

Cap. 03 – Controle de Erro

3.1 – Introdução

3.2 – Modelo de Erro

3.3 – Tipos de Erros de Transmissão

3.4 – Redundância de Dados

3.5 – Tipos de Códigos de Erros

3.6 – Verificação por Paridade

3.7 – Correção de Erros

3.8 – Código de Bloco Linear

3.9 – Verificação por Redundância Cíclica

3.10 – Verificação por Soma Aritmética

Referências Bibliográficas

- Gerard J. Holzmann – “Design and Validation of Computer Protocols” – Prentice Hall; Englewood Cliffs; New Jersey; 1991.
- Andrew S. Tanenbaum - “Computer Networks” - Prentice Hall; Englewood Cliffs; New Jersey; 1989; ISBN 0-13-166836-6
- Paulo Coelho - “Material de Aula” - Arquitetura de Redes de Computadores (FACOM49070 - Mecatrônica)
- Pedro Frosi - “Material de Aula” - Arquitetura de Redes de Computadores (GBC056 - Ciência da Computação)

3.1 - Introdução

- “estatísticas” - ... nro de erros causados na transmissão de dados são muitas ordens de grandeza do nro de erros causados por falhas de “hardware” dos sistemas computacionais;
- ... diferença de magnitude entre uma probabilidade de erro de 10^{-15} e outro de 10^{-4} não pode ser subestimada.
- e.g., ... considere uma taxa de erros de bits de 10^{-15} em uma linha de transmissão com capacidade de 9600 bps.
 - ... teremos 01 erro de bit a cada 3303 anos de operação contínua.
- ... alguns valores para taxas de erros de bits como função do meio de transmissão, p.ex., “hardware” = 10^{-15} ; fibra ótica = 10^{-19} ; cabo coaxial = 10^{-6} e par trançado = 10^{-4} a 10^{-5}

... 3.1 - Introdução

- Obs.: ... não se pode desprezar a diferença em magnitude.
- e.g., ... diferença entre taxas 10^{-15} e 10^{-4} é enorme e se considerarmos a taxa de transmissão de 100.000.000 bps (100 Mbps)
 - 10^{-15} – 01 bit será corrompido a cada 115 dias;
 - 10^{-4} – 10.000 bits serão corrompidos a cada segundo
- ... dependendo das características da linha de transmissão, os dados podem ser reordenados, distorcidos, removidos, e, até ruídos podem ser inseridos como se fossem dados.

... 3.1 - Introdução

- “controle de erro” - ... tem por objetivo aumentar a confiabilidade de transmissão, preferivelmente para um nível de confiabilidade de operação “standalone” de um sist. computacional.
 - ... não se deve esperar dos métodos de controle de erro a captura de todos os possíveis erros que a princípio podem ocorrer !!
 - ... um controle de erro efetivo deve considerar as características de erro do canal a ser usado ... “questão frequentemente negligenciada”.
- Obs.: ... se a taxa de erro de um canal é \ll que a de um equip. periférico, a inclusão de controle de erro é a solução ?!
 - ... inclusão do controle de erro degrada desnecessariamente o desempenho e pode até diminuir a confiabilidade em vez de aumentá-la.

3.2 – Modelo de Erro

- “Discrete Memoryless Channel” - modelo formal para este tipo de canal é o canal discreto sem memória.
 - “canal discreto” - canal é dito discreto porque ele reconhece apenas um número finito de níveis de sinais distintos;
 - “canal sem memória” - canal é dito sem memória porque a probabilidade de um erro é suposta ser independente dos erros anteriores.
- “Canal Simétrico” - $p(\text{bit error } 0) = p(\text{bit error } 1)$

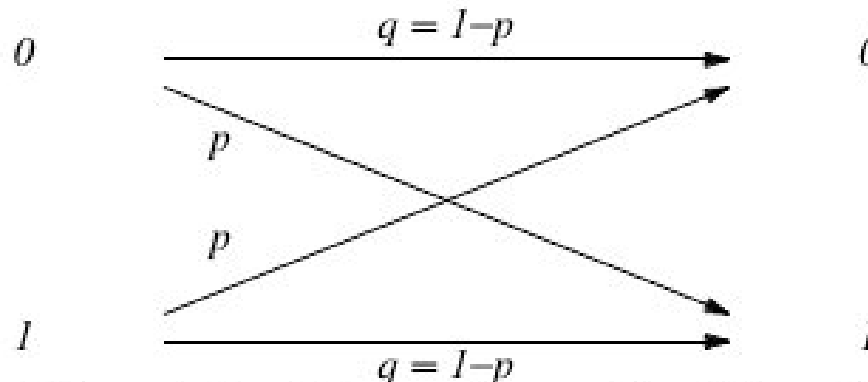


Figure 3.1 — Discrete Memoryless Channel

... 3.2 – Modelo de Erro

- Canal Simétrico Binário - ... estabelece que a probabilidade “Pr” de se ter ao menos “n” bits contíguos livres de erro $\text{EFI} \geq n$ é:
 - $\text{Pr}(\text{EFI} \geq n) = (1 - b)^n$ onde EFI “Error Free Interval”
 - ... onde “n” ≥ 0 e “b” = taxa média de erros de bits.
- ... probabilidade decresce linearmente a medida que o tamanho do intervalo livre de erros decresce (EFI);
- ... de maneira similar, probabilidade que a duração de uma rajada exceda “n” bits decresce linearmente com “n”.
- e.g., conside $b = 0.5$, então $(1-b)^n = 0.5^n$ o que implica que a razão entre 2 resultados consecutivos é linear.

... 3.2 – Modelo de Erro

- Para expressar “Pr” como uma função exponencial do tamanho do intervalo livre de erros ($EFI \geq n$), substitui-se a equação ...
 - $Pr(EFI \geq n) = e^{-b \cdot (n-1)}$
 - ... onde “n” ≥ 1 e “b” = taxa média de erros de bits.
- “acurácia da predição” - comparar os resultados da equação com valores empíricos ou experimentais (melhor forma de avaliar a acurácia da predição)
 - ... estudos mostram que a equação acima (exponencial) prediz intervalos livres de erros de modo melhor que a equação anterior (linear);
 - ... no entanto, a equação pode ainda ser melhorada com a adição de mais um fator que determina o efeito de agrupamento → Benoit Mandelbrot.

3.3 – Erros de Transmissão

- Como os principais erros de transmissão aparecem / se apresentam ?!
- **inserção / remoção de dados** – ... normalmente causado por perda temporária de sincronização entre transmissor e receptor;
- **remoção de dados** - ... também podem ser causados artificialmente por disciplinas inadequadas de controle de fluxo;
- **duplicação de dados** - ... podem ser gerados intencionalmente, p.ex., pelo transmissor ao implementar protocolo com retransmissão;
- **distorção de dados** - ... podem ser gerados por variações nas condições de operação do canal, inserindo distorções nos dados;
- **reordenação de dados** - ... podem potencialmente ser causados quando os dados percorrem diferentes rotas.

... 3.3 – Erros de Transmissão

- ... remoção, duplicação e reordenação são resolvidos com esquemas próprios de controle de fluxo.
- ... não obstante, em todos os casos, inserções e distorções podem ocorrer, não importando a razão que o gerou, assim, fazem-se necessários métodos para verificar a consistência dos dados.

3.4 – Redundância de Dados

- "Detecção de Erro" - somente funcionam se aumentarmos a redundância de dados de algum modo na mensagem.
- Além da detecção de erros de transmissão, o receptor pode também ser capaz de corrigir os erros segundo 02 maneiras:
 - "Forward Error Control" - redundância de dados se dá de tal modo que o receptor reconstroi a mensagem a partir da mensagem distorcida;
 - "Feedback Error Control" - redundância de dados se dá de tal modo que o receptor apenas detecta que a mensagem contém distorções;

... 3.4 – Redundância de Dados

- “Feedback Error Control - redundância de dados se dá de tal modo que o receptor é capaz de apenas detectar erros (se houver)
 - ... códigos de transmissão correspondentes são denominados “Error Detecting Codes” ou Códigos de Detecção de Erros.
 - ... aplicáveis em redes de computadores e Processamento Paralelo (PP) (TC – T... C...) !?!?
- ?!

... 3.4 – Redundância de Dados

- ECCs capazes de detectar e corrigir erros em primitivas (“Forward Error Control”) apresentam complexidade (computabilidade) bem maior que (“Feedback Error Control”)
- “Feedback Error Control” ... há duas possibilidades:
 - se o transmissor apresenta alta taxa de erro, a retransmissão pode ser requerida explicitamente, através de uma confirmação negativa;
 - ... neste caso o receptor simplesmente descarta a primitiva corrompida e espera por uma retransmissão da mensagem;
 - se o transmissor apresenta taxa de erro suficientemente baixa, a ausência de confirmação indica sucesso.

... 3.4 – Redundância de Dados

- “Controle de Erro” - tem o objetivo de diminuir a taxa de erro de um determinado canal de comunicação;
 - ... contudo, nem todos os erros podem ser detectados, e então sempre existe uma ‘Taxa de Erro Residual’ (RER – “Residual Error Rate”);
- Suponha-se que a probabilidade de erros de transmissão em msgs. seja “p”, e que o método de controle de erro identifique uma fração “f” deste conjunto de erros, então:

$$\text{RER} = p * (1 - f)$$

- ... esta equação implica que, por instância, a taxa residual de erros é da ordem de 10^{-9} (10^{-9}) ou menos.

3.5 – Tipos de Codificação

- São 02 os tipos básicos de codificação:
- “Block Codes”
 - todas as “code words” têm possivelmente o mesmo tamanho;
 - codificação para cada msg. de dado pode ser definida estaticamente.
- “Convolution Codes” - há uma relação entre a codificação da primitiva corrente e a codificação das primitivas anteriores ...
 - “code word” produzida depende da mensagem de dados e de um nro. prévio de mensagens codificadas;
 - codificador muda seu estado com cada msg. que é processada.
 - comprimento da “code word” é usualmente estático.

... 3.5 – Tipos de Codificação

- Palavras Códigos ou “Code Words” podem ser classificadas em: “Linear Codes”; “Cyclic Codes” e “Systematic Codes”
- “Linear Codes” - toda combinação linear de palavras códigos válidas produz uma outra palavra código válida (mod-2 sum).
- “Cyclic Codes” - cada deslocamento cíclico (cyclic shift) de uma palavra código válida produz uma palavra código válida (CRC).
- “Systematic Codes” - toda palavra código inclui bits da primitiva original, seguida ou precedida por um grupo de bits de “checks”.

... 3.5 – Tipos de Codificação

- Em todos os casos, as palavras códigos são maiores que as palavras de dados sobre as quais se baseiam;
- ... se o nro. de bits originais é “d” e o nro. de bits adicionais é “e”, a razão “ $d / (d + e)$ ” é chamada “code rate”.
- ... melhorar a “code rate” frequentemente significa aumentar a redundância e por consequência diminuir a “code rate”.
- e.g. ... reduzir taxa de erro de um canal por um fator de $5 * 10^2$ usando o “Forward Error Control”, pode exigir um código com um “code rate” ≤ 0.5 .

3.6 – Verificação por Paridade

- Se a probabilidade de erros de múltiplos bits por msg. é baixa, tudo que é necessário para o controle de erro em um canal simétrico binário é o código de verificação por paridade.
 - ... para toda mensagem que adicionarmos 1 único bit, teremos a soma módulo 2 naquela mensagem igual a 1 (um);
 - ... se um único bit, incluindo o bit de verificação sofre mudança, o cálculo da paridade no receptor indica erro e, mas não pode ser corrigido.

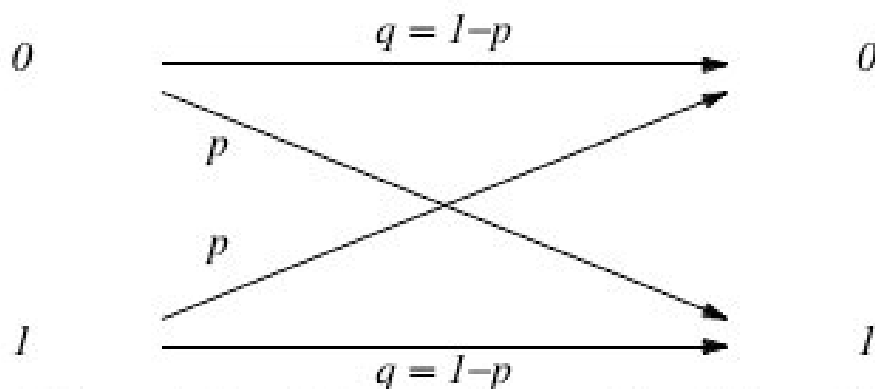


Figure 3.1 — Discrete Memoryless Channel

... 3.6 – Verificação por Paridade

- Se “ $q = 1 - p$ ”, então a probabilidade de transmissão livre de erro para msgs. de “ $n + 1$ ” bits, sendo 1 bit de partidade, é “ $q^{(n+1)}$ ”.
- ... probabilidade de erros de 1 bit em “ $n + 1$ ” bits transmitidos é a probabilidade binomial = “ $(n + 1) * p * q^n$ ”
- “Canal Simétrico” - p (bit error 0) = p (bit error 1)

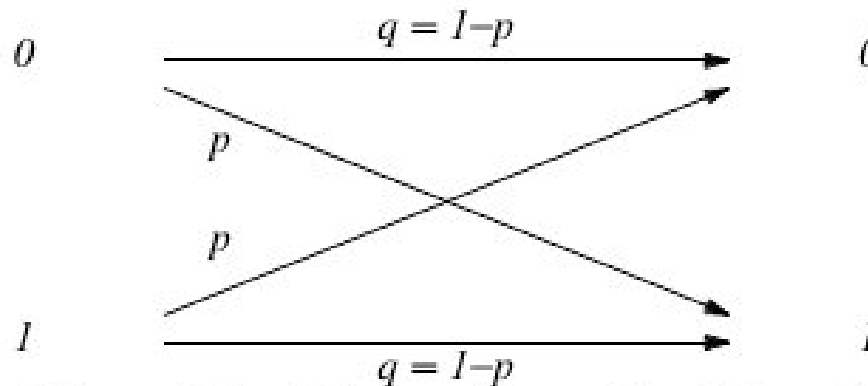


Figure 3.1 — Discrete Memoryless Channel

... 3.6 – Verificação por Paridade

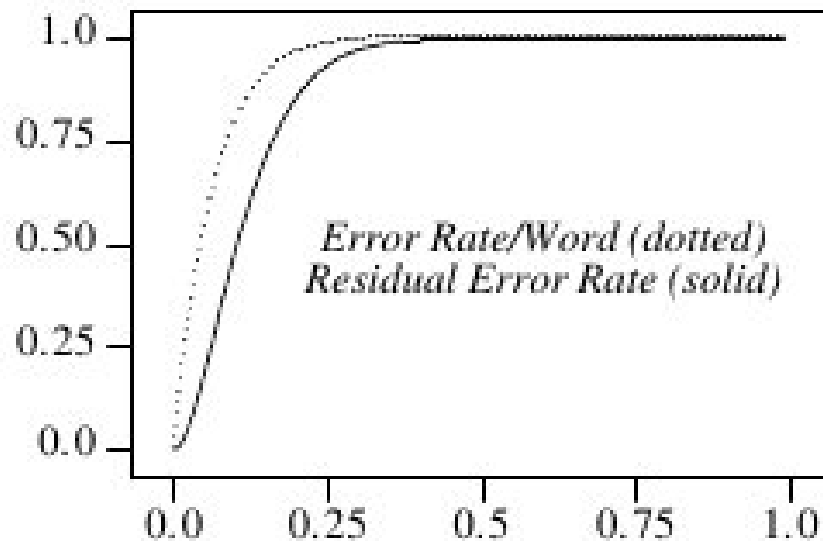
- ... probabilidade de erros de 1 bit em “n + 1” bits transmitidos é a probabilidade binomial, ou seja, ... $(n + 1) * p * q^n$
- ... sob tais circunstâncias, e.g., canal sem memória, a taxa de erro residual da verificação por paridade de 1 bit é

$$1 - q^{(n + 1)} - (n + 1) * p * q^n$$

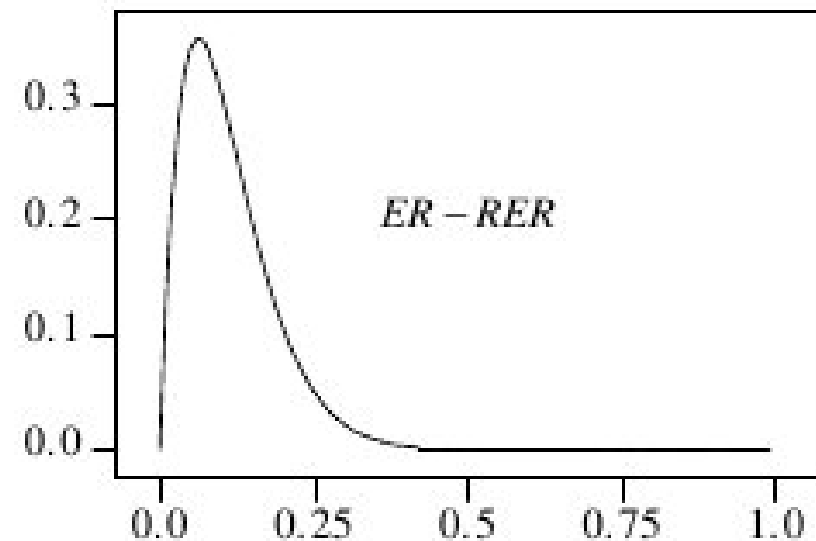
- e.g., ... para $n = 15$ e $p = 10^{-4}$ teremos taxa de erro residual da ordem de 10^{-6} por mensagem ou 10^{-7} por bit.

... 3.6 – Verificação por Paridade

- ... linha contínua mostra como a taxa de erro residual por “code word” aumenta como uma função da taxa “ p ” de erro de bits.



(a) — Error Rate/Bit (p)



(b) — Error Rate/Bit (p)

Figure 3.2 — Residual Error Rate of a 1-bit Parity Check, $n=15$

3.7 – Correção de Erro

- “Forward Error Control” usa somente um conjunto pequeno de combinações disponíveis de bits para codificar as msgs.
 - ... códigos são escolhidos de modo que se tenha relativamente um nro. grande de erros de bits para converter uma msg válida em outra.
- “code rate” - ... para um código de correção de erro é em geral menor que aquele para um código de detecção de erro.

... 3.7 – Correção de Erro

- “Forward Error Correction” - ... por apresentar “code rate” maiores que o “Feedback Error Control”, quando utilizá-la ?!
- Deve ser considerado somente quando as msgs. de controle do RX para o TX for um problema por razões tais como:
 - valores altos para o atraso de transmissão;
 - taxa de erros de bits alta;
 - ausência de canal de retorno (receptor → transmissor).

... 3.7 – Correção de Erro

- e.g., ... problema de comunicação de uma estação espacial (recptor) e o centro de controle (transmissor) na terra;
 - ... sinal de controle para liberar o obturador da câmera ou ajustar um curso, pode exigir vários minutos até alcançar a estação;
 - ... neste cenário, pode não haver tempo suficiente para repetir o sinal no caso de erro de transmissão (sinal pode chegar como pode se perder).
- e.g., taxa alta de erros de bits → neste caso, até mesmo a probabilidade de uma requisição para uma retransmissão ser recebida corretamente é inaceitavelmente baixa.
 - ... o que coloca em dúvida a própria requisição para retransmissão.

... 3.7 – Correção de Erro

- Até mesmo a Verificação de Paridade Simples por “code word” pode ser extendida de um código de detecção de um único erro para código de correção de um único erro.
 - e.g., sequência de 7 bits é extendida de 1 bit, o que torna o nro. de bits na sequência um nro. par ... “Longitudinal Redundancy Check” ou LRC.
 - 1000100 + 0 onde “0” é o bit de paridade.
 - ... na sequência inclui-se redundância para a série de “n” códigos, ou seja, “Vertical Redundancy Check” ou VRC

		<i>LRC</i>
<i>D</i>	= 1000100	0
<i>A</i>	= 1000001	0
<i>T</i>	= 1010100	1
<i>A</i>	= 1000001	0

	0010000	1 <i>VRC</i>

... 3.7 – Correção de Erro

- ... diferença entre 02 “code words” é definida como o nro. de bits nos quais as “code words” se diferem.
 - “Hamming Distance” - mínima diferença entre 02 “code words”
- Código com “Hamming Distance” de “n”, implica que qualquer combinação de até “n – 1” erros de bit pode ser detectada;
 - ... qualquer combinação até “ $(n - 1)/2$ ” de erros de bits por código pode ser corrigida se o receptor interpretar toda palavra de código não válida como a palavra de código válida mais próxima.
 - ... este método é formalmente chamado de “Maximum Likelihood Decoding” ou “Nearest Neighbor Decoding”.

... 3.7 – Correção de Erro

- Aumentando a Distância de Hamming, escolhendo “code words” mais longos, seremos capazes de aumentar a confiabilidade de um código tanto mais quanto maior a “code word”.
- Obs.: Redundância de um Código determina seu poder de detecção e correção de erros de transmissão.

3.8 – “Linear Block Code”

- ... redundância pode ser redefinida como o nro de bits sobre o mínimo exigido para codificar inequivocamente uma mensagem.
- Para codificar uma de “n” msgs. precisamos de “m” bits, sendo, que “m = inteiro superior o mais próximo (\log “n” na base “2”)
- ... pode-se proteger os “m” bits adicionando “c” bits de verificação e escolhendo “n” códigos de um total de $2^{(m+c)}$ códigos;
- ... de tal forma que cada combinação de 02 códigos válidos contemple a maior diferença em bits quanto possível.

3.8 – “Linear Block Code”

- ... pode-se proteger os “m” bits adicionando “c” bits de verificação e escolhendo “n” códigos de um total de $2^{(m+c)}$ códigos;
- ... de tal forma que cada combinação de 02 códigos válidos contemple a maior diferença em bits quanto possível.

Table 3.3 — Parity Protection

c	m	$m/(m+c)$
1	0	0.00
2	1	0.50
3	4	0.57
4	11	0.73
5	26	0.84
6	57	0.90
7	120	0.94
8	247	0.97

3.8 – “Linear Block Code”

- Para “m” bits de informação a ser transmitida, ou seja, para cada palavra de “ 2^m ” possíveis códigos de dados, precisamos de “m + c + 1” palavras para proteger a msg de erros de 1 único bit;
 - ... nro total de códigos é $(m + c + 1) * 2^m$
 - ... $2^{(m + c)}$ palavras possíveis (não incluso a paridade)
- $(m + c + 1) * 2^m = 2^{(m + c)} \rightarrow “m + c + 1 = 2^c”$ o que permite calcular o nro mínimo de bits de verificação dado o nro de bits de dados.
- e.g., ... para 8 bits de dados ($m = 8$), temos $c = 3,66$, ou 4 bits de paridade , não obstante a msg. pode ter até 11 bits.
 - “code rate” = $8 / (8 + 4) = 0.66$

... 3.8 – “Linear Block Code”

- Com uma boa aproximação, o nro de bits de dados que podem ser protegidos cresce exponencialmente com o nro. de bits de verificação, ou seja, “c” bits de verificação.

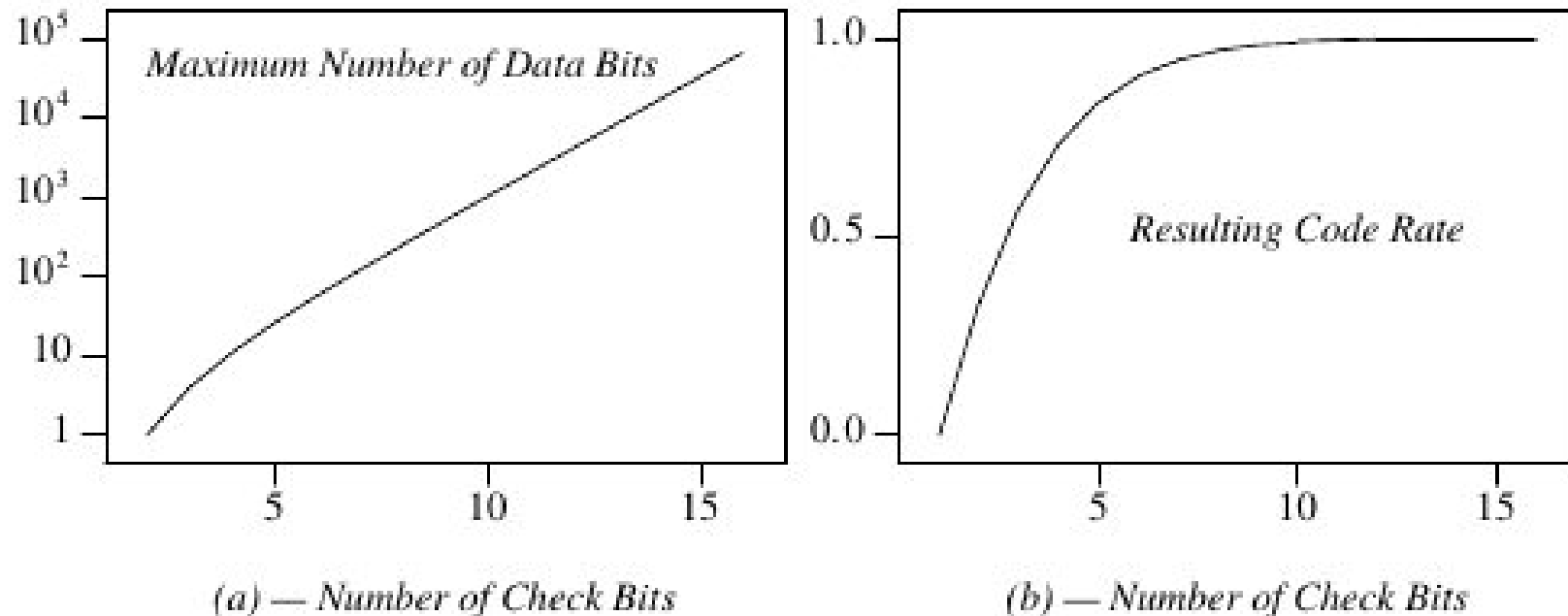


Figure 3.3 — Parity Protection

... 3.8 – “Linear Block Code”

- $(m + c + 1) * 2^m = 2^{(m + c)} \rightarrow “m + c + 1 = 2^c”$ o que permite calcular o nro mínimo de bits de verificação dado o nro de bits de dados.
- ... também podemos calcular o nro máximo de bits de dados para um dado nro de bits de verificação “c”

Table 3.3 — Parity Protection

c	m	m/(m+c)
1	0	0.00
2	1	0.50
3	4	0.57
4	11	0.73
5	26	0.84
6	57	0.90
7	120	0.94
8	247	0.97

... 3.8 – “Linear Block Code”

- Mesmo efeito da Tab. 3.3 (anterior) é ilustrado na Fig. 3.3 para até 16 bits de verificação.

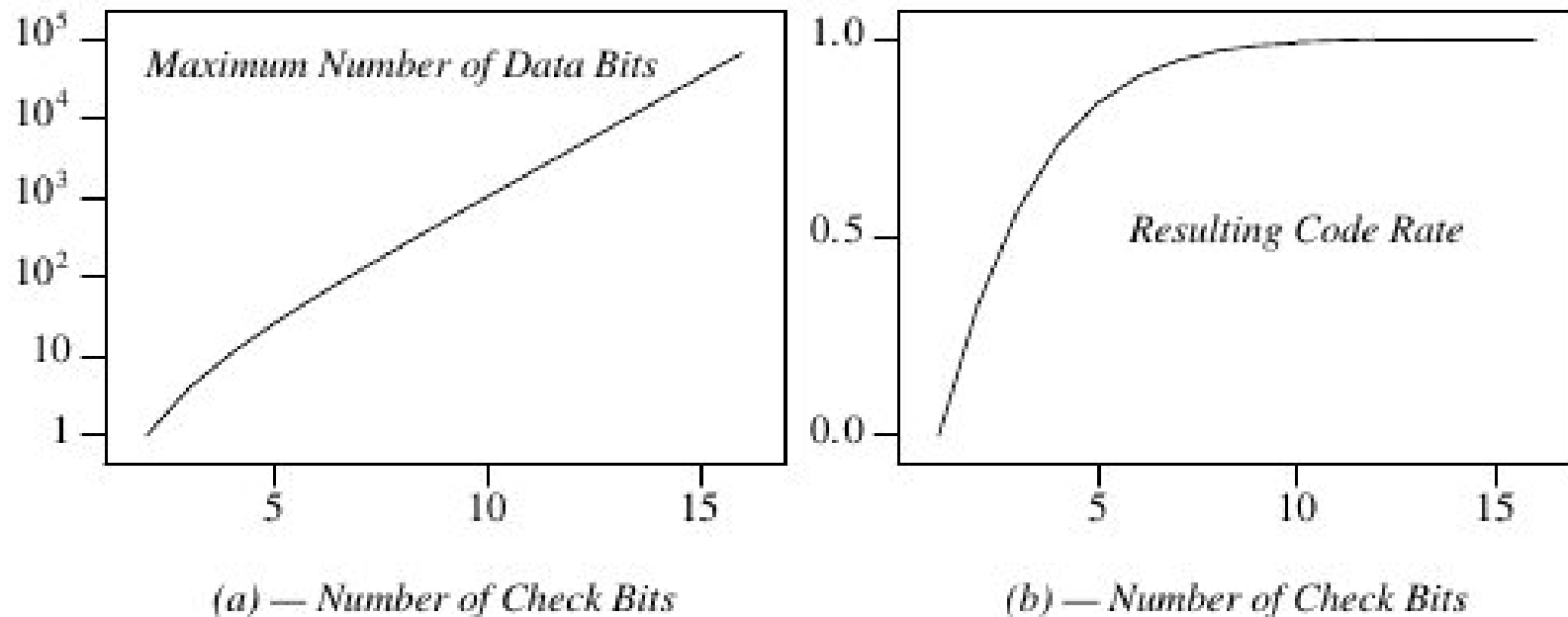


Figure 3.3 — Parity Protection

... 3.8 – “Linear Block Code”

- e.g. “hamming code” - ... em uma mensagem de “n” bits, inclui-se “c” bits de verificação => “n + c” bits na mensagem;
- ... para “c” bits de verificação, teremos “ $n + c = 2^c - 1$ ”, ou seja, nro máximo de bits na mensagem será “ $n = (2^c - 1) - c$ ”;
- ... e.g., para “n = 4” e “c = 3” (bits de paridade) => 7 bits na msg;
para “n = 11” e “c = 4” (bits de paridade) => 15 bits na msg.

Posição do bit		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
bits codificados		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15
bits de paridade	p1	X		X		X		X		X		X		X		X		X		X	
	p2		X	X			X	X			X	X			X	X			X	X	
	p4				X	X	X	X					X	X	X	X					X
	p8								X	X	X	X	X	X	X	X					
	p16																X	X	X	X	X

... 3.8 – “Linear Block Code”

- “Hamming Code” - ... bits na palavra de código são numerados de “1” a “m+c” sendo que “i-ésimo” bit de verificação é colocado na posição “ 2^i ” ... $1 \leq i \leq \log_2(m+c)$
- ... bits de verificação são colocados no palavra de código de tal maneira que a soma da posição dos bits que eles ocupam aponte para o bit errado para qualquer erro de 1 único bit (mas não para 2 bits).

Posição do bit		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
bits codificados		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15	
bits de paridade	p1	X		X		X		X		X		X		X		X		X		X		
	p2		X	X			X	X			X	X			X	X			X	X		...
	p4				X	X	X	X					X	X	X	X					X	
	p8								X	X	X	X	X	X	X	X						
	p16																X	X	X	X	X	

... 3.8 – “Linear Block Code”

- ... bits de verificação são colocados na palavra de código de tal maneira que a soma da posição dos bits que eles ocupam aponte para o bit errado para qualquer erro de 1 único bit;
- ... qdo um posição é escrita como a soma de potências de 2, p.ex., $(1 + 2 + 4)$, estas potências também apontam para os bits de verificação que cobrem o bit em questão ... neste caso bit de dados – 7.

Posição do bit		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
bits codificados		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15
bits de paridade	p1	X		X		X		X		X		X		X		X		X		X	
	p2		X	X			X	X			X	X			X	X			X	X	
	p4				X	X	X	X					X	X	X	X					X
	p8								X	X	X	X	X	X	X	X					
	p16																X	X	X	X	X

... 3.8 – “Linear Block Code” (LBC)

- “Representação Matricial” - método conveniente para “LBC”
 - e.g., ... considere um código com 3 bits de dados (D1, D2 e D3) e 03 bits de verificação (C4, C5 e C6).
 - ... seja $C4 = D1 + D2$; $C5 = D1 + D3$; e $C6 = D2 + D3$, então com algum rearranjo podemos representar as equações na forma de matricial.

$$\begin{bmatrix} C4 \\ C5 \\ C6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} D1 \\ D2 \\ D3 \end{bmatrix}$$

$$\begin{array}{ccccccc} D1 & + & D2 & + & & + & C4 & & = & 0 \\ D1 & + & & + & D3 & & + & C5 & & = & 0 \\ & & D2 & + & D3 & & & + & C6 & = & 0 \end{array}$$

... 3.8 – “Linear Block Code” (LBC)

- C^t é a matriz transposta de palavras de dados, escrita como um vetor de bits e que de acordo com a equação anterior.
 - Matriz de Dados + Verificação * C^t produz o vetor “zero”.
- Podemos representar a matriz na forma: $H * C^t = 0$
 - ... onde H é a Matriz de Paridade (Dados + Verificação).

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \cdot C^t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

... 3.8 – “Linear Block Code” (LBC)

- ... erros de transmissão podem ser formalizados pela adição de um vetor E, ou seja, $H * (C^t + E) = s$
 - ... onde S é chamado de Síndrome.
- Neste código, toda soma módulo 2 de palavras de código válidas produz uma outra palavra de código válida;
 - ... assim, se o vetor de erros E coincide com alguma palavra de código válida, a síndrome é zero e o erro não é detectável !!

3.9 – Cyclic Redundancy Checks

- “Cyclic Redundancy Check” - algoritmo de código cíclico mais utilizado em redes de computadores;
 - ISO especifica um algoritmo similar denominado FCS (Frame Check Sequence) utilizado em seus protocolos IEEE802.
- CRC contempla o produto de bits de verificação de uma primitiva, cuja amostragem é definida segundo um critério.
 - ... bits escolhidos fazem parte do fator (polinômio gerador) e, portanto, é transmitido o produto, ou seja, primitiva * fator;
 - .. no destino faz-se a divisão do que é recebido pelo fator, se resto é zero significa ausência de erro ou erro não detectável;
 - ... método de divisão é específico e o fator (polinômio gerador) usado determinam o leque de erros de transmissão que podem ser detectados.

... 3.9 – Cyclic Redundancy Checks

- Seja uma primitiva - sequência de “n” bits que pode ser representada por um polinômio de grau “n-1”;
 - Somatório $b_i * x^i$ com $i = [0 .. (n-1)]$
 - b_i é o coeficiente do bit na posição “i”
 - x^i indica o literal do bit na posição “i”
- e.g., ... considere a sequência de bits “10011”.
 - Como esta sequência pode ser representada por polinômio ?

$$“1*x^4 + 0*x^3 + 0*x^2 + 1*x + 1” \text{ ou } “x^4 + x + 1”$$

... 3.9 – Cyclic Redundancy Checks

- Sejam as operações binárias (ou-exclusivo):
 - $0 + 0 = 0 - 0 = 0$
 - $0 + 1 = 0 - 1 = 1$
 - $1 + 0 = 1 - 0 = 1$
 - $1 + 1 = 1 - 1 = 0$
- ... não há “carry bit” na adição e nem “borrow bit” na subtração, ou seja, para todo “i” ... $x_i + x_i = 0$
- para multiplicar 02 códigos de dados, basta multiplicar os polinômios correspondentes aos códigos de dados.

... 3.9 – Cyclic Redundancy Checks

- Dado codificado em 4 bits (p.ex., $x^2 + 1$) é multiplicado por “ $x + 1$ ”
- Código gerado é o de paridade cuja “code rate” de $\frac{3}{4}$.
- É também um código cíclico, mas não é simétrico.

Table 3.4 — A Cyclic Code

Data Word	Polynomial	Multiplied By	Produces	Code Word
0 0 0	0	$x + 1$	0	0 0 0 0
0 0 1	1	$x + 1$	$x + 1$	0 0 1 1
0 1 0	x	$x + 1$	$x^2 + x$	0 1 1 0
0 1 1	$x + 1$	$x + 1$	$x^2 + 1$	0 1 0 1
1 0 0	x^2	$x + 1$	$x^3 + x^2$	1 1 0 0
1 0 1	$x^2 + 1$	$x + 1$	$x^3 + x^2 + x + 1$	1 1 1 1
1 1 0	$x^2 + x$	$x + 1$	$x^3 + x$	1 0 1 0
1 1 1	$x^2 + x + 1$	$x + 1$	$x^3 + 1$	1 0 0 1

... 3.9 – Cyclic Redundancy Checks

- e.g., Calcule o polinômio cuja primitiva é “1 0 0 1 1” e o fator (polinômio gerador) é “1 1 0 0”

$$\text{“1 0 0 1 1”} \rightarrow x^4 + x + 1$$

$$\text{“1 1 0 0”} \rightarrow x^3 + x^2$$

$$(x^4 + x + 1) * (x^3 + x^2) = x^7 + x^6 + x^4 + x^3 + x^3 + x^2$$

$$\dots x^7 + x^6 + x^4 + \underline{x^3} + \underline{x^3} + x^2 \rightarrow x^3 + x^3 \text{ é } 0 \dots$$

$$(x^4 + x + 1) * (x^3 + x^2) = x^7 + x^6 + x^4 + x^2$$

... 3.9 – Cyclic Redundancy Checks

- “polinômios” - podem ser objetos de todas as oper. aritméticas.
 - ... como a primitiva é multiplicada por um fator (polinômio gerador) para criar o código a ser transmitido, então aplica-se no destino o processo inverso, e.g., usar o fator (polinômio gerador) como divisor.
-
- e.g., ... seja o polinômio “ $x^7 + x^6 + x^4 + x^3 + x^2$ ” dividido pelo fator “ $x^5 + x^2 + 1$ ” resultando “ $x^2 + x$ ” e resto “ x ”
 - ... para fazer o polinômio divisível pelo fator basta subtrair o resto dele, contudo, o receptor será capaz de somente detectar o erro.
 - “conclusão” - ... mais indicado é usar o resto como um “checksum”.

... 3.9 – Cyclic Redundancy Checks

- “fator” - utilizado para gerar o “checksum”, também é denominado Polinômio Gerador (Generator Polynomial) do código.
 - ... multiplica-se o polinômio da primitiva por um termo igual à parcela de mais alto grau do polinômio gerador de modo que a primitiva seja deslocada para a esquerda tantos bits quantos do “checksum”;
 - ... na sequência, divide-se o polinômio da primitiva pelo polinômio gerador ou fator obtendo-se um quociente e o resto.
- Como o CRC é um código linear, todo padrão de erro E deve ser igual a alguma palavra código T;
 - ... para um código conhecido esta propriedade pode ser usada para calcular a taxa residual de erro.

... 3.9 – Cyclic Redundancy Checks

- “Controle de Erro” - tem o objetivo de diminuir a taxa de erro de um determinado canal de comunicação;
 - ... contudo, nem todos os erros podem ser detectados, e então sempre existe uma ‘Taxa de Erro Residual’ (RER – “Residual Error Rate”);
- Suponha-se que a probabilidade de erros de transmissão em uma msg. seja “p”, e que o método de controle de erro identifique uma fração “f” deste conjunto de erros, então:
 - $RER = p * (1 - f)$... esta equação implica que, por instância, a taxa residual de erros é da ordem de 10^{-9} (10^{-9}) ou menos.

... 3.9 – Cyclic Redundancy Checks

- Seja P um polinômio primitiva e G um polinômio gerador de grau “ r ”, então o resto da divisão é polinômio R com grau “ $r-1$ ”, onde:

$$R = P * x^r / G$$

- Palavra Código “ $T = P * x^r - R$ ”
- Um erro de transmissão adiciona um polinômio E ao código ... e o receptor descobre o erro por:

$$(T + E) / G = T/G + E/G = E/G$$

... 3.9 – Cyclic Redundancy Checks

- Um erro de comunicação é indetectável se $E/G = 0$
 - ... se E é diferente de zero e de grau menor que G , a divisão sempre tem resto, ou seja, todas as rajadas menores ou iguais a “ r ” são detectáveis.
 - ... posição dentro da primitiva T “code word” onde ocorre a rajada de erro é irrelevante para detecção do erro;
- Padrão de Erro “ E ” não se transforma em um múltiplo de G pela simples multiplicação de um fator x^i (obviamente para $x^i \neq G$).
 - ... como a ocorrência do erro é aleatória, então para um código de comprimento “ $n+r$ ” bits é possível calcular a probabilidade de ocorrência de erro (independente da posição).

... 3.9 – Cyclic Redundancy Checks

- Existem “ $2^n + r$ ” possibilidades de erros, sendo que o nro. de múltiplos inteiros de um polinômio gerador de grau “ r ” em uma palavra de código é “ 2^n ”
- ... cada múltiplo pode ser considerado como uma soma finita de “ n ” fatores, no qual cada fator é obtido por um deslocamento à esquerda do polinômio gerador.
- $2^n / 2^{(n+r)} = 1 / 2^r$
- e.g., considere $r = 16 \rightarrow 10^{-5}$ de todos os erros.

... 3.9 – Cyclic Redundancy Checks

- Projetar polinômios geradores para detectar a maior classe possível de erros de comunicação não é uma “tarefa fácil”.
 - ... um bom polinômio gerador tem pelo menos um fator $(x + 1)$, além disso, em sistemas distribuídos isto deve ser compartilhado com todos os pares.
- CRC-12 é um polinômio largamente utilizado, para gerar soma verificação - “checksum” de grau 12

$$x^{12} + x^{11} + x^3 + x^2 + 1$$

- Obs.: ... grau é 12, então este código pode detectar rajadas de erros de até 12 bits, não somente rajadas de 12 bits.

... 3.9 – Cyclic Redundancy Checks

- A CCITT (hoje ITU-T) recomenda polinômio gerador de grau 16.

$$x^{16} + x^{12} + x^5 + 1$$

- ... grau é 16, então este código pode detectar rajadas de erros de até 16 bits, não somente rajadas de 16 bits.
- Em aritmética binária, este código pode ser escrito como:

$$(x+1) * (x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1)$$

- ... pode-se ver que qualquer polinômio multiplicado pelo fator $(x+1)$ tem um número par de parcelas (ie, bits não zero).
- ... verificar o resultado (nro par de parcelas) ... !!!

... 3.9 – Cyclic Redundancy Checks

- Isto significa que todo “E” com um número ímpar de parcelas, produzido por um número ímpar de erros de bits é detectável;
- CRC-CCITT detecta
 - 100% em caso de 2 bits de erro;
 - 99,997% em caso de rajadas de 17 bits;
 - 99,998% em rajadas maiores que 17 bits.
- O polinômio utilizado pelo protocolo BSC (Binary Synchronous Communication) da IBM é muito próximo deste polinômio:

$$x^{16} + x^{15} + x^2 + 1$$

... 3.9 – Cyclic Redundancy Checks

- Comitê IEEE 802 padronizou um CRC-32 bits:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

- Codificação e Decodificação do “checksum” CRC exigem tempo e podem degradar o desempenho do autômato;
- ... implementação é tipicamente feita em “hardware”, por motivos óbvios de necessidade de esforço computacional.
- Observe-se que o desempenho do algoritmos tem uma linearidade com o grau do polinômio gerador.