UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea Magistrale in
Ingegneria Elettronica

Analisi dei codici quantistici LDPC per la correzione d'errore

---

Analysis of quantum LDPC codes for error correction

Relatore:
Prof. Franco Chiaraluce

Laureando:
Alessio Baldelli

Correlatore:
Dott. Massimo Battaglioni

Anno Accademico 2023-2024

"Without error-correction all information processing,
and hence all knowledge creation, is necessarily bounded.
Error-correction is the beginning of infinity."
— David Deutsch

Sommario

Le prestazioni vicine alla capacità del canale simmetrico classico dei codici Low-Density Parity-Check (LDPC), unitamente alla loro efficiente decodifica iterativa, rendono i codici quantum LDPC (QLPDC) un candidato promettente per la correzione degli errori nel dominio quantum.

In questo lavoro di tesi, si presenta un'analisi completa dei codici QLDPC. Infatti, dopo aver richiamato i concetti chiave della quantum information (Capitolo 1) e della teoria della codifica classica e quantum (Capitolo 2), si introducono le caratteristiche e le proprietà dei codici LDPC classici (Capitolo 3) e poi si presentano i codici QLDPC sia dal punto di vista della progettazione, sia in termini dei loro algoritmi di decodifica, rispettivamente nel Capitolo 4 e nel Capitolo 5.

Inoltre, nel Capitolo 6 è presentato il progetto di un simulatore Monte Carlo per la decodifica BP (Belief Propagation) di codici quantum, tra cui QLDPC.

Infine, nell'Appendice A sono presenti le future direzioni che può prendere questa ricerca.

Indice

Ringraziamenti

In primis desidero ringraziare il mio relatore Professore Franco Chiaraluce ed il mio correlatore Dott. Massimo Battaglioni, che mi hanno guidato, per più di due anni, nella stesura di questo elaborato, dandomi l'opportunità di continuare il lavoro di ricerca iniziato in occasione della stesura della tesi triennale.

Non posso non ringraziare le persone che hanno avuto più influenza nel mio percorso educativo: la mia famiglia. Grazie per avermi aiutato a superare i momenti più difficili, senza di voi non avrei mai potuto raggiungere questo importante traguardo!

Non di meno, ci tengo anche a dire un grande grazie a Camilla per avermi dato tanta forza per superare gli ostacoli di questi anni e per esserci sempre stata.

Ringrazio anche il Dott. Paolo Santini per essere stato soprattutto un ottimo punto di riferimento dal punto di vista umano e come ricercatore, oltre che un appassionante docente.

Ringrazio poi i miei cari amici di sempre Pietro, Gianluca e Francesco per avermi supportato in questo percorso, Alessio e Lorenzo per i saggi consigli nei momenti di difficoltà e i miei fedeli colleghi di corso Giovanni e Federico per avermi aiutato durante questo lungo viaggio.

Un grande pensiero infine va anche a Federico che mi guarda dall'alto e che spero sia fiero di me.

Capitolo 1

Fondamenti di Quantum Information

In questo primo capitolo vengono discusse le conoscenze teoriche di base di meccanica quantistica, indispensabili per la Quantum Information. Tale teoria è a sua volta propedeutica alla comprensione della Quantum Error Correction (QEC), vero oggetto di questo lavoro e presente nei prossimi capitoli.

1.1 Qubit, Superposition ed Entanglement

Per la teoria dell'informazione classica un generico bit può assumere il valore 0 (valore logico basso) o 1 (valore logico alto). Invece, un qubit (quantum bit) è, dal punto di vista della quantum infromation, l'unità logica alla base di una computazione quantum. In generale, dal punto di vista matematico, si può affermare che un qubit è interpretabile come un vettore bi-dimensionale a coefficienti complessi che appartiene allo spazio di Hilbert $\mathcal{H}_2$, dove la dimensione è data, appunto, dai due possibili stati.

Il qubit, inoltre, è caratterizzato dalla sovrapposizione (da qui in avanti chiamata *superposition*) tra un possibile stato $|0\rangle$ ed un possibile stato $|1\rangle$, indicati in letteratura anche con $|g\rangle$ (ground) e $|e\rangle$ (excited), rispettivamente, fintanto che non viene misurato o "osservato". In particolare, con la seguente notazione di Dirac (o notazione bra-ket) $|\cdot\rangle$, si indica uno stato quantum che caratterizza il generico qubit, nell'istante di osservazione.

Approfondimento: Notazione Bra-Ket In fisica, soprattutto in meccanica quantistica, è particolarmente comune la notazione bra-ket per esprimere vettori riga e colonna, rispettivamente. Nello specifico, dato un generico vettore ν, vale:

$$\langle\nu| = \begin{bmatrix}\nu_1 & \nu_2 & \nu_3\end{bmatrix},$$

per la notazione $\langle\cdot|$ (bra). Si può scrivere lo stesso vettore con $|\cdot\rangle$ (ket) come

segue:

$$|\nu\rangle = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \end{bmatrix}.$$

N.B. Si è ipotizzato di trovarsi in uno spazio tridimensionale, data la cardinalità di ν, ma questa proprietà, ovviamente, non è vincolante ai fini della notazione.

Il postulato della superposition afferma che due o più stati quantum, dal punto di vista analitico, possono essere sommati, dando come risultato un altro stato quantum parimenti valido. Dunque ogni stato quantum può essere rappresentato come somma di due o più altri stati distinti. Si osserva pertanto che un qubit può essere, allo stesso tempo, sia nello stato $|1\rangle$ che nello stato $|0\rangle$. Allora, lo stato sovrapposto del qubit si descrive come:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,$$

dove $\{|0\rangle\,,\,|1\rangle\}$ costituiscono la base ortonormale del succitato spazio di Hilbert e per cui vale:

$$\langle i|j\rangle = \delta_{i,j} = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}, \tag{1.1}$$

dove $i = 0, 1$, $j = 0, 1$ e con la notazione $\langle \cdot \,|\, \cdot \rangle$ si indica l'operazione di prodotto interno (che corrisponde al noto prodotto scalare) [1]. L'eq.(1.1) sta a significare che il prodotto interno è unitario soltanto se si considerano due stati uguali (entrambi $|0\rangle$ o $|1\rangle$), altrimenti è zero. Tale base definisce un qubit a due livelli in cui ognuno dei due stati (riportati sopra) può essere espresso come:

$$|0\rangle \equiv |g\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{e} \quad |1\rangle \equiv |e\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

chiamati per questo anche *basi computazionali*. In particolare, la notazione $|g\rangle$, $|e\rangle$ sarà utilizzata in sede di presentazione dei Surface code, presente nell'Appendice¥A.

In generale, per due vettori $|v_1\rangle = \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$ e $|v_2\rangle = \begin{pmatrix} a_2 \\ b_2 \end{pmatrix}$, entrambi appartenenti a $\mathbb{R}$, il prodotto scalare vale:

$$\langle v_1|v_2\rangle = \langle v_1|\cdot|v_2\rangle = \begin{pmatrix} a_1 & b_1 \end{pmatrix} \cdot \begin{pmatrix} a_2 \\ b_2 \end{pmatrix} = a_1a_2 + b_1b_2 = ||v_1||\,||v_2||\,\cos(\theta), \tag{1.2}$$

dove $\cdot$ rappresenta il prodotto tra vettori (o matrici) e θ è l'angolo compreso tra i due vettori. Se $\langle v_1|v_2\rangle = 0$, allora i due vettori si dicono ortogonali.

Si può studiare la (1.2) andando a calcolare il prodotto interno $\langle\cdot|\cdot\rangle$ del generico stato $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ con sé stesso. Si ottiene:

$$\langle\psi|\psi\rangle = \begin{pmatrix} \alpha & \beta \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha\alpha + \beta\beta = |\alpha|^2 + |\beta|^2 = 1,$$

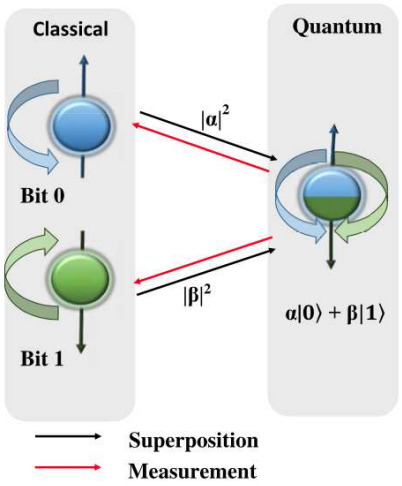

Figura 1.1: Realizzazione di un bit classico e di un qubit usando la rappresentazione mediante lo spin di un elettrone [2].

conosciuta anche come regola di Born. In particolare, si osserva che α e β sono due ampiezze di probabilità che corrispondono a coefficienti complessi, che possono assumere qualsiasi valore, tale che valga la seguente relazione che ne lega i moduli: $|\alpha|^2+|\beta|^2 = 1$. Il loro significato è quello di determinare la probabilità con cui il qubit può collassare in uno dei due stati. Nello specifico, si ha che $|0\rangle$ è caratterizzato da una probabilità pari a $|\alpha|^2$, mentre $|1\rangle$ da $|\beta|^2$.

Si sottolinea ancora una volta che, quando si misura un qubit, esso dà come risultato della misurazione sempre e solo $|0\rangle$ o $|1\rangle$, così come accade per il bit classico.

Esempio Un qubit può essere nello stato

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\,,$$

dunque, quando viene misurato, può dare sia il risultato $|0\rangle$ che il risultato $|1\rangle$ con una probabilità pari al cinquanta per cento, essendo infatti che $|\alpha|^2 = |\beta|^2 = |1/\sqrt{2}|^2 = 0.5$. Si farà riferimento a questo esempio considerato notevole, poiché i due stati sono equiprobabili, anche nel seguito della trattazione.

Dal punto di vista fisico un qubit è un qualunque sistema quantum a due livelli; servendosi pertanto del modello dell'atomo di idrogeno si associa ad uno dei due stati del qubit il verso del momento di spin della particella, secondo il seguente mapping:

$$\begin{cases} |0\rangle \rightarrow |\uparrow\rangle \text{ (spin-up)} \\ |1\rangle \rightarrow |\downarrow\rangle \text{ (spin-down)}, \end{cases}$$

come rappresentato nella Fig. 1.1. Dunque un qubit esiste come superposition dei due stati, ma collassa ad un singolo valore quando viene misurato.

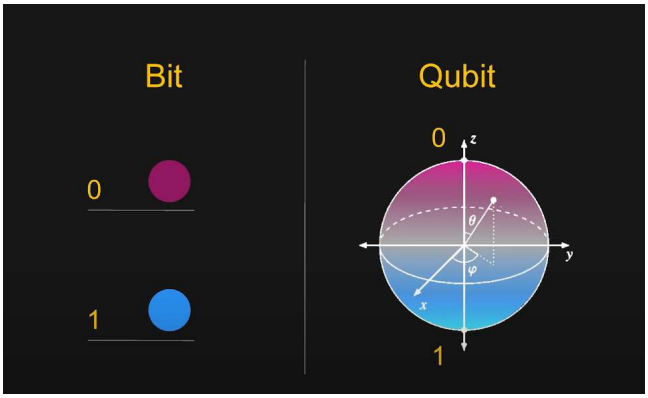

Figura 1.2: Confronto tra i due possibili stati di un bit tradizionale e rappresentazione con la sfera di Bloch dell'informazione contenuta in un qubit.

In alternativa, lo stato di superposition quantum può essere facilmente spiegato rappresentando il generico qubit come la posizione di un punto su una sfera, come appare in Fig. 1.2. La sfera rappresentata in Fig. 1.2 è conosciuta in meccanica quantistica come "Sfera di Bloch", in onore del fisico tedesco Felix Bloch. È dunque una possibile rappresentazione degli stati di un sistema quantum ad 1 qubit. La sfera di Bloch è geometricamente una sfera di raggio unitario i cui punti sulla superficie sono in corrispondenza biunivoca con gli stati del qubit. Il generico stato sovrapposto $|\psi\rangle$ del qubit viene rappresentato nella sfera come segue:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle\,,$$

dove $0 \leq \theta \leq \pi$ e $0 \leq \phi < 2\pi$. I parametri ϕ e θ identificano univocamente un punto di coordinate (x, y, z) sulla sfera unitaria nello spazio euclideo $\mathbb{R}^3$ tramite le seguenti espressioni:

$$\begin{cases} x = \sin(\theta)\cos(\phi) \\ y = \sin(\theta)\sin(\phi) \\ z = \cos(\theta) \end{cases} .$$

L'informazione puntuale rappresentata da ognuno dei possibili infiniti stati che può assumere un qubit è tale da essere rappresentata come un punto sulla superficie di una sfera. Pertanto si può concludere che un qubit può teoricamente contenere una quantità d'informazione enorme rispetto al bit classico.

Concretamente, i ricercatori hanno sviluppato più modalità per ottenere sperimentalmente un qubit; per esempio attraverso le due differenti polarizzazioni di un fotone, o servendosi dell'allineamento di uno spin nucleare in un campo magnetico uniforme oppure tramite i due stati di un elettrone nell'orbita di un singolo atomo, come mostrato in Fig. 1.3 [1]. Nel modello dell'atomo, infatti, l'elettrone può esistere sia nello stato detto di "terra" (ground state, cioè $|g\rangle$), chiamato anche stato fondamentale, sia in quello chiamato "eccitato" (excited state, cioè $|e\rangle$), che corrispondono agli stati $|0\rangle$

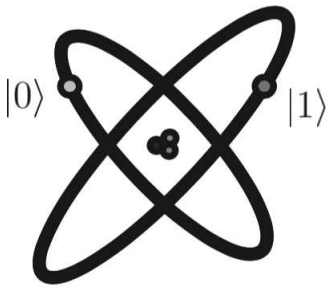

Figura 1.3: Qubit rappresentato da un atomo con due livelli elettronici [1] .

e $|1\rangle$, rispettivamente, del generico qubit.

Un possibile sistema composto da N qubit è rappresentato da un vettore 2^N-dimensionale, descritto come segue:

$$\alpha_0 |00...0\rangle + \alpha_1 |00...1\rangle + ... + \alpha_{2^N-1} |11...1\rangle ,$$

dove $\alpha_i \in \mathbb{C}$ e $\sum_{i=0}^{2^N-1} |\alpha_i|^2 = 1$. Appare ora chiaro che un sistema di N qubit è caratterizzato dalla superposition contemporanea di tutti i 2^N possibili valori, che conferisce a tali sistemi quantum, la proprietà intrinseca conosciuta come *parallelismo* che, teoricamente, permette di avere vantaggi computazionali di ordine esponenziale [1], [2].

L'entanglement, o correlazione quantistica, che letteralmente può essere tradotto con "groviglio", "intreccio", è un fenomeno quantum che presuppone un'azione immediata a distanza e che coinvolge almeno due entità. Per esempio si immagini di prendere in considerazione un evento nello spazio che viene influenzato da un altro evento sempre nello spazio, arbitrariamente distante, in un tempo infinitesimo. Allora con il termine entanglement ci si riferisce all'impossibilità di descrivere le parti di un sistema quantum in maniera indipendente le une dalle altre, a prescindere dalla distanza che le separa. L'entanglement è un fenomeno puramente quantum e non ha alcuna controparte nella fisica classica.

Un importante postulato della meccanica quantistica recita che: *lo spazio di stato di un sistema fisico composito è dato dal prodotto tensoriale degli spazi di stato dei sistemi fisici che lo compongono* [1]. In particolare, ai fini della trattazione, si considera come sistema fisico composito un sistema di N qubit, dove ogni qubit è uno dei sistemi fisici componenti, mentre, per spazio di stato, si intende il generico stato $|\psi\rangle$. Allora lo stato congiunto del macrosistema, composto da N qubit, risulterebbe:

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \ldots \otimes |\psi_N\rangle ,$$

dove $|\psi_i\rangle$ indica lo stato del generico i-esimo qubit e $\otimes$ denota l'operatore di prodotto tensoriale. Invece, nonostante quanto affermato dal postulato, che in generale è sempre valido in meccanica quantistica, tale formulazione

non è assolutamente verificata per la definizione dello stato di un sistema composto da N qubit, nel caso in cui essi siano entangled tra loro [1]. Pertanto, se si considerano dei particolari sistemi quantum, composti da N qubit (entangled) tra loro, lo stato non può essere espresso come prodotto tensoriale dei singoli qubit.

Approfondimento: Prodotto tensore ed entanglement L'operazione di prodotto tensoriale si può esplicitare come segue. Dati $\mathbf{v} \in V$ e $\mathbf{w} \in W$, dove $\mathbf{v} = \alpha |0\rangle + \beta |1\rangle$ e $\mathbf{w} = \gamma |0\rangle + \delta |1\rangle$ e V, W spazi di Hilbert, allora vale:

$$\mathbf{v} \otimes \mathbf{w} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \otimes \begin{pmatrix} \gamma \\ \delta \end{pmatrix} = \begin{pmatrix} \alpha \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \\ \\ \beta \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \end{pmatrix} = \begin{pmatrix} \alpha\gamma \\ \alpha\delta \\ \beta\gamma \\ \beta\delta \end{pmatrix},$$

analogamente, quindi, si ha che:

$$\mathbf{v} \otimes \mathbf{v} = \ \ldots \ = \begin{pmatrix} \alpha^2 \\ \alpha\beta \\ \beta\alpha \\ \beta^2 \end{pmatrix}.$$

Riformulando dal punto di vista matematico il postulato della meccanica quantistica citato in corsivo sopra, si può affermare che se due sistemi sono descritti dagli spazi di Hilbert V e W, allora il sistema complessivo è descritto dal prodotto tensore $V \otimes W$.

Il fenomeno dell'entaglement è una conseguenza diretta di questo assioma. Matematicamente, esso si interpreta come l'esistenza di stati del sistema complessivo che non sono direttamente interpretabili a partire dagli stati delle sue componenti V e W.

Si supponga che i sistemi rappresentati da V e W siano qubit e quindi che $\dim\{V\} = \dim\{W\} = 2$. Si scelga poi una base in V ed una in W, indicandole con $\{|0\rangle, |1\rangle\}$, allora il generico vettore dato dal prodotto tensoriale vale:

$$V \otimes W = \mathbf{p} = c_1 |00\rangle + c_2 |10\rangle + c_3 |01\rangle + c_4 |11\rangle,$$

dove $\mathbf{p}$ è il vettore dato dal prodotto tensoriale tra i due spazi e $|ij\rangle := |i\rangle \otimes |j\rangle$, per $i, j = 0, 1$. Come si può osservare, il generico vettore si ottiene come sovrapposizione di vettori fattorizzabili, ossia corrispondenti ad un elemento in $\mathbf{v} \times W$. Questa proprietà non è però la manifestazione dell'entanglement: si tratta semplicemente della conseguenza dell'assioma della meccanica quantistica che impone di usare gli spazi di Hilbert per rappresentare gli stati di un sistema. Il fenomeno dell'entanglement è qualcosa di diverso. Per il generico vettore $\mathbf{p}$ non è detto che esista un vettore $\mathbf{v} \in V$ ed uno $\mathbf{w} \in W$ che fattorizzano $\mathbf{p}$, ossia per i quali valga $\mathbf{p} = |v\rangle \otimes |w\rangle$. In generale, se nessuno dei due spazi ha dimensione 1, esistono vettori non fattorizzabili per alcuna base e questi vengono chiamati stati entangled.

Per comprendere meglio questo sorprendente fenomeno, si ipotizzi di considerare un sistema composto da soli 2 qubit. Lo stato $|\psi\rangle$ del sistema, risulta:

$$|\psi\rangle = \alpha\,|00\rangle + \beta\,|11\rangle\,, \tag{1.3}$$

considerando naturalmente $\alpha, \beta \neq 0$. Come motivato sopra, non è possibile scrivere il generico stato del sistema come prodotto tensoriale, poiché i due qubit sono entangled, infatti si ha:

$$\begin{aligned}\alpha\,|00\rangle + \beta\,|11\rangle &\neq (a_1\,|0\rangle + b_1\,|1\rangle) \otimes (a_2\,|0\rangle + b_2\,|1\rangle) \\ &= a_1a_2\,|00\rangle + a_1b_2\,|01\rangle + b_1a_2\,|10\rangle + b_1b_2\,|11\rangle\,,\end{aligned}$$

dove a_1 e b_1 sono i coefficienti complessi del primo qubit, mentre a_2 e b_2 quelli del secondo. Particolarizzando poi la (1.3) con gli stessi coefficienti dell'eq.(1.3), si ottiene il seguente stato:

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \tag{1.4}$$

dunque da questo esempio si può osservare che non ci sono singoli stati $|A\rangle$ e $|B\rangle$ dei qubit tali che $|\psi\rangle = |A\rangle \otimes |B\rangle$.

Approfondimento: Stati di Bell Lo stato quantum a due qubit riportato in (1.4) è molto importante ed è conosciuto come Stato di Bell o Coppia EPR, in onore di Einstein, Podolsky e Rosen che ne studiarono le sorprendenti caratteristiche. È uno stato quantum di due qubit che rappresenta l'esempio più semplice di entanglement.

Come è intuibile, lo stato di Bell ha la proprietà che, misurando il primo qubit, si possono ottenere due diversi risultati equiprobabili. Se la misurazione del primo qubit dà esito $|0\rangle$, con probabilità $1/2$, si trova lo stato, di post-misurazione, $|\phi'\rangle = |00\rangle$, mentre se l'esito fosse $|1\rangle$, anch'esso con probabilità $1/2$, si avrebbe lo stato $|\phi'\rangle = |11\rangle$, dopo la misurazione. Di conseguenza, una misurazione del secondo qubit dà sempre lo stesso risultato della misurazione del primo qubit. Cioè, i risultati della misurazione sono massimamente correlati; il fenomeno dell'entanglement è al suo culmine.

In particolare, in base a quanto detto, esistono quattro diverse configurazioni degli stati di Bell, cioè:

$$\begin{aligned}\left|\phi^+\right\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \quad \left|\psi^+\right\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}}; \\ \left|\phi^-\right\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}}, \quad \left|\psi^-\right\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}},\end{aligned}$$

dove la notazione $|\phi^\pm\rangle$ e $|\psi^\pm\rangle$ è utilizzata ad hoc proprio per indicare una di queste particolari tipologie [1]. In particolare, il segno $-$ sta ad indicare un cambiamento di fase nello stato del qubit (o dei qubit, come in questo caso).

I sistemi di comunicazione basati su tecnologia quantum supportano la trasmissione di informazioni sia classiche che quantum. Quando l'informazione da trasmettere è classica, si possono utilizzare tecniche tradizionali

di correzione degli errori per contrastare l'impatto delle alterazioni introdotte dal canale quantum [3], [4]. In particolare, all'inizio, le informazioni classiche vengono codificate utilizzando un codice di correzione d'errore tradizionale. I bit così codificati, poi, vengono mappati sui qubit, che infine sono trasmessi su un canale di comunicazione quantum. La mappatura dei bit classici in qubit può essere effettuata, ad esempio, dal cosiddetto protocollo di codifica super-denso [1], [3], [5].

Al contrario, per un sistema di comunicazione più generale, cioè che supporta la trasmissione di informazioni sia classiche che quantum, per un calcolo affidabile, bisogna ricorrere ai Quantum Error Correction Codes (QECCs). In particolare, proprio come i codici di correzione degli errori tradizionali, i QECCs sfruttano la ridondanza, questa volta nel dominio quantum, consentendo quindi ai qubit di mantenere con alta probabilità i loro stati inalterati e per periodi più lunghi. I QECCs sono, pertanto, indispensabili per concepire un sistema di comunicazione quantum che supporti la trasmissione dell'informazione ed il calcolo.

1.2 Decoerenza quantistica

La teoria della decoerenza quantistica, o desincronizzazione della funzione d'onda, prevece che l'interazione fra i sistemi quantum e l'ambiente esterno determini la perdita della coerenza della funzione d'onda e ciò si traduce, dal punto di vista della trattazione in oggetto, in perdita di informazione.

La decoerenza dovuta all'ambiente costituisce generalmente una fonte importante di alterazioni; in particolare tale fenomeno impedisce l'osservazione della superposition di stati per i sistemi macroscopici. Il carattere destabilizzante dell'ambiente riguarda, di fatto, tutto ciò che può influenzare lo stato del sistema quantum (e quindi può inavvertitamente "misurarlo"), per esempio: un singolo fotone, la vibrazione di una molecola, le particelle dell'aria. In questa teoria l'ambiente non è semplice rumore: esso si comporta come uno strumento che osserva costantemente il sistema; infatti esso non può distinguere tra il contatto casuale e il contatto intenzionale di una misura. Il sistema perde così la coerenza perché lascia sfuggire informazione e quindi rivela al resto dell'universo il suo stato. In termini semplici, la decoerenza ambientale può essere descritta come l'interazione indesiderata, o più specificamente il coinvolgimento, del qubit con parte dell'ambiente, che altera la superposition coerente degli stati di base.

In questa sezione, si esaminano i seguenti canali quantum:
− canale di Pauli;
− canale depolarizzante (o di depolarizzazione),
che sono ampiamente utilizzati per la modellazione della decoerenza ambientale. È noto che esistono anche altri tipi di canali, come quelli denominati "amplitude damping" o "phase damping", che però non saranno presi in esame nel presente lavoro di tesi, in quanto non essenziali in questo contesto. La Fig. 1.4 mostra il legame tra i canali citati.

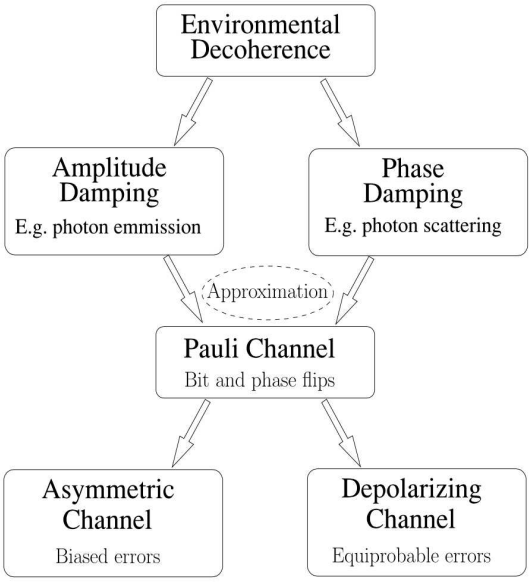

Figura 1.4: Modelli di canale quantum [2].

In particolare, nel seguito si indaga la dualità tra i canali quantum e quelli classici.

1.2.1 Canale di Pauli

Non è possibile simulare classicamente il canale quantum per un sistema composto da N qubit, poiché il sistema risultante avrebbe uno spazio di Hilbert di dimensione 2^N che sarebbe eccessivamente complesso. Pertanto, a causa di questo limite matematico, per facilitare la trattazione, si utilizza il cosiddetto *canale di Pauli*, conosciuto con questo nome in onore del fisico elvetico Wolfgang Pauli, pioniere della meccanica quantistica.

Questo modello di canale viene studiato mediante l'utilizzo degli operatori (chiamati anche matrici o porte) di Pauli a singolo qubit, che sono definiti come segue:

$$\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \mathbf{Y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}.$$

In particolare, $\mathbf{X}$ modifica lo stato del generico qubit realizzando un'operazione di bit-flip (inversione del qubit), $\mathbf{Z}$ opera un phase-flip (inversione della fase) e $\mathbf{Y}$ agisce realizzando sia un bit-flip che un phase-flip. La matrice $\mathbf{I}$ (matrice identità), invece non cambia lo stato del qubit. La Fig. 1.5 illustra l'effetto di questi operatori sulla sfera di Bloch. Si faccia poi riferimento allo schematico di Fig. 1.6 per comprendere il funzionamento di ognuna di queste matrici, a livello analitico e circuitale come porte logiche.

Approfondimento: Operatori di Pauli In meccanica quantistica, le matrici di Pauli sono un insieme di matrici 2×2 complesse hermitiane uni-

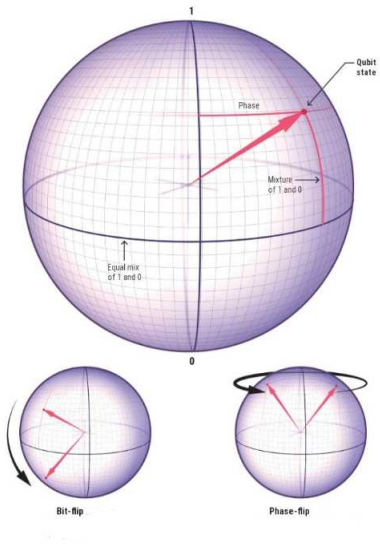

Figura 1.5: Generico stato del qubit, operazione di bit-flip e phase-flip sulla sfera di Bloch. *Bit-flip*: scambia lo stato $|0\rangle$ con lo stato $|1\rangle$; cambia la latitudine del qubit passando dall'emisfero nord a quello sud della sfera. *Phase-flip*: lo stato del qubit ruota di mezzo giro rispetto alla longitudine della sfera.

tarie, cioè delle matrici autoaggiunte che soddisfano la seguente relazione $\mathbf{A}^\dagger\mathbf{A} = \mathbf{A}\mathbf{A}^\dagger = \mathbf{I}$, dove $\mathbf{A}$ è una generica matrice ed $\mathbf{I}$ è la matrice identità.

Prima di procedere oltre è necessario spiegare il significato dell'operatore $(\cdot)^\dagger$. Data una matrice $\mathbf{A}$ ed indicando con $\mathbf{A}^T$ la sua trasposta e con l'asterisco $*$ l'operazione di coniugazione complessa di tutti i suoi elementi, la trasposta coniugata $\mathbf{A}^\dagger$ è data da:

$$\mathbf{A}^\dagger = (\mathbf{A}^T)^* = (\mathbf{A}^*)^T.$$

Inoltre in meccanica quantistica, l'*aggiunto* di una generica matrice $\mathbf{A}$ è definito come:

$$\langle \phi \,|\, \mathbf{A}\psi \rangle = \left\langle \mathbf{A}^\dagger \phi \,\middle|\, \psi \right\rangle,$$

per tutti i vettori $|\psi\rangle$ e $|\phi\rangle$, dove $\langle \cdot | \cdot \rangle$ indica il prodotto interno come già ricordato in precedenza. Dunque, $\mathbf{A}$ risulta un operatore *auto-aggiunto* (self-adjoint) se vale: $\mathbf{A} = \mathbf{A}^\dagger$. Un operatore auto-aggiunto si può, inoltre, esprimere come:

$$\mathbf{A} = \sum_{i=1}^{n} a_i P_i,$$

dove a_i è l'i-esimo autovalore di $\mathbf{A}$ e P_i è la corrispondente proiezione ortogonale sullo spazio degli autovettori con autovalore a_i. Per avere una migliore comprensione degli operatori e della notazione della meccanica quantistica, si faccia riferimento alla Tabella 1.1.

Tipicamente, in letteratura le matrici di Pauli possono trovarsi anche indicate dalla lettera greca σ, valendo le seguenti relazioni:

$$\sigma_0 \equiv \mathbf{I}, \quad \sigma_1 \equiv \sigma_x \equiv \mathbf{X}, \quad \sigma_2 \equiv \sigma_y \equiv \mathbf{Y}, \quad \sigma_3 \equiv \sigma_z \equiv \mathbf{Z}.$$

Tabella 1.1: Notazione degli operatori della meccanica quantistica già introdotti [1].

Notazione	Descrizione
z^*	Complesso coniugato del numero complesso z.
	Vale $(1+i)^* = 1-i$, dove come è noto vale $i = \sqrt{-1}$.
$\|\psi\rangle$	Vettore, conosciuto come *ket*.
$\langle\psi\|$	Vettore duale a $\|\psi\rangle$, conosciuto come *bra*.
$\langle\phi\|\psi\rangle$	Prodotto interno (inner product) tra $\|\phi\rangle$ e $\|\psi\rangle$.
$\|\phi\rangle \otimes \|\psi\rangle$	Prodotto tensoriale (outer product) tra $\|\phi\rangle$ e $\|\psi\rangle$.
$\|\phi\rangle\ \|\psi\rangle$	Notazione abbreviata che indica il prodotto tensoriale tra $\|\phi\rangle$ e $\|\psi\rangle$.
$\mathbf{A}^*$	Complesso coniugato della matrice $\mathbf{A}$.
$\mathbf{A}^T$	Trasposto della matrice $\mathbf{A}$.
$\mathbf{A}^\dagger$	Hermitiano coniugato o aggiunto della matrice $\mathbf{A}$, $\mathbf{A}^\dagger = (\mathbf{A}^T)^*$.
	Vale $\begin{pmatrix} a & b \\ c & d \end{pmatrix}^\dagger = \begin{pmatrix} a^* & c^* \\ b^* & d^* \end{pmatrix}$.
$\langle\phi\|\mathbf{A}\|\psi\rangle$	Prodotto interno tra $\|\phi\rangle$ e $\mathbf{A}\|\psi\rangle$.
	Equivalentemente, si può assumere come il prodotto interno tra $\mathbf{A}^*\|\phi\rangle$ e $\|\psi\rangle$.

Queste matrici,soddisfano anche le seguenti proprietà:

$$\sigma_0^2 = \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbf{I}$$

e

$$\begin{aligned} \sigma_1\sigma_2 &= i\sigma_3, \quad \sigma_2\sigma_1 = i\sigma_3; \\ \sigma_2\sigma_3 &= i\sigma_1, \quad \sigma_3\sigma_2 = i\sigma_1; \\ \sigma_3\sigma_1 &= i\sigma_2, \quad \sigma_1\sigma_3 = i\sigma_2. \end{aligned}$$

Inoltre, determinante e traccia di queste matrici risultano:

$$\det(\sigma_i) = -1, \quad \mathrm{tr}(\sigma_i) = 0, \quad \forall i = 1, 2, 3,$$

di conseguenza si ricava semplicemente che gli autovalori delle tre matrici di Pauli sono ± 1.

Infine $\sigma_1, \sigma_2, \sigma_3$, con l'aggiunta dell'identità, formano un insieme completo di matrici, ovvero una base B dello spazio delle matrici 2×2 hermitiane. Si ottiene:

$$B = c_0\sigma_0 + c_1\sigma_1 + c_2\sigma_2 + c_3\sigma_3,$$

dove c_0, c_1, c_2, c_3 sono i coefficienti complessi.

In un canale di Pauli dunque, si hanno probabilità indipendenti p_x, p_y e p_z, tali che $p_x + p_y + p_z \leq 1$, per cui un qubit di input sia sottoposto ad un errore di Pauli $\mathbf{X}$, $\mathbf{Y}$ o $\mathbf{Z}$, rispettivamente.

Si analizza il funzionamento di ogni gate di Pauli che agisce sullo stato $|\psi\rangle$ di un generico qubit.

Cominciando dall'operatore di identità $\mathbf{I}$, chiamato anche, più semplicemente, gate di ripetizione, si osserva che esso lascia intatto lo stato $|\psi\rangle$,

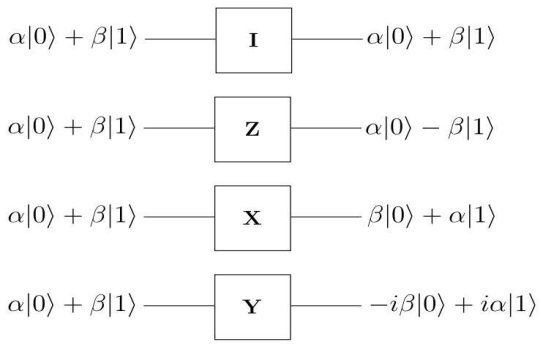

Figura 1.6: Schematico delle 4 porte logiche di Pauli [2].

come mostrato di seguito:

$$\mathbf{I}\,|\psi\rangle = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \equiv \alpha\,|0\rangle + \beta\,|1\rangle\,.$$

L'operatore **Z** invece, porta ad una inversione di fase (phase-flip) ed agisce come segue:

$$\mathbf{Z}\,|\psi\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha \\ -\beta \end{pmatrix} \equiv \alpha\,|0\rangle - \beta\,|1\rangle\,,$$

infatti, rispetto a quanto si verifica con la matrice identità **I**, in questo caso l'addendo $\beta\,|1\rangle$ risulta cambiato di segno, in ragione del $-$, che compare nell'elemento $z_{2,2}$ della matrice **Z**.

L'operatore **X**, diversamente, è un operatore di inversione di bit (bit-flip), risulta dunque analogo alla porta logica "NOT" tradizionale. Pertanto si ottiene:

$$\mathbf{X}\,|\psi\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix} \equiv \beta\,|0\rangle + \alpha\,|1\rangle\,,$$

come si può notare, infatti, il coefficiente β ora è associato allo stato $|0\rangle$ invece che a $|1\rangle$ e di conseguenza α moltiplica $|1\rangle$ al posto di $|0\rangle$.

Infine, **Y** si ricava dal prodotto tra l'operatore **X** e **Z**, premoltiplicati per l'unità immaginaria i (in particolare, vale: $\mathbf{Y} = i\mathbf{XZ}$). Dunque, **Y** è dato dalla combinazione tra un operatore di inversione di bit ed un operatore di inversione di fase, per questo agisce su $|\psi\rangle$ nel seguente modo:

$$\mathbf{Y}\,|\psi\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} -i\beta \\ i\alpha \end{pmatrix} \equiv -i(\beta\,|0\rangle - \alpha\,|1\rangle).$$

Pertanto, la matrice **Y** viene chiamata operatore di inversione di bit e di fase; è responsabile appunto del fenomeno del bit-and-phase-flip.

Si può notare che il canale di Pauli, durante il processo di decoerenza, effettivamente mappa lo stato di ingresso ($|\psi\rangle$) su di una combinazione lineare dello stato originale (operatore **I**), dello stato di inversione di fase

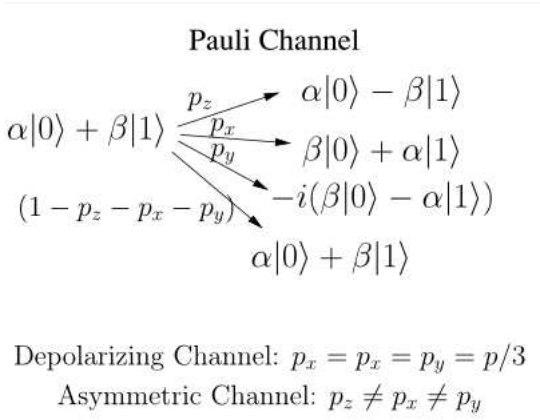

Figura 1.7: Interpretazione dei principali modelli dei canali quantum argomentati [2].

(operatore **Z**), dello stato di inversione di bit (operatore **X**) ed infine dello stato di inversione di bit e di fase (operatore **Y**), pesati in base al valore delle probabilità p_z, p_x e p_y.

Inoltre, una speciale categoria di canale di Pauli, conosciuta come *canale di depolarizzazione* (depolarizing channel) e indicato con N_{DP}, modella lo scenario peggiore ipotizzando che tutti e tre gli errori siano equiprobabili, cioè $p_z = p_x = p_y = p/3$. In particolare, un canale depolarizzante è caratterizzato da una probabilità, detta di *depolarizzazione*, pari a $p/3$ di causare un errore di inversione di fase (operatore di Pauli **Z**) o di inversione di bit (operatore di Pauli **X**) o di inversione di bit e di fase (operatore di Pauli **Y**). Dunque si può affermare che con probabilità $1 - p$ lo stato del qubit rimane immutato, mentre con probabilità di depolarizzazione pari a p si verifica sicuramente uno dei tre possibili errori. I suddetti modelli di canali quantum sono riassunti nella Fig. 1.7.

Si può osservare che, producendo quattro possibili uscite, il canale di Pauli è considerato l'analogo quantum del canale classico quaternario discreto. Tuttavia, mentre questo può determinare solo uno dei quattro possibili errori, l'errore causato dal canale di Pauli può essere interpretato come la superposition dei quattro possibili errori, vale a dire, in base agli operatori, **I**, **Z**, **X** e **Y**.

Il canale di Pauli può essere ulteriormente semplificato utilizzando due canali indipendenti: un canale di inversione del bit ed un canale di inversione della fase, che sono analoghi ai canali classici binari simmetrici, con probabilità di cross-over pari a $(p_x + p_y)$ e $(p_z + p_y)$, rispettivamente. In particolare, il canale classico binario simmetrico, riportato in Fig. 1.8, è caratterizzato da una sorgente binaria che emette solo due possibili simboli; per esempio il bit 0 ed il bit 1. Una volta emesso il simbolo, questo può arrivare a destinazione invariato oppure subire una distorsione e commutare da 0 a 1 o viceversa, con probabilità di cross-over pari a p, a causa della possibile insorgenza di errori. Pertanto p è la probabilità di errore, mentre la probabilità che il simbolo giunga corretto lato ricevitore è pari a $1 - p$.

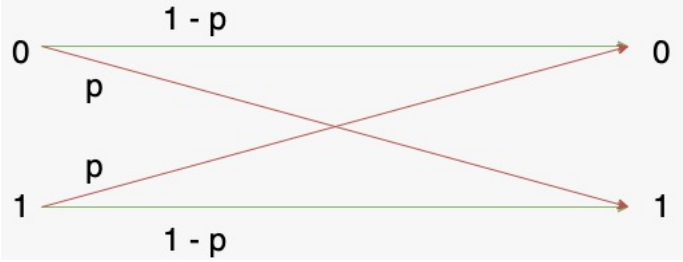

Figura 1.8: Canale binario simmetrico.

Dunque, nel caso del canale di bit-flip, la probabilità di incorrere in un errore è $(p_x + p_y)$, che è la somma della probabilità di inversione del bit e di inversione del bit e della fase del canale di Pauli. Analogamente, per il canale di phase-flip calato nel dominio classico, la probabilità di errore coincide con $(p_z + p_y)$, cioè la somma della probabilità di phase-flip e di bit-and-phase-flip. In entrambi i casi è, giocoforza, presente p_y, che contiene sia il caso dell'inversione del bit che della fase e quindi può verificarsi in entrambi i modelli di canale.

1.2.2 Threshold Theorem

La presenza degli errori, dovuta al rumore, in un sistema quantum non è di per sé un ostacolo invalicabile per giungere ad una condizione di calcolo efficiente. L'apparato teorico che giustifica quanto affermato è il threshold theorem (o quantum fault-tolerance theorem), che segue.

Teorema

È dato un circuito quantum contenente $f(n)$ gate, dove n è la dimensione del problema che deve essere risolto dal circuito e $f(n)$ è una funzione polinomiale di n. Date le ipotesi ragionevoli sul rumore nell'hardware sottostante, tale circuito può essere simulato con probabilità di errore circa ϵ, utilizzando al massimo

$$O(\text{poly}(\log(p(n))/\epsilon)p(n))$$

gate. Queste porte logiche saranno costituite infatti da componenti che falliscono con probabilità al massimo pari a p, a condizione che p sia al di sotto di una certa soglia (threshold) costante, con $p < p_{th}$ [1].

Questo teorema spiega come un quantum computer con un rate di errore arbitrariamente basso possa essere costruito anche a partire da parti difettose, nel senso di rumorose. Come è evidente, questo risultato fa eco al teorema di codifica di canale rumoroso di Shannon (noisy channel coding theorem); infatti, anche in questo caso, viene esplicitamente affermato che utilizzando codici per la correzione di errore è possibile raggiungere affidabilità nella comunicazione.

1.3 Transizione dal dominio classico al dominio quantum

Le leggi della meccanica quantistica rendono le tecniche di codifica nel dominio quantum intrinsecamente diverse dalle loro controparti classiche. Tuttavia, possono essere progettati, a partire dalle famiglie di codici classici già esistenti, codici quantum efficienti, se si tengono in particolare considerazione le tre seguenti problematiche, che per naturali ragioni strutturali, non hanno riscontro nel dominio classico:

1) Teorema di non clonazione;
2) Operazione di misura del qubit;
3) Natura degli errori quantum.

1.3.1 Teorema di non clonazione

La maggior parte dei codici di correzione degli errori classici si basa su tecniche che combinano tra loro i bit. In particolare una classe di codici a blocco, quella dei codici a ripetizione, clona i bit stessi, creando copie multiple del bit di informazione per fornire ridondanza. Sfortunatamente, nel dominio quantum, il teorema di non clonazione spiega che non esiste un operatore unitario che permette di copiare un qubit che si trova in uno stato arbitrario [6]. I codici quantum, allora, sfruttano la ridondanza senza clonare i qubit che portano l' informazione.

La soluzione consiste nel clonare solo gli stati base $|0\rangle$ e $|1\rangle$, chiamati anche basi computazionali, piuttosto che lo stato sovrapposto $|\psi\rangle$. In particolare i qubit ausiliari, chiamati così poiché servono per creare la ridondanza, sono entangled con il qubit di informazione $|\psi\rangle$. Il codificatore allora può replicare q volte, in base alla ridondanza desiderata per la struttura del codice, gli stati $|0\rangle$ e $|1\rangle$ nell'output codificato $\left|\overline{\psi}\right\rangle$. Nel caso di $q = 2$, $\left|\overline{\psi}\right\rangle$ è dato da:

$$\begin{aligned} |\psi\rangle \otimes |0\rangle^{\otimes 2} \rightarrow \left|\overline{\psi}\right\rangle &= \alpha\,|\bar{0}\rangle + \beta\,|\bar{1}\rangle \\ &\equiv \alpha\,|000\rangle + \beta\,|111\rangle\,, \end{aligned}$$

dove l'apice $^{\otimes 2}$ indica che lo stato $|0\rangle$ del qubit è stato replicato due volte. La freccia "$\rightarrow$" rappresenta l'operazione di codifica; a sinistra si trova lo stato sovrapposto prima della codifica e a destra quello codificato.

1.3.2 Operazione di misura del qubit

I codici classici si basano sulla misurazione dei valori dei bit ricevuti, attraverso l'azione del decodificatore. Sfortunatamente, non è possibile misurare un qubit senza perturbarlo, il che comporterebbe inevitabilmente, al momento della misurazione, il collasso degli stati quantum sovrapposti e quindi la perdita di informazione.

I codici quantum, quindi, devono stimare gli errori imposti dal canale senza misurare i qubit ricevuti. Nel dominio classico, lato ricevitore, il

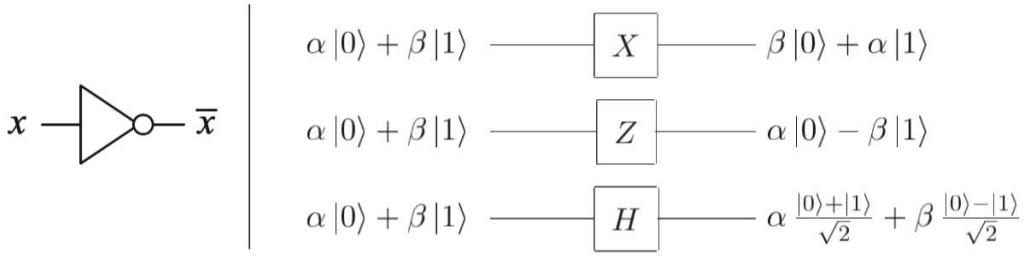

Figura 1.9: Confronto tra la porta logica "NOT" a singolo bit (sinistra) e gli operatori di Pauli e di Hadamard: **X** (bit-flip), **Z** (phase-flip) e $\mathbf{H}_H$ [1].

decodificatore di un codice legge i bit ricevuti e li decodifica. In particolare, se il qubit ricevuto è caratterizzato dal generico stato $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ e viene misurato, esso collasserà, come visto, nello stato $|0\rangle$ o $|1\rangle$ con una probabilità di $|\alpha|^2$ o $|\beta|^2$, rispettivamente.

1.3.3 Natura degli errori quantum

In accordo con il modello illustrato nella Sezione 1.2, i codici quantum devono essere in grado di correggere gli errori di tipo bit-flip, phase-flip e bit-and-phase-flip. Nel dominio classico, quando le sequenze di bit codificate vengono trasmesse, uno 0 può essere invertito in un 1 e un 1 può essere invertito in uno 0. Di conseguenza, il canale di comunicazione classico impone, alle parole di codice trasmesse, solo errori discreti di bit-flip.

Al contrario, quando un qubit viene trasmesso, per esempio su un canale di depolarizzazione, può verificarsi un errore di inversione di bit, di inversione di fase e di inversione di bit e di fase, con la stessa probabilità $p/3$, come discusso nella Sezione 1.2.1. In particolare, l'operazione di bit-flip capovolge la base computazionale $\{|0\rangle, |1\rangle\}$, mentre il phase-flip cambia il segno della somma degli stati. Per operare con le basi computazionali, si usa la matrice di Hadamard $\mathbf{H}_H$ (Hadamard gate, conosciuto anche come operatore di Hadamard) a singolo qubit, che si comporta come segue:

$$|+\rangle \equiv \mathbf{H}_H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}; \quad |-\rangle \equiv \mathbf{H}_H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}},$$

dove $\{|+\rangle, |-\rangle\}$ è la base di Hadamard e $\mathbf{H}_H$ è specificato dalla seguente matrice [1]:

$$\mathbf{H}_H = \frac{1}{\sqrt{2}}(\mathbf{Z} + \mathbf{X}) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Approfondimento: Operatore di Hadamard Questo gate è talvolta descritto come una "radice quadrata della porta logica NOT", in quanto trasforma lo stato $|0\rangle$ in $(|0\rangle + |1\rangle)/\sqrt{2}$ (prima colonna di $\mathbf{H}_H$), cioè uno stato intermedio tra $|0\rangle$ e $|1\rangle$, e trasforma $|1\rangle$ in $(|0\rangle - |1\rangle)/\sqrt{2}$ (seconda colonna

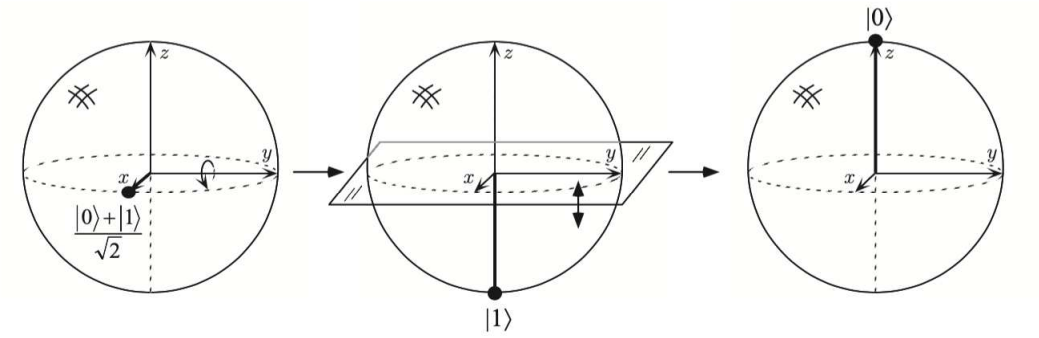

Figura 1.10: Visualizzazione del gate di Hadamard sulla sfera di Bloch, dato lo stato di ingresso $(|0\rangle + |1\rangle)/\sqrt{2}$ [1].

di $\mathbf{H}_H$), che è anch'esso "a metà strada" tra $|0\rangle$ e $|1\rangle$. A tal proposito, alcune importanti porte a singolo qubit sono mostrate in Fig. 1.9 e messe a confronto con il "NOT" classico [1].

Si noti, tuttavia, che $\mathbf{H}_H^2 = \mathbf{I}$, per cui si conclude che applicando $\mathbf{H}_H$ due volte ad un certo stato non lo si modifica. Il gate di Hadamard è una delle porte più utili e vale la pena provare a visualizzarne il funzionamento con la sfera di Bloch. Tale operatore corrisponde infatti ad una rotazione della sfera intorno all'asse y di 90°, seguita da una rotazione intorno all'asse x di 180°, come illustrato nella Fig. 1.10 [1].

Di conseguenza, un errore di phase-flip (operatore di Pauli $\mathbf{Z}$) commuta la base di Hadamard come segue:

$$\mathbf{Z}\,|+\rangle = |-\rangle\,;$$
$$\mathbf{Z}\,|-\rangle = |+\rangle\,,$$

mentre un errore di bit-flip (operatore di Pauli $\mathbf{X}$) agisce cambiando la base computazionale. Si ottiene infatti:

$$\mathbf{X}\,|0\rangle = |1\rangle\,;$$
$$\mathbf{X}\,|1\rangle = |0\rangle\,.$$

1.3.4 Conclusioni

Per riassumere quanto detto finora, si può guardare la Fig. 1.11. Essa rappresenta la transizione dal dominio classico a quello quantum dei codici per la correzione degli errori [7]. Il problema viene scomposto in tre parti, in ragione dei tre macro blocchi che compongono ogni sistema di comunicazione: il codificatore, il canale ed il decodificatore. Per ognuno di questi elementi viene proposto un confronto tra il dominio classico e quello quantum.

Codificatore: i codificatori classici possono copiare i bit d'informazione. Purtroppo, come detto, non esiste alcun operatore di clonazione quantum. Di conseguenza, nei codici quantum i qubit d'informazione sono

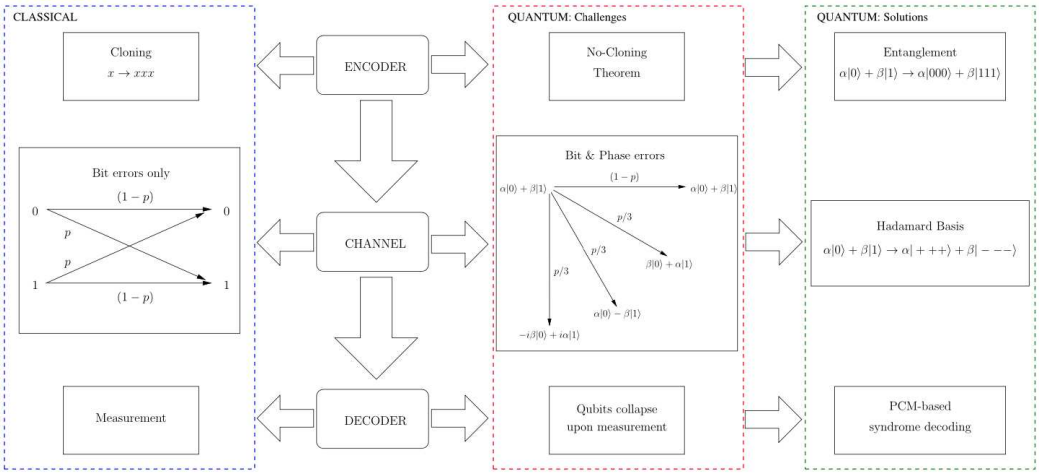

Figura 1.11: Transizione dal dominio classico a quello quantum dei codici per la correzione degli errori. Si ricorda ancora al lettore il significato delle notazioni riportate: $|+++\rangle \equiv |+\rangle \otimes |+\rangle \otimes |+\rangle$ e $|---\rangle \equiv |-\rangle \otimes |-\rangle \otimes |-\rangle$ [2].

entangled con i qubit ausiliari, in modo che le informazioni vengano clonate negli stati di base.

Canale: le informazioni classiche possono riscontrare solo errori di tipo bit-flip, mentre i qubit possono sperimentare errori di tipo bit-flip, phase-flip e quindi anche bit-and-phase-flip. Gli errori aggiuntivi di inversione di fase, nel dominio quantum, possono essere corretti usando la matrice di Hadamard $\{|+\rangle, |-\rangle\}$.

Decodificatore: i decodificatori classici misurano i bit ricevuti per stimare le informazioni trasmesse. Sfortunatamente, i qubit non possono essere misurati senza perturbare il loro stato di sovrapposizione. Pertanto, i codici quantum utilizzano la decodifica della sindrome basata sulla Parity Check Matrix (PCM), cioè valutano i pattern di errore, introdotti dal canale, senza in realtà osservare i qubit ricevuti.

Capitolo 2

Teoria della correzione dell'errore classica e quantum

All'inizio di questo capitolo vengono illustrati i principi generali della teoria della correzione d'errore classica e quantum, con particolare attenzione alle differenze tra queste. Successivamente, la sezione sul formalismo stabilizzatore permette di creare un equivalente quantum dei codici a blocco classici. L'isomorfismo dal dominio di Pauli al dominio classico invece consente di traslare una costruzione quantum basata sugli stabilizzatori nel mondo classico. Infine, l'ultima sezione traccia una classificazione dei principali modelli di codice quantum.

2.1 Codici Classici

La teoria classica della correzione degli errori mediante codici vanta molte applicazioni diverse e importanti, caratterizzate da un livello maturo della tecnologia. Molte tecniche di correzione degli errori classiche hanno importanti implicazioni per la correzione degli errori quantum. L'obiettivo di questa sezione è quello di spiegare, in particolare, la teoria dei codici lineari classici, poiché può essere utilizzata per sviluppare una vasta gamma di buoni codici per la correzione degli errori quantum.

Sia C un codice a blocco che codifica k bit di informazione in una parola di codice di n bit, dove $n = k+m$ e m rappresenta la ridondanza del codice. Tale codice si dice a blocco lineare se e solo se le sue q^k parole di codice formano un sottospazio vettoriale U di dimensione k dello spazio vettoriale V, che comprende tutte le N-uple q-arie. In particolare, d'ora in avanti si farà riferimento al caso binario ($q = 2$). Si dice che un codice che utilizza n bit per codificare k bit di informazione è un codice $C(n, k)$. La dimensione dello spazio vettoriale U è specificata dal numero di vettori linearmente indipendenti nella base; pertanto in U ci sono k vettori (parole di codice) linearmente indipendenti $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1}$ che formano la base da cui si possono generare tutte le altre.

Lo spazio vettoriale $U \subset V$ è chiamato spazio di codice ed è specificato da una matrice generatrice di $\mathbf{G}$ di dimensioni $k \times n$, le cui righe sono proprio

i k vettori di cui sopra ed è composta da elementi binari (zeri e uni). La matrice $\mathbf{G}$ permette di codificare le sequenze di informazione (messaggi) in parole di codice; il messaggio di k bit $\mathbf{u}$ è codificato come

$$\mathbf{v} = \mathbf{u}\mathbf{G},$$

dove il messaggio $\mathbf{u}$ e la parola di codice $\mathbf{v}$ sono trattati come vettori riga, di dimensioni $1 \times k$ e $1 \times n$, rispettivamente. Inoltre, l'operazione di moltiplicazione tra vettori (matrici) e tutte le altre operazioni aritmetiche in questa sezione, sono fatte modulo 2. Un codice lineare a blocco di solito ha più di una base, quindi la matrice generatrice $\mathbf{G}$ non è unica.

Un semplice esempio di codice lineare a blocco è il codice a ripetizione ($k = 1$ e $n = 3$), il quale mappa un singolo bit su una parola di codice di tre bit, ripetendolo tre volte. È specificato dalla seguente matrice generatrice

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \tag{2.1}$$

dove $\mathbf{G}$ mappa i possibili messaggi $\mathbf{m}_0 = (0)$ e $\mathbf{m}_1 = (1)$ nella loro forma codificata, $\mathbf{v}_0 = \mathbf{G}\mathbf{m}_0 = (0, 0, 0)$ e $\mathbf{v}_1 = \mathbf{G}\mathbf{m}_1 = (1, 1, 1)$: questo esempio è un codice $C(3, 1)$.

Un grande vantaggio dei codici lineari, rispetto ai codici generici di correzione degli errori, è la loro specifica compattezza. Un codice generale che codifica i k bit in n bit richiede 2^k parole di codice, ciascuna di lunghezza n, cioè un totale di $n2^k$ bit per specificare una descrizione del codice. Con un codice lineare si devono solo specificare i kn bit della matrice generatrice, un risparmio esponenziale nella quantità di memoria richiesta, infatti tale operazione ha costo computazionale di $O(nk)$.

Questa descrizione compatta si riflette nella capacità di effettuare una codifica e una decodifica efficienti, importanti caratteristiche che i codici lineari classici condividono con i loro cugini quantum, in particolare utilizzando il design dei codici stabilizzatori.

Lo spazio nullo del codice lineare a blocco $C(n, k)$, chiamato C_d, è un sottospazio $(n - k)$-dimensionale di V, dato da
$C_d = \{\mathbf{w} \in V \,|\, \mathbf{w} \cdot \mathbf{v} = \mathbf{0}, \forall\, \mathbf{v} \in C\}$. C_d è un codice $(n, n - k)$ e si dice duale di C. Date poi $\mathbf{h}_0, \ldots, \mathbf{h}_{n-k-1}$ parole di codice linearmente indipendenti che forniscono una base del codice C_d, si definisce la matrice generatrice $\mathbf{H}$ del codice C_d che ha tali parole come righe, quindi dimensione $(n - k) \times n$ ed è formata da elementi binari. Il codice duale C_d in letteratura si può trovare anche come $C^{\perp}$.

Poiché vale che $\mathbf{G}\mathbf{H}^T = \mathbf{0}$, allora il codice C può essere rappresentato univocamente anche da $\mathbf{H}$: una n-upla $\mathbf{v} \in V$ è una parola di codice se e solo se vale

$$\mathbf{v}\mathbf{H}^T = \mathbf{0},$$

dove $\mathbf{H}$ si dice matrice di parità (parity check matrix − PCM) del codice C e C è definito come il kernel di $\mathbf{H}$. Quindi, un codice lineare a blocco può avere due rappresentazioni: la matrice generatrice e la matrice di parità.

Si dice inoltre che un codice è debolmente auto-duale (weakly self-dual) se $C \subseteq C^{\perp}$, e strettamente auto duale se $C = C^{\perp}$. Piuttosto sorprendentemente, la costruzione duale dei codici lineari classici viene utilizzata nello studio di tecniche di correzione d'errore nel dominio quantum; questa è la chiave della costruzione di una classe importante di codici quantum conosciuti come codici CSS, che verranno in seguito approfonditi.

Per passare dalla matrice $\mathbf{H}$ alla matrice $\mathbf{G}$, bisogna scegliere k vettori linearmente indipendenti $\mathbf{y}_1, \ldots, \mathbf{y}_k$ che spannano il kernel di $\mathbf{H}$; in questo modo si ottiene $\mathbf{G}$ come matrice che ha le righe da $\mathbf{y}_1$ a $\mathbf{y}_k$. Per passare invece dalla matrice $\mathbf{G}$ alla matrice $\mathbf{H}$, si considerino $n-k$ vettori linearmente indipendenti $\mathbf{y}_1, \ldots, \mathbf{y}_{n-k}$ ortogonali alle righe di $\mathbf{G}$ e quindi si costruisce $\mathbf{H}$ per righe; esse saranno: $\mathbf{y}_1^T, \ldots, \mathbf{y}_{n-k}^T$. Con ortogonale, si intende che il prodotto interno modulo 2 deve essere zero.

Per esempio, si consideri il codice a ripetizione (3, 1) definito dalla matrice $\mathbf{G}$ della (2.1). Per costruire $\mathbf{H}$ si prendono $n - k = 3 - 1 = 2$ vettori linearmente indipendenti ortogonali alle colonne di $\mathbf{G}$, per esempio (1, 1, 0) e (0, 1, 1) e si definisce la matrice di parità come:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

È facile verificare che $\mathbf{v}\mathbf{H}^T = \mathbf{0}$ solo per le parole di codice $\mathbf{v}_1 = (0, 0, 0)$ e $\mathbf{v}_2 = (1, 1, 1)$, cioè le uniche possibili per un codice a ripetizione.

Attraverso la matrice di parità è possibile decodificare la parola di codice ricevuta. Si supponga che si codifica il messaggio $\mathbf{x}$ come $\mathbf{y} = \mathbf{G}\mathbf{x}$, ma un errore $\mathbf{e}$ dovuto al rumore nel canale di comunicazione produce la parola di codice corrotta $\mathbf{y}' = \mathbf{y} + \mathbf{e}$, si noti che + qui denota l'addizione modulo 2. Poiché $\mathbf{y}\mathbf{H}^T = \mathbf{0}$ per tutte le parole di codice, ne consegue che $\mathbf{y}'\mathbf{H}^T = \mathbf{e}\mathbf{H}^T$. Il vettore $\mathbf{y}'\mathbf{H}^T$ viene chiamato *sindrome* oppure *sindrome d'errore* e si indica con $\mathbf{s}$. La sindrome contiene informazioni sull'errore che si è verificato e per questo permette il recupero della parola di codice originale $\mathbf{y}$. Il concetto di sindrome, come sarà più chiaro nel seguito, può essere esteso anche al dominio quantum, dove svolge un ruolo simile a quello spiegato per il mondo classico.

In generale, per comprendere le capacità correttive di un codice lineare, può essere utilizzato il concetto di *distanza*. Supponiamo che $\mathbf{x}$ e $\mathbf{y}$ siano parole di codice di n bit ciascuna. La funzione distanza di Hamming, o semplicemente distanza, è una forma di distanza hard tra $\mathbf{x}$ e $\mathbf{y}$,si indica con $d(\mathbf{x}, \mathbf{y})$, ed è definita come il numero di posizioni in cui $\mathbf{x}$ e $\mathbf{y}$ differiscono. Così $d(((1, 1, 0, 0), (0, 1, 0, 1)) = 2$, per esempio. Invece, il peso di Hamming, o semplicemente peso, di una parola $\mathbf{x}$ è definito come la distanza di questa parola di codice dalla stringa di tutti zeri, $\mathrm{wt}(x) \equiv d(\mathbf{x}, 0)$, cioè il numero delle posizioni in cui $\mathbf{x}$ non è zero. Si noti che $d(\mathbf{x}, \mathbf{y}) = \mathrm{wt}(\mathbf{x} + \mathbf{y})$.

Per capire l'importanza della distanza nell'ambito della correzione degli errori, si supponga di codificare $\mathbf{x}$ come $\mathbf{y} = \mathbf{x}\mathbf{G}$, utilizzando un codice lineare di correzione degli errori. Il rumore corrompe la stringa di bit codificata producendo $\mathbf{y}' = \mathbf{y} + \mathbf{e}$. Se la probabilità di un bit flip è inferiore a 1/2,

la parola di codice che deve essere decodificata con maggiore probabilità è la parola $\mathbf{y}$ che minimizza il numero di bit flip necessari per passare da $\mathbf{y}$ a $\mathbf{y}$', cioè che minimizza $\mathrm{wt}(\mathbf{e}) = d(\mathbf{y}, \mathbf{y}')$. In pratica ciò può risultare piuttosto inefficiente, poiché determinare la distanza minima $d(\mathbf{y}, \mathbf{y}')$ in generale richiede la ricerca di tutte le 2^k possibili parole di codice $\mathbf{y}$. Negli ultimi decenni i ricercatori si sono impegnati moltissimo per costruire codici con una struttura speciale che consenta di eseguire più efficacemente la correzione degli errori e i codici LDPC ne sono un esempio.

Le proprietà globali del codice possono essere comprese utilizzando la distanza di Hamming. Definiamo la *distanza minima di un codice* come la minima distanza di Hamming tra due parole qualsiasi dello stesso codice

$$d(C) \equiv \min_{\mathbf{x},\mathbf{y} \in C, \mathbf{x} \neq \mathbf{y}} d(\mathbf{x}, \mathbf{y}).$$

Ma $d(\mathbf{x}, \mathbf{y}) = \mathrm{wt}(\mathbf{x} + \mathbf{y})$. Poiché il codice è lineare, $\mathbf{x} + \mathbf{y}$ è ancora una parola di codice, visto che $\mathbf{x}$ e $\mathbf{y}$ lo sono, si osserva che

$$d(C) = \min_{\mathbf{x} \in C, \mathbf{x} = \mathbf{0}} \mathrm{wt}(\mathbf{x}).$$

Impostando $d \equiv d(C)$, per definire il codice C si può usare la più completa notazione $C[n, k, d]$. Saper definire la distanza di un codice è importante perché permette di stabilirne le capacità correttive, infatti un codice con una distanza di almeno $2t+1$ per qualche numero intero t è in grado di correggere fino a t errori su bit, semplicemente decodificando il messaggio codificato corrotto $\mathbf{y}'$ come l'unica parola di codice $\mathbf{y}$ che soddisfa $d(\mathbf{y}, \mathbf{y}') \leq t$.

Esempio: Codice di Hamming Dal punto di vista didattico, una buona classe di codici lineari correttori di errore sono i codici di Hamming. Supponiamo che $r \geq 2$ sia un numero intero e che $\mathbf{H}$ sia la matrice le cui colonne sono tutte le $2^r - 1$ stringhe di lunghezza r bit non nulle. Questa matrice di parità definisce un codice lineare $C_{\mathrm{H}}[2^r - 1, 2^r - r - 1]$ noto come codice di Hamming.

Un esempio di questo codice, particolarmente importante anche per la correzione degli errori nel dominio quantum, è il caso in cui $r = 3$, che porta ad un codice $C_{\mathrm{H}}[7, 4]$, avente la seguente matrice parità:

$$\mathbf{H} = \begin{bmatrix} 0\,0\,0\,1\,1\,1\,1 \\ 0\,1\,1\,0\,0\,1\,1 \\ 1\,0\,1\,0\,1\,0\,1 \end{bmatrix}. \tag{2.2}$$

Tutte le colonne di $\mathbf{H}$ sono diverse tra loro e la distanza minima del codice è 3. Ne consegue che questo codice è in grado di correggere un errore su qualsiasi bit singolo.

Il metodo di correzione degli errori è molto semplice. Si supponga che si verifichi un errore sul bit j-esimo. Un'ispezione della matrice in (2.2) rivela che la sindrome $\mathbf{s}_j = \mathbf{e}_j \mathbf{H}^T$ è proprio una rappresentazione binaria per j, che spiega quale bit capovolgere per correggere l'errore.

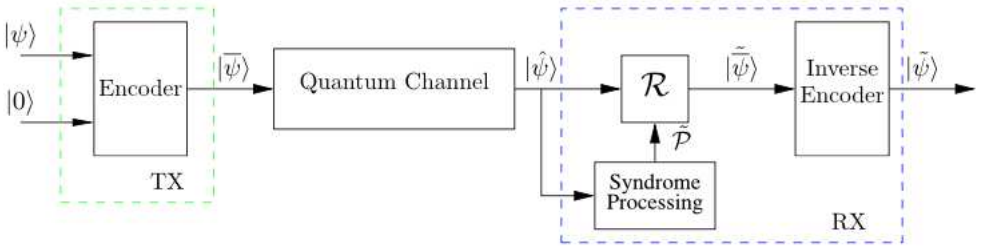

Figura 2.1: Schematico di un sistema di comunicazione quantum che utilizza un QSC per la correzione degli errori [2].

2.1.1 Codici Quantum

In questa sezione sono brevemente riportati i concetti generali relativi ai codici quantum, che poi vengono discussi e approfonditi nel continuo di questo capitolo.

Le idee di base della teoria della correzione d'errore quantum ha l'obiettivo di generalizzare, rispettando i limiti imposti dalla meccanica quantistica, come visto nella Sezione 1.3, le idee introdotte dalla teoria della correzione d'errore classica. Come è noto dal Capitolo 1, nel dominio quantum non si parla di stringhe di simboli, ma si ha a che fare con stati quantum. Pertanto, essi sono codificati da un'operazione unitaria in un codice di correzione degli errori quantum, formalmente definito come un sottospazio C di uno spazio di Hilbert più grande. Già da questo si può osservare la prima grande differenza con il mondo classico.

Invece, per quanto riguarda la decodifica, come per altro brevemente accennato nella Sezione 1.3.4, essa viene effettuata mediante la sindrome, calcolata a partire dalla matrice di parità del codice. Questo è un forte legame con il mondo della correzione d'errore classico ed è l'idea principale su cui si basano le tecniche di decodifica che sono presentate nei capitoli successivi.

2.2 Formalismo Stabilizzatore

La famiglia dei Quantum Stabilizer Code (QSC) si poggia sugli stessi principi di progettazione dei codici a ripetizione. In particolare, i QSC si fondano sulla decodifica della sindrome basata sulla PCM dei codici classici a blocco lineari; quindi, dopo aver trovato l'errore introdotto dal canale, si misurano i qubit della sindrome ausiliaria, invece che osservare i qubit ricevuti. Intuitivamente, il formalismo stabilizzatore [8], [9] può essere interpretato come il duale quantum del paradigma classico di codifica lineare a blocco. Infatti, molti codici classici sfruttano la stessa infrastruttura di base dei codici a blocco lineari. Di conseguenza, il formalismo stabilizzatore fornisce un quadro teorico generale per la progettazione delle versioni quantum dei codici classici noti.

2.2.1 Progettazione di codici tramite il formalismo stabilizzatore

La Fig. 2.1 mostra il modello di un sistema di comunicazione basato su un QSC. Un codice $C(n, k)$ a blocco lineare classico codifica la sequenza di informazione $\mathbf{x}$, costituita da k bit, in una parola di codice $\bar{\mathbf{x}}$, di n simboli, utilizzando $(n - k)$ simboli di parità.

Allo stesso modo, un QSC generico $\mathcal{C}[n, k]$ codifica una parola di informazione $|\psi\rangle$, costituita da k qubit (chiamati qubit *logici*), in una parola di codice $|\overline{\psi}\rangle$, costituita da n qubit (chiamati qubit *fisici*), con l'aiuto di $(n - k)$ qubit ausiliari (noti anche come qubit *ancella*). Si noti che per i codici classici si utilizzano le parentesi tonde perspecificare i parametri del codice, mentre per quelli quantum si utilizzano le quadre.

In particolare, i qubit ausiliari di un QSC sono analoghi ai bit di parità classici. I qubit codificati, che costituiscono la parola di codice $|\overline{\psi}\rangle$, vengono trasmessi nel canale di depolarizzazione (si veda a tal proposito la Sezione 1.2.1), che è caratterizzato dal vettore di errore $\mathcal{P}$, considerando il caso generale ad n qubit. L'output potenzialmente affetto da errori, in uscita dal canale, che si indica con $|\hat{\psi}\rangle$, può quindi essere espresso come:

$$|\hat{\psi}\rangle = \mathcal{P}\,|\overline{\psi}\rangle\,.$$

Il decodificatore di un QSC svolge un processo suddiviso in tre step per correggere gli errori di trasmissione, che include: l'elaborazione della sindrome, il recupero degli errori ($\mathcal{R}$) e la fase di codifica inversa.

Si ricorda poi che la PCM di un codice classico a blocco lineare è progettata in modo da produrre un vettore di sindrome composto da tutti zeri (all-zero) per le parole di codice, mentre si avrà un vettore di sindrome non nullo (non-zero), in cui cioè compare almeno un elemento diverso da zero, per le sequenze che non sono parole di codice. Tali sequenze potranno essere corrette dal decodificatore solo se il numero di errori introdotti dal canale rientra nelle capacità di correzione degli errori del codice stesso. Il vettore della sindrome può risultare diverso da zero anche quando la sequenza ricevuta contiene un numero di errori che eccede le capacità correttive del codice; in questo caso però il decodificatore non è in grado di correggerli.

Allo stesso modo, gli stabilizer generator (la cui definizione sarà fornita successivamente) di un QSC devono essere progettati in modo da produrre un autovalore pari a $+1$ per le parole di codice legittime, ed un autovalore di -1 in presenza di errori. Quindi, continuando il confronto con il dominio classico, si nota che, come la PCM $\mathbf{H}$ specifica completamente lo spazio di codice di un generico codice classico C, gli stabilizer generator definiscono lo spazio di codice di un QSC. Inoltre, si definisce gruppo stabilizzatore completo $\mathcal{H}$ di un QSC l'insieme di tutti gli stabilizer generator ed i loro prodotti.

Osservazione: distanza di un QSC Si può dimostrare che la nozione di *distanza* per un codice quantum è analoga a quella di un codice classico. Allora la distanza di un Quantum Stabilizer Code è definita come il peso

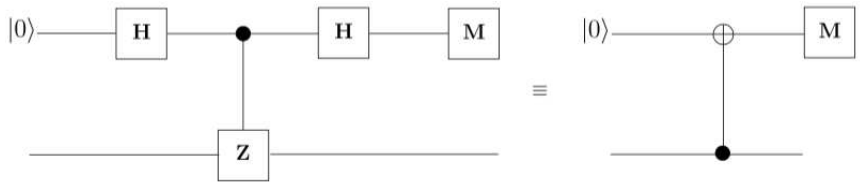

Figura 2.2: Circuito quantum di misurazione dell'operatore **Z** per la correzione di errori di tipo bit-flip [1], [2].

minimo (numero di operatori di Pauli non di identità nello stabilizzatore) di un elemento del gruppo stabilizzatore $\mathcal{H}$. Quindi se $\mathcal{C}[n,k]$ è un codice con distanza d, si dice che $\mathcal{C}$ è un codice stabilizzatore $[n,k,d]$. Analogamente al caso classico, un codice con distanza almeno $2t+1$ è in grado di correggere errori arbitrari su qualsiasi t qubit.

Si elencano di seguito le tre fasi di decodifica per un quantum error correcting code.

1) *Processo di calcolo della sindrome*: mentre lo spazio di codice C di un codice classico a blocco lineare è definito da una PCM **H** che ha $(n-k)$ righe indipendenti, lo spazio di codice $\mathcal{C}$, associato ad un QSC, è descritto da $(n-k)$ operatori g_i indipendenti, per codici da n qubit, ognuno composto da n matrici di Pauli, con $1 \leq i \leq (n-k)$. Questi sono generalmente chiamati stabilizer generator oppure, più brevemente, stabilizzatori (stabilizer) o generatori di Pauli (Pauli generator). In particolare, gli stabilizzatori sono operatori unici, cioè non perturbano lo stato delle parole di codice legittime, infatti danno come risultato un autovalore pari a $+1$, come sarà mostrato successivamente. Altrimenti, gli stabilizzatori producono un autovalore di -1 per le parole di codice affette da errori. Questo è equivalente ai valori 0 e 1 nella sindrome classica, rispettivamente. In alternativa, si può dimostrare che l'autovalore risultante è $+1$, quando l'errore $\mathcal{P}$, indotto dal canale, commuta con lo stabilizzatore g_i, mentre è -1, quando l'errore non commuta con g_i. Tale proprietà può essere matematicamente espressa nel seguente modo:

$$g_i \left|\hat{\psi}\right\rangle = \begin{cases} \left|\overline{\psi}\right\rangle, & \text{se } g_i\mathcal{P} = \mathcal{P}g_i, \\ -\left|\overline{\psi}\right\rangle, & \text{se } g_i\mathcal{P} = -\mathcal{P}g_i, \end{cases}$$

dove $\left|\hat{\psi}\right\rangle = \mathcal{P}\left|\overline{\psi}\right\rangle$.

Gli autovalori risultanti possono essere mappati sulla sindrome classica **s**, utilizzando i circuiti di Fig. 2.2 e di Fig. 2.3. Quindi, l'insieme degli stabilizzatori costituisce la controparte quantum della PCM classica. Tuttavia, gli stabilizzatori devono presentare la proprietà della commutatività addizionale, chiamato per semplicità criterio di commutatività. Per questa proprietà gli stabilizzatori devono formare coppie i cui due elementi sono mutuamente commutativi. In particolare, per una coppia di stabilizzatori

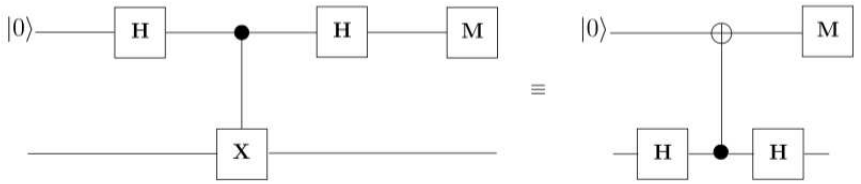

Figura 2.3: Circuito quantum di misurazione dell'operatore **X** per la correzione di errori di tipo phase-flip [1], [2].

g_1 e g_2, si ha:

$$g_1 g_2 \left|\overline{\psi}\right\rangle = g_1 \left|\overline{\psi}\right\rangle = \left|\overline{\psi}\right\rangle , \tag{2.3}$$

ed allo stesso modo, vale:

$$g_2 g_1 \left|\overline{\psi}\right\rangle = g_2 \left|\overline{\psi}\right\rangle = \left|\overline{\psi}\right\rangle . \tag{2.4}$$

Dunque, risulta evidente che il criterio di commutatività non esiste nel dominio classico. Inoltre, il gruppo stabilizzatore associato $\mathcal{H}$, che contiene $(n-k)$ stabilizzatori g_i così come tutti i possibili prodotti tra i vari g_i, forma un sottogruppo abeliano $\mathcal{G}_n$.

Approfondimento: Gruppo abeliano In matematica ed in particolare in algebra astratta, un gruppo si dice abeliano (o commutativo) se è un gruppo la cui operazione binaria interna gode della proprietà commutativa. Dunque il gruppo $(\mathcal{G}_n, \cdot)$ è abeliano se:

$$g_1 \cdot g_2 = g_2 \cdot g_1, \qquad \forall\, g_1, g_2 \in \mathcal{G}_n,$$

dove $\cdot$ sta ad indicare la generica operazione binaria commutativa interna al gruppo, che in questo caso è il prodotto. Anche in questo circostanza il nome deriva dall'ideatore, il matematico norvegese Niels Henrik Abel, che per primo teorizzò questi principi [10], sulla scia del lavoro di Evariste Galois.

Il decodificatore di Fig. 2.1 ("Inverse Encoder" nella figura) elabora la sindrome della sequenza ricevuta $|\hat{\psi}\rangle$ attraverso gli stabilizzatori associati, che vengono implementati utilizzando dei qubit ausiliari. Dunque, i qubit ausiliari collassano nelle sindromi classiche al momento della misurazione, mappando quindi gli autovalori $+1$ e -1 sui bit classici 0 e 1, rispettivamente. I bit di sindrome risultanti vengono poi alimentati da un decodificatore classico della sindrome, per stimare il vettore di errore del canale $\tilde{\mathcal{P}}$.

2) *Error Recovery* ($\mathcal{R}$): Il blocco di recupero degli errori ($\mathcal{R}$) di Fig. 2.1, ripristina il codice $|\tilde{\overline{\psi}}\rangle$, potenzialmente privo di errori, utilizzando il modello di errore stimato $\tilde{\mathcal{P}}$. Naturalmente, se il numero di errori supera la capacità di correzione del codice, il processo di recupero risulta imperfetto. Anzi, in quest'ultimo caso, tale azione correttiva inefficace produrrà più errori di

quanti ce ne fossero in origine.

3) *Codificatore inverso*: Infine, il codificatore inverso di Fig. 2.1 mappa la parola di codice recuperata $|\tilde{\bar{\psi}}\rangle$ su una stima della sequenza di informazione trasmessa $|\tilde{\psi}\rangle$. In particolare, mentre un codificatore tradizionale mappa le sequenze di informazione sulle parole di codice, un codificatore inverso lavora nella direzione inversa, mappando quindi le parole di codice sulle sequenze di informazione.

Si ricorda, dalle (2.3) e (2.4), che gli $(n-k)$ stabilizer generator g_i di un QSC sono sempre commutativi tra loro. Ciò implica che gli operatori costitutivi, cioè le matrici di Pauli $\mathbf{X}$, $\mathbf{Y}$ e $\mathbf{Z}$, devono essere selezionati in modo che tutti gli stabilizzatori risultanti commutino. Esplicitamente, si osserva che gli operatori (detti non di identità) $\mathbf{X}$, $\mathbf{Y}$ e $\mathbf{Z}$ sono intrinsecamente non commutativi tra loro. Ad esempio, si ha:

$$\mathbf{XY} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = \begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix} = i\mathbf{Z},$$

allo stesso modo, vale:

$$\mathbf{YX} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} -i & 0 \\ 0 & i \end{pmatrix} = -i\mathbf{Z}.$$

Ciò implica che gli operatori $\mathbf{XY}$ e $\mathbf{YX}$ sono anti-commutativi, cioè si ottiene:

$$\mathbf{XY} = -\mathbf{YX}.$$

In altre parole, la proprietà di anti-commutatività si raggiunge quando, dati due operatori, il loro prodotto non ordinato dà sempre lo stesso risultato, a meno del segno. Analogamente, si può facilmente dimostrare che:

$$\mathbf{YZ} = i\mathbf{X}, \quad \mathbf{ZY} = -i\mathbf{X} \Rightarrow \mathbf{YZ} = -\mathbf{ZY};$$
$$\mathbf{ZX} = i\mathbf{Y}, \quad \mathbf{XZ} = -i\mathbf{Y} \Rightarrow \mathbf{ZX} = -\mathbf{XZ}.$$

A causa della natura non commutativa degli operatori di Pauli $\mathbf{X}$, $\mathbf{Y}$ e $\mathbf{Z}$, gli stabilizzatori devono essere progettati in modo che ci sia un numero pari di indici (che identificano le posizioni delle matrici di Pauli all'interno dello stabilizzatore) con operatori non di identità diversi tra loro. Ad esempio, gli operatori di Pauli a 3 qubit $\mathbf{ZZI}$ e $\mathbf{XYZ}$ commutano, poiché sono caratterizzati da due indici con operatori non di identità diversi tra loro. In particolare, si tratta delle prime due posizioni, cioè degli indici $i = 1, 2$, in cui si ha: $\mathbf{Z} \neq \mathbf{X}$ e $\mathbf{Z} \neq \mathbf{Y}$. La terza posizione non viene conteggiata in questa valutazione, sebbene $\mathbf{I} \neq \mathbf{Z}$ (in questo caso sarebbero tre le disuguaglianze), poiché $\mathbf{I}$ è la matrice identità.

Al contrario, gli operatori $\mathbf{ZZI}$ e $\mathbf{YZI}$ sono anti-commutativi, dato che è presente un unico indice, corrispondente alla prima posizione, in cui sono presenti operatori non di identità diversi: $\mathbf{Z} \neq \mathbf{Y}$. Mentre, nelle altre

due posizioni si verifica l'uguaglianza tra le matrici: $\mathbf{Z} = \mathbf{Z}$ e $\mathbf{I} = \mathbf{I}$, con quest'ultima che comunque non rientra nel conteggio.

In definitiva, se due stabilizer generator non commutano, essi risultano giocoforza anti-commutativi, poiché le singole matrici di Pauli di cui sono composti saranno sempre commutative o anti-commutative.

2.3 Isomorfismo dal dominio quantum a quello classico

Sulla base della dualità tra i QSC e i codici classici a blocco lineari, discussa nella Sezione 2.2, in questa sezione si presenta l'isomorfismo tra questi due domini, che permette di costruire le versioni quantum dei codici classici conosciuti. In generale, per isomorfismo si intende la corrispondenza biunivoca fra gli elementi di due insiemi; in questo caso si allude alla categoria dei QSC e dei codici classici a blocco lineari (gli insiemi) e ai codici che ne fanno parte (gli elementi). In particolare, i QSC possono essere progettati a partire, tra gli altri, dai codici classici binari e quaternari, utilizzando le leggi di associazione (mapping) della Tabella 2.1, che permettono di passare dal dominio quantum a quello classico e viceversa. Inoltre, questo isomorfismo consente anche di utilizzare le classiche procedure di decodifica del vettore di sindrome, basato sulla PCM, per decodificare i QSC.

Osservazione Nel seguito della trattazione, il dominio quantum viene rappresentato dal dominio di Pauli che, come osservato nella Sottosezione 1.2.1, è una sua possibile declinazione. Pertanto verrà descritto l'isomorfismo dal dominio di Pauli al dominio classico binario e poi dal dominio di Pauli al dominio classico quaternario.

2.3.1 Isomorfismo dal dominio di Pauli al dominio binario

Si ricorda, dalla Sezione 2.2, che gli stabilizzatori costituiscono le controparti quantum della PCM classica. Sulla base di questa dualità, i QSC possono essere descritti utilizzando una PCM binaria equivalente, che a sua volta aiuta a progettare i codici quantum, a partire dalle famiglie di codici classici già esistenti.

In particolare, i QSC possono essere caratterizzati completamente, nel formalismo binario, da una PCM $\mathbf{H}$ binaria equivalente, derivata dagli stabilizer generator associati. Le righe della matrice $\mathbf{H}$ corrispondono agli stabilizzatori, mentre gli operatori di Pauli $\mathbf{I}$, $\mathbf{X}$, $\mathbf{Y}$ e $\mathbf{Z}$, che costituiscono gli stabilizzatori, sono mappati su una coppia di simboli binari, come segue:

$$\mathbf{I} \to (00), \quad \mathbf{X} \to (01), \quad \mathbf{Z} \to (10), \quad \mathbf{Y} \to (11). \tag{2.5}$$

A tal proposito si osservi ancora la Tabella 2.1, in cui con la notazione $(\mathbb{F}_2)^2$ si intende il dominio classico nel caso di due elementi. Quindi un 1 al

Tabella 2.1: Isomorfismo dal dominio quantum a quello classico [2].

Pauli	$(\mathbb{F}_2)^2$	$\mathbf{GF}(4)$
I	00	0
X	01	1
Y	11	$\overline{\omega}$
Z	10	ω
Moltiplicazione	Somma bit-a-bit	Addizione
Commutatività	Prodotto Simplettico	Traccia del prodotto scalare

primo indice rappresenta un operatore **Z**, mentre un 1 in seconda posizione rappresenta un operatore **X**. Infatti la matrice identità **I** è priva di 1, mentre l'operatore **Y** è composto da due 1 poiché combina gli effetti di **Z** e **X**, cioè bit-and-phase-flip.

La PCM **H**, risultante dal mapping dal dominio di Pauli al dominio classico binario della (2.5), cioè la legge di associazione che permette di passare dal dominio quantum al dominio classico binario, può anche essere espressa come:

$$\mathbf{H} = (\mathbf{H}_z|\mathbf{H}_x), \tag{2.6}$$

dove $\mathbf{H}_z$ e $\mathbf{H}_x$ sono matrici binarie di dimensione $(n-k)\times n$, che corrispondono agli operatori **Z** e **X**, rispettivamente. Ciò significa che la matrice $\mathbf{H}_z$ comprenderà solo gli 1 degli operatori **Z** e la matrice $\mathbf{H}_x$ solo i valori 1 degli operatori **X**.

Esempio: Codice a ripetizione a 3 qubit Un caso di studio didatticamente interessante è costituito dal codice a ripetizione a 3 qubit che corregge un errore di tipo bit-flip. Esso si basa sugli stabilizzatori

$$\begin{aligned} g_1 &= \mathbf{ZZI}, \\ g_2 &= \mathbf{ZIZ}. \end{aligned} \tag{2.7}$$

Utilizzando l'isomorfismo dal dominio di Pauli al dominio classico binario dell'eq.(2.5), questi stabilizzatori possono essere riscritti come segue

$$\begin{aligned} g_1 &= 101000, \\ g_2 &= 100010. \end{aligned} \tag{2.8}$$

Di conseguenza, la PCM **H** ad esso associata è data da:

$$\mathbf{H} = \left(\begin{array}{ccc|ccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{array}\right), \tag{2.9}$$

dove, come si può osservare, $\mathbf{H}_x$ è una matrice all-zero, cioè completamente composta da zeri, poiché g_1 e g_2 non contengono operatori di Pauli **X**. Inoltre, si può notare che la matrice $\mathbf{H}_z$ dell'eq.(2.9) è identica alla PCM **H** del codice a ripetizione classico.

Supponendo invece di utilizzare la seguente notazione più generica per indicare gli elementi binari degli stabilizer generator:

$$\begin{aligned} g_1 &= g_{1,1}g_{1,2}g_{1,3}g_{1,4}g_{1,5}g_{1,6}; \\ g_2 &= g_{2,1}g_{2,2}g_{2,3}g_{2,4}g_{2,5}g_{2,6}, \end{aligned} \tag{2.10}$$

la matrice associata $\mathbf{H}$ sarebbe così composta:

$$\mathbf{H} = \left(\begin{array}{ccc|ccc} g_{1,1} & g_{1,3} & g_{1,5} & g_{1,2} & g_{1,4} & g_{1,6} \\ g_{2,1} & g_{2,3} & g_{2,5} & g_{2,2} & g_{2,4} & g_{2,6} \end{array}\right). \tag{2.11}$$

Quindi, un QSC con $[n, k]$ qubit, avente $(n-k)$ stabilizzatori, può essere caratterizzato da una PCM binaria di dimensioni $(n-k) \times 2n$. Dunque, se si va ad estendere la generalizzazione proposta dalla (2.10) e dalla (2.11) al caso con $[n, k]$ qubit e $n-k$ stabilizzatori, di cui sopra, si trovano le seguenti relazioni:

$$\begin{aligned} g_1 &= g_{1,1_1}g_{1,1_2}g_{1,2_1}g_{1,2_2} \cdots g_{1,n_1}g_{1,n_2}; \\ g_2 &= g_{2,1_1}g_{2,1_2}g_{2,2_1}g_{2,2_2} \cdots g_{2,n_1}g_{2,n_2}, \\ &\vdots \\ g_{n-k} &= g_{n-k,1_1}g_{n-k,1_2}g_{n-k,2_1}g_{n-k,2_2} \cdots g_{n-k,n_1}g_{n-k,n_2}, \end{aligned}$$

dove il generico elemento g_{j,i_t} è caratterizzato dal pedice $j = 1, 2, ..., n-k$ che indica la numerosità degli stabilizzatori (e delle righe della matrice $\mathbf{H}$), mentre il pedice $i = 1, 2, ..., n$ rappresenta la quantità di operatori di Pauli che compongono lo stabilizer, cioè il numero di qubit codificati n. Infine il sottopedice $t = 1, 2$ indica l'elemento binario della matrice di Pauli che si sta designando, il primo o il secondo, rispettando il mapping della (2.5). In alternativa, si possono anche sostituire i valori 1, 2 del pedice t con z e x, in base alla regione della matrice considerando, quindi si avrebbe $t = z, x$.

Alla luce di queste considerazioni, la matrice $\mathbf{H}$ può espressa come segue:

$$\mathbf{H} = \left(\begin{array}{cccc|cccc} g_{1,1_z} & g_{1,2_z} & \cdots & g_{1,n_z} & g_{1,1_x} & g_{1,2_x} & \cdots & g_{1,n_x} \\ g_{2,1_z} & g_{2,2_z} & \cdots & g_{2,n_z} & g_{2,1_x} & g_{2,2_x} & \cdots & g_{2,n_x} \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots & \vdots & \vdots\vdots\vdots & \vdots \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots & \vdots & \vdots\vdots\vdots & \vdots \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots & \vdots & \vdots\vdots\vdots & \vdots \\ g_{n-k,1_z} & g_{n-k,2_z} & \cdots & g_{n-k,n_z} & g_{n-k,1_x} & g_{n-k,2_x} & \cdots & g_{n-k,n_x} \end{array}\right),$$

come previsto peraltro dalla (2.6). Inoltre, il rate di codifica classico equivalente R_c può essere determinato come segue:

$$\begin{aligned} R_c &= \frac{2n-(n-k)}{2n} = \frac{n+k}{2n} \\ &= \frac{1}{2}\left(1+\frac{k}{n}\right) = \frac{1}{2}(1+R_Q), \end{aligned}$$

Tabella 2.2: Addizione in $(\mathbb{F}_2)^2$ [2].

+	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

dove R_Q è il rate di codifica quantum associato.

Il formalismo binario della (2.5) trasforma la moltiplicazione degli operatori di Pauli nella somma bit a bit (bit-wise addition) della corrispondente rappresentazione binaria. Per esempio, moltiplicare l'insieme degli operatori di Pauli {**I**,**X**,**Z**,**Y**} per **X** è equivalente alla seconda colonna della Tabella 2.2, se gli operatori di Pauli sono mappati su $(\mathbb{F}_2)^2$ secondo le espressioni della (2.5). Allo stesso modo, la proprietà commutativa degli stabilizzatori, nel formalismo di Pauli, implica che le righe della matrice di parità **H** devono essere ortogonali tra loro, rispettando il prodotto simplettico (indicato anche come "twisted product" e definito di seguito) nel formalismo binario. In particolare, se l'i-esima riga di **H** è indicata come $\mathbf{H}_i = (\mathbf{H}_{z_i}|\mathbf{H}_{x_i})$ seguendo la notazione della (2.6), la commutatività degli stabilizzatori g_i e $g_{i'}$ si trasforma nel prodotto simplettico delle righe $\mathbf{H}_i$ e $\mathbf{H}_{i'}$, che viene calcolato come segue:

$$\mathbf{H}_i \star \mathbf{H}_{i'} = (\mathbf{H}_{z_i} \cdot \mathbf{H}_{x_{i'}} + \mathbf{H}_{z_{i'}} \cdot \mathbf{H}_{x_i}) \mod 2, \tag{2.12}$$

dove $\star$ indica, appunto, il prodotto simplettico e $\cdot$ indica il prodotto tra vettori (d'ora in avanti, per semplicità, la notazione $\cdot$ sarà omessa).

Il prodotto simplettico risultante produce un valore pari a zero, se il numero degli operatori non di identità (**X**, **Y** o **Z**) diversi, presenti negli stabilizzatori g_i e $g_{i'}$, è pari. In tal caso, soddisfacendo il criterio della commutatività. Inoltre, poiché tutti gli stabilizzatori devono essere commutativi, il prodotto simplettico deve essere zero per tutte le righe di **H**. Cioè la PCM **H** deve soddisfare la seguente relazione:

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \mathbf{0} \mod 2. \tag{2.13}$$

Questo, a sua volta, implica che qualsiasi coppia di codici binari classici, caratterizzati dalle PCM $\mathbf{H}_z$ e $\mathbf{H}_x$, che soddisfano la relazione sul prodotto simplettico della (2.13), può essere utilizzata per la costruzione di un QSC valido.

Il prodotto simplettico della (2.13) può anche essere sfruttato per il calcolo della sindrome di un QSC nel dominio binario, ad esempio durante la decodifica della sindrome basata sulla PCM. In particolare, l'isomorfismo dal dominio di Pauli a quello binario della (2.5) trasforma un errore di Pauli da n qubit ($\mathcal{P} \in \mathcal{G}_n$) in un vettore di errore effettivo **P** di lunghezza $2n$, come si può osservare dalla Fig. 2.4. In altre parole, analogamente al caso della matrice **H** nella (2.6), il vettore che rappresenta l'errore effettivo **P**

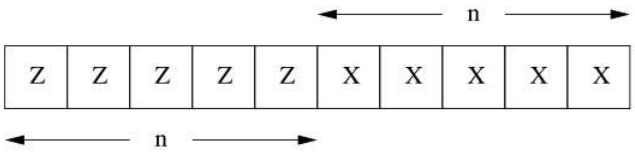

Figura 2.4: Rappresentazione dell'errore effettivo $\mathbf{P}$, corrispondente all'errore di Pauli $\mathcal{P}$ ad n qubit [2].

può essere espresso come $\mathbf{P} = (\mathbf{P}_z|\mathbf{P}_x)$, dove $\mathbf{P}_z$ e $\mathbf{P}_x$ sono legati agli errori di Pauli $\mathbf{Z}$ e $\mathbf{X}$, rispettivamente. Più precisamente, un 1 all'indice t-esimo di $\mathbf{P}_z$ denota un errore di Pauli $\mathbf{Z}$ (phase-flip) sul qubit t-esimo, mentre un 1 all'indice t-esimo di $\mathbf{P}_x$ rappresenta l'occorrenza dell'errore di Pauli $\mathbf{X}$ (bit-flip) sul qubit t-esimo. Allo stesso modo, l'errore di Pauli $\mathbf{Y}$ (bit-and-phase-flip) sul t-esimo qubit produce un 1 all'indice t-esimo di $\mathbf{P}_z$ e di $\mathbf{P}_x$.

Infine, la sindrome di un QSC può essere calcolata nel formalismo binario utilizzando il prodotto simplettico e il vettore di errore effettivo $\mathbf{P}$ come segue:

$$\mathbf{s} = \mathbf{H} \star \mathbf{P}^T = (\mathbf{H}_z \mathbf{P}_x^T + \mathbf{H}_x \mathbf{P}_z^T) \mod 2, \tag{2.14}$$

dove $\mathbf{H}_z$ e $\mathbf{H}_x$ sono utilizzati per correggere errori di bit-flip e di phase-flip, rispettivamente. La sindrome risultante assume quindi valori in $\{0, 1\}$. Pertanto, l'elaborazione della sindrome, nel dominio quantum, può essere effettuata nel dominio classico binario utilizzando la PCM $\mathbf{H}$ e l'errore effettivo $\mathbf{P}$. Ciò implica, a sua volta, che il processo di decodifica quantum è equivalente alla decodifica della sindrome, del codice classico equivalente, che si basa sulla PCM $\mathbf{H}$ [11].

2.3.2 Isomorfismo dal dominio di Pauli al dominio quaternario

Analogamente all'isomorfismo dal dominio di Pauli al dominio classico binario, l'isomorfismo da Pauli al dominio classico quaternario facilita la progettazione dei codici quantum, partendo dai codici classici quaternari già esistenti. In particolare, gli operatori di Pauli $\mathbf{I}$, $\mathbf{X}$, $\mathbf{Y}$ e $\mathbf{Z}$ possono essere trasformati negli elementi del campo di Galois di ordine 4 (GF(4)), utilizzando il mapping riportato di seguito:

$$\mathbf{I} \to 0, \quad \mathbf{X} \to 1, \quad \mathbf{Z} \to \omega, \quad \mathbf{Y} \to \overline{\omega}, \tag{2.15}$$

dove 0, 1, ω e $\overline{\omega}$ sono proprio gli elementi di GF(4).

Approfondimento: Campo di Galois Il campo di Galois (*Galois Field*, da cui l'acronimo GF), chiamato anche campo finito, è un campo che contiene un insieme finito di elementi. Un campo è un insieme in cui sono definite le operazioni di moltiplicazione, addizione, sottrazione e divisione

Tabella 2.3: Addizione in GF(4) [2].

$+$	0	1	ω	$\overline{\omega}$
0	0	1	ω	$\overline{\omega}$
1	1	0	$\overline{\omega}$	ω
ω	ω	$\overline{\omega}$	0	1
$\overline{\omega}$	$\overline{\omega}$	ω	1	0

Tabella 2.4: Moltiplicazione in GF(4) [2].

$\times$	0	1	ω	$\overline{\omega}$
0	0	0	0	0
1	0	1	ω	$\overline{\omega}$
ω	0	ω	$\overline{\omega}$	1
$\overline{\omega}$	0	$\overline{\omega}$	1	ω

e tali operazioni soddisfano le proprietà riassunte nelle Tabelle 2.3, 2.4, 2.5 e 2.6. L'ordine di un campo di Galois corrisponde al numero di elementi che lo compongono; tale numero è necessariamente un numero primo o una potenza di un numero primo.

Si può pertanto dimostrare che, per ogni numero primo p e per ogni numero intero positivo k, esistono campi di ordine p^k, tutti isomorfi tra loro. Inoltre, le regole di addizione e moltiplicazione per un campo di Galois di ordine p (GF(p)), con p numero primo, sono le stesse della somma e della moltiplicazione modulo p, mentre esistono altre leggi quando si considera un campo di ordine p^m (GF(p^m)), con $m > 1$, come nel caso in esame, in cui si ha: $p = 2$ e $m = 2$.

Si può osservare che l'operazione di moltiplicazione nel dominio di Pauli è equivalente all'operazione di addizione in GF(4), mentre il criterio di commutatività (cioè il prodotto simplettico) nel dominio di Pauli è equivalente alla traccia del prodotto scalare in GF(4); si guardi a tal proposito la Tabella 2.1 [9]. In particolare, l'operatore di traccia del prodotto scalare di GF(4) mappa x su $(x + \overline{x})$, dove $\overline{x}$ denota il coniugato di x, dunque vale $\mathrm{Tr}(x) \triangleq x + \overline{x} = x + x^2$, poiché l'operazione di coniugazione nel campo di Galois di ordine 4 è definita come $\bar{x} = x^2$ [12]. Pertanto, tale operazione non ha alcun impatto sugli elementi 0 e 1 di GF(4), mentre gli elementi ω e $\overline{\omega}$ sono scambiati quando si calcola il rispettivo coniugato.

Le regole relative all'addizione e alla moltiplicazione in GF(4) sono mostrate nelle tabelle 2.3 e 2.4, rispettivamente. Inoltre, l'operazione di moltiplicazione delle matrici di Pauli $\{\mathbf{I}, \mathbf{X}, \mathbf{Z}, \mathbf{Y}\}$ con $\mathbf{X}$ è equivalente a sommare l'elemento 1 di GF(4) (corrispondente a $\mathbf{X}$, secondo il mapping della (2.15)) a ciascun elemento di GF(4), come fatto nella seconda colonna della Tabella 2.3. D'altra parte, la relazione commutativa tra due elementi $\hat{A}$ e $\hat{B}$ di GF(4) può essere stabilita utilizzando la traccia del prodotto scalare, come

Tabella 2.5: Prodotto scalare Hermitiano in GF(4) [2].

$\langle , \rangle$	0	1	ω	$\overline{\omega}$
0	0	0	0	0
1	0	1	ω	$\overline{\omega}$
ω	0	$\overline{\omega}$	1	ω
$\overline{\omega}$	0	ω	$\overline{\omega}$	1

Tabella 2.6: Traccia del prodotto scalare in GF(4) [2].

$\mathrm{Tr}\langle , \rangle$	0	1	ω	$\overline{\omega}$
0	0	0	0	0
1	0	0	1	1
ω	0	1	0	1
$\overline{\omega}$	0	1	1	0

segue:

$$\mathrm{Tr}\langle \hat{A}, \hat{B} \rangle = \mathrm{Tr}(\hat{A} \times \overline{\hat{B}}) = 0,$$

dove $\langle , \rangle$ denota il prodotto scalare Hermitiano, mentre la notazione $\times$, in questo caso, denota il prodotto scalare semplice e $\overline{\hat{B}}$ è il coniugato di $\hat{B}$. N.B. Gli elementi di un generico GF(4) sono indicate con $\hat{\cdot}$, per esempio $\hat{x}$.

In particolare, si può dimostrare che $\mathrm{Tr}(\hat{A} \times \overline{\hat{B}}) = 0$ se gli operatori di Pauli associati ad a ed a b commutano, mentre $\mathrm{Tr}(\hat{A} \times \overline{\hat{B}}) = 1$ se tali matrici sono anti-commutative. Esplicitamente:

$$\begin{cases} \mathrm{Tr}(\hat{A} \times \overline{\hat{B}}) = 0, & \text{se } [\hat{A}, \hat{B}] = 0 \\ \mathrm{Tr}(\hat{A} \times \overline{\hat{B}}) = 1, & \text{se } [\hat{A}, \hat{B}] \neq 0. \end{cases} \tag{2.16}$$

Nel caso in esame, si ha che $\mathrm{Tr}(0) = \mathrm{Tr}(1) = 0$, invece $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$. Inoltre, sia il prodotto scalare Hermitiano che la traccia del prodotto scalare stesso tra gli elementi di GF(4) sono riportati nella Tabella 2.5 e nella Tabella 2.6, rispettivamente.

Se un QSC è caratterizzato dalla PCM classica $\hat{\mathbf{H}}$ nel dominio quaternario, allora il vincolo di commutatività degli stabilizzatori g_i e g_i' si trasforma nella traccia del prodotto scalare dell'i-esima e i'-esima colonna di $\hat{\mathbf{H}}$. In particolare, questa relazione può essere formulata come segue:

$$\hat{\mathbf{H}}_i \star \hat{\mathbf{H}}_{i'} = \mathrm{Tr}\langle \hat{\mathbf{H}}_i, \hat{\mathbf{H}}_{i'} \rangle = \mathrm{Tr}\left(\sum_{t=1}^{n} \hat{\mathbf{H}}_{it} \times \overline{\hat{\mathbf{H}}}_{i't} \right) = 0, \tag{2.17}$$

dove $\hat{\mathbf{H}}_{it}$ è l'elemento corrispondente alla riga i ed alla colonna t della matrice $\hat{\mathbf{H}}$. Si può ora osservare che la (2.12) e la (2.17) sono in realtà equivalenti, poiché entrambe rispondono al requisito della commutatività. Dato $\mathbf{H}_i = (\mathbf{H}_{z_i}, \mathbf{H}_{x_i})$ e conoscendo il mapping della (2.15), la matrice $\hat{\mathbf{H}}_i$ può essere espressa come:

$$\hat{\mathbf{H}}_i = \omega \mathbf{H}_{z_i} + \mathbf{H}_{x_i}. \tag{2.18}$$

Sostituendo poi la (2.18) nella (2.17), si ha:

$$\begin{aligned}\hat{\mathbf{H}}_i \star \hat{\mathbf{H}}_{i'} &= \mathrm{Tr}\langle(\omega\mathbf{H}_{z_i} + \mathbf{H}_{x_i}),\, (\omega\mathbf{H}_{z_{i'}} + \mathbf{H}_{x_{i'}})\rangle \\ &= \mathrm{Tr}((\omega\mathbf{H}_{z_i} + \mathbf{H}_{x_i})\,(\overline{\omega}\mathbf{H}_{z_{i'}} + \mathbf{H}_{x_{i'}})) \\ &= \mathrm{Tr}(\mathbf{H}_{z_i}\mathbf{H}_{z_{i'}} + \omega\mathbf{H}_{z_i}\mathbf{H}_{x_{i'}} + \overline{\omega}\mathbf{H}_{x_i}\mathbf{H}_{z_{i'}} + \mathbf{H}_{x_i}\mathbf{H}_{x_{i'}}), \qquad (2.19)\end{aligned}$$

ricordando che $\mathrm{Tr}(1) = 0$ e che $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$. Pertanto, la (2.19) si riduce a:

$$\hat{\mathbf{H}}_i \star \hat{\mathbf{H}}_{i'} = \mathbf{H}_{z_i}\mathbf{H}_{x_{i'}} + \mathbf{H}_{x_i}\mathbf{H}_{z_{i'}} \mod 2,$$

coincidente con la (2.12). Di conseguenza, analogamente alla (2.14), la sindrome nel caso quaternario può essere calcolata come:

$$s_i = \mathrm{Tr}(\hat{s}_i) = \mathrm{Tr}\left(\sum_{t=1}^{n} \hat{\mathbf{H}}_{it} \times \overline{\hat{P}}_t\right), \qquad (2.20)$$

dove s_i è la sindrome corrispondente all'i-esima riga di $\hat{\mathbf{H}}$ e $\hat{P}_t$ è l'elemento t-esimo di $\hat{P}$, che rappresenta l'errore determinato sul t-esimo qubit.

Qualsiasi codice classico lineare quaternario arbitrario, che è auto-ortogonale rispetto alla traccia del prodotto scalare (si veda la (2.17)), può essere utilizzato per la costruzione di un QSC.

Poiché un generico codice lineare quaternario è un spazio vettoriale chiuso rispetto all'operazione di moltiplicazione tra gli elementi di GF(4), questa condizione si riduce a soddisfare il prodotto scalare Hermitiano, piuttosto che la traccia del prodotto scalare stesso [12]. Questo può essere dimostrato come segue.

Sia C un codice lineare classico in GF(4) con parole di codice u e v. Inoltre, si suppone che:

$$\langle u, v\rangle = \alpha + \beta\omega. \qquad (2.21)$$

Per soddisfare il prodotto simplettico, bisogna avere:

$$\mathrm{Tr}\langle u, v\rangle = 0. \qquad (2.22)$$

Poiché $\mathrm{Tr}(\omega) = 1$, la (2.22) è valida solo quando β è nullo nella (2.21). Inoltre, dato che il codice C è lineare in GF(4), la (2.22) permette di ottenere:

$$\mathrm{Tr}\langle u, \overline{\omega}v\rangle = 0,$$

che a sua volta implica che anche α dovrebbe essere zero nella (2.21). Quindi, per un codice classico lineare in GF(4), il prodotto scalare Hermitiano della (2.21) deve essere zero, quando la traccia del prodotto scalare della (2.22) è nulla.

2.3.3 Conclusioni

Gli stabilizzatori possono essere mappati sulle rappresentazioni binarie o quaternarie equivalenti, come riassunto nella Tabella 2.1. Questi mapping,

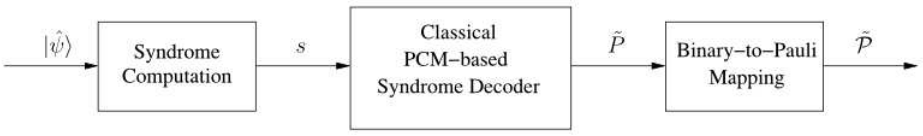

Figura 2.5: Schema a blocchi che rappresenta il processo di elaborazione della sindrome [2].

a loro volta, aiutano nella progettazione dei codici quantum, a partire dai codici classici già esistenti.

Inoltre, poiché un QSC può essere mappato su una PCM binaria o quaternaria classica equivalente, la decodifica della sindrome basata su una PCM classica può essere invocata durante il processo di decodifica. Più esplicitamente, il blocco di elaborazione della sindrome di Fig. 2.1 può essere ampliato, come mostrato in Fig. 2.5. Il processo inizia con il calcolo della sindrome della sequenza ricevuta $|\hat{\psi}\rangle$ utilizzando gli stabilizer generator, che collassano al valore binario 0 o 1 al momento della misurazione. La sequenza della sindrome binaria $\mathbf{s}$ alimenta poi un decodificatore classico basato sulla PCM, che opera sull'equivalente PCM classica associata al QSC per stimare l'errore di canale equivalente $\tilde{P}$ (o $\tilde{\hat{P}}$ nel dominio quaternario). Il decodificatore classico della sindrome basato sulla PCM della Fig. 2.5 è esattamente lo stesso decodificatore che si usa per un qualsiasi codice classico convenzionale, con l'eccezione delle seguenti due differenze:

1) In contrasto con la sindrome di un codice classico, che è data dal prodotto tra la PCM e il trasposto dell'errore di canale ($\mathbf{HP}^T$), la sindrome di un codice quantum viene calcolata utilizzando il prodotto simplettico della (2.14) (o la traccia del prodotto scalare della (2.20)).

2) La decodifica classica convenzionale mira a stimare l'errore più probabile, data la sindrome osservata, mentre la decodifica quantum ha l'obiettivo di stimare l'insieme di errori più probabile, che tiene conto della degenerazione dei codici quantum, come sarà discusso in seguito.

Infine, il mapping dal dominio classico binario al dominio di Pauli della (2.5) (o il mapping dal dominio classico quaternario al dominio di Pauli della (2.15)) viene invocato per mappare l'errore stimato binario (o quaternario) sull'equivalente errore di Pauli $\tilde{\mathcal{P}}$.

2.4 Tassonomia dei codici stabilizzatori

L'isomorfismo tra il dominio classico e quello quantum illustrato nella Sezione 2.3 fornisce un solido quadro teorico per la costruzione dei codici quantum, a partire dai codici classici già conosciuti. In questo capitolo verrà presentata la tassonomia dei codici stabilizzatori, cioè la loro classificazione. Si osservi dunque la Fig. 2.6 che presenta le tre famiglie di QSC proposte nel seguito:

– Codici CSS, nella Sezione 2.4.1;

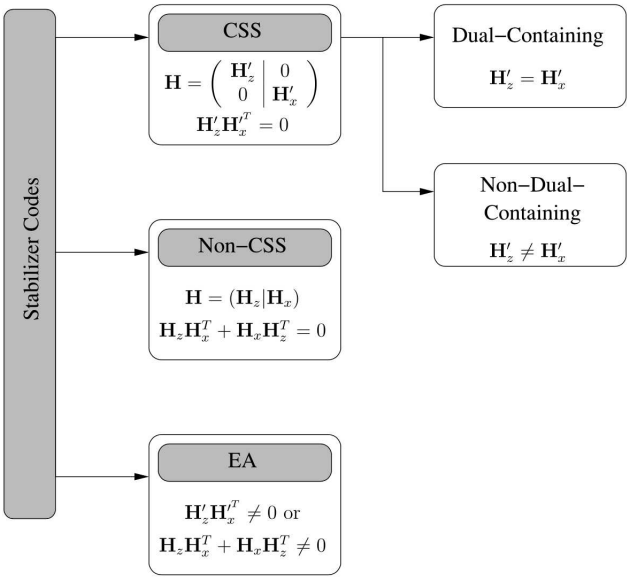

Figura 2.6: Tassonomia dei codici stabilizzatori (CSS: Calderbank-Shor-Steane, EA: Entanglement-Assisted) [2].

– Codici non CSS;
– Codici EA, nella Sezione 2.4.3.
La figura mostra la struttura della PCM $\mathbf{H}$ classica sottostante equivalente per ognuna di queste categorie di codice.

2.4.1 Codici Calderbank-Shor-Steane

I codici Calderbank-Shor-Steane (CSS) [13], [14], [15] sono una classe particolare di codici stabilizzatori costruiti a partire da una coppia di codici classici binari. In particolare, la famiglia dei codici CSS può essere definita come segue.

Un codice CSS $[n, k_1 - k_2]$ può essere progettato a partire dai codici classici a blocco lineari binari $C_1(n, k_1)$ e $C_2(n, k_2)$, se lo spazio di codice di C_1 comprende lo spazio di C_2 ($C_2 \subset C_1$). Inoltre, se sia C_1 che il duale di C_2, cioè $C_2^{\perp}$, presentano la stessa distanza minima di Hamming, pari a $d_{\min}$, allora il codice CSS risultante presenta anch'esso una distanza minima di valore $d_{\min}$. Pertanto, in accordo con la teoria classica, è in grado di correggere contemporaneamente fino a $(d_{\min} - 1)/2$ errori di bit-flip ed altrettanti errori di phase-flip. Più specificamente, il codice binario C_1 viene invocato per correggere le inversioni di bit, mentre il codice $C_2^{\perp}$ viene utilizzato per la correzione delle inversioni di fase. Quindi, se $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono le PCM di C_1 e $C_2^{\perp}$, rispettivamente, allora il codice CSS risultante è caratterizzato dalla seguente PCM:

$$\mathbf{H} = \left[\mathbf{H}_z \middle| \mathbf{H}_x\right] = \left(\begin{array}{c|c} \mathbf{H}'_z & \mathbf{0} \\ \mathbf{0} & \mathbf{H}'_x \end{array}\right), \tag{2.23}$$

dove si ha $\mathbf{H}_z = \begin{pmatrix} \mathbf{H}'_z \\ \mathbf{0} \end{pmatrix}$ e $\mathbf{H}_x = \begin{pmatrix} \mathbf{0} \\ \mathbf{H}'_x \end{pmatrix}$; dunque $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono matrici binarie di dimensioni $(n - k_1) \times n$ e $k_2 \times n$, rispettivamente.

Inoltre, a valle delle considerazioni precedenti, si enuncia il seguente teorema che lega la condizione del prodotto simplettico della (2.13) ai codici CSS.

Teorema

È dato un codice CSS in cui $C_2 \subset C_1$.

Allora la seguente condizione sul prodotto simplettico

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \mathbf{0} \mod 2.$$

si riduce a:

$$\mathbf{H}'_x\mathbf{H}_z'^T = \mathbf{H}'_z\mathbf{H}_x'^T = \mathbf{0}. \tag{2.24}$$

Dimostrazione Dal criterio del prodotto simplettico, è noto che la PCM $\mathbf{H}$ del codice deve soddisfare la seguente relazione:

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \mathbf{0} \mod 2, \tag{2.25}$$

poiché tutti gli stabilizzatori devono essere commutativi e quindi il prodotto simplettico vale zero per tutte le righe della matrice $\mathbf{H}$, come ampiamente spiegato nella Sezione 2.3.1.

Nel caso della costruzione CSS del codice, si osserva che il prodotto $\mathbf{H}_z\mathbf{H}_x^T$ equivale ad una matrice con valori diversi da 0 solo per k_2 elementi delle prime $(n - k_1)$ righe. Si ha cioè:

$$\begin{aligned}
\mathbf{H}_z\mathbf{H}_x^T &= \begin{pmatrix} \mathbf{H}'_z \\ \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{0} & \mathbf{H}_x'^T \end{pmatrix} \\
&= \left(\begin{array}{ccc|cccc}
0 & \dots & 0 & h_{1,1} & h_{1,2} & \dots & h_{1,k_2} \\
0 & \dots & 0 & h_{2,1} & h_{2,2} & \dots & h_{2,k_2} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \dots & 0 & h_{(n-k_1),1} & h_{(n-k_1),2} & \dots & h_{(n-k_1),k_2} \\
0 & \dots & 0 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \dots & 0 & 0 & 0 & \dots & 0
\end{array}\right) = \left(\begin{array}{c|c} \mathbf{0} & \mathbf{H}'_z\mathbf{H}_x'^T \\ \mathbf{0} & \mathbf{0} \end{array}\right),
\end{aligned}$$

pertanto la matrice $\mathbf{H}'_z\mathbf{H}_x'^T$, che ha dimensioni $(n-k_1)\times k_2$, è l'unica sezione di $\mathbf{H}_z\mathbf{H}_x^T$ dove gli elementi sono non nulli. Analogamente, si osserva che vale una relazione simile per la matrice $\mathbf{H}_x\mathbf{H}_z^T$. In particolare questa sarà non nulla per le ultime k_2 righe e $(n-k_1)$ colonne, dualmente al caso precedente.

Dunque, si trova:

$$\mathbf{H}_x\mathbf{H}_z^T = \begin{pmatrix}\mathbf{0}\\ \mathbf{H}'_x\end{pmatrix}\begin{pmatrix}\mathbf{H}_z'^T & \mathbf{0}\end{pmatrix}$$
$$= \left(\begin{array}{cccc|ccc} 0 & 0 & \dots & 0 & 0 & \dots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \dots & 0 & 0 & \dots & 0\\ h_{1,1} & h_{1,2} & \dots & h_{1,(n-k_1)} & 0 & \dots & 0\\ h_{2,1} & h_{2,2} & \dots & h_{2,(n-k_1)} & 0 & \dots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ h_{k_2,1} & h_{k_2,2} & \dots & h_{k_2,(n-k_1)} & 0 & \dots & 0 \end{array}\right) = \left(\begin{array}{c|c}\mathbf{0} & \mathbf{0}\\ \mathbf{H}'_x\mathbf{H}_z'^T & \mathbf{0}\end{array}\right),$$

quindi la matrice $\mathbf{H}'_x\mathbf{H}_z'^T$, che ha dimensioni $k_2 \times (n-k_1)$, è l'unica sezione di $\mathbf{H}_x\mathbf{H}_z^T$ dove gli elementi sono non nulli.

Dunque, fino a qui si è dimostrato che per i codici CSS vale come prodotto simplettico la seguente relazione:

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \left(\begin{array}{c|c}\mathbf{0} & \mathbf{H}'_z\mathbf{H}_x'^T\\ \mathbf{0} & \mathbf{0}\end{array}\right) + \left(\begin{array}{c|c}\mathbf{0} & \mathbf{0}\\ \mathbf{H}'_x\mathbf{H}_z'^T & \mathbf{0}\end{array}\right)$$
$$= \left(\begin{array}{c|c}\mathbf{0} & \mathbf{H}'_x\mathbf{H}_z'^T\\ \mathbf{H}'_z\mathbf{H}_x'^T & \mathbf{0}\end{array}\right) = \mathbf{0} \mod 2, \qquad (2.26)$$

per ipotesi della (2.25). Si osserva dalla (2.26) quindi, che entrambi i prodotti $\mathbf{H}'_x\mathbf{H}_z'^T$ e $\mathbf{H}'_z\mathbf{H}_x'^T$ sono giocoforza nulli mod 2, altrimenti tale relazione non sarebbe rispettata.

Per costruzione, ciascuno degli addendi a primo membro della (2.26) è il trasposto dell'altro, cioè valgono le relazioni:

$$\mathbf{H}_z\mathbf{H}_x^T = \left(\mathbf{H}_x\mathbf{H}_z^T\right)^T$$
$$\mathbf{H}_x\mathbf{H}_z^T = \left(\mathbf{H}_z\mathbf{H}_x^T\right)^T.$$

Pertanto, questa proprietà si verifica anche per le matrici costitutive non nulle, si ottiene quindi:

$$\mathbf{H}'_z\mathbf{H}_x'^T = \left(\mathbf{H}'_x\mathbf{H}_z'^T\right)^T$$
$$\mathbf{H}'_x\mathbf{H}_z'^T = \left(\mathbf{H}'_z\mathbf{H}_x'^T\right)^T,$$

cioè le due matrici contengono le stesse informazioni, ma sono disposte all'interno della matrice in un ordine diverso.

In conclusione, sulla base di quanto dimostrato finora, si può giungere alla seguente relazione:

$$\mathbf{H}'_x\mathbf{H}_z'^T = \mathbf{H}'_z\mathbf{H}_x'^T = \mathbf{0} \mod 2,$$

che, per i codici CSS, va a sostituire la più complessa relazione sul prodotto simplettico espressa dalla (2.13). #

Tabella 2.7: Somma degli elementi di $\mathrm{Z}_4 = \{0, 1, 2, 3\}$ modulo 4.

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Quindi, il processo di progettazione di un QSC si riduce nel trovare una coppia di codici binari le cui PCM sono conformi al criterio del prodotto simplettico espresso dalla (2.24). Poiché la PCM risultante dalla (2.23) ha $(n-k_1+k_2)$ righe, il codice quantum codifica (k_1-k_2) qubit di informazione in n qubit codificati totali. Inoltre, se si ha $\mathbf{H}'_z = \mathbf{H}'_x$, allora il codice risultante è chiamato dual-containing (o auto-ortogonale), ed è caratterizzato da $\mathbf{H}'_z\mathbf{H}'^{T}_z = \mathbf{0}$, che è equivalente ad osservare che $C_1^{\perp} \subset C_1$.

In particolare, nel caso dei codici CSS dual-containing, $C_2(n, k_2)$ è il codice duale di $C_1(n, k_1)$. Di conseguenza, si ha che $k_2 = (n - k_1)$ e i codici CSS dual-containing che ne risultano trasformano $(k_1 - k_2) = (2k_1 - n)$ qubit di informazione in n qubit codificati.

Si classificano invece i restanti modelli di codice CSS, in base alla Fig. 2.6, cioè quelli che hanno $\mathbf{H}'_z \neq \mathbf{H}'_x$, come codici CSS non-dual-containing.

Un codice CSS con $[n, k_1 - k_2]$ qubit, basato sui codici binari C_1 e $C_2^{\perp}$, è implementato trovando i sottoinsiemi unici di C_2 in C_1, in modo che ciascuno dei $2^{k_1-k_2}$ stati sovrapposti possa essere mappato su un sottoinsieme unico di C_2 in C_1.

Osservazione: sottoinsiemi unici Si supponga, a titolo di esempio, di lavorare con dei generici insiemi, invece che con gli spazi di codice. Sia dato allora l'insieme $C_1 = \{0, 1, 2, 3\}$ con $k_1 = 2$ e l'insieme $C_2 = \{0, 2\}$ con $k_2 = 1$, allora l'addizione modulo 4 tra ogni singolo elemento di C_1 e gli elementi di C_2 produce i seguenti sottoinsiemi:

$$
\begin{aligned}
0 + C_2 &\equiv \{0, 2\} = C_2; \\
1 + C_2 &\equiv \{1, 3\} = 1 + C_2; \\
2 + C_2 &\equiv \{2, 0\} = C_2; \\
3 + C_2 &\equiv \{3, 1\} = 1 + C_2.
\end{aligned}
$$

I risultati della somma modulo 4 si giustificano facilmente alla luce della Tabella 2.7.

Dunque, si osserva che ci sono due diversi sottoinsiemi unici di C_2 in C_1, cioè, $\{0, 2\}$ e $\{1, 3\}$. Equivalentemente, si può dire che l'unione dei due sottoinsiemi unici $\{0, 2\}$ e $\{1, 3\}$ di C_2 costituisce l'insieme C_1.

I sottoinsiemi unici di C_2 in C_1, a loro volta, si ricavano attraverso una somma bit-a-bit modulo 2 (Tabella 2.8) tra ogni parola di codice di C_1 e quelle dello spazio di codice di C_2; si guardi, per capirne la ratio, l'esempio di cui sopra.

Tabella 2.8: Somma bit-a-bit modulo 2.

+	0	1
0	0	1
1	1	0

In particolare, se la parola di codice $\mathbf{x}_1 \in C_1$ e la parola di codice $\mathbf{x}_2 \in C_2$, l'operazione di somma normalizzata può essere formulata come segue:

$$|\mathbf{x}_1 + C_2\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\mathbf{x}_2 \in C_2} |\mathbf{x}_1 + \mathbf{x}_2\rangle \,, \tag{2.27}$$

dove + denota la somma modulo 2. Poiché la cardinalità di C_1 è $|C_1| = 2^{k_1}$, mentre quella di C_2 è $|C_2| = 2^{k_2}$, si ottengono $|C_1|/|C_2| = 2^{k_1-k_2}$ sottoinsiemi unici di C_2 in C_1. In altre parole, si hanno $2^{k_1-k_2}$ parole di codice diverse. Di conseguenza, ciascuno stato quantum ortogonale, composto da $2^{k_1-k_2} \cdot (k_1 - k_2)$ qubit, può essere mappato su una sovrapposizione delle parole di codice del sottoinsieme unico.

2.4.1.1 Esempio Operativo: codice di Steane

Si consideri la costruzione del codice di Steane $[7, 1]$, il quale deriva dal codice classico dual-containing di Hamming $(7, 4)$, che ha la seguente PCM:

$$\mathbf{H} = \begin{pmatrix} 1\,1\,0\,1\,1\,0\,0 \\ 1\,0\,1\,1\,0\,1\,0 \\ 0\,1\,1\,1\,0\,0\,1 \end{pmatrix}. \tag{2.28}$$

La PCM $\mathbf{H}$ della (2.28) produce $\mathbf{H}\mathbf{H}^T = \mathbf{0}$, quindi si presta essa stessa a costruire un codice CSS dual-containing.

In particolare, C_1 è il codice di Hamming (7, 4), mentre C_2 è il suo codice duale (codice simplesso), cioè $C_2 = C_1^{\perp}$, con parametri $(7, 3)$. Dato che $\mathbf{H}\mathbf{H}^T = \mathbf{0}$, lo spazio di codice di C_2 è contenuto in quello di C_1, cioè, si ha $C_2 \subset C_1$. Inoltre, sia C_1 che $C_2^{\perp} = C_1$ possono correggere un singolo errore, cioè il codice di Steane è in grado di correggere un solo errrore di bit-flip o di phase-flip. Tale codice CSS può essere costruito trovando i sottoinsiemi unici di $C_1^{\perp}$ in C_1, usando la (2.27), dato che $C_1^{\perp} = C_2$. Ciò si traduce in due sottoinsiemi unici, che sono elencati nella Tabella 2.9. Questi due sottoinsiemi, che contengono tutte le parole di codice possibili, partecipano a definire lo spazio di codice del codice di Hamming. I due stati ortogonali $|0\rangle$ e $|1\rangle$ della parola di informazione a singolo qubit sono quindi codificati come segue:

$$\begin{aligned} |\bar{0}\rangle \equiv & \frac{1}{\sqrt{8}}(|0000000\rangle + |0111001\rangle + |1011010\rangle + |1100011\rangle \\ & + |1101100\rangle + |1010101\rangle + |0110110\rangle + |0001111\rangle), \\ |\bar{1}\rangle \equiv & \frac{1}{\sqrt{8}}(|1111111\rangle + |1000110\rangle + |0100101\rangle + |0011100\rangle \\ & + |0010011\rangle + |0101010\rangle + |1001001\rangle + |1110000\rangle). \end{aligned}$$

Tabella 2.9: Sottoinsiemi unici di $C_1^{\perp}$ in C_1 [2].

Sottoinsieme 1	Sottoinsieme 2
0000000	1111111
0111001	1000110
1011010	0100101
1100011	0011100
1101100	0010011
1010101	0101010
0110110	1001001
0001111	1110000

In altre parole, $|\bar{0}\rangle$ e $|\bar{1}\rangle$ sono le sovrapposizioni, caratterizzate dallo stesso peso di Hamming, di tutte le parole di codice dei due sottoinsiemi della Tabella 2.9. Inoltre, $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono equivalenti alla PCM binaria del codice di Hamming della (2.28). Quindi, gli stabilizzatori associati, per il rilevamento degli errori di bit-flip e di phase-flip, del codice $[7, 1]$ di Steane sono i seguenti:

$$\begin{aligned} g_1 &= \mathbf{ZZIZZII}; \\ g_2 &= \mathbf{ZIZZIZI}; \\ g_3 &= \mathbf{IZZZIIZ}; \\ g_4 &= \mathbf{XXIXXII}; \\ g_5 &= \mathbf{XIXXIXI}; \\ g_6 &= \mathbf{IXXXIIX}. \end{aligned} \tag{2.29}$$

Allora, operando l'isomorfismo dal dominio di Pauli al dominio classico binario, gli stabilizzatori risultano:

$$\begin{aligned} g_0 &= 10\,10\,00\,10\,10\,00\,00 \rightarrow \left(1\ 1\ 0\ 1\ 1\ 0\ 0 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right), \\ g_1 &= 10\,00\,10\,10\,00\,10\,00 \rightarrow \left(1\ 0\ 1\ 1\ 0\ 1\ 0 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right), \\ g_2 &= 00\,10\,10\,10\,00\,00\,10 \rightarrow \left(0\ 1\ 1\ 1\ 0\ 0\ 1 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right), \\ g_3 &= 01\,01\,00\,01\,01\,00\,00 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 1\ 1\ 0\ 1\ 1\ 0\ 0\right), \\ g_4 &= 01\,00\,01\,01\,00\,01\,00 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 1\ 0\ 1\ 1\ 0\ 1\ 0\right), \\ g_5 &= 00\,01\,01\,01\,00\,00\,01 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 0\ 1\ 1\ 1\ 0\ 0\ 1\right). \end{aligned}$$

Pertanto, la matrice che rappresenta il codice CSS di tipo dual-containing in esame risulta:

$$\mathbf{H}_{\text{css}} = \left(\begin{array}{c|c} \mathbf{H} & \mathbf{0} \\ \mathbf{0} & \mathbf{H} \end{array}\right) = \left(\begin{array}{ccccccc|ccccccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right),$$

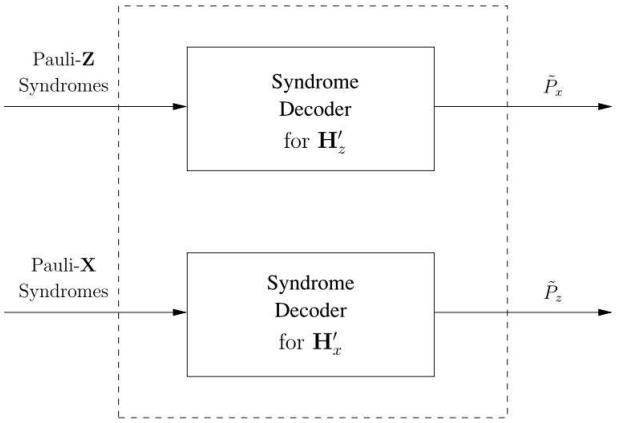

Figura 2.7: Decodificatore di sindrome per i codici quantum di tipo CSS [2].

cioè la si è ricavata semplicemente per righe a partire dalla rappresentazione degli stabilizer generator nel dominio classico binario.

Si può così osservare che nella (2.29) così come nella (2.23), gli stabilizzatori per il rilevamento degli errori di bit-flip e di phase-flip (o, equivalentemente, sindromi) di un codice quantum di tipo CSS sono indipendenti. Pertanto, la stima del bit-flip e del phase-flip può essere effettuata, indipendentemente, da due decodificatori classici basati su sindrome separati, utilizzando le matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$, rispettivamente, come illustrato in Fig. 2.7.

Le curve delle prestazioni di questo codice sono presenti nel Capitolo 6.

Osservazione: BER e QBER Inoltre, quando viene utilizzato il decodificatore rappresentato in Fig. 2.7, la prestazione dei codici CSS, osservata nell'ambito del canale di depolarizzazione, è isomorfa rispetto alle stesse su due canali indipendenti di inversione di fase e di inversione di bit, dove ciascuno ha una probabilità di depolarizzazione pari a $2p/3$. Quindi le prestazioni, in termini di QBER, dei codici CSS possono essere approssimate sommando tra loro i BER dei codici binari costituenti. In particolare, dato che p_e^x e p_e^z sono i BER classici per $\mathbf{H}'_x$ e $\mathbf{H}'_z$, rispettivamente, il codice CSS risultante presenta un QBER di:

$$\text{QBER} = p_e^x + p_e^z - p_e^x p_e^z \approx p_e^x + p_e^z,$$

che è equivalente a $2p_e^z$ per un codice CSS dual-containing con $\mathbf{H}'_x = \mathbf{H}'_z$.

2.4.2 Codici Non CSS

Si è osservato nella sezione precedente che i codici CSS correggono indipendentemente gli errori di bit-flip e di phase-flip. Questo, a sua volta, si traduce in un basso rate di codifica. Al contrario, i codici stabilizzatori non CSS sono in grado di sfruttare la ridondanza in modo più efficiente, poiché correggono gli errori di tipo bit-flip e di tipo phase-flip in modo congiunto. La PCM di un codice non CSS assume la struttura generale espressa nella

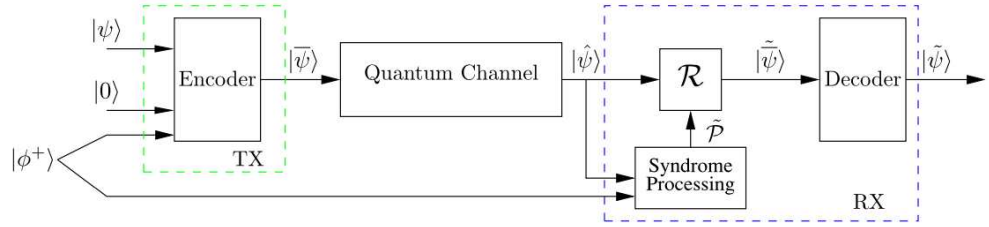

Figura 2.8: Schema a blocchi di un sistema di comunicazione basato su un codice stabilizzatore entanglement-assisted [2].

(2.6). Di conseguenza, per progettare un codice stabilizzatore non CSS si possono utilizzare:
− una coppia di PCM binarie, conformi al criterio del prodotto simplettico espresso dalla (2.13);
− una PCM classica quaternaria, che soddisfi il criterio della traccia del prodotto scalare.

Calderbank, Rains, Shor e Sloane hanno concepito una classe speciale di codici non CSS, chiamati codici Calderbank-Rains-Shor-Sloane (CRSS), che sono costruiti a partire dai codici quaternari classici e possono essere caratterizzati come descritto in [12]. Alla fine della prossima sezione è presente un esempio operativo di un codice non CSS.

2.4.3 Codici Entanglement-Assisted

Si ricorda che i QSC possono essere costruiti a partire dai codici classici binari e quaternari, solo se i codici classici costituenti sono conformi al criterio del prodotto simplettico della (2.13). Di conseguenza, mentre ogni QSC può avere una controparte nel dominio classico, non si può certo affermare il contrario; cioè che ogni codice classico abbia una versione quantum basata sugli stabilizzatori. Pertanto, è stato concepito il formalismo stabilizzatore entanglement-assisted , illustrato in [16] e [17], che si basa su qubit entangled già condivisi con il ricevitore autorizzato su un canale privo di rumore.

In particolare, il formalismo EA permette di trasformare un insieme di generatori di Pauli non commutativi in un insieme di generatori commutativi, che a loro volta costituiscono validi codici stabilizzatori. Di conseguenza, quando i codici classici sottostanti non soddisfano il criterio del prodotto simplettico, viene invocato il formalismo EA per rendere commutativi gli stabilizzatori dati.

La Fig. 2.8 mostra il modello di un sistema di comunicazione basato su un codice stabilizzatore entanglement-assisted (EA-QSC, Entanglement-Assisted − Quantum Stabilizer Code). Nello specifico, un EA-QSC con $[n, k, c]$ qubit codifica una sequenza di informazione di k qubit $|\psi\rangle$ in una parola di codice, composta da n qubit $|\overline{\psi}\rangle$, utilizzando $(n - k - c)$ qubit ausiliari nello stato $|0\rangle$ e c qubit entangled già condivisi, chiamati più

semplicemente ebit (entangled qubit). Gli ebit possono essere concretizzati utilizzando lo stato di Bell $|\phi^+\rangle$ e sono espressi come:

$$|\phi^+\rangle = \frac{|00\rangle^{T_X R_X} + |11\rangle^{T_X R_X}}{\sqrt{2}}. \tag{2.30}$$

La (2.30), che rappresenta una coppia di ebit, indica che il primo qubit è conservato presso il trasmettitore (T_X), mentre il secondo qubit, entangled con il primo, viene inviato al ricevitore (R_X) prima che inizi la trasmissione effettiva, ad esempio durante le ore "non di punta", quando i canali sono poco utilizzati. Le notazioni T_X e R_X nella (2.30) vengono utilizzate per identificare la "metà" dell'ebit (cioè lo stato $|0\rangle$ o $|1\rangle$) del trasmettitore e del ricevitore, rispettivamente. Come detto, generalmente si presuppone che il pre-sharing degli ebit avvenga su un canale non rumoroso. Inoltre, come illustrato nella Fig. 2.8, il trasmettitore utilizza solo la "sua metà" degli ebit ($|0\rangle^{T_X}$ oppure $|1\rangle^{T_X}$) per codificare la sequenza di informazione $|\psi\rangle$ nella parola di codice $|\overline{\psi}\rangle$. Infine, le informazioni codificate vengono inviate su un canale quantum rumoroso. Al ricevitore, la parola di codice ricevuta $|\hat{\psi}\rangle$ è pertanto affetta da rumore ed è combinata con la metà del ricevitore dei c ebit, durante il processo di decodifica.

Gli stabilizzatori di un EA-QSC agiscono congiuntamente sulla parola di codice ricevuta $|\hat{\psi}\rangle$ e sugli ebit del ricevitore, per calcolare il vettore di sindrome. Questo poi viene utilizzato da un decodificatore basato sulla sindrome classica per stimare il modello di errore $\tilde{\mathcal{P}}$. Il resto del processo di elaborazione, al ricevitore, è identico a quello del QSC non assistito, presentato in Fig. 2.1.

Lo stato di Bell della (2.30) ha proprietà uniche che facilitano il mapping di un insieme di generatori non commutativi in un insieme di generatori commutativi. Più esplicitamente, i generatori commutativi a 2 qubit $\mathbf{X}^{T_X}\mathbf{X}^{R_X}$ e $\mathbf{Z}^{T_X}\mathbf{Z}^{R_X}$ stabilizzano lo stato $|\phi^+\rangle$, cioè si ha:

$$[\mathbf{X}^{T_X}\mathbf{X}^{R_X}, \mathbf{Z}^{T_X}\mathbf{Z}^{R_X}] = \mathbf{0}.$$

In particolare, la notazione $[\cdot\,,\,\cdot]$ indica l'operazione che verifica (o meno) la commutatività tra due operatori (in questo caso tra due matrici). Pertanto, considerando due generiche matrici quadrate $\mathbf{A}$ e $\mathbf{B}$ di dimensioni 2×2, si ottiene:

$$[\mathbf{A}, \mathbf{B}] \equiv \mathbf{AB} - \mathbf{BA} = \begin{cases} \mathbf{0}, & \text{se } \mathbf{A} \text{ e } \mathbf{B} \text{ commutano,} \\ 2\mathbf{AB}, & \text{se } \mathbf{A} \text{ e } \mathbf{B} \text{ anti-commutano.} \end{cases}$$

Tuttavia, gli operatori Pauli, che agiscono sui singoli qubit, risultano essere mutuamente ("mutuamente" sta per: "l'uno con l'altro") anti-commutativi, dunque si ottiene:

$$[\mathbf{X}^{T_X}, \mathbf{Z}^{T_X}] \neq \mathbf{0};$$
$$[\mathbf{X}^{R_X}, \mathbf{Z}^{R_X}] \neq \mathbf{0}.$$

Pertanto, se si ha una coppia di generatori non commutativi $\mathbf{X}^{T_X}$ e $\mathbf{Z}^{T_X}$, che agisce solo sull'ebit del trasmettitore, questi due generatori possono essere trasformati in una coppia di generatori commutativi, utilizzando un operatore aggiuntivo (operazione di aumento del generatore) che agisce sull'ebit del ricevitore. In particolare, l'operatore che agisce sugli ebit del ricevitore è scelto proprio per garantire che i generatori risultanti, composti da 2 qubit, abbiano un numero pari di indici associati ad operatori non di identità diversi. Viene così risolta l'anti-commutatività degli operatori iniziali da un singolo qubit.

2.4.3.1 Esempio operativo

Si costruisce ora un EA-QSC a partire da due codici classici binari che hanno le seguenti PCM:

$$\mathbf{H}_z = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix},$$

e

$$\mathbf{H}_x = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

N.B. Questo è un esempio arbitrario a scopo didattico, concepito solo per illustrare la costruzione di codici EA a partire da codici classici noti. Pertanto, il codice classico e quello quantum associati potrebbero non avere buone capacità di correzione degli errori.

Le PCM $\mathbf{H}_z$ e $\mathbf{H}_x$ possono essere concatenate per la costruzione di un codice non CSS avente la seguente matrice di parità:

$$\mathbf{H} = \left(\mathbf{H}_z \middle| \mathbf{H}_x\right) = \left(\begin{array}{cccc|cccc} 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{array}\right). \tag{2.31}$$

Ora bisogna verificare se la matrice $\mathbf{H}$ rispetta il criterio del prodotto simplettico, espresso nella (2.13), per il quale si ha che:

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \mathbf{0} \mod 2.$$

Dopo aver calcolato e le matrici trasposte $\mathbf{H}_x^T$ e $\mathbf{H}_z^T$ e dopo averle sostituite,

l'equazione precedente diventa:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} 0 & 1 & 2 & 2 \\ 1 & 0 & 2 & 2 \\ 2 & 2 & 2 & 4 \\ 2 & 2 & 4 & 4 \end{pmatrix} \neq \mathbf{0} \mod 2.$$

Sfortunatamente si è verificato che la PCM della (2.31) non soddisfa il criterio del prodotto simplettico.

D'altra parte, la PCM $\mathbf{H}$ può essere trasformata nei seguenti generatori di Pauli, utilizzando il mapping dal dominio di Pauli al dominio classico binario della (2.5):

$$\mathbf{H}_Q = \begin{pmatrix} \mathbf{X} & \mathbf{Z} & \mathbf{X} & \mathbf{I} \\ \mathbf{X} & \mathbf{X} & \mathbf{I} & \mathbf{X} \\ \mathbf{Y} & \mathbf{Z} & \mathbf{Z} & \mathbf{X} \\ \mathbf{X} & \mathbf{Y} & \mathbf{Y} & \mathbf{Z} \end{pmatrix}, \tag{2.32}$$

dove il pedice Q sta ad indicare che la matrice $\mathbf{H}_Q$ è l'equivalente, nel dominio quantum, della PCM $\mathbf{H}$ della (2.31). Nello specifico, si ha a che fare con i seguenti generatori:

$$g_1 = \mathbf{XZXI} \rightarrow g_1 = 01100100;$$
$$g_2 = \mathbf{XXIX} \rightarrow g_2 = 01010001;$$
$$g_3 = \mathbf{YZZX} \rightarrow g_3 = 11101001;$$
$$g_4 = \mathbf{XYYZ} \rightarrow g_4 = 01111110,$$

dove $\rightarrow$ indica il mapping dal dominio di Pauli al dominio classico binario, secondo la (2.5), che inevitabilmente risultano non commutativi, dato che la PCM $\mathbf{H}$ non rispetta il criterio del prodotto simplettico.

In particolare, i primi due generatori (o righe) di $\mathbf{H}_Q$ sono anti-commutativi, poiché il numero di indici corrispondenti ad operatori di Pauli non di identità, cioè $\mathbf{X}$, $\mathbf{Z}$ o $\mathbf{Y}$, diversi tra loro, è pari a 1, poiché vale:

$$(i)\, \mathbf{X} = \mathbf{X};$$
$$(ii)\ \mathbf{Z} \neq \mathbf{X};$$
$$(iii)\, \mathbf{X} \neq \mathbf{I};$$
$$(iv)\ \mathbf{I} \neq \mathbf{X},$$

dove la prima valutazione dà esito negativo, poiché i due operatori di Pauli sono uguali, mentre la terza e la quarta valutazione non sono valide in quanto $\mathbf{I}$ è la matrice di identità. Dunque l'unica valutazione che dà esito positivo è la seconda. Pertanto, quando il numero di indici è dispari, cioè 1, come in questo caso, i due generatori designati non sono commutativi.

Diversamente, con lo stesso metodo di cui sopra, si può dimostrare che tutti gli altri generatori (o righe) effettivamente commutano tra loro. In altre parole, solo gli operatori che agiscono sul secondo qubit anti-commutano, mentre gli operatori che agiscono individualmente su tutti gli altri qubit risultano essere commutativi. Per costruire i generatori della (2.32) in modo tale che siano commutativi, le prime due righe della matrice $\mathbf{H}_Q$ possono essere "aumentate" con una coppia di operatori non di identità anti-commutativi, come mostrato di seguito:

$$\mathbf{H}_Q = \left(\begin{array}{cccc|c} \mathbf{X} & \mathbf{Z} & \mathbf{X} & \mathbf{I} & \mathbf{Z} \\ \mathbf{X} & \mathbf{X} & \mathbf{I} & \mathbf{X} & \mathbf{X} \\ \mathbf{Y} & \mathbf{Z} & \mathbf{Z} & \mathbf{X} & \mathbf{I} \\ \mathbf{X} & \mathbf{Y} & \mathbf{Y} & \mathbf{Z} & \mathbf{I} \end{array}\right),$$

dove infatti vale $\mathbf{ZX} = -\mathbf{XZ}$. Come si può facilmente dimostrare, le matrici di identità $\mathbf{I}$ possono essere aggiunte senza modificare il conteggio; in questa sede hanno infatti la funzione di elemento neutro. In questo modo si è ottenuta una matrice $\mathbf{H}_Q$ effettivamente commutativa, cioè che rispetta il criterio del prodotto simplettico e che dunque può essere utilizzata per rappresentare un codice non CSS, come desiderato in partenza.

Si osserva che gli operatori a sinistra della barra verticale (|) agiscono sulle parole di codice trasmesse, composte da n qubit, mentre quelli a destra della barra verticale agiscono sugli ebit del ricevitore. Dunque, in questo esempio di progettazione, è richiesto solo un ebit poiché è presente solo una colonna a destra della barra verticale.

Capitolo 3

Codici classici LDPC

I codici Low-Density Parity-Check (LDPC) furono introdotti da Robert Gallager in [18] e [19] negli anni '60 dello scorso secolo.

Sono codici a blocco lineari caratterizzati dalla dimensione del blocco pari a k bit (di informazione), dalla lunghezza del blocco pari ad n (cioè i bit codificati totali), e quindi, la ridondanza del codice è $m = n - k$. Per rappresentare e costruire efficientemente questi codici viene utilizzata la matrice di parità $\mathbf{H}$ che, come si evince dal nome della struttura del codice, è a bassa densità; in particolare queste matrici, in base alla numerosità degli elementi non nulli, possono essere definite come *sparse* o *molto sparse*.

Nel caso binario, pertanto, la PCM associata al codice LDPC è composta da 0s e da 1s.

I codici LDPC possono essere rappresentati da un grafo bipartito, chiamato così poiché gli elementi (nodi) che si trovano dalla stessa parte del grafo non si connettono mai tra di loro, chiamato grafo di Tanner [20]. Tale grafo è composto da due insiemi di nodi; quello dei nodi variabile e dei nodi di controllo (detti anche nodi check, o di parità). Il numero dei nodi variabile e dei nodi di controllo corrisponde alla lunghezza di blocco del codice e al numero di simboli di parità, rispettivamente. Se un nodo variabile è vincolato da un nodo di controllo, c'è un edge che collega questi due nodi.

Esempio: Grafo di Tanner del codice di Hamming Si osservi la solita PCM $\mathbf{H}$ che, a meno di inessenziali inversioni delle colonne, rappresenta il codice di Hamming, già ampiamente presentato nel caso classico nella Sezione 2.1

$$\mathbf{H} = \begin{pmatrix} 0\ 1\ 1\ 1\ 1\ 0\ 0 \\ 0\ 0\ 1\ 1\ 1\ 1\ 0 \\ 1\ 1\ 0\ 1\ 0\ 1\ 1 \end{pmatrix}$$

e che in questo caso si sfrutta per la costruzione di un codice LDPC arbitrario.

Dal grafo di Tanner riferito a questo codice, riportato nella Fig. 3.1, si nota che le 3 righe e le 7 colonne della matrice sono rappresentate da $\{c_0, c_1, c_2\}$ (nodi di controllo) e da $\{v_0, v_1, v_2, v_3 v_4, v_5, v_6\}$ (nodi variabile),

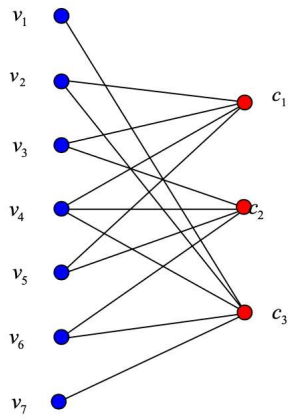

Figura 3.1: Grafo di Tanner del codice di esempio.

rispettivamente.

La matrice di parità è una rappresentazione del grafo di Tanner (e viceversa) perché gli 1s consentono di stabilire quando due nodi (uno variabile e uno di controllo) si collegano tramite un edge:

$$c_i \leftrightarrow v_j \iff \mathbf{H}_{i,j} = 1.$$

Il girth g del codice è la lunghezza del ciclo più breve presente nel grafo di Tanner associato ed è un parametro cruciale per i codici LDPC. I cicli, soprattutto quelli brevi, per esempio di lunghezza 4, degradano le prestazioni dei decodificatori LDPC, poiché influenzano l'indipendenza delle informazioni estrinseche che i nodi si scambiano nella decodifica iterativa [21], [22]. Ne è stato evidenziato uno nella Fig. 3.2 nel grafo di Tanner dell'esempio di cui sopra. Come si vedrà nel prossimo capitolo, la dimensione di un ciclo svolge un ruolo fondamentale anche nella decodifica di codici quantum LDPC.

3.1 Elementi di decodifica iterativa per codici LDPC

Non si conosce un decodificatore ottimo e pratico, inteso come "a Massima Verosimiglianza" (Maximum Likelihood − ML) [23], per decodificare un codice LDPC. Pertanto, in pratica, si usano decodificatori iterativi che hanno prestazioni sub-ottime. Infatti, i codici LDPC con queste tecniche di decodifica rivelano prestazioni eccellenti, molto vicine al limite di Shannon sui canali AWGN e BSC [20], [24], [25], [26], [27].

Per rendere efficiente l'algoritmo della decodifica iterativa, un codice LDPC solitamente è caratterizzato da:

− pochi 1 nella PCM, quindi da una bassa densità della matrice, cioè pochi edge nel grafo di Tanner;

− non più di una sovrapposizione di 1 tra ogni coppia di colonne e di righe (Row-Column Constraint);

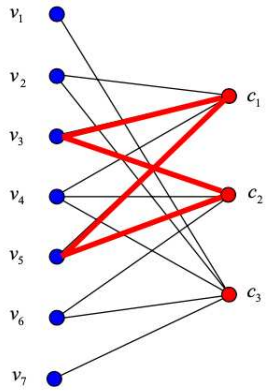

Figura 3.2: Grafo di Tanner del codice di esempio con evidenziato un ciclo di lunghezza 4.

– cicli nel grafo di Tanner più lunghi possibile. Nei grafi bipartiti il girth è sempre maggiore o uguale a 4.

Il decodificatore iterativo funziona cercando una soluzione ottima globale, ma i cicli corti lo obbligano a lavorare localmente. Il problema dei cicli è proprio questo: più questi sono brevi e più l'informazione scambiata dal decodificatore iterativo è autoreferenziale, come sarà più chiaro in seguito.

3.1.1 Algoritmo Belief Propagation

L'algoritmo Belief Propagation (BP) è uno dei principali algoritmi iterativi utilizzati per la decodifica LDPC. È stato introdotto da Gallager nella sua tesi di dottorato ed è conosciuto anche come algoritmo Sum-Product (SPA).

Il criterio ottimo su cui è costruito tale algoritmo è il criterio MAP (Maximum a Posteriori Probability, letteralmente "massima probabilità a posteriori") sul simbolo [23]. In particolare, interessa calcolare APP (A Posteriori Probability, letteralmente "probabilità a posteriori"), cioè la specifica probabilità per cui un bit nella parola di codice trasmessa $\mathbf{v}$ valga 1, data la parola di codice ricevuta $\mathbf{y}$. Supponendo che si sia interessati al j-esimo bit di $\mathbf{v}$, la APP vale

$$\Pr(v_j = 1|\mathbf{y}).$$

Spesso in realtà si fa riferimento al rapporto di verosimiglianza (LR)

$$l(v_j|\mathbf{v}) = \frac{\Pr(v_j = 0|\mathbf{y})}{\Pr(v_j = 1|\mathbf{y})},$$

o al rapporto di verosimiglianza logaritmico (LLR)

$$L(v_j|\mathbf{y}) = \log\left(\frac{\Pr(v_j = 0|\mathbf{y})}{\Pr(v_j = 1|\mathbf{y})}\right),$$

che risulta particolarmente utile quando si lavora sul canale AWGN.

L'algoritmo SPA può calcolare queste quantità, applicando il turbo principio al grafo di Tanner, che prevede lo scambio dell'informazione estrinseca

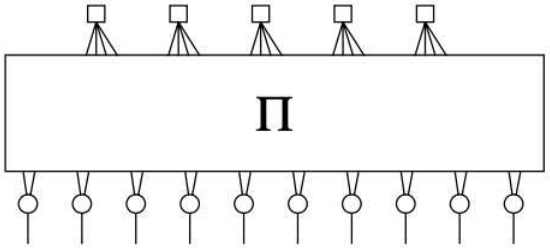

Figura 3.3: Rappresentazione grafica di un codice LDPC come concatenazione di un codice a ripetizione e a bit di parità. I nodi variabile sono rappresentati da piccole circonferenze, connesse al canale (linee singole verso il basso), mentre i nodi check da quadrati. L'interleaver è il blocco centrale "π" [23].

tra i nodi. La spiegazione dettagliata del turbo principio va oltre gli scopi di questo lavoro, dunque si rimanda il lettore appassionato a consultare [23].

A livello generale, si osserva che un codice LDPC può essere visto come codici a bit di parità (nodi check), concatenati tramite un interleaver, a codici a ripetizione (nodi variabile). I codici a bit di parità sono visti come codici esterni (non connessi al canale), mentre i codici a ripetizione sono interpretati come codici interni, cioè direttamente connessi al canale, come mostrato in Fig. 3.3.

I decodificatori per i nodi variabile e i nodi di controllo cooperano per stimare $L(v_j|\mathbf{y})$. Se non ci fossero cicli, o se i cicli fossero lunghissimi, le stime sarebbero accurate e il decodificatore avrebbe prestazioni vicine al MAP. Vale inoltre l'assunzione di indipendenza: gli LLR ricevuti da ogni nodo, provenienti dai suoi vicini, sono indipendenti. Questa assunzione, però, viene meno già quando il numero di iterazioni supera la metà del girth.

Come ogni algoritmo iterativo, c'è effettivamente bisogno di un criterio che ne determina la fine. Le condizioni che possono portare lo SPA a fermarsi sono due; la prima è squisitamente algebrica e consiste in

$$\hat{\mathbf{v}}\mathbf{H}^T = \mathbf{0}, \tag{3.1}$$

dove $\hat{\mathbf{v}}$ è la parola decodificata all'iterazione i-esima dell'algoritmo. Come si può osservare, la (3.1) equivale al calcolo della sindrome. La seconda condizione è prettamente algoritmica e consiste nel raggiungere un prefissato numero di iterazioni. Dopo queste essenziali premesse, si elencano i passi dell'algoritmo.

• **Inizializzazione**: Il decodificatore si inizializza considerando i valori ricevuti dal canale y_i e impostando tutti i messaggi $L_{j\to i}$ a

$$L_j = L(v_j \,|\, y_j) = \log\left(\frac{\Pr(v_j = 0|y_j)}{\Pr(v_j = 1|y_j)}\right).$$

• **Scambio verticale dei messaggi**: Questa fase dell'algoritmo prevede il calcolo e lo scambio dell'informazione dai nodi di controllo ai nodi

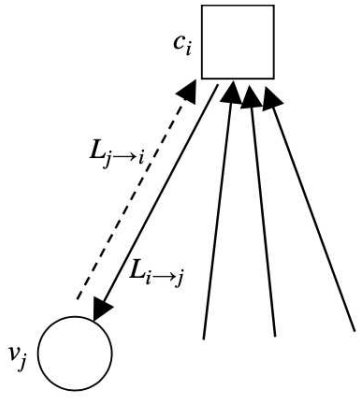

Figura 3.4: Rappresentazione di un decodificatore per codice a bit di parità, riferito al generico nodo di controllo. L'i-esimo nodo di controllo c_i riceve i messaggi sotto forma di LLR da tutti i nodi variabile ad esso collegati, escluso il messaggio $L_{j\to i}$ proveniente dal j-esimo nodo variabile v_j (marginalizzazione). Dopodiché c_i calcola il messaggio $L_{i\to j}$ che viene inviato a v_j [23].

variabile, come rappresentato in Fig. 3.4. Tale procedura è esprimibile con la seguente forma analitica

$$L_{i\to j} = 2\tanh^{-1}\left(\prod_{j'\in N(i)-\{j\}} \tanh\left(\frac{1}{2}L_{j'\to i}\right)\right),$$

dove $N(i)$ è l'insieme dei nodi variabile connessi al nodo di controllo c_i ed escludendo v_j $(N(i) - \{j\}$ si ottiene la marginalizzazione, cioè non si considera nella produttoria l'informazione proveniente dal nodo v_j. Questa equazione esprime la migliore stima estrinseca (cioè che tiene conto della marginalizzazione) del valore di c_i (il segno di $Li \to j$) e l'affidabilità della stima stessa (il valore numerico di $Li \to j$). Quindi, il generico nodo di controllo c_i riceve i messaggi da tutti i v_j nodi variabile ad esso collegati e poi risponde ad ognuno con un'informazione che non tiene conto del messaggio che proveniva da ognuno di questi. Pertanto, ogni nodo variabile riceve un'informazione calcolata a partire da dati diversi.

• **Scambio orizzontale dei messaggi**: Questa fase dell'algoritmo prevede il calcolo e lo scambio dell'informazione dai nodi variabile ai nodi di controllo, come rappresentato in Fig. 3.5. Tale procedura è esprimibile con la seguente forma analitica

$$L_{j\to i} = L_j + \sum_{i'\in N(j)-\{i\}} L_{i'\to j},$$

dove $N(j)$ è l'insieme dei nodi di controllo connessi al nodo variabile v_j ed escludendo c_i $(N(j) - \{i\}$ si ottiene la marginalizzazione, cioè non si considera nella sommatoria l'informazione proveniente dal nodo c_i. Come sopra, questa equazione esprime la migliore stima estrinseca (cioè che tiene conto

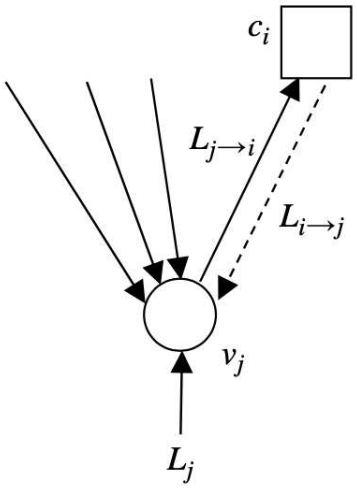

Figura 3.5: Rappresentazione di un decodificatore per codice a ripetizione, riferito al generico nodo variabile. Il j-esimo nodo variabile v_j riceve i messaggi sotto forma di LLR dal canale e da tutti i nodi di controllo ad esso collegati, escluso il messaggio $L_{i\to j}$ proveniente dall'i-esimo nodo di controllo c_i (marginalizzazione). Dopodiché v_j calcola il messaggio $L_{j\to i}$ che viene inviato a c_i [23].

della marginalizzazione) del valore di v_j (il segno di $L_{j\to i}$) e l'affidabilità della stima stessa (il valore numerico di $L_{j\to i}$). Quindi, il generico nodo variabile v_j riceve i messaggi da tutti i c_i nodi di controllo ad esso collegati e poi risponde ad ognuno con un'informazione che non tiene conto del messaggio che proveniva da ognuno di questi. Pertanto, ogni nodo di controllo riceve un'informazione calcolata a partire da dati diversi.

• **Calcolo dell'LLR totale**: Alla fine di ogni iterazione, per tutti i j bisogna calcolare

$$L_j^{\text{totale}} = L_j + \sum_{i\in N(j) L_{i\to j}}$$

si osservi che in questo caso non è presente la marginalizzazione poiché è importante tenere conto di tutte le informazioni che giungono al nodo variabile v_j.

• **Criteri di terminazione dell'algoritmo**: Per modulazioni di ordine 2 che mappano 1 in -1, e 0 in $+1$, la decisione sul valore stimato del bit della parola ricevuta viene presa secondo la regola seguente:

$$\begin{cases} \hat{v}_j = 1, & \text{se } L_j^{\text{total}} < 0, \\ \hat{v}_j = 0, & \text{altrimenti,} \end{cases}$$

per ogni j. Se la (3.1) è verificata o se si è raggiunto il numero massimo di iterazioni, l'algoritmo finisce, altrimenti termina soltanto l'iterazione corrente e riprende dalla fase di "Scambio verticale dei messaggi".

Capitolo 4

Progetto dei codici QLDPC

Analogamente al caso dei codici LDPC classici, che appartengono alla famiglia dei codici a blocco lineari, i codici quantum LDPC (QLDPC − Quantum Low-Density Parity-Check Codes) fanno riferimento alla teoria dei codici stabilizzatori. Pertanto, essi possono essere definiti utilizzando la matrice equivalente alla matrice PCM classica $\mathbf{H}$, introdotta nella Sezione 2.3. Per comodità, se ne riporta l'espressione valida secondo il mapping dal dominio di Pauli al dominio classico binario:

$$\mathbf{H} = (\mathbf{H}_z \,|\, \mathbf{H}_x).$$

In particolare, un codice QLDPC caratterizzato da $[n, k]$ qubit, con rate di codifica di $R_Q = k/n$, equivale a un codice LDPC binario $(2n, n + k)$ con rate di codifica pari a $R_c = (n + k)/2n$. Dunque, i codici QLDPC si possono suddividere in tre categorie principali; riferendosi alla struttura generale dei codici quantum e basata sulla PCM $\mathbf{H}$ associata, presentata nella Sezione 2.4. Vale a dire i codici CSS (Calderbank-Shor-Steane), i codici non CSS e i codici Entanglement-Assisted (EA), come riportato opportunamente nella Fig. 3.1. I codici CSS si classificano ulteriormente in due tipi; cioè come codici *dual-containing* (auto-ortogonali) oppure *non dual-containing* (non auto-ortogonali). A valle di queste doverose premesse, si analizza singolarmente ognuna di queste categorie di codice, nel caso di costruzione QLDPC.

4.1 Codici QLDPC di tipo CSS

Idealmente, per la costruzione di un codice QLDPC con struttura CSS, può essere utilizzata una qualsiasi coppia di codici LDPC binari classici che soddisfi il rigoroso criterio del prodotto simplettico (si osservi, a tal proposito, ancora la Sezione 2.4). Tuttavia, è sconsigliabile scegliere casualmente il duo di codici classici, poiché trovare due codici LDPC che soddisfano il vincolo del prodotto simplettico è molto difficile.

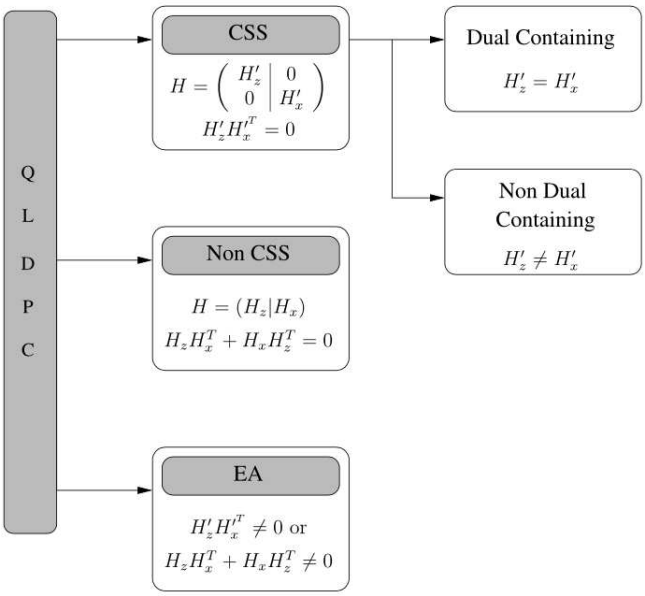

Figura 4.1: Tassonomia dei codici QLDPC (CSS: Calderbank-Shor-Steane, EA: Entanglement-Assisted) [28].

4.1.1 Codici QLDPC di tipo CSS a geometria finita

Nel 2001 questa costruzione classica motivò Postol in [29] a concepire il primo esempio di codice QLDPC di tipo CSS non dual-containing, basandosi su un piccolo codice LDPC $(15, 7)$ classico a geometria finita [23].

Ricordando la seguente relazione che rappresenta la PCM di un generico codice quantum di tipo CSS:

$$\mathbf{H} = \left[\mathbf{H}_z \middle| \mathbf{H}_x\right] = \left(\begin{array}{c|c} \mathbf{H}'_z & \mathbf{0} \\ \mathbf{0} & \mathbf{H}'_x \end{array}\right),$$

si osserva che nel codice di Postol, la PCM del codice LDPC ciclico classico a geometria finita corrisponde a $\mathbf{H}'_z$, mentre la matrice $\mathbf{H}'_x$ è derivata da $\mathbf{H}'_z$ in modo che sia soddisfatto il criterio del prodotto simplettico, cioè deve valere: $\mathbf{H}'_z \mathbf{H}'^T_x = \mathbf{0}$. Poiché entrambe le due PCM costituenti, cioè $\mathbf{H}'_x$ e $\mathbf{H}'_z$, sono cicliche, l'implementazione del codificatore associato è facilitata, poiché può essere realizzato in modo efficiente utilizzando opportuni registri a scorrimento.

Approfondimento: Codice classico ciclico e matrice ciclica Un codice classico ciclico appartiene al sottoinsieme dei codici a blocco lineari. Un codice a blocco lineare C si dice ciclico quando, data la parola di codice c, uno spostamento ciclico degli elementi di c (verso sinistra o verso destra) genera comunque una parola di codice valida. Cioè se vale:

$$\mathbf{c} = (c_1, c_2, \ldots, c_{n-1}, c_n),$$

allora, a fronte di uno spostamento ciclico unitario verso destra, si ottiene la parola di codice

$$\mathbf{c}^{(1)} = (c_2, c_2, \ldots, c_{n-1}, c_n, c_1),$$

che è ancora una parola di codice valida ($\mathbf{c} \in C$) [30].

Una matrice quadrata $\mathbf{A}$, di dimensioni $(n \times n)$, è definita *ciclica* quando consente di generare l'intero spazio vettoriale mediante un unico vettore generatore; in altri termini quando esiste un vettore $\mathbf{v}$ di dimensioni $(n \times 1)$ (vettore colonna) tale che la matrice $[\mathbf{v}\, \mathbf{A}\mathbf{v}\, \ldots\, \mathbf{A}^{(n-1)}\mathbf{v}]$ sia di rango massimo. Come appare evidente, la matrice generatrice e la PCM associata ad un codice ciclico sarà anch'essa ciclica.

Tuttavia, Postol non ha sviluppato un metodo generale per la sua proposta di progetto; questo avrebbe facilitato la costruzione dei codici QLDPC a partire da qualsiasi codice LDPC classico a geometria finita arbitrario.

Questa lacuna è stata colmata successivamente da MacKay *et al.* in [11], dove sono state sviluppate diverse costruzioni sistematiche per i codici QLDPC di tipo CSS, facendo riferimento a livello progettuale, in prima istanza, alla struttura dual-containing, cioè quando il codice contiene il suo duale.

Prima di procedere con tali costruzioni, è necessario richiamare la seguente relazione che rappresenta il rigoroso criterio del prodotto simplettico

$$\mathbf{H}_z\mathbf{H}_x^T + \mathbf{H}_x\mathbf{H}_z^T = \mathbf{0} \bmod 2, \tag{4.1}$$

caratterizzandola nel contesto dei codici QLDPC di tipo CSS dual-containing. Infatti, la (4.1) si riduce a $\mathbf{H}'_z\mathbf{H}'^T_z = \mathbf{0}$ per i codici di tipo CSS dual-containing, dato che per definizione vale $\mathbf{H}'_x = \mathbf{H}'_z$. Questo, a sua volta, implica che la PCM di un codice LDPC classico può essere utilizzata per la costruzione di un codice QLDPC dual-containing sole se sono verificate le seguenti due condizioni:

1) Il peso di riga ρ deve essere pari;

2) ogni coppia di righe deve avere un numero pari di sovrapposizioni di 1, tale condizione viene definita *even overlap*.

Al contrario, nel dominio classico i codici LDPC good hanno al massimo una singola sovrapposizione di 1 tra ogni coppia di righe, poiché si devono evitare i cicli di lunghezza 4.

Approfondimento: Codici good e codici bad Si definisce *bad* un codice a blocco che realizza una probabilità di errore per bit p_b arbitrariamente piccola solo facendo tendere il rate R a 0.

Al contrario, un codice si definisce *good* se realizza una probabilità di errore per bit p_b arbitrariamente piccola a qualsiasi rate R al di sotto della capacità C del canale.

Tali cicli brevi compromettono le prestazioni dell'algoritmo di decodifica associato, pertanto sono considerati dannosi. Di conseguenza, la condizione

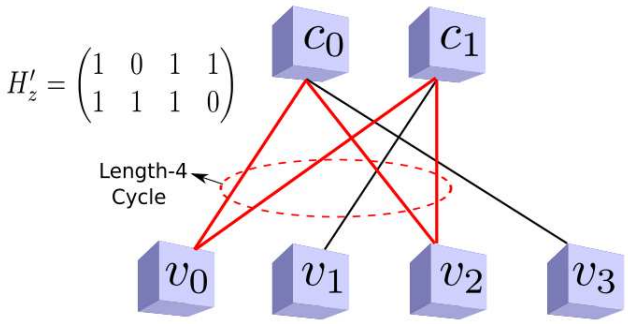

Figura 4.2: Grafo di Tanner di $\mathbf{H}'_z$. La condizione di even overlap tra le righe della matrice genera i cicli di lunghezza 4 [28].

di even overlap si traduce nella presenza inevitabile di cicli di lunghezza 4 nella PCM risultante. Si illustra a titolo di esempio didattico, il seguente codice per spiegare la condizione di even overlap, come mostrato dalla Fig. 4.2. Per un codice binario arbitrario, la PCM $\mathbf{H}'_z$ è data da:

$$\mathbf{H}'_z = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

N.B. La $\mathbf{H}'_z$ potrebbe non rappresentare un codice classico valido, è frutto di una scelta arbitraria.

Siccome il codice classico è dual-containing, lo spazio di codice del codice classico sottostante contiene il suo duale. Quindi, il codice risultante contiene parole di codice aventi peso pari al peso della riga ρ. Pertanto, la distanza minima del codice classico dual-containing non può essere maggiore di ρ. Tuttavia, questo limite superiore non sempre esiste per i codici quantum a causa della natura degenere degli errori nel dominio quantum.

4.1.2 Modelli di errore del canale quantum

In particolare il modello di errore derivante dal canale, che distorce le parole di codice di un QSC, può essere classificato come segue:

1) *Modelli di errore rilevati*: Questi modelli di errore anti-commutano con gli stabilizzatori del codice, producendo una sindrome non triviale.

2) *Modelli di errore non rilevati dannosi*: Questa classe di errori commuta con gli stabilizzatori. Di conseguenza, questi modelli di errore sono dannosi, perché mappano una parola di codice valida su un'altra; quindi corrompono la parola di codice senza innescare una sindrome non triviale. Questa tipologia di errori è causata dalla ridotta distanza minima del codice.

3) *Modelli di errore non rilevati innocui*: Questa è una classe di modelli di errore unica, che non ha una controparte classica. Sono simili ai succitati *Modelli di errore non rilevati dannosi*, infatti anche questi commutano con gli stabilizzatori. Tuttavia, sono innocui nel contesto dei codici quantum, perché sono errori degeneri, cioè appartengono al gruppo degli stabilizzatori, e quindi non corrompono lo stato delle parole di codice valide.

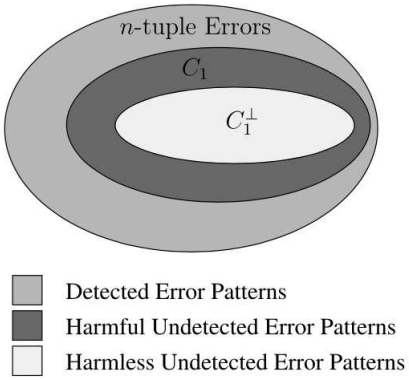

Figura 4.3: Classificazione dei modelli di errore per codici CSS dual-containing [28].

In particolare, per i codici CSS dual-containing, i modelli di errore non rilevati innocui si trovano nello spazio di codice del codice duale $C_1^{\perp}$, come illustrato nella Fig. 4.3. Si deve ricordare infine che, anche se gli errori non rilevati innocui non influenzano la distanza minima del codice quantum risultante, essi portano all'*errore di degenerazione simmetrica*, durante la procedura di decodifica iterativa dei codici QLDPC, che sarà discussa nella Sezione 5.3. Per ulteriori chiarimenti riguardanti la classificazione dei modelli di errore nei QSC, si faccia riferimento alla Sezione 2.4.

4.1.3 Codici QLDPC di tipo CSS a doppio ciclo e a ciclo unico

I codici a doppio ciclo (o biciclici), proposti da MacKay *et al.* in [11], hanno segnato il primo importante passo verso la realizzazione dei codici quantum LDPC di tipo CSS dual-containing. La progettazione del codice proposto si basa su una costruzione semi-random/semi-structured, che soddisfa la relazione dei codici dual-containing, imponendo deliberatamente una struttura globale sulla PCM costituente.

Un codice a doppio ciclo avente peso della riga pari a ρ, lunghezza del blocco di n e $(n-k)$ stabilizzatori, è costruito utilizzando una matrice ciclica $\mathbf{C}_m$ casuale e sparsa di dimensioni $(n/2 \times n/2)$, che ha peso della riga pari a $\rho/2$. Gli elementi diversi da zero in $\mathbf{C}_m$ possono essere scelti sia in modo casuale, sia utilizzando un *difference set* che soddisfi la proprietà per la quale la differenza (modulo $n/2$) tra due elementi del set è diversa da ogni altra differenza tra due diversi elementi del set.

Approfondimento: Insieme delle differenze semplice e perfetto Un insieme delle differenze perfetto è caratterizzato dal gruppo additivo di dimensione n e dalla proprietà unica che ogni intero da 1 a $n-1$ può essere espresso come differenza (modulo n) di soli due interi nell'insieme.

Al contrario, negli insiemi delle differenze semplici, chiamati anche brevemente insiemi delle differenze, ogni differenza si verifica al massimo una

volta, cioè o non si verifica mai oppure si trova solo in un caso.

Per esempio, l'insieme $A = \{1, 2, 4\}$ forma un insieme delle differenze perfetto per il gruppo additivo di dimensione 7, poiché ogni intero da 1 a 6 può essere espresso come la differenza di soli due elementi nell'insieme delle differenze A, dunque si ottiene:

$$1-2 \pmod 7 = 6, \quad 1-4 \pmod 7 = 4; \quad 2-1 \pmod 7 = 1,$$
$$2-4 \pmod 7 = 5, \quad 4-1 \pmod 7 = 3; \quad 4-2 \pmod 7 = 2.$$

Questa matrice $\mathbf{C}_m$ viene poi utilizzata per costruire una *matrice di base* $\mathbf{H}_0$, che è una concatenazione di $\mathbf{C}_m$ e della sua trasposta, cioè si ottiene:

$$\mathbf{H}_0 = \left(\mathbf{C}_m \,|\, \mathbf{C}_m^T\right).$$

Di conseguenza, $\mathbf{H}_0$ rappresenta un codice classico dual-containing che soddisfa la condizione di even overlap, perché ogni sovrapposizione che si verifica in $\mathbf{C}_m$ può anche essere trovata in $\mathbf{C}_m^T$. Inoltre, poiché $\mathbf{H}_0$ è una matrice $(n/2 \times n)$, il risultante codice QLDPC dual-containing ha un rate di codifica $R_Q = 0$ (o equivalentemente $R_c = 1/2$), poiché $k = 0$. Per ottenere un rate di codifica diverso da zero, si scartano k righe di $\mathbf{H}_0$, in modo che i pesi delle colonne della PCM risultante $\mathbf{H}'_z$, di dimensioni $(n/2 - k) \times n$, siano il più possibile uniformi. Questo progetto di codice offre flessibilità nella scelta dei parametri, come per esempio ρ, n e k. Tuttavia, la distanza minima del codice risultante è limitata superiormente da ρ. Questo accade perché le k righe scartate di $\mathbf{H}_0$ sono tutte parole di peso ρ, che non sono contenute nel duale, e quindi contribuiscono a formare i modelli di errore non rilevati dannosi.

MacKay *et al.* hanno anche proposto codici a ciclo unico (o monociclici) in [11], che derivano dagli *insiemi delle differenze perfetti*. La proprietà fondamentale dell'insieme delle differenze perfetto implica che tutte le coppie di righe della PCM devono avere una sola sovrapposizione di 1. Poiché si ha bisogno della condizione di even overlap per ottenere un codice dual-containing, la PCM viene estesa con l'aggiunta di una colonna supplementare composta da tutti 1 (chiamata anche colonna "all-one"). Quindi, ogni coppia di righe nella PCM risultante ha due 1 sovrapposti, che si traducono in un ciclo di lunghezza 4 tra ogni coppia di righe. Così, una PCM di dimensioni (n, k) viene trasformata in una PCM dual-containing di dimensioni $(n + 1, k + 1)$, che ha peso della riga pari a $(\rho + 1)$ (dove ρ è il peso della riga della matrice iniziale e deve essere dispari) e i cui pesi delle colonne sono tutti ρ, tranne l'ultima colonna con tutti 1.

MacKay *et al.* hanno anche osservato che la struttura dei codici a ciclo unico può essere sfruttata per evitare i cicli di lunghezza 4 durante la procedura di decodifica [11]. Più esplicitamente, un codice monociclico può essere visto come la sovrapposizione di due codici; cioè uno di questi due codici ha, ad un'estremità della propria PCM, una colonna composta soltanto da 0s (chiamata anche colonna "all-zero") e l'altro è caratterizzato da una PCM

con una colonna all-one. Per evitare i cicli brevi, ciascuno dei due codici viene decodificato separatamente utilizzando l'algoritmo sum-product di [11]. Se entrambi i decodificatori restituiscono una parola di codice valida, viene scelta la parola di codice con il criterio della massima verosimglianza.

Tuttavia, si può sviluppare una procedura di decodifica migliore, al costo di una maggiore complessità di decodifica. Inoltre, la distanza minima dei codici a ciclo unico, costruiti utilizzando la tecnica degli insiemi delle differenze, è limitata superiormente dal peso della riga, poiché il codice risultante ha parole di codice di peso ρ, che non si trovano nel suo duale. Dato che la scelta di n, k e ρ per gli insiemi delle differenze perfetti è limitata, questo approccio di progetto non offre molta flessibilità nella scelta dei parametri di codice. Al contrario, i codici a doppio ciclo possono essere costruiti da qualsiasi codice LDPC ciclico classico arbitrario.

4.1.4 Codici QLDPC di tipo CSS a geometria euclidea

Per estendere l'applicazione dei codici a ciclo unico di MacKay a una gamma più ampia di parametri di codice, Aly in [31] ha sfruttato i codici LDPC classici a Geometria Euclidea (EG) di tipo II, tratti dal lavoro [32] di Fossorier *et al.*.

Approfondimento: Codici LDPC classici a Geometria Euclidea Le Geometrie Euclidee sui campi finiti formano due grandi famiglie di geometrie finite. Le strutture di questa famiglia di geometrie finite sono state ben studiate in [33], [34], [35]. Sulla base della teoria delle Geometrie Euclidee, si possono costruire due classi di codici LDPC ciclici o quasi ciclici a geometria finita. Essi sono:

1) i codici LDPC a Geometria Euclidea di tipo I;

2) i codici LDPC a Geometria Euclidea di tipo II.

Nel seguito verrà descritto il tipo II, cioè la classe di codice presente in questo elaborato.

Sia $\mathrm{EG}(m, 2^s)$ una Geometria Euclidea sul Campo di Galois di ordine 2^s ($\mathrm{GF}(2^s)$), dove m e s sono due numeri interi positivi. Per ulteriori approfondimenti sul Campo di Galois, si consulti l'*Approfondimento: Campo di Galois* della Sezione 2.3.2. Questa geometria consiste di 2^{ms} punti, dove ogni punto è una semplice m-upla su $\mathrm{GF}(2^s)$. Le m-uple all-zero, cioè $\mathbf{0} = (0, 0, \ldots, 0)$, sono chiamate origine. Invece, una linea in $\mathrm{EG}(m, 2^s)$ consiste in 2^s punti, dunque ci sono $2^{(m-1)s}(2^{ms} - 1)/(2^s - 1)$ linee in $\mathrm{EG}(m, 2^s)$.

La matrice $\mathbf{H}_{\text{EG-II}}(m, s)$ di un codice LDPC a Geometria Euclidea di tipo II è caratterizzata da 2^{ms-1} righe e

$$J = \left(2^{(m-1)s} - 1\right)(2^{ms} - 1)/(2^s - 1)$$

colonne. Le righe della matrice corrispondono ai punti non nulli di $\mathrm{EG}(m, 2^s)$ e le colonne corrispondono alle linee di $\mathrm{EG}(m, 2^s)$ che non passano per l'origine. Il peso della colonna e della riga sono $\gamma = 2^s$ e $\rho = (2^{ms}-1)/(2^s-1)-1$,

rispettivamente. Ogni coppia di righe di questa matrice ha esattamente un 1 in comune e ogni coppia di colonne ha al massimo un 1 in comune. Allora $C_{EG-II}(m, s)$ è un codice LDPC regolare di lunghezza J, chiamato codice LDPC-EG m-dimensionale di tipo II ed ha distanza minima $d_{EG-II}(m, s) = 2^s + 1$. Inoltre, per $m = 2$ il codice $C_{EG-II}(m, s)$ rivela una struttura ciclica, mentre per $m \neq 2$, $C_{EG-II}(m, s)$ è caratterizzato da una struttura non ciclica, ma può facilmente essere reso quasi ciclico [32].

Analogamente agli insiemi delle differenze perfetti, un codice LDPC classico EG di tipo II è rappresentato da una PCM $\mathbf{H}_{\text{EG-II}}$ che ha la caratteristica unica per la quale tutte le coppie di righe hanno un'unica sovrapposizione di 1. Di conseguenza, Aly ha suggerito che il codice caratterizzato dalla matrice $\mathbf{H}_{\text{EG-II}}$, di dimensioni $(n-k) \times n$, può essere convertito in un codice dual-containing in uno dei seguenti due modi:

1) Se il peso della riga di $\mathbf{H}_{\text{EG-II}}$ è dispari, come accade per i codici a ciclo unico, allora viene aggiunta in coda alla matrice una colonna all-one ($\mathbf{1}$), cioè si ottiene:

$$\mathbf{H}'_z = \left(\mathbf{H}_{\text{EG-II}} \,|\, \mathbf{1}\right).$$

2) Se il peso della riga di $\mathbf{H}_{\text{EG-II}}$ invece è pari, come nel caso precedente, per garantire la condizione di even overlap, si inserisce la colonna ($\mathbf{1}$), inoltre viene aggiunta anche, in coda alla PCM, una matrice identità $\mathbf{I}$ di dimensioni $(n-k) \times (n-k)$, per rendere il peso della riga uniforme. Dunque si ha:

$$\mathbf{H}'_z = \left(\mathbf{H}_{\text{EG-II}} \,|\, \mathbf{1} \,|\, \mathbf{I}\right).$$

I codici risultanti offrono rate di codifica elevati. Tuttavia, hanno una distanza minima limitata superiormente dal valore $(\lambda + 1)$, dove λ denota il peso della colonna.

4.1.5 Codici QLDPC di tipo CSS basati su BIBD

Djordjevic in [36] ha ulteriormente approfondito la costruzione dei codici a ciclo unico per la progettazione dei codici QC-QLDPC (Quasi-Cyclic-QLDPC) dual-containing con rate alto, a partire da codici LDPC classici basati sulla tecnica di costruzione Balanced Incomplete Block Design (BIBD) [37], [38], che hanno una distanza minima di almeno $(\lambda + 1)$. Più specificamente, la costruzione BIBD è caratterizzata significativamente dal parametro λ.

Approfondimento: Balanced Incomplete Block Design – BIBD La costruzione BIBD di parametri (ν, b, r, k, λ), indicato con BIBD(ν, b, r, k, λ), distribuisce tutti i ν elementi (o punti) di un insieme V in b sottoinsiemi (o blocchi) di dimensione k, in modo tale che:

- ogni coppia di elementi si trovi esattamente in λ blocchi;
- ogni elemento si trovi esattamente in r blocchi;
- il numero di elementi k in ogni blocco è piccolo rispetto alla dimensione ν dell'insieme V; pertanto, riceve il nome di *incompleto*.

Consideriamo dunque un insieme V di sette numeri, dato da $V = \{1, 2, 3, 4, 5, 6, 7\}$. Quindi, i blocchi $\{1, 2, 4\}$, $\{2, 6, 5\}$, $\{3, 4, 6\}$, $\{4, 5, 7\}$, $\{1, 5, 6\}$, $\{2, 6, 7\}$ e $\{1, 3, 7\}$ costituiscono la costruzione BIBD(7, 7, 3, 3, 1) poiché ci sono 7 elementi (ν) nell'insieme V che sono distribuiti tra 7 blocchi (b), inoltre ogni elemento appare in 3 blocchi (r), costituito ognuno da 3 elementi (k) e ogni coppia di elementi si trova in 1 blocco (λ) [28].

Un codice LDPC basato su BIBD ha esattamente λ sovrapposizioni tra ogni coppia di righe. Dato che i codici LDPC classici good devono avere al massimo una sola sovrapposizione per ogni riga, il fattore λ è impostato a 1 per la progettazione di codici LDPC classici con un girth di almeno 6. Di conseguenza, analogamente all'insieme delle differenze perfetto su cui sono basati i codici LDPC classici, ogni coppia di righe ha una sola sovrapposizione di 1, che può anche essere ottenuta imponendo la struttura del codice a ciclo unico sulla PCM. Djordjevic ha anche progettato i codici QLDPC dual-containing utilizzando tecniche di BIBD, in cui il fattore λ associato è pari. Sfortunatamente però, i codici QLDPC basati su un λ pari hanno prestazioni peggiori rispetto a quelli a ciclo unico basati su BIBD [36].

Dal momento che tutte le suddette costruzioni di codice QLDPC dual-containing hanno portato a una distanza minima limitata superiormente, la ricerca per la costruzione di codici QLDPC non limitati, nel senso di cui sopra, è continuata. Perseguendo questo obiettivo, MacKay *et al.* in [39] hanno proposto un'altra classe di codici QLDPC dual-containing, derivata dai grafi di Cayley. Questi codici sono stati ulteriormente esaminati da Couvreur *et al.* in [40] e [41], dove è stato formalmente dimostrato che il limite inferiore, che caratterizza la distanza minima del codice risultante, è una funzione logaritmica della lunghezza del codice. Quindi la distanza minima può essere migliorata estendendo la lunghezza della parola di codice (o del blocco), anche se, anche in questo caso, solo logaritmicamente. Tuttavia, questo si ottiene al costo di una maggiore complessità di decodifica imposta dal peso della riga crescente, poiché anch'esso aumenta logaritmicamente con la lunghezza del blocco. Inoltre, i progetti di codice basati sui grafi di Cayley possono essere visti come una classe speciale dei codici topologici [42], [43] e [44], i quali sono già noti per avere distanze minime crescenti. È presente un accenno sui codici topologici (surface code) nell'Appendice B.

Si ricorda in seguito che i codici QLDPC dual-containing hanno cicli brevi inevitabili, che compromettono le prestazioni dell'algoritmo di decodifica. Quindi, anche se venissero progettati dei codici QLDPC dual-containing con una distanza minima illimitata, è improbabile che superino le prestazioni delle loro controparti non dual-containing.

4.1.6 Codici QLDPC di tipo CSS basati su codici LDGM

Lou e Garcia-Frias in [45], [46] hanno riacceso l'interesse per i codici QLDPC di tipo CSS non dual-containing, servendosi dei codici LDGM (Low-Density

Generator-Matrix) per la costruzione del codice quantum finale.

Approfondimento: Codici classici Low-Density Generator-Matrix (LDGM) I codici duali degli LDPC sono gli LDGM, pertanto, dalla teoria dei codici duali ne deriva che la matrice di parità del generico codice LDPC corrisponde con la matrice generatrice del codice LDGM (suo duale) associato.

La differenza sostanziale di questa famiglia di codici rispetto ai codici LDPC risiede nel fatto che questi vengono identificati non dalla matrice PCM $\mathbf{H}$, bensì direttamente dalla matrice generatrice $\mathbf{G}$. Per ulteriori approfondimenti sulla matrice di parità e sulla matrice generatrice, si consulti la Sezione 2.1. Fissati i parametri di ingresso n e k, viene generata in modo arbitrario una matrice generatrice $\tilde{\mathbf{G}}$ molto sparsa ("very sparse") in forma sistematica, come segue:

$$\tilde{\mathbf{G}} = [\mathbf{I}_n \ \mathbf{M}],$$

dove $\mathbf{I}_n$ denota la matrice identità di ordine n, mentre la matrice $\mathbf{M}$ è una matrice binaria di dimensioni $n \times (k - n)$.

Infine, si osserva che i codici LDGM appartengono alla famiglia dei codici "bad".

In particolare, se sia la matrice generatrice $\mathbf{G}$ che la PCM $\mathbf{H}$ di un codice LDGM sono sparse, entrambe possono essere usate come componenti di un codice CSS. Siano dunque $\tilde{\mathbf{G}}$ la matrice generatrice e $\tilde{\mathbf{H}}$ la PCM, rispettivamente, di un codice LDGM (n, k). Quindi il codice CSS risultante può essere formulato come segue:

$$\mathbf{H} = \begin{pmatrix} \tilde{\mathbf{H}} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{G}} \end{pmatrix}. \tag{4.2}$$

Dato che $\tilde{\mathbf{H}}$ è una matrice di dimensioni $(n - k) \times n$, mentre $\tilde{\mathbf{G}}$ è una matrice $k \times n$, la PCM $\mathbf{H}$ risultante è una matrice di dimensioni $n \times 2n$. Di conseguenza, il corrispondente codice QLDPC ha rate di codifica pari a zero, come prevedibile, dato l'*Approfondimento* di cui sopra.

Lou e Garcia-Frias [45], [46] hanno suggerito che ciò può essere evitato applicando operazioni lineari di riga sia a $\tilde{\mathbf{G}}$ che ad $\tilde{\mathbf{H}}$ per ridurre il numero di righe. Sfortunatamente, questa riduzione di riga può, a sua volta, creare cicli brevi nella PCM risultante. Per evitare l'impatto negativo di questi cicli brevi, Lou e Garcia-Frias hanno anche concepito un grafo di Tanner modificato, che richiede la tecnica del "code doping", spiegata in [47], per facilitare il processo di decodifica iterativo verso la convergenza. Pertanto, viene raggiunta una prestazione migliore al costo di una maggiore complessità di decodifica.

4.1.7 Codici QC QLDPC e SC QC QLDPC di tipo CSS

Purtroppo, i codici costitutivi di tutti i progetti di tipo CSS citati fin qui, sia nel caso dual-containing che in quello non dual-containing, soffrono della presenza di cicli di lunghezza 4. Per eliminare questi cicli brevi, Hagiwara e Imai in [48] hanno concepito una classe unica di codici QC-QLDPC

(Quasi-Cyclic Quantum Low-Density Parity-Check) di tipo CSS non dual-containing, che hanno girth almeno pari a 6. In particolare, si consideri una matrice di permutazione circolante $\mathbf{T}$ che ha dimensioni $(LP/2 \times LP/2)$, $\rho = L/2$ e $\gamma = L$, che è data da [48]:

$$\mathbf{T} = \begin{pmatrix} t_0 & t_1 & \dots & t_{L/2-1} \\ t_{L/2-1} & t_0 & \dots & t_{L/2-2} \\ \vdots & \vdots & & \vdots \\ t_1 & t_2 & \dots & t_0 \end{pmatrix}, \tag{4.3}$$

dove t_i denota l'indice della matrice di permutazione circolante, le cui dimensioni sono $P \times P$ e $t_i \in [P_\infty] := \{0, 1, ..., P-1\} \cup \{\infty\}$.

Approfondimento: Matrice di permutazione circolante e codici Quasi-Cyclic LDPC Una matrice di permutazione è una matrice dove le righe e le colonne hanno peso unitario, dunque può essere ricavata dalla permutazione delle righe (colonne) di una matrice identità e per questo spesso è indicata con $\mathbf{I}(1)$. La matrice identità $\mathbf{I}$ è pertanto essa stessa una matrice di permutazione. Una matrice di permutazione circolante, chiamata anche più semplicemente matrice circolante, è una matrice di permutazione quadrata in cui ogni riga può essere ricavata operando uno shift (tipicamente ciclico, unitario e verso destra) della riga precedente; dunque l'informazione è racchiusa unicamente nella prima riga e nella regola di shift. Anche in questo caso, la matrice identità $\mathbf{I}$ è una matrice di permutazione circolante.

Si ha allora una matrice di permutazione $\mathbf{I}(1)$ di dimensione $P \times P$ che è data da:

$$\mathbf{I}(1) = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & & & \vdots \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}.$$

Più esplicitamente, $\mathbf{I}(1)$ è una matrice identità di dimensioni $P \times P$ spostata a destra di una posizione. Pertanto, $\mathbf{I}(x) := \mathbf{I}(1)^x$ può essere definita come una matrice identità $P \times P$ spostata a destra di $x \in \mathbb{N}$ posizioni, dove x è noto appunto come indice della matrice di permutazione. Inoltre, per $x = 0$ si definisce $\mathbf{I}(0)$, cioè la matrice identità tradizionale (senza spostamenti), mentre se $x = \infty$, allora la notazione $\mathbf{I}(\infty)$ viene usata per denotare una matrice nulla (all-zero) di dimensione $P \times P$.

È data, inoltre, la matrice $\mathbf{H}_\mathrm{c}$ su $[0, P-1]_\mathbb{Z} \cup \{\infty\}$, i cui elementi sono $(c_{j,l})$ con $0 \leq j \leq d_l$ e $0 \leq l \leq d_r$, cioè si ha:

$$\mathbf{H}_\mathrm{c} = \begin{bmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,d_r-1} \\ c_{1,0} & c_{1,1} & \dots & c_{1,d_r-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{d_l-1,0} & c_{d_l-1,1} & \dots & c_{d_l-1,d_r-1} \end{bmatrix}.$$

Sia allora $\mathbf{H}_C$ una PCM binaria di dimensioni $Pd_l \times Pd_r$, definita come segue:

$$\mathbf{H}_C := (\mathbf{I}(c_{j,l}))_{0\leq j<d_l,\, 0\leq l<d_r}\,, \tag{4.4}$$

dove $[0, P-1]_{\mathbb{Z}} := \{0, 1, \ldots, P-1\}$. Così $\mathbf{H}_{\mathrm{c}}$ è conosciuta come *matrice degli indici* di $\mathbf{H}_C$. Dunque, ci si riferisce alla matrice $\mathbf{H}_C$ come a una *matrice di permutazione quasi-ciclica* (QC) di parametri (d_l, d_r, P), o semplicemente *matrice di permutazione QC*. Esplicitando la (4.4), si ottiene:

$$\mathbf{H}_C = \begin{bmatrix} \mathbf{I}(c_{0,0}) & \mathbf{I}(c_{0,1}) & \ldots & \mathbf{I}(c_{0,d_r-1}) \\ \mathbf{I}(c_{1,0}) & \mathbf{I}(c_{1,1}) & \ldots & \mathbf{I}(c_{1,d_r-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{I}(c_{d_l-1,0}) & \mathbf{I}(c_{d_l-1,1}) & \ldots & \mathbf{I}(c_{d_l-1,d_r-1}) \end{bmatrix}.$$

Il codice lineare associato a questa matrice è chiamato *codice QC LDPC*.

Un codice QC-LDPC si dice *codice*(d_l, d_r, P) QC-LDPC se la matrice degli indici $\mathbf{H}_{\mathrm{c}}$ ad esso associata è caratterizzata da d_l righe e d_r colonne ed il grado della relativa matrice circolante $\mathbf{I}(1)$ è pari a P.

Hagiwara *et al.*, poi hanno dimostrato che $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono derivate dalla matrice $\mathbf{T}$ della (4.3) e soddisfano il criterio del prodotto simplettico, se hanno la seguente forma:

$$\begin{aligned} \mathbf{H}'_z &= (\mathbf{T}_1,\, \mathbf{T}_2) \quad \text{e} \\ \mathbf{H}'_x &= (-\mathbf{T}_2^T,\, -\mathbf{T}_1^T). \end{aligned} \tag{4.5}$$

Inoltre, poiché la cancellazione della riga è un'operazione ammessa dal criterio del prodotto simplettico, possono essere cancellate sia le righe di $\mathbf{H}'_z$ che di $\mathbf{H}'_x$, per ottenere il rate di codifica desiderato. Così, per garantire un girth di valore 6, Hagiwara *et al.* si basano sulla teoria dell'algebra combinatoria al fine di progettare le matrici circolanti $\mathbf{T}_1$ e $\mathbf{T}_2$, che costituiscono $\mathbf{H}'_x$ e $\mathbf{H}'_z$, in modo che tutte le righe di $\mathbf{H}'_x$ e $\mathbf{H}'_z$ abbiano al massimo una singola sovrapposizione [48].

I codici a doppio ciclo di [11] possono essere considerati un caso particolare della costruzione della (4.5), ad esempio quando $P = 1$ e $\mathbf{T}_2 = \mathbf{T}_1^T$. Sfortunatamente però, i codici risultanti non hanno superato le prestazioni dei codici a doppio ciclo di MacKay [11] e la loro distanza minima è, di nuovo, limitata superiormente dal peso della riga.

Tra tutti i codici dual-containing discussi in precedenza, la costruzione a doppio ciclo di MacKay in [11] è quella che offre la prestazione migliore, con una complessità di decodifica accettabile. Tuttavia, tale prestazione risultante non è ancora alla pari con quella dei codici LDPC classici. Ad esempio, il codice a doppio ciclo con rate 1/4 di [11], con $n = 19\,014$, funziona entro circa 5.5 dB dall'Hashing bound, con un WER (Word Error Rate) pari a 10^{-3}. Inoltre, tutti i codici menzionati precedentemente hanno una distanza minima limitata superiormente ad eccezione di quelli basati

sui grafi di Cayley. Nel tentativo di aumentare la distanza minima e quindi di avvicinare le prestazioni del codice alla capacità del canale (Hashing bound), Hagiwara *et al.* hanno esteso il progetto dei codici QC di [48] ai codici Spatially-Coupled (SC) in [49], che sono caratterizzati da prestazioni migliori rispetto alle loro corrispondenti controparti "non-coupled", al costo di una piccola perdita a livello di rate di codifica.

Approfondimento: Matrici con modello a banda sparsa e codici LDPC Spatially-Coupled In questa sede, si fornisce la costruzione dei codici classici SC LDPC a partire dalle matrici con *modello a banda sparsa* $\mathcal{H}$. Siano d_l, d_t e n_c dei numeri interi positivi. L'intero d_l è il peso della colonna di $\mathcal{H}$. Sia poi n_s un numero intero positivo tale che n_s divida d_l. La struttura di $\mathcal{H}$ è caratterizzata da delle sottomatrici di dimensione $d_l \times d_t$, che appaiono n_c volte sulla diagonale principale. Così il peso della colonna di $\mathcal{H}$ è d_l. Dunque, si definisce $d_r := d_t d_l / n_s$. Il peso della riga di $\mathcal{H}$ è d_r nel mezzo e d_t nella parte superiore e in quella inferiore. Inoltre, si definisce $M := d_l + (n_c - 1)n_s$ e $N := n_c d_t$. La dimensione di $\mathcal{H}$ è dunque $M \times N$.

Si dà nel seguito una costruzione formale per la matrice con modello a banda sparsa. Si definisce una matrice $\mathcal{H} = (h_{j,l})$ su $\{0, *\}$ di dimensioni $(d_l + (n_c - 1)n_s) \times (n_c d_t)$ ponendo $h_{j,l} := *$ se $bn_s + 1 \leq j \leq (b+1)n_s$, altrimenti $h_{j,l} = 0$, dove $*$ è un simbolo formale e $b := \lfloor (l-1)/dt \rfloor$. Con la notazione $\lfloor \cdot \rfloor$ si indica l'operazione di parte intera.

Il seguente è un esempio di matrice $\mathcal{H}$ con *modello a banda sparsa*, per i parametri $d_l = 4$, $d_t = 2$, $n_c = 12$ e $n_s = 1$:

$$\mathcal{H} = \left[\begin{array}{cccccccccccccccccccccc}
* & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * & * & * \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * & * & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * \\
\end{array}\right].$$

La matrice con modello a banda sparsa $\mathcal{H}$ contiene $n_c (= 12)$-sottoblocchi di dimensione $dl \times dt = 4 \times 2$.

Successivamente, si definisce un codice SC LDPC(d_l, d_r, n_c) tramite la matrice $\mathcal{H}$. Sia P un numero intero positivo. Si sostituisca poi ogni "$*$" della matrice $\mathcal{H}$ con una matrice di permutazione binaria di dimensione $P \times P$ e ogni 0 di $\mathcal{H}$ con una matrice all-zero di dimensione $P \times P$. La nuova matrice è chiamata matrice LDPC spatially-coupled e il suo kernel è

il codice *LDPC spatially-coupled classico*. L'insieme dei codici SC LDPC ha presenta prestazioni di decodifica con l'algoritmo SP (Sum-Product) molto vicine alle prestazioni di decodifica MAP (Maximum A Posteriori) [50], di un insieme di codici LDPC(d_l, d_t) regolari [51], nel limite della lunghezza di codice e del numero di accoppiamento n_c. Per una matrice con modello a banda sparsa $\mathcal{H}$, si può definire una matrice $\mathcal{H}' = (h'_{j,l})$ su $\{0, *\}$ ponendo $h'_{j,l} := h_{r-j+1,l}$, dove r è il numero di righe di $\mathcal{H}$. Dunque, anche $\mathcal{H}'$ è una matrice con modello a banda sparsa.

Il peso della riga di una PCM di un codice SC LDPC è ragionevolmente grande e le prestazioni di decodifica dell'algoritmo SP del codice associato sono eccezionali [49], per questo ci si aspetta che tali codici superino il problema sul peso della riga.

Di seguito, viene presentata per prima la costruzione dei modelli di codice presenti in [48] e poi viene approfondita con maggiore attenzione la costruzione di [49].

Sia $(\mathbf{H}'_z, \mathbf{H}'_x)$ un codice quantum LDPC di tipo CSS, dove $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono due matrici di parità associate a due codici classici C e D. Se $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono matrici SC LDPC, allora il codice QLDPC di tipo CSS si chiama *codice QLDPC di tipo CSS spatially coupled* (SC-QLDPC CSS code). È noto che i cicli brevi nel grafo di Tanner degradano le prestazioni di decodifica SP; tipicamente quelli di lunghezza 4. In questa sottosezione, si costruisce una coppia di matrici $(\mathbf{H}'_z, \mathbf{H}'_x)$ di tipo SC LDPC che soddisfa le tre seguenti condizioni:

1) $\mathbf{H}'_z \mathbf{H}'^T_x = 0$;

2) I grafi di Tanner di $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono privi di cicli di lunghezza 4;

3) $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono matrici di permutazione QC.

In particolare, a valle di queste tre condizioni, si osserva che le PCM $\mathbf{H}'_z$ e $\mathbf{H}'_x$ scelte saranno matrici SC LDPC basate su matrici di permutazione binarie QC. Allora il codice QLDPC di tipo CSS si chiama codice QLDPC di tipo CSS quasi cyclic spatially coupled (QC SC-QLDPC CSS code).

Si esamina innanzitutto la costruzione sviluppata in [48]. Nel documento originale [48], il metodo per costruire una coppia di PCM $\mathbf{H}'_z$ e $\mathbf{H}'_x$ è flessibile. Per semplicità, in questo elaborato, si concentra l'attenzione sul caso in cui $\mathbf{H}'_z$ e $\mathbf{H}'_x$ hanno lo stesso peso della riga d_r e lo stesso peso della colonna d_l. Siano $\mathbf{H}_\mathrm{c}$ e $\mathbf{H}_\mathrm{d}$, due matrici degli indici su $[0, P-1]_\mathbb{Z} \cup \{\infty\}$ di dimensione $d_l \times d_r$, i cui elementi sono $(c_{j,l})$ e $(d_{k,l})$, rispettivamente. Sia poi $\mathbf{H}'_z$ ($\mathbf{H}'_x$) una matrice QC associata alla relativa matrice degli indici $\mathbf{H}_\mathrm{c}$ ($\mathbf{H}_\mathrm{d}$).

In [48], si è dimostrato che $\mathbf{H}'_z \mathbf{H}'^T_x = 0$, se
$\#\{l \in [0, P-1]_\mathbb{Z} \mid c_{j,l} - d_{k,l} = p \bmod P\}$ è pari per tutti i valori di
$j \in [0, d_l - 1]_\mathbb{Z}$ e per tutti i valori di $p \in [0, P-1]_\mathbb{Z}$. In particolare con la notazione $\#A$ si intende la cardinalità del generico insieme A.

Il grafo di Tanner associato alla matrice $\mathbf{H}'_z$ è privo di cicli di lunghezza 4, se:

$$\#\{c_{j,l} - c_{k,l} \bmod P \mid l \in [0, d_r - 1]_\mathbb{Z}\} = d_r$$

per tutti i valori di $j, k \in [0, d_l - 1]_{\mathbb{Z}}$, tali che $j \neq k$. Si propone ora il seguente teorema che è una versione generalizzata del Teorema 2.4 contenuto in [48].

Teorema 1

Sia P un intero positivo con $P > 2$. Si definisce poi $\mathbb{Z}_P^* := \{z \in [0, P-1]_{\mathbb{Z}} \,|\, \forall\, a \in [0, P-1]_{\mathbb{Z}},\ za = 1\}$. Per σ e $\tau \in \mathbb{Z}_P^*$, si definisce $\mathrm{ord}(\sigma) := \min\{m > 0 \mid \sigma^m = 1\}$, e si definisce $\langle \tau \rangle_\sigma = \{\tau, \tau\sigma, \ldots, \tau\sigma^{\mathrm{ord}(\sigma)-1}\}$.

Siano inoltre d_l, d_r interi e τ_1, τ_2 e $\sigma \in \mathbb{Z}_P^*$ tali che:

$$\begin{aligned}
&\mathrm{ord}(\sigma) \neq \#\mathbb{Z}_P^*;\\
&d_l \geq 2,\ \ d_r \geq 4,\ \ d_r/2 = \mathrm{ord}(\sigma),\ \ 1 \leq d_l \leq \mathrm{ord}(\sigma);\\
&1 - \sigma^j \in \mathbb{Z}_P^*\ \forall\ 1 \leq j < \mathrm{ord}(\sigma);\\
&\tau_2 \notin \langle \tau_1 \rangle_\sigma.
\end{aligned}$$

Siano infine $\mathbf{H}'_z$ e $\mathbf{H}'_x$ due matrici binarie quasi cicliche di parametri (d_l, d_r, P), tali che:

$$\begin{aligned}
\mathbf{H}'_z &= (\mathbf{I}(c_{j,l}))_{0 \leq j < d_l,\, 0 \leq l < d_r},\\
\mathbf{H}'_x &= (\mathbf{I}(d_{j,l}))_{0 \leq j < d_l,\, 0 \leq l < d_r},
\end{aligned}$$

dove

$$c_{j,l} := \begin{cases} \tau_1 \sigma^{-j+l}, & 0 \leq l < d_r/2 \\ \tau_2 \sigma^{-j+l}, & d_r/2 \leq l < d_r, \end{cases}$$

$$d_{j,l} := \begin{cases} -\tau_2 \sigma^{j-l}, & 0 \leq l < d_r/2 \\ -\tau_1 \sigma^{j-l}, & d_r/2 \leq l < d_r. \end{cases}$$

Quindi, si ottiene che $\mathbf{H}'_z \mathbf{H}'^T_x = 0$ e che entrambi i grafi di Tanner di $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono privi di cicli di lunghezza 4.

Osservazione Se si pone $\tau_1 = 1$, l'enunciato è lo stesso del Teorema 2.4 contenuto in [48].

Dal **Teorema 1**, si ottengono due matrici binarie $\mathbf{H}'_z$ e $\mathbf{H}'_x$, di dimensioni $d_l P \times d_r P$, tali che $\mathbf{H}'_z$ e $\mathbf{H}'^T_x = 0$ e che i loro grafi di Tanner siano privi di cicli di lunghezza 4. In questo documento, ci si riferisce a questi codici come codici QC-QLDPC di tipo CSS convenzionali.

*Esempio: Applicazione del **Teorema 1*** Scegliendo come parametri $d_l = 3$, $d_r = 6$, $P = 7$, $\sigma = 2$, $\tau_1 = 1$ e $\tau_2 = 3$, dal **Teorema 1**, si ottiene una coppia di matrici binarie $(\mathbf{H}'_z, \mathbf{H}'_x)$ di dimensioni $d_l P \times d_r P$ tali che $\mathbf{H}'_z$

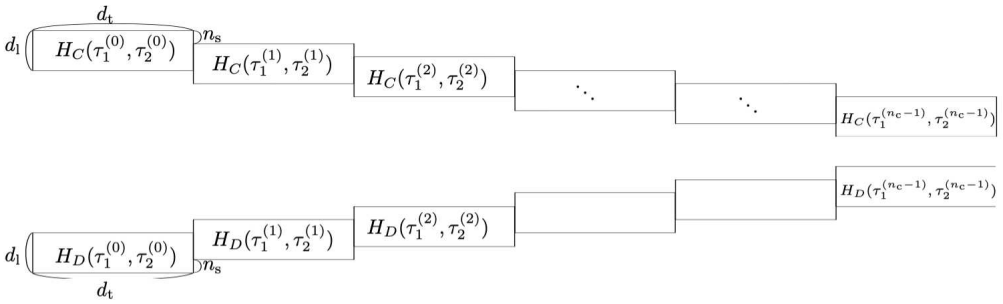

Figura 4.4: Illustrazione delle PCM $\mathbf{H}'_z$ e $\mathbf{H}'_x$ di un codice SC QLDPC di tipo CSS [49].

e $\mathbf{H}'^T_x = 0$ come segue:

$$\mathbf{H}'_z = \begin{pmatrix} I(1) & I(2) & I(4) & I(3) & I(6) & I(5) \\ I(4) & I(1) & I(2) & I(5) & I(3) & I(6) \\ I(2) & I(4) & I(1) & I(6) & I(5) & I(3) \end{pmatrix},$$

$$\mathbf{H}'_x = \begin{pmatrix} I(4) & I(2) & I(1) & I(6) & I(3) & I(5) \\ I(1) & I(4) & I(2) & I(5) & I(6) & I(3) \\ I(2) & I(1) & I(4) & I(3) & I(5) & I(6) \end{pmatrix}.$$

4.1.7.1 Codici quantum SC di tipo CSS

Nel **Teorema 1** le matrici $\mathbf{H}'_z(\tau_1, \tau_2)$ e $\mathbf{H}'_x(\tau_1, \tau_2)$ rappresentano una coppia di PCM associate a τ_1 e τ_2. Siano quindi $\mathbf{H}'_z(\tau_1^{(i)}, \tau_2^{(i)})$ e $\mathbf{H}'_x(\tau_1^{(i)}, \tau_2^{(i)})$ coppie di matrici di numerosità n_s per $i = 0, \ldots, n_c - 1$. Sia inoltre n_s un numero positivo tale che n_s divida d_l. Si costruisce, pertanto, la coppia di matrici con modello a banda sparsa di tipo QC $\mathbf{H}'_z$ e $\mathbf{H}'_x$, come raffigurato nella Fig. 3.1. Dunque si ottiene che $\mathbf{H}'_z\mathbf{H}'^T_x = 0$ poiché $\mathbf{H}'_z(\tau_1^{(i)}, \tau_2^{(i)})\mathbf{H}'_x(\tau_1^{(i)}, \tau_2^{(i)})^T = 0$ per $i = 0, \ldots, n_c - 1$. Pertanto la condizione di ortogonalità espressa dalla relazione $\mathbf{H}'_z\mathbf{H}'^T_x = 0$ non dipende dalla scelta di $\tau_1^{(i)}$, $\tau_2^{(i)}$. Le matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$ scelte hanno il peso della colonna d_l regolare e peso della riga leggermente irregolare. Il peso della riga corrisponde tipicamente a $d_t d_l / n_s$ nelle zone centrali delle matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$ ed assume il valore minimo d_t nei pressi dei bordi.

D'altra parte, la condizione per la quale non ci sono cicli di lunghezza 4 nel grafico Tanner dipende dalla loro scelta, poiché $\mathbf{H}'_z(\tau_1^{(i)}, \tau_2^{(i)})$ e $\mathbf{H}'_z(\tau_1^{(i')}, \tau_2^{(i')})$ condividono una riga comune se $|i - i'| < d_l/n_s$. Similmente a quanto accade per la dimostrazione del **Teorema 1**, si ha il seguente teorema.

Teorema 2

Se $\langle \tau_b^i \rangle_\sigma \cap \langle \tau_b^{i'} \rangle_\sigma = \emptyset$ per qualsiasi $b, b' \in \{1, 2\}$ e $i, i' \in \{0, 1, \ldots, n_c - 1\}$ tali che $|i - i'| < d_l/n_s$, quindi $\mathbf{H}'_z\mathbf{H}'^T_x = 0$ e non ci sono cicli di lunghezza 4

nei grafi di Tanner delle matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$ di tipo QC con modello a banda sparsa.

Nella Fig. 4.5, è presente un esempio delle matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$ dei codici SC QLDPC di tipo CSS con i seguenti parametri: $d_l = 3$, $d_t = 6$, $P = 31$, $n_c = 6$, $n_s = 1$, $(\tau_0^{(0)}, \tau_1^{(0)}) = (16, 4)$, $(\tau_0^{(1)}, \tau_1^{(1)}) = (8, 12)$, $(\tau_0^{(2)}, \tau_1^{(2)}) = (6, 1)$, $(\tau_0^{(3)}, \tau_1^{(3)}) = (3, 11)$, $(\tau_0^{(4)}, \tau_1^{(4)}) = (17, 2)$ e $(\tau_0^{(5)}, \tau_1^{(5)}) = (6, 4)$.

Osservazione I grafi di Tanner per $\mathbf{H}'_z$ e $\mathbf{H}'_x$ sono isomorfi. Questa affermazione è dimostrata da un argomento combinatorio, anche se la prova è omessa in questo lavoro.

$H_C =$

```
16 18 28  4 20  7
28 16 18  7  4 20  8  9 14 12 29 21
18 28 16 20  7  4 14  8  9 21 12 29  6 30 26  1  5 25
                  9 14  8 29 21 12 26  6 30 25  1  5  3 15 13 11 24 27
                                   30 26  6  5 25  1 13  3 15 27 11 24 17 23 22  2 10 19
                                                  15 13  3 24 27 11 22 17 23 19  2 10  6 30 26  4 20  7
                                                              23 22 17 10 19  2 26  6 30  7  4 20
                                                                          30 26  6 20  7  4
```

$H_D =$

```
                                                                          27 24 11 25  5  1
                                                              29 12 21 14  9  8 11 27 24  1 25  5
                                                  20  4  7 28 18 16 21 29 12  8 14  9 24 11 27  5  1 25
                                   30  6 26 25  5  1  7 20  4 16 28 18 12 21 29  9  8 14
                  19 10  2 23 17 22 26 30  6  1 25  5  4  7 20 18 16 28
27 24 11 15  3 13  2 19 10 22 23 17  6 26 30  5  1 25
11 27 24 13 15  3 10  2 19 17 22 23
24 11 27  3 13 15
```

Figura 4.5: Illustrazione delle PCM $\mathbf{H}_C$ e $\mathbf{H}_D$ dei codici SC QLDPC di tipo CSS proposti e definite dai seguenti parametri: $d_l = 3$, $d_t = 6$, $P = 31$, $n_c = 6$, $n_s = 1$. Le matrici $\mathbf{I}(\cdot)$ sono abbreviate per ogni ingresso [49].

4.1.7.2 Risultati Numerici

Si suppone che la trasmissione avvenga su un canale di depolarizzazione [11] con probabilità di depolarizzazione pari a $2f_m/3$, dove f_m può essere vista come la probabilità marginale per gli errori di Pauli $\mathbf{X}$ (bit-flip) e $\mathbf{Z}$ (phase-flip). Supponiamo che il decodificatore conosca il valore della probabilità di depolarizzazione $2f_m/3$.

La Fig. 4.6 confronta i codici SC QLDPC di tipo CSS proposti e i codici QC-QLDPC di tipo CSS convenzionali. Il rate di codifica quantum R_Q di questi codici è 0.50 e 0.49, rispettivamente. Tale perdita di rate è dovuta alla costruzione coupled, ma può essere ridotta aumentando il valore di n_c. Le PCM di questi codici hanno lo stesso peso della colonna, pari a 10. I pesi minimi delle righe sono 40 e 20, rispettivamente. Sia per i codici di tipo CSS proposti che per quelli QC-QLDPC convenzionali, i due codici classici componenti C e D sono isomorfi; in altre parole, ogni coppia di PCM è la stessa fino alla permutazione di colonne e righe. Questo isomorfismo implica che C e D abbiano le stesse prestazioni medie di decodifica. Quindi, per semplicità, si limita l'analisi alle prestazioni di C.

A causa del grande peso minimo della riga della matrice di parità, tutti i codici non presentano errori fino ad un valore del bit error rate di

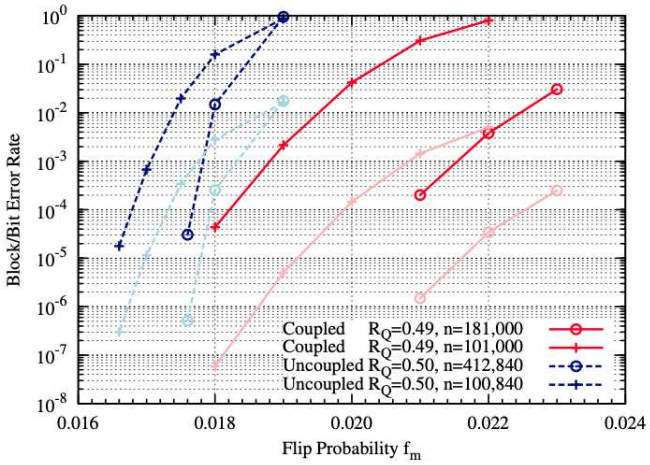

Figura 4.6: Confronto del rate di errore di decodifica del codice SC QLDPC di tipo CSS proposto (rosso) con parametri ($d_l = 10$, $d_t = 20$, $n_c = 50$, $n_s = 5$) e codici QC-QLDPC di tipo CSS (blu) con parametri ($d_l = 10$, $d_r = 40$). Il rate di codifica quantum è $R_Q = 0.49$ e $R_Q = 0.50$, rispettivamente. La lunghezza del codice è di n qubit. Il block error rate ed il bit error rate dei codici costituenti C e D della coppia di codici di tipo CSS proposta (C, D) sono rappresentati con il colore scuro e chiaro, rispettivamente. La probabilità di bit-flip marginale del canale di depolarizzazione e di phase-flip marginale è pari a f_m ed è associata agli errori di Pauli $\mathbf{X}$ e $\mathbf{Z}$, rispettivamente. A causa dell'isomorfismo, i rate di errore dei codici C e D sono identici [49].

10^{-6}. Non sono stati osservati errori non rilevati (undetected) per entrambi i codici. Si prevede quindi che le distanze minime di entrambi i codici siano sufficientemente grandi e vicine ai limiti superiori 40 e 20, rispettivamente. Inoltre, si osserva che, aumentando la lunghezza del codice da $n_1 = 100\,840$ a $n_2 = 412\,840$, i codici quantum QC-LDPC di tipo CSS convenzionali sono caratterizzati da un guadagno di codifica piccolo, pari a $\Delta_{conv} = n_2 - n_2 = 312\,000$. Il codice proposto in [49], invece, supera le prestazioni del codice convenzionale di cui sopra con un incremento della lunghezza di codice minore, passando da $n_1 = 101\,000$ a $n_2 = 181\,000$ ed ottenendo $\Delta_S C = n_2 - n_1 = 80\,000$. Questa prestazione eccellente compensa la perdita in termini di rate.

In conclusione, le simulazioni numeriche, condotte utilizzando per la trasmissione i canali di depolarizzazione, mostrano che i codici proposti in [49] hanno la regione dell'error-floor profonda e che la zona water-fall assicura prestazioni eccellenti. In altre parole, la soglia di saturazione è rispettata anche per i codici QLDPC di tipo CSS.

4.1.8 Considerazioni e confronti sui codici analizzati

Tuttavia, anche per questi ultimi codici considerati, le prestazioni sono rimaste relativamente lontane dall'Hashing bound. In particolare, l'SC QC-QLDPC di [49], con un rate di codifica di 0.49 e una lunghezza pari a $n = 181\,000$, opera entro circa 3.8 dB dall'Hashing bound con un WER di 10^{-3}.

Kasai *et al.* hanno ulteriormente contribuito a questi sviluppi ricavando i codici QC-QLDPC non binari in [52] e [53], partendo dalla progettazione dei codici di [48]. I codici risultanti hanno ottenuto prestazioni migliori rispetto alle loro controparti binarie, al costo di una maggiore complessità di decodifica. È stato dimostrato infatti, che un codice con rate 1/2, avente una lunghezza di $n = 20\,560$ e GF(2^{10}), opera fino a circa 1.9 dB dall'Hashing bound con un WER di 10^{-3}.

I codici SC sono stati ulteriormente esaminati da Andriyanova *et al.* in [54], dove i codici costituenti sono stati ricavati dai codici LDGM classici come in [45] e [46]. Analogamente ai codici quantum EA, Andriyanova *et al.* hanno supposto che alcuni qubit siano trasmessi attraverso un canale silenzioso. Di conseguenza, i codici SC-QLDPC risultanti, basati su codici LDGM, con rate di 1/4 e con lunghezza pari a $n = 76\,800$, sono riusciti a operare entro circa 1.7 dB dall'Hashing bound, ad un WER di 10^{-3}. L'ipotesi di avere dei qubit trasmessi attraverso un canale silenzioso è stata successivamente abbandonata in [55], poiché questi qubit sono stati protetti attraverso la tecnica di riduzione dell'errore dei QTC (Quantum Turbo Codes), che ha comportato solo una perdita modesta nel rate di codifica e un incremento moderato della complessità del codice complessivo [55]. È stato dimostrato che le prestazioni del codice SC-QLDPC risultante, con rate 1/2, assistito da QTC come spiegato sopra e basato sui codici LDGM, con una lunghezza pari a $n = 821\,760$, sono entro circa 0.7 dB dall'Hashing bound ad un WER di 10^{-3}.

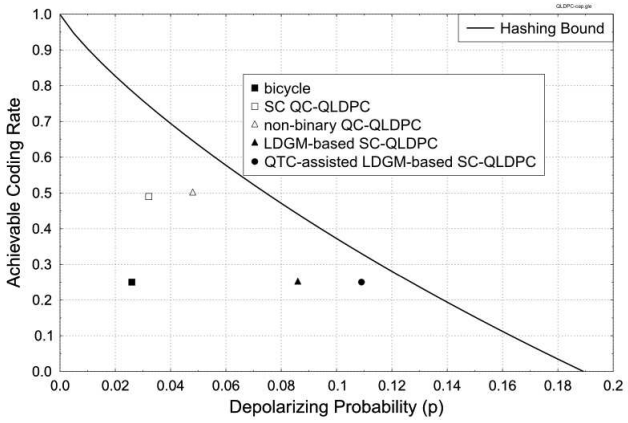

Figura 4.7: Rappresentazione grafica delle prestazioni ottenibili dei codici citati, con un WER nominale di 10^{-3}, confrontata con l'Hashing bound [28].

Infine, la Fig. 4.7 confronta le prestazioni ottenibili, rispetto all'Hashing bound, dei codici di cui sopra, vale a dire:

– il codice QLDPC a doppio ciclo di [11];

– il codice SC QC-QLDPC di [49];

– il codice QC-QLDPC non binario di [52] e [53];

– il codice SC-QLDPC basato sui codici LDGM di [54];

– il codice SC-QLDPC basato sui codici LDGM e assistito da QTC di [55], ad un WER pari a 10^{-3}.

Tutti i principali contributi relativi ai codici QLDPC basati sulla struttura CSS sono riassunti nella Fig. 4.3.

4.2 Codici QLDPC di tipo non CSS

I codici stabilizzatori non CSS sfruttano la ridondanza in modo più efficiente rispetto alle loro controparti CSS. Ad esempio, un codice a blocco di tipo CSS progettato per correggere un singolo errore di bit-flip o di phase-flip richiederebbe una lunghezza di blocco di 7 qubit [13], mentre sono necessari solo 5 qubit per realizzare un codice a blocco di tipo non CSS avente le stesse capacità di correzione [56]. Ulteriori spiegazioni sono presenti nella Sezione 2.4.

Di conseguenza, Camara *et al.* in [57], [58] hanno proposto la costruzione di codici QLDPC non CSS (chiamati anche semplicemente codici QLDPC). Contrariamente alla maggior parte dei codici dual-containing precedenti, che soddisfano il rigoroso criterio del prodotto simplettico nella loro struttura *globale*, il progetto ideato da Camara *et al.* mira a soddisfare il vincolo del prodotto simplettico nella struttura *locale* del singolo codice. In particolare, poiché la PCM di un codice LDPC classico quaternario può essere mappata seguendo l'isomorfismo dal dominio di Pauli al dominio classico quaternario, come spiegato nella Sezione 2.3.2, Camara *et al.* hanno sviluppato un approccio teorico, basato sulla teoria dei gruppi, per la costruzione

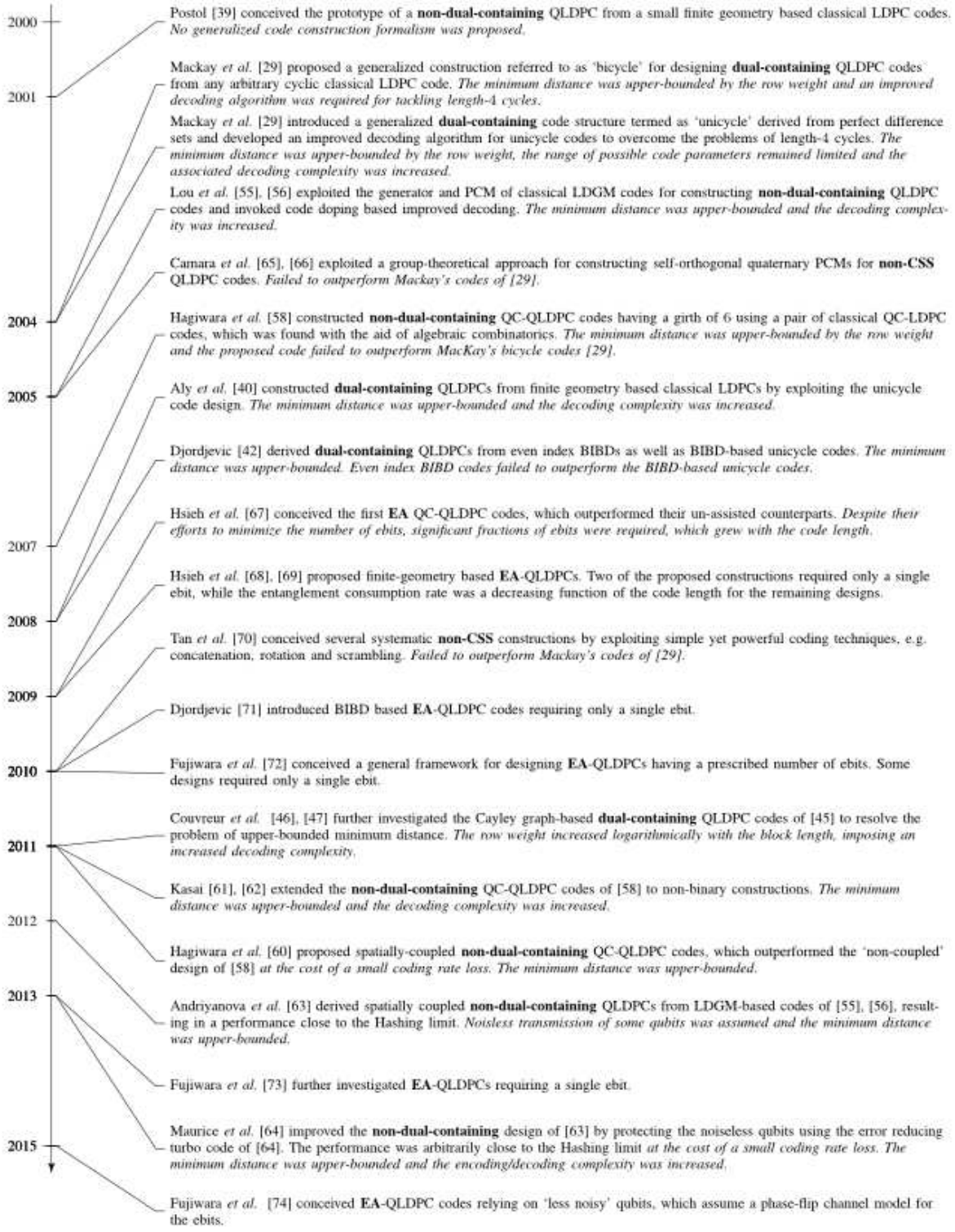

Figura 4.8: Maggiori contributi per lo sviluppo dei codici QLDPC. Il tipo di codice per ogni contributo è evidenziato in **grassetto**, mentre gli svantaggi associati ad ogni progetto proposto sono segnati in *corsivo* [28].

di codici LDPC quaternari auto-ortogonali, che soddisfino il criterio del prodotto simplettico. Si è notato che il grafo di Tanner della PCM associata al codice quaternario auto-ortogonale risultante ha cicli di lunghezza 4. Tuttavia, questi cicli brevi sono imposti dal vincolo di commutatività, dato dal prodotto simplettico. In particolare, ogni colonna di una PCM, associata ad un codice quaternario, deve contenere almeno due elementi diversi da zero, che siano differenti tra loro, cioè l'operatore di Pauli **X**, **Z**, o **Y** in modo che possa correggere sia gli errori di phase-flip che quelli di bit-flip che si verificano su quel qubit. Per maggiori approfondimenti sugli errori indotti dal canale di comunicazione quantum, si faccia riferimento al Capitolo 1.

D'altra parte, ogni coppia di righe della PCM deve avere un numero pari di posizioni occupate da elementi diversi da zero (o operatori di Pauli non di identità) e mutuamente non uguali. Ad esempio, si consideri una colonna con peso 2 di una PCM, composta da due righe, con elementi di valore 1 e ω, rispettivamente. Per soddisfare il vincolo di commutatività, queste due righe devono avere un'altra colonna sovrapposta con elementi mutuamente diversi e differenti da zero; si creano però cicli di lunghezza 4. Intuitivamente, questi cicli brevi sono presenti anche nella PCM **H** dei codici CSS, quando vengono rappresentati nel dominio quaternario. Infatti, questi cicli nei codici CSS dual-containing sono troppi, poiché, oltre ai cicli dovuti al criterio del prodotto simplettico, sono presenti giocoforza anche i cicli aggiuntivi, derivanti dal vincolo della costruzione dual-containing. Tali argomenti verranno ulteriormente approfonditi nella Sezione 5.3.

I codici QLDPC non CSS proposti in [57] e [58] hanno ottenuto risultati migliori, rispetto ai codici a doppio ciclo, nella regione waterfall della loro curva di prestazione, producendo però un error floor più elevato a causa della loro distanza minima ridotta. Si prevede che questa costruzione di tipo non CSS possa avere una distanza minima illimitata, producendo così error floor minori, quando la lunghezza del blocco è sufficientemente elevata. Tuttavia, ciò non è stato esplicitamente dimostrato né in [57] né in [58].

Perseguendo la stessa linea di indagine, Tan e Li in [59] sono stati i primi ricercatori a progettare le PCM $\mathbf{H}'_z$ e $\mathbf{H}'_x$ costituenti di un codice non CSS, utilizzando i codici binari classici. In particolare, hanno concepito diverse costruzioni sistematiche per codici QLDPC non CSS, che imponevano sia strutture globali che locali sui codici binari sottostanti per soddisfare il criterio del prodotto simplettico. Questo risultato si ottiene sfruttando tecniche di codifica semplici ma potenti, che includono la concatenazione, la rotazione e lo scrambling.

I codici così progettati presentano prestazioni migliori rispetto ai codici non CSS di [57] e [58]. Tuttavia, non hanno ancora superato i codici di MacKay di [11]. In conclusione, i principali traguardi raggiunti nel campo dei codici QLDPC di tipo non CSS sono riassunti nella Fig. 4.8.

4.3 Codici QLDPC di tipo Entanglement-Assisted

Esistono codici LDPC classici efficienti, che sono noti per avvicinarsi alla capacità di Shannon per blocchi di grandi dimensioni. Ad esempio, il codice LDPC classico ottimizzato con rate 1/2 di [60] funziona fino a 0.13 dB dal limite di capacità per la trasmissione su un canale AWGN con un BER (Bit Error Rate) di 10^{-6} ed utilizzando una lunghezza di codice di 10^6. In particolare, il picco massimo (turbo cliff) di questo codice LDPC è distante dal limite di Shannon solo per 0.06 dB.

Questo ha ispirato i ricercatori a ottenere prestazioni simili per i codici QLDPC. Sfortunatamente, però, il criterio del prodotto simplettico, o più specificamente il requisito di commutatività degli stabilizzatori, intrinseco al criterio stesso, limita l'applicazione diretta di tali codici classici efficienti nel dominio quantum. Come discusso nella Sezione 4.1 e 4.2, solo i codici classici di una certa categoria, che sono conformi a rigorosi vincoli strutturali, locali o globali, possono essere usati come componenti di un codice quantum. Questo ostacolo può essere superato sfruttando la struttura EA presentata in [17] e [61], che aiuta a importare nel dominio quantum qualsiasi codice classico. Per approfondire la nozione di codici EA, si consulti la Sezione 2.4. Infatti, i qubit precondivisi su canale non rumoroso (chiamati più brevemente ebit) di un codice EA costituiscono una risorsa tanto preziosa quanto costosa, poiché mantenere uno stato entangled senza rumore non è un compito banale. Di conseguenza, un progetto di codice praticamente realizzabile dovrebbe mirare a ridurre al minimo il numero degli ebit.

I primi codici EA-QLDPC sono stati concepiti da Hsieh *et al.* in [62], dove in particolare sono stati progettati dei codici QC-QLDPC di tipo CSS basati sulla tecnologia EA, a partire dalle loro controparti classiche. Hsieh *et al.* hanno scelto le matrici circolanti costituenti del codice QC classico, assicurando che il numero degli ebit richiesto sia ridotto al minimo. Nonostante i loro sforzi, si osserva che è richiesto un numero significativo di ebit, che cresce con la lunghezza del codice stesso; quindi maggiore sarà la lunghezza del codice e maggiore sarà il numero degli ebit necessari. Ciononostante, questi progetti sono stati portati avanti con l'idea che l'alta efficienza dei codici EA deve essere attribuita al numero degli ebit. Inoltre, dal momento che i codici quantum EA di [62] ereditano le stesse caratteristiche del codice classico da cui discendono, specialmente in termini di girth e distanza minima, questi codici EA-QLDPC hanno ottenuto prestazioni migliori rispetto agli attuali codici QLDPC non assistiti.

Lavorando ulteriormente con l'idea di ridurre al minimo il numero degli ebit, Hsieh *et al.* in [63] e [64] hanno concepito i codici EA-QLDPC basati sulla tecnica delle geometrie finite, il cui entanglement consumption rate, che tiene conto di quanto la risorsa entangled viene sfruttata, diminuisce con la lunghezza del codice. Inoltre, due di queste costruzioni di codice richiedono un solo ebit, indipendentemente dalla lunghezza del codice; eliminando così uno dei più grandi svantaggi che caratterizza la famiglia dei codici EA, cioè la restrizione sul numero degli ebit. Per di più, il progetto proposto non impone alcuna limitazione riguardo i codici LDPC classici a geometria

finita sottostanti, che sono presi dal lavoro di Fossorier *et al.* [32].

In [65] è stato presentato un quadro più generale per la progettazione dei codici EA-QLDPC, con un numero prescritto di ebit, derivato dalla teoria combinatoria. Alcuni di questi progetti richiedono solo un singolo ebit; pur assicurando alte prestazioni, alto rate di codifica e bassa complessità. Le condizioni necessarie e sufficienti per la progettazione dei codici EA-QLDPC a singolo ebit sono state ulteriormente studiate in [66]. Inoltre, i codici EA-QLDPC basati sulla tecnica BIBD, che richiedono un solo ebit, sono stati studiati anche in [67].

Successivamente, Fujiwara in [68] ha introdotto la nozione dei codici quantum che si basano sui qubit "less noisy", cioè meno rumorosi (o, altrimenti detti affidabili). In particolare, a differenza di quanto preteso dalla tecnica EA, che richiede degli ebit completamente silenziosi, il progetto di Fujiwara in [68] presuppone che questi qubit ausiliari siano trasmessi attraverso un canale di phase-flip, che è un modello di rumore realistico. In quest'ottica, Fujiwara *et al.* in [69] hanno concepito i codici QLDPC basandosi sui qubit less noisy. I principali contributi apportati nel settore dei codici EA-QLDPC sono riassunti nella Fig. 4.8.

Capitolo 5

Decodifica iterativa dei codici LDPC Quantum

Analogamente ai codici LDPC classici, i codici QLDPC possono essere decodificati utilizzando l'algoritmo Belief Propagation (BP) classico, di cui l'algoritmo Sum-Product (SPA − Sum-Product Algorithm) ne è una sua implementazione, che opera sul grafo di Tanner della corrispondente PCM. Tuttavia, come ampiamente motivato nel Capitolo 1, i qubit collassano al momento della misurazione (o osservazione). Pertanto, per i codici QLDPC è preferibile utilizzare la versione della decodifica classica delle parole di codice basata sul calcolo della sindrome [70]. L'algoritmo BP sottostante può essere implementato sia nel dominio binario che nel dominio quaternario; tali scenari sono discussi nelle Sezioni 5.1 e 5.2, rispettivamente.

5.1 Decodifica binaria

Un canale di depolarizzazione quantum, caratterizzato dalla probabilità di depolarizzazione p, è isomorfo a due canali simmetrici binari indipendenti (BSC − Binary Symmetric Channel) [11], cioè un canale di phase-flip ed un canale di bit-flip, ciascuno con probabilità di cross-over pari a $2p/3$.

In particolare, basandosi sull'isomorfismo dal dominio di Pauli al dominio classico binario, spiegato nella Sezione 2.3.1, un errore di Pauli $\mathcal{P} \in \mathcal{G}_n$, composto da un blocco di n qubit e trasmesso su un canale di depolarizzazione, può essere modellato con un vettore di errore effettivo $\mathbf{P}$, che è un vettore binario di lunghezza $2n$. Il vettore di errore $\mathbf{P}$ può essere rappresentato come $\mathbf{P} = (\mathbf{P}_z, \mathbf{P}_x)$, dove sia $\mathbf{P}_z$ che $\mathbf{P}_x$ sono vettori lunghi n bit e rappresentano gli errori di Pauli $\mathbf{Z}$ e $\mathbf{X}$, rispettivamente. Ciò implica che un errore di Pauli $\mathbf{X}$, che disturba il t-esimo qubit, produce rispettivamente uno 0 e un 1 nell'indice t-esimo e $(n+t)$-esimo di $\mathbf{P}$. Allo stesso modo, un errore di Pauli $\mathbf{Z}$, posizionato sul t-esimo qubit, genera un 1 e uno 0 nel t-esimo e nel $(n+t)$-esimo indice del vettore $\mathbf{P}$, rispettivamente. Infine un errore di Pauli $\mathbf{Y}$ sul t-esimo qubit risulta in un 1 sia nel primo che nel $(n+t)$-esimo elemento di $\mathbf{P}$. Dato che un canale di depolarizzazione è caratterizzato dalla probabilità p di incorrere in un errore di Pauli $\mathbf{X}$, $\mathbf{Y}$ e

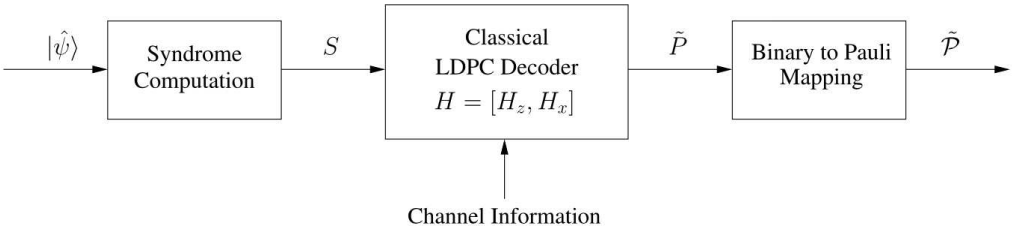

Figura 5.1: Schema a blocchi che rappresenta il processo di elaborazione della sindrome per un codice QLDPC [28].

$\mathbf{Z}$, con probabilità pari a $p/3$ per ognuno, il vettore di errore effettivo $\mathbf{P}$ si riduce a due BSC, con probabilità di crossover di $2p/3$ ciascuno, dove uno rappresenta il canale per gli errori di Pauli $\mathbf{Z}$ e l'altro il canale per gli errori di Pauli $\mathbf{X}$.

Sulla base della suddetta struttura semplificata, che ignora la correlazione tra gli errori di Pauli $\mathbf{X}$ e $\mathbf{Z}$ poiché si sta lavorando nel dominio di binario, i codici QLDPC possono essere decodificati attraverso l'algoritmo BP basato sulla sindrome e rappresentato sul grafo di Tanner del codice binario equivalente, che ha come PCM $\mathbf{H} = (\mathbf{H}_z|\mathbf{H}_x)$ [59]. In particolare, sia $\mathbf{s}$ la sindrome della sequenza osservata, che è data dal prodotto simplettico tra $\mathbf{H}$ e $\mathbf{P}$, come formulato sotto:

$$\mathbf{s} = \mathbf{H} \star \mathbf{P}^T = (\mathbf{H}_z \mathbf{P}_x^T + \mathbf{H}_x \mathbf{P}_z^T) \mod 2. \tag{5.1}$$

La sindrome $\mathbf{s}$ osservata della (5.1) alimenta un decodificatore LDPC classico, basato sulla sindrome, per stimare l'errore $\tilde{\mathbf{P}}$ più probabile determinato dal canale, come illustrato nella Fig. 5.1.

Per una PCM $\mathbf{H}$ di dimensioni $m \times 2n$, dove vale $m = (n-k)$, il vettore di errore stimato $\tilde{\mathbf{P}}$ è di lunghezza $2n$, in cui i primi n bit rappresentano gli errori di phase-flip stimati e sono raccolti dal vettore $\tilde{\mathbf{P}}_z$, mentre gli altri n bit indicano gli errori di bit-flip stimati e vengono rappresentati dal vettore $\tilde{\mathbf{P}}_x$. Infine, questo vettore binario composto da $2n$ bit, viene mappato sul vettore di errore $\tilde{\mathcal{P}}$ (composto da n qubit) basato sull'isomorfismo dal dominio classico binario al dominio di Pauli. In particolare, il valore del t-esimo e del $(n+t)$-esimo elemento del vettore $\tilde{\mathbf{P}}$ sono combinati per stimare l'errore inflitto al t-esimo qubit. Per ulteriori delucidazioni, si consulti la Sezione 2.3.

Per i codici di tipo CSS, si ha che $\mathbf{H}_z = \begin{pmatrix} \mathbf{H}'_z \\ \mathbf{0} \end{pmatrix}$ e $\mathbf{H}_x = \begin{pmatrix} \mathbf{0} \\ \mathbf{H}'_x \end{pmatrix}$. Di conseguenza, il grafo di Tanner della matrice $\mathbf{H}$ è costituito da due grafi di Tanner indipendenti corrispondenti alle matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$. Ciò implica, a sua volta, che gli errori di Pauli $\mathbf{X}$ e $\mathbf{Z}$ possono essere decodificati indipendentemente, utilizzando le matrici $\mathbf{H}'_z$ e $\mathbf{H}'_x$, rispettivamente [11]. Quindi, il rate di errore per ogni qubit (QBER − Qubit Error Rate) di un codice QLDPC di tipo CSS può essere approssimato con la somma del BER associato ai codici classici costituenti. Più esplicitamente, se p_e^x e p_e^z sono i BER classici associati a $\mathbf{H}'_z$ e $\mathbf{H}'_x$ rispettivamente, allora il QBER complessivo è

equivalente a $(p_e^x + p_e^z - p_e^x p_e^z) \approx (p_e^x + p_e^z)$, che si riduce a $2p_e^z$ per un codice CSS dual-containing, dato che vale $\mathbf{H}'_x = \mathbf{H}'_z$, come riportato anche nella Sezione 2.4.

Per una matrice binaria $\mathbf{H}$ di dimensioni $m \times 2n$ associata ad un codice LDPC, il decodificatore classico LDPC di Fig. 5.1, data la sindrome osservata $\mathbf{s}$, mira a trovare l'errore $\mathbf{P}$ di lunghezza $2n$ più verosimile, cioè si ottiene:

$$\tilde{\mathbf{P}} = \underset{\mathbf{P} \in (\mathbb{F}_2)^{2n}}{\operatorname{argmax}} \Pr(\mathbf{P}|\mathbf{s}), \tag{5.2}$$

dove $\Pr(\mathbf{P}|\mathbf{s})$ rappresenta la probabilità che accada l'errore $\mathbf{P} \in (\mathbb{F}_2)^{2n}$ imposto sulle parole di codice trasmesse, dato che la sindrome della sequenza dei qubit ricevuti $|\hat{\psi}\rangle$ è $\mathbf{s} \in (\mathbb{F}_2)^m$. Purtroppo, come detto nella Sezione 3.1, la (5.2) definisce un problema NP completo, come si può verificare in [71]. Un algoritmo sub-ottimale per la risoluzione della (5.2) è costituito dall'algoritmo BP classico, che trova il valore ottimale elemento per elemento, piuttosto che il valore ottimale globale. La decodifica LDPC classica è infatti locale ed è per questo sub-ottima, mentre la decodifica MAP è globale ed è, in questo senso, ottima ma non può essere applicata ai codici LDPC classici. In particolare, per $\mathbf{P} = (P_0, P_1, ..., P_t, ..., P_{2n-1})$, l'algoritmo BP trova un valore per P_t che vale:

$$\tilde{P}_t = \underset{P_t \in \mathbb{F}_2}{\operatorname{argmax}} \Pr(P_t|\mathbf{s}), \tag{5.3}$$

dove $\mathrm{P}(P_t|\mathbf{s})$ è la probabilità marginale del t-esimo bit.

L'algoritmo BP opera scambiando messaggi sul grafo di Tanner associato alla PCM $\mathbf{H}$; esso è costituto dai nodi di controllo c_i per $i \in \{0, m-1\}$ e dai nodi variabile v_t per $t \in \{0, 2n-1\}$. I messaggi inviati dal nodo di controllo i-esimo al nodo variabile t-esimo sono denotati da $m^{P_t}_{c_i \to v_t}$, mentre i messaggi diretti dal t-esimo nodo variabile all'i-esimo nodo di controllo sono dati da $m^{P_t}_{v_t \to c_i}$, dove P_t è l'errore imposto sul nodo variabile t-esimo. La procedura generale di scambio dei messaggi basata sulla sindrome è riassunta nell'Algoritmo 1, che procede come segue [70]:

- **Inizializzazione**: L'algoritmo comincia inizializzando i messaggi $m^{P_t}_{v_t \to c_i}$ secondo il modello di canale $\mathrm{P}_{\mathrm{ch}}(P_t)$. Per un BSC con probabilità di crossover pari a $2p/3$, si ha:

$$\begin{aligned} m^0_{v_t \to c_i} &= 1 - 2p/3, \\ m^1_{v_t \to c_i} &= 2p/3. \end{aligned} \tag{5.4}$$

- **Scambio orizzontale dei messaggi**: Sia $V(c_i)$ l'insieme dei nodi variabile connessi al nodo di controllo c_i, cioè vale $V(c_i) \equiv \{v_t : \mathbf{H}_{it} = 1\}$, quindi $V(c_i) \setminus v_t$ l'insieme $V(c_i)$ escludendo il nodo variabile v_t (*marginalizzazione*). Sia poi $V(i)$ l'insieme degli indici dei nodi variabile connessi all'i-esimo nodo di controllo c_i, dunque $V(i) \setminus t$ è lo stesso insieme meno l'indice di v_t. Come rappresentato nella Fig. 5.2a, in questo passo l'algoritmo passa attraverso le righe della PCM $\mathbf{H}$ (per questo chiamato *orizzontale*)

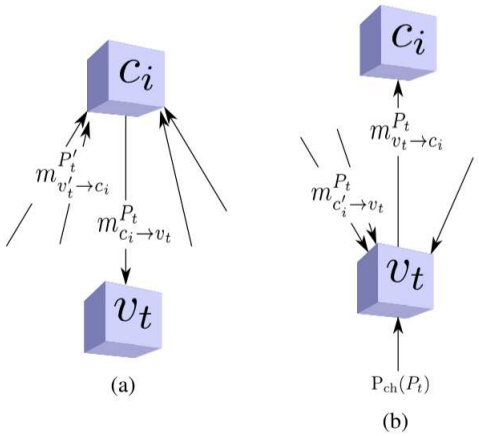

Figura 5.2: Algoritmo Belief Propagation (BP). I nodi di controllo e i nodi variabile sono denotati da c_i e da v_t, rispettivamente [28]. (a) Scambio orizzontale di messaggi. (b) Scambio verticale di messaggi.

ed elabora i messaggi $m^{P_t}_{c_i \to v_t}$, per ogni nodo variabile $v_t \in V(c_i)$ e per ogni $P_t \in \mathbb{F}_2$. Il messaggio $m^a_{c_i \to v_t}$ rappresenta la probabilità che il valore della sindrome osservato per il nodo di controllo c_i sia s_i, dato che il t-esimo nodo variabile è caratterizzato dall'errore ($P_t = a$), con $a \in \{0, 1\}$. Questo può essere matematicamente espresso come segue:

$$m^a_{c_i \to v_t} = K \sum_{\{P_{t'} : t' \in V(i) \backslash t\}} \Pr(s_i | P_t = a, \{P_{t'} : t' \in V(i) \backslash t\}) \prod_{v_{t'} \in V(c_i) \backslash v_t} m^{P_{t'}}_{v_{t'} \to c_i}, \tag{5.5}$$

dove K è la costante di normalizzazione, necessaria per garantire che valga $\sum_{a \in \{0,1\}} m^a_{c_i \to v_t} = 1$, mentre la sommatoria indica che deve essere fatta la somma per ogni possibile combinazione degli elementi del seguente insieme $\{P_{t'} : t' \in V(i) \setminus t\}$. Inoltre $\Pr(s_i | P_t = a, \{P_{t'} : t' \in V(i) \setminus t\})$ è una funzione binaria, che vale 1 quando la somma modulo 2 tra tutti gli elementi della combinazione corrente dell'insieme $\{P_{t'} : t' \in V(i) \setminus t\}$ e $P_t = a$ uguaglia il valore di s_i, altrimenti vale 0.

In accordo con la (5.5), i messaggi $m^a_{c_i \to v_t}$ destinati al t-esimo nodo variabile non tengono conto dei messaggi che scorrono nella direzione opposta lungo lo stesso ramo, cioè i messaggi $m^a_{v_t \to c_i}$. Di conseguenza, i messaggi $m^a_{c_i \to v_t}$ contengono solo l'informazione nuova, raccolta dai messaggi inviati dagli altri nodi variabile ed è per questo definita come *estrinseca*. Ciò assicura che le successive iterazioni di questo algoritmo siano indipendenti.

- **Scambio verticale dei messaggi**: Sia $C(v_t)$ l'insieme dei nodi di controllo connessi al nodo variabile v_t, cioè vale $C(v_t) \equiv \{c_i : \mathbf{H}_{it} = 1\}$, sia poi $C(v_t) \setminus c_i$ l'insieme $C(v_t)$ escludendo il nodo di controllo c_i (*marginalizzazione*). Come mostrato nella Fig. 4.8b, per ogni colonna di $\mathbf{H}$ (per questo chiamato *verticale*), l'algoritmo BP elabora il messaggio $m^{P_t}_{v_t \to c_i}$, per ogni nodo di controllo $c_i \in V(v_t)$ e per ogni $P_t \in \mathbb{F}_2$. In particolare, i messaggi $m^a_{v_t \to c_i}$ vengono elaborati valutando il prodotto tra il canale di informazione

$\Pr_{\text{ch}}(P_t = a)$ e i messaggi $m^a_{c_{i'} \to v_t}$, che scorrono nel nodo variabile v_t, lungo tutti i rami connessi ad esso, ad esclusione del ramo del messaggio $m^a_{c_i \to v_t}$. Dunque, il messaggio estrinseco è calcolato come segue:

$$m^a_{v_t \to c_i} = K \Pr_{\text{ch}}(P_t = a) \prod_{c_{i'} \in C(v_t) \setminus c_i} m^a_{c_{i'} \to v_t}, \tag{5.6}$$

dove K è la costante di normalizzazione, necessaria per garantire che valga $\sum_{a \in \{0,1\}} m^a_{v_t \to c_i} = 1$.

- **Probabilità marginale element-wise**: Infine, la probabilità marginale element-wise $\Pr(P_t|\mathbf{s})$ per ogni $P_t \in \mathbb{F}_2$, è calcolata come segue:

$$\Pr(P_t = a|\mathbf{s}) = K \Pr_{\text{ch}}(P_t = a) \prod_{c_i \in C(v_t)} m^a_{c_i \to v_t}, \tag{5.7}$$

che tiene conto di tutti i messaggi che scorrono nel nodo variabile v_t.

- **Decisione hard e controllo della sindrome**: Come precedentemente indagato nella (5.3), la decodifica definita *hard* prevede di individuare l'errore $\tilde{P}_t$ più verosimile (probabile), che massimizza la probabilità marginale calcolata nella (5.7). Il vettore della sindrome $\tilde{\mathbf{s}} = \mathbf{H}(\tilde{\mathbf{P}}_x \ : \ \tilde{\mathbf{P}}_z)^T$ si calcola basandosi sul vettore di errore stimato $\tilde{\mathbf{P}}$. Se la sindrome $\tilde{\mathbf{s}}$, associata all'errore stimato $\tilde{\mathbf{P}}$, equivale alla sindrome osservata $\mathbf{s}$ il processo si arresta, indicando che è stata trovata la soluzione corretta. In caso contrario, l'algoritmo si ripete dalla fase di scambio del messaggio orizzontale in poi. Questa procedura iterativa continua fino al raggiungimento della condizione $\tilde{\mathbf{s}} = \mathbf{s}$ oppure quando viene raggiunto il numero massimo di iterazioni fissato $\mathrm{I}_{\max}$.

Nel Capitolo 6 è presente lo pseudo-codice che implementa l'Algoritmo 1.

5.2 Decodifica non binaria

Sulla base dell'isomorfismo dal dominio di Pauli al dominio classico quaternario (GF(4)), spiegato dettagliatamente nella Sezione 2.3.2, i codici QLDPC possono essere decodificati invocando l'algoritmo BP non binario, che tiene conto della correlazione tra gli errori di phase-flip e gli errori di bit-flip. Il BP non binario basato sulla sindrome è simile al BP binario dell'Algoritmo 1, considerando le due seguenti modifiche principali:

- L'algoritmo BP non binario sfrutta il modello specifico di canale di depolarizzazione che *non* ignora la correlazione tra gli errori di phase-flip e di bit-flip. Il modello di canale equivalente nel dominio quaternario ha la seguente distribuzione di probabilità:

$$\mathbf{Pr}_{\text{ch}}(\hat{P}_t = \hat{a}) = \begin{cases} 1 - p, & \text{se } \hat{a} = 0 \\ p/3, & \text{se } \hat{a} \in \{1, \omega, \overline{\omega}\}, \end{cases} \tag{5.8}$$

dove $\hat{\mathbf{P}} = (\hat{P}_0, \hat{P}_1, \ldots, \hat{P}_t, \ldots, \hat{P}_{n-1})$ è il vettore di errore nel dominio quaternario e quindi $\hat{P}_t$ denota l'errore determinato sul t-esimo qubit.

- La sindrome s_i, che è stata calcolata come $\mathbf{H}_i(\mathbf{P}_x : \mathbf{P}_z)^T$ nel dominio binario, è ora data dalla traccia del prodotto scalare (o traccia del prodotto interno) tra $\hat{\mathbf{H}}_i$ e $\hat{\mathbf{P}}$ (consultare a questo proposito la Sezione 2.3.2). Dunque vale:

$$s_i = \mathrm{Tr}(\hat{\mathbf{H}}_i \cdot \hat{\mathbf{P}}), \tag{5.9}$$

dove $\hat{\mathbf{H}}_i$ è l'i-esima riga della PCM $\mathbf{H}$ in GF(4) e $i \in \{0, m-1\}$.

Rispetto al BP binario, la decodifica non binaria impone una maggiore complessità, specificamente sulla fase di scambio orizzontale dei messaggi. In particolare, dal momento che la somma nella (5.5) funziona per tutte le possibili sequenze di errore $\{\hat{\mathbf{P}} : \hat{P}_t = \hat{a}\}$, che producono la sindrome s_i per l'i-esimo nodo di controllo, la complessità aumenta sia con il peso della riga sia con la dimensione del campo di Galois associato. Per i codici LDPC classici non binari, questa maggiore complessità viene diminuita utilizzando l'algoritmo di decodifica basato sulla Fast Fourier Transform (FFT) di [72], che può essere convenientemente adattata alla decodifica basata sulla sindrome dei codici QLDPC.

Sulla base della nozione della traccia del prodotto scalare, la (5.9) può essere riscritta come segue:

$$s_i = \mathrm{Tr}(\hat{s}_i) = \mathrm{Tr}\Big(\sum_{t \in V(c_i)} \hat{\mathbf{H}}_{it} \times \overline{\hat{P}}_t \Big), \tag{5.10}$$

dove $\overline{\hat{P}}_t$ è il coniugato di $\hat{P}_t$ e si ottiene che $\hat{s}_i \in \{0, 1, \omega, \overline{\omega}\}$, che può anche essere espresso come:

$$\hat{s}_i = \hat{\mathbf{H}}_{it} \times \overline{\hat{P}}_t + \sum_{t' \in V(c_i) \setminus v_t} \hat{\mathbf{H}}_{it'} \times \overline{\hat{P}}_{t'}. \tag{5.11}$$

A differenza di quanto accade nel dominio binario, dove vale $\mathbf{H}_{it} \in \{0, 1\}$, nel dominio quaternario si ha la seguente relazione $\hat{\mathbf{H}}_{it} \in \{0, 1, \omega, \overline{\omega}\}$, come accade nella (5.11). Pertanto, dati i messaggi $m^{\hat{a}}_{c_i \to v_t}$ e $m^{\hat{a}}_{v_t \to c_i}$, scambiati tra il nodo di controllo c_i e il nodo variabile v_t per $\hat{P}_t = \hat{a}$, si indicano i messaggi equivalenti nel dominio quaternario per $(\hat{\mathbf{H}}_{it} \times \overline{\hat{P}}_t)$, come $\check{m}^{\hat{a}_s}_{c_i \to v_t}$ e $\check{m}^{\hat{a}_s}_{v_t \to c_i}$, rispettivamente, dove vale $(\hat{\mathbf{H}}_{it} \times \overline{\hat{a}}) = \hat{a}_s$.

Basandosi su questa notazione, si può affermare dalla (5.10) e dalla (5.11) che la funzione densità di probabilità (chiamata d'ora in avanti, per brevità, con il suo acronimo PDF − Probability Density Function) del messaggio orizzontale $\check{m}^{\hat{a}_s}_{c_i \to v_t}$ può essere ottenuta attraverso la convoluzione tra le PDF dei messaggi $\check{m}^{\hat{a}_s + \hat{s}_i}_{v_{t'} \to c_i}$ per $v_{t'} \in V(c_i) \setminus v_t$. Dunque si può osservare dalla (5.10) che per un dato valore di s_i, la sindrome stimata $\hat{s}_i$ può assumere due valori. In particolare, in GF(4), si ha che $\mathrm{Tr}(0) = \mathrm{Tr}(1) = 0$, mentre $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$, come esposto nella Sezione 2.3.2. Di conseguenza, per

$s_i = \mathrm{Tr}(\hat{s}_i = 0) = \mathrm{Tr}(\hat{s}_i = 1) = 0$, si ottiene:

$$\begin{aligned}
(i)\ PDF\{\check{m}^0_{c_i \to v_t}\} &= PDF\{\check{m}^1_{c_i \to v_t}\} \\
&= \frac{1}{2}\Bigg(\bigotimes_{v_{t'}} PDF\{\check{m}^0_{v_{t'} \to c_i}\} + \bigotimes_{v_{t'}} PDF\{\check{m}^1_{v_{t'} \to c_i}\} \Bigg), \\
(ii)\ PDF\{\check{m}^\omega_{c_i \to v_t}\} &= PDF\{\check{m}^{\overline{\omega}}_{c_i \to v_t}\} \\
&= \frac{1}{2}\Bigg(\bigotimes_{v_{t'}} PDF\{\check{m}^\omega_{v_{t'} \to c_i}\} + \bigotimes_{v_{t'}} PDF\{\check{m}^{\overline{\omega}}_{v_{t'} \to c_i}\} \Bigg),
\end{aligned} \tag{5.12}$$

dove $\bigotimes$ rappresenta l'operazione di convoluzione e $v_{t'} \in V(c_i) \setminus v_t$. Analogamente, per $s_i = \mathrm{Tr}(\hat{s}_i = \omega) = \mathrm{Tr}(\hat{s}_i = \overline{\omega}) = 1$, si ottiene:

$$\begin{aligned}
(i)\ PDF\{\check{m}^0_{c_i \to v_t}\} &= PDF\{\check{m}^1_{c_i \to v_t}\} \\
&= \frac{1}{2}\Bigg(\bigotimes_{v_{t'}} PDF\{\check{m}^\omega_{v_{t'} \to c_i}\} + \bigotimes_{v_{t'}} PDF\{\check{m}^{\overline{\omega}}_{v_{t'} \to c_i}\} \Bigg), \\
(ii)\ PDF\{\check{m}^\omega_{c_i \to v_t}\} &= PDF\{\check{m}^{\overline{\omega}}_{c_i \to v_t}\} \\
&= \frac{1}{2}\Bigg(\bigotimes_{v_{t'}} PDF\{\check{m}^0_{v_{t'} \to c_i}\} + \bigotimes_{v_{t'}} PDF\{\check{m}^1_{v_{t'} \to c_i}\} \Bigg).
\end{aligned} \tag{5.13}$$

La complessa operazione di convoluzione espressa nella (5.12) e nella (5.13) può essere implementata efficientemente moltiplicando le PDF corrispondenti nel dominio della frequenza con l'ausilio dell'algoritmo basato sulla FFT spiegato in [72].

5.3 Problemi di decodifica e metodi euristici migliorativi

L'algoritmo BP utilizzato per decodificare i codici LDPC fornisce la soluzione esatta solo quando il grafo di Tanner sottostante è un albero.

Approfondimento: grafo ad albero Si dice albero un grafo G connesso, non orientato e senza cicli. Per essere tale, il grafo deve rispettare almeno una delle seguenti richieste:

- possedere un solo cammino per ogni coppia di nodi;
- essere *aciclico massimale*, ossia garantire la condizione per cui se gli si aggiunge un altro ramo per unire due suoi nodi si forma un ciclo;
- deve costituire un *grafo connesso minimale*, cioè garantire la condizione per cui se si rimuove un suo ramo si perde il collegamento tra due nodi.

Se G ha un numero finito di n vertici, gli enunciati precedenti sono equivalenti ai due che seguono:

- G è connesso e possiede $n - 1$ rami;

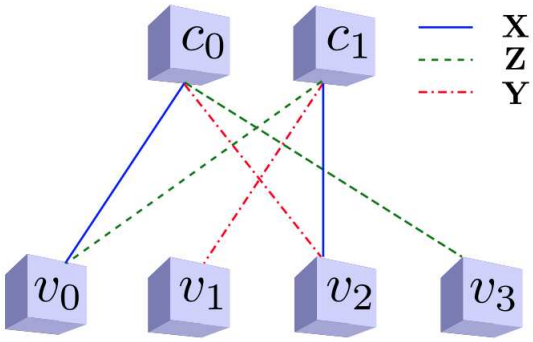

Figura 5.3: Grafo di Tanner di una coppia di stabilizer generator, dove c_0 e c_1 sono i nodi di controllo per i generatori $g_0 = \mathbf{XIYZ}$ e $g_1 = \mathbf{ZYXI}$, rispettivamente. I rami connessi ai nodi variabile v_0 e v_2 costituiscono un ciclo di lunghezza 4 [28].

• G è aciclico (senza cicli) e possiede $n - 1$ rami.

Si può definire albero dunque un grafo connesso tale che eliminando uno qualsiasi dei suoi rami si perde la connessione tra i nodi.

Tuttavia, l'algoritmo BP fornisce approssimazioni ragionevolmente buone anche in presenza di cicli, a condizione che il girth della matrice di parità associata al codice LDPC sia sufficientemente ampia: almeno 6. Ciò è stato dimostrato dalla capacità di avvicinarsi ai codici LDPC classici, ad esempio in [73] e [74]. Purtroppo, i cicli brevi di lunghezza 4 sono inevitabili nella costruzione dei codici QLDPC e questi, a loro volta, compromettono la procedura di decodifica iterativa.

Gli inevitabili cicli di lunghezza 4 presenti nei codici QLDPC sono il risultato della proprietà di commutatività degli stabilizzatori. In particolare, gli stabilizer generator che costituiscono un codice stabilizzatore devono commutare, cioè dovrebbero essere caratterizzati da un numero pari di posti con operatori di Pauli non di identità diversi. Questo aspetto è motivato e approfondito nella Sezione 2.2.

In altre parole, se una coppia di operatori di Pauli anti-commutativa agisce sul t-esimo nodo variabile in una coppia di stabilizer generator, allora ci dovrebbe essere un'altra coppia di operatori di Pauli anti-commutativa che agisce sul nodo variabile t'-esimo nella stessa coppia di generatori, per garantire che questi due stabilizer generator commutino tra loro. Ad esempio, i generatori:

$$\begin{aligned} g_0 &= \mathbf{XIYZ}, \\ g_1 &= \mathbf{ZYXI}, \end{aligned} \tag{5.14}$$

commutano perché ci sono due coppie di operatori di Pauli anti-commutative che agiscono sul primo e sul terzo qubit, rispettivamente, cioè $\{\mathbf{X}, \mathbf{Z}\}$ e $\{\mathbf{Y}, \mathbf{X}\}$. N.B. Questo è solo un esempio arbitrario per illustrare il concetto di commutatività e i cicli brevi risultanti; i generatori g_0 e g_1 di questo esempio potrebbero non costituire un buon codice stabilizzatore.

La PCM risultante, nell'ipotesi di isomorfismo dal dominio di Pauli al dominio classico quaternario, è:

$$\hat{\mathbf{H}} = \begin{pmatrix} 1 & 0 & \overline{\omega} & \omega \\ \omega & \overline{\omega} & 1 & 0 \end{pmatrix},$$

Si specifica che in questa sede con la notazione $\hat{\mathbf{H}}$ si fa riferimento alla PCM di un codice quantum in GF(4).

Questo a sua volta implica che le righe della PCM $\hat{\mathbf{H}}$ risultante hanno un numero pari di sovrapposizioni (cioè due), che danno luogo a cicli brevi nel grafo di Tanner, come illustrato nella Fig. 5.3. Dal momento che qui l'idea chiave è quella di avere operatori non di identità diversi, una possibile strada percorribile potrebbe essere quella di assegnare solo un singolo tipo di operatore non di identità a ciascun nodo variabile del grafo di Tanner. Se si assegna così solo l'operatore di Pauli $\mathbf{X}$ al nodo variabile v_t in modo che non anti-commuta con qualsiasi coppia di generatori, allora non si è in grado di rilevare né l'errore di Pauli $\mathbf{X}$, né l'errore di Pauli $\mathbf{Y}$ che agiscono su v_t. Questo produrrebbe un codice indesiderabile e non utile per scopi pratici, che ha distanza minima pari ad 1.

Si può pertanto concludere che:

1) ogni colonna di una PCM corrispondente ad un codice QLDPC deve avere almeno due operatori di Pauli non di identità diversi e

2) ogni coppia di righe deve avere un numero pari di posizioni con operatori di Pauli non di identità diversi tra loro.

Di conseguenza, tutte le strutture QLDPC di tipo CSS e non CSS hanno un grafo di Tanner di girth 4. È interessante osservare, a questo punto, che questi cicli brevi possono essere evitati nel corrispondente formalismo binario. Consideriamo pertanto l'esempio fornito nella (5.14), che può essere espresso nella forma binaria come segue, seguendo l'opportuno isomorfismo:

$$\begin{aligned} g_0 &\rightarrow \left(0\ 0\ 1\ 1 \mid 1\ 0\ 1\ 0\right); \\ g_1 &\rightarrow \left(1\ 1\ 0\ 0 \mid 0\ 1\ 1\ 0\right), \end{aligned}$$

da cui si ricava la PCM associata

$$\mathbf{H} = \left(\begin{array}{cccc|cccc} 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Poiché questi generatori binari hanno solo una singola sovrapposizione di "1" (settima posizione), il ciclo breve di lunghezza 4 non esiste più. Tuttavia, si ricorda dalla Sezione 5.2 che la decodifica binaria ignora la correlazione tra gli errori di Pauli $\mathbf{X}$ e $\mathbf{Z}$, che però ne degrada le prestazioni. Occorre quindi trovare un compromesso tra questi due aspetti conflittuali.

Il problema dei cicli brevi è presente, in modo significativo, sia nei codici QLDPC *binari* di tipo CSS dual-containing che nei codici EA QLDPC *binari* di tipo CSS dual-containing, cioè quelli aventi $\mathbf{H}'_x = \mathbf{H}'_z$. Essi possono essere chiamati codici CSS *omogenei*, dal momento che le PCM usate per correggere gli errori di bit-flip e di phase-flip sono identiche.

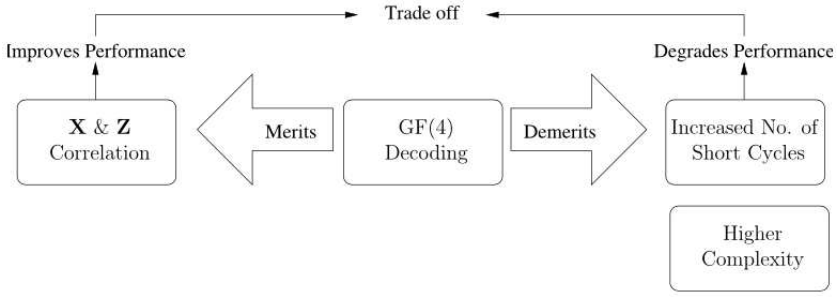

Figura 5.4: Vantaggi e svantaggi della decodifica in GF(4) rispetto alla decodifica binaria [28].

Sia $\hat{\mathbf{H}}$ la PCM risultante di dimensioni $m \times n$ in GF(4) come segue:

$$\hat{\mathbf{H}} = \begin{pmatrix} \omega \mathbf{H}'_z \\ \mathbf{H}'_z \end{pmatrix}, \tag{5.15}$$

dove $\mathbf{H}'_z$ è una matrice binaria, proveniente da un opportuno codice CSS dual-containing binario.

Si osserva pertanto che l'i-esima riga e la $(i + m/2)$-esima riga si sovrappongono completamente, dando luogo a numerosi cicli di lunghezza 4. Inoltre, la costruzione dual-containing del codice ha anche i cicli brevi addizionali, all'interno della matrice $\mathbf{H}'_z$, come discusso nella sezione 4.1, che esistono anche nel formalismo binario.

Esempio: Trasformazione della matrice di parità dal dominio binario a quello quaternario Sia

$$\mathbf{H}'_z = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

la matrice binaria di un codice CSS dual-containing di partenza dal quale si vuole ricavare un codice in GF(4). Dalla (5.15) discende che:

$$\hat{\mathbf{H}} = \begin{pmatrix} \omega & 0 & \omega & \omega \\ 0 & \omega & \omega & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix},$$

con $m = 4$ e $n = 4$. Da questo esempio dovrebbe essere anche più chiaro il motivo che spiega perché si moltiplica la prima matrice $\mathbf{H}'_z$ per ω. In questo modo, la matrice binaria $\mathbf{H}'_z$ infatti avrà al posto degli 1s tutti simboli $\mathbf{H}'_z$ e quindi la PCM $\hat{\mathbf{H}}$ sarà caratterizzata anche da simboli di GF(4) che non si trovano nel dominio binario.

Per $i = \{0, 1\}$, come esplicitato sopra, si ha che in entrambi i casi, l'i-esima riga e la $(i + m/2)$-esima riga si sovrappongono completamente (riga 1 e 3, riga 2 e 4).

Nella Sezione 5.4 è presente un esempio che approfondisce in maniera più puntuale queste nozioni.

Tabella 5.1: Cicli brevi inevitabili in varie strutture di codice (• = cicli presenti, ∘ = cicli assenti, •• = numerosi cicli presenti) [28].

Tipo di Codice	Cicli brevi inevitabili: Formalismo binario	Cicli brevi inevitabili: Formalismo in GF(4)
CSS dual-containing (omogeneo)	•	••
CSS non-dual-containing	∘	•
Non CSS	∘	•
EA CSS (omogeneo)	∘	••
Tutti gli altri codici EA	∘	∘

La Tabella 5.1 riassume la presenza dei cicli brevi inevitabili in varie strutture di codice diverse, mentre la Fig. 5.4 rappresenta i vantaggi e gli svantaggi della decodifica in GF(4) rispetto alla sua controparte binaria.

La nozione di codice degenere è un altro aspetto unico, che distingue un codice quantum da uno classico. Si ricorda dalla Sezione 4.1.2 infatti che gli errori, che differiscono solo per il gruppo stabilizzatore, hanno lo stesso impatto sulle parole di codice trasmesse e possono quindi essere corretti dalla stessa operazione di recupero. Questo a sua volta migliora le prestazioni dei codici quantum. Sfortunatamente, la decodifica iterativa impiegata per i codici QLDPC non tiene conto di questa degenerazione. In particolare, piuttosto che trovare l'errore più probabile, come accade nella (5.2), l'algoritmo di decodifica dovrebbe trovare l'insieme degli errori più probabile, sommando le probabilità di tutti gli errori degeneri [75], [76]. Inoltre, i codici QLDPC sono altamente degeneri rispetto alle altre famiglie di codici quantum, questo perché gli stabilizer generator di un codice QLDPC sono sparsi di natura. Di conseguenza, sono presenti molti errori degeneri con peso basso, che dominano la probabilità di errore del sottoinsieme. È quindi più plausibile che l'errore più probabile $\tilde{P}$ della (5.2) non coincida con il sottoinsieme degli errori più probabile per i codici QLDPC. Tuttavia, anziché sfruttare i vantaggi dell'elevata degenerazione associata ai codici sparsi, la decodifica iterativa, basata sulla probabilità marginale e utilizzata per i codici QLDPC, è compromessa dalla degenerazione, come spiegato in [75] e [76]. Questo perché gli errori degeneri con lo stesso peso hanno la stessa distribuzione di probabilità marginale, che può essere attribuita alla simmetria della distribuzione di probabilità del canale descritto dalla (5.8).

Si esamina il caso di studio riportato in [75]. Si consideri un codice stabilizzatore a 2 qubit avente come generatori **XX** e **ZZ**. Si supponga inoltre che **IX** sia l'errore, codificato durante la trasmissione su un canale di depolarizzazione, la cui PDF è riportata nella (5.8). La sindrome risultante è $\mathbf{s} = (01)$ e il corrispondente insieme di errori degeneri è $\{\mathbf{XI}, \mathbf{IX}, \mathbf{YZ}, \mathbf{ZY}\}$. Di conseguenza, la probabilità marginale condizionata dell'errore, su ciascuno dei due qubit, è data da:

$$\Pr(\hat{P}_t = \hat{a}|\mathbf{s}) = \begin{cases} 1-p, & \text{se } \hat{a} = 0 \\ p/3, & \text{se } \hat{a} \in \{1, \omega, \overline{\omega}\}, \end{cases} \tag{5.16}$$

dove $t = \{0, 1\}$. Quindi, la probabilità marginale è identica per entrambi i qubit. Questa simmetria induce il decodificatore a rilevare lo stesso errore su entrambi i qubit. Tuttavia, dopo che si è applicata un'operazione di recupero per correggere l'errore **IX**, utilizzando gli errori degeneri associati, cioè $\{\mathbf{XI}, \mathbf{IX}, \mathbf{YZ}, \mathbf{ZY}\}$, in nessun caso appare più questa simmetria; da qui si sviluppa il concetto di *errore di degenerazione simmetrico*, espresso in [75]. Inoltre, poiché il modello di canale rappresentato nella (5.8) è polarizzato verso l'operatore di identità **I**, la probabilità di assenza di errori domina a bassi livelli di rumore.

Poulin e Chung hanno studiato vari metodi euristici in [75] per rompere la simmetria esibita dalle probabilità marginali della (5.16). Tra quelli investigati, il metodo Random Perturbation (Perturbazione Casuale) fornisce le prestazioni migliori. Esso mira a rompere la simmetria degenere, perturbando casualmente la PDF del canale della (5.8), per i qubit coinvolti nei nodi di controllo "frustrati", mettendo così fine alla situazione di stallo del processo di decodifica. I nodi di controllo per i quali la sindrome calcolata non corrisponde alla sindrome osservata sono noti come nodi di controlli frustrati [75].

Il Random Perturbation inizia con l'algoritmo BP standard non binario, che conduce all'errore di canale stimato $\hat{\tilde{\mathbf{P}}}$. Se la sindrome calcolata per $\hat{\tilde{\mathbf{P}}}$ non è uguale alla sindrome di canale osservata $\mathbf{s}$, le probabilità di canale associate a tutti i nodi variabile v_t, connessi ad un nodo di controllo frustrato arbitrario c_i, vengono perturbate. Normalizzando si ottengono le seguenti relazioni:

$$\begin{aligned}
&\mathbf{P}_{\text{ch}}(\hat{P}_t = 0) \rightarrow \mathbf{P}_{\text{ch}}(\hat{P}_t = 0);\\
&\mathbf{P}_{\text{ch}}(\hat{P}_t = 1) \rightarrow (1 + \delta_1)\mathbf{P}_{\text{ch}}(\hat{P}_t = 1);\\
&\mathbf{P}_{\text{ch}}(\hat{P}_t = \omega) \rightarrow (1 + \delta_\omega)\mathbf{P}_{\text{ch}}(\hat{P}_t = \omega);\\
&\mathbf{P}_{\text{ch}}(\hat{P}_t = \overline{\omega}) \rightarrow (1 + \delta_{\overline{\omega}})\mathbf{P}_{\text{ch}}(\hat{P}_t = \overline{\omega}),
\end{aligned} \tag{5.17}$$

dove δ_1, δ_ω e $\delta_{\overline{\omega}}$ sono variabili casuali nell'intervallo $[0, \delta]$, con δ scelto a priori. L'algoritmo BP non binario viene eseguito nuovamente con queste probabilità di canale modificate per T_{pert} iterazioni e $\hat{\tilde{\mathbf{P}}}$ viene nuovamente stimato. Se tutti i nodi di controllo ora sono soddisfatti, il processo termina. Altrimenti, il valore delle probabilità di canale perturbate nella (5.17) vengono ripristinate e il processo viene ripetuto, prendendo in considerazione un altro nodo di controllo frustrato arbitrario.

Un altro metodo euristico per diminuire il problema della degenerazione simmetrica è stato concepito in [77] da Wang *et al.*. Più in particolare, questi hanno proposto il metodo Enhanced Feedback (Feedback Migliorato), che è simile alla tecnica Random Perturbation, infatti anche in questo caso si ha l'obiettivo di perturbare le probabilità di canale. Questa perturbazione però, si basa sia sugli stabilizer generator coinvolti nei nodi di controllo frustrati, sia sul modello di canale stesso. Analogamente al metodo Random Perturbation dunque, l'algoritmo Enhanced Feedback seleziona arbitrariamente un nodo di controllo frustrato c_i. Seleziona poi anche un nodo

variabile v_t collegato a c_i. Sia quindi $\tilde{s}_i$ il valore della sindrome dell'i-esimo nodo di controllo associata all'errore stimato $\tilde{\hat{\mathbf{P}}}$ e sia s_i l'i-esima sindrome di canale osservata. La probabilità di canale per v_t viene quindi perturbata come segue:

- Se $\tilde{s}_i = 0$ e $s_i = 1$, allora:

$$\mathbf{P}_{\text{ch}}(\hat{P}_t = \hat{a}) = \begin{cases} p/2, & \text{se } \hat{a} = 0 \text{ oppure } \hat{a} = \hat{\mathbf{H}}_{it}, \\ (1-p)/2, & \text{altrimenti.} \end{cases}$$

- Se $\tilde{s}_i = 1$ e $s_i = 0$, allora:

$$\mathbf{P}_{\text{ch}}(\hat{P}_t = \hat{a}) = \begin{cases} (1-p)/2, & \text{se } \hat{a} = 0 \text{ oppure } \hat{a} = \hat{\mathbf{H}}_{it}, \\ p/2, & \text{altrimenti.} \end{cases}$$

I valori perturbati vengono inviati al decodificatore BP non binario standard, che fornisce una nuova stima dell'errore di canale. Il processo di perturbazione si ripete, fino a quando tutti i nodi di controllo sono soddisfatti o fino a che il numero massimo di feedback n_a viene raggiunto. Poiché queste perturbazioni sono più affidabili rispetto a quelle della tecnica precedente, si può affermare che il metodo Enhanced Feedback abbia prestazioni migliori rispetto al metodo euristico Random Perturbation di [11].

5.4 Decodifica non binaria modificata

Si ricorda dalla Sezione 5.3 che i codici QLDPC di tipo CSS omogenei sono caratterizzati da un numero eccessivo di cicli brevi. Nell'isomorfismo dal dominio di Pauli al dominio classico quaternario, l'i-esima e la $(i + m/2)$-esima riga della PCM $\hat{\mathbf{H}}$, associata ad un codice QLDPC omogeneo, sono uguali a meno del prodotto per ω, cioè si ha $\hat{\mathbf{H}}_i = \omega\hat{\mathbf{H}}_{i+m/2}$, come si può osservare dalla (5.15).

Esempio: codice di Steane Si consideri il codice di Steane $[7, 1]$ di [15], che deriva dal codice classico dual-containing di Hamming $(7, 4)$ binario. La PCM del codice di Hamming $(7, 4)$ binario è data da:

$$\mathbf{H} = \begin{pmatrix} 1\,0\,0\,1\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,1 \\ 0\,0\,1\,0\,1\,1\,1 \end{pmatrix}. \tag{5.18}$$

La PCM $\mathbf{H}$ dell'eq.(5.18) produce $\mathbf{H}\mathbf{H}^T = \mathbf{0}$, quindi si presta essa stessa a costruire un codice CSS dual-containing binario. Per ulteriori delucidazioni, si invita il lettore a consultare la Sezione 2.4.1.1. Dunque, la PCM binaria del codice CSS dual-containing risultante è

$$\mathbf{H}_{\text{CSS}} = \left(\begin{array}{c|c} \mathbf{H} & \mathbf{0} \\ \mathbf{0} & \mathbf{H} \end{array}\right) = \left(\begin{array}{ccccccc|ccccccc} 1&0&0&1&0&1&1&0&0&0&0&0&0&0 \\ 0&1&0&1&1&0&1&0&0&0&0&0&0&0 \\ 0&0&1&0&1&1&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&1&0&1&1 \\ 0&0&0&0&0&0&0&0&1&0&1&1&0&1 \\ 0&0&0&0&0&0&0&0&0&1&0&1&1&1 \end{array}\right),$$

secondo la definizione di codice di tipo CSS dual-containing. Si osserva che $\mathbf{H}'_z = \mathbf{H}'_x = \mathbf{H}$ per definizione di codice CSS dual-containing, dunque vale che $\mathbf{H}_z = \begin{pmatrix}\mathbf{H}\\ \mathbf{0}\end{pmatrix}$ e $\mathbf{H}_x = \begin{pmatrix}\mathbf{0}\\ \mathbf{H}\end{pmatrix}$. Poiché però in questa sede interessa studiare le PCM associate ai codici quaternari, bisogna trasformare $\mathbf{H}_{\text{CSS}}$ nella sua controparte quaternaria. Pertanto, si ricavano gli m stabilizer generator del codice binario, che corrispondono alle righe della PCM $\mathbf{H}_{\text{CSS}}$. Essi sono:

$$
\begin{aligned}
g_0 &= \mathbf{ZIIZIZZ},\\
g_1 &= \mathbf{IZIZZIZ},\\
g_2 &= \mathbf{IIZIZZZ},\\
g_3 &= \mathbf{XIIXIXX},\\
g_4 &= \mathbf{IXIXXIX},\\
g_5 &= \mathbf{IIXIXXX},
\end{aligned}
$$

dove gli stabilizzatori g_0, g_1 e g_2 rappresentano la matrice $\mathbf{H}'_z$, mentre g_3, g_4 e g_5 costituiscono $\mathbf{H}'_x$, poiché vale:

$$
\begin{aligned}
g_0 &= 10\,00\,00\,10\,00\,10\,10 \rightarrow \left(1\ 0\ 0\ 1\ 0\ 1\ 1 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right),\\
g_1 &= 00\,10\,00\,10\,10\,00\,10 \rightarrow \left(0\ 1\ 0\ 1\ 1\ 0\ 1 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right),\\
g_2 &= 00\,00\,10\,00\,10\,10\,10 \rightarrow \left(0\ 0\ 1\ 0\ 1\ 1\ 1 \middle| 0\ 0\ 0\ 0\ 0\ 0\ 0\right),\\
g_3 &= 01\,00\,00\,01\,00\,01\,01 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 1\ 0\ 0\ 1\ 0\ 1\ 1\right),\\
g_4 &= 01\,00\,01\,01\,00\,01\,00 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 0\ 1\ 0\ 1\ 1\ 0\ 1\right),\\
g_5 &= 00\,01\,01\,01\,00\,00\,01 \rightarrow \left(0\ 0\ 0\ 0\ 0\ 0\ 0 \middle| 0\ 0\ 1\ 0\ 1\ 1\ 1\right),
\end{aligned}
$$

dove la parte scritta a destra di $\rightarrow$ sono le righe della matrice $\mathbf{H}_{\text{CSS}}$. In queto caso è stato fatto il procedimento opposto rispetto alla Sezione 2.4.1.1, dove la matrice si è ricavata a partire dagli stabilizer.

A questo punto si procede con il mapping dal dominio di Pauli al dominio classico quaternario e si ottengono i seguenti stabilizzatori:

$$
\begin{aligned}
\hat{g}_0 &\rightarrow (\omega\,\omega\,0\,\omega\,\omega\,0\,0),\\
\hat{g}_1 &\rightarrow (0\,\omega\,0\,\omega\,\omega\,0\,\omega),\\
\hat{g}_2 &\rightarrow (0\,0\,\omega\,0\,\omega\,\omega\,\omega),\\
\hat{g}_3 &\rightarrow (1\,1\,0\,1\,1\,0\,0),\\
\hat{g}_4 &\rightarrow (0\,1\,0\,1\,1\,0\,1),\\
\hat{g}_5 &\rightarrow (0\,0\,1\,0\,1\,1\,1),
\end{aligned}
$$

per semplicità di notazione si è usata la notazione $\hat{g}_i$ per indicare gli stabilizer generator nel dominio quaternario. Di conseguenza, la corrispondente PCM del codice di Steane a 7 qubit, risulta:

$$
\hat{\mathbf{H}} = \begin{pmatrix}\omega\mathbf{H}'_z\\ \mathbf{H}'_z\end{pmatrix} = \begin{pmatrix}
\omega & 0 & 0 & \omega & 0 & \omega & \omega\\
0 & \omega & 0 & \omega & \omega & 0 & \omega\\
0 & 0 & \omega & 0 & \omega & \omega & \omega\\
1 & 0 & 0 & 1 & 0 & 1 & 1\\
0 & 1 & 0 & 1 & 1 & 0 & 1\\
0 & 0 & 1 & 0 & 1 & 1 & 1
\end{pmatrix}, \tag{5.19}
$$

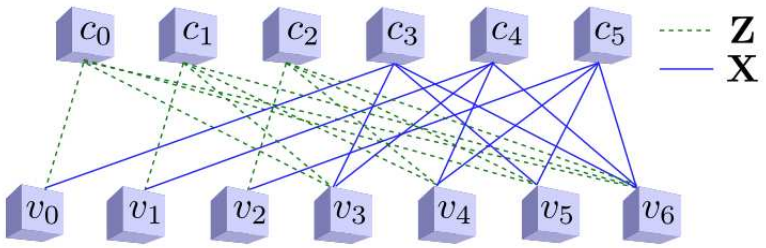

Figura 5.5: Grafo di Tanner del codice di Steane a 7 qubit [28].

Tabella 5.2: Elenco di tutti i possibili valori di $\hat{s}_i$ e dei corrispondenti valori di $\hat{s}_{i+m/2}$ e delle sindromi binarie $s_i = \mathrm{Tr}(\hat{s}_i)$ e $s_{i+m/2} = \mathrm{Tr}(\hat{s}_{i+m/2})$ [28].

$\hat{s}_i$	$\hat{s}_{i+m/2}$	s_i	$s_{i+m/2}$
0	0	0	0
1	$\overline{\omega}$	0	1
ω	1	1	0
$\overline{\omega}$	ω	1	1

il cui grafo di Tanner è riportato nella Fig. 5.5.

Come si evince dalla Fig. 5.5, esistono cicli brevi di lunghezza 4 tra tutti i nodi variabile collegati ai nodi di controllo c_i e c_{i+3}. La natura dual-containing del codice di Steane provoca anche alcuni cicli brevi addizionali, come abbiamo motivato precedentemente per i codici dual-containing.

Tuttavia, in questa sede l'attenzione è rivolta soltanto verso i cicli risultanti dalla struttura CSS omogenea. Per diminuire l'impatto di questi cicli brevi, si propone un grafo di Tanner modificato, che unisce i nodi di controllo c_i e $c_{i+m/2}$ in un unico *supernodo*, eliminando così i cicli. Il grafo di Tanner modificato risultante è dato in Fig. 5.6. Sulla base del grafo di Tanner modificato della Fig. 5.6, i messaggi orizzontali scambiati tra i supernodi $(c_i, c_{i+m/2})$ e i nodi variabile v_t mirano a soddisfare i nodi di controllo che compongono il supernodo, cioè c_i e $c_{i+m/2}$, simultaneamente. Pertanto, bisogna modificare di conseguenza la (5.12) e la (5.13) dell'algoritmo BP non-binario.

Dato che si ha $\hat{\mathbf{H}}_i = \omega\hat{\mathbf{H}}_{i+m/2}$, anche $\hat{s}_i$ e $\hat{s}_{i+m/2}$ sono correlate in modo simile, cioè si ottiene $\hat{s}_i = \omega\hat{s}_{i+m/2}$. Pertanto, la Tabella 5.2 elenca i possibili valori di $\hat{s}_{i+m/2}$ per tutti i valori di $\hat{s}_i$, insieme alle corrispondenti sindromi binarie $s_i = \mathrm{Tr}(\hat{s}_i)$ e $s_{i+m/2} = \mathrm{Tr}(\hat{s}_{i+m/2})$. Dunque, come si evince dalla Tabella 5.2, per ogni valore di s_i (o di $s_{i+m/2}$), ci sono due possibili valori di $\hat{s}_i$ (o di $\hat{s}_{i+m/2}$). Come motivato nella Sezione 5.2, ciò accade perché valgono le seguenti relazioni sulla traccia $\mathrm{Tr}(0) = \mathrm{Tr}(1) = 0$ e $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$. D'altra parte, per ogni coppia $(s_i,\, s_{i+m/2})$, esiste un valore unico di $\hat{s}_i$ e di $\hat{s}_{i+m/2}$.

Di conseguenza, per il supernodo $C_i = (c_i,\, c_{i+m/2})$, le PDF della (5.12)

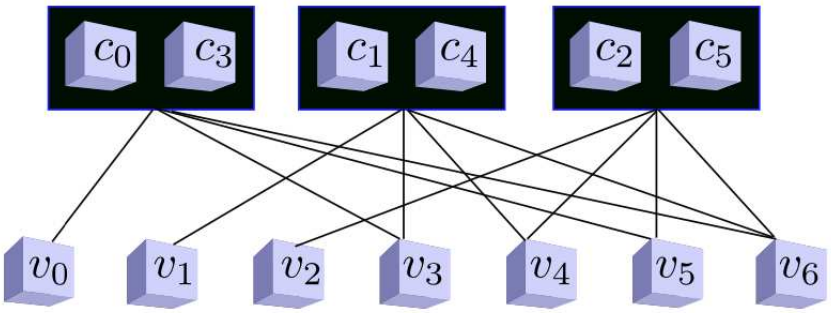

Figura 5.6: Grafo di Tanner modificato del codice di Steane a 7 qubit. I nodi di controllo c_i e $c_{i+m/2}$ sono combinati per formare un unico *supernodo* [28].

e della (5.13) possono essere modificate come segue:

• Se le sindromi di canale osservate sono $(s_i, s_{i+m/2}) = (0, 0)$, allora:

$$PDF\{\check{m}^{\hat{a}_s}_{C_i \to v_t}\} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s}_{v_{t'} \to C_i}\}. \tag{5.20}$$

• Se le sindromi di canale osservate portano a $(s_i, s_{i+m/2}) = (0, 1)$, allora si ottiene:

$$PDF\{\check{m}^{\hat{a}_s}_{C_i \to v_t}\} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s+1}_{v_{t'} \to C_i}\}. \tag{5.21}$$

• Se invece le sindromi di canale osservate soddisfano la seguente relazione $(s_i, s_{i+m/2}) = (1, 0)$, allora si ha:

$$PDF\{\check{m}^{\hat{a}_s}_{C_i \to v_t}\} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s+\omega}_{v_{t'} \to C_i}\}. \tag{5.22}$$

• Se infine per le sindromi di canale osservate vale la seguente relazione $(s_i, s_{i+m/2}) = (1, 1)$, allora si ha:

$$PDF\{\check{m}^{\hat{a}_s}_{C_i \to v_t}\} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s+\overline{\omega}}_{v_{t'} \to C_i}\}, \tag{5.23}$$

dove vale $\hat{a}_s = (\hat{\mathbf{H}}_{it} \times \overline{\hat{a}})$, per $\hat{a} \in \{0, 1, \omega, \overline{\omega}\}$.

Quindi, le eq.(5.20) − (5.23) garantiscono che entrambi i nodi di controllo c_i e $c_{i+m/2}$, che costituiscono il supernodo C_i, siano soddisfatti contemporaneamente. Ciò si ottiene senza alcuna complessità aggiuntiva. Infatti, il metodo proposto in [28] richiede meno calcoli rispetto all'algoritmo standard BP non-binario, perché il numero di nodi di controllo è ridotto alla metà.

Si consideri il codice di Steane della (5.19) per spiegare la procedura di decodifica. Si suppone che, quando la parola di codice a 7 qubit viene trasmessa su un canale di depolarizzazione quantum, con una probabilità di depolarizzazione pari a $p = 0.26$, un errore di Pauli $\mathbf{X}$ viene determinato sul primo qubit, cioè si ottiene $\mathcal{P} = \mathbf{XIIIIII}$. Richiamando la (5.10) per il

calcolo della sindrome per i codici QLDPC quaternari, la sindrome osservata viene calcolata come segue. Prima di tutto si calcola il prodotto della matrice di parità per il coniugato del vettore di errore $\overline{\hat{\mathbf{P}}}$

$$\hat{\mathbf{s}} = \sum_{t \in V(c_i)}^{n} (\hat{\mathbf{H}}_{it} \times \overline{\hat{P}}_t) = \begin{pmatrix} \omega & 0 & 0 & \omega & 0 & \omega & \omega \\ 0 & \omega & 0 & \omega & \omega & 0 & \omega \\ 0 & 0 & \omega & 0 & \omega & \omega & \omega \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \omega \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \ \forall i \quad (5.24)$$

dove $\hat{\mathbf{H}}_{it}$ è l'elemento (i,t)-esimo della PCM e $\overline{\hat{P}}_t$ è l'elemento t-esimo del coniugato del vettore di errore. In seguito si calcola la sindrome per ogni riga del vettore risultante, quindi:

$$\begin{aligned} s_0 &= \mathrm{Tr}(\hat{s}_0) = \mathrm{Tr}(\omega) = 1; \\ s_1 &= \mathrm{Tr}(\hat{s}_1) = \mathrm{Tr}(\omega) = 0; \\ s_2 &= \mathrm{Tr}(\hat{s}_2) = \mathrm{Tr}(\omega) = 0; \\ s_3 &= \mathrm{Tr}(\hat{s}_3) = \mathrm{Tr}(\omega) = 0; \\ s_4 &= \mathrm{Tr}(\hat{s}_4) = \mathrm{Tr}(\omega) = 0; \\ s_5 &= \mathrm{Tr}(\hat{s}_5) = \mathrm{Tr}(\omega) = 0, \end{aligned}$$

quindi il vettore di sindrome risultante è:

$$\mathbf{s} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

A valle di quanto spiegato, si esegue l'algoritmo BP non binario standard sul grafo di Tanner di Fig. 5.3 per stimare l'errore di canale. Esso procede come segue:

- **Inizializzazione**: I messaggi $m^{\hat{a}}_{v_t \to c_i}$, che sono inviati dai nodi variabile $v_t \in \{v_0, v_1, \ldots, v_6\}$ ai nodi di controllo $c_i \in \{c_0, c_1, \ldots, c_5\}$ per $\hat{a} \in \{0, 1, \omega, \overline{\omega}\}$, sono inizializzati secondo la probabilità di depolarizzazione del canale $p = 0.26$, cioè si ottiene:

$$m^{\hat{a}}_{v_t \to c_i} = \begin{cases} 0.74, & \text{se } \hat{a} = 0 \\ 0.0867, & \text{se } \hat{a} \in \{1, \omega, \overline{\omega}\}, \end{cases} \quad (5.25)$$

secondo la (5.16).

- **Scambio orizzontale del messaggio**: I messaggi orizzontali $m^{\hat{a}}_{c_i \to v_t}$ equivalenti a quelli della (5.5), che sono mandati dai nodi di controllo c_i ai

nodi variabile v_t, possono essere elaborati utilizzando l'algoritmo basato sulla FFT di [72]. I suoi passi sono spiegati brevemente nel seguito:

Step 1 (*PDF di* $\check{m}^{\hat{a}_s}_{v_t \to c_i}$) : dalla Sezione 5.2 si ha che:

$$\hat{a}_s = \hat{\mathbf{H}}_{it} \times \overline{\hat{a}}. \tag{5.26}$$

Di conseguenza, la PDF di $\check{m}^{\hat{a}_s}_{v_t \to c_i}$ può essere ottenuta servendosi della PDF corrispondente di $m^{\hat{a}}_{v_t \to c_i}$, in accordo con il valore di $\hat{\mathbf{H}}_{it}$, utilizzando la (5.26). Si consideri che la PDF del messaggio $m^{\hat{a}}_{v_0 \to c_0}$ è equivalente a $(m^0_{v_0 \to c_0}, m^1_{v_0 \to c_0}, m^{\omega}_{v_0 \to c_0}, m^{\overline{\omega}}_{v_0 \to c_0})$. L'ingresso di $\hat{\mathbf{H}}$ corrispondente è $\hat{\mathbf{H}}_{00} = \omega$. Dunque, utilizzando la (5.26) e le regole della moltiplicazione in GF(4), che possono essere recuperate dalla Sezione 2.3.2, si ottiene $\hat{a}_s = (0, \omega, 1, \overline{\omega})$ per $\hat{a} = (0, 1, \omega, \overline{\omega})$. Ciò implica che la PDF di $\check{m}^{\hat{a}_s}_{v_0 \to c_0}$ è equivalente a $(m^0_{v_0 \to c_0}, m^1_{v_0 \to c_0}, m^{\omega}_{v_0 \to c_0}, m^{\overline{\omega}}_{v_0 \to c_0})$.

Per la PCM $\hat{\mathbf{H}}$ della (5.19), si può generalizzare il calcolo di $\check{m}^{\hat{a}_s}_{v_t \to c_i}$, come segue:

$$PDF\{\check{m}^{\hat{a}_s}_{v_t \to c_i}\} = (m^0_{v_t \to c_i}, m^{\omega}_{v_t \to c_i}, m^1_{v_t \to c_i}, m^{\overline{\omega}}_{v_t \to c_i}), \tag{5.27}$$

se $c_i \in \{c_0, c_1, c_2\}$, mentre si ottiene:

$$PDF\{\check{m}^{\hat{a}_s}_{v_t \to c_i}\} = (m^0_{v_t \to c_i}, m^1_{v_t \to c_i}, m^{\omega}_{v_t \to c_i}, m^{\overline{\omega}}_{v_t \to c_i}), \tag{5.28}$$

se $c_i \in \{c_3, c_4, c_5\}$. Inoltre, data la PDF iniziale della (5.25), la (5.27) e la (5.28) si riducono a:

$$\check{m}^{\hat{a}_s}_{v_t \to c_i} = \begin{cases} 0.74, & \text{se } \hat{a}_s = 0 \\ 0.0867, & \text{se } \hat{a}_s \in \{1, \omega, \overline{\omega}\}, \end{cases} \tag{5.29}$$

per $c_i \in \{c_0, c_1, \ldots, c_5\}$.

Step 2 (*FFT della PDF di* $\check{m}^{\hat{a}_s}_{v_t \to c_i}$)*:* Si ricorda dalla Sezione 5.2 che l'operazione di convoluzione richiesta nella (5.12) e nella (5.13) è equivalente a moltiplicare le corrispondenti PDF nel dominio della frequenza. L'FFT della PDF della (5.29) può essere calcolata usando la *matrice della FFT*, come segue:

$$\mathcal{F}\{\check{m}^{\hat{a}_s}_{v_t \to c_i}\} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \check{m}^0_{v_t \to c_i} \\ \check{m}^1_{v_t \to c_i} \\ \check{m}^{\omega}_{v_t \to c_i} \\ \check{m}^{\overline{\omega}}_{v_t \to c_i} \end{pmatrix}, \tag{5.30}$$

dove $\mathcal{F}$ denota l'operazione di FFT. Quindi, l'FFT della PDF della (5.29) è equivalente a:

$$\mathcal{F}\{\check{m}^{\hat{a}_s}_{v_t \to c_i}\} = \begin{pmatrix} 1 \\ 0.6533 \\ 0.6533 \\ 0.6533 \end{pmatrix}. \tag{5.31}$$

Step 3 (Convoluzione delle PDF): Le operazioni di convoluzione della (5.12) e della (5.13), che sono utilizzate per calcolare i messaggi orizzontali

relativi al nodo variabile v_t, possono essere eseguite utilizzando l'algoritmo della FFT come segue:

$$\bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s}_{v_{t'}\to c_i}\} \equiv \mathcal{F}^{-1}\left\{\prod_{v_{t'}} \mathcal{F}\{\check{m}^{\hat{a}_s}_{v_{t'}\to c_i}\}\right\}, \tag{5.32}$$

dove $\mathcal{F}^{-1}$ denota l'operazione di IFFT (Inverse FFT) e $v_{t'} \in V(c_i)\backslash v_t$. Data la PCM $\hat{\mathbf{H}}$ della (5.19) e l'FFT della (5.31), si ottiene:

$$\prod_{v_{t'}} \mathcal{F}\{\check{m}^{\hat{a}_s}_{v_{t'}\to c_i}\} \equiv \begin{pmatrix} 1 \\ 0.2788 \\ 0.2788 \\ 0.2788 \end{pmatrix}. \tag{5.33}$$

Dunque, l'operazione di FFT inversa, applicata alla (5.33), viene realizzata moltiplicando l'equazione stessa con la matrice dell'FFT, cioè la stessa presente nella (5.30). Più esplicitamente, si ha:

$$\begin{aligned} \mathcal{F}^{-1}\left\{\prod_{v_{t'}} \mathcal{F}\{\check{m}^{\hat{a}_s}_{v_{t'}\to c_i}\}\right\} &= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0.2788 \\ 0.2788 \\ 0.2788 \end{pmatrix} \\ &= \begin{pmatrix} 1.8364 \\ 0.7212 \\ 0.7212 \\ 0.7212 \end{pmatrix}. \end{aligned} \tag{5.34}$$

Infine, la PDF della (5.34) è normalizzata per produrre il risultato della (5.32), cioè si ottiene:

$$\bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s}_{v_{t'}\to c_i}\} \equiv \begin{pmatrix} 0.4591 \\ 0.1803 \\ 0.1803 \\ 0.1803 \end{pmatrix}.$$

Step 4 (PDF di $\check{m}^{\hat{a}_s}_{c_i\to v_t}$): La PDF dei messaggi $\check{m}^{\hat{a}_s}_{c_i\to v_t}$ può essere elaborata utilizzando la (5.12) oppure la (5.13), in base al valore della sindrome osservata, che è stata calcolata tramite la (5.24). Poiché la sindrome derivata dalla (5.24) vale 1 per il primo nodo di controllo c_0, si usa la (5.13) per calcolare la PDF dei messaggi uscenti dal nodo di controllo c_0. Dunque si ottiene:

$$\check{m}^{\hat{a}_s}_{c_0\to v_t} = \begin{pmatrix} 0.1803 \\ 0.1803 \\ 0.3197 \\ 0.3197 \end{pmatrix}. \tag{5.35}$$

Inoltre, la sindrome della (5.24) ha valore pari a 0 per tutti gli altri nodi di controllo. Pertanto, si usa la (5.12) per $c_i \neq c_0$, che produce la seguente

PDF:

$$\check{m}_{c_i \to v_t}^{\hat{a}_s} = \begin{pmatrix} 0.3197 \\ 0.3197 \\ 0.1803 \\ 0.1803 \end{pmatrix}. \tag{5.36}$$

Step 5 (PDF di $\check{m}_{c_i \to v_t}^{\hat{a}}$): Con l'intento di recuperare i messaggi $m_{c_i \to v_t}^{\hat{a}}$ dalla PDF di $\check{m}_{c_i \to v_t}^{\hat{a}_s}$, i valori delle PDF risultanti dalla (5.35) e dalla (5.36) devono essere scambiati, come si è fatto nello Step 1. In particolare, l'operazione di scambio, che è richiesta per questa fase, è l'inverso di quella effettuata nello Step 1. Si considerino ora i nodi di controllo $c_i \in \{c_0, c_1, c_2\}$, per i quali i valori diversi da zero di $\hat{\mathbf{H}}$ sono sempre uguali a ω (o, equivalentemente, facendo riferimento al grafo di Tanner della Fig. 5.3, tutti i rami uscenti da questi nodi di controllo sono contrassegnati con l'operatore di Pauli $\mathbf{Z}$). Inoltre, si ricorda dallo Step 1 che $\hat{a}_s = (0, \omega, 1, \overline{\omega})$ per $\hat{a} = (0, 1, \omega, \overline{\omega})$, quando $\hat{\mathbf{H}}_{it} = \omega$. Questo implica che la PDF di $m_{c_i \to v_t}^{\hat{a}}$ è equivalente a $(\check{m}_{c_i \to v_t}^{0}, \check{m}_{c_i \to v_t}^{\omega}, \check{m}_{c_i \to v_t}^{1}, \check{m}_{c_i \to v_t}^{\overline{\omega}})$, per $c_i \in \{c_o, c_1, c_2\}$. Per tutti gli altri nodi di controllo, la PDF di $m_{c_i \to v_t}^{\hat{a}}$ è la stessa del messaggio $\check{m}_{c_i \to v_t}^{\hat{a}_s}$, dato che si ha $\hat{\mathbf{H}}_{it} = 1$. Dunque, le PDF risultanti sono come segue:

$$m_{c_i \to v_t}^{\hat{a}} = \begin{pmatrix} 0.1803 \\ 0.3197 \\ 0.1803 \\ 0.3197 \end{pmatrix},$$

per $c_i = c_0$, mentre si ottiene:

$$m_{c_i \to v_t}^{\hat{a}} = \begin{pmatrix} 0.3197 \\ 0.1803 \\ 0.3197 \\ 0.1803 \end{pmatrix},$$

per $c_i \in \{c_1, c_2\}$, infine si ha:

$$m_{c_i \to v_t}^{\hat{a}} = \begin{pmatrix} 0.3197 \\ 0.3197 \\ 0.1803 \\ 0.1803 \end{pmatrix},$$

per tutti gli altri nodi di controllo $c_i \in \{c_3, c_4, c_5\}$.

- **Scambio verticale del messaggio**: Successivamente si calcolano i messaggi verticali $m_{v_t \to c_i}^{\hat{a}}$ utilizzando la (5.6). Ad esempio, si consideri il messaggio $m_{v_0 \to c_0}^{\hat{a}}$, che è destinato dal nodo variabile v_0 al nodo di controllo c_0. Poiché nel grafo di Tanner della Fig. 5.3, il nodo variabile v_0 è collegato soltanto ai nodi di controllo c_0 e c_3, il messaggio $m_{v_0 \to c_0}^{\hat{a}}$ può essere calcolato come:

$$m_{v_0 \to c_0}^{\hat{a}} = K \mathrm{P}_{\mathrm{ch}}(P_0 = \hat{a}) \times m_{c_3 \to v_0}^{\hat{a}} = \begin{pmatrix} 0.8005 \\ 0.0937 \\ 0.0529 \\ 0.0529 \end{pmatrix}.$$

Tabella 5.3: Probabilità marginale $\Pr(P_t = \hat{a}|\mathbf{s})$ dopo la prima iterazione, quando viene utilizzato l'algoritmo di decodifica BP non binario standard sul grafo di Tanner del codice di Steane a 7 qubit per la trasmissione attraverso un canale di depolarizzazione, avente $p = 0.26$, che determina un errore di Pauli **X** sul primo qubit (si ha $\mathcal{P} = \mathbf{XIIIIII}$) [28].

t	$\hat{a} = 0$	$\hat{a} = 1$	$\hat{a} = \omega$	$\hat{a} = \overline{\omega}$	$\tilde{P}_t$
0	0.7189	0.1493	0.0475	0.0842	0
1	0.8552	0.0565	0.0565	0.03185	0
2	0.8552	0.0565	0.0565	0.03185	0
3	0.8392	0.0983	0.0313	0.0313	0
4	0.9205	0.0343	0.0343	0.0109	0
5	0.8392	0.0983	0.0313	0.0313	0
6	0.9100	0.0601	0.0191	0.0108	0

- **Probabilità marginale element-wise**: Le probabilità marginali element-wise dell'errore sul nodo variabile v_t, data la sindrome osservata $\mathbf{s}$, possono essere calcolate utilizzando la (5.7). Si consideri di nuovo il nodo variabile v_0, che è connesso ai nodi di controllo c_0 e c_3. Di conseguenza, la distribuzione di probabilità marginale dell'errore P_t risultante, inflitto al nodo variabile v_t, può essere calcolata come segue:

$$\Pr(P_0 = \hat{a}|\mathbf{s}) = K\mathrm{P}_{\mathrm{ch}}(P_0 = \hat{a}) \times m^{\hat{a}}_{c_0 \to v_0} \times m^{\hat{a}}_{c_3 \to v_0} = \begin{pmatrix} 0.7189 \\ 0.1493 \\ 0.0475 \\ 0.0842 \end{pmatrix}.$$

Il processo viene ripetuto per tutti i nodi variabile. Le probabilità marginali risultanti sono raccolte nella Tabella 5.3.

- **Decisione hard e controllo della sindrome**: Infine, viene utilizzata la decisione hard per trovare l'errore $\tilde{P}_t$ più probabile che massimizza la probabilità marginale calcolata nello step precedente. I valori risultanti di $\tilde{P}_t$ sono elencati nell'ultima colonna della Tabella 5.3. In particolare, si osserva che la probabilità di "no-error", cioè di assenza di errore, domina per tutti i nodi variabile. La sindrome specifica, riferita all'errore stimato $\tilde{P}_t$ risultante, non corrisponde alla sindrome osservata $\mathbf{s}$ della (5.24). Quindi, l'algoritmo si ripete dalla fase di scambio del messaggio orizzontale in poi.

La Fig. 5.7a grafica la probabilità marginale risultante $\Pr(P_t = \hat{a}|\mathbf{s})$ per il primo qubit, al procedere delle iterazioni. Come visto dalla Fig. 5.7a, l'algoritmo di decodifica standard non riesce a convergere. Successivamente, si utilizza l'algoritmo BP non binario modificato proposto da [28] per analizzare l'impatto e le differenze, dal punto di vista delle prestazioni, rispetto al

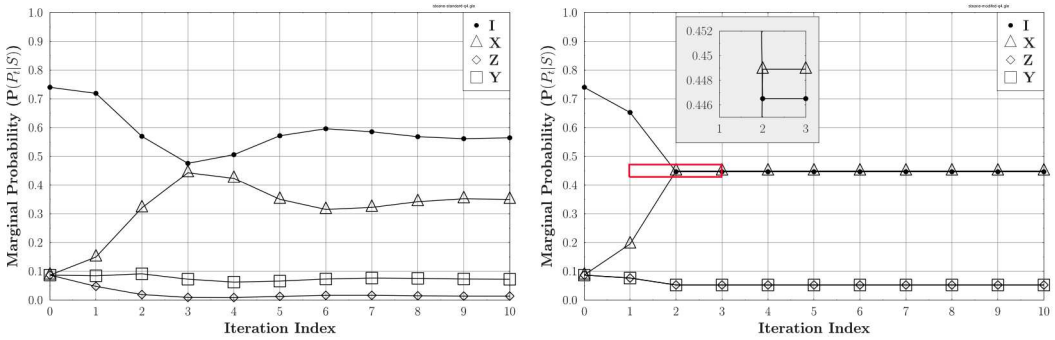

Figura 5.7: Evoluzione della probabilità marginale per il primo qubit del codice di Steane a 7 qubit per la trasmissione attraverso un canale di depolarizzazione, avente $p = 0.26$, che determina un errore di Pauli **X** sul primo qubit, cioè si ha $\mathcal{P} = \mathbf{XIIIIII}$. Come si può osservare, l'algoritmo standard BP non riesce a convergere, mentre il BP modificato proposto da [28] converge alla soluzione corretta in due iterazioni [28]. (a) BP non binario standard. (b) BP non binario modificato.

precedente. Si ricorda dalla Fig. 5.6 che i nodi di controllo c_i e c_{i+3} sono uniti in un unico supernodo C_i. I corrispondenti valori della sindrome osservati della (5.24) sono anch'essi uniti tra di loro e producono i seguenti risultati: $(s_0, s_3) = (1, 0)$, $(s_1, s_4) = (0, 0)$ e $(s_2, s_5) = (0, 0)$. Di conseguenza, l'algoritmo BP modificato differisce dal BP standard nello Step 4 dello **scambio orizzontale del messaggio**, poiché tiene conto dei supernodi, piuttosto che dei singoli nodi di controllo. Servendosi delle eq.(5.20) − (5.23), lo Step 4 dello **scambio orizzontale del messaggio** può essere effettuato come segue:

Step 4 (PDF di $\check{m}^{\hat{a}_s}_{c_i \to v_t}$): Poiché la sindrome osservata per il supernodo C_0 è $(s_0, s_3) = (1, 0)$, si utilizza la (5.22) per calcolare la PDF dei messaggi uscenti da questo supernodo. Di conseguenza, si arriva a:

$$\check{m}^{\hat{a}_s}_{C_0 \to v_t} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s + \omega}_{v_{t'} \to C_0}\} = \begin{pmatrix} 0.1803 \\ 0.1803 \\ 0.4591 \\ 0.1803 \end{pmatrix}.$$

Inoltre, poiché per tutti gli altri supernodi la sindrome è $(s_i, s_{i+3}) = (0, 0)$, si usa la (5.20) per calcolare le PDF corrispondenti. Quindi, si ottiene:

$$\check{m}^{\hat{a}_s}_{C_i \to v_t} = \bigotimes_{v_{t'}} PDF\{\check{m}^{\hat{a}_s}_{v_{t'} \to C_i}\} = \begin{pmatrix} 0.4591 \\ 0.1803 \\ 0.1803 \\ 0.1803 \end{pmatrix},$$

per $C_i \in \{C_1, C_2\}$.

La parte restante dell'algoritmo di decodifica è la stessa del BP non binario standard, sebbene in questo caso si abbiano tre supernodi nel grafo

Tabella 5.4: Probabilità marginale $\Pr(P_t = \hat{a}|\mathbf{s})$ dopo la prima iterazione, quando si utilizza l'algoritmo di decodifica BP non binario modificato sul grafo di Tanner del codice di Steane a 7 qubit, per la trasmissione attraverso un canale di depolarizzazione, avente $p = 0.26$, che determina un errore di Pauli **X** sul primo qubit (si ha $\mathcal{P} = \mathbf{XIIIIII}$) [28].

t	$\hat{a} = 0$	$\hat{a} = 1$	$\hat{a} = \omega$	$\hat{a} = \overline{\omega}$	$\tilde{P}_t$
0	0.1946	0.6525	0.0764	0.0764	0
1	0.8788	0.0404	0.0404	0.0404	0
2	0.8788	0.0404	0.0404	0.0404	0
3	0.8271	0.0969	0.0380	0.0380	0
4	0.9486	0.0171	0.0171	0.0171	0
5	0.8271	0.0969	0.0380	0.0380	0
6	0.9241	0.0425	0.0167	0.0167	0

di Tanner modificato della Fig. 5.6, differentemente dai sei nodi di controllo del grafo di Tanner della Fig. 5.5. Le probabilità marginali risultanti sono riportate nella Tabella 5.4, inoltre la Fig. 5.7b grafica la probabilità marginale per il primo qubit, mentre le iterazioni procedono. Si può infine osservare dalla Fig. 5.7b che l'algoritmo BP modificato proposto da [28] converge, alla stima corretta, in appena due iterazioni.

5.5 Algoritmo BP riponderato per grafi con cicli

L'algoritmo Belief Propagation è in grado di fornire un'approssimazione ragionevolmente buona al problema di ottimizzazione della (5.3), a condizione che il grafo di Tanner sottostante abbia una girth sufficientemente elevata. Tuttavia non è garantito che l'algoritmo converga, infatti può anche capitare che, in presenza di cicli, converga su una soluzione errata [78], [79]. Inoltre, può richiedere un gran numero di iterazioni per raggiungere la convergenza, soprattutto in presenza di rumore elevato, imponendo così una maggiore complessità.

Queste carenze dell'algoritmo BP classico sono dovute principalmente al fatto che i messaggi, se ci sono cicli brevi nel grafo di Tanner associato, diventano dipendenti rispetto al tempo. In alternativa, tali messaggi vengono interpretati come "over-confident" oppure "over-estimated". Per diminuire l'impatto di questi messaggi, Wainwright *et al.* hanno concepito in [78] il metodo Tree-Reweighted Belief Propagation (TRW-BP), che lavora per interazioni a coppie e migliora la convergenza del BP classico, ricalcolando i rami del grafo sottostante con le Edge Appearance Probabilities (EAP), nozione da loro proposta. L'EAP di un ramo rappresenta la probabilità di apparizione di quel ramo in un albero di spanning scelto arbitrariamente, a

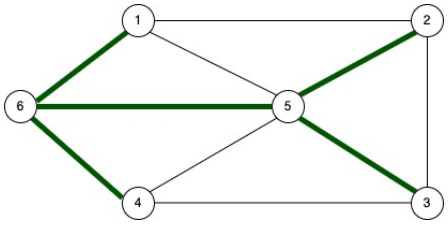

Figura 5.8: Esempio di un semplice albero di spanning. Ci sono $n = 6$ vertici, collegati da $n - 1 = 5$ archi.

partire dal grafo di partenza.

Approfondimento: Albero di spanning Un albero di spanning (chiamato anche albero ricoprente) è un albero, rintracciabile all'interno di un grafo, che contiene tutti i vertici del grafo ma soltanto un sottoinsieme degli archi, cioè solo quelli necessari per connettere tra loro tutti i vertici con uno e un solo cammino. Ne è riportato un esempio nella Fig. 5.8.

Di seguito vengono elencate le principali proprietà degli alberi di spanning:

• Possiede $n - 1$ archi, dove n è il numero dei vertici.

• Possiede un numero di archi minimale per la proprietà di connessione: rimuovendo un arco qualsiasi, il grafo non è più connesso.

• Possiede un numero di archi massimale per la proprietà di aciclicità: aggiungendo un arco fra due vertici qualsiasi, il grafo non è più aciclico.

• All'interno dell'albero esiste un unico cammino semplice fra due nodi.

L'algoritmo TRW-BP è stato esteso alle interazioni di ordine superiore in [80] e [81], dove gli EAP sono stati sostituiti dalle Factor Appearance Probabilities (FAP) dei nodi. La FAP indica la probabilità di apparizione di un nodo di controllo in un potenziale albero di spanning, considerato tra tutti i possibili alberi di spanning costruiti a partire dal grafo considerato [80], [81].

Sulla base dell'algoritmo TRW-BP esteso, Wymeersch *et al.* hanno riformulato la fase di scambio verticale del messaggio del BP classico, presente nella (5.6), come segue:

$$m^{a}_{v_t \to c_i} = K \mathrm{P}_{\mathrm{ch}}(P_t = a)(m^{a}_{c_i \to v_t})^{\rho_i - 1} \prod_{c_{i'} \in C(v_t) \setminus c_i} (m^{a}_{c_{i'} \to v_t})^{\rho_{i'}}, \tag{5.37}$$

dove ρ_i è la FAP dell'i-esimo nodo di controllo [80], [81]. Analogamente, la valutazione della probabilità marginale element-wise della (5.7) è stata conseguentemente modificata:

$$\Pr(P_t = a|\mathbf{s}) = K \mathrm{P}_{\mathrm{ch}}(P_t = a) \prod_{c_i \in C(v_t)} (m^{a}_{c_i \to v_t})^{\rho_i}. \tag{5.38}$$

Sia la (5.37) che la (5.38) si riducono all'algoritmo BP classico per $\rho_i = 1, \forall\, i$.

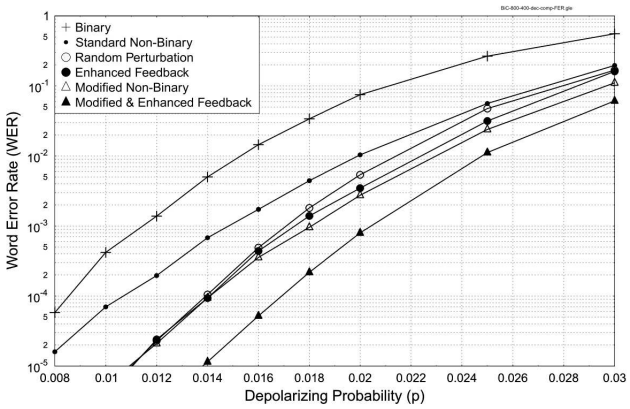

Figura 5.9: Confronto delle prestazioni del WER ottenibile dell'algoritmo BP modificato rispetto agli altri schemi di decodifica esistenti, utilizzando i parametri di simulazione della Tabella 5.5 [28].

La tecnica TRW-BP richiede l'ottimizzazione di ρ_i per tutti i nodi. Per ridurre questo lavoro di ottimizzazione, Wymeersch *et al.* in [80], [81] hanno anche proposto l'algoritmo URW-BP, che prevede di considerare un valore FAP uniforme per tutti i nodi, cioè si ha $\rho_i = \rho, \forall\, i$.

Altre variazioni del TRW-BP, per i codici LDPC binari classici, sono state studiate in [82], [83], [84] e [85]. Queste dimostrano che il TRW-BP migliora effettivamente la convergenza dei codici LDPC binari quando il numero di iterazioni non è troppo alto. Sulla base di questi risultati, nella prossima Sezione 5.7 si analizza anche l'impatto dell'algoritmo URW-BP sulla decodifica non binaria dei codici QLDPC.

5.6 Risultati riguardanti la decodifica non binaria modificata

Con l'intento di quantificare il guadagno in termini di prestazioni ottenibili dell'algoritmo BP non binario modificato della Sezione 5.4, proposto da Babar *et al.* in [28], in questo capitolo si confrontano le sue performance rispetto a quelle degli altri algoritmi di decodifica trattati nel Capitolo 5.

Il primo sistema studiato è rappresentato nella Tabella 5.5 e si basa sul codice $[1600, 400]$ a doppio ciclo con rate $R_Q = \frac{1}{4}$ di MacKay, in cui ogni riga della PCM associata ha peso pari a 30. La prestazione del WER corrispondente, registrato per varie probabilità del canale di depolarizzazione, è riportato nella Fig. 5.6, dove si sono considerati i seguenti decodificatori:

1) **Binario:** l'algoritmo di decodifica BP binario della Sezione 5.1;

2) **Binario non Standard:** l'algoritmo di decodifica BP non binario della Sezione 5.2;

3) **Random Perturbation:** la tecnica Random Perturbation [75] della Sezione 5.3;

4) **Enhanced Feedback:** il metodo Enhanced Feedback [77] della Sezione 5.3;

Tabella 5.5: Sistema 1: parametri usati per la simulazione [28].

Matrice QLDPC $\mathbf{H}_z$	
Tipo di codice	Codice di Mackay a doppio ciclo
Qubit codificati	$n = 1600$
Qubit di informazione	$k = 400$
E-bit	$c = 0$
Peso della riga	30
Decodificatore QLDPC	
Iterazioni della decodifica standard	$\mathrm{I}_{\max} = 90$
Iterazioni Random	$T_{\mathrm{pert}} = 40$
Indice di perturbazione casuale	$\delta = 0.1$
Numero massimo di feedback	$n_a = 40$

5) **BP non binario modificato:** l'algoritmo BP modificato non binario [28] della Sezione 5.4;

6) **BP modificato & Enhanced Feedback:** l'algoritmo BP modificato non binario della Sezione 5.4 [28], utilizzato in modo congiunto con il metodo Enhanced Feedback [77], della Sezione 5.3.

Per tutti gli schemi di decodifica si sono usate un massimo di $\mathrm{I}_{\max} = 90$ iterazioni. Inoltre, sia per la tecnica Random Perturbation che per il metodo Enhanced Feedback, si è impostato $T_{\mathrm{pert}} = 40$, mentre l'indice di perturbazione casuale è $\delta = 0.1$ e il numero massimo di feedback è stato fissato a $n_a = 40$. Per l'Enhanced Feedback sono stati usati i parametri di decodifica di [77] che sono riportati nella Tabella 5.5. Ogni algoritmo di decodifica itera finché non viene trovato un errore valido o fino a quando non viene raggiunto il numero massimo di iterazioni. Inoltre, il WER considera sia gli errori di blocco rilevati che quelli non rilevati.

Si può osservare nella Fig. 5.9 che il decodificatore binario mostra le prestazioni peggiori. Utilizzando il decodificatore binario infatti, si ottiene un WER di 10^{-4} ad una probabilità del canale di depolarizzazione pari a $p = 0.0085$, che aumenta a $p = 0.010\,75$ con il decodificatore standard non binario. Ciò significa che il decodificatore può far fronte ad un incremento della probabilità di depolarizzazione pari al $(\frac{(0.010\,75-0.0085)}{0.0085}) \times 100) = 26\%$. Inoltre, i decodificatore della tecnica Random Perturbation, del metodo Enhanced Feedback e dell'algoritmo BP non binario modificato sono caratterizzati da prestazioni simili a bassi livelli di rumore, aumentando così la probabilità di depolarizzazione tollerabile a $p = 0.014$ ad un WER di 10^{-4}, che corrisponde ad un aumento di p del $(\frac{(0.014-0.010\,75)}{0.010\,75} \times 100) = 30\%$, rispetto al decodificatore standard non binario. Inoltre, con la configurazione BP modificato & Enhanced Feedback, la probabilità di depolarizzazione tollerabile aumenta a $p = 0.017$ ad un WER di 10^{-4}, che è equivalente ad un aumento del $(\frac{(0.017-0.014)}{0.014} \times 100) = 21\%$ circa, rispetto a $p = 0.014$. La Tabella 5.6 riassume questi risultati.

Le prestazioni del BP non binario modificato ad un WER di 10^{-4} sono simili a quelle dei metodi euristici, vale a dire Random Perturbation ed

Tabella 5.6: Probabilità di depolarizzazione realizzabile (p) con un WER di 10^{-4}, basata sulla Fig. 5.9 [28].

N.	Metodo di decodifica	p	Miglioramento
1	Binario	0.0085	-
2	Standard non binario	0.010 75	26% w.r.t. Dec. 1
3	Random Perturbation	0.014	30% w.r.t. Dec. 2
4	Enhanced Feedback	0.014	30% w.r.t. Dec. 2
5	BP modificato non binario	0.014	30% w.r.t. Dec. 2
6	BP modificato & Enhanced Feedback	0.017	21% w.r.t. Dec. 5

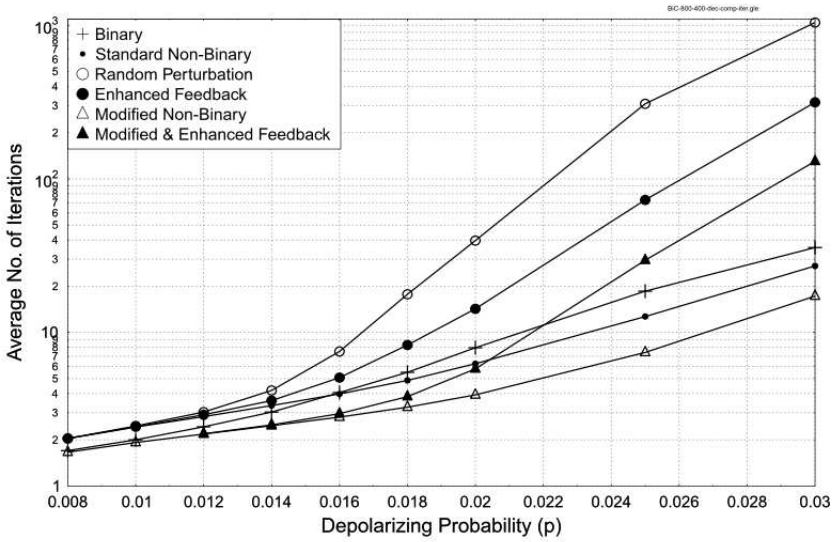

Figura 5.10: Confronto tra il numero medio di iterazioni di decodifica effettuate dalla tecnica BP modificata e gli schemi di decodifica esistenti utilizzando i parametri di simulazione della Tabella 5.5 [28].

Enhanced Feedback. Tuttavia, la tecnica BP non binaria modificata impone una minore complessità di decodifica in termini di numero medio di iterazioni di decodifica, come evidenziato nella Fig. 5.10. Di conseguenza, si può osservare che il BP non binario modificato proposto in [28] converge più velocemente rispetto agli altri schemi di decodifica esistenti. In particolare, in presenza di alti livelli di rumore, il decodificatore non binario modificato di [28] ottiene prestazioni migliori, sia rispetto al decodificatore del Random Perturbation che dell'Enhanced Feedback. Ciò avviene sia in termini di WER, come appare in Fig. 5.9, sia in termini di numero medio di iterazioni, viste in Fig. 5.10. Rispetto al decodificatore non binario standard, l'algoritmo non binario modificato produce sempre un WER inferiore e utilizza in media meno iterazioni di decodifica. Inoltre, dalla Fig. 5.10 si può osservare che il metodo BP modificato & Enhanced Feedback si serve di meno iterazioni rispetto al metodo Enhanced Feedback. In aggiunta, la prestazione del primo è anche superiore in termini di curva WER, come si

Tabella 5.7: Confronto delle prestazioni in termini di WER realizzabile e di numero medio di iterazioni di decodifica (I_{avg}) per una probabilità di depolarizzazione di $p = 0.016$, sulla base delle curve della Fig. 5.9 e della Fig. 5.10 [28].

N.	Metodo di decodifica	WER	I_{avg}
1	Binario	$1.57 \cdot 10^{-2}$	3.98
2	Standard non binario	$1.47 \cdot 10^{-3}$	4.28
3	Random Perturbation	$4.57 \cdot 10^{-4}$	7.52
4	Enhanced Feedback	$4.57 \cdot 10^{-4}$	5.08
5	BP modificato non binario	$3.67 \cdot 10^{-4}$	2.81
6	BP modificato & Enhanced Feedback	$5.27 \cdot 10^{-5}$	2.96

Tabella 5.8: Sistema 2: parametri usati per la simulazione [28].

Matrice QLDPC	
Tipo di codice	Codice EA-QLDPC omogeneo
Qubit codificati	$n = 816$
Qubit di informazione	$k = 404$
E-bit	$c = 404$
Peso della riga	6
Peso della colonna	3
Decodificatore QLDPC	
Iterazioni della decodifica standard	$I_{max} = 90$
Iterazioni Random	$T_{pert} = 40$
Indice di perturbazione casuale	$\delta = 0.1$
Numero massimo di feedback	$n_a = 81$

può osservare dalla Fig. 5.9.

Ciò è dovuto al fatto che il BP modificato della Sezione 5.4 è caratterizzato da una convergenza più veloce rispetto alla decodifica non binaria standard. In particolare per $p \leq 0.017$, nella regione di interesse corrispondente al WER desiderato di 10^{-4}, il metodo BP modificato & Enhanced Feedback, impone quasi la stessa complessità di decodifica rispetto a quella dell'algoritmo BP non binario modificato, quando è usato da solo. Tuttavia, il primo presenta un WER molto più basso del secondo. Nella Tabella 5.7 si confrontano le prestazioni di tutti gli schemi di decodifica citati con una probabilità di depolarizzazione di $p = 0.016$.

Di seguito invece, si confrontano le prestazioni dei diversi schemi di decodifica del secondo sistema di [28], basandosi sul codice EA-QLDPC omogeneo di [77], avente $n = 816$, $k = 404$ e $c = 404$, che deriva dal codice LDPC$(816, 408)$ classico di MacKay, con peso della riga pari a 6 e peso della colonna di 3. Per tutti gli schemi di decodifica, si è usato un massimo di $I_{max} = 90$ iterazioni. Inoltre, per i metodi Random Perturbation ed Enhanced Feedback si è scelto $T_{pert} = 40$, mentre l'indice di perturbazione casuale è fissato a $\delta = 0.1$ e per il decodificatore Enhanced Feedback il numero

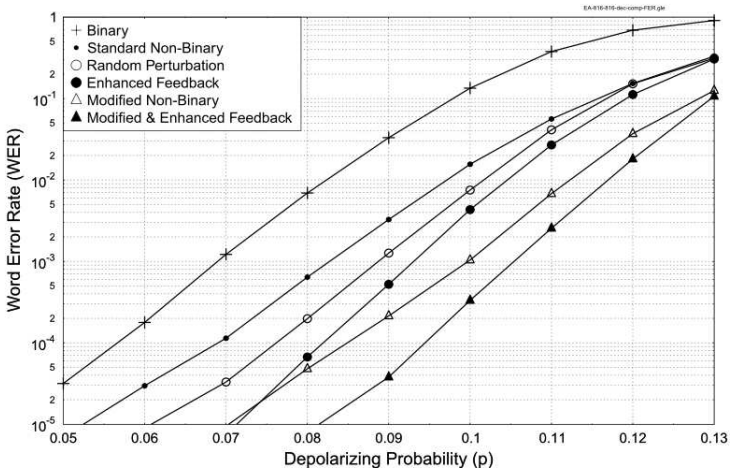

Figura 5.11: Confronto delle prestazioni del WER ottenibile tra il BP modificato e gli schemi di decodifica esistenti, utilizzando i parametri di simulazione della Tabella 5.8 [28].

massimo di feedback previsto è $n_a = 81$. Questi parametri di simulazione sono riassunti nella Tabella 5.8. Le curve di prestazione del WER risultanti vengono confrontate nella Fig. 5.11, mentre il numero medio di iterazioni di decodifica, invocate per diverse probabilità del canale di depolarizzazione, viene confrontato nella Fig. 5.12.

Come osservato dalla Fig. 5.11, il decodificatore binario raggiunge un WER di 10^{-4} con una $p = 0.057$, che aumenta a $p = 0.069$ quando viene utilizzato il decoder non binario standard. Di conseguenza, il decodificatore non binario standard aumenta la probabilità di depolarizzazione tollerabile del $(\frac{(0.069-0.057)}{0.057} \times 100) = 21\%$ rispetto al decodificatore binario. Questo incremento è ulteriormente aumentato a $p = 0.076$ nel caso del metodo Random Perturbation, che corrisponde ad un aumento del $(\frac{(0.076-0.069)}{0.069} \times 100) = 10\%$ circa e a $p = 0.082$ per l'Enhanced Feedback, che rappresenta un aumento del $(\frac{(0.082-0.069)}{0.069} \times 100) = 19\%$. Al contrario, l'algoritmo BP modificato non binario, proposto da [28], mostra un WER di 10^{-4} intorno a $p = 0.085$, che corrisponde ad un aumento del $(\frac{(0.085-0.069)}{0.069} \times 100) = 23\%$ rispetto al decodificatore standard non binario. Invece, il metodo BP mdoficato & Enhanced Feedback fornisce un ulteriore aumento fino a $p = 0.0945$, che rappresenta un incremento del $(\frac{(0.0945-0.085)}{0.085} \times 100) = 11\%$.

Questi risultati sono riportati nella Tabella 5.9. In termini di numero medio di iterazioni di decodifica, il BP non binario modificato di [28] ha sempre prestazioni migliori sia rispetto al decodificatore non binario standard, sia rispetto alle soluzioni Random Perturbation ed Enhanced Feedback, come illustrato nella Fig. 5.12.

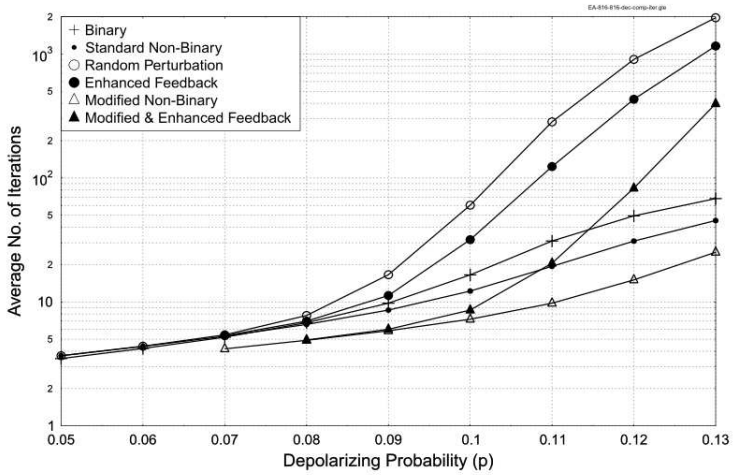

Figura 5.12: Confronto tra il numero medio di iterazioni di decodifica invocate dal BP modificato e gli altri schemi di decodifica esistenti, utilizzando i parametri di simulazione della Tabella 5.8 [28].

Tabella 5.9: Probabilità di depolarizzazione realizzabile (p) con un WER di 10^{-4}, basata sulla Fig. 5.9 [28].

N.	Metodo di decodifica	p	Miglioramento
1	Binario	0.057	-
2	Standard non binario	0.069	21% w.r.t. Dec. 1
3	Random Perturbation	0.076	10% w.r.t. Dec. 2
4	Enhanced Feedback	0.082	19% w.r.t. Dec. 2
5	BP modificato non binario	0.085	23% w.r.t. Dec. 2
6	BP modificato & Enhanced Feedback	0.0945	11% w.r.t. Dec. 5

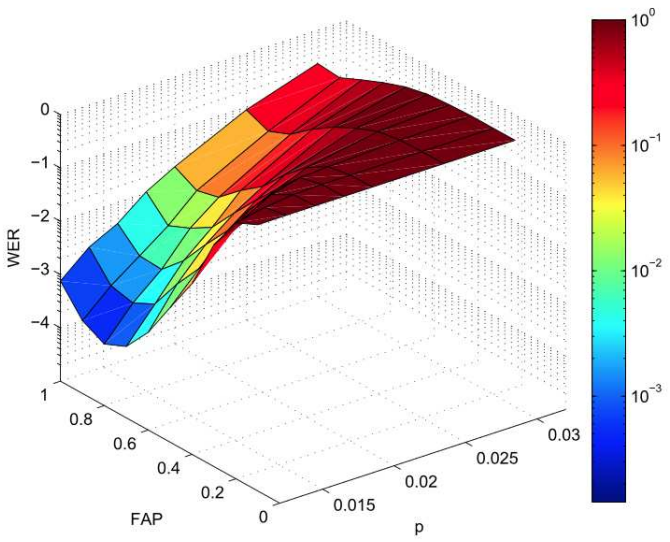

Figura 5.13: Ottimizzazione URW-BP: impatto dei valori FAP variabili sulle prestazioni del WER a varie probabilità del canale di depolarizzazione p. $I_{max} = 10$ iterazioni [28].

5.7 Risultati riguardanti l'algoritmo BP Riponderato Uniformemente

Dato che i codici a doppio ciclo presentano numerosi cicli brevi, per l'analisi dell'URW-BP (URW − Uniformly-Reweighted BP) della Sezione 5.5, che viene combinato con il decodificatore non binario modificato della Sezione 5.4 di [28], si utilizza il primo sistema della Tabella 5.5. In particolare, si uniscono lo Step di scambio orizzontale del messaggio dell'algoritmo BP non binario modificato con lo step di scambio verticale del messaggio del metodo URW-BP.

Si inizia determinando euristicamente il valore ottimale ρ del FAP, che varia sia con la probabilità del canale di depolarizzazione che con il numero massimo di iterazioni di decodifica. Le Fig. 5.13, 5.14 e 5.15 mostrano l'impatto di ρ sulle prestazioni del WER a diverse probabilità del canale di depolarizzazione p; per $I_{max} = 10$, 20 e 90 iterazioni, rispettivamente. Dalle Fig. 5.13 − 5.15, si può osservare che il WER varia al variare di ρ, raggiungendo il valore minimo quando ρ arriva al suo ottimo. Il valore ottimale di ρ è diverso per ogni valore di p, poiché tende a $\rho = 1$ all'aumentare del valore di p o all'aumentare del numero massimo accettabile di iterazioni I_{max}. I valori ottimizzati di ρ, risultanti per diverse probabilità del canale di depolarizzazione p e per un diverso numero massimo di iterazioni, sono raccolti nella Tabella 5.10.

Per quantificare il guadagno di prestazione ottenuto con l'URW-BP, si confrontano nella Fig. 5.16 le prestazioni dell'URW-BP ottimizzato con il BP non binario modificato, proposto da [28], per $I_{max} = 10$, 20 e 90 iterazioni. In questa sede, l'URW-BP ottimizzato si basa sui migliori valori di ρ elencati nella Tabella 5.10. Le curve di prestazione della Fig. 5.16 rivelano

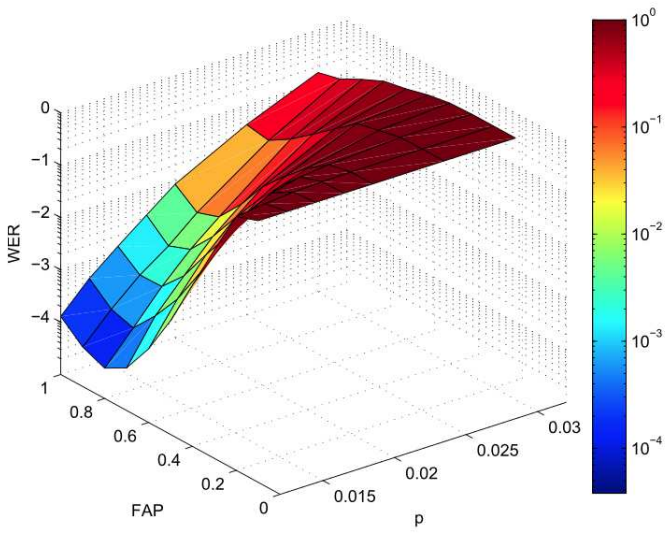

Figura 5.14: Ottimizzazione URW-BP: impatto dei valori FAP variabili sulle prestazioni del WER a varie probabilità del canale di depolarizzazione p. $I_{max} = 20$ iterazioni [28].

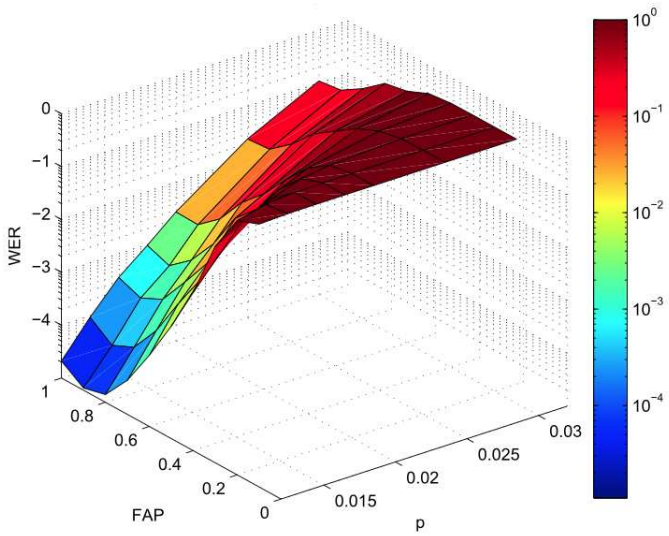

Figura 5.15: Ottimizzazione URW-BP: impatto dei valori FAP variabili sulle prestazioni del WER a varie probabilità del canale di depolarizzazione p. $I_{max} = 90$ iterazioni [28].

Tabella 5.10: Ottimizzazione di ρ per diversi valori di p e numero massimo di iterazioni I_{max} per il sistema 1 della Tabella 5.5, sulla base delle curve delle Fig. 5.13 - 5.15 [28].

I_{max}	p **0.012**	p **0.014**	p **0.016**	p **0.018**	p **0.02**	p **0.025**	p **0.03**
10	0.8	0.8	0.8	0.9	0.9	0.9	0.9
20	0.8	0.8	0.9	0.9	0.9	0.9	1
90	0.9	0.9	0.9	0.9	0.9	1	1

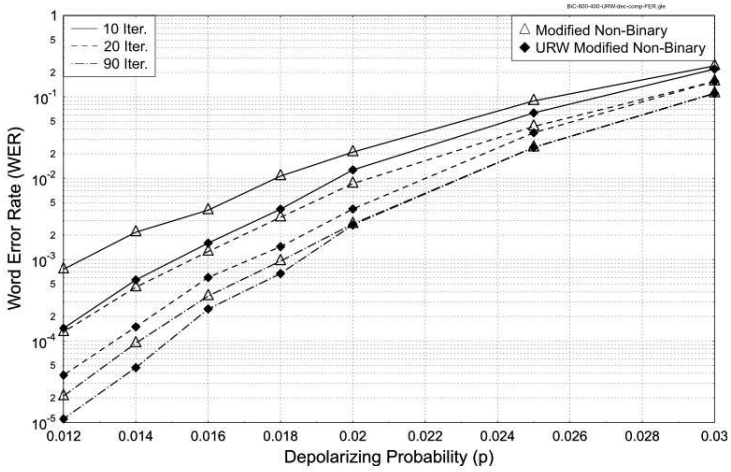

Figura 5.16: Prestazioni del WER ottenibili con l'algoritmo URW-BP, con i migliori valori di ρ elencati nella Tabella 5.10, rispetto al BP non binario modificato, se utilizzato da solo [28].

Tabella 5.11: Confronto delle prestazioni dell'URW-BP con il BP modificato per il sistema 1 della Tabella 5.5, basato sulle curve della Fig. 5.16. Le probabilità p elencate sono calcolate per un WER di 10^{-3} [28].

I_{max}	p del **BP Modificato**	p del **URW-BP**	**Incremento**
10	0.0125	0.0155	24%
20	0.015	0.017	13%
90	0.018	0.0185	3%

che il miglioramento del WER è inferiore per valori più elevati di p e per valori più elevati del numero massimo di iterazioni accessibili. Ad esempio, quando viene utilizzato un massimo di $I_{max} = 10$ iterazioni di decodifica con un WER di 10^{-3}, il metodo URW-BP aumenta $p = 0.0125$ fino a $p = 0.0155$, che corrisponde ad un incremento circa del $(\frac{(0.0155-0.0125)}{0.0125} \times 100) = 24\%$. Al contrario, per un massimo di $I_{max} = 20$ iterazioni, l'URW-BP aumenta $p = 0.015$ fino a $p = 0.017$, ad un WER di 10^{-3}. Ciò equivale ad un aumento del $(\frac{(0.017-0.015)}{0.015} \times 100) = 13\%$. Infine, ad un numero massimo ancora più elevato di iterazioni, cioè per $I_{max} = 90$, l'algoritmo URW-BP raggiunge un WER di 10^{-3} a $p = 0.0185$, che è "solo" un aumento pari al $(\frac{(0.0185-0.018)}{0.018} \times 100) = 3\%$, rispetto all'algoritmo non binario modificato.

Questi valori sono riassunti nella Tabella 5.11. Quindi, l'idea di "riponderare" le probabilità del messaggio è più utile a basse probabilità di depolarizzazione e per valori piccoli del numero massimo accessibile di iterazioni I_{max}. Questo perché a più alte probabilità di depolarizzazione (e valori similmente più grandi del numero massimo di iterazioni) i messaggi sono altamente correlati.

5.8 Linee guida per la progettazione

I codici QLDPC possono essere costruiti a partire dai codici binari e quaternari classici imponendo il rigoroso criterio del prodotto simplettico sulla PCM risultante, che assicura che gli stabilizer generator associati commutino tra loro. Le linee guida di progettazione per la costruzione dei codici QLDPC possono essere riassunte come segue:

• Un codice QLDPC con $[n, k]$ qubit, avente rate di codifica quantum di $R_Q = k/n$ può essere costruito a partire da un codice LDPC $(2n, n + k)$ binario classico, con un rate di codifica classico di $R_c = (n + k)/2n$, se la PCM **H** associata soddisfa il rigoroso criterio del prodotto simplettico.

• Idealmente, le righe della PCM **H** dovrebbero avere al massimo una singola sovrapposizione di 1 (o un valore diverso da zero nel formalismo GF(4)) per evitare cicli di lunghezza 4 nel grafo di Tanner associato, che degradano le prestazioni dell'algoritmo di decodifica iterativo. Purtroppo, il criterio del prodotto simplettico richiede un numero di sovrapposizioni tra le righe di **H**, questo risulta nella presenza inevitabile di cicli di lunghezza 4. Una grande sfida progettuale quindi è quella di costruire codici QLDPC validi nonostante la presenza dei cicli di lunghezza 4 inevitabili.

• Si possono sfruttare quattro principali strutture globali della PCM **H** per la progettazione di codici QLDPC, vale a dire: di tipo CSS dual-containing, di tipo CSS non dual-containing, non CSS e le soluzioni Entanglement-Assisted, tutte riportate nella Fig. 3.1.

Le sfide di progettazione associate a ciascuna di queste strutture sono riassunte di seguito:

◦ **Codici di tipo CSS Dual-containing (Sezione 4.1):** I codici a doppio ciclo di MacKay sono considerati tra i migliori, nella categoria dei codici CSS dual-containing, ma le loro prestazioni non sono ancora alla pari dei codici LDPC classici. Questo accade perché questo tipo di costruzione è estremamente caratterizzata dalla presenza di cicli brevi, che esistono sia nel dominio binario che quaternario.

◦ **Codici di tipo CSS non dual-containing (Sezione 4.1):** È difficile trovare una coppia di PCM binarie sparse che soddisfino il criterio del prodotto simplettico e che, allo stesso tempo, costituiscano codici QLDPC validi. *Finora*, solo i codici SC QC-QLDPC e i codici QC-QLDPC non binari sono noti per avere prestazioni vicine al limite di Hashing. Ma questo avviene al costo di una maggiore complessità oppure se si utilizzano una certa quantità di ebit silenziosi e pre-condivisi.

◦ **Codici di tipo non CSS (Sezione 4.2):** Idealmente, le costruzioni non CSS sono preferibili rispetto ai codici CSS perché sfruttano i qubit ridondanti in modo più efficace. Tuttavia, trovare codici good QLDPC di tipo non CSS rimane, *al momento*, una sfida aperta.

◦ **Codici EA (Sezione 4.3):** La tecnica Entanglement-Assisted può aiutare a raggiungere prestazioni paragonabili a quelle dei codici LDPC classici. Anche in questo caso, tuttavia, sono necessari gli ebit pre-condivisi, cioè una risorsa preziosa che richiede un notevole sovraccarico di trasmis-

sione. Di conseguenza, la ricerca è tesa a minimizzare il numero di ebit richiesti.

• Inoltre, è auspicabile che il codice QLDPC risultante abbia le seguenti caratteristiche:

◦ La PCM deve essere strutturata, ad esempio una struttura ciclica o quasi ciclica, per facilitarne l'implementazione;

◦ Distanza minima molto grande o almeno sufficientemente elevata per lunghezze dei blocchi elevate.

I codici QLDPC possono essere decodificati utilizzando l'algoritmo BP basato sulla sindrome, nel dominio binario o nel dominio non binario. Oltre all'ovvia minore complessità della decodifica binaria, le due principali differenze tra questi due domini di decodifica sono:

• In contrasto con la decodifica binaria, che presuppone che gli errori di bit-flip e di phase-flip siano indipendenti, la decodifica non binaria tiene conto della correlazione tra loro, quindi i loro effetti sono maggiori.

• Il numero di cicli di lunghezza 4 è più alto nel formalismo non binario, rispetto a quello binario. Questo tende a degradare le prestazioni del decodificatore non binario.

Quindi, ci sono due caratteristiche dei codici in conflitto. Tuttavia, l'algoritmo BP non binario ottiene prestazioni migliori rispetto al BP binario per entrambi i codici QLDPC studiati in questo elaborato [28].

Dal punto di vista della decodifica, le problematiche possono essere riassunte come segue:

• **Degenerazione:** I codici quantum sono intrinsecamente degeneri di natura. Questo può migliorare le prestazioni di decodifica associate se il decodificatore tiene conto di questa degenerazione. Sfortunatamente, l'algoritmo BP non sfrutta questa degenerazione. Infatti, poiché il BP si basa sulla probabilità marginale, la presenza di errori degeneri ne compromette le prestazioni.

• **Cicli brevi:** I cicli brevi di lunghezza 4 inevitabili presenti nei codici QLDPC degradano le prestazioni dell'algoritmo BP. Questo problema diventa ancora peggiore nel caso dei codici CSS omogenei, quando vengono decodificati nel dominio non binario.

I metodi euristici, vale a dire Random Perturbation e Enhanced Feedback, sono noti per diminuire, in una certa misura, entrambi questi problemi. Tuttavia, questa diminuzione si ottiene al costo di una maggiore complessità di decodifica. Pertanto, Babar *et al.* in [28] hanno concepito un algoritmo di decodifica non binario modificato per i codici QLDPC di tipo CSS omogenei, che allevia con successo il problema dei cicli di lunghezza 4 inevitabili. Il decodificatore modificato di [28] presenta prestazioni superiori, dal punto di vista del WER, nonostante la sua minore complessità di decodifica, rispetto alle tecniche di decodifica all'avanguardia. Inoltre, tale decodificatore può anche essere utilizzato in modo congiunto con i metodi euristici per ottenere ulteriori vantaggi in termini di prestazioni.

Infine, si è anche dimostrato che l'algoritmo URW-BP può essere sfruttato per contrastare il problema dei cicli brevi.

Capitolo 6

Simulatore di un sistema di comunicazione quantum

In questo capitolo viene presentato il progetto di un simulatore software che è stato utilizzato per simulare il passaggio dell'informazione nel canale quantum e la ricezione della stessa, utilizzando diversi modelli di codice per la correzione d'errore. Il codificatore, che si trova lato trasmettitore, non è stato introdotto nel simulatore; infatti, non viene generata nessuna parola di codice, e ciò equivale a trasmettere la parola "tutti zeri". Questo è comunque sufficiente per gli scopi in oggetto. È possibile attuare tale strategia per la linearità dei codici utilizzati.

Il simulatore svolge simulazioni Monte Carlo per stimare la probabilità d'errore del sistema. Per sua natura, la simulazione Monte Carlo calcola una serie di realizzazioni possibili del fenomeno in esame, con il peso proprio della probabilità di tale evenienza. Una volta calcolato questo campione casuale, la simulazione esegue delle "misure" delle grandezze di interesse su tale campione.

Nell'ambito delle simulazioni svolte, infatti, il vettore di errore introdotto dal canale di depolarizzazione viene generato casualmente ed ha peso che dipende dalla probabilità di depolarizzazione; in particolare maggiore è tale probabilità e maggiore sarà la probabilità di avere peso alto nel vettore di errore. Dopodiché, viene realizzata la decodifica, che si ritiene soddisfatta se il vettore di errore stimato dal decodificatore uguaglia quello effettivo,

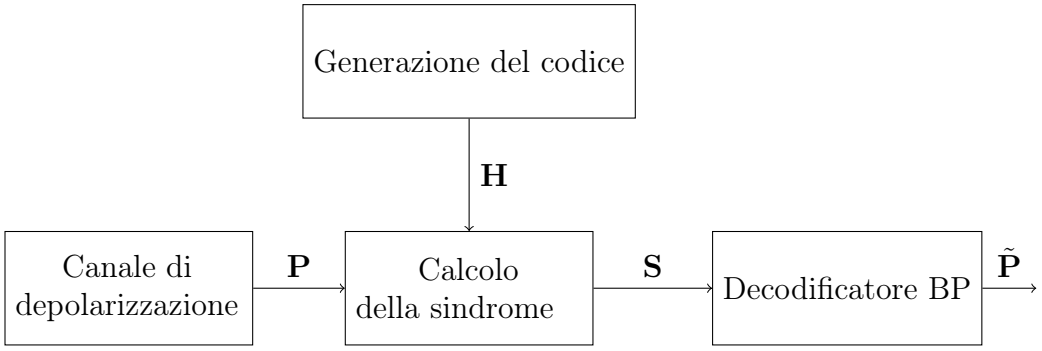

Figura 6.1: Schema a blocchi del simulatore software.

Algorithm 1 Syndrome-Based BP

Set $P_{\text{ch}}(0) \leftarrow (1 - 2p/3)$ and $P_{\text{ch}}(1) \leftarrow 2p/3$.
Initialize $m^a_{v_t \to c_i} \leftarrow P_{\text{ch}}(a)$, $\forall\, v_t, c_i \in C(v_t)$ and $a \in \{0, 1\}$.
for iter $\leftarrow$ 1 to $\mathtt{I}_{\max}$ **do**
  for all $i \in \{0, (m-1)\}$, $v_t \in V(c_i)$ and $a \in \{0, 1\}$ **do**
    $m^a_{c_i \to v_t} \leftarrow k \sum_{P:P_t=a} \mathrm{P}(S_i|P) \prod_{v_{t'} \in V(c_i) \setminus v_t} m^{P_{t'}}_{v_{t'} \to c_i}$.
  end for
  for $t \leftarrow 0$ to $(2n-1)$ **do**
    for all $c_i \in C(v_t)$ and $a \in \{0, 1\}$ **do**
      $m^a_{v_t \to c_i} \leftarrow k \mathrm{P}_{\text{ch}}(P_t = a) \prod_{c_{i'} \in C(v_t) \setminus c_i} m^a_{c_{i'} \to v_t}$.
    end for
    for all $a \in \{0, 1\}$ **do**
      $\mathrm{P}(P_t = a|S) \leftarrow k \mathrm{P}_{\text{ch}}(P_t = a) \prod_{c_i \in C(v_t)} m^a_{c_i \to v_t}$.
    end for
    $\tilde{P}_t \leftarrow \arg\max_{P_t \in \mathbb{F}_2} \mathrm{P}(P_t|S)$.
  end for
  $\tilde{S} \leftarrow H(\tilde{P}_x : \tilde{P}_z)^T$.
  if $(\tilde{S} = S)$ **then**
    return $\tilde{P}$.
  end if
end for

Figura 6.2: Pseudo-codice del decodificatore BP che è stato implementato via software [28].

precedentemente generato. Tale decodifica, ogni volta con un vettore di errore diverso, viene ripetuta fino a che non si raggiunge una soglia prefissata T di decodifiche errate; a questo punto di incrementa di un valore prefissato la probabilità di depolarizzazione e si ricomincia la procedura. La simulazione termina quando si raggiunge la soglia di decodifiche errate per la probabilità di depolarizzazione massima prefissata.

Come si può osservare dalla Fig. 6.1, il simulatore tiene conto di quattro blocchi: codificatore, canale di depolarizzazione, generazione del codice e decodificatore. Per l'implementazione software è stato usato il linguaggio di programmazione Python ed alcuni dei blocchi di codice che assolvono le funzioni di cui sopra sono presenti nell'Appendice B.

Il blocco di "Generazione del codice", che si può osservare nella Fig. 6.1, si occupa della generazione della matrice $\mathbf{H}$, che è diversa per ogni codice e serve per calcolare la sindrome (blocco "Calcolo della Sindrome").

Per quanto riguarda invece il canale quantum in oggetto, come ampiamente spiegato nella Sezione 5.1, è stato modellato come un canale di depolarizzazione. Si sono utilizzati quindi due BSC indipendenti, ognuno con probabilità di cross-over pari a $2p/3$, con p che è la probabilità di depolarizzazione. Per ulteriori chiarimenti sul canale di depolarizzazione, si faccia riferimento alla Sezione 1.2.

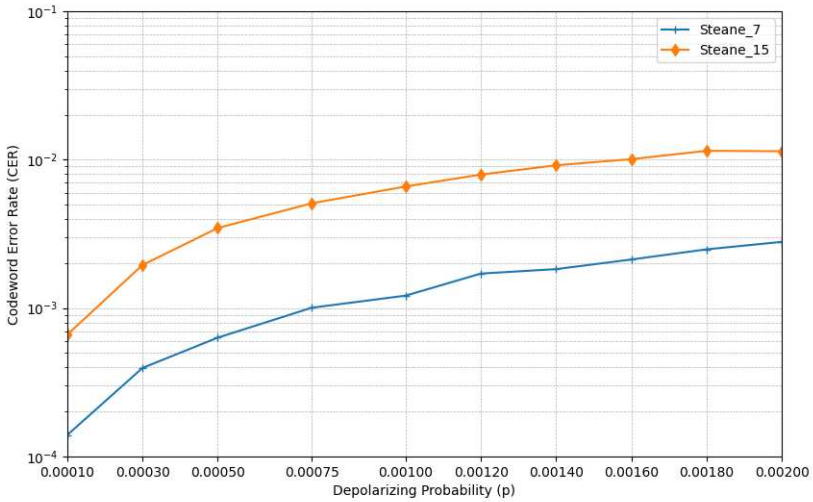

Figura 6.3: Prestazioni in termini di CER (Codeword Error Rate) raggiunte dal codice di Steane binario $[7, 1]$ (curva azzurra) e $[15, 7]$ (curva arancione), utilizzando il decodificatore BP-SPA$_{\mathrm{H}}$.

6.1 Decodificatore

Il decodificatore merita un'analisi più approfondita.

Seguendo quanto riportato prima nel Capitolo 3 e poi nel Capitolo 5, è stato implementato via software un decodificatore Belief Propagation, in particolare nella sua variante Sum-Product, seguendo proprio il modello matematico presentato nella Sezione 5.1, sintetizzato nello pseudo-codice riportato in Fig. 6.2. Tale decodificatore da qui in avanti verrà chiamato BP-SPA$_{\mathrm{H}}$.

Inoltre, è stato testato anche il decodificatore BP+OSD mediante la libreria Python rispettivamente di [86], [87]. Questo decodificatore verrà chiamato nel seguito BP+OSD$_l$.

Per tutti i decodificatori deve essere fissato, prima dell'inizio della simulazione, il numero massimo di iterazioni $\mathrm{I}_{\max}$ del decodificatore.

6.1.1 Codice di Steane

Il primo modello di codice quantum per la correzione degli errori considerato è il codice di Steane binario, che è un codice CSS dual-containing. In particolare, sono state indagate le prestazioni di questo codice quando $n = 7$, cioè si intende il codice di Steane $[7, 1]$, che è già stato approfonditamente analizzato nella Sezione 2.4.1.1 e nella Sezione 5.4, rispettivamente per il caso binario e quaternario. Inoltre, anche quando $n = 15$, cioè il codice di Steane $[15, 7]$, che discende dal codice di Hamming $(15, 11, 3)$.

Per la simulazione (vedere B.2 nell'Appendice B) si è utilizzato il decodificatore BP-SPA$_{\mathrm{H}}$ e si è posto $T = 300$ e $\mathrm{I}_{\max} = 90$. Come si può osservare dalla Fig. 6.3, il codice di Steane ha prestazioni migliori nel caso $[7, 1]$; infat-

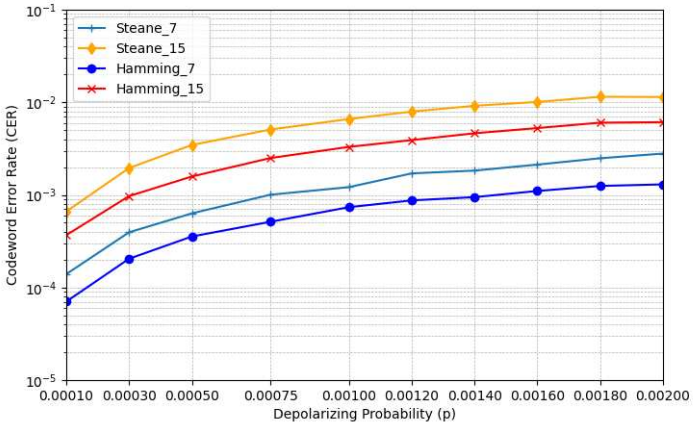

Figura 6.4: Prestazioni in termini di CER (Codeword Error Rate) raggiunte dai codici binari di Steane $[7, 1]$ (curva azzurra) e Steane $[15, 7]$ (curva arancione) a confronto con i rispettivi codici di Hamming $(7, 4)$ (curva blu) e $(15, 11)$ (curva rossa). Si è utilizzato il decodificatore BP-SPA$_{\mathrm{H}}$.

ti la curva blu, per ogni probabilità di depolarizzazione testata (p), si trova sotto la curva arancione: questo significa che la prima ha prestazioni migliori della seconda. Questo fenomeno può sembrare a prima vista sorprendente dato che il codice che codifica blocchi più grandi corregge effettivamente meno. In realtà, poiché Steane $[7, 1]$ codifica 1 qubit logico in 7 qubit fisici, mentre Steane $[15, 11]$ codifica 7 qubit di logici in 15 qubit fisici, il primo assicura più ridondanza per ogni bit rispetto al secondo. Inoltre, sapendo che entrambi discendono da un codice di Hamming, la distanza minima di questi due codici è la stessa e vale $d_{\min} = 3$. Quindi la capacità correttiva (garantita) è di 1 errore per entrambi i codici. Bisogna comunque osservare che, in questo caso, la distanza minima non è un parametro che permette di valutare *totalmente* la bontà dei codici presi in esame, visto che si sta adoperando un decodificatore probabilistico e non ottimo (MAP).

Per rendere ancora più evidente la differenza che c'è tra le prestazioni dei codici classici e dei corrispettivi quantum, si è riportato in Fig. 6.4 il confronto tra i codici di Hamming associati ai codici di Steane riportati in Fig. 6.3. Si può osservare facilmente che il codice di Hamming che costituisce il codice di Steane considerato si trova sempre "sotto" di esso; ciò significa che ha prestazioni migliori in termini di CER.

6.1.2 Codice a ciclo unico

Il secondo modello di codice analizzato è il codice quantum CSS a ciclo unico di Mackay [11] e spiegato nella Sezione 4.1.3.

In particolare, per la simulazione presente si è scelto il seguente insieme delle differenze perfetto $S_p = \{0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256\}$, di dimensione $n = 273$. A questo punto si costruisce una ma-

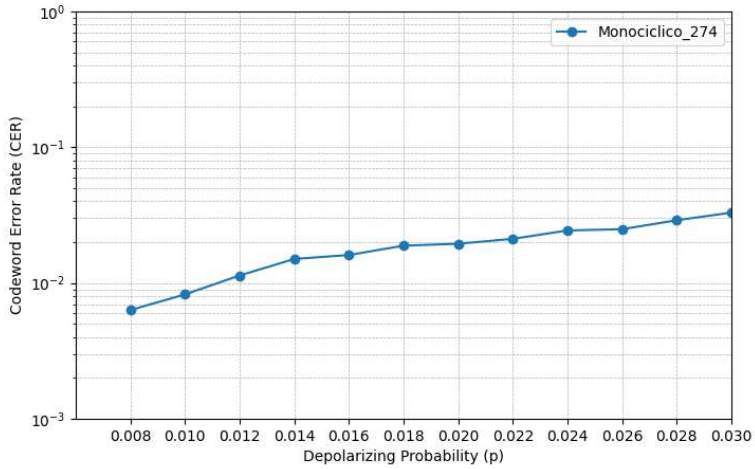

Figura 6.5: Prestazioni in termini di CER (Codeword Error Rate) raggiunte dal codice a ciclo unico [274, 194], utilizzando il decodificatore BP+OSD$_l$.

trice $\mathbf{C}$ ciclica sparsa, dove la prima riga ha gli 1s soltanto nelle posizioni specificate dai numeri presenti nel perfect difference set di cui sopra. Quindi, la matrice si deriva per righe operando lo spostamento unitario verso destra della riga precedente. Tale matrice ha dimensioni $n \times n$ e peso di riga $\rho = \#S_p$, dove $\#S_p$ indica la cardinalità dell'insieme S_p. A questo punto si ricava la matrice di parità $\mathbf{H}$ del codice posizionando a destra dell'ultima colonna di $\mathbf{C}$ una ulteriore colonna di tutti 1s. Quindi $\mathbf{H}$ ha dimensioni $n \times (n+1)$ e peso di riga pari a $\#S_p + 1$, in accordo con quanto spiegato nella Sezione 4.1.3. In questo modo si è definito un codice dual-containing $\mathcal{C}[n+1, k+1]$. Come si può facilmente notare la matrice $\mathbf{H}$ non ha rango pieno, cioè non ha righe tutte linearmente indipendenti, e questo spiega il motivo per cui le sue dimensioni non sono $(n-k) \times (n+1)$. In particolare, si può verificare che il numero m di righe linearmente indipendenti (vincoli di parità) di $\mathbf{H}$, nel caso in cui $n = 273$ è pari a 82, di conseguenza il numero di qubit di informazione è $k = n - m = 191$, quindi si ha un codice $[274, 194]$, descritto dalla $\mathbf{H}$ di dimensioni 273×274, con peso di riga $\rho = 17 + 1 = 18$. Sapendo che, per questi codici, la distanza minima è limitata superiormente dal peso di riga, come spiegato nella Sezione 4.1.3, si trova che $d = 18$.

Successivamente, prima di eseguire la decodifica, si costruisce un'altra matrice, che si chiama $\mathbf{H}'$, che è identica a $\mathbf{H}$, ma al posto dell'ultima colonna all-one prevede una colonna all-zero. A questo punto si utilizza un decodificatore BP, che nelle simulazioni presenti è BP+OSD$_l$, per entrambi i codici sottesi dalle matrici $\mathbf{H}$ e $\mathbf{H}'$, e si dichiara la decodifica riuscita se entrambi i decodificatori ritornano la stessa parola di codice. Per la simulazione sono state fissate $T = 100$ e $\mathrm{I}_{\max} = 100$; le prestazioni del codice possono essere osservate nella Fig. 6.5.

Approfondimento: BP con OSD Il decodificatore BP+OSD$_l$ fa riferimento al metodo di post-processing "Ordered Statistics Decoding" (OSD).

Anche se originariamente è stato concepito come metodo per ridurre il tasso di errore nei codici LDPC classici da Fossorier e Lin in [88], l'OSD è stato applicato per la prima volta nel dominio quantum da Panteleev e Kalachev [89] e si è dimostrato un decodificatore sorprendentemente efficace per i codici QLDPC random. La procedura seguente si applica ugualmente alla decodifica dei componenti $\mathbf{H}_x$ e $\mathbf{H}_z$ di un codice quantum CSS.

Poiché le matrici di parità spesso non hanno il rango di colonna pieno, ed è questo il caso anche del codice a ciclo unico di cui sopra, non è possibile risolvere l'equazione della sindrome $\mathbf{H}^{-1}\mathbf{s} = \mathbf{e}$ che richiederebbe l'inversione della matrice $\mathbf{H}$. Si osservi che, per semplicità di notazione, nella formula precedente si è fatto riferimento al caso classico; infatti il vettore di errore è stato indicato con $\mathbf{e}$. Tuttavia, per qualsiasi matrice di parità è possibile trovare un sottoinsieme delle sue colonne, specificate dall'indice $[S]$, che sono linearmente indipendenti. Queste colonne formano una base che può essere utilizzata per definire una sotto-matrice $\mathbf{H}_{[S]}$ con rango di colonna pieno, formata selezionando le colonne dall'originale matrice $\mathbf{H}$. Poiché questa sotto-matrice ha il rango di colonna pieno, può essere invertita per dare una soluzione all'equazione della sindrome $\mathbf{H}^{-1}\mathbf{s} = \mathbf{e}$.

Ogni scelta della base $\mathbf{H}_{[S]}$ corrisponde a una soluzione unica $\mathbf{e}_{[S]}$, eliminando ogni ambiguità potenziale dovuta alla degenerazione degli errori nel dominio quantum. È possibile selezionare $\mathbf{H}_{[S]}$ come insieme di casuale, partendo dalle colonne di $\mathbf{H}$, ma questo approccio è probabile che non si traduca in una buona soluzione (cioè di peso basso) per $\mathbf{e}_{[S]}$.

In particolare, per la simulazione del presente lavoro, è stato usato un metodo OSD chiamato "combination sweep strategy", cioè una variante del metodo originariamente proposto in [88]. Le fasi di questa procedura sono le seguenti:

• I bit nella componente $\mathbf{e}_{[T]}$ della soluzione OSD sono ordinati in base alle "decisioni" soft dell'algoritmo BP, dove con $[T]$ si indica l'insieme dei bit che non sono inclusi nell'insieme della base $[S]$, cioè $[T] \notin [S]$.

• Vengono cercate tutte le configurazioni di peso uno di $\mathbf{e}_{[T]}$.

• Vengono cercate tutte le configurazioni di peso due nei primi λ bit di $\mathbf{e}_{[T]}$.

Il numero totale di configurazioni considerate è uguale a $k' + \binom{\lambda}{2}$, dove $k' = n - \text{rank}(\mathbf{H})$ è la lunghezza del vettore $\mathbf{e}_{[T]}$.

Nel caso in esame del codice a ciclo unico, è stato impostato $\lambda = 1$ a causa delle dimensioni del codice. In sostanza, il terzo step dell'algoritmo sarebbe quasi ininfluente.

Conclusioni

I QECC sono essenziali per far avvicinare le tecnologie quantum ad un livello che sia il più possibile "fault-tolerant". Sfortunatamente però, la teoria della codifica classica non può essere applicata direttamente ai codici nel dominio quantum a causa delle intrinseche differenze tra i due mondi. In particolare, diversamente dal generico bit classico, il qubit non può essere copiato e collassa al valore di un bit dopo essere stato misurato. Inoltre, l'inversione di bit è l'unico tipo di errore che si può verificare durante la trasmissione dell'informazione su un canale classico, mentre un canale quantum può determinare sia errori di tipo bit-flip che errori di phase-flip. A causa di questa differenza, si modella il canale quantum come un canale di depolarizzazione, che è equivalente a una coppia di canali binari simmetrici; uno per gli errori di phase-flip ed uno per gli errori di bit-flip.

Pertanto, non è possibile mappare direttamente i codici classici sulle loro controparti quantum. Tuttavia, si osserva che i codici quantum possono essere progettati attraverso il formalismo stabilizzatore e poi trasferiti dal dominio quantum a quello classico (o viceversa) grazie alle tecniche di isomorfismo, che servono per mappare la rappresentazione del codice. Esistono varie possibili classi di codici quantum, ma in questo lavoro quella che si è approfondita maggiormente, a causa della sua semplicità, è la configurazione CSS, sia nel caso binario che in quello quaternario.

I codici classici LDPC sono stato dell'arte per molte applicazioni di codifica di canale classica, tra cui il 5G ed alcuni tipi di comunicazioni satellitari. Per quanto riguarda scopi di affidabilità nelle comunicazioni, questi vengono decodificati utilizzando l'algoritmo iterativo BP che ne assicura prestazioni eccellenti sui maggiori sui modelli di canale di comunicazione classici, quali l'AWGN e il BSC.

I codici QLDPC, che sono l'argomento principale del presente lavoro, pur avendo una storia molto più giovane rispetto a quella dei loro "cugini" classici, sono caratterizzati da varie possibili costruzioni. Ognuna di queste è stata particolarizzata dalle prestazioni presentate dai vari ricercatori presi come riferimento. Anche se queste sono ancora evidentemente lontane da quelle raggiunte nel mondo classico, i codici QLDPC sono tra i migliori nel dominio quantum e quindi si affermano come un valido candidato per raggiungere la fault-tolerance sul quale investire in termini di ricerca.

Come sopra per il caso classico, anche i codici QLDPC possono essere decodificati tramite l'algoritmo BP (SPA), osservando gli opportuni accorgimenti dovuti al cambio di dominio e facendo riferimento al canale di depola-

rizzazione, sia per il caso binario, che per quello quaternario. Le prestazioni in decodifica dei codici QLDPC, che ne determinano la bontà in termini di affidabilità della comunicazione, vengono presentate fissando il modello di codice utilizzato (per esempio il codice biciclico di Mackay con $n = 273$) e variando il decodificatore, oltre che il campo (caso binario e quaternario).

Si è implementato via software (utilizzando il linguaggio di programmazione Python) un simulatore per la comunicazione su un canale di depolarizzazione, prendendo il codice di Steane, lungamente approfondito in questo lavoro di Tesi e dei codici QLDPC. Il decodificatore scelto è il BP di cui sopra ed è stato realizzato mediante una funzione, che si può osservare sotto forma di codice, insieme alla parte più importante del simulatore.

Appendice A

Future direzioni della ricerca: Surface Code

È possibile costruire un qubit *logico* da un insieme di qubit *fisici*, in modo tale che il qubit logico performi molto meglio rispetto ai singoli qubit fisici.

Dunque, un possibile approccio alla costruzione di un calcolatore quantistico è quello di utilizzare i surface code [90], [91] utilizzati come stabilizer code [9], che permettono appunto di progettare dei qubit logici, a partire da un insieme di qubit fisici. I surface code sono un'evoluzione dei codici torici (toric codes) ideati da Alexei Kitaev in [42], [92], [93], [94] che si basano su un modello semplice, dal punto di vista topologico, che distribuisce i qubit a disposizione sulla superficie di un toro. La geometria toroidale utilizzata da Kitaev non si rivelò però utile lato correzione degli errori, così vennero sviluppate delle versioni planari (quindi *surface code*, letteralmente "codici di superficie") da Bravyi e Kitaev [90] prima, così come da Freedman e Meyer poi [95].

Uno dei vantaggi significativi dei surface code è la loro relativa tolleranza agli errori locali, come è stato descritto per la prima volta da Preskill e colleghi in [96].

Raussendorf e collaboratori hanno poi scoperto che l'operazione logica CNOT poteva essere implementata da trasformazioni braid (trasformazioni a "treccia") su un'unica superficie, cioè ottenendo una semplificazione molto significativa [97], [98], [99]. Questi autori hanno anche valutato le tolleranze di errore per un'implementazione completamente planare utilizzando solo porte quantum ad un qubit e a due qubit adiacenti, arrivando ad una soglia di errore dello 0,75% per operazione.

La tolleranza all'errore dei surface code, insieme ad un semplice layout fisico bidimensionale con solo l'accoppiamento dei qubit adiacenti, rende l'architettura dei surface code uno degli approcci più realistici per costruire un computer quantistico a stato solido.

Il prezzo pagato per l'alta tolleranza all'errore è che le implementazioni del surface code coinvolgono un gran numero di qubit fisici [100]. Ci vogliono un minimo di tredici qubit fisici per implementare un singolo qubit logico. Effettivamente però, un qubit logico, ragionevolmente tollerante all'errore,

che può essere utilizzato efficacemente in un surface code, richiede dai 10^3 ai 10^4 qubit fisici.

Il numero di qubit fisici necessari per definire un qubit logico dipende fortemente dal rate di errore nei qubit fisici. I rate di errore appena sotto la soglia richiedono un numero maggiore di qubit fisici per qubit logico, mentre i rate di errore sostanzialmente più piccoli della soglia consentono un numero minore di qubit fisici. Supponendo un rate di errore pari a circa un decimo del rate di soglia, si ha bisogno di circa 14 500 qubit fisici per ogni qubit logico per dar luogo ad un rate di errore logico sufficientemente basso per eseguire correttamente gli algoritmi quantum.

A.1 Surface Code

Si implementa il surface code su un array bidimensionale (matrice) di qubit fisici, come mostrato nella Fig. A.1. I qubit presenti sono sia *data qubit*, rappresentati da cerchi aperti nella Fig. A.1a, in cui sono memorizzati gli stati quantum computazionali, cioè i *qubit di misura*, rappresentati da cerchi pieni. Tutti i data qubit e i qubit di misura devono soddisfare i requisiti di base per il calcolo: inizializzazione, rotazioni a singolo qubit e porte a due qubit controlled-NOT (CNOT) tra qubit adiacenti. Inoltre, per eseguire una versione topologica della trasformazione di Hadamard, i data qubit e i qubit di misura devono essere in grado di scambiare i loro stati quantum (operazione SWAP). È inoltre richiesto un metodo per misurare l'operatore **Z** per ogni qubit.

Approfondimento: Gate Controlled-NOT Questa porta logica ha due qubit di ingresso, che sono il qubit di controllo (control qubit) ed il qubit di destinazione (target qubit), rispettivamente. In uscita sono presenti ancora due qubit; in particolare lo stato del qubit di controllo rimane sempre inalterato, mentre quello del qubit di destinazione potrebbe variare. La rappresentazione circuitale è riportata in Fig. A.2, dove la linea più in alto indica il qubit di controllo $|A\rangle$, mentre l'altra il qubit di destinazione $|B\rangle$. Il funzionamento è descritto nel seguito.

Se il qubit di controllo è impostato su 0, il qubit di destinazione rimane inalterato. Se il qubit di controllo, invece, è impostato su 1, lo stato del qubit di destinazione viene rovesciato. Analiticamente, si ha:

$$\mathrm{CNOT}(|\psi_0\rangle, |\psi_1\rangle) = |\psi_0\rangle \otimes |\psi_0 \oplus \psi_1\rangle\,, \tag{A.1}$$

dove $|\psi_0\rangle$ è il qubit di controllo, $|\psi_1\rangle$ è il qubit di destinazione, mentre $\otimes$ indica il prodotto tensoriale e $\oplus$ la somma modulo 2. Dunque si hanno 4 diversi possibili casi:

$$\begin{aligned} &(i)\,|00\rangle \to |00\rangle\,; \qquad (ii)\,|01\rangle \to |01\rangle\,; \\ &(iii)\,|10\rangle \to |11\rangle\,; \qquad (iv)\,|11\rangle \to |10\rangle\,. \end{aligned}$$

In altri termini, l'uscita può essere vista come un'operazione reversibile di una porta Exclusive-OR classica; quindi, la porta CNOT può essere

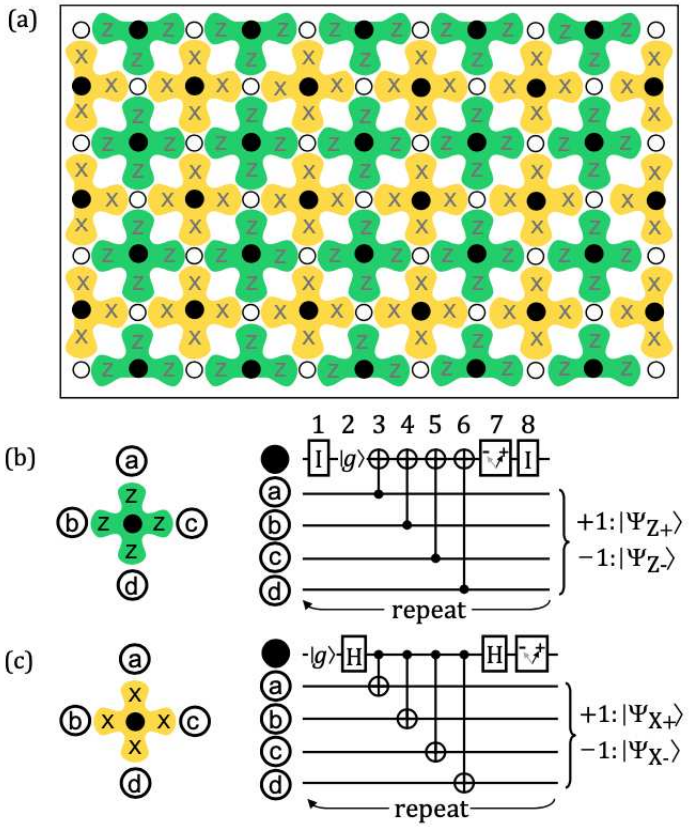

Figura A.1: (a) Implementazione bidimensionale del surface code [101]. I data qubit sono cerchi aperti, mentre i qubit di misura sono cerchi pieni. I qubit di misura tipo Z sono colorati di verde e i qubit di misura tipo X colorati di giallo. Lontano dai bordi, ogni data qubit entra in contatto con quattro qubit di misura, invece ogni qubit di misura tocca quattro data qubit. Il qubit di misura deve eseguire infatti misurazioni a quattro terminali. Sui bordi, i qubit di misura contattano solo tre data qubit ed eseguono misurazioni a tre terminali, mentre i data qubit entrano in contatto con due (nel caso dei vertici della matrice) o tre qubit di misura. La linea solida che circonda l'array ne indica il contorno. (b) Sequenza geometrica delle operazioni (a sinistra) e circuito quantum (a destra) per un ciclo di un surface code per un qubit di misura tipo Z, che stabilizza $g_Z = \mathbf{Z}_a\mathbf{Z}_b\mathbf{Z}_c\mathbf{Z}_d$. (c) Geometria e circuito quantum per un qubit di misura tipo X, che stabilizza $g_X = \mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$. I due operatori di identità $\mathbf{I}$, per il processo di misura tipo Z, assicurano che la tempistica del qubit di misura tipo X corrisponda a quella del qubit di misura tipo Z. Nella figura (c), il primo qubit è sottoposto a due operazioni di Hadamard $\mathbf{H}_H$. Gli operatori di identità sono posizionati all'inizio e alla fine della sequenza, riducendo l'impatto di eventuali errori durante questi passaggi [101].

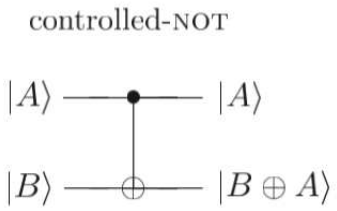

Figura A.2: Gate CNOT a due qubit [1].

considerata la controparte quantum della porta XOR. Si tratta di una generalizzazione della porta XOR classica, poiché l'azione della porta CNOT può essere riassunta come $|A, B\rangle \rightarrow |A, A \oplus B\rangle$, che è esattamente ciò che fa la porta XOR. Diversamente dalla sua controparte classica, in cui i due ingressi sono combinati per produrre una singola uscita XOR irreversibile, il funzionamento di una porta CNOT è di fatto reversibile, poiché si possono ricostruire i due ingressi (controllo e destinazione) a partire dalle due corrispondenti uscite [1].

I qubit di misura sono utilizzati per stabilizzare e manipolare lo stato quantum dei data qubit. Ci sono due tipi di qubit di misura, i qubit di misura tipo Z ("measure-Z" qubit), colorati di verde nella Fig. A.1a, e i qubit di misura tipo X ("measure-X" qubit), colorati di giallo; questi sono chiamati nella letteratura dei surface code *sindrome* Z e *sindrome* X, rispettivamente. Ogni data qubit è accoppiato con due qubit di misura tipo Z e con due qubit di misura tipo X e ogni qubit di misura è accoppiato, a sua volta, con quattro data qubit. Un qubit di misura tipo Z è usato per forzare i suoi data qubit adiacenti a, b, c e d in un autostato del prodotto tra gli operatori $g_Z = \mathbf{Z}_a\mathbf{Z}_b\mathbf{Z}_c\mathbf{Z}_d$; ogni qubit di misura tipo Z quindi misura uno stabilizzatore. Un qubit di misura tipo X forza, allo stesso modo, i suoi data qubit adiacenti in un autostato dato da $g_X = \mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$, quindi, anche in questo caso, misura uno stabilizzatore.

A.2 Stato di quiescenza del surface code

I qubit di misura tipo Z e tipo X, che stabilizzano il surface code, sono gestiti in una sequenza molto particolare, con un ciclo completo mostrato in Fig. A.1b e c, per una singola misura tipo Z e tipo X, rispettivamente. Dopo aver inizializzato ogni qubit di misura nel suo stato fondamentale $|g\rangle$, il cuore della computazione comprende una sequenza di quattro operazioni CNOT, la cui formulazione analitica è presente nell'eq.(A.1), seguite da una misurazione proiettiva. Per il qubit di misura tipo Z, i CNOT utilizzano il qubit di misura come target qubit e i quattro data qubit adiacenti come qubit di controllo, con la misurazione proiettiva che produce un autostato dello stabilizzatore $g_Z = \mathbf{Z}_a\mathbf{Z}_b\mathbf{Z}_c\mathbf{Z}_d$ (gli autostati sono elencati nella Tabella A.1). Per il qubit di misura tipo X, i quattro CNOT utilizzano i data qubit adiacenti come qubit di destinazione e il qubit di misura come qubit

Tabella A.1: Autostati per gli stabilizzatori a quattro qubit $g_Z = \mathbf{Z}_a\mathbf{Z}_b\mathbf{Z}_c\mathbf{Z}_d$ e $g_X = \mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$.

Autovalori	g_Z	g_X
$+1$	$\|gggg\rangle$	$\|++++\rangle$
	$\|ggee\rangle$	$\|++--\rangle$
	$\|geeg\rangle$	$\|+--+\rangle$
	$\|eegg\rangle$	$\|--++\rangle$
	$\|egge\rangle$	$\|-++-\rangle$
	$\|gege\rangle$	$\|+-+-\rangle$
	$\|egeg\rangle$	$\|-+-+\rangle$
	$\|eeee\rangle$	$\|----\rangle$
-1	$\|ggge\rangle$	$\|+++-\rangle$
	$\|ggeg\rangle$	$\|++-+\rangle$
	$\|gegg\rangle$	$\|+-++\rangle$
	$\|eggg\rangle$	$\|-+++\rangle$
	$\|geee\rangle$	$\|+---\rangle$
	$\|egee\rangle$	$\|-+--\rangle$
	$\|eege\rangle$	$\|--+-\rangle$
	$\|eeeg\rangle$	$\|---+\rangle$

controllo, la sequenza inoltre include anche un gate di Hadamard applicato al qubit di misura prima e dopo le porte CNOT; la misurazione proiettiva produce quindi un autostato dello stabilizzatore $g_X = \mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$. Quindi, dopo la misurazione proiettiva di tutti i qubit di misura nell'array, lo stato $|\psi\rangle$ di tutti i data qubit soddisfa in modo simultaneo $g_Z|\psi\rangle = Z_{abcd}|\psi\rangle$, con autovalori $Z_{abcd} = \pm 1$, e $g_X|\psi\rangle = X_{abcd}|\psi\rangle$ con autovalori $Z_{abcd} = \pm 1$. Dopo la misurazione, il ciclo viene ripetuto.

Approfondimento: Ciclo del surface code e circuiti stabilizzatori Ora si vogliono analizzare con maggiore dettaglio i circuiti presenti nelle Fig. A.1b e c. Per semplicità, si studia un sistema con solo due data qubit a e b, stabilizzato da un qubit di misura tipo X e tipo Z. La generalizzazione a quattro qubit è semplice. Il layout semplificato e i corrispondenti circuiti stabilizzatori sono mostrati nella Fig. A.3 e sono costituiti da due porte CNOT invece che da quattro CNOT, come nel circuito completo.

L'obiettivo è che questo circuito stabilizzi i due data qubit a e b in un autostato simultaneo di $g_{\hat{X}} = \mathbf{X}_a\mathbf{X}_b$ e $g_{\hat{Z}} = \mathbf{Z}_a\mathbf{Z}_b$, cioè precisamente gli stati di Bell elencati nella Tabella A.2. Dunque, utilizzando uno stato arbitrario entangled dei due data qubit come input per il circuito in Fig. A.3, si verifica quanto sopra. Si noti che il circuito rende entangled tra loro tutti i qubit durante le operazioni CNOT; pertanto si possono scrivere gli stati quantum nella forma $|\psi_X\psi_a\psi_b\psi_Z\rangle$, cioè il primo elemento è lo stato del qubit di misura tipo X, il secondo e il terzo quelli dei data qubit a e b, rispettivamente, e il quarto elemento quello del qubit di misura tipo Z. In

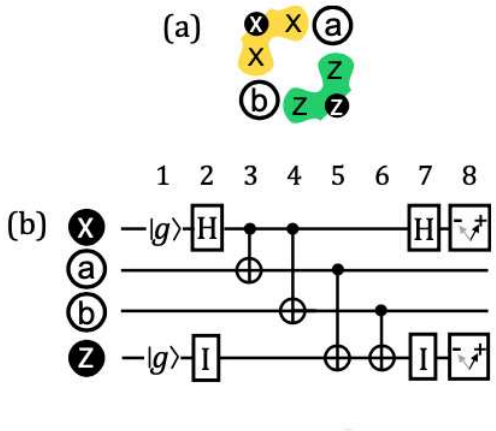

Figura A.3: (a) Due data qubit a e b sono stabilizzati da un qubit di misura tipo Z e da un qubit di misura tipo X, collegati come mostrato. (b) Circuito quantum per i due qubit di misura che operano sui due data qubita. L'ordine CNOT è critico; in primo luogo, il qubit di misura tipo X agisce come control qubit della prima porta CNOT sul data qubit a e della seconda porta CNOT sul data qubit b. I due CNOT sono preceduti e seguiti da una porta di Hadamard $\mathbf{H}_H$. Il qubit di misura tipo Z, invece, è il target qubit della terza porta CNOT, dove il data qubit a ne è il qubit di controllo. La quarta porta CNOT prevede come qubit di controllo il data qubit b e come target qubit, sempre il qubit di misura tipo Z. I due operatori di identità $\mathbf{I}$, per il processo di misura tipo Z, sincronizzano le operazioni del qubit di misura tipo Z e del qubit di misura tipo X. Gli operatori di misura corrispondenti agli step dal 3 al 6 della sequenza di controllo, seguiti dalla misurazione proiettiva alla fine del circuito, sono indicati sotto le porte CNOT pertinenti, con il qubit di misura tipo X che stabilizza il prodotto $g_{\hat{X}} = \mathbf{X}_a\mathbf{X}_b$ e il qubit di misura tipo Z che stabilizza $g_{\hat{Z}} = \mathbf{Z}_a\mathbf{Z}_b$ [101].

Tabella A.2: Autostati degli operatori a due qubit $g_{\hat{Z}} = \mathbf{Z}_a\mathbf{Z}_b$ e $g_{\hat{X}} = \mathbf{X}_a\mathbf{X}_b$. I quattro autostati sono gli stati di Bell per questo sistema.

$g_{\hat{Z}}$	$g_{\hat{X}}$	
$+1$	$+1$	$\frac{\lvert gg\rangle+\lvert ee\rangle}{\sqrt{2}}$
$+1$	-1	$\frac{\lvert gg\rangle-\lvert ee\rangle}{\sqrt{2}}$
-1	$+1$	$\frac{\lvert ge\rangle+\lvert eg\rangle}{\sqrt{2}}$
-1	-1	$\frac{\lvert ge\rangle-\lvert eg\rangle}{\sqrt{2}}$

particolare, questo ordine rende i CNOT più facili da calcolare, il qubit di misura tipo X controlla i data qubit a e b, mentre i data qubit controllano il qubit di misura tipo Z.

Lo stato dopo l'esecuzione del passo N-esimo del circuito è quindi $|\psi_N\rangle$.

Ora vengono elencati gli step da effettuare, numerandoli.

Step 1: I qubit di misura tipo X e tipo Z vengono reimpostati al loro stato fondamentale. I data qubit possono essere entangled in uno stato generale a due qubit, usando la notazione $|\psi_a\psi_b\rangle$ dove ψ_a rappresenta lo stato del qubit a e ψ_b quello del qubit b. Si può quindi scrivere lo stato del data qubit risultante nella seguente forma:

$$|\psi_{ab}\rangle = A\,|gg\rangle + B\,|ge\rangle + C\,|eg\rangle + D\,|ee\rangle\,, \tag{A.2}$$

dove i coefficienti A, B, C e D sono complessi. Lo stato del circuito completo è ora:

$$\begin{aligned}|\psi_1\rangle &= |g\rangle \otimes (A\,|gg\rangle + B\,|ge\rangle + C\,|eg\rangle + D\,|ee\rangle) \otimes |g\rangle \\ &= A\,|gggg\rangle + B\,|ggeg\rangle + C\,|gegg\rangle + D\,|geeg\rangle\,.\end{aligned}$$

Step 2: L'operatore di identità $\mathbf{I}$ lascia invariato lo stato del qubit di misura tipo Z. Il qubit di misura tipo X, invece, subisce una trasformazione di Hadamard, per cui vale $|g\rangle \to |+\rangle = |g\rangle + |e\rangle$ e $|e\rangle \to |-\rangle = |g\rangle - |e\rangle$ (ignorando per comodità la normalizzazione a $1/\sqrt{2}$). Per maggiore chiarezza, si faccia riferimento alla Sezione 1.3.3. Lo stato del circuito completo è ora:

$$\begin{aligned}|\psi_2\rangle = {} & A\,|gggg\rangle + A\,|eggg\rangle + B\,|ggeg\rangle + B\,|egeg\rangle \\ & + C\,|gegg\rangle + C\,|eegg\rangle + D\,|geeg\rangle + D\,|eeeg\rangle\,.\end{aligned}$$

Step 3: La prima porta CNOT trasforma gli stati di un qubit di controllo e di un target qubit $|ct\rangle$ secondo la legge $|gg\rangle \to |gg\rangle$, $|ge\rangle \to |ge\rangle$, $|eg\rangle \to |ee\rangle$ e $|ee\rangle \to |eg\rangle$. Applicando il CNOT tra il qubit di misura tipo X (qubit di controllo) e il data qubit a (target qubit), cioè dal 1° al 2° elemento, si ottiene:

$$\begin{aligned}|\psi_3\rangle = {} & A\,|gggg\rangle + A\,|eegg\rangle + B\,|ggeg\rangle + B\,|eeeg\rangle \\ & + C\,|gegg\rangle + C\,|eggg\rangle + D\,|geeg\rangle + D\,|egeg\rangle\,.\end{aligned}$$

Step 4: Il secondo CNOT usa come qubit di controllo il qubit di misura tipo X e come qubit di destinazione il data qubit b, si lavora cioè con il 1° e con il 3° elemento. Si ottiene:

$$\begin{aligned}|\psi_4\rangle = {} & A\,|gggg\rangle + A\,|eeeg\rangle + B\,|ggeg\rangle + B\,|eegg\rangle \\ & + C\,|gegg\rangle + C\,|egeg\rangle + D\,|geeg\rangle + D\,|eggg\rangle\,.\end{aligned}$$

Step 5: Il terzo CNOT usa come qubit di controllo il data qubit a e come qubit di destinazione il qubit di misura tipo Z, si lavora cioè con il 2°

e con il 4° elemento. Si ottiene:

$$|\psi_5\rangle = A\,|gggg\rangle + A\,|eeee\rangle + B\,|ggeg\rangle + B\,|eege\rangle + C\,|gege\rangle + C\,|egeg\rangle + D\,|geee\rangle + D\,|eggg\rangle\,.$$

Step 6: Il quarto ed ultimo CNOT usa come qubit di controllo il data qubit b e come qubit di destinazione il qubit di misura tipo Z, si lavora cioè con il 3° e con il 4° elemento. Si ottiene:

$$|\psi_6\rangle = A\,|gggg\rangle + A\,|eeeg\rangle + B\,|ggee\rangle + B\,|eege\rangle + C\,|gege\rangle + C\,|egee\rangle + D\,|geeg\rangle + D\,|eggg\rangle\,.$$

Step 7: Il qubit di misura tipo X viene sottoposta alla seconda trasformazione di Hadamard, per cui si ha:

$$\begin{aligned}|\psi_7\rangle &= A\,|{+}ggg\rangle + A\,|{-}eeg\rangle + B\,|{+}gee\rangle + B\,|{-}ege\rangle \\ &\quad + C\,|{+}ege\rangle + C\,|{-}gee\rangle + D\,|{+}eeg\rangle + D\,|{-}ggg\rangle \\ &= (A+D)\,|g\rangle \otimes (|gg\rangle + |ee\rangle) \otimes |g\rangle + (A-D)\,|e\rangle \otimes (|gg\rangle - |ee\rangle) \otimes |g\rangle \\ &= (B+C)\,|g\rangle \otimes (|ge\rangle + |eg\rangle) \otimes |e\rangle + (B-C)\,|e\rangle \otimes (|ge\rangle - |eg\rangle) \otimes |e\rangle\,.\end{aligned} \tag{A.3}$$

Step 8: Le misurazioni $\hat{Z}$ finali vengono eseguite sia sul qubit di misura tipo Z che sul qubit di misura di misura tipo X. Ogni qubit ha due possibili risultati: $+1$ per lo stato $|g\rangle$ e -1 per lo stato $|e\rangle$. I quattro possibili risultati e la corrispondente proiezione sullo stato finale dei data qubit sono:

$$\begin{aligned}\{M_X,\, M_Z\} = &\{+1,\,+1\}; \quad |\psi_{ab}\rangle = |gg\rangle + |ee\rangle\,, \\ &\{-1,\,+1\}; \qquad\quad |gg\rangle - |ee\rangle\,, \\ &\{+1,\,-1\}; \qquad\quad |ge\rangle + |eg\rangle\,, \\ &\{-1,\,-1\}; \qquad\quad |ge\rangle - |eg\rangle\,.\end{aligned} \tag{A.4}$$

Le probabilità di proiettare il risultato della misura su ciascuno di questi stati è dato dal modulo al quadrato di ogni corrispondente ampiezza di probabilità della (A.3).

Si noti che i risultati della misurazione e gli stati dei due dati qubit corrispondenti sono precisamente quelli che appaiono nella Tabella A.2 per gli stati di Bell. Quindi, dato uno stato di input arbitrario, alla fine del circuito, si proiettano i due dati qubit su uno dei quattro autostati di $g_{\hat{Z}} = \mathbf{Z}_a\mathbf{Z}_b$ e $g_{\hat{X}} = \mathbf{X}_a\mathbf{X}_b$.

Se si prende uno degli stati di uscita della (A.4) e lo si utilizza come input per il circuito stabilizzatore, si può facilmente verificare che l'output riproduce con precisione l'input, con gli stessi risultati di misurazione.

Per esempio, se si considera il risultato della misurazione $M_X,\, M_Z = -1,\, -1$, i data qubit finiscono nello stato $|ge\rangle - |eg\rangle$. Questo è lo stato di ingresso della (A.2) con i coefficienti $A = D = 0$ e $B = -C = 1$. Guardando poi la (A.3), si osserva che l'ampiezza delle seguenti somme algebriche tra i coefficienti $A+D$, $A-D$ e $B+C$ saranno tutte pari a zero, il

che implica che lo stato di uscita verrà sempre proiettato allo stato di input $|ge\rangle - |eg\rangle$. Quindi, come previsto, il circuito stabilizzatore restituisce gli stessi risultati di misurazione e lo stesso stato dei data qubit.

Approfondimento: Notazione Una lettera maiuscola in grassetto, ad esempio $\mathbf{X}$, designa un operatore di Pauli (o matrice di Pauli). Uno stabilizzatore come $g_X = \mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$, invece, è il prodotto tensore di quattro operatori $\mathbf{X}_j$, quindi sarebbe rappresentato da una matrice $2^4 \times 2^4 = 16 \times 16$; il suo risultato X_{abcd} dopo la misura è un autovalore di questa matrice. Si noti che misurare il prodotto $\mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d$ non produce lo stesso risultato della misurazione di ogni singolo operatore $\mathbf{X}_a$, $\mathbf{X}_b$, $\mathbf{X}_c$ e $\mathbf{X}_d$, in quanto i qubit non sono in generale in un prodotto di autostati dei singoli operatori $\mathbf{X}_j$.

I qubit di misura rappresentati in Fig. A.1b e c funzionano tutti in lockstep, cioè ogni passo del ciclo mostrato nella figura deve essere completato sull'intero array bidimensionale prima dell'inizio dello step successivo.

I codici stabilizzatori hanno la notevole proprietà di non funzionare nello stato fondamentale del sistema, ma nello stato $|\psi\rangle$, che è il risultato della misurazione simultanea di tutti gli stabilizzatori. Per questo esso viene chiamato *stato quiescente* o *stato di quiescenza*. Lo stato di quiescenza $|\psi\rangle$ viene selezionato in modo casuale completando un intero ciclo del surface code, che è la sequenza mostrata in Fig. A.1b e c, iniziando ad esempio con tutti i data qubit e i qubit di misurazione nel loro stato fondamentale $|g\rangle$.

Una volta selezionato, in assenza di errori, lo stesso stato quiescente $|\psi\rangle$ sarà mantenuto da ogni ciclo successivo della sequenza, con ogni qubit di misura che dà un risultato di misurazione X_{abcd} o Z_{abcd} uguale a quello del ciclo precedente. Questo accade perché tutti gli stabilizzatori g_X e g_Z commutano tra loro. Tale risultato è triviale per gli stabilizzatori che non hanno alcun qubit in comune, infatti gli operatori $\mathbf{X}$ e $\mathbf{Z}$ che agiscono su qubit diversi commutano sempre. Gli stabilizzatori che hanno qubit in comune, invece, condivideranno sempre almeno due di questi qubit, quindi ogni stabilizzatore g_X condivide due data qubit con ogni stabilizzatore g_Z adiacente e viceversa.

Quindi, per uno stabilizzatore g_X e g_Z che misura i data qubit a e b in comune, si ottiene:

$$\begin{aligned}[\mathbf{X}_a\mathbf{X}_b\mathbf{X}_c\mathbf{X}_d \ , \ & \mathbf{Z}_a\mathbf{Z}_b\mathbf{Z}_e\mathbf{Z}_f] \\ &= (\mathbf{X}_a\mathbf{Z}_a)(\mathbf{X}_b\mathbf{Z}_b)\mathbf{X}_c\mathbf{X}_d\mathbf{Z}_e\mathbf{Z}_f - (\mathbf{Z}_a\mathbf{X}_a)(\mathbf{Z}_b\mathbf{X}_b)\mathbf{X}_c\mathbf{X}_d\mathbf{Z}_e\mathbf{Z}_f \\ &= \mathbf{0},\end{aligned}$$

in questo modo si ottiene un segno meno commutando $\mathbf{X}_a$ con $\mathbf{Z}_a$, così come se ne ottiene un altro commutando $\mathbf{X}_b$ con $\mathbf{Z}_b$.

Esiste un numero enorme di stati di quiescenza che possono essere selezionati dalle misure dello stabilizzatore: se ci sono N qubit di misura nell'array, ci saranno 2^N risultati di misura. Le misurazioni alla fine di ogni ciclo di surface code proiettano casualmente i data qubit su uno di questi

stati quiescenti. Per l'array in Fig. A.1, con $N = 38$ qubit di misura, ci sono $2^{38} \approx 3 \times 10^{11}$ possibili stati di quiescenza.

Appendice B

Implementazione software del simulatore

```python
# Syndrome-Based Belief Propagation (BP) algorithm.
def syndrome_based_BP(p, I_max, C, V, H, syndrome):
    ########## Parameters:
    # p (float): error probability.
    # I_max (int): maximum number of iterations.
    # C (dict): dictionary which associates each variable node vt its set of check nodes ci.
    # V (dict): dictionary which associates each check node ci its set of variable nodes vt.
    # H (np.array): PCM.
    # syndrome (np.array): syndrome vector.
    ########### Returns:
    # P_t_given_S_estimated: estimated errro vector.

    # Do marginalization
    def remove_value_from_list(V_i, v_t):
        # Returns a new list that contains all the elements of V_i except v_t
        return [value for value in V_i if value != v_t]

    ###################################################
    ############## BP SP ALGORITHM ####################
    ###################################################
    # Step 1: Initialization of channel probabilities.
    P_ch = {0: (1 - 2*(p/3)), 1: (2*(p/3))}

    # Step 2: Initialization of messages m_vt_to_ci
    m_vt_to_ci = {}
    # Cycle that iters on variable nodes.
    for vt in range(n):
        # Cycle that iters on check nodes.
        for ci in C[vt]:
            # m_vt_to_ci is a nested dictionary.
            m_vt_to_ci[(vt, ci)] = {0: P_ch[0], 1: P_ch[1]}

    # Messages exchange
    for iter in range(1, I_max + 1):
        # Step 3: Updating of messages ma_ci_to_vt: horizontal message exchange
        m_ci_to_vt = {}
        # Cycle that iters on check nodes.
        for i in range(m):
            # Cycle that iters on variable nodes linked to check node ci.
            for vt in V[i]:
                # Compute non-normalized messages for a = 0 and a = 1.
                unnormalized_messages = {}
                for a in [0, 1]:
                    summation = 0
                    valid_combinations = []
                    # XOR between the i-th syndrome value and the current value of a.
                    xor = syndrome[i] ^ a
                    # Assign to valid_combinations the right value.
                    if xor == 0:
                        valid_combinations = valid_combiantions_0
                    else:
                        valid_combinations = valid_combiantions_1
                    num_comb = len(valid_combinations)
                    # Cycle on combinations.
                    for index_comb1 in range(num_comb):
                        prod = 1
                        V_prime = remove_value_from_list(V[i], vt)
                        # Cycle that iters on binary elements of the current combination.
                        for index_comb2, vt_prime in zip(range(k), V_prime):
                            prod *= m_vt_to_ci[(vt_prime, i)][valid_combinations[index_comb1][
                                index_comb2]]
                        # Update summation.
                        summation += prod
                    # Save non-normalized message.
                    unnormalized_messages[a] = summation
```

```
                # Non-normalized summation.
                sum_unnormalized = unnormalized_messages[0] + unnormalized_messages[1]
                # Compute normalization factor K.
                K = 1 / sum_unnormalized if sum_unnormalized != 0 else 1  # Evita divisioni per 0.
                # Normalization.
                for a in [0, 1]:
                    if (i, vt) not in m_ci_to_vt:
                        m_ci_to_vt[(i, vt)] = {}
                    m_ci_to_vt[(i, vt)][a] = unnormalized_messages[a] * K

        # Step 4: Update of messages ma_vt_to_ci.
        # Cycle that iters on variable nodes.
        for t in range(n):
            # Cycle that iters on check nodes linked to variable node vt.
            for ci in C[t]:
            # Compute non-normalized messages for a = 0 and a = 1.
                unnormalized_messages = {}
                for a in [0, 1]:
                    product = P_ch[a]
                    for ci_prime in C[t]:
                        # Marginalization.
                        if ci_prime != ci:
                            product *= m_ci_to_vt[(ci_prime, t)][a]
                    unnormalized_messages[a] = product
                # Non-normalized summation.
                sum_unnormalized = unnormalized_messages[0] + unnormalized_messages[1]
                # Compute normalization factor K.
                K = 1 / sum_unnormalized if sum_unnormalized != 0 else 1  # Avoid ratios by 0.
                # Normalization.
                for a in [0, 1]:
                    m_vt_to_ci[(t, ci)][a] = unnormalized_messages[a] * K

        # Step 5: Element-wise marginal probability.
        P_t_given_S = {}
        # Cycle that iters on variable nodes.
        for t in range(n):
        # Compute non-normalized probabilities.
            unnormalized_probs = {}
            for a in [0, 1]:
                product = P_ch[a]
                for ci in C[t]:
                    product *= m_ci_to_vt[(ci, t)][a]
                unnormalized_probs[a] = product
             # Summation of non-normalized probabilities.
            sum_unnormalized = unnormalized_probs[0] + unnormalized_probs[1]
            # Compute normalization factor K.
            K = 1 / sum_unnormalized if sum_unnormalized != 0 else 1  # Avoid ratios by 0.
            # Normalization.
            for a in [0, 1]:
                P_t_given_S[(t, a)] = unnormalized_probs[a] * K

        # Step 6: Estimation of the t-th element of the error vector P.
        P_t_given_S_estimated = np.zeros(n, dtype=int)
        for t in range(n):
            P_t_given_S_estimated[t] = max([0, 1], key=lambda a: P_t_given_S.get((t, a), float('-inf
                '')))

        # Step 7: Compute estimated syndrome S_tilde.
        P_t_given_S_estimatedT = P_t_given_S_estimated.reshape(-1, 1)
        S_tilde = H @ P_t_given_S_estimatedT %2

        # Step 8: syndrome checking.
        if np.array_equal(S_tilde, syndrome):
            return P_t_given_S_estimated

    # Returns the final estimated error vector P if the condition is not met but the iterations are
        finished.
    return P_t_given_S_estimated
```

Listing B.1: Algoritmo BP nella variante SP, implementato sulla base dello pseudo-codice di [28].

```
# Simulator:
for p in np.arange(p_min, p_max + p_step, p_step):
    frame_errati = 0
    frame_corretti = 0
    frame_trasmessi = 1
    # Compute p_cross.
    p_cross = (2/3)*p
    # Maximum number of wrong frames accepted.
    th = 100

    # Classe del Decoder BP
    bpd_H = bposd_decoder(
        H, # PCM
        p_cross = p_cross, # Error rate
        max_iter =  100, # Maximum number of iterations
        bp_method = "product_sum", # Type of BP
        osd_order = 1,
        osd_method = "osd_cs" # combination-sweep method
        )
    # Classe del Decoder BP.
    bpd_H_prime = bposd_decoder(
```

```
        H_prime, # PCM
        p_cross = p_cross, # Error rate
        max_iter = 100, # Maximum number of iterations
        bp_method = "product_sum", # Type of BP
        osd_order = 1,
        osd_method = "osd_cs" # combination-sweep method
        )

    while (frame_errati < th):
        # Compute error vector for H and H_prime.
        P_errH = channel_error((N+1), p_cross)
        P_errH_T = P_errH.reshape(-1, 1)

        # Compute syndrome.
        syndrome_H = H @ P_errH_T %2
        syndrome_H_prime = H_prime @ P_errH_T %2
        syndrome_H = syndrome_H.astype(int)
        syndrome_H_prime = syndrome_H_prime.astype(int)

        # Syndrome-based BP decoding:
        # the estimated error is found and saved in the decoded variable.
        decoded_H = syndrome_based_BP(p, I_max, C_dic, V_dic, H, syndrome_H)
        decoded_H_prime = syndrome_based_BP(p, I_max, C_dic_H_prime, V_dic_H_prime, H_prime,
             syndrome_H_prime)

        if np.array_equal(decoded_H, decoded_H_prime):
            frame_corretti += 1
        else:
            frame_errati += 1

        # Compute CER.
        CER = frame_errati/frame_trasmessi

        frame_trasmessi += 1

    # Fill the axis.
    x_axis.append(p); y_axis.append(CER)
    print(f"p=",p,",CER=",CER)
    j += 1
```

Listing B.2: Simulatore utilizzato per ricavare i rusltati del Capitolo 6.

Bibliografia

[1] Michael Nielsen e Isaac Chuang. «Quantum Computation and Quantum Information (Cambridge University Press, Cambridge, England)». In: (2000).

[2] Zunaira Babar et al. «Duality of Quantum and Classical Error Correction Codes: Design Principles and Examples». In: *IEEE Communications Surveys Tutorials* 21.1 (2019), pp. 970–1010. DOI: 10.1109/COMST.2018.2861361.

[3] Zunaira Babar, Soon Xin Ng e Lajos Hanzo. «Near-capacity code design for entanglement-assisted classical communication over quantum depolarizing channels». In: *IEEE Transactions on Communications* 61.12 (2013), pp. 4801–4807.

[4] Zunaira Babar, Soon Xin Ng e Lajos Hanzo. «EXIT-chart aided code design for symbol-based entanglement-assisted classical communication over quantum channels». In: *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*. IEEE. 2014, pp. 1–5.

[5] Charles Henry Bennett e Stephen Wiesner. «Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states». In: *Physical Review Letters* 69.20 (1992), p. 2881.

[6] William Wootters e Wojciech Zurek. «A single quantum cannot be cloned». In: *Nature* 299.5886 (1982), pp. 802–803.

[7] Zunaira Babar et al. «The road from classical to quantum codes: A hashing bound approaching design procedure». In: *IEEE Access* 3 (2015), pp. 146–176.

[8] Daniel Gottesman. «Class of quantum error-correcting codes saturating the quantum Hamming bound». In: *Physical Review A* 54 (3 1996), pp. 1862–1868. DOI: 10.1103/PhysRevA.54.1862. URL: https://link.aps.org/doi/10.1103/PhysRevA.54.1862.

[9] Daniel Gottesman. *Stabilizer codes and quantum error correction*. California Institute of Technology, 1997.

[10] Hans Kurzweil e Bernd Stellmacher. *The Theory of Finite Groups: An Introduction*. Vol. 1. Springer, 2004.

[11] David John MacKay, Graeme Mitchison e Paul McFadden. «Sparse-graph codes for quantum error correction». In: *IEEE Transactions on Information Theory* 50.10 (2004), pp. 2315–2330.

[12] Robert Calderbank et al. «Quantum error correction via codes over GF(4)». In: *IEEE Transactions on Information Theory* 44.4 (1998), pp. 1369–1387. DOI: 10.1109/18.681315.

[13] Andrew Martin Steane. «Multiple-particle interference and quantum error correction». In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (1996), pp. 2551–2577.

[14] Robert Calderbank e Peter Willstone Shor. «Good quantum error-correcting codes exist». In: *Physical Review A* 54 (2 1996), pp. 1098–1105. DOI: 10.1103/PhysRevA.54.1098. URL: https://link.aps.org/doi/10.1103/PhysRevA.54.1098.

[15] Andrew M Steane. «Error correcting codes in quantum theory». In: *Physical Review Letters* 77.5 (1996), p. 793.

[16] Garry Bowen. «Entanglement required in achieving entanglement-assisted channel capacities». In: *Physical Review A* 66.5 (2002), p. 052313.

[17] Todd Brun, Igor Devetak e Min-Hsiu Hsieh. «Correcting quantum errors with entanglement». In: *science* 314.5798 (2006), pp. 436–439.

[18] R. Gallager. «Low-density parity-check codes». In: *IRE Transactions on Information Theory* 8.1 (1962), pp. 21–28. DOI: 10.1109/TIT.1962.105768.

[19] Robert G Gallager. «Low-Density Parity Check Codes. Cambridge». In: *Massachusttes* (1963).

[20] R. Tanner. «A recursive approach to low complexity codes». In: *IEEE Transactions on Information Theory* 27.5 (1981), pp. 533–547. DOI: 10.1109/TIT.1981.105640.

[21] M. Sipser e D.A. Spielman. «Expander codes». In: *IEEE Transactions on Information Theory* 42.6 (1996), pp. 1710–1722. DOI: 10.1109/18.556667.

[22] Niclas Wiberg. «Codes and decoding on general graphs». In: (1996).

[23] William Ryan e Shu Lin. *Channel Codes: Classical and Modern*. Cambridge University Press, 2009.

[24] Y. Kou, S. Lin e M.P.C. Fossorier. «Low-density parity-check codes based on finite geometries: a rediscovery and new results». In: *IEEE Transactions on Information Theory* 47.7 (2001), pp. 2711–2736. DOI: 10.1109/18.959255.

[25] R. Lucas et al. «Iterative decoding of one-step majority logic deductible codes based on belief propagation». In: *IEEE Transactions on Communications* 48.6 (2000), pp. 931–937. DOI: 10.1109/26.848552.

[26] David J.C. MacKay e Radford M. Neal. «Near Shannon Limit Performance of Low Density Parity Check Codes». In: *Electronics Letters* 32 (1996), pp. 1645–1646.

[27] D.J.C. MacKay. «Good error-correcting codes based on very sparse matrices». In: *IEEE Transactions on Information Theory* 45.2 (1999), pp. 399–431. DOI: 10.1109/18.748992.

[28] Zunaira Babar et al. «Fifteen years of quantum LDPC coding and improved decoding strategies». In: *IEEE Access* 3 (2015), pp. 2492–2519.

[29] Michael Postol. «A proposed quantum low density parity check code». In: *arXiv preprint quant-ph/0108131* (2001).

[30] John Proakis et al. *Communication systems engineering*. Vol. 2. Prentice Hall New Jersey, 1994.

[31] Salah Aly. «A Class of Quantum LDPC Codes Constructed From Finite Geometries». In: (2008), pp. 1–5. DOI: 10.1109/GLOCOM.2008.ECP.217.

[32] Y. Kou, S. Lin e M.P.C. Fossorier. «Low-density parity-check codes based on finite geometries: a rediscovery and new results». In: *IEEE Transactions on Information Theory* 47.7 (2001), pp. 2711–2736. DOI: 10.1109/18.959255.

[33] Henry Berthold Mann et al. «Analysis and design of experiments.» In: *Analysis and design of experiments.* (1949).

[34] Anne Penfold Street e Deborah J Street. «Combinatorics of Experimental Design (Oxford science publications)». In: (1987).

[35] RD Carmichael. «Introduction to the theory of groups of finite˜». In: *Order, Dover, New York* (1956).

[36] Ivan B. Djordjevic. «Quantum LDPC Codes from Balanced Incomplete Block Designs». In: *IEEE Communications Letters* 12.5 (2008), pp. 389–391. DOI: 10.1109/LCOMM.2008.080083.

[37] B. Vasic e O. Milenkovic. «Combinatorial constructions of low-density parity-check codes for iterative decoding». In: *IEEE Transactions on Information Theory* 50.6 (2004), pp. 1156–1176. DOI: 10.1109/TIT.2004.828066.

[38] B. Ammar et al. «Construction of low-density parity-check codes based on balanced incomplete block designs». In: *IEEE Transactions on Information Theory* 50.6 (2004), pp. 1257–1269. DOI: 10.1109/TIT.2004.828144.

[39] David JC MacKay et al. «More sparse-graph codes for quantum error-correction». In: *http://www. inference. phy. cam. ac. uk/mackay/QECC. html* (2007).

[40] Alain Couvreur, Nicolas Delfosse e Gilles Zémor. «A Construction of Quantum LDPC Codes From Cayley Graphs». In: *IEEE Transactions on Information Theory* 59.9 (2013), pp. 6087–6098. DOI: 10.1109/TIT.2013.2261116.

[41] Alain Couvreur, Nicolas Delfosse e Gilles Zémor. «A Construction of Quantum LDPC Codes From Cayley Graphs». In: *IEEE Transactions on Information Theory* 59.9 (2013), pp. 6087–6098. DOI: 10.1109/TIT.2013.2261116.

[42] A Yu Kitaev. «Fault-tolerant quantum computation by anyons». In: *Annals of Physics* 303.1 (2003), pp. 2–30.

[43] Hector Bombin e Miguel A Martin-Delgado. «Homological error correction: Classical and quantum codes». In: *Journal of mathematical physics* 48.5 (2007), p. 052105.

[44] Hector Bombin e Miguel Angel Martin-Delgado. «Topological quantum distillation». In: *Physical review letters* 97.18 (2006), p. 180501.

[45] H. Lou e J. Garcia-Frias. «Quantum error-correction using codes with low-density generator matrix». In: (2005), pp. 1043–1047. DOI: 10.1109/SPAWC.2005.1506298.

[46] Hanqing Lou e Javier Garcia-Frias. «On the Application of Error-Correcting Codes with Low-Density Generator Matrix over Different Quantum Channels». In: (2006), pp. 1–6.

[47] S. ten Brink. «Code doping for triggering iterative decoding convergence». In: (2001), pp. 235–. DOI: 10.1109/ISIT.2001.936098.

[48] Manabu Hagiwara e Hideki Imai. «Quantum Quasi-Cyclic LDPC Codes». In: (2007), pp. 806–810. DOI: 10.1109/ISIT.2007.4557323.

[49] Manabu Hagiwara et al. «Spatially coupled quasi-cyclic quantum LDPC codes». In: (2011), pp. 638–642.

[50] Shrinivas Kudekar, Thomas J. Richardson e Rüdiger L. Urbanke. «Threshold Saturation via Spatial Coupling: Why Convolutional LDPC Ensembles Perform So Well over the BEC». In: *IEEE Transactions on Information Theory* 57.2 (2011), pp. 803–834. DOI: 10.1109/TIT.2010.2095072.

[51] Tom Richardson e Ruediger Urbanke. *Modern coding theory*. Cambridge university press, 2008.

[52] Kenta Kasai et al. «Non-binary quasi-cyclic quantum LDPC codes». In: (2011), pp. 653–657. DOI: 10.1109/ISIT.2011.6034212.

[53] Kenta Kasai et al. «Quantum Error Correction Beyond the Bounded Distance Decoding Limit». In: *IEEE Transactions on Information Theory* 58.2 (2012), pp. 1223–1230. DOI: 10.1109/TIT.2011.2167593.

[54] Iryna Andriyanova, Denise Maurice e Jean-Pierre Tillich. «Spatially coupled quantum LDPC codes». In: (2012), pp. 327–331. DOI: 10.1109/ITW.2012.6404686.

[55] Denise Maurice, Jean-Pierre Tillich e Iryna Andriyanova. «A family of quantum codes with performances close to the hashing bound under iterative decoding». In: (2013), pp. 907–911. DOI: 10.1109/ISIT.2013.6620358.

[56] Raymond Laflamme et al. «Perfect quantum error correcting code». In: *Physical Review Letters* 77.1 (1996), p. 198.

[57] Raymond Laflamme et al. «Perfect quantum error correcting code». In: *Physical Review Letters* 77.1 (1996), p. 198.

[58] Thomas Camara, Harold Ollivier e Jean-Pierre Tillich. «A class of quantum LDPC codes: construction and performances under iterative decoding». In: (2007), pp. 811–815. DOI: 10.1109/ISIT.2007.4557324.

[59] Yuan Li e Xingxiang Liu. «Family of fast quantum stabilizer codes constructions». In: *Optical review* 17.2 (2010), pp. 47–49.

[60] T.J. Richardson, M.A. Shokrollahi e R.L. Urbanke. «Design of capacity-approaching irregular low-density parity-check codes». In: *IEEE Transactions on Information Theory* 47.2 (2001), pp. 619–637. DOI: 10.1109/18.910578.

[61] Todd Brun, Igor Devetak e Min-Hsiu Hsieh. «General entanglement-assisted quantum error-correcting codes». In: (2007), pp. 2101–2105. DOI: 10.1109/ISIT.2007.4557160.

[62] Min-Hsiu Hsieh, Todd A Brun e Igor Devetak. «Entanglement-assisted quantum quasicyclic low-density parity-check codes». In: *Physical Review A* 79.3 (2009), p. 032340.

[63] Min-Hsiu Hsieh, Wen-Tai Yen e Li-Yi Hsu. *Performance of entanglementassisted quantum LDPC codes constructed from finite geometries*. 2009.

[64] Min-Hsiu Hsieh, Wen-Tai Yen e Li-Yi Hsu. «High Performance Entanglement-Assisted Quantum LDPC Codes Need Little Entanglement». In: *IEEE Transactions on Information Theory* 57.3 (2011), pp. 1761–1769. DOI: 10.1109/TIT.2011.2104590.

[65] Yuichiro Fujiwara et al. «Entanglement-assisted quantum low-density parity-check codes». In: *Physical Review A* 82.4 (2010), p. 042338.

[66] Yuichiro Fujiwara e Vladimir D. Tonchev. «A Characterization of Entanglement-Assisted Quantum Low-Density Parity-Check Codes». In: *IEEE Transactions on Information Theory* 59.6 (2013), pp. 3347–3353. DOI: 10.1109/TIT.2013.2247461.

[67] Ivan B Djordjevic. «Photonic entanglement-assisted quantum low-density parity-check encoders and decoders». In: *Optics letters* 35.9 (2010), pp. 1464–1466.

[68] Yuichiro Fujiwara. «Quantum error correction via less noisy qubits». In: *Physical Review Letters* 110.17 (2013), p. 170501.

[69] Yuichiro Fujiwara, Alexander Gruner e Peter Vandendriessche. «High-Rate Quantum Low-Density Parity-Check Codes Assisted by Reliable Qubits». In: *IEEE Transactions on Information Theory* 61.4 (2015), pp. 1860–1878. DOI: 10.1109/TIT.2015.2398436.

[70] David JC MacKay. *Bayesian inference and sampling theory, information theory, inference, and learning algorithms*. 2003.

[71] E. Berlekamp, R. McEliece e H. van Tilborg. «On the inherent intractability of certain coding problems (Corresp.)» In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386. DOI: 10.1109/TIT.1978.1055873.

[72] T.J. Richardson e R.L. Urbanke. «The capacity of low-density parity-check codes under message-passing decoding». In: *IEEE Transactions on Information Theory* 47.2 (2001), pp. 599–618. DOI: 10.1109/18.910577.

[73] D.J.C. MacKay. «Good error-correcting codes based on very sparse matrices». In: *IEEE Transactions on Information Theory* 45.2 (1999), pp. 399–431. DOI: 10.1109/18.748992.

[74] Sae-Young Chung et al. «On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit». In: *IEEE Communications Letters* 5.2 (2001), pp. 58–60. DOI: 10.1109/4234.905935.

[75] David Poulin e Yeojin Chung. «On the iterative decoding of sparse quantum codes». In: *arXiv preprint arXiv:0801.1241* (2008).

[76] Daniel A Lidar e Todd A Brun. *Quantum error correction*. Cambridge university press, 2013.

[77] Yun-Jiang Wang et al. «Enhanced Feedback Iterative Decoding of Sparse Quantum Codes». In: *IEEE Transactions on Information Theory* 58.2 (2012), pp. 1231–1241. DOI: 10.1109/TIT.2011.2169534.

[78] M.J. Wainwright, T.S. Jaakkola e A.S. Willsky. «A new class of upper bounds on the log partition function». In: *IEEE Transactions on Information Theory* 51.7 (2005), pp. 2313–2335. DOI: 10.1109/TIT.2005.850091.

[79] Tanya G. Roosta, Martin J. Wainwright e Shankar S. Sastry. «Convergence Analysis of Reweighted Sum-Product Algorithms». In: *IEEE Transactions on Signal Processing* 56.9 (2008), pp. 4293–4305. DOI: 10.1109/TSP.2008.924136.

[80] Henk Wymeersch, Federico Penna e Vladimir Savić. «Uniformly reweighted belief propagation: A factor graph approach». In: *2011 IEEE International Symposium on Information Theory Proceedings*. 2011, pp. 2000–2004. DOI: 10.1109/ISIT.2011.6033905.

[81] Henk Wymeersch, Federico Penna e Vladimir Savic. «Uniformly Reweighted Belief Propagation for Estimation and Detection in Wireless Networks». In: *IEEE Transactions on Wireless Communications* 11.4 (2012), pp. 1587–1595. DOI: 10.1109/TWC.2012.021412.111509.

[82] Jingjing Liu e Rodrigo C. de Lamare. «Knowledge-aided reweighted belief propagation decoding for regular and irregular LDPC codes with short blocks». In: (2012), pp. 984–988. DOI: 10.1109/ISWCS.2012.6328515.

[83] Jingjing Liu e Rodrigo C. de Lamare. «Low-Latency Reweighted Belief Propagation Decoding for LDPC Codes». In: *IEEE Communications Letters* 16.10 (2012), pp. 1660–1663. DOI: 10.1109/LCOMM.2012.080312.121307.

[84] Jingjing Liu, Rodrigo C. de Lamare e Henk Wymeersch. «Locally-optimized reweighted belief propagation for decoding finite-length LDPC codes». In: (2013), pp. 4311–4316. DOI: 10.1109/WCNC.2013.6555271.

[85] Henk Wymeersch et al. «Comparison of reweighted message passing algorithms for LDPC decoding». In: *2013 IEEE International Conference on Communications (ICC)*. 2013, pp. 3264–3269. DOI: 10.1109/ICC.2013.6655048.

[86] Joschka Roffe. *LDPC: Python tools for low density parity check codes*. Ver. 0.1.0. 4 Gen. 2022. URL: https://pypi.org/project/ldpc/.

[87] Joschka Roffe et al. «Decoding across the quantum low-density parity-check code landscape». In: *Physical Review Research* 2.4 (dic. 2020). ISSN: 2643-1564. DOI: 10.1103/physrevresearch.2.043423. URL: http://dx.doi.org/10.1103/PhysRevResearch.2.043423.

[88] M.P.C. Fossorier e Shu Lin. «Soft-decision decoding of linear block codes based on ordered statistics». In: *IEEE Transactions on Information Theory* 41.5 (1995), pp. 1379–1396. DOI: 10.1109/18.412683.

[89] Pavel Panteleev e Gleb Kalachev. «Degenerate Quantum LDPC Codes With Good Finite Length Performance». In: *Quantum* 5 (nov. 2021), p. 585. ISSN: 2521-327X. DOI: 10.22331/q-2021-11-22-585. URL: http://dx.doi.org/10.22331/q-2021-11-22-585.

[90] Sergey B Bravyi e A Yu Kitaev. «Quantum codes on a lattice with boundary». In: *arXiv preprint quant-ph/9811052* (1998).

[91] Eric Dennis et al. «Topological quantum memory». In: *Journal of Mathematical Physics* 43.9 (2002), pp. 4452–4505.

[92] A Yu Kitaev. «Quantum error correction with imperfect gates». In: *Quantum communication, computing, and measurement* (1997), pp. 181–188.

[93] A Yu Kitaev. «Quantum computations: algorithms and error correction». In: *Russian Mathematical Surveys* 52.6 (1997), p. 1191.

[94] A Yu Kitaev. «Fault-tolerant quantum computation by anyons. e-print». In: *arXiv preprint quant-ph/9707021* (1997).

[95] Michael H Freedman e David A Meyer. «Projective plane and planar quantum codes». In: *Foundations of Computational Mathematics* 1 (2001), pp. 325–332.

[96] Chenyang Wang, Jim Harrington e John Preskill. «Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory». In: *Annals of Physics* 303.1 (2003), pp. 31–58.

[97] Robert Raussendorf, Jim Harrington e Kovid Goyal. «A fault-tolerant one-way quantum computer». In: *Annals of physics* 321.9 (2006), pp. 2242–2270.

[98] Robert Raussendorf e Jim Harrington. «Fault-tolerant quantum computation with high threshold in two dimensions». In: *Physical review letters* 98.19 (2007), p. 190504.

[99] Robert Raussendorf, Jim Harrington e Kovid Goyal. «Topological fault-tolerance in cluster state quantum computation». In: *New Journal of Physics* 9.6 (2007), p. 199.

[100] Andrew W Cross, David P DiVincenzo e Barbara M Terhal. «A comparative code study for quantum fault-tolerance». In: *arXiv preprint arXiv:0711.1556* (2007).

[101] Austin G Fowler et al. «Surface codes: Towards practical large-scale quantum computation». In: *Physical Review A* 86.3 (2012), p. 032324.

[102] Manabu Hagiwara e Hideki Imai. «Quantum Quasi-Cyclic LDPC Codes». In: (2007), pp. 806–810. DOI: 10.1109/ISIT.2007.4557323.

[103] Robert Gray Gallager. «Low-density parity-check codes». In: *IRE Transactions on Information Theory* 8.1 (1962), pp. 21–28. DOI: 10.1109/TIT.1962.1057683.

[104] Austin G Fowler, Ashley M Stephens e Peter Groszkowski. «High-threshold universal quantum computation on the surface code». In: *Physical Review A* 80.5 (2009), p. 052312.

[105] David S Wang, Austin G Fowler e Lloyd CL Hollenberg. «Surface code quantum computing with error rates over 1%». In: *Physical Review A* 83.2 (2011), p. 020302.

[106] Jack Edmonds. «Paths, trees, and flowers». In: *Canadian Journal of mathematics* 17 (1965), pp. 449–467.

[107] Jack Edmonds. «Maximum matching and a polyhedron with 0, 1-vertices». In: *Journal of research of the National Bureau of Standards B* 69.125-130 (1965), pp. 55–56.

[108] Austin G Fowler e Simon J Devitt. «A bridge to lower overhead quantum computation». In: *arXiv preprint arXiv:1209.0510* (2012).

Elenco delle figure

Elenco delle tabelle