiar, usa-se um alfabeto formado só com palavras de disparidade não-positiva, que chamaremos regressivo. Abaixo do limiar, usa-se o alfabeto progressivo, no qual as palavras têm disparidade positiva ou nula. Fica dessa forma garantido o balanceamento.

Para evitar o aparecimento de raias espectrais em f_0/n e seus múltiplos, é necessário que a probabilidade de ocorrência de marcas seja a mesma em todas as janelas temporais {13}. A fim de assegurar a suavidade do espectro do sinal, vamos então impor $p(0)=p(1)=1/2$ em todas as janelas temporais, mediante a adoção das seguintes regras de formação de códigos:

a) Se o alfabeto for único, a inclusão da qualquer palavra-código deve ser acompanhada pela inclusão de sua complementar (que resulta da troca de marcas por zero).

b) Havendo dois alfabetos, o progressivo deverá ser formado pelo conjunto das palavras complementares às do regressivo. Quando duas palavras complementares não estiverem no mesmo alfabeto, elas codificarão o mesmo bloco de entrada.

Como podemos notar, a imposição dois critérios acima descarta o uso de códigos com $\langle D_i \rangle \neq 0$ [eqs.(1)] . Algum códigos

conhecidos da literatura (como por exemplo o 5B-7B com apenas um alfabeto da codificação) não se enquadram, portanto, nestas condições.

Outra medida conveniente é fazer com que uma determinada palavra-código presente nos dois alfabetos codifique, em ambos, a mesma palavra-fonte. Com isto, a decodificação poderá se processar sem o conhecimento da SDC terminal.

A tabela abaixo nos dá um exemplo da aplicação dessas regras num código 3B-4B onde se emprega dois alfabetos.

PALAVRAS-FONTE	PALAVRAS-CÓDIGO			
	ALFABETO PROGRESSIVO (S1)		ALFABETO REGRESSIVO (S2)	
	PALAVRA	DISPARIDADE	PALAVRA	DISPARIDADE
000	1010	0	1010	0
001	1001	0	1001	0
010	0110	0	0110	0
011	0101	0	0101	0
100	1100	0	0011	0
101	1110	2	0001	-2
110	1101	2	0010	-2
111	1011	2	0100	-2

Tabela II.1 - Exemplo de código 3B-4B

A comutação dos alfabetos constitui a chamada lei de codificação ou lei do código e pode ser descrita pelo seguinte diagrama de estados.

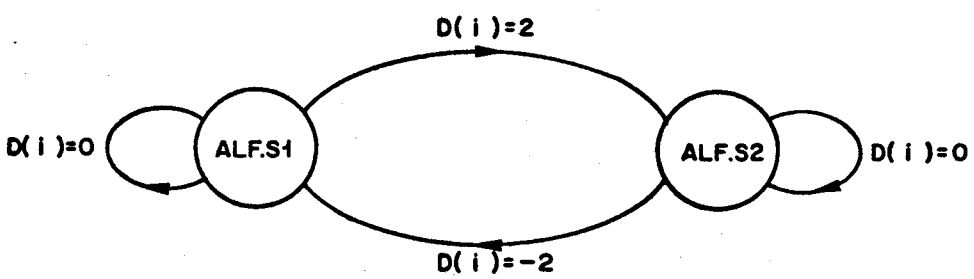


Fig. II.1 - Lei do código 3B-4B da tabela anterior.

Existem porém vários conjuntos de palavras-código que podem ser escolhidos sem violar as regras adotadas na estrutura do código. Um dos fatores que determinará nossa escolha será a procura do maior balanceamento possível do sinal codificado.

II.4 - O MÁXIMO DESBALANÇO P E O MÍNIMO EXCESSO DE VELOCIDADE:

O intervalo dinâmico P da soma digital corrida é dado pela soma de duas parcelas:

$$P = S + 2D \tag{7}$$

Onde S é o intervalo dinâmico da SDC terminal e D é a extensão, para cima e para baixo, desse intervalo, causada pelo efeito das disparidades intermediárias das palavras-códigos. Ambas as extensões são iguais por causa das condições de simetria já impostas sobre a estrutura dos códigos.

Seja d_{max} a máxima disparidade dentre as palavras-código escolhidas e convenvionemos como zero o mínimo valor para a SDCT (isto é, SDCT(0) = 0 na equação (1b)). Então, quando a SDCT valer d_{max} ou mais, aplica-se o alfabeto regressivo, caso contrário o valor terminal zero não voltaria a ser alcançado; quando for $(d_{max}-1)$ ou menos, aplica-se o progressivo, senão os valores negativos poderiam ser atingidos.

Contudo, o valor $(d_{max}-1)$ só será atingido se n for ímpar, pois aí as disparidades terminais também seriam ímpares e sua soma poderá ter qualquer paridade. Se n for par, as disparidades terminais e d_{max}, em consequência, também serão pares, de modo que o valor terminal $(d_{max}-1)$ não poderá ser atingido.

Podemos construir então um diagrama de estados geral que descreva a lei de um código mB-nB para os casos de n par e ímpar. Nesse diagrama cada estado corresponde a um valor da SDCT e sobre os ramos temos os possíveis valores de disparidades das palavras-código. A linha tracejada delimita as regiões de aplicabilidade dos alfabetos progressivo e regressivo , assinalados respectivamente por S1 e S2.

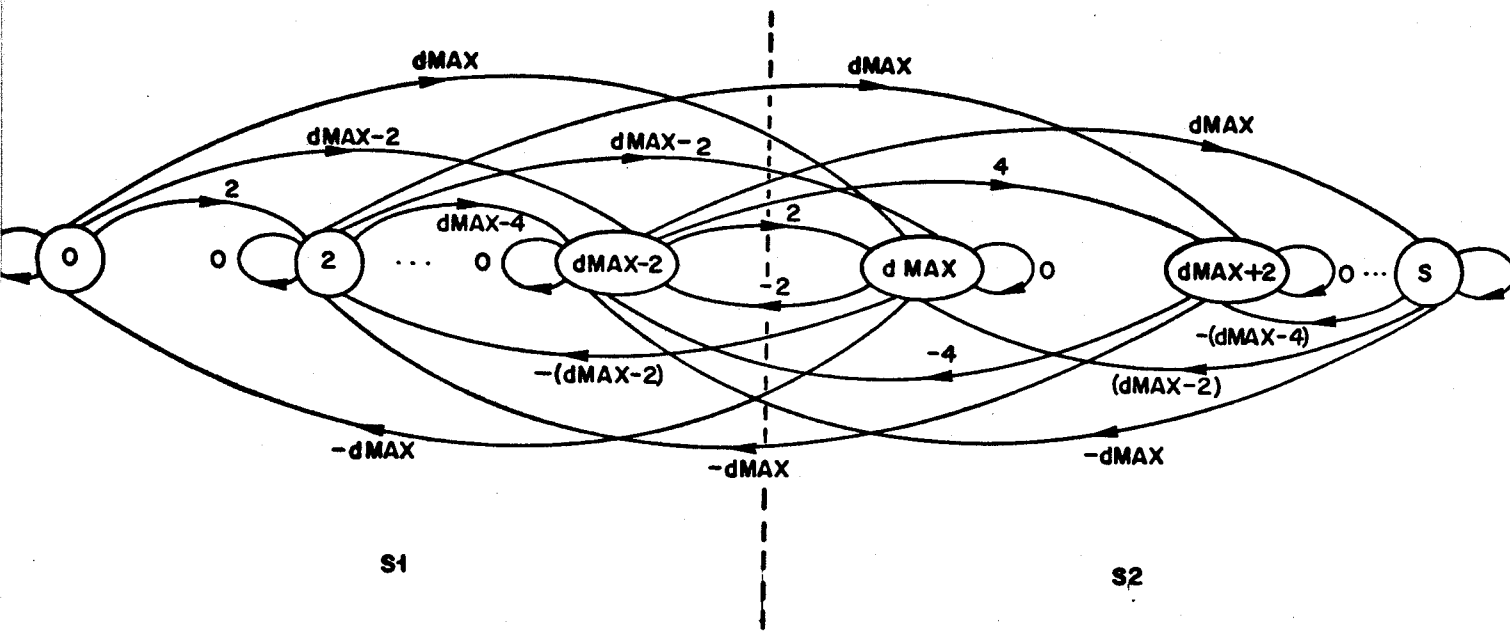


Fig. II.2 - Lei de um código mB-nB onde n é par.

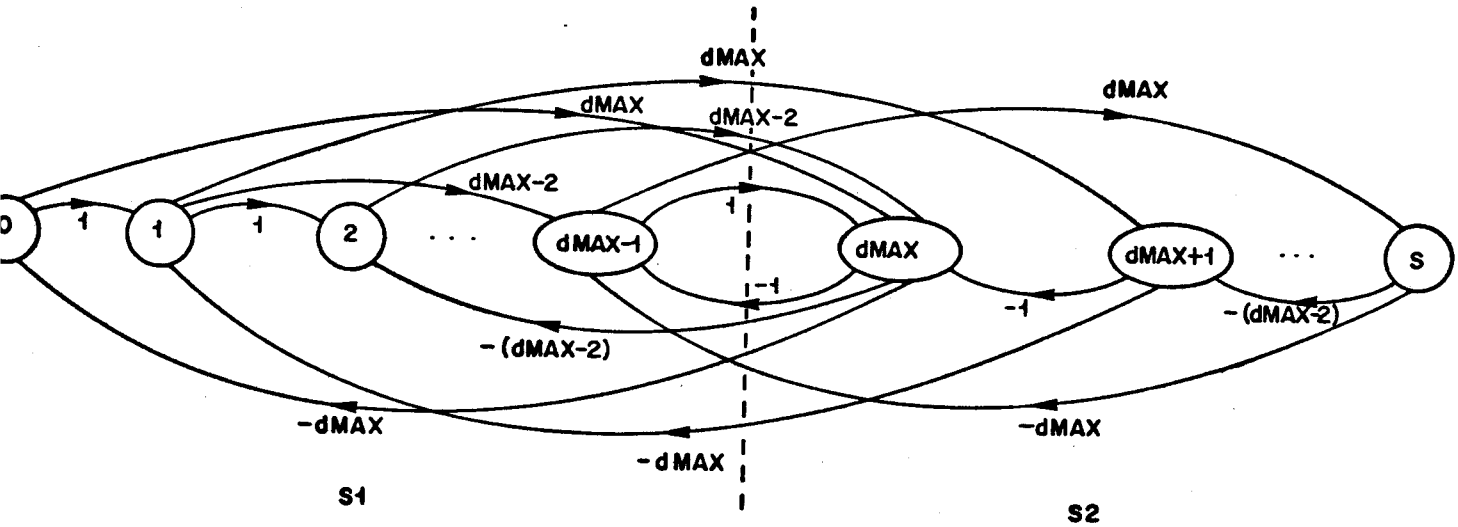


Fig. II.3 - Lei de um código mB-nB onde n é ímpar.

É claro que se $d_{max}=0$, existirá apenas um alfabeto e a lei do código reduz-se à forma elementar de um único estado da SDCT que será sempre nula.

Sendo assim, podemos calcular os seguintes valores para S:

$$S = 2\,d_{max}-1 \quad ; \; d_{max} \text{ ímpar} \qquad (8a)$$

$$S = 2\,d_{max}-2 \quad ; \; d_{max} \text{ par} > 0 \qquad (8b)$$

$$S = 0 \quad ; \; d_{max} = 0 \qquad (8c)$$

Quanto à extensão D, poderá assumir qualquer valor entra zero e D^{n}_{max}, dado pela palavra que começa com o maior número de marcas compatíveis com n, ou seja:

$$D^n_{max} \leq n/2 \quad \text{para n par} \tag{9a}$$

$$D^n_{max} \leq n-1/2 \quad \text{para n ímpar} \tag{9b}$$

Como $D \leq D^n_{max}$, as duas condições acima podem ser juntadas numa mesma inadequação:

$$D \leq \text{int}(n/2) \tag{10}$$

Onde int(.) designa a parte inteira do argumento.

Das equações (7) e (8), deduzimos que o par (d_{max},D) especifica P e temos:

$$P = 2(d_{max}+D)-1 \quad ; \; d_{max} \text{ ímpar} \tag{11a}$$

$$P = 2(d_{max}+D)-2 \quad ; \; d_{max} \text{ par} > 0 \tag{11b}$$

$$P = 2D \quad ; \; d_{max} = 0 \tag{11c}$$

Sendo que, nos três casos acima, o valor de D está sujeito à expressão (9).

Resta-nos saber o número $R(n, d_{max},D)$ de palavras de comprimento n disponíveis para a formação de um alfabeto (unico, progressivo ou regressivo) para cada par (d_{max},D) que especifique um certo valor de P. Tratando-se de um alfabeto regressivo, essas palavras serão todas aquelas de comprimento n e disparidade terminal entre $-d_{max}$ e zero, cujas disparidades intermediárias são confinadas por:

$$i_{min} = -d_{max}-D \tag{12a}$$

$$i_{max} = D \tag{12b}$$

O nível i_{min} se justifica porque a SDC, partindo de um nível terminal mínimo d_{max}, não pode cair abaixo de -D. O nível i_{max} porque se a SDC partir do nível máximo S, não poderá ultrapassar (S+D). Se houver um único alfabeto, i_{max} e i_{min} serão respectivamente D e -D, mantendo-se a validade das equações acima.

O cálculo de $R(n, d_{max},D)$ por técnicas combinatórias é bastante complicado. Desenvolvemos então um método simples para a sua determinação, utilizando-se de ábacos baseados no conhecido

triângulo de Pascal, cuja estrutura é mostrada abaixo:

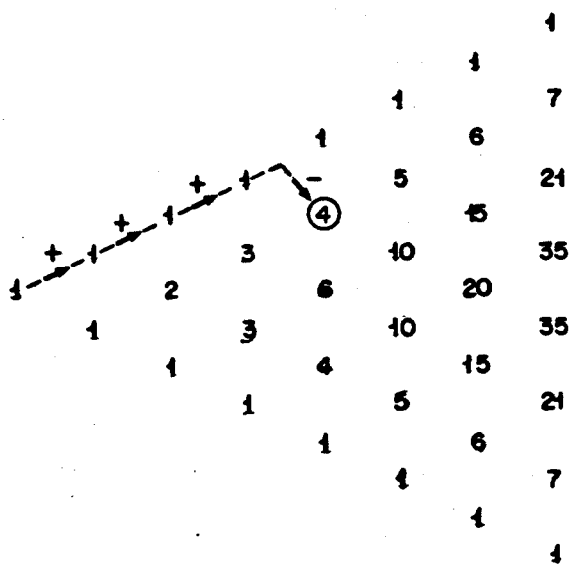


Fig. II.4 - O triângulo de Pascal

Neste triângulo, cada coluna é inicializada e finalizada pela unidade, sendo que seus elementos intermediários podem ser obtidos pela soma dos dois elmentos adjacentes (para cima e para baixo) da coluna interior. Por exemplo o elemento assinalado pelo círculo é obtido por: 3+1=4. O seu valor, em virtude da própria estrutura do triângulo, irá corresponder exatamente ao número de caminhos que, percorrendo o triângulo a partir de seu vértice, chega ao elemento em questão, passando apenas uma vez por cada coluna e deslocando-se sempre entre linhas adjacentes. Se simbolizarmos por "+" cada passo dado a umalinha de cima e por "-" a uma linha de baixo, iremos encontrar os quatros caminhos que levam até o elemento tomado no exemplo:

-+++, +-++, ++-+ e +++- (assinalado na figura).

Façamos agora uma analogia entre a obtenção de um elemento qualquer e a construção de uma palavra binária, associando a passagem de uma coluna à seguinte ao incremento no tamanho n dessa palavra. Se o deslocamento for para a linha superior, corresponderemos à adição de um dígito 1 na palavra, o caso contrário corresponderá à adição do 0. A disparidade da palavra encontrada,

ao término de um caminho qualquer será dada pelo desnível da linha final em relação à do ponto de partida (diferença entre o número de subidas e de descidas).

Enquadramos portanto o nosso triângulo na seguinte tabela:

d \ n	0	1	2	3	4	5	6	7	8
4					1	.	.	.	.
3				1					
2			1		(4)				
1		1		3					
0	1		2		6				
-1		1		3					
-2			1		4				
-3				1					
-4					1				

Tabela II.2 -

Onde, voltando ao exemplo anterior, o número (4) significa que existem quatro palavras de comprimento n=4 e disparidade d=2.

Até agora, determinamos apenas o número R(n,d) de palavra de tamanho n e disparidade d, o que poderia ser feito pelo mero cálculo de $\binom{n}{d}$. No entanto, para chegarmos a $R(n,d_{max},D)$ basta o simples truncamento do ábaco nas linhas superior e inferior que correspondam, respectivamente, aos valores i_{max} e i_{min}, que limitam as disparidades intermediárias das palavras disponíveis. A partir deste truncamento, construímos novas tabelas, seguindo as mesmas regras de formação, até o valor de n que nos interessar.

A título de ilustração, mostramos a tabela construída para P=3. Para este valor de intervalo de SDC, existirá apenas uma tabela possível, uma vez que só há um par (d_{max},D) que o especifique.

Na tabela, $R(n,d_{max},D)$ é calculado para todos os valores de $n \leq 12$, somando-se, em cada coluna, as quantias correspondentes às diversas disparidades terminais permitidas entre $-d_{max}$ e zero (pois estamos lidando, por opção, com alfabetos regressivo ou único), indicadas por * nas tabelas.

Naquelas colunas em que $n < d_{max}$, a excursão S da SDCT não é totalmente ocupada e naquelas em que temos $D > int\ (n/2)$, a excursão D não é totalmente ocupada (eq. 10). Em ambos os casos o valor efetivo de P será inferior aquele para o qual a tabela foi construída e, por esta razão, o valor de $R(n, d_{max}, D)$ será ignorado nestas colunas.

Uma vez determinado $R(n, d_{max},D)$, o máximo valor de m aplicável para o argumento será dado por:

$$2^{m_{max}} \leq R(n, d_{max},D) < 2^{m_{max}+1} \qquad (13a)$$

ou

$$m_{max} = int\left[\log_2 R(n,d_{max},D)\right] \qquad (13b)$$

Este valor de m nos dará, para cada n, o mínimo excesso de velocidade que se pode conseguir sem alterar o desbalanço P . O seu valor, em porcentagem, é calculado pela expressão:

$$\Delta(\%) = \frac{n - m_{max}}{m_{max}} \times 100 \qquad (14)$$

Tabela II.3 - Cálculo de $R(n,d_{max},D)$, m_{max} e Δ para o único par $(d_{max},D) = (1,1)$ que gera P=3.

$(d_{max},D)=(1,1)$.·. P = 3													
OBS	d^n	1	2	3	4	5	6	7	8	9	10	11	12
D	1	1		2		5		13		34		89	
	0		2		5		13		34		89		233
*	-1	1		3		8		21		55		144	
$-d_{max}-D$	2		1		3		8		21		55		144
R(n,1,1)		**		3		8		21		55		144	
m_{max}		**		1		3		4		5		7	
$\Delta(\%)$		**		200		66,7		75		80		57	

Pela tabela do exemplo, vemos que, para n < 12, os códigos mais eficientes de máximo desbalanço P=3 são os códigos 7B-11B, com 57% de excesso de velocidade.

Quando existirem vários pares (d_{max},D) que definem um mesmo P, deveremos construir uma tabela para cada par e identificar o mínimo excesso de velocidade pela comparação de todos os valores de Δ dessas tabelas. O menor deles identificará os códigos m_{max}B-nB maximamente eficientes para o desbalanço e complexidade especificada. Havendo empate, escolhe-se a opção de menor complexidade, isto é, a de menor valor de n.

II.5) LIMITES FUNDAMENTAIS DA EFICIÊNCIA

Na tabela II.3 podemos verificar um aumento não-linear de R(n,1,1) a medida que incrementamos n, nota-se inclusive que, neste caso em particular, seus valores correspondem exatamente aos números de Fibonacci tomados de forma intercalada. Já em Δ(%), notamos um comportamento oscilatório, cujos picos parecem diminuir com n. Esses comportamentos nos levaram a examinar o caso em que n tende a infinito, na intenção de procurar um limite assintótico para a eficiência.

Chamando de $\vec{V}_n$ o vetor formado pelos elementos de enésima coluna da tabela, teremos que:

$$\vec{V}_n = A\,\vec{V}_{n-2} \qquad \text{onde}$$

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \quad \text{e} \quad \vec{V}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{15}$$

Resulta então que:

$$\vec{V}_n = A^{(n-1)/2}\ \vec{V}_1 \quad \text{, para todo n ímpar} \tag{16}$$

Possuindo a matriz A dois autovalores distintos, podemos escrever {15} :

$$A = D \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} D^{-1} \tag{17}$$

onde D é a matriz formada pelos autovetores de A.

Então, substituindo A por sua forma diagonalizada na equação (16) teremos:

$$\vec{V}_n = D \begin{bmatrix} \lambda_1^{(n-1)/2} & 0 \\ 0 & \lambda_2^{(n-1)/2} \end{bmatrix} D^{-1} \vec{V}_1 \qquad (18)$$

Como R(n,1,1) é uma combinação linear de elementos de $\vec{V}_n$, podemos escrever:

$$R(n,1,1) = C_1 \times \lambda_1^{(n-1)/2} + C_2 \lambda_2^{(n-1)/2} \qquad (19)$$

onde C_1 e C_2 são constantes.

Os autovalores de A são dados pela equação característica:

$$\det(A-\lambda I) = \det \begin{bmatrix} 1-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} = \lambda^2 - 3\lambda + 1 = 0 \qquad (20)$$

$$\therefore \quad \lambda_{1,2} = \frac{3 \pm \sqrt{5}}{2} \qquad (21)$$

Como R(1,1,1)=1 e R(3,1,1)=3 , temos em (19):

$$C_1 + C_2 = 1 \qquad (22a)$$

$$C_1\lambda_1 + C_2\lambda_2 = 3 \qquad (22b)$$

Sendo que R(1,1,1) é tomado acima como mero resultado numérico.

Resolvendo para C_1 e C_2 e substituindo em (19), obtemos:

$$R(n,1,1) = \frac{5+3\sqrt{5}}{10} \times \left(\frac{3+\sqrt{5}}{2}\right)^{(n-1)/2} + \frac{5-3\sqrt{5}}{10} \times \left(\frac{3-\sqrt{5}}{2}\right)^{(n-1)/2} \qquad (23)$$

Para n grande, o primeiro termo, que corresponde ao maior autovalor, vai predominar, caracterizando um crescimento exponencial de R(n,1,1) com n grande, o que é típico dos números de Fibonacci. Para cada n, os códigos $m_{max}B$-nB maximamente eficientes serão dados por:

$$2^{m_{max}} \leq R(n,1,1) < 2^{m_{max}} + 1 \qquad (24a)$$

$$\therefore \quad m_{max} \leq \log_2 R(n,1,1) < m_{max} + 1 \qquad (24b)$$

Então, para n grande, teremos:

$$m_{max} \leq \log_2 \left\{ \frac{5+3\sqrt{5}}{10} \times \left(\frac{3+\sqrt{5}}{2} \right)^{(n-1)/2} \right\} < m_{max}+1 \qquad (25a)$$

$$\therefore$$

$$m_{max} \leq \log_2 \frac{5+3\sqrt{5}}{2} + \frac{n-1}{2} \log_2 \frac{3+\sqrt{5}}{2} < m_{max}+1 \qquad (25b)$$

Como m_{max} e n são grandes, podemos desprezar os termos constantes, reescrevendo a equação como sendo:

$$m_{max} \leq \frac{n}{2} \log_2 \frac{3+\sqrt{5}}{2} < m_{max}+1 \qquad (25c)$$

Da expressão acima, verificamos que m_{max} é limitado por uma função linear de n; para $n \to \infty$, este limite é alcançado, podendo-se escrever:

$$\lim_{n\to\infty} \frac{m_{max}}{n} = \frac{1}{2} \log_2 \frac{3+\sqrt{5}}{2} \qquad (26)$$

Este valor nos dá portanto a menor relação m_{max}/n possível par o caso do exemplo. Sendo assim, o mínimo excesso de velocidade Δ_{min} a ser tolerado para $(d_{max},D)=(1,1)$, $\therefore$ P=3 é, em porcentagem:

$$\Delta_{min}(\%) = \lim_{n\to\infty} \left(\frac{n - m_{max}}{m_{max}} \right) \times 100\% =$$

$$= \left[\frac{2}{\log_2 \frac{3+\sqrt{5}}{2}} - 1 \right] \times 100\% \cong 44{,}04\% \qquad (27)$$

Isto significa que, se quisermos limitar o desbalanço entre marcas e zeros a apenas três em qualquer parte da sequência codificada, será necessário tolerar um excesso de velocidade de pelo menos 44,04% em relação à taxa inicial de dígitos, por maior que seja a complexibilidade do código.

Para o caso de P qualquer, pode-se repetir o desenvolvimento acima realizado para P=3, obtendo assim uma eficiência assintótica para os códigos mB-nB quando n tende a infinito.

Tendo em vista que os coeficientes do desenvolvimento de $R(n,d_{max},D)$ em potências de seus autovalores são irrelevantes para o resultado e que este só depende do maior autovalor, o procedimento pode ser abreviado pelo seguinte roteiro, para cada valor fixado de P:

1 - Determinar todos os pares (d_{max},D) que especificam P de acordo com as equações (11).

2 - Para cada par, construir a tabela de cálculos de $R(n,d_{max},D)$ e, através dela, obter a matriz de transição entre duas colunas consecutivas. As colunas consideradas serão aquelas de mesma paridade que P, pois n e P não podem ter paridades diferentes.

A forma genérica dessa matriz será:

$$A = \begin{bmatrix} a_{11} & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 2 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 2 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & & \\ 0 & 0 & 0 & 0 & \cdots & 2 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & a_{kk} \end{bmatrix} \qquad (28)$$

onde:

a_{11} = 1 ou 2

a_{kk} = 1 ou 2

Como se pode ver, a matriz acima será sempre Hermitiana e, em consequência, seus autovetores são ortogonais {15}. Sendo assim, a matriz A terá sempre autovalores distintos, o que nos garante a validade da equação (17).

O elemento a_{11} será unitário se o primeiro elemento não identicamente nulo do vetor estiver na primeira linha da tabela e será igual a 2 se o vetor começar na segunda linha. Portanto, $a_{11} = 1$ se D e P possuirem a mesma paridade e $a_{11} = 2$ se suas paridades forem diferentes. Analogamente, sendo K a ordem da matriz, $a_{kk}=1$ se $(d_{max}+D)$ e P tiverem a mesma paridade, ou seja , se D for par, e $a_{kk}=2$ se D for ímpar.

As condições acima colocadas são facilmente compreendidas se pensarmos que, estando o primeiro elemento de $\vec{V}_n$ na primeira linha, só haverá um caminho, dentro da tabela, ligando - o ao primeiro elemento de $\vec{V}_{n+2}$, enquanto que se êle estiver na segunda linha, haverá dois caminhos. Este raciocínio pode ser aplicado ao último elemento e também aos intermediários, basta considerarmos que a existência de dois caminhos entre dois elementos de colunas consecutivas indica que o elemento da coluna anterior é somado duas vezes para a obtenção do posterior, um único caminho significa que o elemento em questão é somado uma vez só.

As linhas da tabelas de cálculo são numeradas de $(-d_{max} - D)$ a D, havendo portanto $(d_{max}+2D+1)$ linhas ocupadas alternadamente pelos elementos do vetor gerador de $R(n,d_{max},D)$. A ordem do vetor será então:

$$K = \frac{d_{max}+2D+1}{2} \quad , d_{max} \text{ ímpar} \qquad (29a)$$

$$K = \frac{d_{max}+2D}{2} \quad , d_{max} \text{ par, D ímpar} \qquad (29b)$$

$$K = \frac{d_{max}+2D}{2} + 1 \quad , d_{max} \text{ par, D par} \qquad (29c)$$

As equações (29) e as regras de determinação de a_{11} e a_{kk} permitem que se construa a matriz A sem prévia consulta à tabela de cálculo de $R(n,d_{max},D)$. Determinando-se as matrizes A correspondentes aos pares (d_{max},D) que geram um certo valor de P, é comum observar a repetição de matrizes observadas em valores menores de P. Assim sendo, se o cálculo for feito para valores crescentes de P, só será preciso executar os cálculos do passo seguinte para as matrizes novas de cada P.

3 - Para cada matriz A, achar a máxima raiz de sua equação característica

$$\det(A - \lambda I) = 0 \qquad (30)$$

e em seguida identificar a maior dessas soluções entre todas as matrizes correspondentes a P, que chamaremos $\lambda_{max}(P)$.

4 - O limite assintótico de velocidade será dado, em porcentagem, por:

$$\Delta_{min}(P) = \left[\frac{2}{\log_2 \lambda_{max}(P)} - 1 \right] \times 100\% \cdot \qquad (31)$$

A validação rigorosa desse roteiro exigiria ainda que se demonstrasse a não nulidade do coeficiente associado às potências de λ_{max} no desenvolvimento do respectivo $R(n,d_{max},D)$ em série de potência dos autovalores. Embora isso não tenha sido demonstrado de forma genérica, em nenhum caso pesquisado esse coeficiente foi zero.

Na seção seguinte, mostramos as tabelas elaboradas para P entre 2 e 10 e os respectivos cálculos de $R(n,d_{max},D)$, m_{max} e $\Delta(\%)$ para uma complexidade de até n=16. Esses resultados, juntamente com os valores assintóticos obtidos, mas levarão a tirar uma conclusão geral sobre a utilidade das diversas classes de códigos mB-nB.

II.6 - AS TABELAS DE CÁLCULO DE $R(n,d_{max},D)$

Pesquisamos os casos referentes a $2 \leq P \leq 10$. Para P = 1, não há possibilidade de se transmitir informação pois o único sinal binário possível com este desbalanceamento é a própria onda quadrada. Em nossas tabelas, o sinal * indicará sempre as disparidades terminais permitidas (entre $-d_{max}$ e zero) para as palavras-código, enquanto que ** indicará quando o valor de n for tal que D > int(n/2) ou $d_{max} > n$, isto é, quando o P efetivo não corresponder ao desbalanço para o qual a tabela foi construída.

$d_{max} = 0$
$D = 1$

$\therefore P = 2$

OBS.	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	1	1		2		4		8		16		32		64		128	
*	0		2		4		8		16		32		64		128		256
$-D-d_{max}$	-1	1		2		4		8		16		32		64		128	
$R(n,d_{max},D)$			2		4		8		16		32		64		128		256
m_{max}			1		2		3		4		5		6		7		8
Excesso de Veloc. (%)			100		100		100		100		100		100		100		100

Tab. II.4.a

$d_{max} = 2$

$D = 0$

$\therefore P = 2$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
*D	0		1		2		4		8		16		32		64		128
	-1	1		2		4		8		16		32		64		128	
*-D-d_{max}	-2		1		2		4		8		16		32		64		128
$R(n,d_{max},D)$			2		4		8		16		32		64		128		256
m_{max}			1		2		3		4		5		6		7		8
Excesso de Veloc. (%)			100		100		100		100		100		100		100		100

Tab. II.4.b

$d_{max} = 1$

$D = 1$

$\therefore P = 3$

OBS.	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	1	1		2		5		13		34		89		233		610	
	0		2		5		13		34		89		233		610		1597
*	-1	1		3		8		21		55		144		377		987	
$-d_{max}-D$	-2		1		3		8		21		55		144		377		987
$R(n,d_{max},D)$		**		3		8		21		55		144		377		987	
m_{max}		**		1		3		4		5		7		8		9	
Excesso de Veloc. (%)		**		200		66,7		75		80	·	57,1		62,5		66,7	

Tab. II.5

$d_{max} = 0$

$D = 2$

$\therefore P = 4$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	2		1		3		9		27		81		243		729		2187
	1	1		3		9		27		81		243		729		2187	
*	0		2		6		18		54		162		486		1458		4374
	-1	1		3		9		27		81		243		729		2187	
$-d_{max}-D$	-2		1		3		9		27		81		243		729		2187
$R(n, d_{max}, D)$			**		6		18		54		162		486		1458		4374
m_{max}			**		2		4		5		7		8		10		12
Excesso de Veloc. (%)			**		100		50		60		42,9		50		40		33,3

Tab. II.6.a

$d_{max} = 2$

$D = 1$ $\therefore P = 4$

OBS.	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	1	1		2		5		14		41		122		365		1094	
*	0		2		5		14		41		122		365		1094		3281
	-1	1		3		9		27		81		243		729		2187	
*	-2		1		4		13		40		121		364		1093		3280
$-D-d_{max}$	-3			1		4		13		40		121		364		1093	
$R(n, d_{max}, D)$			3		9		27		81		243		729		2187		6561
m_{max}			1		3		4		6		7		9		11		12
Excesso de Veloc. (%)			100		33,3		50		33,3		42,9		33,3		27,3		33,3

Tab. II.6.b

$d_{max} = 1$

$D = 2$

$\therefore P = 5$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	2		1		3		9		28		89		286		924		2993
	1	1		3		9		28		89		286		924		2993	
	0		2		6		19		61		197		638		2069		6714
*	-1	1		3		10		33		108		352		1145		3721	
	-2		1		4		14		47		155		507		1652		5373
$d_{max}-D$	-3			1		4		14		47		155		507		1652	
$R(n,d_{max},D)$		**		**		10		33		108		352		1145		3721	
m_{max}		**		**		3		5		6		8		10		11	
Excesso de Veloc. (%)		**		**		66,7		40		50		37,5		30		36,4	

Tab. II.7.a

$d_{max} = 3$

$D = 0$

$\therefore P = 5$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	0		1		2		5		13		34		89		233		610
*	-1	1		2		5		13		34		89		233		610	
	-2		1		3		8		21		55		144		377		987
* $D-d_{max}$	-3			1		3		8		21		55		144		377	
$R(n,d_{max},D)$		**		3		8		21		55		144		377		987	
m_{max}		**		1		3		4		5		7		8		9	
Excesso de Veloc. (%)		**		200		66,7		75		80		57,1		62,5		66,7	

Tab. II.7.b

$d_{max} = 0$

$D = 3$ $\therefore P = 6$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	3			1		4		14		48		164		560		1912	
	2		1		4		14		48		164		560		1912		6528
	1	1		3		10		34		116		396		1352		4616	
*	0		2		6		20		68		232		792		2704		9232
	-1	1		3		10		34		116		396		1352		4616	
	-2		1		4		14		48		164		560		1912		6528
$-d_{max}-D$	-3			1		4		14		48		164		560		1912	
$R(n,d_{max},D)$			**		**		20		68		232		792		2704		9232
m_{max}			**		**		4		6		7		9		11		13
Excesso de Veloc. (%)			**		**		50		33,3		42,9		33,3		27,3		23,1

Tab. II.8.a

$d_{max} = 2$

$D = 2$

$\therefore P = 6$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	2		1		3		9		28		90		296		988		3328
	1	1		3		9		28		90		296		988		3328	
*	0		2		6		19		62		206		692		2340		7944
	-1	1		3		10		34		116		396		1352		4616	
*	-2		1		4		15		54		190		660		2276		7816
	-3			1		5		20		74		264		924		3200	
$-d_{max}-D$	-4				1		5		20		74		264		924		3200
$R(n, d_{max}, D)$			**		10		34		116		398		1352		4616		15760
m_{max}			**		3		5		6		8		10		12		13
Excesso de Veloc. (%)			**		33,3		20		33,3		25		20		16,7		23,1

Tab. II.8.b

$d_{max} = 4$
$D = 0$

$\therefore P = 6$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
*D	0		1		2		5		14		41		122		365		1094
	-1	1		2		5		14		41		122		365		1094	
*	-2		1		3		9		27		81		243		729		2187
	-3			1		4		13		40		121		364		1093	
*-d_{max}-D	-4				1		4		13		40		121		364		1093
$R(n, d_{max}, D)$			**		6		18		54		162		486		1458		4374
m_{max}			**		2		4		5		7		8		10		12
Excesso de Veloc. (%)			**		100		50		60		429		50		40		33,3

Tab. II.8.c

$d_{max} = 1$

$D = 3$

$\therefore P = 7$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	3			1		4		14		48		165		571		1988	
	2		1		4		14		48		165		571		1988		6953
	1	1		3		10		34		117		406		1417		4965	
	0		2		6		20		69		241		846		2977		10490
*	-1	1		3		10		35		124		440		1560		5525	
	-2		1		4		15		55		199		714		2548		9061
	-3			1		5		20		75		274		988		3536	
$-d_{max}-D$	-4				1		5		20		75		274		988		3536
$R(n,d_{max},D)$		**		**		**		35		124		440		1560		5525	
m_{max}		**		**		**		5		6		8		10		12	
Excesso de Veloc.(%)		**		**		**		40		50		37,5		30		25	

Tab. II.9.a.

$d_{max} = 3$

$D = 1$ $\therefore P = 7$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	1	1		2		5		14		42		131		417		1341	
	0		2		5		14		42		131		417		1341		4334
*	-1	1		3		9		28		89		286		924		2993	
	-2		1		4		14		47		155		507		1652		5373
*	-3			1		5		19		66		221		728		2380	
$-d_{max}-D$	-4				1		5		19		66		221		728		2380
$R(n,d_{max},D)$		**		4		14		47		155		507		1652		5373	
m_{max}		**		2		3		5		7		8		10		12	
Excesso de Veloc. (%)		**		50		66,7		40		28,6		37,5		30		25	

Tab. II.9.b

$d_{max} = 0$

$D = 4$

$\therefore P = 8$

OBS.	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	4				1		5		20		75		275		1000		3625
	3			1		5		20		75		275		1000		3625	
	2		1		4		15		55		200		725		2625		9500
	1	1		3		10		35		125		450		1625		5875	
*	0		2		6		20		70		250		900		3250		11750
	-1	1		3		10		35		125		450		1625		5875	
	-2		1		4		15		55		200		725		2625		9500
	-3			1		5		20		75		275		1000		3625	
$-d_{max}-D$	-4				1		5		20		75		275		1000		3625
$R(n, d_{max}, D)$			**		**		**		70		250		900		3250		11750
m_{max}			**		**		**		6		7		9		11		13
Excesso de Veloc. (%)			**		**		**		33,3		42,9		33,3		27,3		23,1

Tab. II.10.a

$d_{max} = 2$

$D = 3$ $\therefore P = 8$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	3			1		4		14		48		165		572		2001	
	2		1		4		14		48		165		572		2001		7056
	1	1		3		10		34		117		407		1429		5055	
*	0		2		6		20		69		242		857		3054		10930
	-1	1		3		10		35		125		450		1625		5875	
*	-2		1		4		15		56		208		768		2821		10320
	-3			1		5		21		83		318		1196		4445	
	-4				1		6		27		110		428		1624		6069
$-D - d_{max}$	-5					1		6		27		110		428		1624	
$R(n, d_{max}, D)$			**		**		35		125		450		1625		5875		21250
m_{max}			**		**		5		6		8		10		12		14
Excesso de Veloc. (%)			**		**		20		33,3		25		20		16,7		14,3

Tab. II.10.b

$d_{max} = 4$

$D = 1$ $\therefore P = 8$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	1	1		2		5		14		42		132		428		1416	
*	0		2		5		14		42		132		428		1416		4744
	-1	1		3		9		28		90		296		988		3328	
*	-2		1		4		14		48		164		560		1912		6528
	-3			1		5		20		74		264		924		3200	
*	-4				1		6		26		100		364		1288		4488
$-D-d_{max}$	-5					1		6		26		100		364		1288	
$R(n,d_{max},D)$			**		10		34		116		396		1352		4616		15760
m_{max}			**		3		5		6		8		10		12		13
Excesso de Veloc. (%)			**		33,3		20		33,3		25		20		16,7		23,1

Tab. II.10.c

$d_{max} = 1$

$D = 4$

$\therefore P = 9$

OBS.	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	4				1		5		20		75		275		1001		3639
	3			1		5		20		75		275		1001		3639	
	2		1		4		15		55		200		726		2638		9604
	1	1		3		10		35		125		451		1637		5965	
	0		2		6		20		70		251		911		3327		12190
*	-1	1		3		10		35		126		460		1690		6225	
	-2		1		4		15		56		209		779		2898		10760
	-3			1		5		21		83		319		1208		4535	
	-4				1		6		27		110		429		1637		6172
$-d_{max}-D$	-5					1		6		27		110		429		1637	
$R(n,d_{max},D)$		**		**		**		**		126		460		1690		6225	
m_{max}		**		**		**		**		6		8		10		12	
Excesso de Veloc. (%)		**		**		**		**		50		37,5		30		25	

Tab. II.11.a

$d_{max} = 3$

$D = 2$

$\therefore P = 9$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	2		1		3		9		28		90		297		1000		3417
	1	1		3		9		28		90		297		1000		3417	
	0		2		6		19		62		207		703		2417		8382
*	-1	1		3		10		34		117		406		1417		4965	
	-2		1		4		15		55		199		714		2548		9061
*	-3			1		5		21		82		308		1131		4096	
	-4				1		6		27		109		417		1548		5644
$-D-d_{max}$	-5					1		6		27		109		417		1548	
$R(n,d_{max},D)$		**		**		15		55		199		714		2548		9061	
m_{max}		**		**		3		5		7		9		11		13	
Excesso de Veloc. (%)		**		**		66,7		40		28,6		22,2		18,2		15,4	

Tab. II. 11.b

$d_{max} = 5$

$D = 0$ $\therefore P = 9$

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
OBS	0		1		2		5		14		42		131		417		1341
*	-1	1		2		5		14		42		131		417		1341	
	-2		1		3		9		28		89		286		924		2993
*	-3			1		4		14		47		155		507		1652	
	-4				1		5		19		66		221		728		2380
*-D-d_{max}	-5					1		5		19		66		221		728	
$R(n, d_{max}, D)$		**		**		9		28		89		286		924		2993	
m_{max}		**		**		3		4		6		8		9		11	
Excesso de Veloc. (%)		**		**		66,7		75		50		37,5		44,4		36,4	

Tab. II.11.c

$d_{max} = 0$

$D = 5$

$\therefore P = 10$

OBS	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	5					1		6		27		110		419		1608	
	4				1		6		27		110		419		1608		6097
	3			1		5		21		83		309		1189		4489	
	2		1		4		15		56		209		770		2881		10754
	1	1		3		10		35		126		461		1692		6265	
*	0		2		6		20		70		252		922		3384		12530
	-1	1		3		10		35		126		461		1692		6265	
	-2		1		4		15		56		209		770		2881		10754
	-3			1		5		21		83		309		1189		4489	
	-4				1		6		27		110		419		1608		6097
$-d_{max}$ -D	-5					1		6		27		110		419		1608	
$R(n, d_{max}, D)$			**		**		**		**		252		922		3384		12530
m_{max}			**		**		**		**		7		9		11		13
Excesso de Veloc. (%)			**		**		**		**		42,9		33,3		27,3		23,1

Tab. II.12.a

$d_{max} = 2$

$D = 4$ $\therefore P = 10$

OBS.	d \ n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	4				1		5		20		75		275		1001		3640
	3			1		5		20		75		275		1001		3640	
	2		1		4		15		55		200		726		2639		9619
	1	1		3		10		35		125		451		1638		5979	
*	0		2		6		20		70		251		912		3340		12294
	-1	1		3		10		35		126		461		1702		6315	
*	-2		1		4		15		56		210		790		2975		11200
	-3			1		5		21		84		329		1273		4885	
	-4				1		6		28		119		483		1910		7432
	-5					1		7		35		154		637		2547	
$-D-d_{max}$	-6						1		7		35		154		637		2547
$R(n,d_{max},D)$			**		**		**		126		461		1702		6315		23494
m_{max}			**		**		**		6		8		10		12		14
Excesso de Veloc. (%)			**		**		**		33,3		25		20		16,7		14,3

Tab. II.12.b

$d_{max} = 4$

$D = 2$

$\therefore P = 10$

OBS.	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D	2		1		3		9		28		90		297		1001		3431
	1	1		3		9		28		90		297		1001		3431	
*	0		2		6		19		62		207		704		2430		8486
	-1	1		3		10		34		117		407		1429		5055	
*	-2		1		4		15		55		200		725		2625		9500
	-3			1		5		21		83		318		1196		4445	
*	-4				1		6		28		118		471		1820		6889
	-5					1		7		35		153		624		2444	
$-D-d_{max}$	-6						1		7		35		153		624		2444
$R(n,d_{max},D)$			**		11		40		145		525		1900		6875		24875
m_{max}			**		3		5		7		9		10		12		14
Excesso de Veloc. (%)			**		33,3		20		14,3		11,1		20		16,7		14,3

Tab. II.12.c

$d_{max} = 6$
$D = 0$

$\therefore$ P = 10

OBS	n / d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
*D	0		1		2		5		14		42		132		428		1416
	-1	1		2		5		14		42		132		428		1416	
*	-2		1		3		9		28		90		396		988		3228
	-3			1		4		14		48		164		560		1912	
*	-4				1		5		20		74		264		924		3200
	-5					1		6		26		100		364		1288	
*-D-d_{max}	-6						1		6		26		100		364		1288
R(n, d_{max}, D)			**		**		20		68		232		792		2704		9232
m_{max}			**		**		4		6		7		9		11		13
Excesso de Veloc. (%)			**		**		50		33,3		42,9		33,3		27,3		23,1

Tab. II.12.d

Para P=2, o par (d_{max},D) poderá ser (0,1) ou (2,0). Analisando suas respectivas tabelas, verificamos que o excesso de taxa $\Delta(\%)$ se mantém igual a 100% para todos os valores de n, As matrizes de transição entre colunas serão:

$$A = \begin{bmatrix} 2 \end{bmatrix} \quad \text{para o caso } \underline{a} \text{ e}$$

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad \text{para o caso } \underline{b}$$

Seguindo o roteiro do cálculo do limite assintótica de $\Delta(\%)$, iremos obter para as duas tabelas:

$$\lambda_{max}(2) = 2 \quad , \text{ portanto}$$

$$\Delta_{min}(2) = \left[\frac{2}{\log_2} - 1 \right] \times 100 = 100\%$$

o que está de acordo com o esperado, ou seja, mesmo que aumentemos indefinidamente a complexidade, se mantivermos P=2, o aumento na taxa será sempre de 100%.

O caso de P=3 corresponde ao já tomado como exemplo anteriormente. Podemos notar que o código 7B-11B continua sendo o mais eficiente para esse desbalanço, embora tenhamos extendido o valor de n até 16. Para melhorar a eficiência, seria necessário um nível de complexidade ainda maior, sendo que o limite, como já vimos, é de 44,04%.

Analisando as tabelas relativas a P=4, verificamos que a classe de códigos que se apresenta mais eficiente é a do 11B-14B ($\Delta(\%)$=27,3%). Porém, sua complexidade já é um tanto significativa para os recursos atuais de lógica e memória. Para uma complexidade menor exigida, haveria a opção de se usar o 3B-4B, com 33% de excesso de velocidade. No entanto, para P=4, os códigos 3B-4B apresentam uma dificuldade em termos de sincronismo, conforme iremos averiguar no capítulo seguinte. Essa dificuldade só será superada com o aumento do desbalanço para P=6, onde, de acordo com a tabela II.7.b, há também a possibilidade de uso do 3B-4B.

As matrizes de transição entre colunas são:

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{bmatrix} \qquad \text{p/o caso } \underline{a} \quad \text{e}$$

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad \text{p/o caso } \underline{b} \quad ,$$

sendo as equações características dadas, respectivamente, por:

$$\lambda^3 + 4\lambda^2 - 3 = 0 \qquad (32a)$$

e

$$\lambda^2 - 4\lambda + 3 = 0 \qquad (32b)$$

Nota-se que as equações acima possuem as mesmas raízes, executando-se o zero que só se faz presente na equação (32.a) . Resolvendo-as, encontramos:

$$\lambda_{max}(4)=3 \quad \text{e} \quad \Delta_{min}(4) = \left[\frac{2}{\log_2 3} - 1\right] \times 100 = 26{,}19\%$$

Verifica-se também que, nas tabelas II.6, o número $R(n,d_{max},D)$ é sempre multiplicado por três, a cada incremento de n, sendo $R(1,0,2)=2$ no caso a (esse valor possui apenas significado numérico) e $R(1,2,1) = 3$ no caso b. Chegamos então a:

$$R(n,0,2) = 2 \times 3^{n/2-1} \qquad \text{e} \qquad (33a)$$

$$R(n,2,1) = 3 \times 3^{n/2-1} \qquad (33b)$$

o que nos confirma o resultado de $\lambda_{max}=3$ para as duas tabelas e mostra que os coeficientes dos demais autovalores, no desenvolvimento de $R(n,0,2)$ e $R(n,2,1)$, são nulos.

O valor encontrado para Δ_{min} é razoavelmente próximo ao obtido com o uso do 11B-14B, o que nos leva a concluir que é praticamente desnecessário um aumento da complexidade, neste caso, para $n > 16$, pois o ganho na eficiência seria muito pequeno.

Aumentando a excursão da SDC para P=5, surgem alguns códigos razoavelmente eficientes como o 5B-7B, 8B-11B, e 10B-13B.

Dentre eles, o mais viável para a implementação é o 5B-7B que, por outro lado, é o menos eficiente dos três.

Esses códigos, conforme indicado na Tab.II.7.a, possuem um alfabeto formado apenas por palavras de disparidade terminal igual a-1. Este alfabeto será o regressivo e não o único, pois, caso contrário, teríamos $\leq D(i)> \neq 0$ e as condições de simetria não seriam obedecidas. Sendo assim, todo o código de n ímpar terá necessariamente, dois alfabetos de codificação.

A matriz de transição da tabela II.7.a é:

$$A=\begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad \text{e temos } \lambda^3-5\lambda^2+6\lambda-1=0 \qquad (34)$$

De onde tiramos que a maior raiz vale aproximadamente 3,247.

Da tabela II.7.b, temos:

$A=\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$, cuja equação característica é a mesma que a obtida para o caso de P=3, ou seja, o seu maior auto-valor será $\frac{3+\sqrt{5}}{2} = 2,618$.

Concluímos então que $\lambda_{max}(5) = 3,247$ e, portanto,

$$\Delta_{min}(5) = \left[\frac{2}{\log_2 3,247} - 1\right] \times 100 = 17,71\% \quad , \text{ que é}$$

razoavelmente inferior ao menor excesso de taxa encontrado para $n \leq 16$.

Com desbalanço P=6 , aparece o 5B-6B, que apresenta um bom compromisso entre complexidade e excesso de velocidade; o 12B-14B tem melhor eficiência mas é bem mais complexo .

Das três matrizes de transição correspondentes, a que nos fornece o maior autovalor é a da tabela II.8.a, dada por:

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$, cuja equação característica é:

$$\lambda^3 - 6\lambda^2 + 10\lambda - 4 = 0 \qquad (35)$$

onde

$$\lambda_{max}(6) = 2 + \sqrt{2} = 3,414$$

e

$$\Delta_{min}(6) = \left[\frac{2}{\log_2 3,414} - 1 \right] \times 100\% = 12,90\%$$

Extendendo esse procedimento de cálculo de $\lambda_{max}(P)$ e $\Delta_{min}(P)$ para até P=10, chegamos aos resultados expostos na tabela II.13:

P	Equação Característica	λ_{max}	$\Delta_{min}(\%)$
2	$\lambda - 2 = 0$	2	100
3	$\lambda^2 - 3\lambda + 1 = 0$	2,618	44,04
4	$\lambda^2 - 4\lambda + 3 = 0$	3	26,19
5	$\lambda^3 - 5\lambda^2 + 6\lambda - 1 = 0$	3,247	17,71
6	$\lambda^3 - 6\lambda^2 + 10\lambda - 4 = 0$	3,414	12,90
7	$\lambda^4 - 7\lambda^3 + 15\lambda^2 - 10\lambda + 1 = 0$	3,532	9,861
8	$\lambda^4 - 8\lambda^3 + 21\lambda^2 - 20\lambda + 5 = 0$	3,618	7,805
9	$\lambda^5 - 9\lambda^4 + 28\lambda^3 - 35\lambda^2 + 15\lambda - 1 = 0$	3,6825	6,344
10	$\lambda^5 - 10\lambda^4 + 36\lambda^3 - 56\lambda^2 + 35\lambda - 6 = 0$	3,732	5,265

Tabela II.13 - Limites Assintóticos para $n \to \infty$.

Por inspecção da tabela II.13, verificamos um norma para a formação das equações características das matrizes de transição que geram $\lambda_{max}(P)$. Seja essa equação dada por:

$$A^{(P)}(\lambda) = a_0^{(P)} \lambda^u + a_1^{(P)} \lambda^{u-1} + \ldots + a_{u-1} \lambda + a_u \qquad (36)$$

onde o índice (P) diz respeito ao desbalanço referente à equação .

A ordem do polinômio acima será sempre:

$$u = P/2 \quad \text{para P par} \tag{37a}$$

e

$$u = P+1/2 \quad \text{para P ímpar} \tag{37b}$$

Os valores absolutos de seus primeiros termos podem ser obtidos da seguinte forma:

$$|a_0^{(P)}| = 1 \tag{38a}$$

$$|a_1^{(P)}| = P \tag{38b}$$

enquanto que o do último termo será:

$$|a_u^{(P)}| = 1 \quad \text{para P ímpar} \tag{39a}$$

e

$$|a_u^{(P)}| = \frac{P+2}{2} \quad \text{para P par} \tag{39b}$$

Os seus termos intermediários são calculados pela seguinte fórmula de recorrência:

$$|a_i^{(P)}| = |a_1^{(P-1)}| + |a_{i-1}^{(P-2)}| \quad \text{onde } 1 < i < u^* \tag{40}$$

Por fim, os sinais do coeficientes serão sempre intercalados de forma que, se $a_i > 0$, $a_{i+1} < 0$.

Conhecidas essas leis de formação da equação característica, podemos construí-la para qualquer P, sem precisar calculá-la a partir das diversas matrizes de transição que, a medida que aumentam de ordem, tornam mais difíceis a obtenção dos autovalores.

Verificado o comportamento dos diversos valores de $\lambda_{max}(P)$, extraímos também a seguinte relação:

$$\lambda_{max}(P) = 2 + \sqrt{\lambda_{max}\left(\frac{P-2}{2}\right)} \quad , P \text{ para} \geq 6 \tag{41}$$

* Essa relação também é válida para i=1, desde que P > 4.

Daí, tiramos os valores:

$$\lambda_{max}(2) = 2$$
$$\lambda_{max}(6) = 2 + \sqrt{2}$$
$$\lambda_{max}(14) = 2 + \sqrt{2 + \sqrt{2}}$$
$$\lambda_{max}(30) = 2 + \sqrt{2 + \sqrt{2 + \sqrt{2}}}$$

e

$$\lambda_{max}(4) = 3$$
$$\lambda_{max}(10) = 2 + \sqrt{3}$$
$$\lambda_{max}(22) = 2 + \sqrt{2 + \sqrt{3}}$$
$$\lambda_{max}(46) = 2 + \sqrt{2\sqrt{2 + \sqrt{3}}}$$

Podemos então partir de um P inicial qualquer e será possível obter o valor de $\lambda_{max}(P)$ para todo P par, sem conhecer a respectiva equação característica.

Claramente, todas essas séries de λ_{max} tendem a 4 e, portanto, Δ_{min} tende a zero quando P tender a infinito. Como é razoável esperar, desbalanços muito grandes permitem que o excesso de velocidade seja tão pequeno quanto for desejado.

Se fizermos $\lambda_{max}(P) = 4-\varepsilon$, com ε pequeno e P grande, temos:

$$\lambda_{max}(2P) \quad 2+ \sqrt{\lambda_{max}(P)} = 2+ \sqrt{4-\varepsilon} \cong 4 - \frac{\varepsilon}{4} \qquad (42)$$

$$\Delta_{min}(P) = \frac{2}{\log_2 \lambda_{max}(P)} - 1 =$$

$$= \frac{2\ \ell n 2}{\ell n(4-\varepsilon)} - 1 \cong \frac{\varepsilon}{8\ell n 2} \qquad (43a)$$

e

$$\Delta_{min}(2P) = \frac{\varepsilon}{32\ell n 2} \qquad (43b)$$

Portanto, a duplicação de P resulta na divisão de $\Delta_{min}(P)$ por quatro. Conclui-se que $\Delta_{min}(P)$ tende a zero conforme P^{-2}, quando P tender a infinito.

Em suma, pode-se obter imediatamente $\lambda_{max}(P)$ e, em consequência, $\Delta_{min}(P)$, quando P for par pela equação (41). Para P ím-

par teríamos que construir a equação característica através da regras descritas e calcular sua maior raiz, o que não é difí - cil pois sabemos que será um número cada vez mais próximo de quatro.

II.7 - OS CÓDIGOS ÚTEIS E SUA VARIEDADE

Através das tabelas de cálculo de $R(n,d_{max},D)$, chega - mos aos códigos considerados como maximamente eficientes para um dado P e n. Estes códigos estão reunidos na tabela abaixo , onde levamos em conta apenas quatro níveis de complexidade ($n \leq 4,8,12$ e 16).

P	$\Delta_{min}(\%)/\ m_{max}B\text{-}nB$			
	$n \leq 4$	$n \leq 8$	$n \leq 12$	$n \leq 16$
2	100/1B-2B	100/1B-2B	100/1B-2B	100/1B-2B
3	200/1B-3B	66,7/3B-5B	57,1/7B-11B	57,1/7B-11B
4	33,3/3B-4B	33,3/3B-4B	33,3/3B-4B	27,3/11B-14B
5	200/1B-3B	40/5B-7B	37,5/8B-11B	30/10B-13B
6	33,3/3B-4B	20/5B-6B	20/5B-6B	16,7/12B-14B
7	50/2B-3B	40/5B-7B	28,6/7B-9B	25/12B-15B
8	33,3/3B-4B	20/5B-6B	20/5B-6B	14,3/14B-16B
9	-	40/5B-7B	22,2/9B-11B	15,4/13B-15B
10	33,3/3B-4B	14,3/7B-8B	11,1/9B-10B	11,1/9B-10B

Tabela II.14 - Códigos maximamente para $2 \leq P \leq 10$ e quatro níveis de complexidade. Os códigos úteis estão indicados por retângulos circunscritos.

Havendo, para um dado valor de P, empate na eficiência de dois ou mais códigos, consideramos o menos complexo dentre eles.

A análise efetuada nos leva às seguintes conclusões:

a) Nem todo aumento do desbalanço P gera alívio no excesso de velocidade Δ, podendo inclusive piorá-lo.

b) Nem todo aumento da complexidade melhora a eficiência.

Nessas condições, somente faz sentido a aplicação de alguns códigos que iremos chamar de códigos úteis.

Um código será definido como útil quando for mais eficiente do que todos os outros maximamente eficientes que forem mais balanceados e/ou menos complexos, isto é, não haverá nenhum outro cujo valores de Δ, P e n sejam, simultaneamente, menores do que o do código útil. Na tabela II.14 eles estão assinalados por retângulos circunscritos. A utilidade de um código dependerá dos níveis de complexidade considerados distintos.

Uma vez determinada a classe de códigos úteis para as condições especificadas de balanceamento, eficiência e complexidade, é preciso escolher um deles. Isso equivale a escolher $2^{m_{max}}$ palavras entre as $R(n,d_{max},D)$ disponíveis.

A quantidade C(m,n) de codificações possíveis (a menos , é claro, de permutações entre palavras) para um código mB-nB será então:

$$C(m,n) = \binom{R(n,d_{max},D)}{2^m} \quad , \; d_{max} \neq 0 \qquad (44a)$$

$$C(m,n) = \binom{1/2\ R(n,d_{max},D)}{2^{m-1}} , \; d_{max} = 0 \qquad (44b)$$

onde a distinção se deve a que no caso de alfabeto único ($d_{max}=0$), as palavras precisam ser incluídas ou excluídas juntamente com suas complementares (seção II.3).

A tabela II.15 mostra C(m,n) para os dez códigos úteis identificados para $P \leq 10$, $n \leq 16$. Em geral esses números são muito grandes, exceto nos casos de 1B-2B, 3B-5B, 3B-4B e 5B-6B . A escolha da codificação mais apropriada pode obedecer a vários critérios, Tais como minimização da flutuação RMS do nível de base, sincronizabilidade dos blocos, fator de multiplicação de erros de linha, etc.

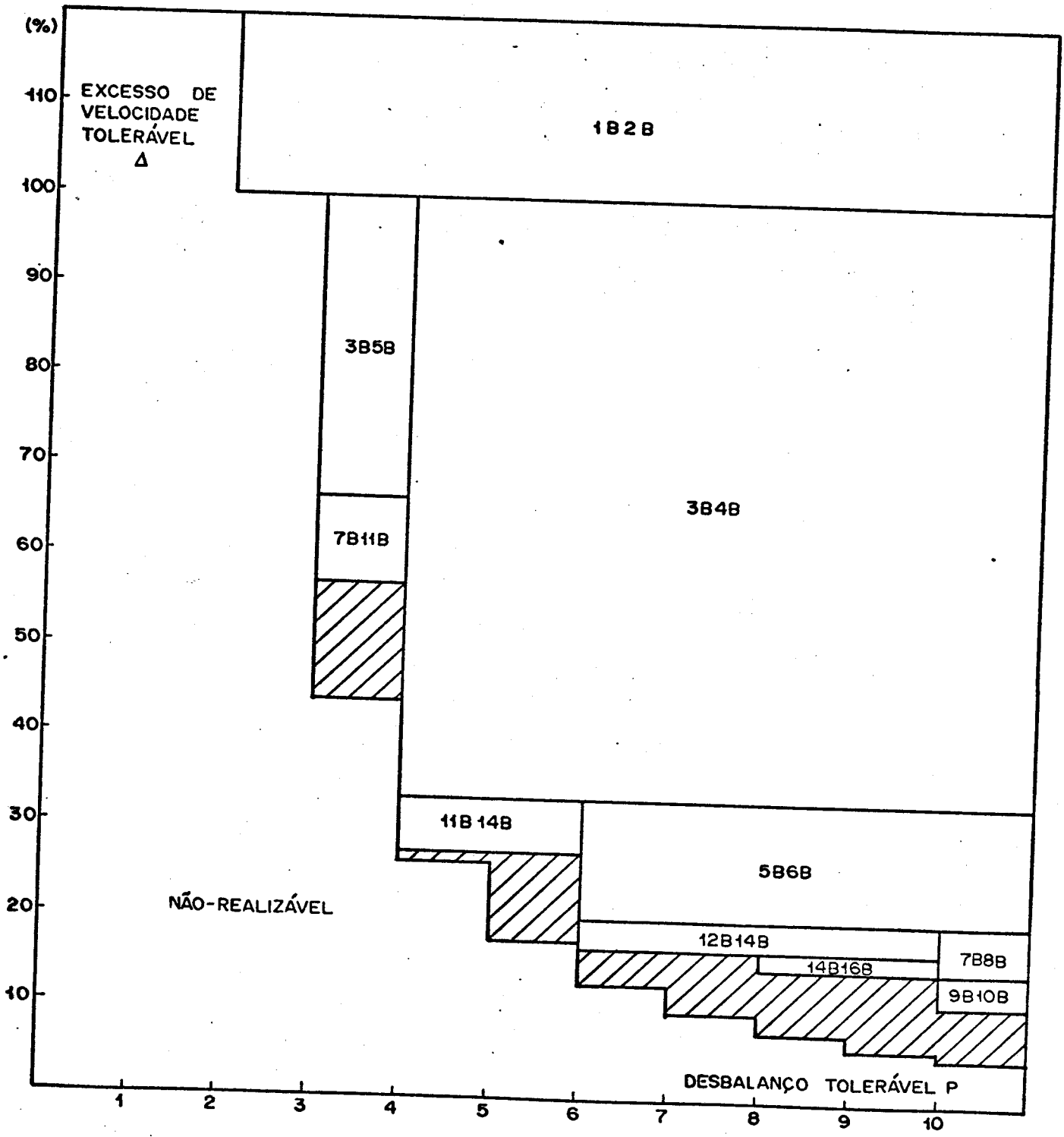


Figura II.5 - Os códigos úteis e suas regiões de utilidade, para quatro níveis da complexidade ($n \leq 4, 8, 12$ e 16). A área hachuriada só é realizável com códigos de alta complexidade ($n > 16$). A região inferior em branco ultrapassa os limites assintóticos de Δ para o desbalanço correspondente.

Código	$(d_{max},D)/P$	$R(n,d_{max},D)$	$C(m,n)$
1B-2B	(0,1)/2	2	1
3B-5B	(1,1)/3	8	1
7B-11B	(1,1)/3	144	$\binom{144}{122}$
3B-4B	(2.1)/4	9	$\binom{9}{8}$= 9
11B-14B	(2,1)/4	2.187	$\binom{2187}{2048}$
5B-6B	(2,2)/6	34	$\binom{34}{32}$=561
12B-14B	(2,2)/6	4.616	$\binom{4616}{4096}$
14B-16B	(2,3)/8	21.250	$\binom{21250}{16384}$
7B-8B	(4,2)/10	145	$\binom{145}{128}$
9B-10B	(4,2)/10	525	$\binom{525}{512}$

Tabela II.15 - A variedade dos códigos úteis.

II.8 - CONCLUSÃO:

A Figura II.5 apresenta, de maneira ilustrativa, um resumo das conclusões tiradas neste capítulo. Nela encontramos as regiões de utilidade de cada código útil, dada pelo conjunto de valores toleráveis de P e Δ no qual um determinado código útil é o menos complexo de todos. Cada código deverá operar nas condições relativas ao vértice inferior esquerdo de sua região de utilidade.

As regiões hachuriadas só são atingíveis com códigos de alta complexidade (n > 16) enquanto que a área inferior em branco não é realizável, pois ultrapassa os limites assintóticos de excesso de velocidade, calculados para cada desbalanço P. No apêndice II.A relacionamnos esses limites, obtidos através dos ábacos com os previstos pela teoria da informação.

A análise da figura nos permite uma escolha apropriada de um código, de acordo com as condições de velocidade, complexida-

de e balanceamento, requeridas pelo sistema em que se vai empregá-lo.

Em sistemas de baixa velocidade, o 1B-2B apresenta-se como a solução mais simples possível e com mínimo desbalanço. Para taxas mais altas, pode-se optar pelo 3B-4B que é bem mais eficiente e mais simples do que outros possíveis candidatos, como o 3B-5B e o 7B-11B.

No caso de 140 Mbits/s, o 3B-4B já apresenta um excesso de taxa relativamente alto (33%). Entretanto, além do problema já citado de sincronizabilidade, que iremos explicar depois, este código, com P=4, só possui nove palavras disponíveis para codificar oito blocos de entrada, condicionando, dessa forma, o uso de mais de um canal de serviço ao aumento do desbalanço para P=6.

Para diminuir o excesso de velocidade, poderíamos empregar o 11B-14B, mas essa classe de código possui uma complexidade demasiadamente alta. A melhor solução seria o 5B-6B que, com P = 6, permite um excesso de taxa de 20% e possibilita o uso de dois canais de serviço.

Outra solução de complexidade tolerável é a do 7B-8B, que reduz o excesso de faixa até 14,3% e eleva a excursão de SDC até P=10. Como já vimos no primeiro capítulo, essa diferença de aproximadamente 6% no excesso de taxa que o 7B-8B fornece em relação ao 5B-6B, permitiria um espaçamento um pouco maior entre os repetidores. Porém, além de ser um pouco mais complexo, o seu desbalanço maior provocaria um aumento do valor pico-a-pico das flutuações do nível de base.

Uma vez escolhida a classe de códigos a ser empregada no sistema, deve-se formar o conjunto de palavras que serão utilizadas na codificação. Um critério que nos ajuda a escolhe-las é a minimização da flutuação RMS do nível de base (a flutuação pico-a-pico permanece, pois está diretamente ligada a P). Por exemplo, no código 5B-6B, existem 34 palavras disponíveis para codificar as 32 de entrada. No alfabeto regressivo, essas 34 palavras são constituídas por 15 de disparidade -2 e 19 de disparidade 0 (exclue-se a palavra 111000, cuja disparidade intermediária excede D=2).

Reescrevendo abaixo o trecho da tabela II.8.b até n=6 , notamos, pela inspeção dos losangos mostrados, que existem ape

nas sete palavras disponíveis que fazem a SDC atingir os seus valores-limite: 110100, 110010, 101100, 011100, 110001 e 110000, que atingem o limite superior, e 000011 que chega até o inferior. Dessas, só a primeira atinge o limite mais que uma vez (no 2º e no 4º bits). Se quizermos então minimizar a probabilidade de ocorrência dos valores limites da SDC e, em consequência, da flutuação RMS, deveríamos exlcuir esta palavra e mais uma entre as outras seis, reduzindo bastante o número de codificações possíveis que era, inicialmente, 561. Insistindo na minimização do desbalanço, deveríamos excluir , entre as seis, uma das que passou mais vezes pelo penúltimo nível, conseguindo assim o código 5B-6B mais balanceado de todos. Parece mais razoável, entretanto, que a escolha final seja feita pelo critério de sincronizabilidade de blocos, que vai ser objeto do terceiro capítulo.

OBS	d \ n	1	2	3	4	5	6
D	2		1		3		9
	1	1		3		9	
*	0		2		6		19
	-1	1		3		10	
*	-2		1		4		15
	-3			1		5	
$-d_{max}$-D	-4				1		5

Figura II.5 - Tabela de cálculo de $R(n,d_{max},D)$ para P=6 e $(d_{max},D)=(2,2)$ até n=6.

Finalmente, uma vez estabelecidos os conjuntos de palavras-código que formarão os alfabetos, é preciso alocar a cada uma delas, uma palavra-fonte, obedecendo as restrições já colocadas sobre a estrutura da codificação. Esta alocação, que não é estudada neste trabalho, pode ser otimizada, juntamente com as regras de decodificação das palavras não utilizadas, no sentido de minimizar o fator de multiplicação dos erros de linha.

APÊNDICE II-A

LIMITES DE EFICIÊNCIA PREVISTOS PELA TEORIA DA INFORMAÇÃO

É interessante comparar os valores assintóticos de eficiência, obtidos na seção II.6 com os previstos pela teoria da informação para um sinal digital binário com soma digital corrida confinada.

Para um desbalanço P=2, existem três níveis de SDC, conforme mostrado na figura A.1.a- Nela verificamos que, a cada símbolos, um não carregará informação, pois terá probabilidade 1 de ocorrência. Daí, concluimos que a máxima informação média por símbolo será de 1/2 bit e ocorrerá quando $p = 1 - p = 0{,}5$. O código 1B-2B é, portanto, maximamente eficiente, de acordo com a teoria da informação, para uma excursão de SDC igual a dois.

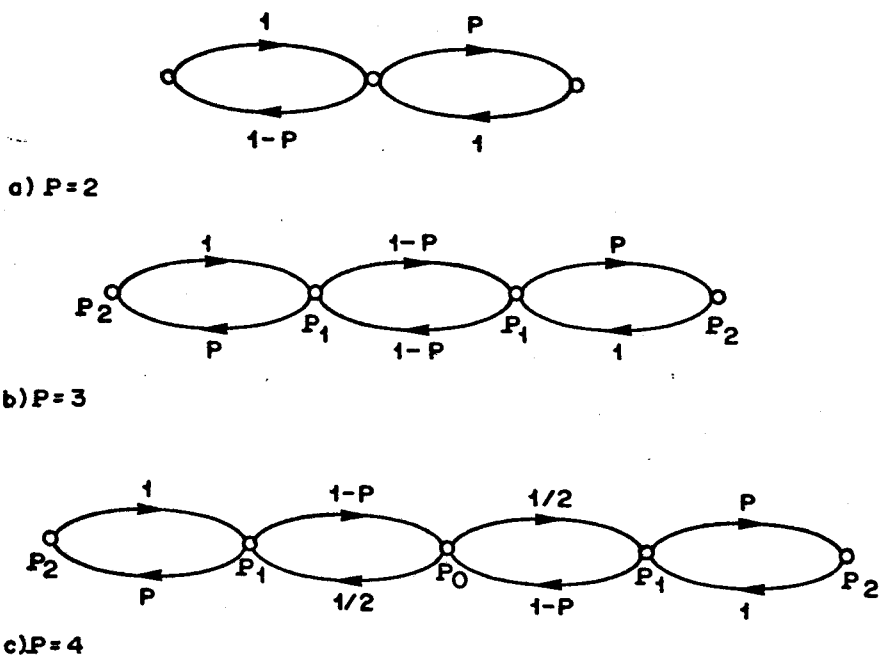


Fig. A.1 - Diagrama de estados de soma digital corrida.

Se P=3, existem quatro estados da SDC, onde P_1 é a probabilidade da SDC se encontrar nos níveis intermediários e P_2 nos níveis extremos do diagrama (fig. b). A probabilidade de transição é dada por 1,p e 1-p, conforme indica a figura.

Daí, temos

$$\begin{cases} P_2 = pP_1 \\ 2P_1 + 2P_2 = 1 \end{cases}, \quad P=3 \qquad (45a)$$

$$\therefore \quad P_1 = \frac{1}{2(1+p)}, \quad P=3 \qquad (45b)$$

A taxa de informação por símbolo é dada por:

$$\beta = 2P_1 \left[- p \log_2 p - (1-p) \log_2 (1-p) \right] \qquad (45c)$$

Substituindo (45b) em (45c), podemos encontrar a probabilidade de transição p_{max} que maximiza a taxa de informação por símbolo. Obtivemos:

$$p_{max} = \frac{3 - \sqrt{5}}{2} = 0,381, \quad P=3 \qquad (46a)$$

e

$$\beta_{max} = \frac{1}{2} \log_2 \frac{3 + \sqrt{5}}{2}, \quad P=3 \qquad (46b)$$

que é exatamente o valor encontrado para $\Delta_{min}(3)$, ou seja, a eficiência assintótica dos códigos mB-nB para P=3 tende ao mesmo valor previsto como máximo de eficiência para códigos com este desbalanço.

Analogamente, podemos dizer a p_{max} e β_{max} para P=4, partindo da figura B.1.c.

Temos:

$$\begin{cases} P_2 = pP_1 \\ P_1 = 1/2\ P_0 + P_0 \\ P_0 + 2P_1 + 2P_2 = 1 \end{cases}, \quad P=4 \qquad (47a)$$

$$\therefore \quad \begin{cases} P_0 = (1-p)/2 \\ P_1 = 1/4 \end{cases} \qquad (47b)$$

$$\therefore \quad \beta = \frac{1-p}{2} + \frac{1}{2} \left[-p \log_2 p - (1-p) \log_2 (1-p) \right] \qquad (48)$$

e, daí:

$$p_{max} = 1/3 \quad , P=4 \tag{49}$$

$$\beta_{max} = 1/2 \log_2 3 \quad , P=4 \tag{50}$$

Chegamos então que também para P=4, o excesso de taxa Δ_{min} tende ao mesmo valor do da teoria da informação. Intuimos portanto que os códigos mB-nB tendem assintoticamente ao máximo absoluto da eficiência previsto para qualquer desbalanço, a medida que aumentamos sua complexidade. Entretanto, não conseguimos a prova matemática dessa propriedade para todo P.

Nota-se ainda que, para P=2, a eficiência ótima não é alcançada assintoticamente, mas ocorre para os códigos 1B-2B-

CAPÍTULO III

SINCRONIZABILIDADE DE BLOCOS EM CÓDIGOS mB-nB

III.1 - INTRODUÇÃO

Conforme mencionamos anteriormente, a introdução de redundância através de uma codificação de linha deve ser feita de modo a permitir, entre outras coisas, que o sistema seja auto - sincronizável. No caso dos códigos mB-nB, o sistema estará sincronizado enquanto for mantido o alinhamento de blocos, isto é, quando os dígitos recebidos forem agrupados consecutivamente , formando blocos que correspondem exatamente às palavras-códigos enviadas na transmissão; dessa forma, estará garantida a correta decodificação do sinal. Esta operação é executada através de uma estratégia de alinhamento que nos informará se o sistema está ou não em sincronismo e, caso não esteja, procurará recuperá-lo o quanto antes.

A estratégia de alinhamento é montada com base na verificação de violação no código, que podem se dar pelo aparecimento de palavras-código que não fazem parte do conjunto escolhido para a tabela de codificação, as quais chamaremos palavras proibidas, ou por transgressões da própria lei do código, estabelecida para seus estados terminais. Essas violações podem, no entanto, ocorrer devido a erros de linha e a estratégia adotada deve discernir, o melhor possível, quando a violação for devido a esses erros ou se houve realmente uma perda de alinhamento. Existem então dois parâmetros que nos servem para avaliar a eficiência de uma estratégia:

a) Tempo de recuperação de alinhamento (Tr): É o tempo que o sistema demora para sincronizar-se novamente, após uma perda de alinhamento.

b) Tempo de retenção de alinhamento: (Te): É o tempo que o sistema levará para acusar uma perda de alinhamento inexistente, por causa dos erros de linha.

Esses valores são, na realidade, tempos médios e são calculados estatisticamente. Observamos, de imediato, a conveniên-

cia de se ter um tempo de recuperação bastante curto e um tempode retenção o mais longo possível.

Definindo por $P_v(s=1)$ a probabilidade de se detetar uma violação, estando o sistema defasado de i bits da posição correta, onde $0 \leq i < n$, concluímos que:

a) Devemos ter $P_v(s=0)$ muito pequena para garantirmos um tempo de retenção suficientemente grande. O valor de $P_v(s=0)$ é proporcional à probabilidade de erros de linha (Pe) que deve , obviamente, ser a menor possível. Para Pe=0 a probabilidade $P_v(s=0)$ seria nula também.

b) Devemos ter $P_v(s\neq 0)$ elevada para garantirmos um tempo de recuperação suficientemente pequeno. O seu valor depende, fundamentalmente, do conjunto de palavras-código escolhidas dentre as $R(n,d_{max},D)$ disponíveis para uma determinada classe. Para efeito de cálculo de $P_v(s\neq 0)$, iremos considerar nula a probabilidade de P_e de erros de linha.

Neste capítulo, desenvolvemos modelamentos e programas que nos permitem calcular $P_v(s=i)$ para i=1,2,...,n-1, seja para violações por palavra proibida como por violação da lei do código. Obtendo $P_v(s=i)$ e, consequentemente, o número de palavras-código que o sistema desalinhado necessita para detetar uma violação, é possível encontrar o tempo de recuperação para uma dada estratégia de alinhamento.

A análise, da tabela II.5 nos levou a concluir que o código 5B-6B é o mais adequado para a transmissão em 140 Mb/s, segundo os compromissos de balanceamento, complexidade e eficiência. Há ainda as opções do 7B-8B que é mais eficiente, porém menos balanceado e mais complexo, e do 3B-4B, que apresenta um excesso de velocidade maior do que o desejado. Os demais códigos úteis mostraram um excesso de taxa muito elevado (1B-2B , 3B-5B e 7B-11B) ou uma complexidade ainda excessiva (11B-14B , 12B-14B, 14B-16B e 9B-10B).

Por esta razão, os modelos aqui desenvolvidos são imediatamente aplicáveis a códigos do tipo (n-1)B-nB para n par , embora seus princípios sejam possíveis de ser extendidos a qualquer código mB-nB.

III.2 - SINCRONIZAÇÃO POR PALAVRAS PROIBIDAS:

III.2.a - Cálculos de $P_V(s=1)$

Os códigos de bloco do tipo (n-1)B-nB terão, necessariamente, dois alfabetos de codificação para o caso de n par e maior do que dois. Podemos comprovar isto supondo:

$\left.\begin{matrix} n = 4 \\ n-1 = 3 \end{matrix}\right\}$, daí tiramos que $2^3 > \binom{4}{2}$, o que, como vimos na seção II.3, obriga a existência de dois alfabetos. Fazendo-se $n' = n+2$, teremos:

$$2^{n'-1} = 4 \times 2^{n-1} \tag{51}$$

e

$$\binom{n'}{n'/2} = \frac{n'!}{n'/_2!\ n'/_2!} =$$

$$= \frac{(n+2)\ (n+1)}{(n/2+1)^2} \times \frac{n!}{n/2!\ n/2!} = \frac{n^2+3n+2}{n^2/4+n+1} \times \binom{n}{n/2} \tag{52}$$

Podemos verificar que:

$$\frac{n^2+3n+2}{n^2/4+n+1} = 4\,\frac{n^2+3n+2}{n^2+4n+4} < 4 \quad \text{p/ } n > 0 \tag{53}$$

Concluímos então que, para o 3B-4B, não é possível a codificação com um só alfabeto, pois $2^{n-1} > \binom{n}{n/2}$. Para os demais códigos (n-1)B-nB esta relação vai se manter pois o primeiro termo crescerá mais do que o segundo. Logo, estes códigos terão sempre dois alfabetos de codificação e sua lei será descrita pelo diagrama geral de estados, dado na Fig. II.2.

Para mostrar o procedimento de cálculo de $P_V(s)$ por palavra proibida, vamos tomar como exemplo um código 5B-6B dado pela tabela III.1.

Estando o decodificador desalinhado por i bits as palavras recebidas não serão as mesmas que foram enviadas na trans-

PALAVRAS-FONTE	ALFABETO S1 (Progressivo)	d	ALFABETO S2 (Regressivo)	d
00000	000111	0	000111	0
00001	001011	0	001011	0
00010	001101	0	001101	0
00011	001110	0	001110	0
00100	010011	0	010011	0
00101	010101	0	010101	0
00110	010110	0	010110	0
00111	011001	0	011001	0
01000	011010	0	011010	0
01001	011100	0	011100	0
01010	100011	0	100011	0
01011	100101	0	100101	0
01100	100110	0	100110	0
01101	101001	0	101001	0
01110	101010	0	101010	0
01111	101100	0	101100	0
10000	110001	0	110001	0
10001	110010	0	110010	0
10010	110100	0	110100	0
10011	111000	0	111000	0
10100	010111	+2	101000	-2
10101	011011	+2	100100	-2
10110	011101	+2	100010	-2
10111	100111	+2	011000	-2
11000	101011	+2	010100	-2
11001	101101	+2	010010	-2
11010	101110	+2	010001	-2
11011	110011	+2	001100	-2
11100	110101	+2	001010	-2
11101	110110	+2	001001	-2
11110	111001	+2	000110	-2
11111	111010	+2	000101	-2

Tabela III.1 - Exemplo de um Código 5B-6B

missão, e sim, geradas a partir destas. O receptor irá entender como uma palavra-código o conjunto de bits formado pelos (n-i) últimos associados aos i primeiros bits de duas palavra-códigos enviadas consecutivamente.

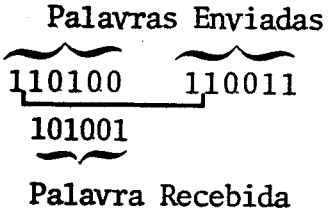


Fig. III.1 - Exemplo de sistema defasado de 1 bit (s=1)

No exemplo acima, o sistema deveria decodificar a primeira palavra enviada como sendo 10010 (vide tabela), devido ao deslocamento, a palavra recebida será decodificada como 01101, havendo, portanto, cinco dígitos errados no sinal. Se a palavra gerada for permitida, não haverá violação e este erro passará despercebido pelo sistema. Nosso intuito, nesta seção, é buscar a tabela que nos forneça os maiores valores de $P_V(s=i)$, permitindo ao sistema detetar e recuperar o sincronismo perdido rapidamente(baixo valor de tr).

Para calcular a $P_V(s=1)$ para a codificação tomada como exemplo, geramos, para cada uma das palavras permitidas, novas palavras, adicionando dígitos 0 e 1 ao seu final, conforme indicado na tabela III.2. Nela, indicamos por um "V" os casos em que surge uma palavra-proibida e por "N" nos casos contrários.

Podemos escrever que:

$$P_V(s=1) = \sum_{i=1}^{N_V} P_i \times p_i(\ell) \quad , \text{ onde} \qquad (54)$$

N_V = número de violação contadas, no caso $N_V = 10$

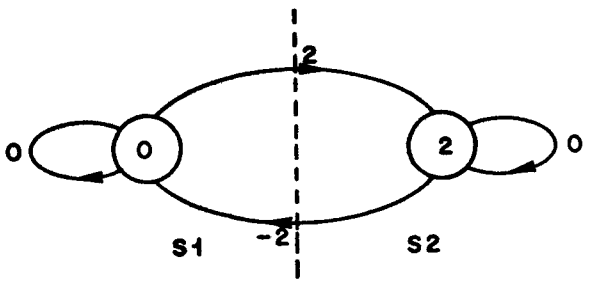
P_i = probabilidade de aparecer a i-ésima palavra código que gera violação.

$p_i(\ell)$ = probabilidade de que o bit cinsecutivo a essa palavra seja o que gera a proibida (ℓ = 0 e/ou 1).

PALAVRAS-PRESENTES EM S1 E S2	0 ↓	1 ↓	PALAVRAS-PRESENTES EM S1	0 ↓	1 ↓	PALAVRAS-PRESENTES EM S2	0 ↓	1 ↓
000111	N	V	010111	N	V	101000	V	N
001011	N	N	011011	N	V	100100	V	N
001101	N	N	011101	N	V	100010	V	N
001110	N	N	100111	N	V	011000	V	N
010011	N	N	101110	N	N	010001	N	N
010101	N	N	101011	N	N	010100	N	N
010110	N	N	101101	N	N	010010	N	N
011001	N	N	110011	N	N	001100	N	N
011010	N	N	110101	N	N	001010	N	N
011100	N	N	110110	N	N	001001	N	N
100011	N	N	111001	N	N	000110	N	N
100101	N	N	111010	N	N	000101	N	N
100110	N	N						
101001	N	N						
101010	N	N						
101100	N	N						
110001	N	N						
110010	N	N						
110100	N	N						
111000	V	N						

Tabela III.2 - Aparecimento de palavras proibidas para s = 1 bit.

No caso analisado, a probabilidade de se estar num alfabeto ou noutro é a mesma e a comutação é dada por:



Então, palavras presentes nos dois alfabetos, como são equiprováveis, terão probabilidade $P_i = 1/32$ de ocorrência, enquanto as que se apresentam num alfabeto apenas teraõ $P_i=1/64$. Observa-se tamabém que as palavras presentes em S1 e S2 podem ser seguidas por 0 ou 1, com a mesma probabilidade, uma vez que não se pode precisar com qual alfabeto estamos lidando e o número total de palavras iniciadas por zero ou por um é o mesmo.

Por outro lado, se estamos lidando com uma palavra presente apenas em S1, por exemplo, sabemos que, conforme sua disparidade seja 0 ou +2, a palavra seguinte fará parte de S1 mesmo ou de S2. No nosso exemplo, essas palavras terão sempre d=+2 e serão seguidas por alguma do alfabeto S2 onde temos $p(\ell=0)=19/32$ e $p(\ell=1)= 13/32$. Analogamente, em S1 temos $p(\ell=0)=13/32$ e $p(\ell=1)=19/32$.

A expressão (54) pode ser reduzida à seguinte forma:

$$P_V(s=1) = P_0 + P_2 \quad , \text{ onde} \tag{55a}$$

$$P_0 = \sum_{i=1}^{N'_V} P'_i \times p'_i(\ell) \tag{55b}$$

e

$$P_i = \sum_{i=1}^{N''_V} P''_i \times p''_i(\ell) \tag{55c}$$

Onde os termos de P_0 e P_i correspondem aos da equação (54) e dizem respeito, em particular, às palavras de disparidades 0 e $\pm$ 2, respectivamente. Temos, então:

$$N'_V = 2 \quad , P_i = 1/32 \quad \text{e} \quad p_i(\ell=0,1)=1/2 \tag{56a}$$

$$N''_V = 8,\ P''_i = 1/64 \quad e \quad \begin{cases} p_i(\ell=0)=13/32 \text{ em S1} \\ p_i(\ell=1)=19/32 \text{ em S2} \end{cases} \qquad (56b)$$

sendo que as violações sempre ocorreram pela introdução de um zero após uma palavra de disparidade -2 ou pela introdução de 1 após uma de disparidade +2. Aliás, como os nossos códigos serão sempre simétricos, podemos escrever:

$$P_2 = 2 \sum_{i=1}^{N''_V/2} P''_i \times p''_i(\ell) = 2\left(\frac{N''_V}{2}\right) P''_i\, p_i(\ell) \qquad (57)$$

levando em conta apenas as palavras contidas só em S1 ou só em S2.

Chegamos então a:

$$P_V(s=1)=N'_V\, P'_i p'_i(\ell)+2\left(\frac{N''_V}{2}\right) P''_i\, p''_i(\ell) = 2 \times \frac{1}{32} \times \frac{1}{2} + 2x4x \frac{1}{64} \times \frac{13}{32} =$$

$$= \frac{1}{32} + \frac{13}{256} = \frac{21}{256} \cong 8{,}203\%$$

Esta probabilidade pode ser considerada baixa, de modo que é conveniente modificar a tabela para melhorar a sincronizabilidade do código. Não estamos nos preocupando, por enquanto, com o balanceamento e, por isso, as codificações tomadas como exemplo não correspondem àquelas que nos fornecem um desbalanço mínimo (P=6) para o 5B-6B. Tendo, posteriormente, um modelo geral e um programa para o cálculo de $P_V(s=1)$, calcularemos esses valores e, comparando com os de outras codificações menos balanceadas, estabeleceremos um compromisso entre sincronizabilidade e balanceamento.

Restringindo-nos ainda a palavras permitidas de disparidades 0,+2 e -2, podemos aumentar $P_V(s=1)$, com base nos seguintes princípios:

a) Dentre as 15 palavras de disparidade +2, teremos que proibir três. É conveniente que essas três palavras não gerem uma a outra pois estaríamos perdendo oportunidade de gerar proibidas a partir das permitidas.

b) Como as palavras de disparidade +4 são proibidas e podem ser geradas a partir de outras de disparidade +2, convém escolher essas que geram disparidade +4.

c) Convém permitir também as palavras de disparidade +2 geradas a partir de disparidade +4 e proibir, preferencialmente, as que possam ser geradas a partir de palavras de disparidade 0 e +2.

As palavras permitidas de disparidade negativa satisfazem automaticamente esses princípios, uma vez que serão as complementares das já escolhidas de disparidade positiva.

Segundo essas condições, proibimos as palavras: 000110, 111001, 001010, 110101, 010010 e 101101. Construindo uma tabela semelhante à Tabela III.2, chegamos a que:

$$P_0 = N'_v\ P'_i\ p'_i(\ell=0,1) = 6 \times \frac{1}{32} \times \frac{1}{2} = \frac{3}{32} \qquad (58a)$$

$$P_2 = 2 \times \left(\frac{N''_v}{2}\right) \times P''_i \times p''_i\ (\ell=1) = 2 \times 8 \times \frac{1}{64} \times \frac{15}{32} = \frac{15}{128} \qquad (58b)$$

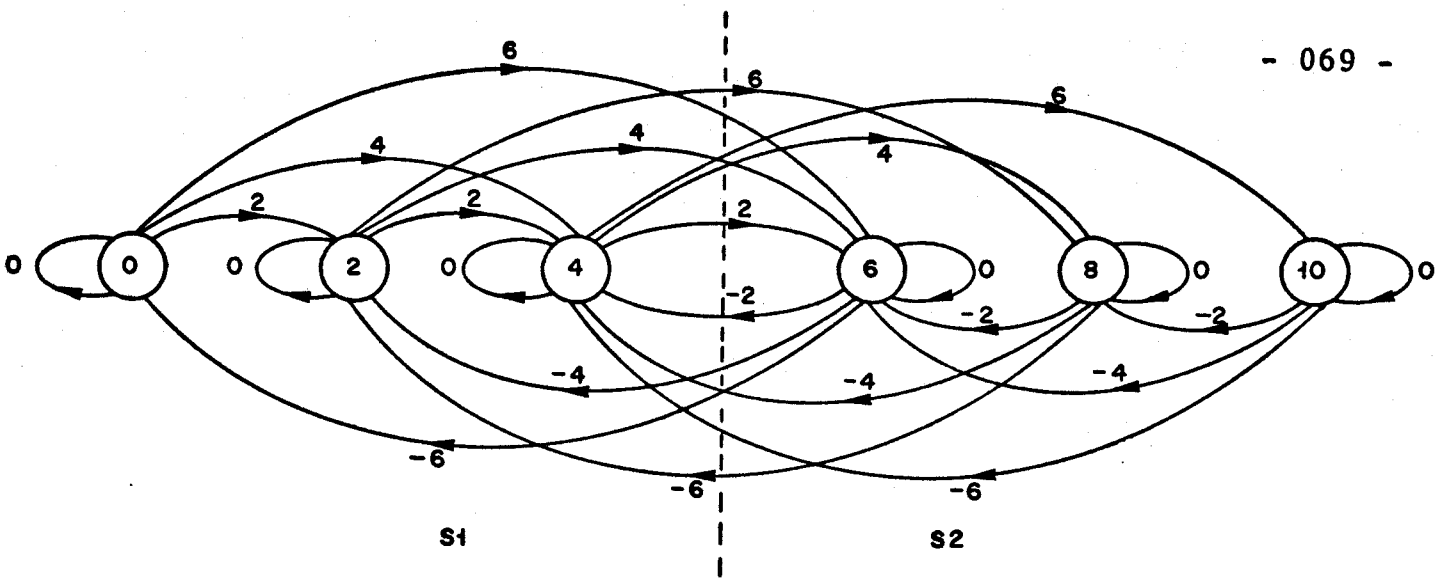
$$P_v(s=1) = \frac{3}{32} + \frac{15}{128} = \frac{27}{128} = 21{,}09\% \qquad (59)$$

Este valor, embora sensivelmente melhor, ainda não é o máximo $P_v(s=1)$ possível, pois existem palavras (p.exemplo, a palavra 111111) que nunca serão geradas a partir das palavras de disparidades 0, +2 e -2, para 1 bit de deslocamento.

Para s=1 bit, cada palavra pode ser gerada duas vezes. Então, para maximizarmos $P_v(s=1)$, devemos fazer com que todas as palavras proibidas sejam geradas duas vezes, sempre e partir de palavras permitidas.

Por exemplo, selecionamos como proibidas metade das palavras de disparidade 0, permitindo todas as disparidades ± 2, ± 4 e ± 6. Detetamos assim, as violações apontadas na tabela III.3.

Neste caso, temos $d_{max}=6$ e a lei da codificação para os estados terminais é a seguinte:



As probabilidades q_d de ocorrer uma palavra-código de disparidade ± d, dado que estamos no alfabeto correspondente (S1 para +D e S2 para -D) são:

$q_0 = 10/32$ $\quad q_4 = 6/32$

$q_2 = 15/32$ $\quad q_6 = 1/32$

PALAVRAS DE D = 0	0 ↓	1 ↓	PALAVRAS DE D = 2	0 ↓	1 ↓	PALAVRAS DE D = +4 e +6	0 ↓	1 ↓
000111	V	N	111100	V	N	111110	N	N
001011	V	N	111010	N	N	111101	N	N
001101	V	N	111001	N	N	111011	N	N
010011	V	N	110110	N	N	110111	N	N
011100	V	N	110101	N	N	101111	N	N
101010	N	V	110011	V	N	011111	N	N
101100	N	V	101110	N	N	111111	N	N
110001	N	V	101101	V	N			
110010	N	V	101011	V	N			
110100	N	V	100111	V	N			
			011110	N	N			
			011101	N	N			
			011011	N	N			
			010111	N	N			
			001111	N	N			

Tabela III.3

Com elas, podemos obter a probabilidade p_j de se estar num estado de SDCT=j, teremos:

$$\begin{cases} p_0 = p_0\, q_0 + p_4\, q_6 \\ p_2 = p_0\, q_2 + p_2(q_0+q_1) + p_4\, q_4 \\ p_4 = p_4(q_0+q_2) + p_2(q_2+q_4) + p_0(q_4+q_6) \end{cases} \qquad (60a)$$

e $$p_0 + p_2 + p_4 = 1/2 \quad , \qquad (60b)$$

sabendo que, por simetria, $p_0 = p_{10}$, $p_2 = p_8$ e $p_4 = p_6$.

Resolvendo o sistema, encontramos :

$$p_0 = p_{10} = 0{,}0167$$

$$p_2 = p_8 = 0{,}1169$$

$$p_4 = p_6 = 0{,}3674 \qquad (61)$$

Seguindo o mesmo raciocínio, teremos:

$$P_0 = N'_v\, P'_i\, p'_i(\ell=0,1) = 10 \times \frac{1}{32} \times \frac{1}{2} = \frac{5}{32} \qquad (62)$$

e

$$P_2 = 2 \left(\frac{N''_v}{2}\right) P''_i \quad p''_i(\ell=0) \qquad (63)$$

Desta vez, porém, $p''_i(\ell=0)$ irá depender do estado da SDCT em que o sistema se encontra, pois, após uma palavra de disparidade +2, o sistema poderá ou não comutar de alfabeto. Se tivermos SDCT=0 ou 2, o alfabeto S1 permaneceria e teríamos $p''_i(\ell=0)$= 11/32; caso estivéssemos com a SDCT=4, uma palavra de disparidade de dois seria necessariamente seguida por uma outra do alfabeto S2, onde $p''_i(\ell=0)$= 21/32.

Temos ainda que a probabilidade de se estar nos estados de SDCT=0,2 e 4, dado que se está num dos três, pois estamos lidando com S1, será:

$$p_0/S_1 = \frac{p_0}{p(S1)} = \frac{0{,}0167}{1/2} \qquad (64a)$$

$$p_2/S1 = \frac{p_2}{p(S1)} = \frac{0,1169}{1/2} \qquad (64b)$$

$$p_4/S1 = \frac{p_4}{p(S1)} = \frac{0,3674}{1/2} \qquad (64c)$$

e daí, obtemos:

$$p_i''(\ell=0) = 2x\ (0,0167+0,1169)x\ \frac{11}{32} + 0,3674\ x\ \frac{21}{32} \qquad (65a)$$

enquanto que:

$$P_i'' = \frac{1}{64} \quad e \quad \frac{N_V''}{2} = 5 \qquad (65b)$$

Portanto,

$$P_2 = 2\ x\ \frac{N_V''}{2}\ x\ P_i''\ x\ p_i(\ell=0) = 0,090$$

e

$$P_V(s=1) = P_0 + P_2 = 24,63\% \qquad (66)$$

Embora a melhoria seja pequena em relação ao resultado anterior (eq. 59), notamos que a perda de balanceamento favoreceu a sincronizabilidade.

Esses valores que obtivemos não são, todavia, decisivos, pois é necessário conhecer $P_V(s=i)$, com i=2,3,4,5, para que possamos julgar realmente a sincronizabilidade de uma dada codificação e obtermos o tempo de recuperação de sincronismo. O cálculo desses valores torna-se, porém, muito complexo para defasagens maiores do que um bit, principalmente se aumentarmos n (como no 7B-8B, por exemplo).

Desenvolvemos então um programa que calcula $P_V(s=2)$, com violação por palavra proibida, para qualquer valor de i e com um código do tipo (n-1)B-nB onde n é par. Esse programa está baseado no modelamento que descrevemos a seguir.

III.2.b - Modelo geral para a Obtenção de $P_V(s)$

Considerando M o valor decimal de uma palavra binária de n bits, pertencente ao conjunto de palavras escolhidas para a codificação e M* o de uma outra palavra, gerada a partir das permiti

das, por causa de uma defasagem de s bits no alinhamento da recepção, teremos que os diversos valores de M* para cada palavra M serão:

$$M^*_\ell = 2^s \left[M \bmod 2^{(n-s)} \right] + \ell \,, \ell = 0,1,2,\ldots 2^s-1 \qquad (65)$$

Por sua vez, o conjunto de palavras M_ℓ que podem gerar uma determinada M* será:

$$M_\ell = \frac{M^* - (M^* \bmod 2^s)}{2^s} + \ell \times 2^{(n-s)} \,, \ell=0,1,2,\ldots 2^s-1 \qquad (66)$$

Onde, na expressão (65) o primeiro termo da soma corresponde ao valor decimal restante de uma palavra binária M, após a retirada de seus s bits mais significativos, multiplicado por 2^s, pois esses bits constantes serão deslocados e passarão a ser os mais significativos. O termo ℓ pode assumir os diversos valores dados pelas possíveis combinações dos s bits que serão adicionados ao final da palavra M*.

Por exemplo, se n=6 e s=2 e temos:

M = 010011 = 19,

então temos, para s=2: 01 ¦ 0011 ($19 \bmod 2^4=3$) e, deslocando a esquerda:

0011 □□ (ℓ) onde ℓ poderá valer ($3 \times 2^2=12$)

- 00 = 0
- 01 = 1
- 10 = 2
- 11 = 3 = 2^s-1

Daí, as palavras geradas serão:

$M^*_0 = 12 + 0 = 12 = 001100$

$M^*_1 = 12 + 1 = 13 = 001101$

$M^*_2 = 12 + 2 = 14 = 001110$

$M^*_3 = 12 + 3 = 15 = 001111$

A expressão (66) é obtida se imaginamos o processo inverso. isto é, dada uma palavra M*, chegar às M_ℓ palavras que a podem

gerar com um deslocamento de s bits.

Sabemos também que a comutação de alfabetos de um código (n-1)B-nB é dada pelo diagrama da figura II.2, de onde tiramos que o sistema fica no alfabeto S1 para SDCT=0,2,4,...d_{max}-2 e no S2 para SDCT=d_{max}, d_{max}+2, ...2(d_{max}-1).

As probabilidades q_d da ocorrência das diversas disparidades são:

$$q_d = \frac{\text{nº de palavras de disparidades d existentes}}{2^{n-1}} \quad (67)$$

para d = 0,2,4,... d_{max}, dado que se está em S1, sendo claro que $q_d = q_{-d}$.

Com elas, podemos conseguir os valores de p_j para estados de SDCT=j construindo um sistema de maneira análoga ao das equações (60), só que genérico.

Teremos:

$$p_0 q_0 + p_{d_{max}} q_{d_{max}} = p_0$$

$$p_0 q_2 + p_2 q_0 + p_{d_{max}} q_{d_{max}-2} + p_{d_{max}+2} q_{d_{max}} = p_2$$

$$p_0 q_j + p_2 q_{j-2} + \ldots + p_j q_0 + p_{d_{max}} q_{d_{max}-j} + p_{d_{max}+2} q_{d_{max}+2-j} + \ldots$$

$$\ldots + p_{d_{max}+j} q_{d_{max}} = p_j \quad (68a)$$

e ainda

$$p_0 + p_2 + p_4 + \ldots p_{d_{max}-2} = 1/2$$

Na forma matricial, o sistema fica:

$$[Y] \times \begin{bmatrix} p_0 \\ p_2 \\ p_4 \\ \vdots \\ p_{d_{max}-2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1/2 \end{bmatrix} \Longrightarrow \begin{bmatrix} p_0 \\ p_2 \\ p_4 \\ \vdots \\ p_{d_{max}-2} \end{bmatrix} = [Y]^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1/2 \end{bmatrix} \quad (69)$$

O produto do segundo termo da expressão acima resulta na última coluna da matriz $[Y]^{-1}$,

$$\therefore \quad \begin{bmatrix} p_0 \\ p_2 \\ \vdots \\ p_{d_{max}-2} \end{bmatrix} = 1/2 \begin{bmatrix} y^{-1}_{1,d_{max}/2} \\ y^{-1}_{2,d_{max}/2} \\ \vdots \\ y^{-1}_{d_{max}/2\,,\,d_{max}/2} \end{bmatrix} \text{, onde} \qquad (70)$$

$$[Y] = \begin{bmatrix} q_0^{-1} & 0 & 0 & \cdots & 0 \\ q_2 & q_0^{-1} & 0 & \cdots & 0 \\ q_4 & q_2 & q_0^{-1} & \cdots & 0 \\ \vdots & \vdots & & & \\ q_{d_{max}-4} & q_{d_{max}-6} & q_{d_{max}-8} & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix} +$$

$$\begin{bmatrix} 0 & \cdots & 0 & 0 & q_{d_{max}} \\ 0 & \cdots & 0 & q_{d_{max}} & q_{d_{max}-2} \\ 0 & \cdots & q_{d_{max}} & q_{d_{max}-2} & q_{d_{max}-4} \\ \vdots & & & & \vdots \\ 0 & \cdots & q_8 & q_6 & q_4 \\ 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \qquad (71)$$

Por fim, sabemos que:

$$\begin{bmatrix} p_0 \\ p_2 \\ \vdots \\ p_{d_{max}-2} \end{bmatrix} = \begin{bmatrix} p_{2d_{max}-2} \\ p_{2d_{max}-4} \\ \vdots \\ p_{d_{max}} \end{bmatrix} \qquad (72)$$

Com os valores de $p_j = p(SDCT=j)$, partimos para o cálculo das probabilidades de violação que são dadas por:

$$P_V(s) = \sum_{i=1}^{N_V} P_i \times p_i(\ell) \quad , \text{ onde} \qquad (73)$$

N_V = número de violações que aparecem após aplicarmos a equação (65) em todas as palavras permitidas

P_i = probabilidade de ocorrência da palavra que gera a i-ésima violação

$p_i(\ell)$= probabilidade de ocorrência do valor de ℓ (eq. 65) que gera a i-ésima violação.

Vamos considerar que estamos, inicialmente, no alfabeto S1. Sendo assim, escrevemos:

$$P_V(s)/S1 = \sum_{i=1}^{N'_V} P'_i \times p'_i(\ell) \qquad (74)$$

onde estamos aplicando a equação (65) e contando as violações apenas nas palavras do alfabeto S1. As probabilidades P'_i e $p'_i(\ell)$ são condicionadas ao fato de estarmos lidando com esse alfabeto. Então:

$$P'_i = \frac{1}{2^{n-1}} \text{ para qualquer palavra} \qquad (75)$$

e $p'_i(\ell)$ será a probabilidade do valor de ℓ gerar violação sendo que estávamos anteriormente em S1. A palavra-código seguinte , que irá nos fornecer o valor de ℓ , poderá estar em qualquer um dos dois alfabetos, conforme a palavra anterior tiver provocado ou não uma comutação.

Se a palavra anterior (geradora) tem uma disparidade d, a probabilidade de ter havido uma comutação será:

$$C = 2 \left[p_{d_{max}-d} + p_{d_{max}+2-d} + \cdots + p_{d_{max}-2} \right] , \qquad (76)$$

isto é nada mais do que a probabilidade do sistema ter sido encontrado em estado de $d_{max} - d \leq SDCT \leq d_{max}-2$, dado que estava na região de aplicabilidade de S1.

A probabilidade de não haver comutação será $\overline{C} = 1 - C$.

Para encontrarmos $p'_i(\ell)$ teríamos então que contar, em cada alfabeto, quantas vezes ocorre o valor para os primeiros s bits de cada palavra permitida e, dividindo este número pelo total 2^{n-1} de palavras de cada alfabeto, achar as probabilidades:

$$p'_i(\ell)/S1,2 = \frac{\text{nº de vezes que os s primeiros bits resultam } \ell \text{ no alf.S1,2}}{2^{n-1}} \tag{77}$$

assim, obteríamos finalmente:

$$p'_i(\ell) = p'_i(\ell)/S1 \times \overline{C} + p'_i(\ell)/S2 \times C \tag{78}$$

Com um programa podemos, sem grandes problemas, obter P'_i, $p'_i(\ell)$ e, aplicando (eq. 65) nas palavras-código de S1, obter N'_v. Estaria calculado o valor $P_v(s)/S1$.

Por fim, temos que:

$$P_v(s) = P_v(s)/S1 \times prob(S1) + P_v(s)/S2 \times prob(S2) =$$

$$= \frac{1}{2} P_v(s)/S1 + \frac{1}{2} P_v(s)/S2 \tag{79}$$

Porém, é fácil mostrar que se as palavras do alfabeto progressivo e regressivo forem complementares ou idênticas, teremos:

$$P_v(s)/S1 = P_v(s)/S_2 \quad \text{e, daí :}$$

$$P_v(s) = P_v(s)/S1 = \sum_{i=1}^{N'_v} p_i(\ell) \times \frac{1}{2^{n-1}} \tag{80}$$

Todo este procedimento de cálculo, que será executado por computador, pode ser resumido pelo seguinte roteiro:

1) Calcula-se os valores de $q_d (0 \leqslant d \leqslant d_{max})$ e, consequentemente de p_j $(0 \leqslant j = SDCT \leqslant d_{max}-2)$.

2) Aplica-se a equação (65) sobre todas as palavras do alfabeto progressivo, para todos os valores de ℓ.

3) Para os valores de ℓ que provocarem uma violação, obtemos $p'_i(\ell)$ através da contagem desses valores de ℓ existentes em

cada alfabeto e do cálculo de C e $\overline{C}$ para a palavra-código que gerou a violação.

4] Após percorremos todas as palavras-código, somamos os $p'_i(\ell)$ obtidos e dividimos por $1/2^{n-1}$.

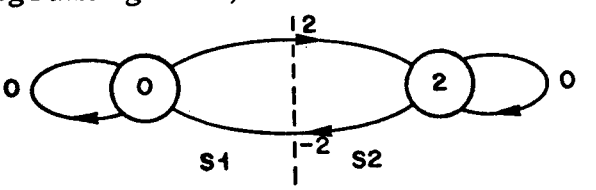
Caso os códigos não tivessem as condições de simetria impostas no Capítulo II, poderíamos refazer o procedimento para S2 e aplicar, no fim, a equação (79).

Com isto, conseguimos $P_V(s)$ para qualquer defasagem, independentemente do valor de n. No apêndice III.A, apresentamos o programa CODJM.FOT, desenvolvido sobre este modelamento.

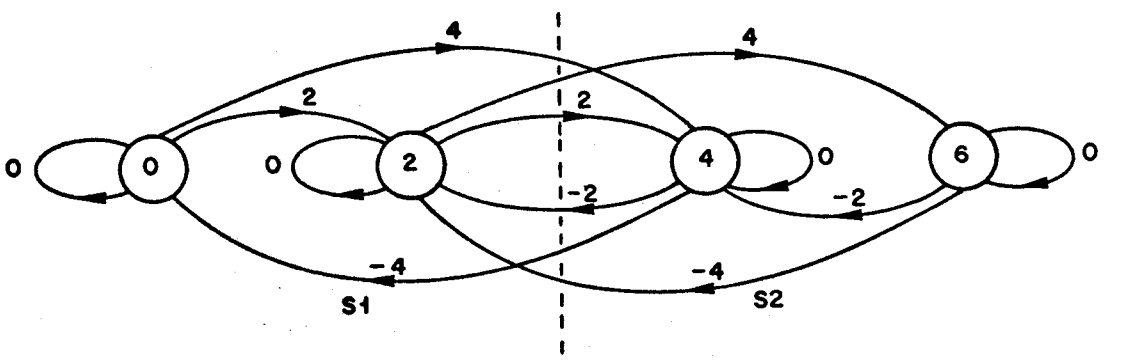
III.3 - SINCRONIZAÇÃO POR VIOLAÇÃO DA LEI DO CÓDIGO

III.3.a - Diagramas de deteção de transições proibidas

Conforme já comentamos, um código (n-1)B-nB, com n par, irá possuir um diagrama de estados de SDCT de acordo com o diagrama geral apresentado na Fig. II.2. Os casos de interesse prático para nós serão os de $d_{max}=2$, que engloba o 3B-4B e o 5B - 6B maximamente balanceados (P=4 e P=6) e de $d_{max}=4$, onde se aplica os códigos 7B-8B e 9B-10B, também maximamente balanceados (P=10). Nestes casos, a lei do código será uma particularização daquele diagrama geral, de acordo com as figuras abaixo:



a) dMAX=2 (3B-4B e 5B-6B)



b) dMAX = 4 (7B-8B e 9B-10B)

Fig. III.2 - Diagramas de estados da SDCT para códigos (n-1)B-nB.

Uma perda de sincronismo provoca, geralmente, uma violação nesta lei, isto é, ocorrerá uma transição imprevista pelo diagrama, que podemos chamar de transição proibida. Essas transições serão devidas ao aparecimento de uma palavra cuja disparidade não é compatível com o estado de SDCT que o sistema, desalinhado, supunha estar.

Nossa intenção é desenvolver um modelo que calcule a $P_v(s=i)$ por este critério e, com esse valor, obter o número médio de palavras-código necessárias para detetarmos uma violação com o sistema desalinhado.

É preciso discernir dois aspectos diferentes: Na transmissão existem os estados reais de SDCT, através dos quais é executada a comutação dos alfabetos, de acordo com as palavras-código que vão sendo enviadas; por outro lado, na recepção, se o sistema estiver fora de sincronismo, irão aparecer outros valores de SDCT imprevistos, gerando o que iremos chamar de estados de observação, observados apenas na recepção quando se perde o alinhamento. Estes estados ocorrem porque a sequência de palavras-código, uma vez defasada, gera disparidades diferentes das previstas na codificação.

Faz-se necessária, é claro, uma estratégia, pois um erro de linha também pode gerar uma disparidade e, em consequência, uma transição proibida entre estados. Detetaremos, por tanto, a falta de sincronismo quando o número de transições proibidas for tal que comprove que estamos lidando com estados de observação diferentes dos reais, gerados pelo desalinhamento do sistema.

Após uma perda de sincronismo, o sistema necessita se autolocalizar, ou seja, ele espera a chegada de uma palavra-código cuja disparidade defina o estado de SDCT em que ele se encontra. Esse estado não corresponde ao real e, assim, haverá posteriormente uma transição proibida. Criamos então um diagrama que representa esse procedimento.

Na figura III.3, as transições do estado inicial aos intermediários representa a auto-localização, pois o aparecimento de uma palavra de disparidade +2 e -2 obriga o sistema a ir pa-

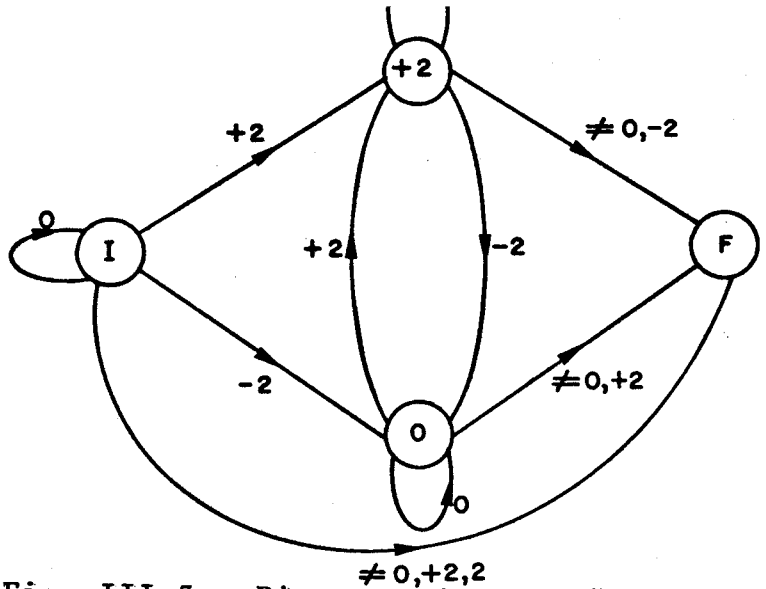


Fig. III.3 - Diagrama de deteção de transições proibidas para $d_{max}=2$.

ra os estados de SDCT igual a dois e zero, respectivamente. As transições entre os estados de SDCT=0,2 são permitidas e, até aí, não terá sido detetada nenhuma diferença entre o comporta - mento dos estados de observação e dos estados reais. Enfim, o estado F é aquele para o qual se dirige todas as transições proibidas e, uma vez alcançado, acusa um violação. Caso haja, de início, uma disparidade proibida ($\neq$ 0,+2,-2), a violação é detetada imediatamente.

Para $d_{max}=4$, podemos construir um diagrama análogo, um pouco mais complicado. Temos, nesse caso, quatro estados reais (SDCT=0,2,4,6). Eles são definidos pelas seguintes sequências de disparidade de palavras-código:

d = 2, d = 2 e d = -2, d = 2 definem SDCT = 4
d = 2, d =-2 e d = -2, d = 2 definem SDCT = 2
d = 2, d = 4 e d = -2, d = 4 definem SDCT = 6
d = 2, d =-4 e d = -2, d =-4 definem SDCT = 0.

Essas sequências são facilmente retiradas da análise da figura III.2.b. Elas podem, é claro, serem precedidas por diversas palavras de disparidade 0, +4, e -4.

Temos ainda que uma disparidade +4 não pode ser seguida por outra positiva, nem uma disparidade -4 por outra negativa . Finalmente, o aparecimento de uma disparidade diferente de 0, $\pm$2 e $\pm$ 4 leva-nos, imediatamente, a detetar violação.

Podemos, então, construir o diagrama:

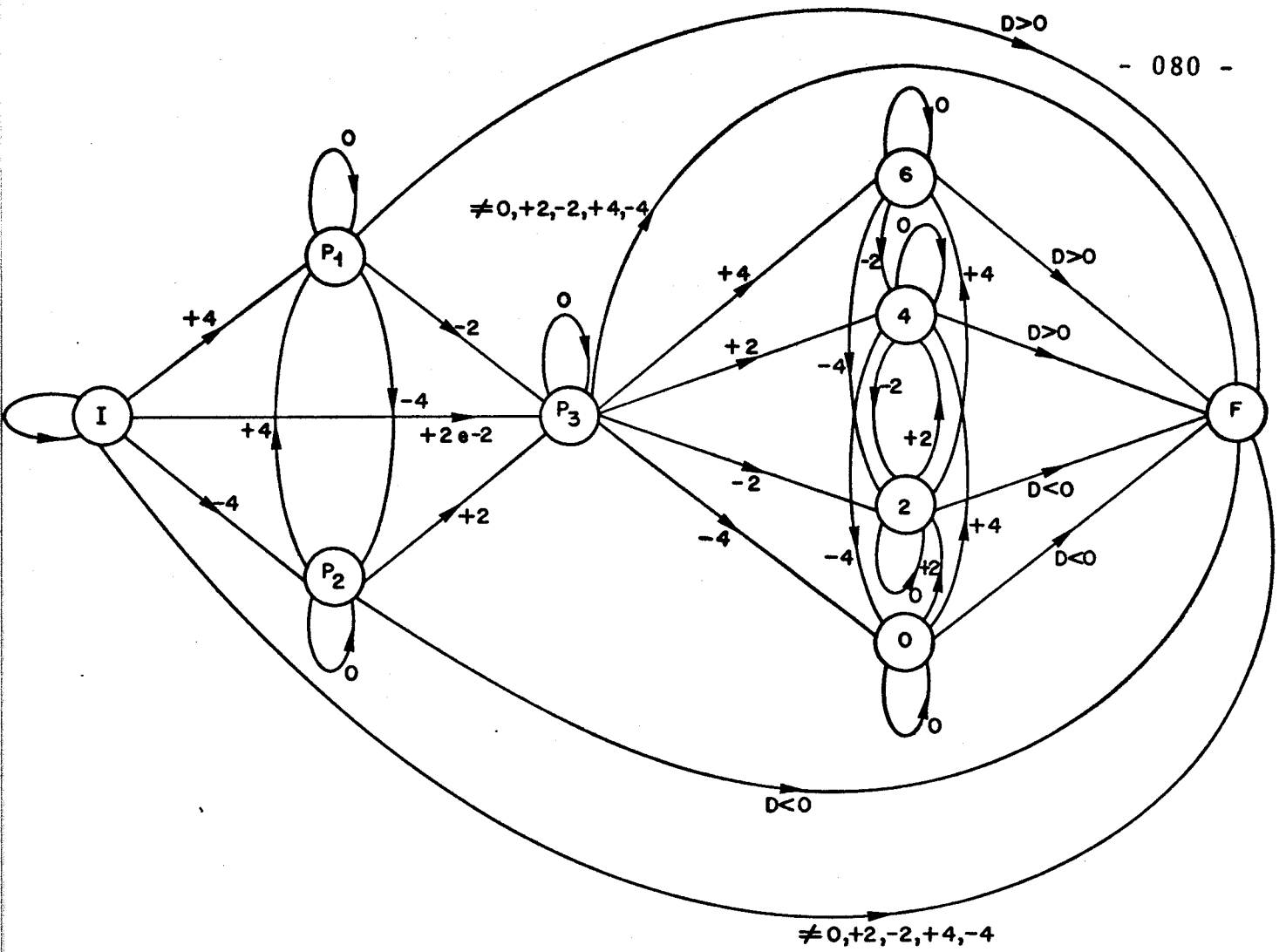


Fig. III.4 - Diagrama de deteção de transições proibidas para $d_{max}=4$.

Nossa idéia imediata foi associar, a cada ramo dos diagramas III.3 e III.4, a probabilidade de aparecer a disparidade indicada nesses ramos, para cada valor da defasagem s. Assim, encontraríamos, o número médio de passos para chegar a F, onde cada passo corresponde ao tempo de uma palavra-código, encarando o diagrama como um processo de Markov {20} . No entanto, isso não pode ser feito, pois os estados desses diagramas não são Markovianos porque a função densidade de probabilidade $p_{ij}(t)$ de transição de um estado i a outro estado j depende da história do processo anterior ao estado i {21} .

Embora estes diagramas não permitam uma análise Markoviana para o cálculo do tempo médio de se dar uma violação, eles são os mais adequados para serem implementados na prática, em termos de circuito, para contar o número de violações ocorridas e permitir ao sistema, através de uma determinada estratégia, decidir se o sistema está ou não em sincronismo. Estes diagramas possuem

a grande vantagem de permanecerem constantes (com a mesma topologia), qualquer que seja a defasagem s do sistema.

Restaria ainda a possibilidade de utilizarmos os diagramas de deteção de transições proibidas e, através de um programa de simulação, obtermos a curva da probabilidade $P_v(s)$ de violação em função do tempo (dado, preferencialmente, em palavras-código). Decidimos, porém, abandonar essa opção e partir para uma análise Markoviana, desenvolvida sobre os estados de observação do sistema defasado. Esta análise, descrita a seguir, gerou um programa que nos permitiu obter aquele mesmo histograma de uma maneira mais elegante do que através de simulação.

III.3.b - Fluxograma de Estados de Observação

Vamos supor um sistema defasado de 1 bit, cujo código possua $d_{max}=2$. Sendo assim, teremos na recepção quatro estados de observação, gerados pelo acréscimo que esse bit (0 ou 1) provoca nas disparidades terminais 0 e 2, conforme indica a figura:

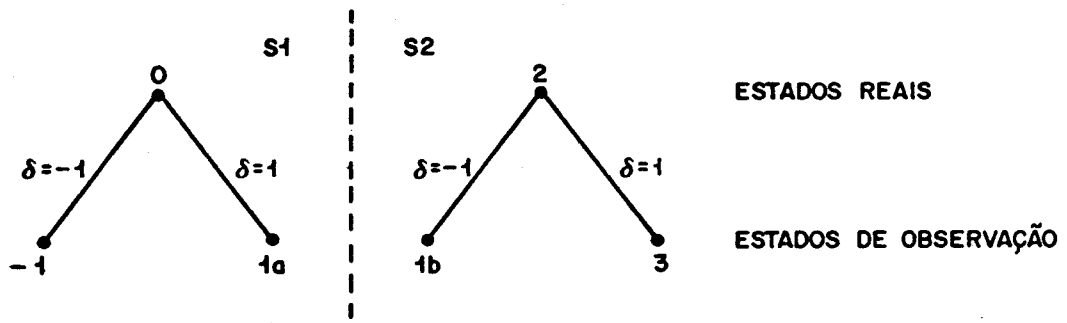


Fig. III.5 - Aparecimento dos estados de observação para $d_{max}=2$ e defasagem s=1 bit.

Na figura, temos que $\delta = -1$ quando o bit, a partir do qual começaremos a tomar a sequência de palavras defasadas, for 0 e $\delta = 1$ quando o mesmo bit for 1. Nota-se que os estados 1a e 1b, embora possuam a mesma disparidade terminal, são distintos, uma vez que procedem de estados reais diferentes. Fazendo-se assim, mantemos a probabilidade de se partir de um desses estados, rumo a qualquer outro, independente da história do processo anterior ao estado de partida, ou seja, os nossos estados de observação serão Markovianos.

A partir da criação dos estados de observação, que se dá no momento da perda de alinhamento, começará a haver transições entre esses estados. Enquanto as transições forem entre estados de disparidades adjacentes (diferença de zero ou dois), não haverá violação, pois eles estarão sendo encarados como os estados reais de SDCT=0,2. Do momento que um outro estado entrar em cena, detetar-se-á a violação, pois a transição a esse outro estado corresponderá ao surgimento de uma disparidade ou de uma sequência de disparidades proibidas.

Vamos tomar como exemplo uma codificação do tipo 5B-6B, já analisada no ítem anterior, onde proibimos as palavras: 000110, 111001, 001010, 110101, 010010, 101101 e as demais palavras-código de disparidades 4, -4, 6 e -6, (Tabela III.4).

Poderemos montar o seguinte diagrama de transições entre estados de observação:

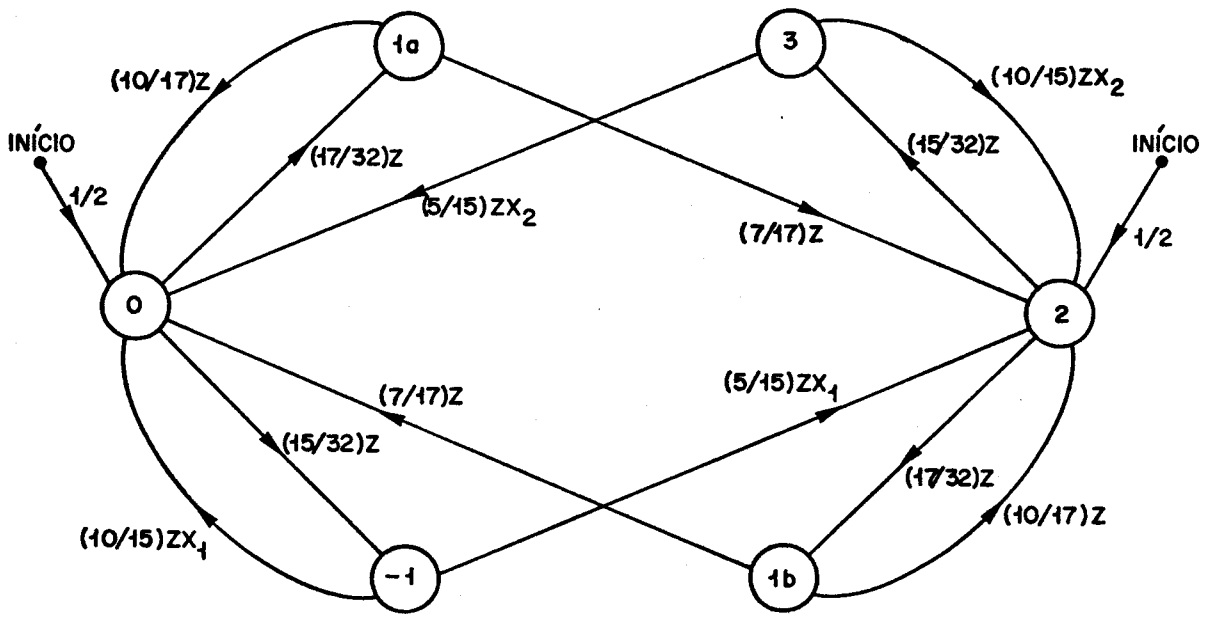


Fig. III.6 - Diagrama de estados de observação para um código 5B-6B, s=1.

O processo pode iniciar-se tanto a partir de SDCT=0 como de SDCT=3, com probabilidade 1/2 para cada. A transição entre estados de observação se faz via estados reais para facilitar a compreensão e a análise do processo. Na codificação tomada como exemplo temos que, no alfabeto progressivo existem 15 palavras iniciadas por 0 e 17 iniciadas por 1. De cada uma daquelas 15, 10 possuem disparidade zero e 5 possuem disparidade dois (daí o

PALAVRAS PERMITIDAS				
ENTRADA	ALFABETO PROGRESSIVO (S1)	d	ALFABETO REGRESSIVO (S2)	d
0	000111	0	000111	0
1	001011	0	001011	0
2	001101	0	001101	0
3	001110	0	001110	0
4	010011	0	010011	0
5	010101	0	010101	0
6	010110	0	010110	0
7	011001	0	011001	0
8	011010	0	011010	0
9	011100	0	011100	0
10	100011	0	100011	0
11	100101	0	100101	0
12	100110	0	100110	0
13	101001	0	101001	0
14	101010	0	101010	0
15	101100	0	101100	0
16	110001	0	110001	0
17	110010	0	110010	0
18	110100	0	110100	0
19	111000	0	111000	0
20	001111	+2	110000	-2
21	010111	+2	101000	-2
22	011011	+2	100100	-2
23	011101	+2	100010	-2
24	011110	+2	100001	-2
25	100111	+2	011000	-2
26	101011	+2	010100	-2
27	110110	+2	001001	-2
28	110011	+2	001100	-2
29	101110	+2	010001	-2
30	111010	+2	000101	-2
31	111100	+2	000011	-2

Tabela III.4

retorno, com possibilidade 10/15 e 5/15, aos estados reais 0 e 2, respectivamente). Verificando as demais possibilidade de transição, através da tabela que mostramos a seguir, chega-se ao diagrama da Fig. III.6.

No diagrama, temos ainda que cada dois passos (Z^2,portanto) demora o tempo de uma palavra-código, o fato de associarmos a Z, passos de diferentes durações (1 bit do estado real ao de observação e 5 bits do de observação ao real), não afeta o resultado, pois obteremos:

$$\bar{t} = \frac{\bar{n} - 1}{2} \quad , \text{ onde:} \tag{81}$$

$\bar{n}$ = número médio de passos para se detetar violação

$\bar{t}$ = número médio de palavras-código para se detetar violação,

sendo que subtraímos 1 de $\bar{n}$ para descontar o passo inicial (bit inicial) que deu origem à defasagem.

Usamos também as variáveis x_1 e x_2 para marcar quando o sistema passa pelos estados 1 e 3, respectivamente.

Será acusada uma violação quando o estado - 1 for atingido, tendo o sistema passado pelo +3 ou quando este for atingido depois de se passar pelo estado -1 (enquanto um desses dois eventos não acontecer, o valor da SDCT não sofrerá uma variação menor que dois e a defasagem não será percebida). Portanto o estado terminal do processo será -1 ou 3.

Podemos então dizer que:

$$\bar{n} = \bar{n}(-1) \times P(-1) + \bar{n}(+3) \times P(+3) \quad , \text{ onde} \tag{82}$$

$\bar{n}(-1),(+3)$ = número médio de passos que o sistema leva para terminar no estado -1,+3, condicionando-se que o estado terminal será -1, +3.

e $P(-1),(+3)$ = probabilidade de que o estado terminal seja -1, +3.

Existirá, assim, uma função de transferência da forma:

$$P_{\ell}(x_1, x_2, z) = \sum_{i_1=0}^{\infty} \sum_{i_2=0}^{\infty} \sum_{j=0}^{\infty} a^{(\ell)}_{i_1,i_2,j}\, x_1^{i_1}\, x_2^{i_2}\, z^j \,, \tag{83}$$

onde o índice $\ell = 1$ ou 2, conforme o estado final seja -1 ou 3.

As condições de finalização do processo podem ser descritas assim:

estados finais	condições
-1	$i_i = 0,\ i_2 > 0$
3	$i_i > 0,\ i_2 = 0$

pois o processo, após passar pelo estado 3, ao chegar ao estado 1 terminará, mantendo $i_1 = 0$ (para $i_2 > 0$). Analogamente, teremos $i_i > 0$ e $i_3 = 0$, quando o processo terminar em -3.

A função de transferência P_1 será então:

$$P_1(x_1, X_2, Z) = \sum_{i_1=0}^{\infty} \sum_{i_2=0}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{i_1,i_2,j}\, x_1^{i_1}\, x_2^{i_2}\, Z^j, \tag{84}$$

onde os termos $a_{i_1,i_2,j}$ representam as probabilidades do sistema chegar no estado -1 com j passos, havendo passado i_2 vezes pelo estado -3 e i_1 vezes pelo próprio estado -1. Como queremos saber o número médio de passos e sabemos que $i_1 = 0$ e $i_2 > 0$, teremos que descobrir:

$$\bar{n}(-1)\, P(-1) = \sum_{i_2=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j}\ x\ j, \tag{85}$$

sendo que o termo $a^{(1)}_{0,i_2,j}$ nos dá a probabilidade de se chegar, pela primeira vez, ao estado -1 em j passos, passando i_2 vezes por i_3. O valor

$$\sum_{i_2=1}^{\infty} a^{(1)}_{0,i_2,j}$$

nos dá a probabilidade total de se chegar a -1 em j passos, independentemente do número (maior que 0) de passagens pelo estado +3. É claro que $a_{0,i_2,j} = 0$ para $i_2 > j$.

Para calcularmos q expressão (85), vamos fazer uso de:

$$P_1(0,x_2,Z) = \sum_{i_2=0}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j} \; x_2^{i_2} \; Z^j \quad \text{e,} \qquad (86a)$$

daí:

$$P_1(0,1,Z) = \sum_{i_2=0}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j} \; Z^j \qquad (86b)$$

e

$$P_1(0,0,Z) = \sum_{j=0}^{\infty} a^{(1)}_{0,0,j} \; Z^j \qquad (86c)$$

De (86), tiramos que:

$$P_1(0,1,Z) - P_1(0,0,Z) = \sum_{i_2=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j} \; Z^j \qquad (87a)$$

$$\frac{\partial}{\partial Z}\left[P_1(0,1,Z) - P_1(0,0,Z)\right] = \sum_{i_2=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j} \; j \; x \; Z^{j-1} \qquad (87b)$$

e finalmente:

$$\frac{\partial}{\partial Z}\left[P_1(0,1,Z) - P_1(0,0,Z)\right]_{Z=1} = \sum_{i_2=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,i_2,j} \; x \; j = \bar{n}(-1)P(-1) \qquad (88)$$

Nosso trabalho será, portanto, calcular $P_1(0,1,Z)$ e $P_1(0,0,Z)$ do início ao estado -1, no fluxograma da figura III.6, através da regra de Mason. Vamos, com esse fim, encontrar $P_1(0, X_2, Z)$ e depois substituir $X_2 = 1$ e $X_2 = 0$ na expressão obtida. Fazendo, no fluxograma, $X_1 = 0$, teremos:

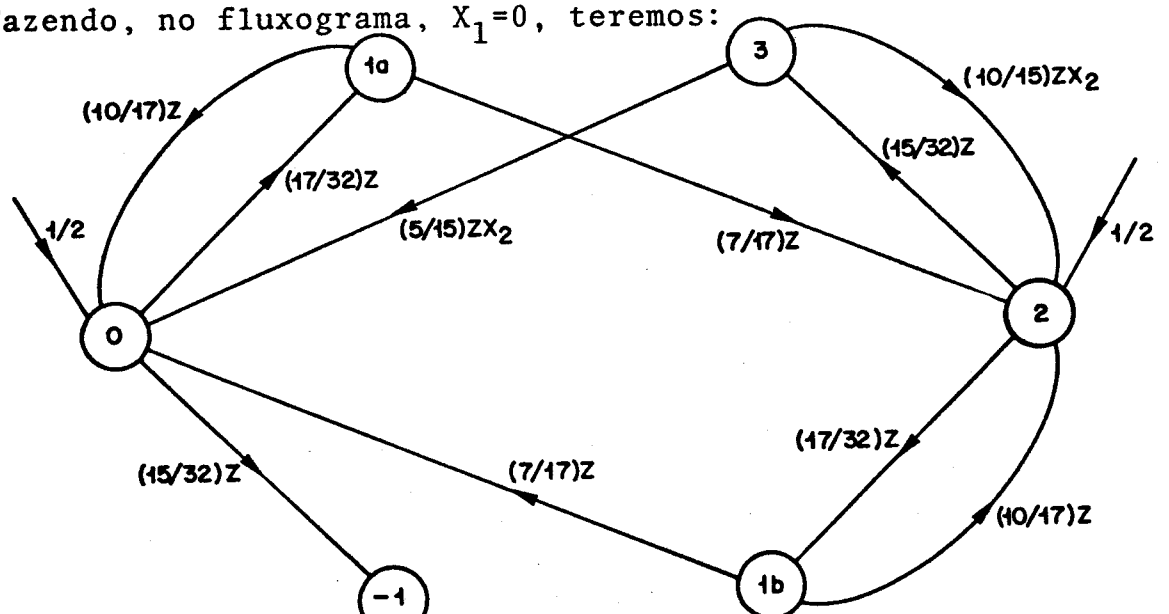


Figura III.7 - Fluxograma para $x_j = 0$.

Antes de aplicar as regras de Mason, podemos simplificar o fluxograma usando técnicas de redução de grafos {22}, encontrando:

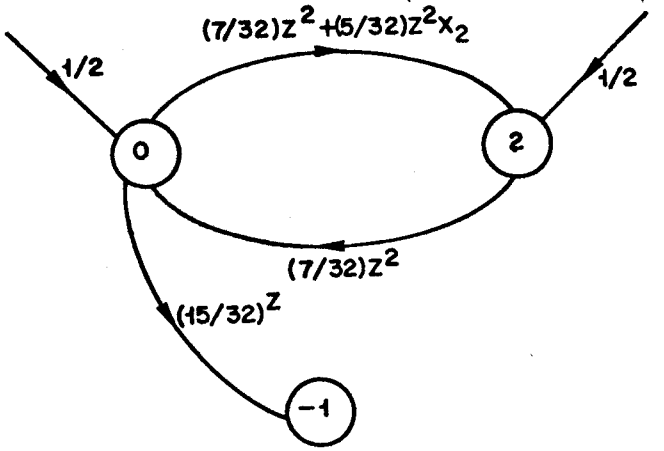


Fig.III.8 - Fluxograma Reduzido

Resolvendo, por Mason, temos: $P_1(0,X_2,Z) = \frac{1}{\Delta} \sum_n T_n \Delta_n$,

onde $\Delta = 1 - \frac{20}{32} Z^2 - \frac{10}{32} Z^2 X_2 + \frac{51}{(32)^2} Z^4 + \frac{65}{(32)^2} Z^4 X_2$ (89)

e mais:

$$\Delta_1 = 1 - \frac{10}{32} Z^2 X_2 - \frac{10}{32} Z^2 \quad (90a)$$

$$T_1 = \frac{15}{32} Z \quad (90b)$$

$$\Delta_2 = 1 \quad (91a)$$

$$T_2 = \frac{105}{(32)^2} Z^3 + \frac{75}{(32)^2} Z^3 X_2 \quad (91b)$$

Através das expressões (89), (90) e (91) chegamos a:

$$P_1(0,X_2,Z) = \frac{1}{2} \left\{ \frac{\frac{15}{32} Z - \frac{150}{(32)^2} Z^3 X_2 - \frac{150}{(32)^2} Z^3 + \frac{105}{(32)^2} Z^3 + \frac{75}{(32)^2} Z^3 X_2}{1 - \frac{20}{32} Z^2 - \frac{10}{32} Z^2 X_2 + \frac{51}{(32)^2} Z^4 + \frac{65}{(32)^2} Z^4 X_2} \right\} \quad (92)$$

de onde tiramos que:

$$P_1(0,1,Z) = \frac{1}{2} \left\{ \frac{\frac{15}{32} Z - \frac{120}{(32)^2} Z^3}{1 - \frac{30}{32} Z^2 + \frac{116}{(32)^2} Z^4} \right\} = \frac{1}{2} \frac{N_1}{D_1} \quad (93a)$$

e

$$P_1(0,0,Z) = \frac{1}{2}\left\{\frac{\frac{15}{32}Z - \frac{45}{(32)^2}Z^3}{1 - \frac{20}{32}Z^2 + \frac{51}{(32)^2}Z^4}\right\} = \frac{1}{2}\cdot\frac{N_2}{D_2} \quad (93b)$$

Chamando $N'_{1,2} = \left.\frac{dN_{1,2}}{dZ}\right|_{Z=1}$ e $D'_{1,2} = \left.\frac{d\,D_{1,2}}{dZ}\right|_{Z=1}$, escrevemos:

$$\bar{n}(1)P(1) = \frac{\partial}{\partial Z}\left[P(0,1,Z) - P(0,0,Z)\right]_{Z=1} = \frac{1}{2}\left[\frac{N'_1 D_1 - N_1 D'_1}{D_1^2} - \frac{N'_2 D_2 - N_2 D'_2}{D_2^2}\right]_{Z=1} \quad (94)$$

Calculando esses valores, encontramos:

$N_1 = 0,3516$; $N'_1 = 0,1172$; $D_1 = 0,1758$; $D'_1 = -1,4219$

$N_2 = 0,4248$; $N'_2 = 0,3369$; $D_2 = 0,4248$ $D'_2 = -1,0508$

Então, teremos $\bar{n}(-1)\,P(-1) = \frac{1}{2}\,(16,8415 - 3,2667) = 6,7875$

Pela simetria do grafo, podemos afirmar que:

$\bar{n}(-1)P(-1) = n(-3)\,P(-3)$ e daí, chegar que:

$$n = 2\bar{n}(-1)\,P(-1) = 13,5749 \quad (95)$$

Portanto:

$$\bar{t} = \frac{\bar{n}-1}{2} = 6,2875 \quad , \quad (96)$$

que é o número médio de palavras-códigos, defasadas de 1 bit, necessárias para se detetar uma violação da lei do código.

Além desse valor, podemos também estar interessados no histograma que nos dê $P_v(s=1)$ em função do tempo, dado em número de palavras necessárias.

Uma vez que:

$\bar{n} = 2\,\frac{\partial}{\partial Z}\left[P_1(0,1,Z) - P_1(0,0,Z)\right]_{Z=1}$, podemos definir

uma função:

$\bar{n}_1(Z) = 2\,[P_1(0,1,Z) - P_1(0,0,Z)]$, cujos coeficientes a_i de Z^i nos dão a probabilidade de chegarmos ao fim do processo em i passos, portanto, em $\frac{i-1}{2}$ palavras-código. Temos que:

$$\bar{n}_1(Z) = \frac{\frac{15}{32}Z - \frac{120}{(32)^2}Z^3}{1 - \frac{30}{32}Z^2 + \frac{116}{(32)^2}Z^4} - \frac{\frac{15}{32}Z - \frac{45}{(32)^2}Z^3}{1 - \frac{20}{32}Z^2 + \frac{51}{(32)^2}Z^4}$$

$$\bar{n}_1(Z) = \frac{0{,}0732\,Z^3 + 0{,}0023\,Z^5 - 0{,}0009\,Z^7}{1 - 1{,}5625\,Z^2 + 0{,}7490\,Z^4 - 0{,}1175\,Z^6 + 0{,}0056\,Z^8} \qquad (97)$$

Podemos, por confirmação, verificar que $n_1(Z=1)=1$, pois o estado -1 é final e será necessariamente alcançado.

Igualando a expressão (97) ao polinômio

$$\sum_{i=0}^{\infty} a_i\,Z^i \quad ,$$

podemos retirar os termos a_i, obtendo a fórmula de recorrência:

$$a_i = b_i + 1{,}5625\,a_{i-2} - 0{,}7490\,a_{i-4} + 0{,}1175\,a_{i-6} - 0{,}0056\,a_{i-8}$$

(i impar) (98)

onde

$$a_1 = a_{-1} = a_{-3} = a_{-5} = 0$$

e

b_i corresponde aos termos do numerador de (97).

Obtendo alguns valores, temos:

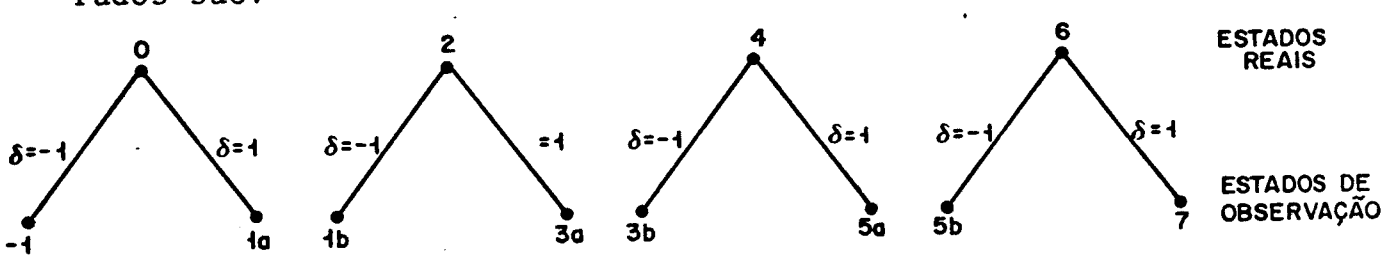
$a_1 = 0 \quad = 0$
$a_3 = 0{,}0732 \quad = 7{,}32\%$
$a_5 = 0{,}1168 \quad = 11{,}68\%$
$a_7 = 0{,}1268 \quad = 12{,}68\%$
$a_9 = 0{,}1192 \quad = 11{,}92\%$
$a_{11} = 0{,}1046 \quad = 10{,}46\%$
$a_{13} = 0{,}0886 \quad = 8{,}86\%$ ....

e $a_{par}=0$, de modo que o número de palavras códigos necessárias é, obviamente, inteiro.

Aplicamos este procedimento para defasagens s= 2a5, ob - tendo os resultados mostrados no apêndice III.B.

Pensamos, após a obtenção desses resultados. em pros - seguir com este modelamento, extendendo-o para o caso de d_{max}= 4, para futuramente dar-lhe um tratamento computacional. Entre tanto, descobrimos a impossibilidade de aplicar este procedi - mento a códigos que apresentam mais de dois estados de SDCT,pe las razões que passaremos a expor.

Vamos supor um código 7B-8B, que apresenta $d_{max}=4$, com o sistema desalinhamento por 1 bit. Os estados de observação ge rados são:



Imaginemos então que tenha sido emitida a seguinte se - quência de palavras-código: 10101010-01110110-11100111-00101010, que é perfeitamente possível, como podemos observar na figura III.2.b, se partirmos de SDCT=0.

Vamos construir então um fluxograma, análogo ao da figu- ra III.5, sendo que, para uma melhor visualização, vamos dese nhar apenas as transições provocadas por aquela sequência e não todas transições possíveis:

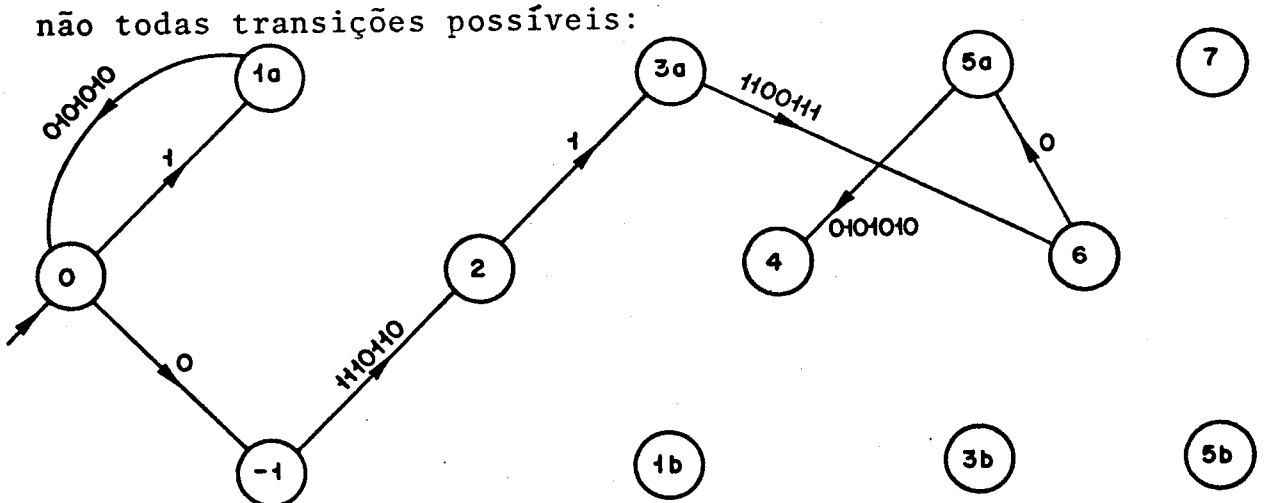
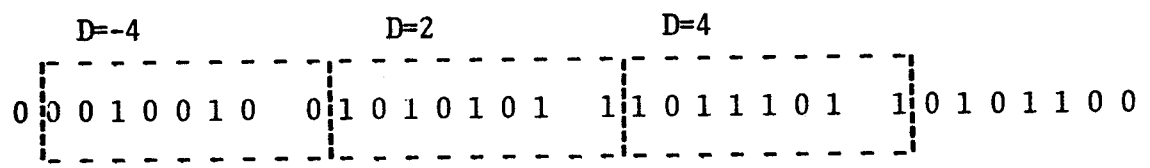


Figura III.9

Os estados visitados foram, pela ordem: 1a, -1, 3a, 5a ; vemos que o sistema deteterá uma violação pois de -1 a 3a houve um salto de 4 na disparidade terminal e, de 3a a 5a, o salto foi de 2, sendo que a lei do código não permite uma sequência de duas palavras de disparidade +4 e +2 seguidas.

Mostrando abaixo as palavras defasadas percebemos essa violação:

Suponhamos agora uma outra sequência : 00010010-01010101-11011101-10101100, que também é possível, partindo de SDCT=4 . Nesse caso, para defasagem s=1, não se perceberá violação pois temos:



e essa sequência de disparidades (-4.2.4) é permitida pela lei.

No diagrama, representaríamos assim:

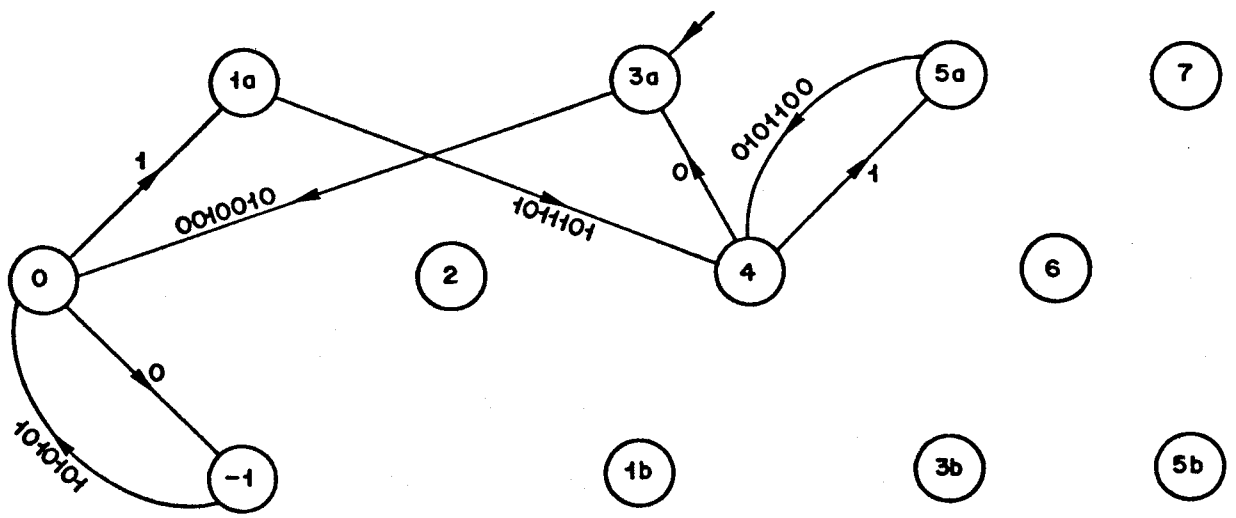


Figura III.10

Podemos verificar que os estados de observação visitados são; 3a, -1, 1a, 5; ou seja, os mesmos do exemplo anterior, porém numa ordem diferente. Concluímos que a deteção de violação não se faz, como no caso da $d_{max}=2$, apenas pelo exame de quais estados foram atingidos, mas depende também da ordem de passagem por esses estados. A análise Markoviana deste tipo de grafo, onde a ordem da transição entre os estados é decisiva, não nos pareceu possível. Restava-nos apenas a simulação dos diagramas de deteção de transição proibida para encontrar $P_V(s)$ no caso de $d_{max}=4$, que é onde se enquadra o 7B-8B.

Uma outra solução foi porém encontrada, permitindo o cálculo por computador, de $P_V(s)$ para qualquer d_{max}. Essa solução está descrita a seguir.

III.3.c - Fluxograma dos macro-estados

Como já vimos, os estados dos diagramas das figuras III.3 e III.4 não são Markovianos. Percebemos, porém, que quando o sistema está num desses estados, ele terá, obviamente, um certo valor de SDCT. Supondo $d_{max}=2$ e o sistema defasado por um bit, esses valores poderão ser -1, 1 ou 3, e a SDCT estará variando de acordo com o fluxograma da figura III.6, ou seja, dentro de um dos quatro estados de observação gerados pela defasagem.

É possível, então, re-desenhar o diagrama da figura III.3, encaixando cada um de seus estados como um conjunto dos estados Markovianos de observação (macro-estado). Chega-se, desta forma ao diagrama da figura III.11, onde temos que cada macro-estado corresponde a uma situação diferente:

- I : Após a perda de sincronismo, o sistema espera uma palavra cuja disparidade o permita se auto-localizar.
- 0,+2 : Correspondem aos estados reais de SDCT, é a situação que o sistema permanece enquanto não percebe uma violação.
- F : Estado para o qual se dirige após detetar a violação da lei.

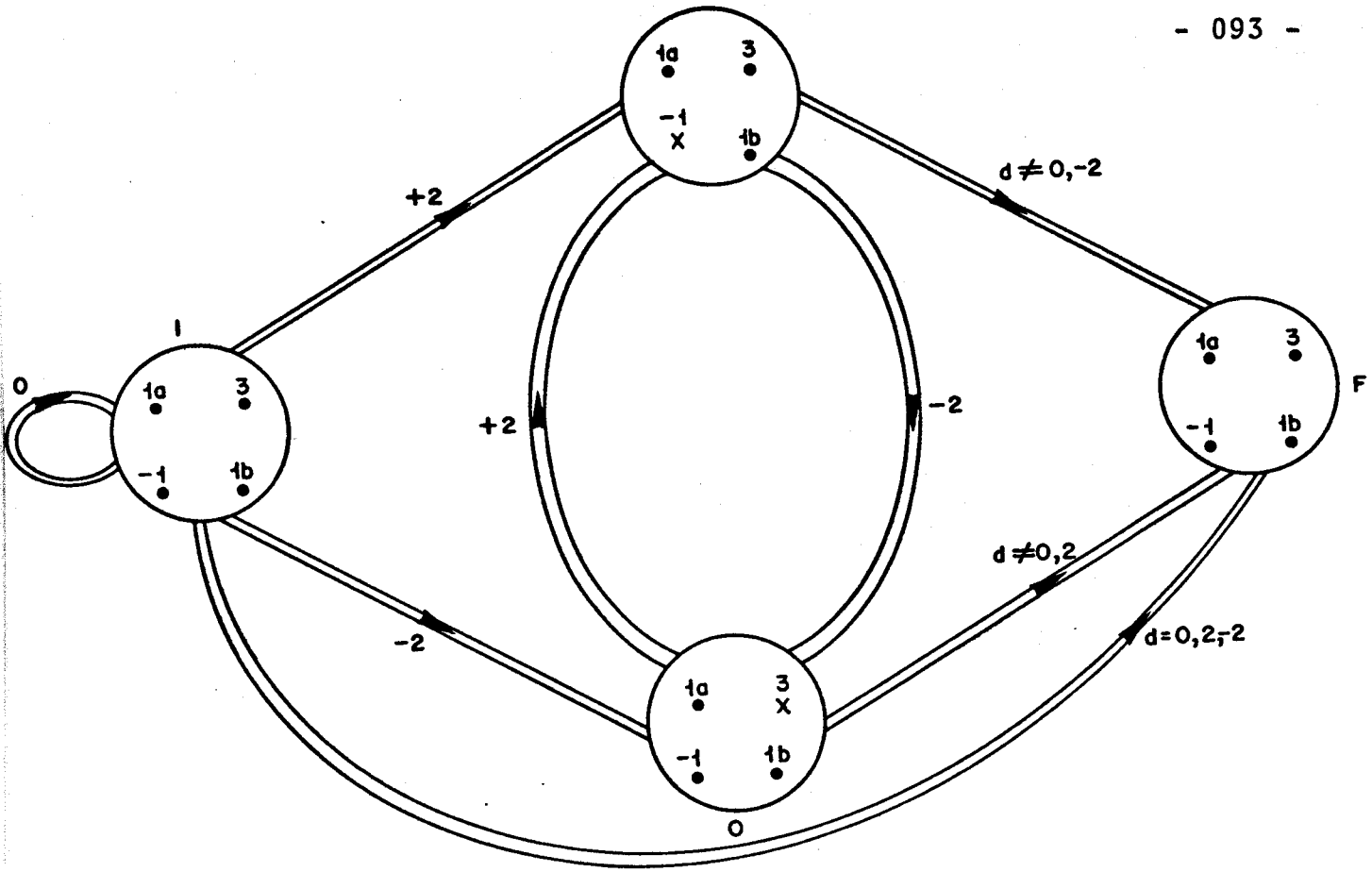


Fig. III.11 - Diagrama de macro- estados para $d_{max} = 2$ e s = 1 bit.

Na realidade, em qualquer uma dessas quatro situações, o sistema defasado de um bit terá SDCT igual a -1, 1 ou 3, e os estados markovianos que descrevem esse comportamento são os de observação (-1,1a,1b e 3). Sendo assim, cada situação corres - ponde um macro-estado não-markoviano que contêm os estados de observação.

A figura III.11 representa então um processo de Markov , com 16 estados diferentes, onde as setas duplas significam um conjunto de transições com probabilidades associadas. Essas transições serão as seguintes:

a) I ⟹ +2 e 0 ⟹ +2

- -1 → 1a
- -1 → 1b
- 1a → 3
- 1b → 3

b) I ⟹ 0 e +2 ⟹ 0

- 3 → 1a
- 3 → 1b
- 1b → -1
- 1a → -1

c) $I \Longrightarrow F$ { -1 → 3 ; 3 → -1

d) $+2 \Longrightarrow F$ { -1 → 3 ; -1 → 1a ; -1 → 1b ; 1a → 3 ; 1b → 3 ; 3 → -1

e) $0 \Longrightarrow F$ { 3 → -1 ; 1a → -1 ; 1b → -1 ; 3 → 1a ; 3 → 1b ; 1 → 3

f) $I \Longrightarrow I$, $0 \Longrightarrow 0$ e $+2 \Longrightarrow +2$ { -1 → -1 ; 1a → 1a ; 1b → 1b ; 3 → 3 ; 1a → 1b ; 1b → 1a

Nota-se que os estados assinalados por x no desenho (-1 na situação +2 e 3 na situação 0) nunca serão atingidos, pois os macro-estados +2 e 0 só são alcançados através de uma disparidade +2 e -2, respectivamente, e essas disparidades não poderão gerar um valor de SDCT igual a -1 e 3. Temos ainda que o macro-estado F pode ser reduzido a um único estado e a probabilidade $p_{i,F}$ de ser ir de um estado i qualquer até o estado F será:

$$p_{i,F} = p_{i,-1} + p_{i,1a} + p_{i,1b} + p_{i,3} \qquad (99)$$

onde -1, 1a, 1b, 3 são os respectivos estados de observação contidos em F.

As probabilidades de transição entre os estados de observação podem ser obtidas da tabela. Com esses valores, contrui -

mos a matriz:

$$R = \begin{bmatrix} p_{-1,-1} & p_{-1,1a} & p_{-1,1b} & p_{-1,3} \\ p_{1a,-1} & p_{1a,1a} & p_{1a,1b} & p_{1a,3} \\ p_{1b,-1} & p_{1b,1a} & p_{1b,1b} & p_{1b,3} \\ p_{3,-1} & p_{3,1a} & p_{3,1b} & p_{3,3} \end{bmatrix} \qquad (100)$$

A matriz R nos permite obter a matriz de transição para o fluxograma da figura III.11. Quando existir a transição de um estado i ao j, o elemento $p_{i,j}$ será um dos valores de R, conforme os estados de observação aos quais i e j correspondam. Quando a transição não existir, teremos $p_{i,j} = 0$. A matriz de transição, que chamaremos T, terá dimensão 13 x 13 e sua última linha será nula, uma vez que o último estado (F) é final e dele não parte nenhuma transição.

O processo se inicia num dos quatro estados pertencentes a I, as condições iniciais serão dadas pelas probabilidades de se gerar um desses quatro estados, a partir da defasagem de 1 bit. Teremos, então o seguinte vetor das condições iniciais:

$$\vec{p}_0 = \begin{bmatrix} p_{-1} & p_{1a} & p_{1b} & p_3 & 0 & 0 & \ldots\ldots 0 \end{bmatrix} \quad \text{, de dimensão 13.}$$

Os termos acima são obtidos fazendo-se:

$$p_{-1} = p_0 \times p'(i=0) = 1/2\ p'(i=0) \qquad (101a)$$

$$p_{1a} = p_0 \times p'(i=1) = 1/2\ p'(i=1) \qquad (101b)$$

$$p_{1b} = p_2 \times p''(i=0) = 1/2\ p''(i=0) \qquad (101c)$$

$$p_2 = p_2 \times p''(i=1) = 1/2\ p''(i=1) \qquad (101d)$$

onde as probabilidades $p_0=p_2=1/2$ correspondem a probabilidade de termos, antes da perda de sincronismo, SDCT igual a 0 ou 2. Os valores p'(i) e p''(i) nos dão a probabilidade de que o bit i

de defasagem seja zero ou um, conforme estejamos no alfabeto progressivo ou regressivo, respectivamente.

Com a matriz de transição e o vetor de condições iniciais, conseguiremos finalmente o vetor:

$$\vec{p}_n = \vec{p}_0 \times T^n \qquad (102)$$

que nos dá a probabilidade de se estar em cada estado do fluxograma para n passos executados {20}. Basta-nos inclusive o último elemento deste vetor que nos dará a probabilidade de ocorrência de violação para 1,2,...n passos, onde cada passo corresponde ao tempo de uma palavra-código. Enconstramos, assim o histograma desejado.

A principal desvantagem desse método é que o diagrama pode tornar-se muito complexo. O número de macro-estados só varia com d_{max}, mas o número de estados de observação varia com a defasagem e é maior quando s=n/2. Como os códigos que nos detemos são o 5B-6B e o 7B-8B, o caso mais complexo é o do 7B-8B, para s=4, onde ocorre que o número de macro-estados é oito (Fig. III.4), cada qual podendo abrigar até vinte estados de observação. Teríamos então uma matriz de transição cuja dimensão pode valer até 161 x 161 (contando o estado final).

Nosso último obstáculo foi, portanto, desenvolver o programa BOLJM (mostrado no apêndice III.C) que, tendo como dados de entrada às palavras permitidas dos códigos, nos constrói o fluxograma e o resolve, fornecendo o histograma desejado.

III.4 - Violação devido a erros de linha

Até agora, estivemos preocupados em obter $P_V(s)$ para $s \neq 0$, tanto por palavra proibida como por violação da lei do código e, por consequência, o número de palavras-código necessárias, em média, para detetarmos a violação. Esses valores nos dão um indicativo da sincronizabilidade do código, no tocante ao seu tempo de recuperação de sincronismo. Concluímos que devemos procurar uma tabela que nos forneça valores altos de $P_V(s\neq0)$, para que as violações sejam detetadas rapidamente, permitindo ao sistema recupe

rar logo o sincronismo perdido.

É também conveniente tecer um breve comentário sobre a probabilidade $P_v(s=0)$, de ocorrer violação com o sistema sincronizado, por causa dos erros de linha. Essa probabilidade, sendo pequena, nos possibilitará um valor grande para o tempo de retenção.

Chamando de P_e a probabilidade de erros de linha e desconsiderando a possibilidade de mais de um erro numa mesma palavra-código, poderemos achar, sem muita dificuldade, o valor de $P_v(s=0)$.

Se o sincronismo estiver sendo feito por palavras proibidas, poderemos executar o seguinte procedimento:

a) Substituir, em todas as palavras-código permitidas de um dos alfabetos, cada um de seus bits pelo complementar.

b) Contar o número de vezes (k) que essa substituição faz surgir uma palavra proibida.

c) Acharemos então: $$P_v(s=0) = \frac{k}{n \times 2^{m.}} \times P_e \quad , \qquad (103)$$

onde estamos tratando de um código qualquer mB-nB.

Esta operação só precisa ser executada em um dos alfabetos, devido à simetria dos códigos.

Quando a sincronização se faz por violação da lei de código, iremos perceber que sempre que o erro de linha provocar uma mudança de disparidade na palavra-código, mudando-a, por exemplo, de d para d', haverá, mais cedo ou mais tarde, uma violação na lei. Se supormos que o erro aconteceu quando tínhamos SDCT=D , então, após a palavra errada, a recepção julgará estar num estado de SDCT=D+d', enquanto o estado real será D+d; isso vai, obrigatoriamente, provocar uma violação, quando se atingir o estado real de SDCT=0 (para d > d') ou o de valor máximo de SDCT=s (para d' > d).

Ora, um erro de linha sempre muda a disparidade da palavra, pois substitui um dígito 1 por 0 ou vice-versa, de forma que, se

gundo o raciocínio acima, teríamos sempre ; $d' = d \pm 2$, Sendo assim, concluímos que:

$$P_v(s=0) = P_e, \qquad (104)$$

sempre que estejamos sincronizando o sistema através da lei do código.

Cabe acrescentar ainda que este erro de linha fará com que chegue à recepção uma palavra-código diferente da emitida. Essa palavra, quando decodificada, pode resultar numa outra que possua mais do que um bit diferentes dos da palavra-fonte original, havendo portanto o que chamaremos de fator de multiplicação de erros de linha. Deixamos então em aberto um interessante campo de estudo que seria o da alocação das palavras-código em relação às palavras-fonte, no sentido de minimizar esse fator.

III.5 - Os Canais de Serviço

A operação e manutenção de um sistema de comunicações é bastante facilitada pela disponibilidade de canais de serviço. A utilização de códigos de bloco possibilita a obtenção desses canais de forma simples e econômica, fazendo uso da própria redundância do código.

De acordo com o que já foi visto, podemos dizer que a comutação dos alfabetos de um código (n-1)B-nB se faz da seguinte maneira:

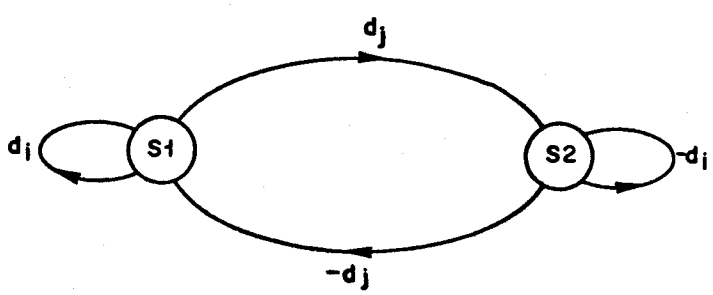


Fig. III.12

Onde d_i e d_j são conjuntos de disparidades que podem, conforme o código em questão, provocar ou não uma comutação de alfabeto.

Substituindo, por exemplo, uma palavra qualquer do alfabeto S2 por outra até então proibida, constrói-se um novo alfabeto S2 que nos possibilita a obtenção de um canal de serviço (s1) , conforme mostra o diagrama da Figura III.13:

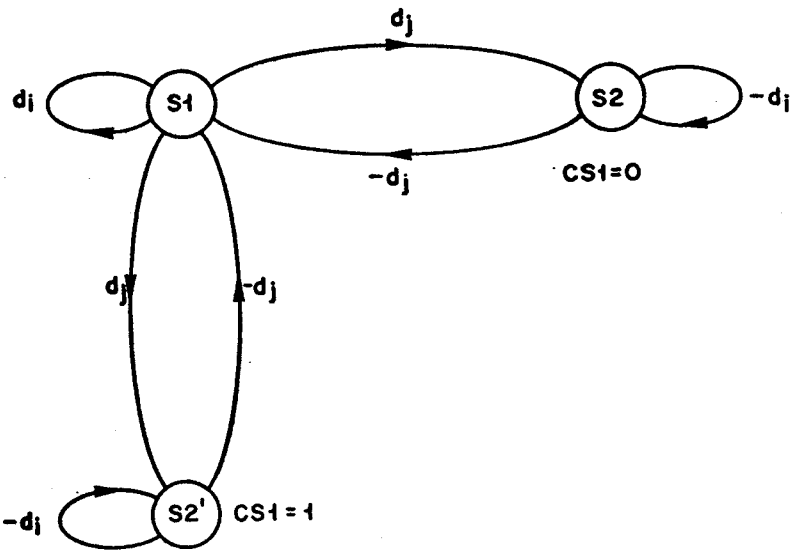


Figura III.13

Mantendo as condições de simetria, que tornam suave o espectro do sinal (Cap.II), teríamos que construir um alfabeto S1' formado pelas palavras complementares às de S2'. Ficaríamos então como dois pares de alfabeto, alternando-os de acorco com o valor do canal de serviço:

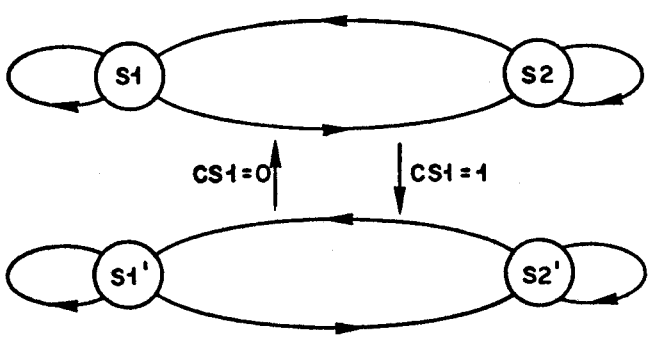


Figura III.14

Com a alternância de quatro pares de alfabeto, passaremos a ter dois canais disponíveis, sendo que cada par corresponderá aos

valores de (CS1, CS2) = (0,0); (1,0); (0,1); (1,1). Isto é obtido com o acréscimo de dua novas palavras e suas complementares . Teremos os alfabetos constituídos de forma que: $(p_1,p_2) \in S1$, $(p_1',p_2) \in S1'$, $(p_1,p_2') \in S1''$ e $(p_1',p_2') \in S1'''$; onde p_1 e p_2 eram palavras já permitidas antes da introdução dos canais de serviço e p_1' e p_2' eram proibidas e passaram a ser utilizadas. Iremos fazer uso também das complementares $\overline{p_1'}$ e $\overline{p_2'}$ para constituir S2', S2'' e S2'''.

A fim de evitar um aumento no desbalanço do código, não se deve introduzir uma palavra com disparidade maior do que d_{max}.

Os cálculos de $P_v(s)$ devem então levar em conta a existência desses canais de serviço.

Em se tratando de violação por palavra proibida e supondo que p(s=0) = p(s=1)=1/2 para todos os canais, podemos tratar cada par de alfabetos individualmente, aplicando sobre ela o procedimento indicado na seção III.2.

Obteremos então:

$$P_v(s) = \left(P_v(s)\Big|_{S1,S2} + P_v(s)\Big|_{S1',S2'} + \ldots + P_v(s)\Big|_{S1^{(N)},S2^{(N)}} \right) x \frac{1}{N}, \qquad (105)$$

onde N é o número de pares de alfabetos empregados.

Convém frizar porém que, ao aplicar o procedimento de cálculo a um determinado par de alfabetos, devemos considerar como permitidas tanto as palavras pertencentes a esses alfabetos como também as dos demais. É evidente então que a sincronizabilidade piora a medida que utilizamos mais canais de serviço, pois o número de palavras proibidas diminui.

Quando usamos as violações da lei de código para sincronizar o sistema, a introdução de canais de serviço,e, consequentemente, de novas palavras, altera a matriz R da equação (100). Teremos que levar em conta quais as palavras que existem em todos os alfabetos e quais fazem parte de apenas alguns, pois estas terão probabilidade de ocorrência menor do que aquelas. Ao se obter as probabilidades de transição entre os estados de observação, esses fatores são considerados e o programa implementado se encarrega

desses cálculos . Enfim, o modelo descrito em III.3, que se resume no fluxograma da figura III.11, continua o mesmo, variando apenas os valores das probabilidades associadas a cada transição entre os estados deste fluxograma. Ao contrário do caso de sincronismo por palavras proibidas, o comportamento da sincronizabilidade com a introdução de novas palavras não é, desta feita, imediatamente previsível.

O cálculo de $P_v(s=0)$ continuará obedecendo às equações (103) e (104), para as duas formas de sincronização, sendo que, na equação (103), teremos um valor menor para k e, consequentemente, para $P_v(s=0)$, devido à diminuição do número de palavras proibidas.

Tendo visto todos estes aspectos, resta-nos aplicar os modelos através dos programas, e obter os resultados sobre a sincronizabilidade dos códigos que nos interessar. Estes resultados estão mostrados nos próximo capítulo. Com os compromissos estabelecidos sobre a complexidade, balanceamento, eficiência e sincronizabilidade, poderemos chegar a codificação mais adequada à transmissão óptica em 140 Mb/s.

APÊNDICE III-A

O PROGRAMA CODJM.FOT

Este programa calcula $P_V(s=1)$, para $1 \leq i \leq n-1$, de um código (n-1)B-nB, pelo critério de sincronização por ocorrência de palavras proibidas, segundo o modelo descrito na seção II.2.b deste trabalho.

Para operar o programa, deve-se criar um arquivo de entrada e inserir os seguintes dados:

a) Na primeira linha, o valor de n do código e o número de canais de serviço (pode ser zero) a serem utilizados. Esses dois valores devem ser digitados sucessivamente e precedidos sempre de um espaço em branco.

b) Em seguida deve-se inserir, uma em cada linha, as palavras permitidas do alfabeto progressivo S1. Logo abaixo delas, insere-se as palavras que serão utilizadas somente nos demais alfabetos progressivos (S1',S1", etc.), caso haja canais de serviço. Todos os dígitos de cada palavra devem também ser precedida por um espaço em branco.

Ao se executar o programa ele irá pedir, via terminal, o número dos arquivos de entrada e de saída que desejamos utilizar.

No arquivo de saída é fornecida a caracterização do código (valor de n e d_{max}), as palavras-código empregadas nos alfabetos progressivos e as probabilidades de ocorrência de palavras proibidas para todas as defasagens possíveis do sistema.

O programa pressupõe as condições de simetria dadas no capítulo II, ou seja, encara o alfabeto regressivo como sendo o conjunto das palavras complementares às do progressivo. Caso queiramos utilizá-lo para um código que não respeite essas condições, devemos modificá-lo de maneira que tenhamos os dois alfabetos como dados de entrada. Também a 2ª parte do programa, onde se aplica a equação (65) sobre as palavras permitidas, deve ser alterada em função dessa mudança, pois será ne

APÊNDICE III-A

O PROGRAMA CODJM.FOT

Este programa calcula $P_v(s=1)$, para $1 \leq i \leq n-1$, de um código (n-1)B-nB, pelo critério de sincronização por ocorrência de palavras proibidas, segundo o modelo descrito na seção II.2.b deste trabalho.

Para operar o programa, deve-se criar um arquivo de entrada e inserir os seguintes dados:

a) Na primeira linha, o valor de n do código e o número de canais de serviço (pode ser zero) a serem utilizados. Esses dois valores devem ser digitados sucessivamente e precedidos sempre de um espaço em branco.

b) Em seguida deve-se inserir, uma em cada linha, as palavras permitidas do alfabeto progressivo S1. Logo abaixo delas, insere-se as palavras que serão utilizadas somente nos demais alfabetos progressivos (S1',S1", etc.), caso haja canais de serviço. Todos os dígitos de cada palavra devem também ser precedida por um espaço em branco.

Ao se executar o programa ele irá pedir, via terminal, o número dos arquivos de entrada e de saída que desejamos utilizar.

No arquivo de saída é fornecida a caracterização do código (valor de n e d_{max}), as palavras-código empregadas nos alfabetos progressivos e as probabilidades de ocorrência de palavras proibidas para todas as defasagens possíveis do sistema.

O programa pressupõe as condições de simetria dadas no capítulo II, ou seja, encara o alfabeto regressivo como sendo o conjunto das palavras complementares às do progressivo. Caso queiramos utilizá-lo para um código que não respeite essas condições, devemos modificá-lo de maneira que tenhamos os dois alfabetos como dados de entrada. Também a 2ª parte do programa, onde se aplica a equação (65) sobre as palavras permitidas, deve ser alterada em função dessa mudança, pois será ne

cessária a obtenção de $P_V(s)/S1$ e $P_V(s)/S2$ (vide equação (79), na seção III.2.b).

Por fim, se desejarmos testar um código mB-nB (mantendo-se n par) para $m \neq n-1$, será preciso que m seja um dado do programa, o que implicará em outras pequenas modificações.

Uma vez obtido os resultados, o número médio de palavras necessárias para se detetar uma violação é encontrado fazendo-se:

$$\bar{n}(s=i) = \frac{1}{P_V(s=1)}$$

A seguir, mostramos a listagem comentada do programa:

```
      DIMENSION IAUF1(1[illegible],[illegible]),IAUF2(13[illegible],[illegible]),ICS1(6,5),ICS2(6,5)
      DIMENSION IS1(8[illegible],5),IS2([illegible],5),IPMT(5),IC([illegible],2)
      DIMENSION A([illegible]),B([illegible]),X(5,[illegible]),Y(5,5),Y2(5,5)
      DIMENSION [illegible](5,[illegible]),[illegible]([illegible]),[illegible](2[illegible]),[illegible](2[illegible])
      DIMENSION [illegible](12[illegible],2),[illegible](13[illegible],2),[illegible](13[illegible],2)

C     *********************
C     *                   *
C     * PROGRAMA CODIGO   *
C     *                   *
C     *********************

C     OBJETIVO :
C     --------
C               CALCULO DE PROBABILIDADES DE OCORRENCIA DE
C               PALAVRAS PROIBIDAS PARA UM CODIGO DE BLO-
C               COS, USANDO O SISTEMA PUMA DE SINCRONISMO
C
C               ESTE PROGRAMA E PARTE INTEGRANTE DO TRABA-
C               LHO DE TESE DE MESTRADO DE JOAO R. I. ROMANO
C               E [illegible] ALLEMAN (ORIENTADOR), INTITULADA :
C               'CODIGOS BALANCEADOS PARA TRANSMISSAO DIGI-
C               TAL POR FIBRA OPTICA'

C     DADOS DE ENTRADA :
C     ----- -- -------
C                       N, NUMERO DE CANAIS DE SERVICO
C                       (NCS) E AS PALAVRAS PERMITIDAS DO
C                       ALFABETO PROGRESSIVO (MATRIZES IAUF1
C                       E ICS1)

C     SUBROTINA UTILIZADA :
C     --------- ---------
C                           MINV (PARA INVERSAO DE MATRIZES)

C     LEITURA DE DADOS :

      WRITE(5,1010)
      READ(5,1050)IR
      WRITE(5,1020)
      READ(5,1030)IL
      READ(IR,1040)N,NCS
      NUM=2**(N-1)
      READ(IR,1050)((IAUF1(I,J),J=1,N),I=1,NUM)
      IF(NCS.EQ.0)  GO TO 30
      READ(IR,1060)((ICS1(I,J),J=1,N),I=1,NCS)

C     PARTE I : CARACTERIZACAO DO CODIGO
C               ------------- -- ------
C     * OBTENCAO DOS ALFABETOS REGRESSIVOS
C     * OBTENCAO DOS VETORES A(I) E B(I)
C     * [illegible] DE [illegible] ([illegible])
```

```
      DO 20 I=1,NCS
      DO 10 J=1,N
      ICS2(I,J)=L-ICS1(I,J)
10    CONTINUE
20    CONTINUE
30    DO 50 I=1,NGR
      DO 40 J=1,N
      IALF2(I,J)=L-IALF1(I,J)
40    CONTINUE
50    CONTINUE
      DO 60 I=1,6
      ICONT(I)=0
60    CONTINUE
      DO 80 I=1,NGR
      NV1=0
      NV2=0
      NO=0
      DO 70 J=1,N
      NO=NO+(2*I-IALF1(I,J)-1)
      NV1=NV1+(2**(N-J))*IALF1(I,J)
      NV2=NV2+(2**(N-J))*IALF2(I,J)
70    CONTINUE
      NO=NO/2+1
      ICONT(NO)=ICONT(NO)+1
      G=ICONT(NO)
      IA1(G,NO)=NV1
      IA2(G,NO)=NV2
80    CONTINUE
      IF(NCS.EQ.0)  GO TO 110
      G=1
      DO 100 I=1,NCS
      DO 90 J=1,N
      NV1=NV1+(2**(N-J))*ICS1(I,J)
      NV2=NV2+(2**(N-J))*ICS2(I,J)
90    CONTINUE
      IC(G,1)=NV1
      IC(G,2)=NV2
      G=G+1
100   CONTINUE
110   NO=6
120   IF(ICONT(NO).NE.0)  GO TO 130
      NO=NO-1
      GO TO 120
130   IC4=2*(NO-1)
C
C     O VETOR A(I) NOS DA A PROBABILIDADE DE OCORRENCIA
C     DE UMA PALAVRA DE DISPARIDADE 2*(I-1) PARA O ALFA-
C     BETO PROGRESSIVO
C
      DO 140 I=1,NC
      A(I)=FLOAT(ICONT(I))/(2.**(N-1))
140   CONTINUE
C
C     O VETOR P(I) NOS DA A PROBABILIDADE DE SE ESTAR
C     NUM ESTADO DE SOMA = 2*(I-1)
C
      IF(IC4.NE.2)  GO TO 150
      P(1)=0.5
      GO TO 350
150   DO 220 I=1,NC-1
```

```
      DO 210 J=1,NC-1
      IF(I-(NC-1))  180,190,210
  180 IF(I-J)  170,190,190
  170 A1(I,J)=0.
      GO TO 210
  190 Y1(I,J)=A(1)-1.
      GO TO 210
  200 L=I-J+1
      A1(I,J)=A(L)
      GO TO 210
  210 Y1(I,J)=1.
      CONTINUE
      CONTINUE
      DO 270 I=1,NC-1
      DO 260 J=1,NC-1
      IF(I-(NC-1))  230,230,240
  230 IF(I+J-NC)  240,250,250
  240 A2(I,J)=0.
      GO TO 260
  250 L=2*NC-1-J
      Y2(I,J)=A(L)
      CONTINUE
      CONTINUE
      DO 290 I=1,NC-1
      DO 280 J=1,NC-1
      X(I,J)=A1(I,J)+Y2(I,J)
      CONTINUE
      CONTINUE
      L=0
      DO 310 I=1,NC-1
      DO 300 J=1,NC-1
      L=L+1
      AY(L)=X(J,I)
      CONTINUE
      CONTINUE
      NSK=(NC-1)
      CALL MINV(AY,NDP,DET,LS,ME)
      L=0
      DO 330 I=1,NC-1
      DO 320 J=1,NC-1
      L=L+1
      XINV(J,I)=AY(L)
      CONTINUE
      CONTINUE
      DO 340 I=1,NC-1
      P(I)=XINV(I,NDP)/2.
      CONTINUE
      NR=NC-1
      WRITE(IA,10070) NR,N,IDM
      WRITE(IA,10080)
      DO 360 I=1,NUM
      WRITE(IA,10090)(INDF1(I,J),J=1,N)
      CONTINUE
      IF(NCS.EQ.0)  GO TO 360
      WRITE(IA,10100)
      DO 370 I=1,NCS
      WRITE(IA,10110)(ICS1(I,J),J=1,N)
      CONTINUE
```

```
C       PART II : CALCUL DES PROBABILITES DE VIOLATION
C                 ------- --- -------------- -- --------
C
300     DO 500 IS=1,N-1
        DO 400 I=1,(2**IS)
        DO 390 J=1,2
        JCONT(I,J)=0
390     CONTINUE
400     CONTINUE
        G=1
        DO 420 J=1,NC
        IX1=ICONT(J)
        DO 410 I=1,IX1
        XJX1=FLOAT(IA1(I,J))/(2.**(N-IS))
        IA1(G,1)=IFIX(XJX1)
        XJX2=FLOAT(IA2(I,J))/(2.**(N-IS))
        IA1(G,2)=IFIX(XJX2)
        G=G+1
410     CONTINUE
420     CONTINUE
        DO 450 K=1,(2**IS)
        DO 440 M=1,G-1
        IF(IA1(M,1).NE.(K-1))  GO TO 430
        JCONT(K,1)=JCONT(K,1)+1
430     IF(IA1(M,2).NE.(K-1))  GO TO 440
        JCONT(K,2)=JCONT(K,2)+1
440     CONTINUE
450     CONTINUE
        DO 460 I=1,(2**IS)
        PIA(I,1)=FLOAT(JCONT(I,1))/(2.**(N-1))
        PIA(I,2)=FLOAT(JCONT(I,2))/(2.**(N-1))
460     CONTINUE
        PPG=0.
        DO 570 J=1,NC
        IX2=ICONT(J)
        DO 560 I=1,IX2
        DO 550 M=0,(2**IS)-1
        IA3=IA1(I,J)
        IA4=2**(N-IS)
        MPG=MOD((IA3,IA4)*(2**IS)+M
        DO 480 K=1,NC
        IX5=ICONT(K)
        DO 470 L=1,IX5
        IF(MPG.EQ.IA1(L,K))  GO TO 550
        IF(MPG.EQ.IA2(L,K))  GO TO 550
470     CONTINUE
480     CONTINUE
        IF(NCS.EQ.0)  GO TO 510
        DO 500 G1=1,NCS
        DO 490 G2=1,2
        IF(MPG.EQ.IC(G1,G2))  GO TO 550
490     CONTINUE
500     CONTINUE
510     PIA1=PIA(M+1,1)
        PIA2=PIA(M+1,2)
        PI2=0.
        IF((NC-J+1)-(NC-1))520,530,540
520     DO 530 L=(NC-J+1),(NC-1)
        PI2=PI2+P(L)
530     CONTINUE
```

```
5[illegible]    PAV=[illegible]
         PA1=1.-PA[illegible]
         PI=PI*[illegible]
         PPG=PPG+PI
550      CONTINUE
560      CONTINUE
570      CONTINUE
         PPG=PPG/([illegible])
         WRITE(IW,1120)[illegible],PPG
5[illegible]    CONTINUE
         STOP
10010    FORMAT(/,10X,' [illegible]UIVO DE E[illegible] = ',S)
10020    FORMAT(/,10X,' [illegible]UIVO DE SAIDA = ',S)
100[illegible]    FORMAT(I2,I2)
100[illegible]    FORMAT(I2,I2)
1[illegible]    FORMAT(8I2)
1[illegible]    FORMAT(6I[illegible])
1[illegible]    FORMAT(//,' [illegible] TIPO',I2,'[illegible]',I1,'[illegible]',' [illegible]=',I2,//)
1[illegible]    FORMAT(//,' [illegible] PROGRESSIVO',/)
1[illegible]    FORMAT(2I[illegible])
1[illegible]    FORMAT(//,' [illegible] DISPONIVEIS PARA [illegible] DE SERVICO',/)
1[illegible]    FORMAT([illegible])
1[illegible]    FORMAT(//,'[illegible] DE VIDA [illegible] PARA',I2,' DESCOBERTOS =',F10.5)
         END
```

APÊNDICE III-B

FLUXOGRAMAS DE ESTADOS DE OBSERVAÇÃO PARA DEFASAGENS s >1

Na seção III.3.b, mostramos que uma defasagem de 1 bit no alinhamento do sistema provoca o aparecimento de estados de ob - servação que podem ser agrupados num processo Markoviano, confor me ilustra o diagrama da figura III.5.

Verificamos que este diagrama constitui um meio eficaz de se obter o número médio de palavras-código necessárias para se dar uma violação, assim como o histograma da probabilidade $P_v(s=1)$ em função do tempo (dado em número de palavras-código).

Neste apêndice, mostramos um resumo do procedimento e os resultados encontrados para defasagens s = 2 a 5 , do mesmo código 5B-6B já tomado como exemplo (Tab. III.4). Não mostraremos os cál culos em detalhe pois eles são análogos aos apresentados no caso de s=1 bit, sendo que se tornam mais longos e complexos para defa sagens diferentes de um. Esta complexidade maior é devido ao au - mento do número de estados de observação e, consequentemente, das condições de finalização do processo, o que nos obriga a lidar com um maior número de funções de trasnferência. É nosso objetivo expor, para cada valor de s, o fluxograma original, as condições de finalização e os resultados.

a) s = 2 bits: Cada estado real irá gerar três de observação e o diagrama será o seguinte:

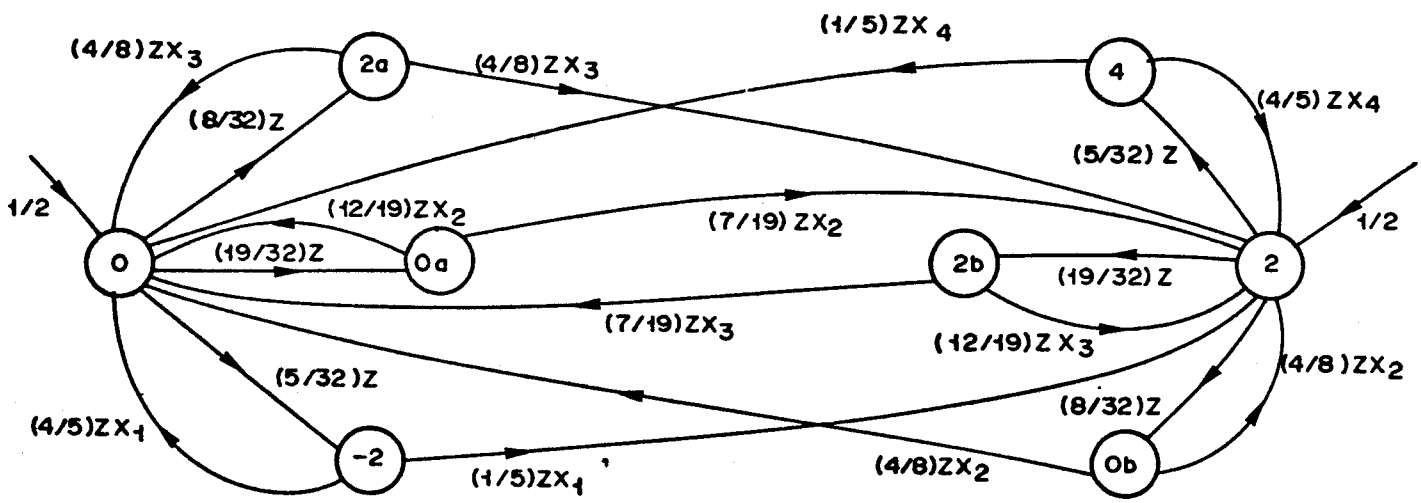


Fig. III.B.1 - Fluxograma para s=2 bits.

No diagrama, as variáveis X_1, X_2, X_3 e X_4 marcam a passagem do processo pelos estados de disparidade terminal $-2, 0, 2$ e 4, respectivamente.

Qualquer um dos estados pode ser terminal e as condições de finalização são as seguintes:

Estados Finais	Condições
-2	$i_1=0$ e $\begin{cases} i_4>0 \text{ e } i_2,i_3=0 \\ \text{ou } i_3>0 \text{ e } i_2,i_4=0 \\ \text{ou } i_3,i_4>0 \text{ e } i_2=0 \\ \text{ou } i_3,i_2>0 \text{ e } i_4=0 \end{cases}$
0a e 0b	$i_2=0$ e $\begin{cases} i_4>0 \text{ e } i_1,i_3=0 \\ \text{ou } i_4,i_3>0 \text{ e } i_1=0 \end{cases}$
2a e 2b	$i_3=0$ e $\begin{cases} i_1>0 \text{ e } i_2,i_4=0 \\ \text{ou } i_1,i_2>0 \text{ e } i_4=0 \end{cases}$
4	$i_4=0$ e $\begin{cases} i_1>0 \text{ e } i_2,i_3=0 \\ \text{ou } i_2>0 \text{ e } i_1,i_3=0 \\ \text{ou } i_1,i_2>0 \text{ e } i_3=0 \\ \text{ou } i_2,i_3>0 \text{ e } i_1=0 \end{cases}$

Temos que:

$$P_\ell = (X_1,X_2,X_3,X_4,Z) = \sum_{i_1=0}^{\infty} \sum_{i_2=0}^{\infty} \sum_{i_3=0}^{\infty} \sum_{i_4=0}^{\infty} \sum_{j=0}^{\infty} a^{(\ell)} X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4} Z^j \qquad (106)$$

onde $\ell = 1,2,3$ ou 4 de acordo com o estado de observação tomado como final. O número médio de passos necessários para o processo chegar ao fim é:

$$\bar{n} = \bar{n}(-2)\ P(-2) + \bar{n}(0)\ P(0) + \bar{n}(2)\ P(2) + \bar{n}(4)\ P(4) \tag{107a}$$

onde, da mesma forma que no caso de s=1, podemos escrever

$$\bar{n} = 2\left[\bar{n}(-2)\ P(-2) + \bar{n}(0)\ P(0)\right] \quad , \tag{107b}$$

devido à simetria do fluxograma.

Analisando as condições de finalização, tiramos que:

$$\bar{n}(-2)\ P(2) = \sum_{j=0}^{\infty}\sum_{i_4=0}^{\infty} a^{(1)}_{000i_4j}\ x\ j + \sum_{j=0}^{\infty}\sum_{i_3=1}^{\infty} a^{(1)}_{000i_30j}\ x\ j +$$

$$\sum_{j=0}^{\infty}\sum_{i_3=1}^{\infty}\sum_{i_4=1}^{\infty} a_{00i_3i_4j}\ x\ j + \sum_{j=0}^{\infty}\sum_{i_2=1}^{\infty}\sum_{i_3=1}^{\infty} a_{0i_2i_30j}\ x\ j \tag{108a}$$

e

$$\bar{n}(0)\ P(0) = \sum_{i_4=1}^{\infty}\sum_{j=0}^{\infty} a^{(2)}_{000i_4j}\ x\ j + \sum_{j=0}^{\infty}\sum_{i_3=1}^{\infty}\sum_{i_4=1}^{\infty} a^{(2)}_{00i_3i_4j}\ x\ j \tag{108b}$$

Com algumas transformações, é possível escrever as expressões (108) em termos das funções de transferência $P_1(X_1,X_2,X_3,X_4,Z)$ e $P_2(X_1,X_2,X_3,X_4,Z)$. Chega-se a:

$$\bar{n}(-2)P(-2) = \frac{\partial}{\partial Z}\Big[P_1(0,0,1,1,Z) + P_1(0,1,1,0,Z) - P_1(0,1,0,0,Z) - P_1(0,0,1,0,Z)\Big]_{Z=1} \tag{109a}$$

e

$$\bar{n}(0)P(0) = \frac{\partial}{\partial Z}\Big[P_2(0,0,1,1,Z) - P_2(0,0,1,0,Z)\Big]_{Z=1} \tag{109b}$$

Teremos então que obter cada um desses termos através do uso das regras de Mason e de redução dos fluxogramas, de modo análogo ao como foi feita a resolução da equação (88) na seção III. 3.b.

Os resultados por nós obtidos foram:

$$\bar{n}(-2)\ P(-2) = 5,6088 \quad \text{e} \quad \bar{n}(0)\ P(0) = 1,0326,$$

então:

$$\bar{n} = 2\left[5,6088 + 1,0326\right] = 13,2828 \text{ passos e}$$

e

$$\bar{t} = \frac{\bar{n}-1}{2} = 6,1414 \text{ palavras-código - necessárias, em mé -}$$

dia, para se detetar uma violação.

A função da transferência $\bar{n}(Z)$ é dada por:

$$\bar{n}_2(Z) = 2\Big[P_1(0,0,1,1,Z) + P_1(0,1,1,0,Z) - P_1(0,1,0,0,Z) - P_1(0,0,1,0,Z) + P_2(0,0,1,1,Z) - P_2(0,0,1,0,Z)\Big]_{Z=1} \qquad (110a)$$

$$n_2(Z) = \frac{0,1084Z^3 - 0,0895Z^5 + 0,00617Z^7 + 0,0009Z^9}{1-2,2250Z^2 + 1,589Z^4 - 0,4940Z^6 + 0,0572Z^8 - 0,0023Z^{10} + 0,0001\,Z^{12}}, \qquad (110b)$$

de onde podemos tirar o histograma de $P_v(s=2)$, igualando $n(Z)$ a um polinômio do tipo

$$\sum_{i=0}^{\infty} a_i Z^i.$$

b) s = 3 bits: Neste caso existem oito estados de observação e o fluxograma é, para esta codificação, o mais complexo, pois $s=n/2$.

Todos os estados podem ser finais e temos:

$$\bar{n} = \bar{n}(-3)\ P(-3) + \bar{n}(-1)\ P(-1) + \bar{n}(1)\ P(1) + \bar{n}(3)\ P(3) + \bar{n}(5)\ P(5) \qquad (111)$$

Mas como

$$\begin{cases} \bar{n}(-3)\ P(-3) = \bar{n}(5)\ P(5) \\ \bar{n}(-1)\ P(-1) = \bar{n}(5)\ P(3) \end{cases} \quad \text{, por simetria,}$$

então:

$$\bar{n} = 2\left[\bar{n}(-3)\ P(-3) + \bar{n}(-1)\ P(-1)\right] + \bar{n}(1)\ P(1) \qquad (112)$$

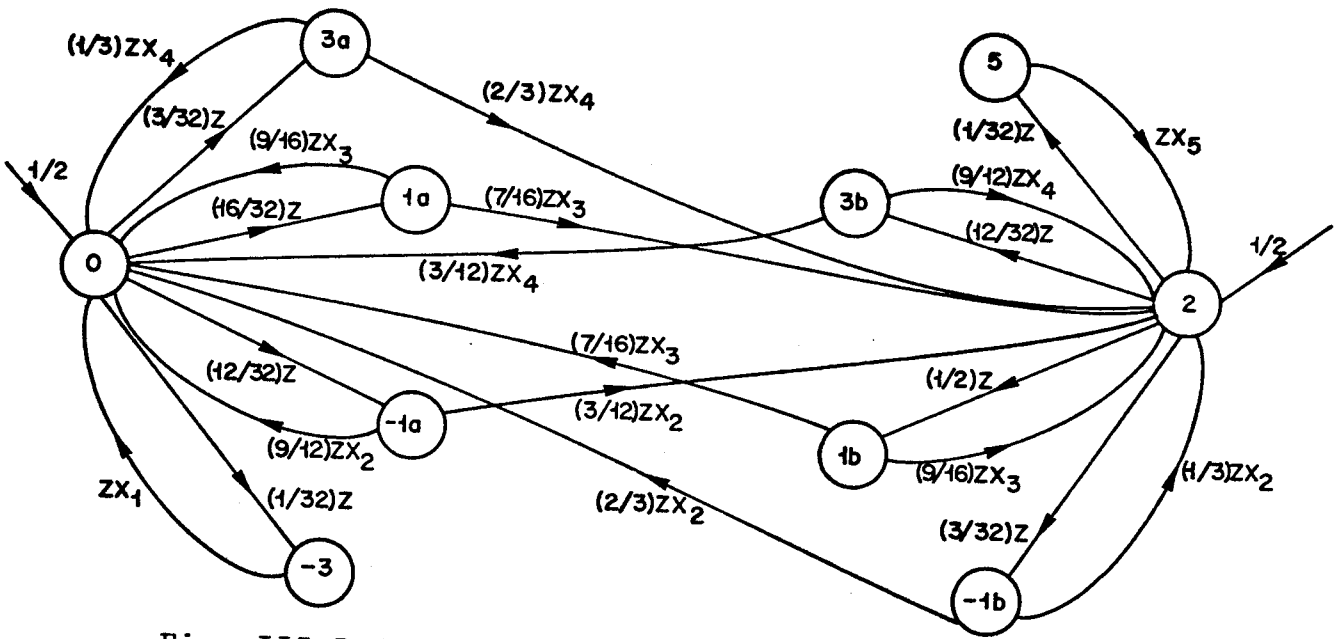


Fig. III.B.2 - Fluxograma para s = 3 bits.

Basta saber, portanto, as condições de se finalizar o processo nos estados de SDCT igual a -3, -1 e 1, para obtermos $\bar{n}$. Essas condições são as seguintes:

Estados finais	Condições
-3	$i_1=0$ e $\begin{cases} i_5, i_4 > 0 \text{ e } i_2, i_3 = 0 \\ \text{ou } i_4 > 0 \text{ e } i_2 i_3, i_5 = 0 \\ \text{ou } i_4, i_3 > 0 \text{ e } i_2, i_5 = 0 \\ \text{ou } i_3 > 0 \text{ e } i_2, i_4, i_5 = 0 \\ \text{ou } i_2, i_3 > 0 \text{ e } i_4, i_5 = 0 \end{cases}$
-1a e -1b	$i_2=0$ e $\begin{cases} i_5, i_4 > 0 \text{ e } i_1, i_3 = 0 \\ \text{ou } i_4 > 0 \text{ e } i_1, i_3, i_5 = 0 \\ \text{ou } i_3, i_4 > 0 \text{ e } i_1, i_5 = 0 \\ \text{ou } i_5 > 0 \text{ e } i_2, i_3, i_4 = 0 \end{cases}$

Estados Finais

1a e 1b

Condições

$$i_3=0 \text{ e } \begin{cases} i_5>0 \text{ e } i_1,i_2,i_4=0 \\ i_5,i_4>0 \text{ e } i_1,i_2=0 \\ i_1,i_2>0 \text{ e } i_4,i_5=0 \\ i_1>0 \text{ e } i_2,i_4,i_5=0 \end{cases}$$

Dessas condições, tiramos que:

$$\bar{n}(-3)\,P(-3) = \sum_{i_4=1}^{\infty}\sum_{i_5=1}^{\infty}\sum_{j=0}^{\infty} a^{(1)}_{000i_4i_5j}\,x\,j + \sum_{i_4=1}^{\infty}\sum_{j=0}^{\infty} a^{(1)}_{000i_40j}\,xj +$$

$$+ \sum_{i_3=1}^{\infty}\sum_{i_4=1}^{\infty}\sum_{j=0}^{\infty} a^{(1)}_{00i_3i_40j}\,x\,j + \sum_{i_3=1}^{\infty}\sum_{j=0}^{\infty} a^{(1)}_{00i_300j}\,xj +$$

$$\sum_{i_2=1}^{\infty}\sum_{i_3=1}^{\infty}\sum_{j=0}^{\infty} a^{(1)}_{0i_2i_300j}\,x\,j \qquad (113a)$$

$$\bar{n}(-1)P(-1) = \sum_{i_4=1}^{\infty}\sum_{i_5=1}^{\infty}\sum_{j=0}^{\infty} a^{(2)}_{000i_4i_5j}\,x\,j + \sum_{i_4=1}^{\infty}\sum_{j=0}^{\infty} a^{(2)}_{000i_40j}\,x\,j +$$

$$\sum_{i_5=1}^{\infty}\sum_{j=0}^{\infty} a^{(2)}_{0000i_5j}\,x\,j + \sum_{i_3=1}^{\infty}\sum_{i_4=1}^{\infty}\sum_{j=0}^{\infty} a^{(2)}_{00i_3i_40j}\,x\,j$$

(113b)

$$\bar{n}\,(1)\,P(1) = \sum_{i_4=1}^{\infty}\sum_{i_5=1}^{\infty}\sum_{j=0}^{\infty} a^{(3)}_{000i_4i_5j}\,x\,j + \sum_{i_5=1}^{\infty}\sum_{j=0}^{\infty} a^{(3)}_{0000i_5j}\,x\,j +$$

$$+ \sum_{i_1=1}^{\infty}\sum_{i_2=1}^{\infty}\sum_{j=0}^{\infty} a^{(3)}_{i_1i_2000j}\,x\,j + \sum_{i_1=1}^{\infty}\sum_{j=0}^{\infty} a^{(3)}_{i_10000j}\,x\,j$$

..

(113c)

Podemos notar que essas expressões já se tornam bem mais complicadas que as dos casos anteriores. Podemos colocá-las em termos de funções de transferência, sabendo que:

$$P\,(X_1,X_2,X_3,X_4,X_5,Z) = \sum_{i_1=0}^{\infty}\sum_{i_2=0}^{\infty}\sum_{i_3=0}^{\infty}\sum_{i_4=0}^{\infty}\sum_{i_5=0}^{\infty}\sum_{j=0}^{\infty} a_{i_1 i_2 i_3 i_4 i_5 j} \ldots$$

$$X_1^{i_1}\, X_2^{i_2}\, X_3^{i_3}\, X_4^{i_4}\, X_5^{i_5}\, Z^j \qquad (114)$$

onde $\ell = 1$ a 5

então é possível obter:

$$\bar{n}(-3)\,P(-3) = \frac{\partial}{\partial Z}\Big[P_1(0,0,0,1,1,Z) - P_1(0,0,0,0,1,Z) + P_1(0,0,1,1,0,Z) -$$

$$- P_1(0,0,0,1,0,Z) + P_1(0,1,1,0,0,Z) - P_1(0,1,0,0,0,Z) -$$

$$- P_1(0,0,1,0,0,Z) + P_1(0,0,0,0,0,Z)\Big]_{Z=1} \qquad (115a)$$

$$n(-1)P(-1) = \frac{\partial}{\partial Z}\Big[P_2(0,0,0,1,1,Z) + P_2(0,0,1,1,0,Z) - P_2(0,0,0,1,0,Z) -$$

$$- P_2(0,0,1,0,0,Z)\Big]_{Z=1} \qquad (115b)$$

$$\bar{n}(1)P(1) = \frac{\partial}{\partial Z}\Big[P_3(0,0,0,1,1,Z) - P_3(0,0,0,1,0,Z) + P_3(1,1,0,0,0,Z) -$$

$$- P_3(0,1,0,0,0,Z)\Big]_{Z=1} \qquad (115c)$$

Teríamos que obter então diversas funções de transferência da forma descrita na equação (111), onde existirá sempre três variáveis, no mínimo, iguais a zero. Cada uma dessas funções é obtida de um fluxograma diferente, que corresponde sempre a uma forma reduzida do fluxograma original (Fig, III.B.2).

Após esse trabalho, chegamos a:

..

$\bar{n}(-3)\ P(-3) = 0,7751$

$\bar{n}(-1)\ P(-1) = 4,7767$

$\bar{n}(1)\ \ P(1) = 0,1907$

Então, $n = 2\left[0,7751 + 4,7767\right] + 0,1907 = 11,2943$

e $\bar{t} = \frac{\bar{n}-1}{2} = 5,1471$ palavras-código.

A função de transferência encontrada para o processo é:

$$n_3(Z) = \frac{0,1172\ Z^3 - 0,0910\ Z^5 + 0,0194\ Z^7 + 0,0002\ Z^9 - 0,0002 Z^{11}}{1 - 2,0938\ Z^2 + 1,6689\ Z^4 - 0,6444\ Z^6 + 0,1262\ Z^8 - 0,0120\ Z^{10} + 0,0005\ Z^{12}} \qquad (116)$$

c) <u>s = 4 bits</u>: Encontra-se, por inspeção da Tabela III.4, o seguinte fluxograma:

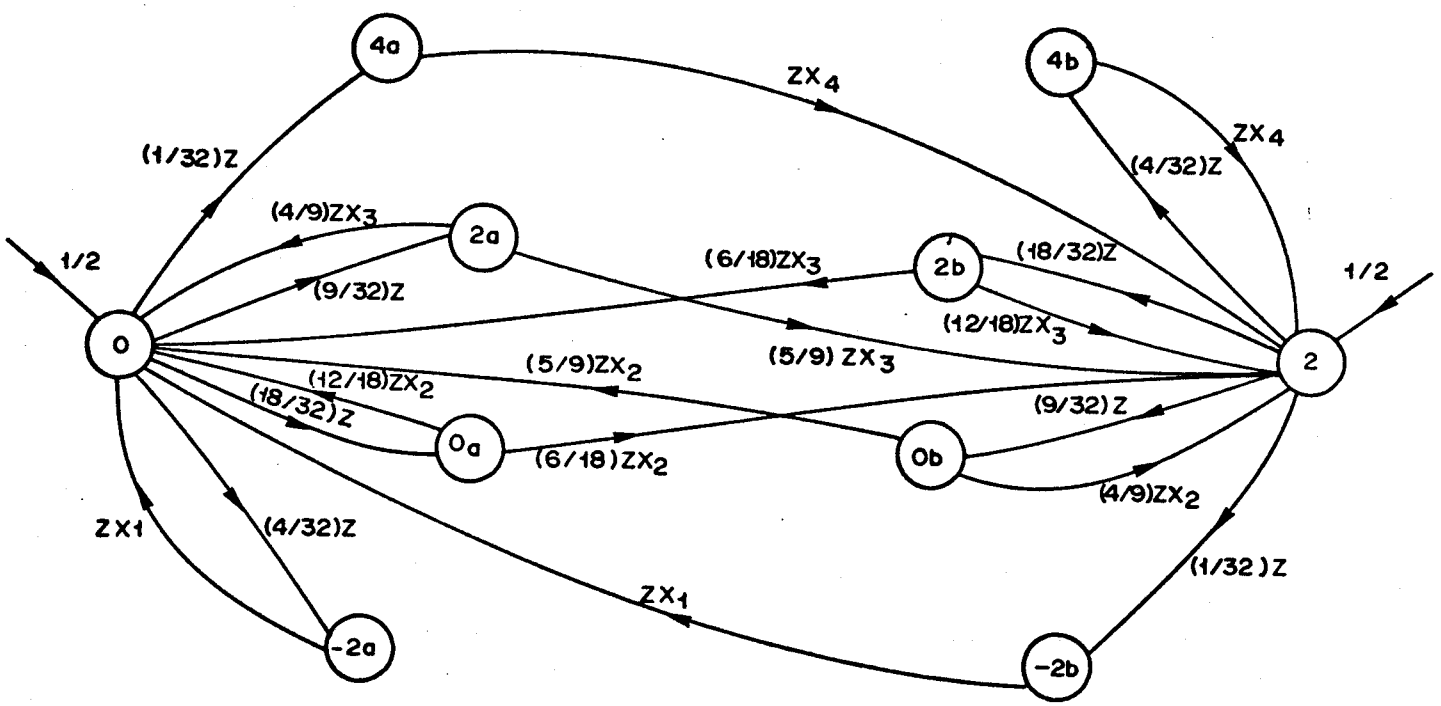


Fig. III.B.3 - Fluxograma para s = 4 bits.

Existem oito estados distintos para apenas quatro valores de disparidade terminal. Por simetria perceberemos que:

$$\bar{n}(-2)\ P(-2) = \bar{n}(4)\ P(4) \text{ e } \bar{n}(0)\ P(0) = \bar{n}(2)\ P(2) \quad ,$$

de forma que:

$$\bar{n}=\bar{n}(-2)P(-2)+\bar{n}(0)P(0)+\bar{n}(2)P(2)+\bar{n}(4)P(4) = 2\left[\bar{n}(-2)P(-2)+\bar{n}(0)P(0)\right] \tag{117}$$

As condições para o processo terminar em estados de SDCT igual a -2 ou 0 são:

Estados Finais	Condições
-2a e -2b	$i_1=0$ e ($i_4 > 0$ e $i_2, i_3 = 0$ ou $i_4, i_3 > 0$ e $i_2 = 0$ ou $i_3 > 0$ e $i_2, i_4 = 0$ ou $i_2, i_3 > 0$ e $i_4 = 0$)
0a e 0b	$i_2=0$ e ($i_4 > 0$ e $i_1, i_3 = 0$ ou $i_4, i_3 > 0$ e $i_1 = 0$)

E, então:

$$\bar{n}(-2)\ P(-2) = \sum_{i_3=1}^{\infty} \sum_{i_4=i}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{00i_3i_4j}\ x\ j + \sum_{i_4=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{000i_4j}\ xj + \sum_{i_3=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{00i_30j}\ xj + \sum_{i_2=1}^{\infty} \sum_{i_3=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0i_2i_30j}\ x\ j \tag{118a}$$

$$\bar{n}(0)\ P(0) = \sum_{i_4=1}^{\infty} \sum_{j=0}^{\infty} a^{(2)}_{000i_4j}\ x\ j + \sum_{i_3=1}^{\infty} \sum_{i_4=1}^{\infty} \sum_{j=0}^{\infty} a^{(2)}_{00i_3i_4j}\ x\ j \tag{118b}$$

Se

$$P_{\ell} = \sum_{i_1=0}^{\infty} \sum_{i_2=0}^{\infty} \sum_{i_3=0}^{\infty} \sum_{i_4=0}^{\infty} \sum_{j=0}^{\infty} a^{(\ell)}_{i_1 i_2 i_3 i_4 j} \; X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4} Z^j$$

onde $\ell = 1,2,3,4$ conforme os estados terminais forem de SDCT = -2,0,2 ou 4, poderemos chegar a :

$$\bar{n}(-2)\, P(-2) = \frac{\partial}{\partial Z}\Big[P_1(0,0,1,1,Z) - P_1(0,0,1,0,Z) + P_1(0,1,1,0,Z) - P_1(0,1,0,0,Z)\Big]_{Z=1} \qquad (119a)$$

$$n(0)\, P(0) = \frac{\partial}{\partial Z}\Big[P_2(0,0,1,1,Z) - P_2(0,0,1,0,Z)\Big]_{Z=1} \qquad (119b)$$

Resolvendo os fluxogramas correspondentes a cada uma dessas funções, teremos:

$$\bar{n}(-2)\, P(-2) = 5,6407$$

e

$$\bar{n}(0)\, P(0) = 1,0187 \quad , \text{ portanto:}$$

$$\bar{n} = 2\,[5,6407 + 1,0187] = 13,3188$$

e $\bar{t} = \frac{\bar{n}-1}{2} = 6,1594$ palavras-código.

A função de transferência, para essa defasagem, será:

$$n_4(Z) = \frac{0,1045\, Z^3 - 0,0778\, Z^5 - 0,0011\, Z^7 + 0,0018\, Z^9}{1 - 2,1250\, Z^2 + 1,6142\, Z^4 - 0,5304\, Z^6 + 0,0723\, Z^8 - 0,0037\, Z^{10} + 0,0001\, Z^{12}} \qquad (120)$$

d) s = 5 bits: Para esta defasagem, o fluxograma volta a apresentar uma topologia mais simples, uma vez que são gerados menos estados de observação e aparecem apenas três valores de disparidade terminal.

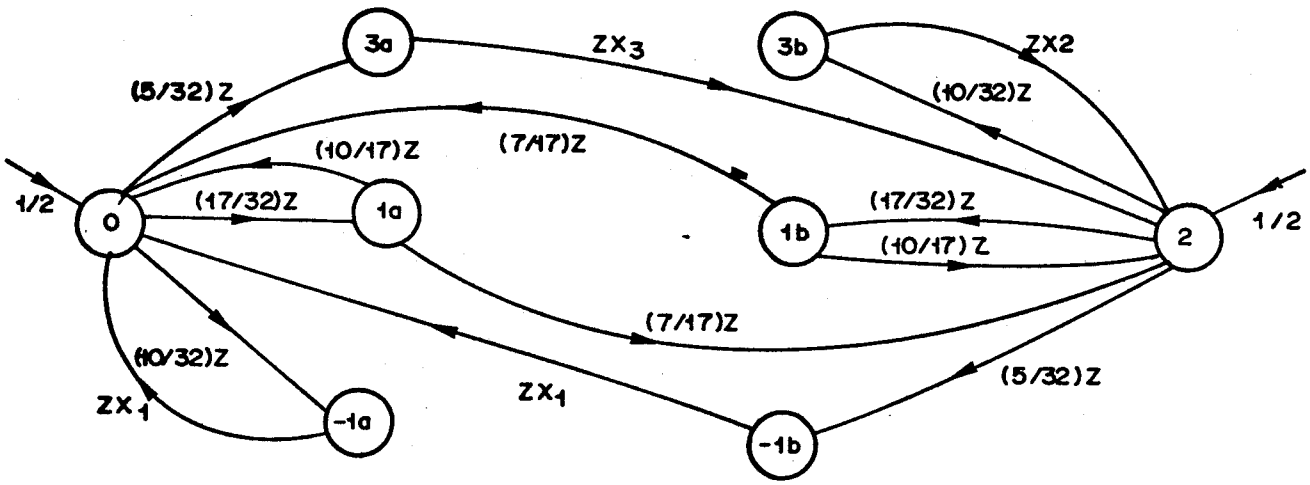


Fig. III.B.4 - Fluxograma para s = 5 bits.

Desta vez, não precisamos utilizar uma variável para marcar as passagens pelos estados 1a e 1b, pois eles não poderão ser finais e nem irão influenciar as condições de término do processo, que são as seguintes:

Estados Finais	Condições
-1a e -1b	$i_1 = 0$ e $i_3 > 0$
3a e 3b	$i_3 = 0$ e $i_1 > 0$

Sendo que $n(-1)\ P(-1) = n(3)\ P(3)$; então, podemos dizer que:

$$\bar{n} = 2\ [\bar{n}(-1)\ P(-1)]\ , \text{ onde} \qquad (121)$$

$$\bar{n}(-1)\ P(-1) = \sum_{i_2=1}^{\infty} \sum_{j=0}^{\infty} a^{(1)}_{0,1_2,j} \times j \qquad (122)$$

$$n(-1)\ P(-1) = \frac{\partial}{\partial Z} \left[P(0,1,Z) - P(0,0,Z) \right]_{Z=1} \qquad (123)$$

Resolvendo os fluxogramas, obtivemos os mesmos resultados apresentados para o caso de s=1 bit (seção III.3.b):

$\bar{n}$ = 13,3188 passos.

$\bar{t} = 6{,}1594$ palavras-código

e $$n_5(Z) = \bar{n}_1(Z) = \frac{0{,}0732\ Z^3 + 0{,}0023\ Z^5 - 0{,}0009\ Z^7}{1 - 1{,}5625\ Z^2 + 0{,}7490\ Z^4 - 0{,}1175\ Z^6 + 0{,}0056\ Z^8} \quad (124)$$

Estes resultados que apresentamos nos serviram para confirmar os obtidos através do programa BOLJM.FOT, que é baseado nos fluxogramas de macro-estados (seção III.3.c). Testando, neste programa, a mesma codificação que empregamos nos cálculos apresentados (Tab. III.4), obtivemos os mesmos resultados para os histogramas de $P_V(s)$. Sendo assim, concluímos que os diagramas de macro-estados e o programa que foi desenvolvido sobre eles constituem um meio eficaz para se analisar a sincronização, por violação da lei, de um código (n-1)B-nB.

APÊNDICE III-C

O PROGRAMA BOLJM.FOT

O objetivo deste programa é fornecer resultados sobre o sincronizabilidade de um código (n-1)B-nB, segundo o critério de ocorrência de violações na lei do código.

Com este fim, seguimos o modelamento descrito na seção III.3.C. A partir das palavras permitidas do alfabeto progres - sivo (ou alfabetos, caso haja canais de serviço), o programa nos fornece a matriz de transição do fluxograma de macro-estados (vide figura III.11). Com essa matriz, obtem-se o histograma de $P_V(s=i)$ em função do número de palavras-código necessárias para se detetar violação, assim como o valor médio e o valor máximo que esse número atinge. Define-se como número máximo de palavras código necessárias para se detetar uma violação aquele valor cuja probabilidade de ser excedido é de apenas 1%.

O programa está previsto para funcionar com $d_{max}=2$ e $d_{max}=4$. Nesse último caso existem nove macro-estados que, se n for igual a oito, pode conter até vinte estados de observação , possuindo a nossa matriz uma dimensão de até 180 x 180.

Para operá-lo, devemos criar um arquivo semelhante ao do CODJM, sendo que, se o número de canais de serviço for NCS , devemos digitar primeiramente as $(2^{n-1}-NCS)$ palavras-códigos que estão presentes em todos os alfabetos progressivos. Em seguida , inserimos as palavras restantes , que estarão presentes só em metade dos alfabetos, pois são utilizadas para comutar os valores dos canais de serviço transmitidos.

No arquivo de saída, além da caracterização do código e das palavras empregadas, teremos os coeficientes da função de transferência C(Z), obtida para os fluxogramas relativos a cada defasagem . Esses coeficientes estão armazenados no vetor C(I) , onde temos que $P_V(s) = C(I + 1)$, sendo $P_V(s)$ a probabilidade de se detetar uma violação (com defasagem de s bits) num tempo relativo a I palavras-código.

É também fornecido, no mesmo arquivo de saída, os números médio e máximo de palavras-código necessárias para o surgimento de uma proibida. Para estabelecer um limite na eficiência de sincronismo, fizemos com que se o número médio for superior a 200, o programa seja interrompido e apenas assinale este fato.

Mostramos a seguir a listagem comentada deste programa:

```
      DIMENSION IALF1(130,10),IALF2(130,10),ICS1(6,10),ICS2(6,10)
      DIMENSION ICOMT(6),A(6),W(25),PSI(3)
      DIMENSION X(5,5),Z1(5,5),Z2(5,5),SOMA(100)
      DIMENSION DIM(5,5),LL(200),MP(200)
      DIMENSION PR(9,9),IC(30)
      DIMENSION MA1(10,10),MA2(10,10),COM(5,2)
      DIMENSION VOLT1(6,4),VOLT2(5,4),R(20,20)
      DIMENSION PS(1000),P1(150),S(200),ROW(20,20),I(180,180)

C*********************
C                    *
C PROGRAMA DOBLIM.FOR *
C                    *
C*********************


OBJETIVO :
--------

          CALCULO DE PROBABILIDADES DE VIOLACAO DA LEI
          DE UM CODIGO DE BLOCOS, USANDO O SISTEMA FO-
          RA DE SINCRONISMO. CALCULO DO NUMERO MEDIO E
          MAXIMO DE PALAVRAS-CODIGO NECESSARIAS PARA
          QUE ESTA VIOLACAO ACONTECA

          ESTE PROGRAMA E PARTE INTEGRANTE DO TRABALHO
          DE TESE DE MESTRADO DE JUNO M.T.ROMANO E M.
          ARGOLLO(ORIENTADOR), INTITULADA : ' CODIGOS
          BALANCEADOS PARA TRANSMISSAO DIGITAL POR FI-
          BRAS OPTICAS '


DADOS DE ENTRADA :
----- -- -------

                    N,NUMERO DE CANAIS DE SERVICO (NCS)
                    E AS PALAVRAS PERMITIDAS DO ALFABE-
                    TO PROGRESSIVO (MATRIZES IALF1 E
                    ICS1)


SUB-ROTINA UTILIZADA :
---------- ---------

                         MINV (PARA INVERSAO DE MATRIZES)



LEITURA DE DADOS

      WRITE(5,10010)
      READ(5,10030)R
      WRITE(5,10020)
      READ(5,10030)TM
      READ(1R,10040)N,NCS
      NUM=2**(N-1)-NCS
      READ(1R,10050)((IALF1(I,J),J=2,(N+1)),I=1,NUM)
      IF(NCS.EQ.0)  GO TO 10
      READ(1R,10060)((ICS1(I,J),J=2,N+1),I=1,(2*NCS))
```

```
      PARTE I : CARACTERIZACAO DO CODIGO
                ------------- -- ------
      * DIFERENCA DO ALFABETO AGRESSIVO
      * DIFERENCAS DOS V-PESOS S(I) E PSI(I)
      * ENTROPIA DE ORDEM (IORD)

      DO 20 I=1,5
      ICONT(I)=0
      CONTINUE
      DO 40 I=1,NDIM
      ND=0
      DO 30 J=2,M+1
      IALF2(I,J)=1-IALF1(I,J)
      ND=ND+(M*IALF1(I,J)-1)
      CONTINUE
      IALF1(I,1)=ND
      IALF2(I,1)=-ND
      ND=ND/2+1
      ICONT(ND)=ICONT(ND)+2
      CONTINUE
      IF(NDIS.EQ.0)  GO TO 70
      DO 60 I=1,(2*NDIS)
      ND=0
      DO 50 J=2,M+1
      ICS2(I,J)=1-ICS1(I,J)
      ND=ND+(M*ICS1(I,J)-1)
      CONTINUE
      ICS1(I,1)=ND
      ICS2(I,1)=-ND
      ND=ND/2+1
      ICONT(ND)=ICONT(ND)+1
      CONTINUE
      ND=0
      IF(ICONT(ND).NE.0)  GO TO 90
      NC=NC-1
      GO TO 80
      IDIM=2*(NC-1)

C     VETOR W(I) NOS DA A PROBABILIDADE DE OCORRENCIA
C     DE UMA PALAVRA DE DISPARIDADE 2*(I-1) PARA O ALFA-
C     BETO PROGRESSIVO

      DO 100 I=1,NC
      W(I)=FLOAT(ICONT(I))/(2.**N)
      CONTINUE

C     VETOR PSI(I) NOS DA A PROBABILIDADE DE SE
C     ESTAR NUM ESTADO DE SOMA = 2*(I-1)

      IF(IDIM.NE.2)  GO TO 110
      PSI(1)=0.5
      GO TO 310
      DO 180 I=1,NC-1
      DO 170 J=1,NC-1
      IF((I-(NC-1)))  120,160,150
      IF(I-J)  130,140,150
      A(I,J)=0.
      GO TO 170
      A(I,J)=W(1)-1.
      GO TO 170
```

```
150       L=I-J+1
          X1(I,J)=G(L)
          GO TO 170
160       X1(I,J)=1.
170       CONTINUE
180       CONTINUE
          DO 230 I=1,NC-1
          DO 220 J=1,NC-1
          IF(I-(NC-1))  190,200,200
190       IF(I+J-NC)200,210,210
200       X2(I,J)=0.
          GO TO 220
210       L=2*NC-1-J
          X2(I,J)=G(L)
220       CONTINUE
230       CONTINUE
          DO 250 I=1,NC-1
          DO 240 J=1,NC-1
          Y(I,J)=Y1(I,J)+Y2(I,J)
240       CONTINUE
250       CONTINUE
          L=0
          DO 270 I=1,NC-1
          DO 260 J=1,NC-1
          L=L+1
          AY(L)=Y(J,I)
260       CONTINUE
270       CONTINUE
          NDIM=NC-1
          CALL MINV(AY,NDIM,DET,LS,MS)
          L=0
          DO 290 I=1,NC-1
          DO 280 J=1,NC-1
          L=L+1
          YINV(J,I)=AY(L)
280       CONTINUE
290       CONTINUE
          DO 300 I=1,NC-1
          PSI(I)=YINV(I,NDIM)/2.
300       CONTINUE
310       NP=N-1
          WRITE(IW,1000)ME,N,ID1
          WRITE(IW,1050)
          DO 320 I=1,NDIM
          WRITE(IW,1060)(AUGP1(I,J),J=2,N+1)
320       CONTINUE
          IF(NCS.EQ.0)  GO TO 340
          WRITE(IW,1100)
          DO 330 I=1,2**NCS
          WRITE(IW,1110)(ICS1(I,J),J=2,N+1)
330       CONTINUE

C
C         PARTE II : OBTENCAO DO FLUXOGRAMA DE MACRO-ESTADOS
C                    -------- -- ---------- -- -------------
C         * CALCULO DA MATRIZ P(I,J)
C         * CALCULO DA MATRIZ T(I,J)
C         * CALCULO DO VETOR PO(I)
C
          DO 1130 L=1,N-1
          DO 350 I=1,10
```

```
      DO 350 J=1,10
      MAT1(I,J)=0
      MAT2(I,J)=0
350   CONTINUE
360   CONTINUE
      DO 380 I=1,NUM
      ND1=0
      ND2=0
      DO 370 J=IS+2,N+1
      ND1=ND1+(2*IALF1(I,J)-1)
      ND2=ND2+(2*IALF2(I,J)-1)
370   CONTINUE
      M1=(IALF1(I,1)-ND1+IS)/2+1
      K1=(ND1+N-IS)/2+1
      M2=(IALF2(I,1)-ND2+IS)/2+1
      K2=(ND2+N-IS)/2+1
      MAT1(M1,K1)=MAT1(M1,K1)+1
      MAT2(M2,K2)=MAT2(M2,K2)+1
380   CONTINUE
      IF(NCS.EQ.0)   GO TO 410
      DO 400 I=1,(2*NCS)
      ND1=0
      ND2=0
      DO 390 J=IS+2,N+1
      ND1=ND1+(2*ICS1(I,J)-1)
      ND2=ND2+(2*ICS2(I,J)-1)
390   CONTINUE
      M1=(ICS1(I,1)-ND1+IS)/2+1
      K1=(ND1+N-IS)/2+1
      M2=(ICS2(I,1)-ND2+IS)/2+1
      K2=(ND2+N-IS)/2+1
      MAT1(M1,K1)=MAT1(M1,K1)+1
      MAT2(M2,K2)=MAT2(M2,K2)+1
400   CONTINUE
410   DO 430 I=1,IS+1
      IX1=0
      IX2=0
      DO 420 J=1,N+1-IS
      IX1=IX1+MAT1(I,J)
      IX2=IX2+MAT2(I,J)
420   CONTINUE
      PR(I,1)=FLOAT(IX1)/(2.**N)
      PR(I,2)=FLOAT(IX2)/(2.**N)
430   CONTINUE
      K1=1
      K2=NC-1
      K3=1
      NINC=0
      DO 470 L=1,2
      I1=0
      DO 460 I=K1,K2,K3
      I1=I1+1
      J1=0
      DO 450 J=1,IS+1
      AUX=PSI(I)*PR(J,L)
      IF(AUX)   440,450,440
440   NINC=NINC+1
      J1=J1+1

C VETOR PL(I) F... AS CONDICOES INICIAIS DO
```

```
      PROCESSO MARKOVIANO

      PR(NINC)=U+LE(NRA)
      ID(NINC)=IDM*(L-1)+2*(I1-1)+2*(J-1)-IS
      IDA(J1,L)=PR(I,J)
450   CONTINUE
460   CONTINUE
      K1=NC-1
      K2=1
      K3=-1
470   CONTINUE
      DO 490 I=1,NINC/IDM
      DO 480 J=1,NC
      VOUT1(I,J)=0
      VOUT2(I,J)=0
480   CONTINUE
490   CONTINUE
      L1=0
      L2=0
      DO 580 I=1,IS+1
      A1=-1.*PR(I,1)
      A2=-1.*PR(I,2)
      IF(.NOT.A1.AND..NOT.A2)  GO TO 580
      IF(PR(I,1))  510,510,500
500   L1=L1+1
510   IF(PR(I,2))  530,530,520
520   L2=L2+1
530   DO 570 J=1,N+1-IS
      M=2*(I-1)-IS
      K=2*(J-1)+IS-N
      IF(MAT1(I,J))  550,550,540
540   MK=(M+K)/2+1
      VOUT1(L1,MK)=FLOAT(MAT1(I,J))/((2.**N)*PR(I,1))
550   IF(MAT2(I,J))  570,570,560
560   MK=1-(M+K)/2
      VOUT2(L2,MK)=FLOAT(MAT2(I,J))/((2.**N)*PR(I,2))
570   CONTINUE
580   CONTINUE

      A MATRIZ A NOS DA A PROBABILIDADE DE TRANSICAO
      ENTRE OS DIVERSOS ESTADOS DE OBSERVACAO

      DO 600 M=1,NINC
      DO 590 K=1,NINC
      A(M,K)=0.
590   CONTINUE
600   CONTINUE
      J1=NC
      J2=1
      DO 710 KALF=1,2
      DO 700 KO=1,IDM/2
      DO 690 I=1,NINC/IDM
      J3=0
      DO 680 J=1,J1,J2
      J3=J3+1
      DO 670 L=1,NINC/IDM
      M=(KALF-1)*(NINC/2)+(KO-1)*(NINC/IDM)+I
      K=(KALF-1)*(NINC/2)+(KO+J-2)*(NINC/IDM)+L
      GO TO (610,640),KALF
      IF((KO+J-1)-IDM/2)  620,620,630
```

```
620       R(*,K)=VOLT1(I,T)*(DA(L,1)
          GO TO 670
630       R(*,K)=VOLT1(I,T)*(DA(L,2)
          GO TO 670
640       IF(KC-JJ)  650,660,660
650       R(*,*)=VOLT2(I,JJ)*(DA(L,1)
          GO TO 670
660       R(*,K)=VOLT2(I,T)*(DA(L,2)
670       CONTINUE
680       CONTINUE
690       CONTINUE
700       CONTINUE
          JJ=KC-1
          JJ=-1
710       CONTINUE
720       CONTINUE
          DO 740 I=1,NINC
          DO 730 J=1,NINC
          RD(I,J)=DBLE(R(I,J))
730       CONTINUE
740       CONTINUE
C
C         A MATRIZ R E A MATRIZ DE TRANSICAO DO
C         PROGRAMA TOTAL
C
          GO TO (750,670),IC+1/2
750       DO 770 M=1,3*NINC+1
          DO 760 K=1,4*NINC
          R(M,K)=0.000
760       CONTINUE
770       CONTINUE
          DO 790 I=1,NINC
          DO 780 J=1,NINC
          N=ID(J)-ID(I)
          IF(N.EQ.0)  GO TO 790
          IF(N.EQ.1)  GO TO 810
          IF(N.EQ.-2)  GO TO 820
          DO 780 L=1,2*NINC,NINC
          R(L+I,3*NINC+J)=RD(I,J)
780       CONTINUE
          GO TO 830
790       DO 800 L=1,2*NINC,NINC
          R(L+I,L+J)=RD(I,J)
800       CONTINUE
          GO TO 830
810       R(I,NINC+J)=RD(I,J)
          R(NINC+I,3*NINC+J)=RD(I,J)
          R(2*NINC+I,NINC+J)=RD(I,J)
          GO TO 830
820       R(I,2*NINC+J)=RD(I,J)
          R(NINC+I,2*NINC+J)=RD(I,J)
          R(2*NINC+I,3*NINC+J)=RD(I,J)
830       CONTINUE
840       CONTINUE
          DO 860 I=1,3*NINC+1
          G=0.000
          DO 850 J=3*NINC+1,4*NINC
          G=G+R(I,J)
850       CONTINUE
          R(I,3*NINC+1)=G
```

```
      DO 1 I=1,6
      L=3*NINC+I
      DO 10 J=1,L
      DO 20 K=1,9*NINC
      R(K,J)=0.0D0
   10 CONTINUE
   20 CONTINUE
      DO 480 I=1,NINC
      DO 470 J=1,NINC
      NJ=ID(J)-ID(I)
      IF(NJ.EQ.0) GO TO 410
      IF(NJ.EQ.1) GO TO 430
      IF(NJ.EQ.-2) GO TO 440
      IF(NJ.EQ.2) GO TO 450
      IF(NJ.EQ.-4) GO TO 460
      DO 400 L=1,7*NINC,NINC
      R(L+I,3*NINC+J)=RD(I,J)
  400 CONTINUE
      GO TO 470
  410 DO 420 L=1,7*NINC,NINC
      R(L+I,L+J)=RD(I,J)
  420 CONTINUE
      GO TO 470
  430 R(I,3*NINC+J)=RD(I,J)
      R(NINC+I,2*NINC+J)=RD(I,J)
      R(2*NINC+I,3*NINC+J)=RD(I,J)
      R(3*NINC+I,5*NINC+J)=RD(I,J)
      R(4*NINC+I,6*NINC+J)=RD(I,J)
      R(5*NINC+I,6*NINC+J)=RD(I,J)
      R(6*NINC+I,5*NINC+J)=RD(I,J)
      R(7*NINC+I,6*NINC+J)=RD(I,J)
      GO TO 470
  440 R(I,3*NINC+J)=RD(I,J)
      R(NINC+I,3*NINC+J)=RD(I,J)
      R(2*NINC+I,6*NINC+J)=RD(I,J)
      R(3*NINC+I,6*NINC+J)=RD(I,J)
      R(4*NINC+I,5*NINC+J)=RD(I,J)
      R(5*NINC+I,6*NINC+J)=RD(I,J)
      R(6*NINC+I,8*NINC+J)=RD(I,J)
      R(7*NINC+I,6*NINC+J)=RD(I,J)
      GO TO 470
  450 R(I,NINC+J)=RD(I,J)
      R(NINC+I,2*NINC+J)=RD(I,J)
      R(2*NINC+I,3*NINC+J)=RD(I,J)
      R(3*NINC+I,4*NINC+J)=RD(I,J)
      R(4*NINC+I,5*NINC+J)=RD(I,J)
      R(5*NINC+I,3*NINC+J)=RD(I,J)
      R(6*NINC+I,4*NINC+J)=RD(I,J)
      R(7*NINC+I,5*NINC+J)=RD(I,J)
      GO TO 470
  460 R(I,2*NINC+J)=RD(I,J)
      R(NINC+I,2*NINC+J)=RD(I,J)
      R(2*NINC+I,3*NINC+J)=RD(I,J)
      R(3*NINC+I,7*NINC+J)=RD(I,J)
      R(4*NINC+I,6*NINC+J)=RD(I,J)
      R(5*NINC+I,7*NINC+J)=RD(I,J)
      R(6*NINC+I,8*NINC+J)=RD(I,J)
      R(7*NINC+I,8*NINC+J)=RD(I,J)
  470 CONTINUE
```

```
 900      CONTINUE
          DO 1000 I=1,M+NFIN+1
          G=1.000
          DO 990 J=M+NFIN+1,MAXIND
          G=G+T(I,J)
 990      CONTINUE
          IF(G.GT.C(M+NFIN+1))G=G
 1000     CONTINUE
          M=M+NFIN+1
C
C
C         PARTE III : OBTENÇÃO DA FUNÇÃO DE TRANSFERÊNCIA
C                     -------- -- ------ -- ------------
C         * CÁLCULO DE SEUS COEFICIENTES (VETOR C(I))
C         * CÁLCULO DE VM E NFIN
C
 1010     G=0.000
          P=0.9900
          NFIN=200
          DO 1020 I=MAXIND+1,LX
          P1(I)=0.000
 1020     CONTINUE
          M=2
C
C
C         OS COEFICIENTES DA FUNÇÃO DE TRANSFERÊNCIA,DADOS NO
C         VETOR C(I),NOS FORNECE O HISTOGRAMA DA PROBABILIDADE
C         DE VALIDAÇÃO EM FUNÇÃO DO NÚMERO DE PALAVRAS-CÓDIGO
C         NECESSÁRIAS PARA SE OBTER-LA
C
          C(1)=0.000
 1030     DO 1050 I=1,LX
          P1(I)=0.000
          DO 1040 J=1,LX
          P1(I)=P1(I)+PO(J)*T(I,J)
          PO=P1(J)
 1040     CONTINUE
 1050     CONTINUE
          DO 1060 I=1,LX
          PO(I)=P1(I)
 1060     CONTINUE
          C(M)=PO(LX)
          G=G+C(M)
          IF(G-P)  1080,1070,1070
C
C
C         NFIN É O NÚMERO MÁXIMO DE PALAVRAS-CÓDIGO NECESSÁ-
C         RIAS PARA SE OBTER UMA VIOLAÇÃO
C
 1070     NFIN=M-1
          P=2.000
 1080     M=M+1
          IF(M-2*NFIN)  1030,1030,1090
C
C
C         VM É O NÚMERO MÉDIO DE PALAVRAS-CÓDIGO NECESSÁRIAS
C         PARA SE OBTER UMA VIOLAÇÃO
C
 1090     VM=0.000
          DO 1100 I=2,2*NFIN
          VM=VM+C(I)*CONJAC(I-1)
 1100     CONTINUE
          WRITE(IW,1110)VM
          M=1
```

```
      K2=11
1111  WRITE(I6,1113)(C(I),I=K1,K2)
      IF(K2.GE.(NFIN+1))  GO TO 1120
      K1=K1+N1
      K2=K2+N1
      IF(K2.GE.(NFIN+1))  K2=(NFIN+1)
      GO TO 1110
1120  WRITE(I6,1114)V,MEAN
1130  CONTINUE
      STOP
1.010 FORMAT(//,10X,' NUMERO DE ENTRADA = ',I)
1.020 FORMAT(//,10X,' NUMERO DE SAIDA = ',I)
1.030 FORMAT(I2,I2)
1.040 FORMAT(I2,I2)
1.050 FORMAT(8I2)
1.060 FORMAT(8I2)
1.070 FORMAT(//,' CODIGO DO TIPO',I2,'A-',I1,'B',' COM NMAX=',I2,//)
1.080 FORMAT(//,' DISTRIBUICAO PROGRESSIVA',/)
1.090 FORMAT(10I2)
1.100 FORMAT(//,' PALAVRAS UTILIZADAS PARA CANAIS DE SERVICO',/)
1.110 FORMAT(10I2)
1.120 FORMAT(//,' COEF. DE FUNCAO DE TRANSFERENCIA PARA IS=',I2,/)
1.130 FORMAT(10F13.9)
1.140 FORMAT(//,' TAMANHO MEDIO=',F13.9,' NUMERO MAXIMO=',I4,/)
      END
```

CAPÍTULO IV

CONSIDERAÇÕES FINAIS

IV.1 - Resultados Obtidos sobre a Sincronizabilidade

Nosso objetivo neste capítulo é fornecer e comentar os diversos resultados que obtivemos sobre a sincronizabilidade dos códigos, seja por palavra proibida como por violação da lei. Esses resultados foram conseguidos através dos dois programas apresentados no capítulo anterior.

Vamos deter nossos comentários sobre os códigos que demonstramos ser de maior interesse para sistemas de 140 Mb/s, ou seja, o 3B-4B, o 7B-8B e o 5B-6B.

a) Código 3B-4B:

Segundo a tabela II.6.b, este código opera com dois estados da SDCT (SDCT=0,2, pois d_{max}=2) e um desbalanço máximo de no mínimo P=4. Detendo-se ainda nessa mesma tabela, podemos observar, pela coluna relativa a n=2, que todas as palavras disponíveis para o alfabeto regressivo atingem uma disparidade intermediária igual a 0 ou -2, para os seus dois primeiros bits . Isto nos permite concluir que, para uma defasagem s = 2 bits no alinhamento, serão gerados os seguintes estados de observação pelo estado real de SDCT=2;

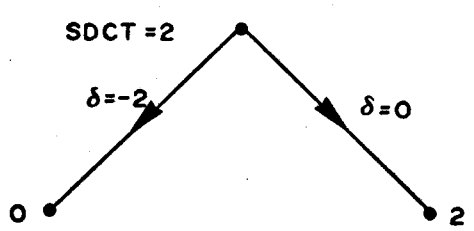


Fig. IV.1 - Estados de Observação Gerados para s=2 bits.

onde δ corresponde aos possíveis valores de disparidade acumulada para os primeiros dois bits de cada palavra .

É claro que, por simetria, o estado real de SDCT=0 irá gerar estados de observação com disparidades terminais iguais a 0 e +2.

Concluímos então que não haverá transições proibidas entre esses estados, uma vez que todos possuem disparidades terminais 0 ou +2, que são os valores correspondentes às dos estados reais, entre os quais todas as transições são permitidas . Portanto o código 3B-4B não será sincronizável, para s=2 bits e desbalanço P=4, pela ocorrência de violações na lei do código.

As palavras geradas para essa defasagem terão disparidade 0 ou -2 (para o alfabeto regressivo). Se estivermos trabalhando com um canal de serviço, todas as palavras que possuem esses valores de disparidade estarão sendo utilizadas na codificação, de forma que o código torna-se também não-sincronizável por palavra proibida.

Esses obstáculos só são transponíveis se elevamos o desbalanço para P=6. Sendo assim, nos parece mais razoável a utilização do código 5B-6B, que também apresenta P=6 e possui um excesso de taxa mais tolerável para sistemas de alta velocidade.

Estas particularidades, que observamos na Tabela II.6. b, nos possibilitam tirar conclusões mais gerais, ou seja, podemos perceber que um confinamento maior da SDC prejudica a sincronização por violação da lei, pois torna mais rara a ocorrência de estados de observação com disparidades terminais mais extremas, isto é, com valores mais distantes dos relativos aos estados reais, sendo que esses estados são justamente os principais responsáveis pelas transições proibidas.

Nota-se então um compromisso entre sincronizabilidade e balanceamento e as exigências de cada projeto deve possibilitar a opção entre o favorecimento de um ou outro fator.

Os dados obtidos por computador para o 5B-6B, expostos mais adiante, confirmaram essa piora de sincronizabilidade para um maior balanceamento.

b) Código 7B-8B

Para operarmos com esse código mantendo um desbalanço P=10, é necessário, conforme podemos observar pela tabela II . 12.c, proibir as seguintes palavras do alfabeto progressivo:

- Todas as palavras de disparidade terminal +6 e +8.
- As palavras: 00001111, 00010111, 00011011, 00011101, 00011110, 00100111, 01000111 e 10000111 de disparidade zero,
- E a palavra 00011111 de disparidade igual a +2.

Restam ainda um número grande de candidatas, de forma que a variedade de tabelas de codificação para este código útil é muito grande (vide tabela II.15).

A título de ilustração, testamos primeiramente uma tabela que visa minimizar a excursão RMS da soma digital corrida, excluindo as palavras que atingem os valores limites de disparidade intermediária, principalmente aquelas que os atingem mais de uma vez. Essa escolha é feita por inspeção da tabela II.12.c.

Foram proibidas então as seguintes palavras-código: 00101011, 00101101, 00110011, 00110101, 01001011, 01001101, 01010011, 01100011, 10100011, 10010011, 10001110, 10001101, 10001011, 00101111, 00110111, 00111011, 00111101 e 00111110.

Obtivemos assim os seguintes valores, efetuando a sincronização por palavra proibida, conforme mostrados na Tabela IV.1.

Testando essa mesma codificação no programa BOLJM.FOT, conseguimos um valor médio bem menor para o número de palavras código necessárias. Na tabela IV.2 mostramos também o seu valor máximo n_{max}:

Defasagem	P_V=Prob. de violação(%)	$\bar{n}$=nº médio de pal.código
s = 1 bit	5,238	19,09
s = 2 bits	4,944	20,23
s = 3 bits	5,476	18,26
s = 4 bits	5,219	19,16
s = 5 bits	6,898	14,50
s = 6 bits	6,559	15,25
s = 7 bits	7,455	13,41

Tabela IV.1 - Sincronizabilidade por palavra proibida para um código 7B-8B.

Pode-se notar que: para esta codificação, o critério de sincronização por violação da lei apresenta-se bem mais vantajoso. A adição de algumas palavras novas para permitir a trasmis - são de canais de serviço não provocou variações sensíveis nestes dois resultados:

s(bits)	$\bar{n}$ (palavras-código)	n_{max}(palavras-código)
1	8,369	26
2	10,74	42
3	8,827	28
4	9,964	38
5	8,610	27
6	10,28	40
7	8,553	27

Tabela IV.2 - Sincronizabilidade por violações da lei para um código 7B-8B.

Para efeito de comparação, testamos uma outra tabela que não reduz o desbalanço RMS, permitindo as palavras que atingem os valores máximos de disparidades intermediárias e proibin-

do as seguinte: 01111001, 01111100, 10011110 e 10111100, de disparidade terminal +2 e 01101111, 01111011, 01111110, 10101111, 10111011, 10111110, 11010111, 11011101, 11100111, 11101101, 11110011, 11110110 e 11111010 de disparidade +4.

Encontramos os respectivos resultados para violação por palavra proibida e por lei do código:

s(bits)	P_V (%)	$\bar{n}$(palavras-código)
1	10,578	9,454
2	10,435	9,583
3	11,141	8,976
4	11,928	8,384
5	10,528	9,498
6	10,375	9,639
7	10,358	9,654

Tabela IV.3 - Sincronizabilidade por palavra proibida para outro código 7B-8B.

s(bits)	n(Palavras-código)	n_{max}(palavras-código)
1	8,833	28
2	9,323	35
3	8,739	27
4	9,284	35
5	8,631	27
6	9,522	36
7	8,984	28

Tabela IV.4 - Sincronizabilidade por violação da lei para outro código 7B-8B.

Como se pode ver, a degradação no desbalanço RMS produziu uma melhora da sincronizabilidade. Esta melhora foi bem sig

nificativa no caso de violação por ocorrência de palavra proibida.

Conforme iremos ver mais adiante, é possível desenvolver algumas estratégias de sincronismo onde o tempo de recuperação depende apenas do número médio de palavras-código necessárias para se detetar violação na defasagem de pior caso, ou seja, naquela defasagem que apresenta um valor maior para $\bar{n}$. Optando-se por este tipo de estratégia, o melhor resultado seria o da tabela IV.4, onde o valor máximo de $\bar{n}$ foi 9,522 (s=6 bits).

Este número pode ser considerado um pouco elevado, comparando-se por exemplo aos resultados obtidos para o 5B-6B. É de se esperar que existem codificações mais eficientes, quanto ao sincronismo, para o 7B-8B, se estivermos dispostos a sacrificar o balanceamento.

A inclusão de todas as palavras de disparidade zero melhora, geralmente, a sincronizabilidade. Entretanto, isto vai acarretar um aumento da excursão da SDC para P=14.

c) Código 5B-6B:

Este código opera com desbalanço P=6 e excesso de velocidade Δ= 20%, mostrando condições para transmissão em 140Mb/s.

A fim de manter P=6, devemos proibir todas as palavras de disparidade $\pm$ 4 e $\pm$ 6, além de omitir 000111 e 111000 dos alfabetos progressivo e regressivo, respectivamente. Sobram 34 palavras para codificar 32 blocos de entrada, de forma que é possível transmitir no máximo dois canais de serviço sem prejuízo do balanceamento nem da suavidade espectral.

A inclusão desses canais alteraram muito pouco os valores obtidos para P_V e $\bar{n}$, seja para sincronizabilidade por palavra proibida como por violação da lei do código.

Os melhores valores encontrados nos casos de sincronização por ocorrência de palavras proibidas deram-se quando selecionamos para comutar os canais de serviço, seguindo a nomenclatura já usada na seção III.5, as palavras:

$p_1 = 110011$

$p_2 = 100111$

$p'_1 = 110110$

$p'_2 = 101101$

Existem então quatro pares distintos de alfabetos e o resultado final, mostrado na tabela IV.5, é calculado ao se aplicar a equação (105) sobre os valores de P_V obtidos para cada um deles.

s(bits)	P_V (%)	$\bar{n}$ (palavas-código)
1	14,967	6,804
2	15,210	6,575
3	14,430	6,481
4	15,308	6,532
5	14,844	6,737

Tabela IV.5 - Sincronizabilidade por palavra-proibida para um código 5B-6B-

Efetuando a sincronização por ocorrência de violações na lei do código para essa mesma codificação, encontramos os resultados expostos na tabela IV.6.

s (bits)	$\bar{n}$ (palavras-código)	n_{max}(palavras-código)
1	6,557	23
2	7,591	34
3	5,839	21
4	7,591	34
5	6,557	23

Tabela IV.6 - Sincronizabilidade por violação da lei para um código 5B-6B.

Nesta tabela, observamos que os maiores valores de $\bar{n}$ e n_{max} ocorrem para s=2 e 4 bits. Mudando as palavras empregadas na comutação dos canais de serviço, conseguimos diminuir um pouco esses valores, piorando entretanto o comportamento verificado, em média, para as diversas defasagens. O menor valor de pior caso para $\bar{n}$ foi obtido para :

p_1 = 101010

p_2 = 011001

p'_1 = 010101

p'_2 = 100110 ,

conforme indicado na tabela IV.7.

s (bits)	n (palavras-código)	n_{max} (palavras-código)
1	6,873	24
2	7,536	34
3	5,864	21
4	7,536	34
5	7,103	25

Tabela IV.7 - Sincronização por violação da lei para outro código 5B-6B.

Por fim, passamos a permitir a palavra código 000111 proibindo , em seu lugar, a 101101 e aumentando o desbalanço para P=8. Esta degradação no balanceamento nos forneceu uma melhora razoável na sincronizabilidade.

Os melhores resultados foram encontrados para:

p_1 = 100111

p_2 = 110011

p'_1 = 110101

p'_2 = 111001

e estão reunidos nas tabelas IV.8 e IV.9.

s(bits)	P_v(%)	$\bar{n}$ (palavras-código)
1	20,313	4,923
2	20,239	4,941
3	21,582	4,643
4	20,183	4,882
5	20,069	4,983

Tabela IV.8 - Sincronizabilidade por palavra proibida para um código 5B-6B com P=8.

s(bits)	$\bar{n}$ (palavras-código)	n_{max} (palavras-código)
1	6,464	23
2	6,055	27
3	4,948	18
4	6,055	27
5	6,661	24

Tabela IV.9 - Sincronizabilidade por violação da lei para um código 5B-6B com P=8.

IV.2 - Os Tempos de Retenção e de Recuperação:

Os valores obtidos através dos programas permitem o cálculo do tempo de recuperação de sincronismo para uma determinada estratégia de alinhamento adotada para o sistema de comunicação. A escolha dessa estratégia é fundamentalmente uma questão de projeto, sobre a qual não é nosso objetivo encaminhar soluções.

A estratégia deve ser tal que decida, em função da densidade de violações detetadas num espaço de tempo sobre o alinhamento ou não do sistema. Por exemplo, a solução encontrada por

Hosoe [1] para o sistema ELO-34 consistia na contagem de violações a cada 512 blocos emitidos pelo codificador. Se fossem encontradas 16 ou mais palavras proibidas, o sistema era considerado fora de sincronismo.

Existe também a possibilidade de se criar estados de busca [18], sobre os quais o sistema evolue conforme a ocorrência ou não de algum tipo de violação.

Essas duas espécies de estratégia podem ser analisadas como processos de Markov, possibilitando o cálculo dos tempos de recuperação e retenção de sincronismo. O sistema será mais sincronizável à medida que a probabilidade de ocorrência de violação for máxima, para as possíveis defasagens, e a probabilidade de erros de linha for mínima.

Por fim, entendemos que a busca do sincronismo pode ser executada de duas maneiras diferentes.

Primeiramente, através de um único circuito contador de violações, de forma que ao se perceber uma perda de sincronismo, seria enviada uma ordem de deslocamento de um bit sobre a sequência recebida pelo decodificador. Esse procedimento seria feito até encontrar-se a fase correta para a decodificação da mensagem. Uma busca desse tipo será mais eficiente quanto menor for o valor médio de $\overline{n}$, calculado sobre todas as possíveis defasagens.

Pode-se porém empregar um contador para cada fase possível da sequência transmitida. Sendo assim, quando a fase considerada correta fornecer uma alta densidade de violações, o sistema irá imediatamente em busca daquela que apresentasse a menor densidade, descobrindo assim de quantos bits, o sistema estava desalinhado. Neste caso, o receptor seria mais complexo, mas teríamos uma recuperação mais rápida do sincronismo e nos preocuparíamos em minimizar apenas o pior caso de $\overline{n}$.

IV.3 - Conclusão:

A apresentação dos resultados neste capítulo se propõe a dar subsídios para uma decisão de projeto sobre a codificação e a forma de se efetuar o sincronismo em sistemas de altas ta -

xas de transmissão. Esta decisão irá depender da classe de código escolhida, do tipo de estratégia empregada e das condições impostas sobre o balanceamento e a sincronizabilidade.

Conforme o caso, obtivemos melhores resultados para a sincronização por palavra proibida ou por violação da lei. Caso se opte por esse último critério, deve-se usar os diagramas das figuras III.3 e III.4 (conforme d_{max} for dois ou quatro) para a implementação de um circuito detetor de violações.

A teoria desenvolvida nos capítulos II e III indica, para o caso concreto de transmissão em 140 Mb/s., o código 5B - 6B como o mais conveniente. Ela pode porém, juntamente com os programas, servir de instrumento de análise para outros sistemas e outras codificações podem ser testadas, a fim de se obter resultados sobre a sincronizabilidade, de acordo com o interesse prático.

BIBLIOGRAFIA

{1} - HOSOE, P.T. e D.S. ARANTES, "Codificação Digital para Comunicações Ópticas", FEC/UNICAMP, Fev./83.

{2} - BALICCO, E. e G. IUDICELLO, "Binary Coding in Fiber Optic Digital Transmission with Respect to Timing Extration and Error Monitoring", TELETRA S.p.A., Mar./76.

{3} - PERSONICK, S.D., "Optical Fiber Transmission Systems" , Plenum Press, New York, 1981.

{4} - CHIQUITO,J.G. e H. WALDMAN, "Equalização Variável em Repetidores MCP", FEC/UNICAMP, Dez./83.

{5} - PERSONICK,S.D., "Baseband Linearity and Equalization in Fiber Optic Digital Communication Systems", The Bell System Technical Journal, Vol.52, pp.1175-1194, 1973.

{6} - MOSCHIM, E. e H. WALDMAN, "Um Estudo sobre Sistemas de Transmissão Digital por Fibras Ópticas", FEC/UNICAMP, Jul./83.

{7} - PERSONICK,S.D., "Receiver Design for Digital Fiber Optical Communication Systems I", The Bell System Technical Journal, Vol. 52, pp. 843-886, 1973.

{8} - MUOI, T.V. and J.L. HULLET, "Receiver Design for Multile - vel Digital Optical Fiber Systems", IEEE Transactions on Communications, Sept./75.

{9} - WALDMAN, H.; C. BEZZAN; E. MOSCHIM e C. ALMEIDA, "Modelos Básicos para o Detetor e o Amplificador de Receptores Ópticos", NT nº 2, FEC/UNICAMP, Fev./82.

{10} - Technical Staff of CSELT - "Optical Fibre Communication", Torino, Italy; Mc Graw-Hill Book Company, 1980.

{11} - WALDMAN, H. e J.M.T. ROMANO, "Existência e Variedade de Códigos Úteis do Tipo mB-nB", 2º Simpósio Brasileiro de Telecomunicações, Campinas, Set./84.

{12} - WALDMAN, H. e J.M.T. ROMANO, "Limites Fundamentais de Eficiência em Códigos mB-nB", 2º Simpósio Brasileiro de Telecomunicações, Campinas, Set./84.

{13} - BENNETT, W.R., "Statistics of Regenerative Digital Transmission", The Bell System Technical Journal, Vol. 37, pp. 1501-1542, 1958.

{14} - ROUSSEAU, M., "Block-Codes for Optical-Fibre Communication", Electronics Letters, Vol. 12, pp. 478-479 , July./76.

{15} - ANGOT,A.,"Compléments de Mathématiques", Collection Technique et Scientifique du C.N.E.T., Revue d'Optique , Paris, 1961.

{16} - HOHN, F.E., "Elementary Matrix Algebra", The Mac Millan Company, New York, 1973.

{17} - WALDMAN,H. and J.M.T. ROMANO, "Balance Versus Efficiency in mB-nB Codes", a ser publicado, Campinas, 1984.

{18} - BRUGIA,O.; M.DÈCIMA and U. DE JULIO, "Character Alignment for High-Capacity PCM Systems Using MS43 Line Code", IEEE Transactions on Communications, Vol. COM-23, pp 803-813, Aug./75.

{19} - MASON, S.J., "FeedBack Theory - Some Properties of Signal Flow Graphs", Proceedings of the IRE, pp. 1144-1156, Sep./53.

{20} - SITTLER,R.W., "Systems Analysis of Discrete Markov Processes", IRE Transactions on Circuit Theory, pp. 257 - 266, Dec./56.

{21} - HUGGINS,W.H., "Signal-Flow Graphs and Random Signals" , Proceedings of the IRE, pp. 74-86, Jan./57.

{22} - BALDINI,R.F. e D.S.ARANTES, "Estratégias de Sincronização de Quadro para Sistemas AMDT", FEC/UNICAMP, Set./83.