CÓDIGOS CONCATENADOS CORRETORES DE ERROS QUÂNTICOS

Antonio Carlos Aido de Almeida e Reginaldo Palazzo Jr.

Resumo - Neste artigo, fazemos uma revisão dos códigos corretores de erros quânticos (CCEQs) construídos a partir da concatenação de um código *phase flip* com um código *bit flip*. Em particular, dois exemplos são revistos em detalhes: o código de bloco quântico (CBQ) de taxa 1/9 proposto por Shor, e o código convolucional quântico (CCQ) de taxa 1/4 proposto por Almeida e Palazzo. O objetivo deste artigo é introduzir o estudo de CCEQs à comunidade de engenheiros de telecomunicações.

Palavras-chave: Códigos Corretores de Erros Quânticos, Códigos Estabilizadores, Códigos Concatenados, Códigos de Bloco, Códigos Convolucionais.

Abstract - In this article, we make a review of quantum error-correcting codes (QECCs) constructed from the concatenation of a phase flip code with a bit flip code. In particular, two examples are reviewed in details: the rate-1/9 quantum block code (QBC) proposed by Shor, and the rate-1/4 quantum convolutional code (QCC) proposed by Almeida and Palazzo. The purpose of this article is to introduce the study of QECCs to the community of telecommunication engineers.

Keywords: Quantum Error-Correcting Codes, Stabilizer Codes, Concatenated Codes, Block Codes, Convolutional Codes.

1. INTRODUÇÃO

Códigos corretores de erros quânticos (CCEQs) têm sido desenvolvidos para proteger a informação quântica dos efeitos de erros de descoerência (veja [1] para uma revisão). O surgimento de CCEQs cada vez mais eficientes tem elevado a confiabilidade de armazenamento e transmissão de informação quântica e permitido a realização de computações quânticas com um número cada vez maior de qubits.

Em analogia com a teoria clássica, duas grandes classes de CCEQs têm sido desenvolvidas: a classe dos códigos de bloco quânticos (CBQs) e a classe dos códigos convolucionais quânticos (CCQs). Dentro de cada uma destas classes, os primeiros, e também os mais simples, CCEQs a serem estudados foram os CCEQs estabilizadores concatenados. Estes CCEQs são construídos a partir da concatenação de um código phase flip com um código bit flip, ambos gerados a partir de códigos corretores de erros clássicos (CCECs) conhecidos.

Neste artigo, faremos uma revisão dos dois primeiros CCEQs estabilizadores concatenados a serem estudados: o CBQ de Shor de taxa 1/9, [2], gerado a partir do código de bloco clássico (CBC) de repetição de taxa 1/3; e o CCQ de taxa 1/4 e três memórias, recentemente proposto em [3], gerado a partir de um código convolucional clássico (CCC) de taxa 1/2 e duas memórias. Ambos são capazes de corrigir um erro quântico arbitrário sobre qualquer qubit da palavra-código.

O objetivo desta revisão é introduzir, através de uma linguagem mais acessível, o estudo de CCEQs à comunidade de engenheiros de telecomunicações e despertar, dentro desta comunidade, o interesse por sublclasses mais gerais de CCEQs.

Este artigo está organizado da seguinte forma: na seção 2, revisamos alguns conceitos fundamentais da informação quântica, da codificação quântica e do formalismo estabilizador; nas seções 3 e 4, construímos o código de Shor e o CCQ [(4, 1, 3)]; finalmente, na seção 5, apontamos futuras linhas de pesquisa dentro da subclasse dos CCEQs estabilizadores concatenados e descrevemos subclasses mais gerais de CCEQs. Para um melhor acompanhamento da nomenclatura utilizada ao longo deste texto, apresentamos uma lista de acrônimos na seção 6.

2. CONCEITOS FUNDAMENTAIS

2.1 INFORMAÇÃO QUÂNTICA

A unidade fundamental da informação clássica é o bit, uma variável aleatória que pode assumir dois valores: 0 ou 1. A unidade de informação quântica correspondente é o bit quântico ou qubit. O qubit é um vetor em um espaço vetorial complexo bidimensional com produto interno, ou seja, um vetor no espaço de Hilbert. De acordo com a notação introduzida por Dirac, podemos denotar os elementos de uma base ortonormal deste espaço por $|0\rangle$ e $|1\rangle$[1]. Estes vetores são representados por:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{e} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{1}$$

Esta base é chamada base computacional. Desta forma, o estado de um qubit arbitrário normalizado pode ser escrito

Este trabalho foi financiado pela FAPESP (Números dos Processos: 02/07473-7 e 04/10979-5). Os autores Antonio Carlos Aido de Almeida e Reginaldo Palazzo Jr. (E-mails: aido@dt.fee.unicamp.br, palazzo@dt.fee.unicamp.br) são pesquisadores do Departamento de Telemática da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas (UNICAMP).

[1]Para o leitor não familiarizado com os postulados e com a notação da mecânica quântica, sugerimos a consulta das referências [4, 5, 7].

como:

$$|\psi\rangle = a|0\rangle + b|1\rangle, \tag{2}$$

no qual $|a|^2 + |b|^2 = 1$, com $a, b \in \mathbb{C}$ (o corpo dos números complexos). A base conjugada é definida como $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$.

Muitos sistemas físicos diferentes podem ser usados para representar um qubit, como por exemplo os dois estados (fundamental e excitado) de um elétron orbitando um átomo; as duas diferentes polarizações de um fóton; e o alinhamento de um spin nuclear em relação a um campo magnético uniforme.

O estado quântico de n qubits pode ser expresso como um vetor em um espaço 2^n-dimensional (ou seja, o produto tensorial de n espaços bidimensionais). Podemos escolher como uma base ortonormal para este espaço os estados nos quais cada qubit tem um valor definido, $|0\rangle$ ou $|1\rangle$. Desta forma, um vetor arbitrário normalizado pode ser escrito nesta base como:

$$|\psi\rangle = \sum_{x=0}^{2^n-1} a_x|x\rangle, \tag{3}$$

no qual associamos a cada seqüência de 0s e 1s, o número que ela representa na notação decimal, um valor entre 0 e 2^{n-1}. Note que o conjunto de vetores $|x\rangle$, com x inteiro e $0 \leq x \leq 2^n - 1$, forma uma base canônica para o espaço de Hilbert. Os coeficientes a_x são números complexos satisfazendo a relação $\sum_x |a_x|^2 = 1$.

Com múltiplos qubits, existem estados puros que podem não ser escritos como o produto tensorial de estados de um qubit. Por exemplo, o estado de dois qubits

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{4}$$

não pode ser decomposto na forma expressa em (3). Diz-se então que tal estado encontra-se *entrelaçado* ou *emaranhado*. Como veremos mais adiante, os estados emaranhados desempenham uma função crucial na correção de erros quânticos. Em particular, o estado (4) é conhecido como estado de Bell ou par EPR (de Einstein-Podolsky-Rosen).

O resultado de uma medição do estado (2) é *não* determinístico − a probabilidade de obtermos o resultado $|0\rangle$ é $|a|^2$ e a probabilidade de obtermos $|1\rangle$ é $|b|^2$. A normalização assegura que a probabilidade de obter algum resultado seja exatamente 1. Esta medição implementa um de dois operadores de projeção, as projeções sobre a base $\{|0\rangle,\ |1\rangle\}$. Esta não é a única medição que podemos fazer sobre um qubit. Na verdade, podemos projetar sobre qualquer base do espaço de Hilbert de um qubit.

Se tivermos múltiplos qubits, podemos medir um número de diferentes qubits independentemente, ou podemos medir alguma propriedade comum dos qubits, o que corresponde a fazer a projeção sobre alguma base emaranhada do sistema. Se medirmos todos os n qubits do estado (3) através da projeção de cada um deles sobre a base $\{|0\rangle,\ |1\rangle\}$, a probabilidade de obter o resultado $|x\rangle$ é $|a_x|^2$. Em particular, o par EPR, (4), tem uma propriedade importante: a medição de um qubit sempre fornece o mesmo resultado da medição do outro qubit. Isto é, os resultados das medidas estão *correlacionados*.

Para a teoria dos CCEQs, será interessante estudarmos o comportamento da ação das matrizes de Pauli (ou operadores de Pauli) sobre um qubit. Na base $\{|0\rangle,\ |1\rangle\}$, estas matrizes são escritas como:

$$\sigma_x \equiv X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \sigma_z \equiv Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$
$$\text{e} \quad \sigma_y \equiv iY = iXZ = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}. \tag{5}$$

Além das matrizes de Pauli e da matriz identidade I, um outro operador importante para o sistema de um qubit é o operador de Hadamard, que promove a mudança da base computacional para a base conjugada. Ou seja, o operador de Hadamard é uma rotação de π radianos em torno do eixo $(x + z)/\sqrt{2}$. Na base $\{|0\rangle,\ |1\rangle\}$, este operador é escrito como:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{6}$$

Para um sistema com n qubits, σ_i^j denota o produto tensorial da matriz σ_i atuando sobre o qubit j com as matrizes I atuando sobre todos os outros qubits. O grupo de Pauli sobre n qubits, denotado por $\mathcal{G}_n$, é gerado por σ_i^j, para $i = x,\ y,\ z$ e $j = 1,\ \ldots,\ n$. Poderemos também nos referir ao grupo de Pauli como sendo o menor subgrupo real gerado por σ_x e σ_z.

A medição realizada em um qubit corresponde à medição do autovalor de σ_z. Os correspondentes operadores de projeção são $(I \pm \sigma_z)/2$. Para uma partícula de spin-1/2, esta medição é realizada através da medição do spin da partícula ao longo do eixo z. Também poderíamos fazer uma medição ao longo do eixo x ou y, o que corresponde à medição do autovalor de σ_x ou σ_y. Neste caso, os operadores de projeção são $(I \pm \sigma_x)/2$ e $(I \pm \sigma_y)/2$, respectivamente.

Uma propriedade importante dos estados quânticos tem profundas implicações para a correção de erros quânticos: o teorema da não clonagem, que afirma que é impossível fazer uma cópia perfeita de um estado quântico arbitrário desconhecido, [6].

2.2 CODIFICAÇÃO QUÂNTICA

O processamento de informação quântica é freqüentemente descrito como uma série de operações unitárias e de medições em algum sistema físico. Imperfeições nestas operações e interações com o meio ambiente circundante (descoerência) são inevitáveis em qualquer processamento de informação quântica.

Portanto, para que um computador quântico funcione na prática, temos um grande obstáculo a superar: proteger a informação quântica de erros. Sem esta proteção, um computador quântico certamente irá falhar. Qualquer estratégia efetiva para impedir que erros ocorram em um computador quântico deve proteger contra pequenos erros unitários em um circuito quântico, bem como contra erros de descoerência.

Contra os erros de descoerência foi desenvolvida a teoria dos CCEQs. Quanto aos erros operacionais, intrínsecos às portas lógicas, foi desenvolvida a teoria quântica de

tolerância a falhas, que não será abordada aqui (ver, por exemplo, a seção 10.6 de [7]).

Para proteger os estados quânticos dos efeitos do ruído, alguns desafios devem ser superados, a saber:

1. **Erros de fase**. Além dos erros bit flip:

$$|0\rangle \rightarrow |1\rangle, \quad |1\rangle \rightarrow |0\rangle, \tag{7}$$

podem existir erros phase flip:

$$|0\rangle \rightarrow |0\rangle, \quad |1\rangle \rightarrow -|1\rangle. \tag{8}$$

Um erro phase flip é grave, porque faz o estado $(|0\rangle + |1\rangle)/\sqrt{2}$ transformar-se no estado ortogonal $(|0\rangle - |1\rangle)/\sqrt{2}$. A codificação clássica não oferece proteção contra erros desta natureza.

2. **Erros pequenos**. Como observado anteriormente, a informação quântica é contínua. Se pretendemos que um qubit esteja no estado $a|0\rangle + b|1\rangle$, um erro pode mudar a e b por uma quantidade da ordem de ε, e estes pequenos erros podem se acumular ao longo do tempo. Os esquemas clássicos são projetados para corrigir apenas erros (bit flip) grandes.

3. **Medição causa destruição do estado original**. Na correção de erros clássicos observamos a saída do canal e decidimos qual procedimento de decodificação devemos adotar. As observações da mecânica quântica em geral destroem o estado quântico sob observação e tornam impossível a recuperação do estado original.

4. **Não-Clonagem**. Com a codificação clássica protegemos a informação fazendo cópias extras dela. Mas sabemos que a informação quântica não pode ser copiada com perfeita fidelidade.

Felizmente, como veremos mais adiante, nenhum destes problemas é capaz de impedir a construção de CCEQs.

Um CCEQ pode ser visto como um mapeamento de k qubits (um espaço de Hilbert com 2^k dimensões) em n qubits (um espaço de Hilbert com 2^n dimensões), onde $n > k$. Os k qubits são os qubits lógicos (ou qubits de informação) que desejamos proteger de erros. Os n qubits são os qubits físicos resultantes da codificação. Os $n - k$ qubits adicionais permitem-nos armazenar os k qubits lógicos de uma forma redundante tal que os n qubits físicos não sejam facilmente alterados.

O processo de construção de CCEQs concatenados exige a definição de dois canais quânticos, a saber:

- **O canal bit flip**. Suponha que desejamos enviar qubits através de um canal que troca os estados-base de um qubit de $|0\rangle$ para $|1\rangle$ e vice-versa com probabilidade p e preserva o qubit de erros com probabilidade $1 - p$. Ou seja, com probabilidade p o estado $|\psi\rangle$ é levado ao estado $X|\psi\rangle$, sendo X o operador bit flip. Assim, se $|\psi\rangle = a|0\rangle + b|1\rangle$, temos $X|\psi\rangle = a|1\rangle + b|0\rangle$. Este canal é chamado de canal bit flip e é equivalente aos canais binários clássicos sem memória e com erros ocorrendo aleatoriamente. Portanto, para proteger os qubits dos efeitos do ruído deste canal, é necessário construir um código corretor de erros bit flip.

- **O canal phase flip**. Suponha que desejamos enviar qubits através de um canal que preserva um qubit com probabilidade $1 - p$ e troca a fase relativa dos estados $|0\rangle$ e $|1\rangle$ do qubit com probabilidade p. Mais precisamente, o operador phase flip Z é aplicado ao qubit com probabilidade p e, portanto, quando o estado $a|0\rangle + b|1\rangle$ é transmitido, teremos à saida deste canal o estado $a|0\rangle - b|1\rangle$. Este canal é chamado de canal phase flip. Assim, para proteger os qubits dos efeitos do ruído deste canal, é necessário construir um código corretor de erros phase flip. Não existe equivalente clássico para o canal phase flip, pois canais clássicos não têm nenhuma propriedade equivalente à fase. No entanto, existe um meio fácil de tratar o canal phase flip como um canal bit flip. Suponha que ao invés de trabalharmos com a base computacional $\{|0\rangle,\ |1\rangle\}$, trabalhemos com a base conjugada $\{|+\rangle,\ |-\rangle\}$ para o qubit. Com respeito a esta base, o operador Z leva o estado $|+\rangle$ para o estado $|-\rangle$ e vice-versa, isto é, este operador atua como se fosse um operador bit flip com respeito aos símbolos $+$ e $-$.

2.3 O FORMALISMO ESTABILIZADOR

A idéia básica do formalismo estabilizador é que muitos estados quânticos podem ser descritos mais facilmente pelos operadores que os estabilizam do que pelos próprios estados quânticos. Por exemplo, o par EPR, (4), é estabilizado pelos operadores X_1X_2 (ou seja, o operador X aplicado sobre o primeiro e o segundo qubits) e Z_1Z_2 (ou seja, o operador Z aplicado sobre o primeiro e o segundo qubits).

Muitos códigos quânticos, incluindo os deste artigo, podem ser descritos de forma muito mais compacta usando estabilizadores do que a descrição por vetores de estado. Isto é possível devido ao uso inteligente da teoria de grupos através do formalismo estabilizador. Dois grupos de operadores são usados para descrever o subespaço do código quântico [8]:

1. **O Grupo Estabilizador:**

   - O grupo estabilizador S é um subgrupo abeliano do grupo multiplicativo de Pauli $\mathcal{G}_n$. A menos de fatores multiplicativos ± 1 e $\pm i$, $\mathcal{G}_n$ é escrito como $\mathcal{G}_n = \{I,\ X,\ Y,\ Z\}^{\otimes n}$.
   - O subespaço do código $\mathcal{C}$ é o maior subespaço de $\mathcal{H}^{\otimes n}$ estabilizado por S:

$$|\psi\rangle \in \mathcal{C} \Longleftrightarrow S|\psi\rangle = |\psi\rangle. \tag{9}$$

   - Equivalentemente, se os M_i's são $n - k$ geradores independentes de S, então:

$$|\psi\rangle \in \mathcal{C} \Longleftrightarrow \forall\ i,\ M_i|\psi\rangle = |\psi\rangle. \tag{10}$$

   Estas equações, chamadas de síndromes, definem o subespaço do código.

2. **O Grupo de Pauli Lógico:** Os operadores lógicos deixam o subespaço do código $\mathcal{C}$ globalmente invariante, mas possuem uma ação não trivial sobre este espaço. É

possível exigir que tais operadores reproduzam exatamente as relações de comutação do grupo de Pauli para os qubits lógicos. Isto é matematicamente expresso por:

$$\overline{X}_i,\ \overline{Z}_i\ \in\ N(S)/S, \tag{11}$$

$$\{\overline{X}_i,\ \overline{Z}_i\} = 0, \tag{12}$$

$$\forall\ i \neq j,\ \ [\overline{X}_i,\ \overline{X}_j] = [\overline{Z}_i,\ \overline{Z}_j] = [\overline{X}_i,\ \overline{Z}_j] = 0. \tag{13}$$

Em (11), $N(S)$ é o normalizador de S, em (12) $\{.,.\}$ denota anticomutador e em (13) $[.,.]$ denota comutador[2].

A aplicação do formalismo estabilizador à teoria dos CCEQs ficará clara com a apresentação do código de Shor e do CCQ [(4, 1, 3)].

3. O CÓDIGO DE SHOR

Considere o CBC de repetição de três bits. A matriz geradora deste código é escrita como:

$$\mathbf{G} = \left[\begin{array}{ccc} 1 & 1 & 1 \end{array}\right]. \tag{14}$$

Este CBC pode ser usado na construção de um CBQ para o canal bit flip com a seguinte operação de codificação:

$$|u\rangle \mapsto |u,\ u,\ u\rangle, \tag{15}$$

na qual $u \in \{0, 1\}$.

Através da transformada de Hadamard, é possível obter também a operação de codificação do correspondente CBQ para o canal phase flip:

$$|u\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle), \tag{16}$$

ou, mais compactamente,

$$|u\rangle \mapsto \frac{1}{2\sqrt{2}} \sum_{p,\ q,\ r=(0,\ 0,\ 0)}^{(1,\ 1,\ 1)} (-1)^{(p+q+r)u} |p,\ q,\ r\rangle. \tag{17}$$

Os CBQs gerados pelas operações (15) e (16) são capazes de corrigir, respectivamente, um erro X e um erro Z, pois o CBC associado tem distância $d_c = 3$. Para determinarmos os geradores destes CBQs, devemos encontrar uma matriz de verificação de paridade para a matriz geradora (14), como por exemplo:

$$\mathbf{H} = \left[\begin{array}{ccc} 1 & 1 & 0 \\ 0 & 1 & 1 \end{array}\right]. \tag{18}$$

As linhas da matriz (18) são usadas para escrever os geradores dos CBQs bit flip e phase flip. No caso do CBQ bit flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Zs, ou seja, temos os geradores Z_1Z_2 (primeira linha de $\mathbf{H}$) e Z_2Z_3 (segunda linha de $\mathbf{H}$). Já no caso do CBQ phase flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Xs, ou seja, temos os geradores X_1X_2 (primeira linha de $\mathbf{H}$) e X_2X_3 (segunda linha de $\mathbf{H}$).

Analogamente, a linha da matriz (14) é usada para escrever os operadores lógicos sobre os qubits de informação dos CBQs bit flip e phase flip. No caso do CBQ bit flip, os 1s devem ser substituídos por operadores Xs, ou seja, temos o operador lógico $\overline{X} = X_1X_2X_3$. Já no caso do CBQ phase flip, os 1s devem ser substituídos por operadores Zs, ou seja, temos o operador lógico $\overline{Z} = Z_1Z_2Z_3$.

A detecção de possíveis erros X e Z sobre as palavras-código geradas pelas operações (15) e (16) é feita através da medição dos geradores de cada um dos CBQs. É fácil de verificar que existe um mapeamento entre as síndromes clássicas $s = \{0,\ 1\}$ e os autovalores $\alpha = \{+1,\ -1\}$ obtidos com a medição destes geradores. Este mapeamento é estabelecido pela relação $s = (1 - \alpha)/2$ (mod 2). Portanto, podemos usar esta relação para adaptar o algoritmo de votação majoritária clássico ao contexto quântico. Esta técnica permite-nos identificar sem ambigüidades o vetor "erro de bit" sobre os qubits da palavra-código gerada pela operação (15) e o vetor "erro de fase" sobre os blocos da palavra-código gerada pela operação (16). Veja a Tabela 1.

Considere agora que o CBC de repetição de taxa 1/3 seja concatenado ao seu CBC equivalente de taxa 3/9. O CBC resultante da concatenação é um código de repetição de taxa 1/9 com a seguinte matriz geradora:

$$\mathbf{G}_C = \left[\begin{array}{ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]. \tag{19}$$

Este CBC tem distância $d_c = 9$ e, portanto, pode corrigir até quatro erros clássicos. Estes quatro erros clássicos estão associados aos quatro erros quânticos da base de um erro quântico arbitrário (X, Z, $Y = XZ$, I). Portanto, o correspondente CBQ de taxa 1/9, conhecido como código de Shor, pode corrigir um erro quântico arbitrário sobre um qubit. A operação de codificação do código de Shor pode ser escrita como:

$$|u\rangle \mapsto \frac{1}{2\sqrt{2}} \sum_{p,\ q,\ r=(0,\ 0,\ 0)}^{(1,\ 1,\ 1)} (-1)^{(p+q+r)u} |p,\ p,\ p,\ q,\ q,\ q,\ r,\ r,\ r\rangle, \tag{20}$$

na qual $u = \{0,\ 1\}$. Ou, mais explicitamente,

$$\begin{aligned} |0\rangle &\mapsto \frac{(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)}{2\sqrt{2}}, \\ |1\rangle &\mapsto \frac{(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)}{2\sqrt{2}}. \end{aligned} \tag{21}$$

A ordem da concatenação é importante. É fácil de verificar que se tivéssemos primeiro codificado um qubit com o código bit flip, e, em seguida, codificado cada um dos três qubits com o código phase flip, teríamos chegado ao final com um "código universal" do tipo $|000\ldots000\rangle$, $|000\ldots001\rangle$, $|000\ldots010\rangle$, ..., $|111\ldots101\rangle$, $|111\ldots110\rangle$, $|111\ldots111\rangle$, incapaz, portanto, de garantir a correção de qualquer erro Z ou X.

[2]Dados dois operadores A e B, o comutador entre A e B é definido como $[A,\ B] = AB - BA$. Analogamente, o anticomutador entre A e B é definido como $\{A,\ B\} = AB + BA$.

As linhas da matriz (18) podem ser usadas para escrever os geradores do código de Shor. Para isto, considere que a matriz de verificação de paridade do CBC de taxa 3/9, seja denotada por $\mathbf{H}_X$, e que a matriz de verificação de paridade do CBC de taxa 1/3, expandida para o CBC de taxa 1/9 (para isto, basta tomar cada uma das linhas da matriz de verificação de paridade do CBC de taxa 1/3 como sequências de informação para o CBC de taxa 3/9), seja denotada por $\mathbf{H}_Z$. Com as matrizes $\mathbf{H}_X$ e $\mathbf{H}_Z$ podemos construir a matriz de verificação de paridade $\mathbf{H}_C$ da matriz geradora (19):

$$\mathbf{H}_C = \left[\begin{array}{c} \mathbf{H}_X \\ \mathbf{H}_Z \end{array} \right], \tag{22}$$

na qual

$$\mathbf{H}_X = \begin{array}{ccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}, \tag{23}$$

e

$$\mathbf{H}_Z = \begin{array}{ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}. \tag{24}$$

Usamos as linhas da matriz $\mathbf{H}_X$ para obter os geradores Z e as linhas da matriz $\mathbf{H}_Z$ para obter os geradores X. Os 0s e 1s da matriz $\mathbf{H}_X$ devem ser substituídos, respectivamente, por operadores Is e Zs, e os 0s e 1s da matriz $\mathbf{H}_Z$ devem ser substituídos, respectivamente, por operadores Is e Xs. Veja a Tabela 2. Usamos as síndromes obtidas com as medições dos geradores Z e X no algoritmo de votação majoritária para detectar o "vetor erro de bit" sobre os qubits da palavra-código (21) e o "vetor erro de fase" sobre os blocos da palavra-código (21).

Por exemplo, suponha que obtemos o seguinte conjunto de autovalores com a medição dos oito geradores da Tabela 2: $(+1,\ -1,\ +1,\ +1,\ +1,\ +1,\ -1,\ +1)$. O algoritmo de votação majoritária nos diz que ocorreu um erro X no terceiro qubit e um erro Z no primeiro bloco (ou seja, em um dos três primeiros qubits). Para corrigirmos ambos os erros, basta aplicar o operador X sobre o terceiro qubit e o operador Z sobre *qualquer* um dos três primeiros qubits.

Os operadores lógicos $\overline{X}$ e $\overline{Z}$ atuando sobre os qubits de informação são obtidos através da linha da matriz (19). Para obter $\overline{X}$, os 1s da matriz (19) devem ser substituídos por operadores Zs, e para obter $\overline{Z}$, os 1s da matriz (19) devem ser substituídos por operadores Xs. Portanto, temos $\overline{X} = Z_1Z_2Z_3Z_4Z_5Z_6Z_7Z_8Z_9$ e $\overline{Z} = X_1X_2X_3X_4X_5X_6X_7X_8X_9$.

Muitos conceitos importantes sobre a teoria de correção de erros quânticos estão ilustrados no código de Shor:

1. **Medir as síndromes não implica em medir os qubits.** As síndromes podem ser obtidas sem o ganho de qualquer informação sobre os estados codificados. Os erros unitários (X e Z) podem ser invertidos sem o conhecimento da informação codificada. Isto é análogo ao método de decodificação clássica que projeta o estado recebido do canal sobre uma classe lateral do arranjo padrão e inverte o erro mais provável (de peso mínimo).

2. **A redundância como forma de proteger a informação.** A redundância é usada para inserir o espaço das palavras-código em um espaço maior, de forma a fazer com que os erros corrigíveis mapeiem o espaço do código em subespaços ortogonais (sem intersecção). Isto é o análogo quântico das esferas de Hamming.

3. **Os erros são locais e a informação codificada é não local.** É importante enfatizar a hipótese central que serve de base para a construção de um código quântico − erros afetando diferentes qubits são, em boa aproximação, não correlacionados. Assumimos que um evento que causa um erro em dois qubits é muito menos provável que um evento que causa um erro em um qubit. É claro que uma questão física é saber se esta hipótese é justificada ou não − podemos facilmente imaginar processos que causam erros em dois qubits de uma só vez. Se tais erros correlacionados forem comuns, a codificação falha.

4. **O código de Shor pode corrigir um erro quântico arbitrário ocorrendo em um qubit**. O código de Shor pode corrigir um erro quântico arbitrário porque os códigos clássicos associados aos códigos bit flip, phase flip e bit-phase flip têm distâncias três (o que garante a correção de um erro X), três (o que garante a correção de um erro Z) *e* nove (o que garante a correção de um erro cuja *base* é gerada pelos erros X, Z, XZ e I), respectivamente. O grupo de erros que este código corrige é composto por dois geradores, $\langle X, Z\rangle$. Isto é aparentemente impossível, já que o espaço de síndromes é finito e o número possível de erros é infinito. Porém, o *continuum* de erros pode ser *discretizado*. Para ver isto, suponha que um erro quântico arbitrário E tenha ocorrido em um qubit. Como E pode ser sempre escrito como $E = c_iI + c_xX + c_yY + c_zZ$, o estado corrompido é a superposição $E|\psi\rangle = c_i|\psi\rangle + c_xX|\psi\rangle + c_yY|\psi\rangle + c_zZ|\psi\rangle$. As medições de síndrome, as quais identificam os erros I, X, Y e Z, *projetam* o estado corrompido sobre um dos quatro termos, todos eles corrigíveis. Em outras palavras, *quantizamos* os erros. Embora os erros na informação quântica possam ser pequenos, fazemos medidas que projetam o estado sobre um estado sem nenhum erro ou sobre um estado com um erro que pertence a um conjunto discreto de erros que sabemos como corrigir [9].

5. **Erros Z atuando em diferentes qubits dentro do mesmo bloco têm efeitos idênticos.** Não é possível, nem necessário, distinguir tais erros. Um código quântico é *não degenerado* se todos os erros corrigíveis podem ser identificados sem ambiguidade; caso contrário, é degenerado. O código de Shor é, portanto, um código degenerado.

Como veremos mais adiante, estes conceitos estão presentes também no CCQ [(4, 1, 3)].

4. UM CCQ [(4, 1, 3)]

Considere o codificador convolucional clássico (2, 1, 2) ótimo com a seguinte matriz geradora:

$$\mathbf{G}(D) = \left[\begin{array}{cc} 1+D^2 & 1+D+D^2 \end{array} \right]. \qquad (25)$$

O CCC (2, 1, 2) gerado por este codificador tem $d_{free} = 5$ e, portanto, pode corrigir até dois erros clássicos. Este CCC (2, 1, 2) pode ser usado na construção de um CCQ [(2, 1, 2)] para o canal bit flip com a seguinte operação de codificação:

$$\bigotimes_{t=0}^{+\infty} |u_t\rangle \longmapsto \bigotimes_{t=0}^{+\infty} |v_t^{(1)},\ v_t^{(2)}\rangle, \qquad (26)$$

na qual

$$\begin{aligned} v_t^{(1)} &= u_t + u_{t-2}, \\ v_t^{(2)} &= u_t + u_{t-1} + u_{t-2}, \end{aligned} \qquad (27)$$

para todo $u_t \in \{0,\ 1\}$. Definimos $u_{-1} = u_{-2} = 0$. Para uma sequência de informação finita com N qubits, cada um dos 2^N estados da base é codificado em um estado com $2(N+2)$ qubits.

Através da transformada de Hadamard, é possível obter também a operação de codificação do CCQ [(2, 1, 2)] para o canal phase flip, a saber:

$$\bigotimes_{t=0}^{+\infty} |u_t\rangle \longmapsto \bigotimes_{t=0}^{+\infty} \left\{ \frac{1}{2}(|0\rangle + (-1)^{v_t^{(1)}}|1\rangle)\ (|0\rangle + (-1)^{v_t^{(2)}}|1\rangle) \right\}, \qquad (28)$$

ou, mais compactamente,

$$\bigotimes_{t=0}^{+\infty} |u_t\rangle \longmapsto \bigotimes_{t=0}^{+\infty} \left\{ \frac{1}{2} \sum_{(p_t,\ q_t)=(0,\ 0)}^{(1,\ 1)} (-1)^{v_t^{(1)} p_t + v_t^{(2)} q_t} |p_t,\ q_t\rangle \right\}. \qquad (29)$$

Aqui, para uma sequência de informação finita com N qubits, cada um dos 2^N estados da base é codificado em uma superposição com $2^{2(N+2)}$ estados, cada um dos quais com $2(N+2)$ qubits.

Os CCQs [(2, 1, 2)] gerados pelas operações (26) e (28) são capazes de corrigir, respectivamente, até dois erros X e dois erros Z. Para determinarmos os geradores destes CCQs, devemos encontrar uma matriz de verificação de paridade para a matriz geradora (25). Por se tratar de um codificador de taxa $1/2$, temos:

$$\mathbf{H}(D) = \left[\begin{array}{cc} 1+D+D^2 & 1+D^2 \end{array} \right]. \qquad (30)$$

As linhas da matriz (30) na forma semi-infinita são usadas para escrever os geradores dos CCQs [(2, 1, 2)]. No caso do CCQ bit flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Zs, e no caso do CCQ phase flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Xs. Veja a Tabela 3. Para uma sequência de informação finita com N qubits, há a necessidade de considerar somente $2(N+2)-N = N+4$ geradores para descrever o subespaço dos CCQs [(2, 1, 2)] truncados.

Analogamente, as linhas da matriz (25) na forma semi-infinita são usadas para escrever os operadores lógicos sobre os qubits de informação dos CCQs [(2, 1, 2)]. No caso do CCQ bit flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Xs, e no caso do CCQ phase flip, os 0s e 1s devem ser substituídos, respectivamente, por operadores Is e Zs. Veja a Tabela 4. Para uma sequência de informação finita com N qubits, há a necessidade de considerar somente N operadores lógicos.

A detecção de possíveis erros X e Z sobre as palavras-código geradas pelas operações (26) e (28) é feita através da medição dos geradores de cada um dos CCQs [(2, 1, 2)]. É fácil de verificar que existe um mapeamento entre as síndromes clássicas $s_t = \{0,\ 1\}$ e os autovalores $\alpha_t = \{+1,\ -1\}$ destes geradores. Este mapeamento é estabelecido pela relação $s_t = (1-\alpha_t)/2$ (mod 2)(para $t = 0,\ 1,\ 2, \ldots$). Portanto, podemos usar esta relação para adaptar o algoritmo de decodificação de síndromes (ADS) de Reed e Truong [10] ao contexto quântico. Esta técnica permite-nos identificar sem ambigüidades o vetor "erro de bit" sobre os qubits da palavra-código gerada pela operação (26) e o vetor "erro de fase" sobre os blocos da palavra-código gerada pela operação (28).

Para que possamos usar o ADS no processo de detecção de possíveis erros X e Z, temos que obter as soluções gerais da equação de síndromes $\mathbf{s}(D) = \mathbf{e}(D)\mathbf{H}^T(D)$ para o codificador CCC (2, 1, 2) com matriz geradora (25). De acordo com [10], estas soluções são:

$$\begin{aligned} \mathbf{e}^{(1)}(D) &= D\mathbf{s}(D) + \mathbf{t}(D) + D^2\mathbf{t}(D), \\ \mathbf{e}^{(2)}(D) &= \mathbf{s}(D) + D\mathbf{s}(D) + \mathbf{t}(D) + D\mathbf{t}(D) + D^2\mathbf{t}(D), \end{aligned} \qquad (31)$$

nas quais $\mathbf{t}(D)$ é um polinômio arbitrário do anel de polinômios $F[D]$.

Os valores de $\mathbf{t}(D)$, $D\mathbf{t}(D)$ e $D^2\mathbf{t}(D)$ ao longo da treliça do ADS podem ser obtidos através da Tabela 5. Definimos o estado inicial da treliça como $(D\mathbf{t}(D),\ D^2\mathbf{t}(D)) = (0,\ 0)$. Além disso, definimos $\mathbf{s}(D) = 0$ antes do estágio 0. O ADS então seleciona o caminho na treliça com o menor peso de Hamming [10].

Os dois exemplos a seguir mostram como o ADS pode ser usado para detectar possíveis erros X e Z sobre as palavras-código geradas pelas operações (26) e (28).

Suponha que usemos o CCQ bit flip para codificar dois qubits de informação. Neste caso, são necessários apenas seis geradores da Tabela 3 para descrever o subespaço do CCQ bit flip truncado. Suponha também que o conjunto de autovalores obtido com a medição destes seis geradores seja o conjunto $(-1,\ +1,\ -1,\ -1,\ +1,\ +1)$. As correspondentes síndromes clássicas são, portanto, $(1,\ 0,\ 1,\ 1,\ 0,\ 0)$. Pode-se verificar que, para este conjunto de síndromes, o ADS seleciona o vetor de erro $\hat{e} = [10\ 01\ 00\ 00]$ como estimativa. No contexto quântico, isto significa que ocorreu um erro X no primeiro e no quarto qubits. Portanto, para recuperar o estado inicial, temos que aplicar um operador X no primeiro e no quarto qubits.

Suponha agora que usemos o CCQ phase flip para codificar dois qubits de informação. Suponha também que o conjunto de autovalores obtido com a medição dos seis geradores seja o conjunto $(+1,\ -1,\ -1,\ -1,\ +1,\ +1)$. As correspondentes síndromes clássicas são, portanto, $(0,\ 1,\ 1,\ 1,\ 0,\ 0)$. Pode-se verificar que, para este conjunto de síndromes, o

ADS seleciona o vetor de erro $\hat{e} = [00\ 10\ 00\ 00]$ como estimativa. No contexto quântico, isto significa que ocorreu um erro Z no terceiro bloco (ou, no terceiro qubit). Portanto, para recuperar o estado inicial, temos que aplicar o operador Z no terceiro qubit.

Com o codificador CCC (4, 2, 1) equivalente trivial do codificador CCC (2, 1, 2)[3] é possível construir um CCQ [(4, 2, 1)] capaz de corrigir até dois erros X. A operação de codificação deste CCQ é escrita de forma análoga à operação (26).

A concatenação do codificador CCC (2, 1, 2) com o seu equivalente trivial CCC (4, 2, 1) dá origem a um codificador CCC (4, 1, 3). Veja a Figura 1. Este codificador tem a seguinte matriz geradora na forma semi-infinita:

$$\mathbf{G}_C = \begin{array}{ccccccc} \mathbf{G}_{C,0} & \mathbf{G}_{C,1} & \mathbf{G}_{C,2} & \mathbf{G}_{C,3} & & & \\ & \mathbf{G}_{C,0} & \mathbf{G}_{C,1} & \mathbf{G}_{C,2} & \mathbf{G}_{C,3} & & \\ & & \ddots & \ddots & \ddots & \ddots & \end{array}, \quad (32)$$

na qual $\mathbf{G}_{C,0} = [1110]$, $\mathbf{G}_{C,1} = [1000]$, $\mathbf{G}_{C,2} = [1001]$ e $\mathbf{G}_{C,3} = [1011]$.

O CCC (4, 1, 3) gerado por este codificador tem $d_{free} = 9$. Veja o diagrama de estados na Figura 2. O CCQ [(4, 1, 3)] associado tem distância $d_q = 3$, ou seja, é capaz de corrigir um erro quântico geral. A operação de codificação para este CCQ pode ser escrita como:

$$\bigotimes_{t=0}^{+\infty} |u_t\rangle \longmapsto \bigotimes_{t=0}^{+\infty} \left\{ \sum_{(p_t,\, q_t)=(0,0)}^{(1,1)} \frac{1}{2}(-1)^{v_t^{(1)} p_t + v_t^{(2)} q_t} |w_t^{(1)}, w_t^{(2)}, w_t^{(3)}, w_t^{(4)}\rangle \right\}, \quad (33)$$

na qual

$$\begin{aligned} v_t^{(1)} &= u_t + u_{t-2}, \\ v_t^{(2)} &= u_t + u_{t-1} + u_{t-2}, \end{aligned} \quad (34)$$

para todo $u_t \in \{0,\ 1\}$ e com $u_{-1} = u_{-2} = 0$, e

$$\begin{aligned} w_t^{(1)} &= p_t + p_{t-1}, \\ w_t^{(2)} &= p_t + p_{t-1} + q_{t-1}, \\ w_t^{(3)} &= q_t + q_{t-1}, \\ w_t^{(4)} &= q_t + q_{t-1} + p_t, \end{aligned} \quad (35)$$

com $p_{-1} = q_{-1} = 0$.

Para um melhor entendimento da dinâmica de geração deste CCQ, considere o exemplo de uma sequência de informação com dois qubits. Cada um dos quatro estados da base é codificado em uma superposição de 2^2 estados de comprimento 4 no estágio 0, uma superposição de 2^4 estados de comprimento 8 no estágio 1, uma superposição de 2^6 estados de comprimento 12 no estágio 2, uma superposição de 2^8 estados de comprimento 16 no estágio 3 e uma superposição de 2^8 estados de comprimento 20 no estágio 4. Neste estágio, temos uma palavra-código válida para dois qubits de informação. Repare que o comprimento deste código cresce durante uma unidade de tempo além do número de estados. Isto ocorre porque o código bit flip é o segundo da cadeia de concatenação e possui memória unitária. Generalizando, cada um dos 2^N estados da base de uma sequência de informação com N qubits é codificado em uma superposição de $2^{2(N+2)}$ estados, cada um dos quais com $2(2(N+2)+2) = 4N+12$ qubits.

As linhas da matriz (30) na forma semi-infinita podem ser usadas para escrever os geradores do CCQ [(4, 1, 3)]. Para isto, considere que a matriz de verificação de paridade do codificador CCC (4, 2, 1) seja denotada por $\mathbf{H}_X$, e que a matriz de verificação de paridade do codificador CCC (2, 1, 2), expandida para o codificador CCC (4, 1, 3) (para isto, basta tomar cada uma das linhas da matriz de verificação de paridade do codificador CCC (2, 1, 2) como sequências de informação para o codificador CCC (4, 2, 1)), seja denotada por $\mathbf{H}_Z$. Com as matrizes $\mathbf{H}_X$ e $\mathbf{H}_Z$ podemos construir a matriz de verificação de paridade $\mathbf{H}_C$ da matriz geradora (32):

$$\mathbf{H}_C = \begin{array}{ccccccc} \mathbf{H}_X & & & & & & \\ \mathbf{H}_Z & & & & & & \\ \mathbf{H}_{X,0} & & & & & & \\ \mathbf{H}_{X,1} & \mathbf{H}_{X,0} & & & & & \\ & \mathbf{H}_{X,1} & \mathbf{H}_{X,0} & & & & \\ & & \mathbf{H}_{X,1} & \mathbf{H}_{X,0} & & & \\ & & & \mathbf{H}_{X,1} & \mathbf{H}_{X,0} & & \\ & & & & & \ddots & \ddots \\ \mathbf{H}_{Z,-0} & \mathbf{H}_{Z,0} & & & & & \\ \mathbf{H}_{Z,-1} & \mathbf{H}_{Z,1} & \mathbf{H}_{Z,0} & & & & \\ \mathbf{H}_{Z,-2} & \mathbf{H}_{Z,2} & \mathbf{H}_{Z,1} & \mathbf{H}_{Z,0} & & & \\ & \mathbf{H}_{Z,-2} & \mathbf{H}_{Z,2} & \mathbf{H}_{Z,1} & \mathbf{H}_{Z,0} & & \\ & & \ddots & \ddots & \ddots & \ddots & \end{array}, \quad (36)$$

na qual

$$\mathbf{H}_{X,0} = \begin{bmatrix} 1100 \\ 1011 \end{bmatrix}, \quad \mathbf{H}_{X,1} = \begin{bmatrix} 1110 \\ 0011 \end{bmatrix},$$

$$\begin{aligned} \mathbf{H}_{Z,0} &= \begin{bmatrix} 1011 \end{bmatrix}, & \mathbf{H}_{Z,-0} &= \begin{bmatrix} 1110 \end{bmatrix}, \\ \mathbf{H}_{Z,1} &= \begin{bmatrix} 0010 \end{bmatrix}, & \mathbf{H}_{Z,-1} &= \begin{bmatrix} 1101 \end{bmatrix}, \\ \mathbf{H}_{Z,2} &= \begin{bmatrix} 0110 \end{bmatrix}, & \mathbf{H}_{Z,-2} &= \begin{bmatrix} 1110 \end{bmatrix}. \end{aligned} \quad (37)$$

Usamos as linhas da matriz $\mathbf{H}_X$ para obter os geradores Z e as linhas da matriz $\mathbf{H}_Z$ para obter os geradores X. Os 0s e 1s da matriz $\mathbf{H}_X$ devem ser substituídos, respectivamente, por operadores Is e Zs, e os 0s e 1s da matriz $\mathbf{H}_Z$ devem ser substituídos, respectivamente, por operadores Is e Xs. Veja a Tabela 6. Para uma sequência de informação finita com N qubits, precisamos considerar somente a medida de $(4-2)((N+3)+1) = 2N+8$ geradores Z e de $(2-1)((N+2)+2) = N+4$ geradores X para descrever o subespaço do CCQ [(4, 1, 3)] truncado. Usamos as síndromes obtidas com a medição dos geradores Z e X nas soluções (31) para detectar, através de duas treliças do ADS, o "vetor erro de bit" sobre os qubits da palavra-código (33) e o "vetor erro de fase" sobre os blocos do CCQ phase flip [(2, 1, 2)] associado. Identificado o "vetor erro de fase" sobre os blocos do CCQ phase flip [(2, 1, 2)] associado, pode-se então determinar para qual qubit da

[3] O codificador CCC (4, 2, 1) equivalente trivial do codificador CCC (2, 1, 2) é o codificador CCC (4, 2, 1) com a *mesma* matriz geradora do codificador CCC (2, 1, 2).

palavra-código (33) o(s) erro(s) de fase se propagou (ou se propagaram)[4]

O exemplo a seguir mostra como o ADS pode ser usado para detectar possíveis erros X e Z sobre um mesmo qubit (ou seja, um erro Y sobre um qubit) do CCQ [(4, 1, 3)].

Suponha que usemos o CCQ [(4, 1, 3)] para codificar dois qubits de informação. Neste caso, são necessários apenas dez geradores Z e seis geradores X da Tabela 6 para descrever o subespaço do CCQ [(4, 1, 3)] truncado. Suponha também que o conjunto de autovalores obtido com a medição dos doze geradores Z seja o conjunto $(+1, -1, -1, -1, +1, +1, +1, +1, +1, +1, +1, +1)$ e que o conjunto de autovalores obtido com a medição dos seis geradores X seja o conjunto $(-1, +1, -1, +1, +1, +1)$. As correspondentes síndromes clássicas são, portanto, $(0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)$ e $(1, 0, 1, 0, 0, 0)$. Pode-se verificar que, para estes conjuntos de síndromes, o ADS seleciona os vetores de erro $\hat{e} = [00\ 10\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00]$ e $\hat{e} = [01\ 00\ 00\ 00]$ como estimativas. No contexto quântico, isto significa que ocorreu um erro X no terceiro qubit do CCQ [(4, 1, 3)] e um erro Z no segundo bloco do CCQ phase flip [(2, 1, 2)] associado. Pode-se verificar que um erro Z no segundo bloco do CCQ phase flip [(2, 1, 2)] associado propaga-se para um erro Z no terceiro qubit do CCQ [(4, 1, 3)]. Portanto, para recuperar o estado inicial, temos que aplicar um operador X e um operador Z no terceiro qubits.

Se somos capazes de detectar e corrigir um erro X, Z e Y sobre qualquer qubit do CCQ [(4, 1, 3)], podemos afirmar que somos capazes também de corrigir um erro quântico arbitrário sobre qualquer qubit do CCQ [(4, 1, 3)].

Os operadores lógicos $\overline{X}$ e $\overline{Z}$ atuando sobre os qubits de informação são obtidos através das linhas da matriz (32). Para obter $\overline{X}$, os 0s e 1s da matriz (32) devem ser substituídos, respectivamente, por operadores Is e Zs, e para obter $\overline{Z}$, os 0s e 1s da matriz (32) devem ser substituídos, respectivamente, por operadores Is e Xs. Veja a Tabela 7. Para uma sequência de informação finita com N qubits, há a necessidade de considerar somente N operadores lógicos.

5. PERSPECTIVAS DE PESQUISA

Após a descoberta do código de Shor em 1995, muitos outros CBQs foram estudados. O surgimento de CBQs cada vez mais eficientes que o código de Shor, como o código de Steane de taxa 1/7 [11, 12] e o código perfeito[5] de taxa 1/5 [13] fez com que o estudo de uma subclasse de CBQs concatenados ficasse em segundo plano. Além disso, foi estudada a conexão entre a teoria de CBQs e a geometria ortogonal [14] e desenvolvida uma teoria de CBQs sobre o $GF(4)$ [15]. Atualmente, o estudo concentra-se na busca de CBQs eficientes para sistemas quânticos cujos estados estão emaranhados.

Antes do CCQ [(4, 1, 3)], Chau [16, 17] e Ollivier e Tillich [18, 19] já haviam proposto a construção de CCQs, sem contudo apresentar um método sistemático de codificação e decodificação. O CCQ [(4, 1, 3)] foi o primeiro CCQ a utilizar um processo sistemático de construção, capaz de ser estendido para outras taxas e memórias [21]. Após a publicação deste CCQ em 2004, Forney [20] apresentou alguns CCQs de taxa 1/3 capazes de corrigir um erro quântico arbitrário. Estes CCQs foram baseados em CCCs auto-ortogonais escolhidos do $GF(4)$ e em construções do tipo CSS (de Calderbank, Shor e Steane) conhecidas da classe dos CBQs.

O estudo de uma subclasse de CCQs concatenados é particularmente interessante pela simplicidade de codificação e decodificação, sobretudo se o objetivo for a construção de CCQs capazes de corrigir mais de um erro quântico arbitrário. Em particular, o problema da complexidade de decodificação para CCQs estabilizadores de subclasses mais gerais capazes de corrigir mais de um erro quântico arbitrário não parece, em princípio, ser um problema fácil. Além da simplicidade, o estudo de CCQs concatenados também é interessante porque deixa explícita a importância que a memória desempenha no processo de codificação e decodificação, conceito este ainda não abordado com suficiente clareza em todas as construções já propostas para CCQs. O estudo do papel da memória na construção de CCQs concatenados também pode servir de base para um melhor entendimento do papel da memória em subclasses mais gerais de CCQs estabilizadores, como a subclasse CSS proposta por Forney.

6. LISTA DE ACRÔNIMOS

- CCEQs: códigos corretores de erros quânticos
- CBQs: códigos de bloco quânticos
- CCQs: códigos convolucionais quânticos
- CCECs: códigos corretores de erros clássicos
- CBCs: códigos de bloco clássicos
- CCCs: códigos convolucionais clássicos
- EPR: Einstein-Podolsky-Rosen
- ADS: algoritmo de decodificação de síndromes
- CSS: Calderbank-Shor-Steane

REFERÊNCIAS

[1] J. Preskill, *A Course on Quantum Computation and Quantum Information - Lecture Notes*, California Institute of Technology, 1998; disponível em www.theory.caltech.edu/people/preskill/ph229.

[2] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory", *Phys. Rev. A*, 52, pp. 2493-2496, 1995.

[3] A. C. A. de Almeida and R. Palazzo Jr., "A Concatenated [(4, 1, 3)] Quantum Convolutional Code", *2004 IEEE Information Theory Workshop*, San Antonio, Texas, 2004.

[4] O processo de identificação de erros Z aqui é mais complexo porque um CCQ não pode ser decomposto em um produto tensorial de blocos de qubits.

[5] Diz-se que é perfeito porque satura o limitante quântico de Hamming [7]

[4] C. C. Tannoudji, B. Diu and F. Laloë, *Quantum Mechanics - Volume One*, Hermann and John Wiley and Sons, France, 1977.

[5] J. J. Sakurai, *Modern Quantum Mechanics*, Addison-Wesley Publishing Company, USA, 1994.

[6] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned", *Nature*, 299, pp. 802-803, 1982.

[7] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, UK, 2000.

[8] D. Gottesman, *Stabilizer codes and quantum error correction*, PhD Thesis, California Institute of Technology, USA, 1997; disponível em arXiv e-print quant-ph/9705052.

[9] E. Knill and R. Laflamme, "A theory of quantum error-correcting codes", *Phys. Rev. A*, 55, pp. 900-911, 1997.

[10] I. S. Reed and T. K. Truong, "New syndrome decoder for (n,1) convolutional codes", *Electronics Letters*, 19(9), pp. 344-346, 1983.

[11] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist", *Phys. Rev. A*, 54, pp. 1098-1105, 1996.

[12] A. M. Steane, "Error-correcting codes in quantum theory", *Phys. Rev. Lett.*, 77, pp. 793-797, 1996.

[13] R. Laflamme, C. Miquel, J. P. Paz and W. K. Zurek, "Perfect quantum error correction code", *Phys. Rev. Lett.*, 77, pp. 198-201, 1996.

[14] A. R. Calderbank, E. M. Rains, P. W. Shor and N. J. A. Sloane, "Quantum error correction and orthogonal geometry", *Phys. Rev. Lett.*, 78, pp. 405-408, 1997.

[15] A. R. Calderbank, E. M. Rains, P. W. Shor and N. J. A. Sloane, "Quantum error correction via codes over GF(4)", *IEEE Transactions on Information Theory*, 44, pp. 1369-1387, 1998.

[16] H. F. Chau, "Quantum convolutional error correcting codes", *Phys. Rev. A*, 58, pp. 905-909, 1998.

[17] H. F. Chau, "Good quantum convolutional error correction codes and their decoding algorithm exist", *Phys. Rev. A*, 60, pp. 1966-1974, 1999.

[18] H. Ollivier and J. P. Tillich, "Description of a quantum convolutional code", *Phys. Rev. Lett.*, 91, pp. 21-24, 2003.

[19] H. Ollivier and J. P. Tillich, "Quantum convolutional codes: fundamentals", submetido à revista *IEEE Trans. Inform. Theory*, 2004; disponível em www.arxiv.org/abs/quant-ph/0401134.

[20] G. D. Forney and S. Guha, "Simple Rate-1/3 Convolutional and Tail-Biting Quantum Error-Correcting Codes", 2005; disponível em www.arxiv.org/abs/quant-ph/0501099.

[21] A. C. A. de Almeida, *Códigos Convolucionais Quânticos Concatenados*, Tese de Doutorado, Universidade Estadual de Campinas (UNICAMP), Brasil, Outubro de 2004.

Autovalores de $(Z_1Z_2,\ Z_2Z_3)$	Erro Detectado	Correção
$(+1,+1)$	nenhum	nenhuma
$(-1,+1)$	1^{o}qubit	X_1
$(+1,-1)$	3^{o}qubit	X_3
$(-1,-1)$	2^{o}qubit	X_2

Tabela 1. Relação entre os autovalores obtidos das medições dos geradores do CBQ bit flip e as correspondentes síndromes clássicas. Para o CBQ phase flip, basta substituir o operador Z pelo operador X e vice-versa.

$M_{X,1}$	Z	Z	I	I	I	I	I	I	I
$M_{X,2}$	I	Z	Z	I	I	I	I	I	I
$M_{X,3}$	I	I	I	Z	Z	I	I	I	I
$M_{X,4}$	I	I	I	I	Z	Z	I	I	I
$M_{X,5}$	I	I	I	I	I	I	Z	Z	I
$M_{X,6}$	I	I	I	I	I	I	I	Z	Z
$M_{Z,1}$	X	X	X	X	X	X	I	I	I
$M_{Z,2}$	I	I	I	X	X	X	X	X	X

Tabela 2. Os geradores do código de Shor.

M_0	Z	Z	I	I	I	I	I	I	I	I
M_1	Z	I	Z	Z	I	I	I	I	I	I
M_2	Z	Z	Z	I	Z	Z	I	I	I	I
M_3	I	I	Z	Z	Z	I	Z	Z	I	I
$\vdots$					$\ddots$		$\ddots$		$\ddots$	
M_∞	I	I	I	I	I	I	I	I	Z	Z.

Tabela 3. Geradores do CCQ [(2, 1, 2)] para o canal bit flip. Para obtermos os geradores do CCQ [(2, 1, 2)] para o canal phase flip, basta substituir o operador Z pelo operador X.

$\overline{X}_1$	X	X	I	X	X	X	I	I	I	I
$\overline{X}_2$	I	I	X	X	I	X	X	X	I	I
$\vdots$					$\ddots$		$\ddots$		$\ddots$	
$\overline{X}_\infty$	I	I	I	I	X	X	I	X	X	X.

Tabela 4. Operadores lógicos do CCQ [(2, 1, 2)] para o canal bit flip. Para obtermos os operadores lógicos do CCQ [(2, 1, 2)] para o canal phase flip, basta substituir o operador X pelo operador Z.

$D\mathbf{t}(D), D^2\mathbf{t}(D)$	$\mathbf{t}(D)$=0	$\mathbf{t}(D)$=1
$a = 00$	$a = 00$	$c = 10$
$b = 01$	$a = 00$	$c = 10$
$c = 10$	$b = 01$	$d = 11$
$d = 11$	$b = 01$	$d = 11$

Tabela 5. Tabela de estados do registro de deslocamento para $\mathbf{t}(D)$.

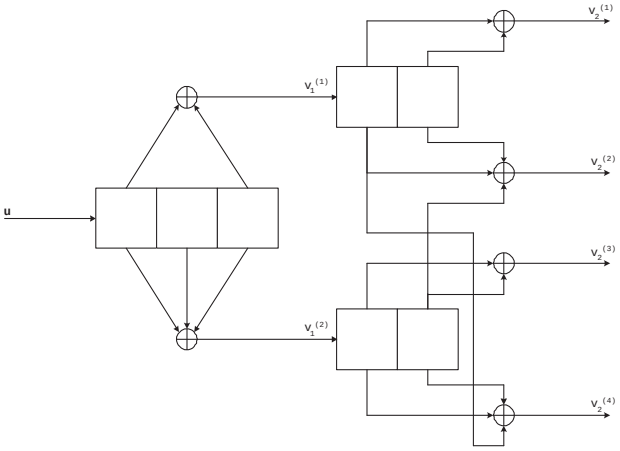

Figura 1. Concatenação dos codificadores CCC (2, 1, 2) e CCC (4, 2, 1).

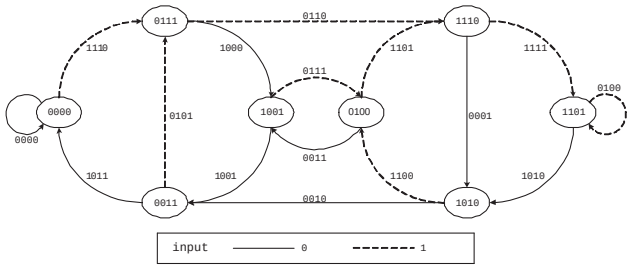

Figura 2. Diagrama de estados para o codificador CCC (4, 1, 3).

$M_{X,0}$	Z	Z	I	I	I	I	I	I	I	I	I	I	I	I	I	I
$M_{X,1}$	Z	I	Z	Z	I	I	I	I	I	I	I	I	I	I	I	I
$M_{X,2}$	Z	Z	Z	I	Z	Z	I	I	I	I	I	I	I	I	I	I
$M_{X,3}$	I	I	Z	Z	Z	I	Z	Z	I	I	I	I	I	I	I	I
$M_{X,4}$	I	I	I	I	Z	Z	Z	I	Z	Z	I	I	I	I	I	I
$M_{X,5}$	I	I	I	I	I	I	Z	Z	Z	I	Z	Z	I	I	I	I
$M_{X,6}$	I	I	I	I	I	I	I	I	Z	Z	Z	I	Z	Z	I	I
$M_{X,7}$	I	I	I	I	I	I	I	I	I	I	Z	Z	Z	I	Z	Z
$\vdots$													$\ddots$		$\ddots$	
$M_{X,\infty}$	I	I	I	I	I	I	I	I	I	I	I	I	I	I	Z	Z.
$M_{Z,0}$	X	X	X	I	X	I	X	X	I	I	I	I	I	I	I	I
$M_{Z,1}$	X	X	I	X	I	I	X	I	X	I	X	X	I	I	I	I
$M_{Z,2}$	X	X	X	I	I	X	X	I	I	I	X	I	X	I	X	X
$M_{Z,3}$	I	I	I	I	X	X	X	I	I	X	X	I	I	I	X	I
$\vdots$									$\ddots$				$\ddots$			
$M_{Z,\infty}$	I	I	I	I	I	I	I	I	X	X	X	I	X	I	X	X.

Tabela 6. Geradores do CCQ [(4, 1, 3)] descrito pela operação de codificação (33).

$\overline{X}_1$	Z	Z	Z	I	Z	I	I	I	Z	I	I	Z	Z	I	Z	Z
$\overline{X}_2$	I	I	I	I	Z	Z	Z	I	Z	I	I	I	Z	I	I	Z
$\vdots$									$\ddots$				$\ddots$			
$\overline{X}_\infty$	Z	Z	Z	I	Z	I	I	I	Z	I	I	Z	Z	I	Z	Z.
$\overline{Z}_1$	X	X	X	I	X	I	I	I	X	I	I	X	X	I	X	X
$\overline{Z}_2$	I	I	I	I	X	X	X	I	X	I	I	I	X	I	I	X
$\vdots$									$\ddots$				$\ddots$			
$\overline{Z}_\infty$	X	X	X	I	X	I	I	I	X	I	I	X	X	I	X	X.

Tabela 7. Operadores lógicos do CCQ [(4, 1, 3)] descrito pela operação de codificação (33).