Constant Time Decoding of Quantum Expander Codes and Application to Fault-Tolerant Quantum Computation

Antoine Grospellier

Sorbonne Université

École doctorale Informatique, Télécommunications et Électronique (Paris)

INRIA Paris, équipe SECRET

Décodage des Codes Expanseurs Quantiques et Application au Calcul Quantique Tolérant aux Fautes

Par Antoine Grospellier
Thèse de doctorat d'informatique

Dirigée par Omar Fawzi et Anthony Leverrier

Présentée et soutenue publiquement le 8 Novembre 2019

devant un jury composé de :

Pr. Earl CAMPBELL	Sheffield University	Examinateur
Pr. Eleni DIAMANTI	Sorbonne Université	Examinatrice
Pr. Omar FAWZI	ENS Lyon	Directeur
Pr. Daniel GOTTESMAN	Perimeter Institute	Rapporteur
Pr. Anthony LEVERRIER	INRIA Paris	Directeur
Pr. Jean-Pierre TILLICH	INRIA Paris	Examinateur

rapportée par

Pr. Daniel GOTTESMAN	Perimeter Institute
Pr. Gilles ZÉMOR	Université de Bordeaux

Sorbonne Université

École doctorale Informatique, Télécommunications et Électronique (Paris)

INRIA Paris, team SECRET

Constant Time Decoding of Quantum Expander Codes and Application to Fault-Tolerant Quantum Computation

PhD thesis in computer science
by Antoine Grospellier

Advisors: Omar Fawzi and Anthony Leverrier

Prof. Earl CAMPBELL	Sheffield University	Examiner
Prof. Eleni DIAMANTI	Sorbonne Université	Examiner
Prof. Omar FAWZI	ENS Lyon	Advisor
Prof. Daniel GOTTESMAN	Perimeter Institute	Reviewer
Prof. Anthony LEVERRIER	INRIA Paris	Advisor
Prof. Jean-Pierre TILLICH	INRIA Paris	Examiner
Prof. Gilles ZÉMOR	Université de Bordeaux	Reviewer

Résumé

Le calcul quantique tolérant aux fautes est un ensemble de techniques dont le but est d'effectuer des calculs quantiques de manière fiable en utilisant des composants bruités. Dans ce contexte, l'utilisation de codes correcteurs quantiques maintient le nombre d'erreurs présentes dans le système en dessous d'un seuil tolérable. L'un des principaux problèmes de ce domaine est d'évaluer le coût minimum (en mémoire et en temps) nécessaire pour transformer un calcul quantique idéal en un calcul tolérant aux fautes. Dans cette thèse, nous montrons que la famille des codes expanseurs quantiques associée à l'algorithme de décodage small-set-flip peut être utilisée dans la construction de ref. [46] pour réaliser du calcul quantique tolérant aux fautes avec coût constant en mémoire.

La famille de codes correcteurs ainsi que le décodeur que nous étudions ont été introduits dans ref. [67] où un modèle de bruit adverse est considéré. En nous appuyant sur les résultats de cet article, nous analysons le comportement des codes expanseurs quantiques face à un modèle de bruit stochastique qui est pertinent dans le cadre du calcul tolérant aux fautes [38], [37]. De plus, nous montrons que l'algorithme de décodage peut être parallélisé pour fonctionner en temps constant. Cette propriété est essentielle pour éviter que les erreurs ne s'accumulent pendant que l'algorithme est exécuté.

Au-delà des résultats théoriques décrits ci-dessus, nous avons effectué une analyse numérique des codes expanseurs quantiques dans le but d'évaluer leurs performances en pratique [49]. Le modèle de bruit choisi pour ces simulations consiste à générer des erreurs de types X et Z de manière indépendante et identiquement distribuée sur les qubits. Les résultats obtenus pour ces codes de rendement constant sont prometteurs puisque nos simulations montrent que leur seuil est décent et que leurs performances à taille finie sont bonnes.

Abstract

Fault tolerant quantum computation is a technique to perform reliable quantum computation using noisy components. In this context, quantum error correcting codes are used to keep the amount of errors under a sustainable threshold. One of the main problems of this field is to determine the minimum cost, in terms of memory and time, which is needed in order to transform an ideal quantum computation into a fault-tolerant one. In this PhD thesis, we show that the family of quantum expander codes and the small-set-flip decoder can be used in the construction of ref. [46] to produce a fault-tolerant quantum circuit with constant space overhead.

The error correcting code family and the decoder that we study has been introduced in ref. [67] where an adversarial error model was examined. Based on the results of this article, we analyze quantum expander codes subjected to a stochastic error model which is relevant for fault-tolerant quantum computation [38], [37]. In addition, we show that the decoding algorithm can be parallelized to run in constant time. This is very relevant to prevent errors from accumulating while the decoding algorithm is running.

Beyond the theoretical results described above, we perform a numerical analysis of quantum expander codes to measure their performance in practice [49]. The error model used during these simulations generates X and Z type errors on the qubits with an independent and identically distributed probability distribution. Our results are promising because they reveal that these constant rate codes have a decent threshold and good finite length performance.

Contents

Remerciements

Merci à mes deux directeurs de thèse Anthony Leverrier et Omar Fawzi qui ont su me guider dans mes réflexions pour aboutir aux résultats présentés dans ce document. Merci à Anthony dont l'aide au quotidien à été indispensable pendant ces trois années passées à l'INRIA. Merci à Omar qui, bien que localisé à l'ENS Lyon, m'a fourni une aide toute aussi précieuse. J'ai tout particulièrement apprécié les visites d'Omar à l'INRIA pendant lesquelles nous avons pu discuter de vive voix tous les trois et progresser de manière particulièrement efficace face à nos problèmes. Grâce à eux, j'ai pu me familiariser avec beaucoup de notions d'informatique quantique plus ou moins proches de mon sujet de thèse. En particulier, merci à Omar qui m'a initié à cette discipline et sans qui, je n'aurais pas fait de thèse dans ce domaine.

Merci également à Vivien Londe dont la thèse a débuté et s'est terminée en même temps que la mienne. Nos discussions m'ont fait comprendre de nombreux concepts que je n'aurais certainement pas saisis sans lui. Je me souviens en particulier de ses explications à propos des symboles de Schläfli, de la géométrie hyperbolique et des polytopes abstraits. Sa bonne humeur et son sens de l'humour ont égaillé mes journées de travail.

Je tiens à remercier l'équipe SECRET ainsi que l'INRIA de Paris qui m'ont permis de mener à bien cette thèse dans de parfaites conditions. En particulier, je remercie Anne Canteaut et Christelle Guiziou pour m'avoir permis de participer aux conférences et autres manifestations scientifiques sans avoir à me soucier des tâches administratives. Merci à tous les autres membres de l'équipe, j'ai été très content de rencontrer et de travailler avec des personnes aussi compétentes et sympathiques: Simon Apers, Ivan Bardet, Xavier Bonnetain, Christina Boura, Rémi Bricout, Rodolfo Canto Torres, Kévin Carrier, Kaushik Chakraborty, André Chailloux, Pascale Charpin, Daniel Coggia, Thomas Debris, Shouvik Ghorai, Lucien Grouès, Matthieu Lequesne, Gaëtan Leurent, Andrea Olivo, María Naya-Plasencia, Léo Perrin, André Schrottenloher, Nicolas Sendrier, Ferdinand Sibleyras, Jean-Pierre Tillich, Valentin Vasseur, merci à tous.

Merci à David Poulin et Anirudh Krishna pour leur acceuil lors de ma visite de 2 mois dans leur équipe à l'Université de Sherbrooke. Ce séjour s'est conclu par la rédaction d'un article de recherche dont les résultats sont présentés dans ce manuscrit.

Merci à Daniel Gottesman et à Gilles Zémor d'avoir accepté d'être rapporteurs pour ma thèse, leurs commentaires et conseils ont été d'une grande utilité. Merci également à Earl Campbell, Eleni Diamanti et Jean-Pierre Tillich d'avoir accepté d'être membres du jury pour ma soutenance.

Je tiens bien sûr à remercier également toute ma famille en commençant par mes parents et ma soeur qui m'ont encouragé tout au long de mes études. Leurs conseils ont été indispensables et c'est grâce à eux que j'ai pu mener à bien cette thèse, les vacances en leur compagnie étant indispensables pour me ressourcer. Merci à ma grand-mère dont le dynamisme est un exemple pour moi, je ne peux qu'être admiratif devant ce qu'elle a réalisé tout au long de sa vie. Merci également à tous les autres membres de ma famille. Merci à Clara qui me supporte depuis maintenant trois ans et sans qui la vie

à Paris aurait été monotone. Je lui suis très reconnaissant pour ses nombreux conseils et encouragements qui m'ont porté lors de ces trois années.

Merci également à tous mes amis, qu'ils s'intéressent à l'informatique quantique ou non. Je pense en particulier à Thomas et Balthazar (même si je n'ai pas pu les voir suffisamment pendant les week-ends passés à Paris) et à Pauline, Grégoire, Léo et Hugo que j'ai toujours plaisir à voir lors de mes congés.

List of publications

1. Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Efficient decoding of random errors for quantum expander codes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 521–534. ACM, 2018.

2. Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Constant overhead quantum fault-tolerance with quantum expander codes. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 743–754. IEEE, 2018.

3. Antoine Grospellier and Anirudh Krishna. Numerical study of hypergraph product codes. *arXiv preprint arXiv:1810.03681*, 2018.

Chapter 1

Introduction (Français)

Un ordinateur quantique est un objet capable d'accomplir des calculs plus rapidement que les meilleurs algorithmes classiques, en tirant avantage des lois de la physique quantique. Shor a par exemple montré dans ref. [97] que cette technologie accélèrerait de manière exponentielle la résolution du problème de factorisation des entiers. D'autres exemples sont présentés dans [25] tels que l'algorithme de Grover qui permet une accélération quadratique pour une requête dans une base de données non structurée [50], ou bien l'algorithme HHL utilisé pour résoudre des systèmes linéaires [53]. Du point de vue pratique, l'un des objectifs à court terme est d'atteindre la suprématie quantique en construisant un dispositif quantique capable de faire une tâche spécifique plus rapidement que les super-ordinateurs actuels [11, 54, 13]. Pour résumer, un circuit quantique utilise des *portes quantiques* qui agissent sur des *états quantiques* utilisés pour stocker de l'information quantique dont l'unité de base est appelée le *qubit* (qui est une abréviation pour "bit quantique").

Aujourd'hui, plusieurs architectures sont en compétition telles que les circuits basés sur les supra-conducteurs ou bien les ions piégés. Toutefois, malgré de nombreux efforts et des progrès intéressants, les ordinateurs quantiques en sont encore au stade expérimental [60, 4, 75, 31, 114, 23, 33]. Le principal problème est qu'un tel système est inévitablement sujet à du bruit et aux phénomènes de décohérence. Ainsi, les états quantiques sont fragiles et une protection active est nécessaire pour y stocker et manipuler de l'information. La suprématie quantique pourrait être atteinte grâce à des technologies avec un niveau de bruit intermédiaire (dites "noisy intermediate-scale quantum technology") dont le but est de construire des appareils avec un bruit suffisamment petit pour accomplir un calcul non trivial qui soit hors de portée pour les ordinateurs classiques [87]. Toutefois, cette stratégie n'est pas viable pour effectuer des calculs longs et contrairement au cas des machines classiques, il est peu probable qu'il soit possible de supprimer complètement le bruit qui affecte les circuits quantiques. Heureusement, étant donné un circuit quantique C, il est possible de construire un autre circuit C' appelé *circuit tolérant aux fautes*, qui effectue le même calcul que C mais fonctionne également si ses composants sont bruités.

Pour formaliser l'idée de bruit, un circuit quantique est divisé en étapes élémentaires

appelées *sites* qui représentent les endroits où une erreur peut advenir. Par exemple, un *site de porte* représente un endroit dans le circuit où une porte est appliquée, et un *site d'attente* fait référence à un qubit sur lequel aucune action n'est faite pendant que des portes sont appliquées sur d'autres sites. Étant donné que les composants utilisés pour construire un circuit quantique sont bruités, certains sites sont *défaillants* ce qui signifie que l'opération effectuée par ce site n'est pas celle escomptée. Le modèle de bruit le plus simple est le modèle d'erreurs *iid* (indépendantes et identiquement distribuées) ou les sites sont défaillants indépendamment avec une certaine probabilité p. Le résultat fondamental dans le domaine du calcul tolérant aux fautes est appelé le *théorème du seuil* et a été prouvé par Aharonov et Ben-Or dans ref. [1]. Ils ont montré que si les sites sont sujets à un modèle de bruit iid de paramètre $p < p_{\rm th}$ ($p_{\rm th}$ est une constante universelle appelée *seuil*), alors pour tout circuit quantique C et pour toute probabilité cible $\epsilon > 0$, il existe un circuit tolérant aux fautes qui simule C et dont la probabilité d'échec est au plus ϵ. Au-delà du modèle de bruit iid, cette thèse de doctorat s'intéresse au modèle de bruit plus général appelé *stochastique local* [44, 46]. Nous voudrions mettre l'accent sur le fait que la tolérance aux fautes est également possible avec d'autres modèles de bruit [1, 2, 105].

Il existe deux manières naturelles de mesurer l'efficacité d'un protocole de tolérance aux fautes, à savoir le *coût en temps* et le *coût en espace*. Soit C un circuit quantique et soit C′ un circuit tolérant aux fautes pour C. Supposons que C (resp. C′) ait $|\mathrm{C}|$ sites (resp. $|\mathrm{C}'|$ sites), qu'il utilise m qubits (resp. m' qubits) et nécessite t étapes pour fonctionner (resp. t' étapes pour fonctionner). Alors, le coût en temps est défini comme le rapport t'/t et le coût en espace est m'/m. Par exemple, avec le théorème du seuil habituel de ref. [1], les coûts en temps et en espace sont polylogarithmiques en $|\mathrm{C}|$:

$$\frac{|\mathrm{C}'|}{|\mathrm{C}|} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right), \quad \frac{t'}{t} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right), \quad \frac{m'}{m} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right). \quad (1.1)$$

La principale motivation de cette thèse est ref. [46] où il est démontré qu'il est possible d'atteindre un coût en espace constant en utilisant des *codes correcteurs d'erreurs* bien choisis. En partant d'un état avec k qubits, l'idée de la correction d'erreurs quantiques est de l'encoder dans $n > k$ qubits en ajoutant de la redondance, de telle manière que des erreurs sur un petit nombre de qubits n'est pas problématique pour retrouver l'état initial. La procédure utilisée pour retrouver l'état initial est appelée *décodeur* et le temps d'exécution de cet algorithme classique est un paramètre critique. En effet, pour chaque site de C, le circuit tolérant aux fautes fourni par [46] nécessite d'exécuter le décodeur. Dans cette thèse de doctorat, nous nous intéressons aux *codes expanseurs quantiques* et nous montrons que le décodeur *small-set-flip* qui leur est associé peut être parallélisé pour fonctionner en temps constant. En suivant ref. [46], nous obtenons un protocole de tolérance aux fautes avec un coût en espace constant. De plus, chaque étape du circuit quantique résultant nécessite d'exécuter un circuit classique de profondeur constante (ceci est réaliste si les calculs classiques sont suffisamment rapides).

Le but de ce résumé est de décrire et d'expliquer notre résultat. Entre autres, nous aurons besoin de définir certains termes utilisés en correction d'erreurs classiques et de discuter les concepts de produits d'hypergraphes et de codes expanseurs classiques.

1.1 Correction d'erreurs classiques

Le domaine de l'informatique qui étudie la correction d'erreurs est appelé *théorie des codes* et a été initié par le travail de Claude Shannon en 1948 [94]. Shannon a montré qu'une transmission peut être décomposée en deux étapes: le *codage source* et le *codage de canal*. Le codage source est utilisé pour la compression de données [111, 26], mais nous n'en discutons pas dans cette thèse et nous nous focalisons sur l'étape du codage de canal où les codes correcteurs sont utilisés. Dans [94], Shannon a défini le concept de *capacité* qui est le taux maximal auquel de l'information peut être transmise de manière fiable sur un canal bruité donné. Il a également prouvé que la capacité de n'importe quel canal est atteinte par des codes correcteurs aléatoires qui, malheureusement, ne peuvent pas être implémentés de manière efficace. Alors que ces résultats révolutionnaires étaient publiés, Richard Hamming a introduit la première famille fonctionnelle de codes correcteurs qui est maintenant appelée le *code de Hamming* [52].

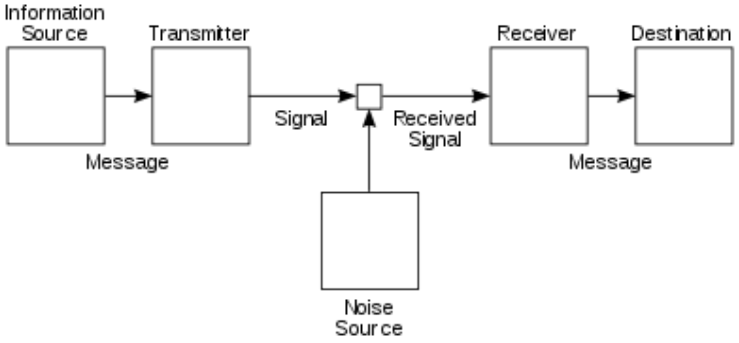

Figure 1.1: Le diagramme de Shannon pour la communication [94].

Les codes correcteurs d'erreurs sont souvent présentés à l'aide du scénario de communication où deux protagonistes, généralement appelés Alice et Bob, souhaitent communiquer. L'information classique est représentée par une suite de *bits* (des 0 et des 1) et donc un message de k bits est un vecteur binaire $s \in \mathbb{F}_2^k = \{0,1\}^k$. Alice et Bob sont éloignés et le seul canal de communication auquel ils ont accès est bruité. Cela signifie que certains des bits qu'Alice envoie à Bob sont corrompus par le canal. Lorsqu'Alice envoie un message $x \in \mathbb{F}_2^n$ à travers le canal bruité, Bob obtient comme sortie un autre message $y \in \mathbb{F}_2^n$ suivant une distribution de probabilité donnée qui dépend de x. Deux modèles de canaux importants sont le *canal binaire à effacement* et le *canal binaire symétrique*. Tous deux prennent en entrée un unique bit, c'est à dire que $n = 1$. Pour le canal binaire à effacement de paramètre $p \in [0,1]$, le bit qu'Alice a envoyé est correctement transmis avec probabilité $1-p$ et est effacé avec probabilité p. Lorsque Bob reçoit un 0 ou un 1, il sait que le bit est correct et lorsque le bit est effacé, il reçoit le symbole '?'. Pour le canal binaire symétrique, chaque bit transmis est *flippé* ($0 \mapsto 1$ and $1 \mapsto 0$) avec probabilité p et est correct avec probabilité $1-p$. Lorsque le canal est utilisé une seule fois, Alice peut envoyer un bit à Bob mais elle peut aussi

l'utiliser $n \in \mathbb{N}$ fois d'affilée pour envoyer n bits. Dans ce cas, nous supposons que les n utilisations du canal sont indépendantes en terme de probabilité.

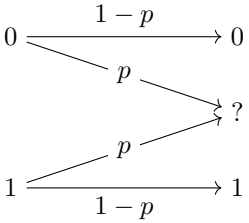

Figure 1.2: le canal binaire à effacement.

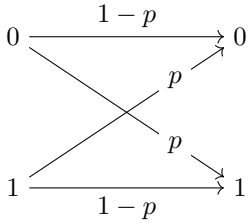

Figure 1.3: le canal binaire symétrique.

Dans ce qui suit, nous nous intéresserons principalement au cas du canal binaire symétrique. Le but d'Alice est d'envoyer un message arbitraire $s \in \mathbb{F}_2^k$ à Bob de telle manière qu'il puisse retrouver s avec une bonne probabilité. Bien évidemment, puisque n'importe quel message $s \in \mathbb{F}_2^k$ aurait pu être envoyé, Bob n'a aucun moyen de corriger un bit-flip sous l'hypothèse qu'Alice envoie s directement à travers le canal. À la place, elle peut envoyer $x \in \mathbb{F}_2^n$ une version encodée de s choisie de telle manière que Bob peut retrouver s même si le canal a flippé quelques bits de x. En particulier, les bits de x doivent contenir de la redondance et donc la condition $n > k$ doit être vérifiée. Un code correcteur d'erreurs $\mathcal{C} \subseteq \mathbb{F}_2^n$ satisfait $\left|\mathcal{C}\right| = \left|\mathbb{F}_2^k\right| = 2^k$ et est défini comme l'ensemble des mots binaires x, appelés les *mots de code*, qu'Alice peut envoyer à travers le canal.

Un code correcteur $\mathcal{C}$ est appelé un $[n, k]$ *code linéaire* lorsque l'ensemble des mots de code est un sous-espace de dimension k de $\mathbb{F}_2^n$ et, à partir de maintenant, les codes que nous considérerons sont linéaires. La *distance minimale* est le poids minimum d'un mot de code différent de zéro. Si la distance minimale d est connue, on dit que le code est un $[n, k, d]$ code linéaire. Les bits de s sont appelés les *bits logiques*, les bits de x sont appelés les *bits physiques* et le nombre de bits physiques est appelé la *taille du bloc*.

La Figure 1.4 représente le protocole basé sur les codes correcteurs d'erreurs qu'utilisent Alice et Bob pour communiquer avec un canal bruité:

- Alice encode le message s en un mot de code x et envoie x à Bob via le canal bruité.
- Bob obtient y comme sortie du canal et essaye d'inférer le mot de code x qu'Alice a envoyé, sa conjecture est notée $\hat{x}$.
- Bob décode $\hat{x}$ en un message $\hat{s}$ en appliquant l'inverse de la fonction qu'Alice a utilisée pendant l'étape d'encodage.

Le protocole ci-dessus est un succès lorsque $s = \hat{s}$ et est un échec lorsque $s \neq \hat{s}$. Dans cette thèse, nous nous intéressons aux codes linéaires pour lesquels la première et la troisième étape peuvent être faites facilement en utilisant de l'algèbre linéaire. Ainsi, nous allons principalement nous intéresser à la seconde étape, appelée l'*étape de correction d'erreurs* ou l'*étape de décodage*, où Bob déduit $\hat{x}$ à partir de y en utilisant un algorithme appelé l'*algorithme de décodage* ou le *décodeur*. L'un des principaux

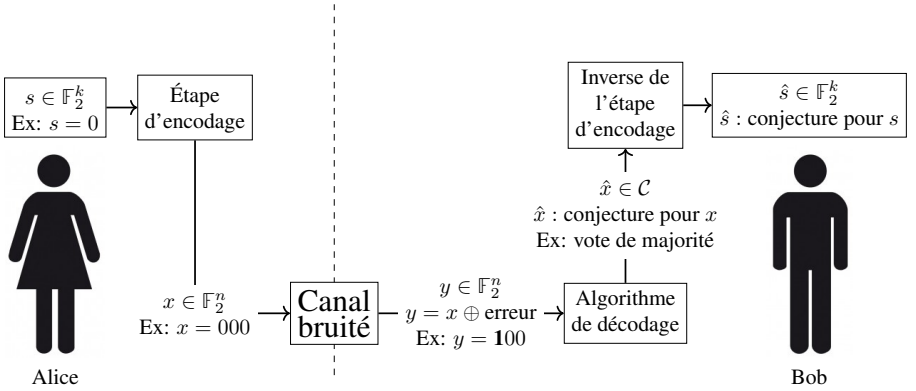

Figure 1.4: une communication classique avec des codes correcteurs d'erreurs. Par exemple, Alice envoie le bit $s \in \mathbb{F}_2$ trois fois à Bob qui peut le retrouver en utilisant un vote de majorité (ceci est appelé le code de répétition à 3 bits).

objectifs de la théorie des codes est de trouver des familles de codes correcteurs pour lesquels le rapport k/n est large et qui admettent un décodeur efficace en temps dont la probabilité d'échec est faible.

L'un des principaux résultats de ref. [94] est la possibilité d'utiliser un canal bruité pour transmettre de l'information de manière fiable avec un taux constant. Au lieu d'utiliser un unique code correcteur d'erreurs, considérons une famille de codes $\mathcal{C}_1, \mathcal{C}_2, \ldots$ où $\mathcal{C}_i$ encode k_i bits logiques avec n_i bits physiques. Lorsque le rapport k_i/n_i converge et n_i tend vers l'infini, la limite $r \in [0, 1]$ est appelée le *rendement asymptotique* de la famille. De plus, quand un canal binaire symétrique de paramètre p est utilisé, le *seuil* de $\mathcal{C}_i$ est le nombre réel maximal $p \in [0, 1]$ tel que la probabilité d'échec tend vers 0 lorsque i tend vers l'infini. Nous disons que $\mathcal{C}_i$ permet une transmission fiable via le canal binaire symétrique quand le paramètre p est en dessous du seuil de la famille de codes. Un canal binaire symétrique avec paramètre fixe $p \in (0, 1)$ ne peut pas transmettre de l'information de manière fiable avec un rendement arbitrairement proche de 1. En fait, il existe une limite appelée la *capacité* du canal qui est une borne supérieure sur le rendement asymptotique de n'importe quelle famille de code qui permet une transmission fiable. Le notion de capacité a été introduite par Shannon dans [94] et est égale à $C_{\text{BEC}}(p) := 1 - p$ pour le canal binaire à effacement et à $C_{\text{BSC}}(p) := 1 - h_2(p)$ pour le canal binaire symétrique où $h_2(p) := -p \log_2(p) - (1 - p) \log_2(1 - p)$ est l'entropie binaire.

Dans sa thèse en 1962 [42], Robert Gallager a introduit la famille des codes LDPC (Low Density Parity Check) et a suggéré de les décoder avec un algorithme appelé *décodeur itératif*. Toutefois à cette époque, les ordinateurs n'étaient pas suffisamment puissants pour implémenter le décodeur itératif et donc les codes LDPC n'avaient pas d'intérêt pratique. À la fin des années 1990, il y a eu un regain d'intérêt pour les codes LDPC avec les papiers [70, 71, 73, 72]. De nos jours, ces codes sont utilisés de manière intensive dans les réseaux de communication [91, 95].

Afin de définir les codes LDPC, nous avons besoin d'expliquer les concepts de *matrice de parité* et de *graphe de Tanner*. Soit $\mathcal{C}$ un code linéaire et soit H une matrice binaire telle que $\mathcal{C} = \ker(H)$. On dit alors que H est une *matrice de parité* pour $\mathcal{C}$. Un *graphe de Tanner* G pour $\mathcal{C}$ est un graphe biparti construit à partir de H de la manière suivante: les sommets gauches de G représentent les colonnes de H, les sommets droits représentent les lignes de H et deux sommets sont reliés si et seulement si l'entrée correspondante dans H est égale à 1 [104]. Par définition, une famille de codes est LDPC si et seulement si les degrés dans le graphe de Tanner sont bornés supérieurement par une constante uniforme.

Au début des années 1990, les *turbo codes* de Berrou, Glavieux et Thitimajshima [9] ont fait prendre conscience à la communauté scientifique qu'il est possible d'implémenter en pratique des codes correcteurs dont la probabilité de succès est haute et dont le rendement est proche de la capacité du canal. Les turbo codes sont basés sur les *codes de convolution* [59] introduits par Elias dans [36]. Pour un code de convolution, un flux de bits est encodé dans un autre flux de bits. Dans cette thèse, nous nous intéressons aux *codes par blocs* tels que les codes LDPC qui, par opposition aux codes de convolution, encode un bloc de bits dans un autre bloc plus large (un bloc est une chaîne de bits de taille fixée).

Pour simplifier, les codes classiques que nous considérerons sont des *codes LDPC réguliers* comme définis par Gallager dans sa thèse [42]. Par définition, un code régulier est tel que les sommets droits dans le graphe de Tanner ont degré d_V et les sommets gauche ont degré d_C où $d_V, d_C \in \mathbb{N}$ sont des entiers fixés. On remarque toutefois que les performances des codes LDPC réguliers ne sont pas aussi bonnes que celles d'autres familles de codes LPDC. Par exemple, ref. [83] montre qu'une famille de codes réguliers ne peut pas atteindre la capacité du canal binaire à effacement. Il a également été démontré que les codes irréguliers ont de meilleurs seuils [72, 73, 71, 70] et de meilleures performances à taille finie [90, 110, 113, 115]. Plus généralement, de nombreuses familles de codes ont été introduites dans le but d'augmenter la probabilité de succès et de réduire le temps nécessaire aux étapes d'encodage et de décodage. On peut par exemple citer les codes concaténés [40], les codes algébriques de Reed-Solomon [103, 64, 89] et plus récemment les codes LDPC spatialement couplés [39], les codes polaires [3] et les codes irréguliers basés sur les protographes [106]. De plus, il existe de nombreuses techniques pour optimiser les codes LDPC [24, 93, 57, 107].

Pour plus de détails à propos des codes correcteurs d'erreurs classiques, voir [111, 92, 59].

1.2 Information quantique

Cette partie introduit les concepts fondamentaux de la théorie de l'information quantique dont nous aurons besoin pour expliquer nos résultats, voir [82] pour plus de détails à propos d'information quantique et de calcul quantique.

Un *état pur* sur n qubits $|\psi\rangle \in \mathbb{C}^{2^n}$ ("ket psi") est défini mathématiquement par un vecteur complexe normalisé pour la norme Euclidienne avec 2^n entrées. Il est pratique de voir $|\psi\rangle$ comme un élément de $\left(\mathbb{C}^2\right)^{\otimes n}$ où tout au long de ce manuscrit, $\otimes$ est le

produit tensoriel (ou le *produit de Kronecker*). Le vecteur ligne $\langle\psi| := |\psi\rangle^\dagger$ ("bra psi") est défini comme étant le transpose conjugué de $|\psi\rangle$. De plus, pour $i \in [\![0; 2^n - 1]\!]$, l'état $|i\rangle$ est le vecteur colonne dont la seule entrée non-nulle est un 1 à la $i^{\text{ème}}$ position.

La dynamique des états quantiques purs est divisée en deux types d'opérations: les *évolutions unitaires* et les *mesures*. Une évolution unitaire est de la forme $U : |\psi\rangle \mapsto U|\psi\rangle$ où U est n'importe quelle matrice unitaire complexe de dimension $2^n \times 2^n$. Parmi les matrices unitaires, le *groupe de Pauli* est particulièrement important et est défini par:

$$\mathcal{P}_n := \left\{ \alpha P : \alpha \in \left\{1, -1, i, -i\right\} \text{ et } P \in \left\{\mathbb{1}, X, Y, Z\right\}^{\otimes n} \right\}$$

où X est appelé le *bit-flip* et Z est appelé le *phase-flip*:

$$X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Y := iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Pour une erreur de Pauli $P \in \mathcal{P}_n$, le poids de P désigné par $|P|$ est défini comme étant le nombre de qubits sur lesquels P agit de manière triviale. Formellement, si nous écrivons $P = \alpha P_1 \otimes \ldots \otimes P_n$ où $\alpha \in \{1, -1, i, -i\}$ est une phase globale et $P_1, \ldots, P_n \in \{\mathbb{1}, X, Y, Z\}$ sont des matrices de Pauli agissant sur un qubit, alors $|P|$ est défini par:

$$|P| := \#\Big\{ i \in [\![1; n]\!] : P_i \neq \mathbb{1} \Big\}.$$

Une matrice Π est un *projecteur orthogonal* lorsque $\Pi^2 = \Pi$ et $\Pi = \Pi^\dagger$ où $\Pi^\dagger$ est l'opérateur adjoint de Π (l'opérateur adjoint est la matrice transpose conjuguée).
En général, les mesures d'états purs $|\psi\rangle \in \mathbb{C}^{2^n}$ ont pour sortie un bit classique $b \in \mathbb{F}_2$ et modifient $|\psi\rangle$ (on dit que $|\psi\rangle$ *est réduit* à un autre état quantique). Pour la correction d'erreurs, nous nous intéressons aux *mesures projectives* définies par deux projecteurs orthogonaux Π_0, Π_1 de taille $2^n \times 2^n$ satisfaisant $\Pi_0 + \Pi_1 = \mathbb{1}$ où $\mathbb{1}$ est la matrice identité:

- Avec probabilité $\langle\psi|\,\Pi_0\,|\psi\rangle$: la sortie de la mesure est $b = 0$ et l'état est réduit à $\dfrac{\Pi_0\,|\psi\rangle}{\sqrt{\langle\psi|\,\Pi_0\,|\psi\rangle}}$.
- Avec probabilité $\langle\psi|\,\Pi_1\,|\psi\rangle$: la sortie de la mesure est $b = 1$ et l'état est réduit à $\dfrac{\Pi_1\,|\psi\rangle}{\sqrt{\langle\psi|\,\Pi_1\,|\psi\rangle}}$.

En particulier, une *mesure de Pauli* associée à un opérateur de Pauli $P \in \mathcal{P}_n$ est la mesure projective définie à partir de Π_0 et Π_1 où:

- $\Pi_0 := (\mathbb{1} + P)/2$ le projecteur orthogonal associé à la valeur propre $+1$ de P.
- $\Pi_1 := (\mathbb{1} - P)/2$ le projecteur orthogonal associé à la valeur propre -1 de P.

En général, un système quantique S n'est pas isolé dans le sens où il interagit avec d'autres systèmes que l'on appelle l'*environnement*. Dans ce cas, l'état de S n'est pas nécessairement un état pur mais est toujours un mélange probabiliste d'états purs. Une *matrice de densité* est un moyen pratique de représenter un tel mélange par un opérateur semi-défini positif de taille $2^n \times 2^n$ avec des coefficients complexes et trace 1. Par exemple, la matrice de densité associée à un état pur $|\psi\rangle$ est $|\psi\rangle\langle\psi|$ où $\langle\psi| = |\psi\rangle^\dagger$ comme précédemment. Plus généralement, soient $|\psi_1\rangle, \ldots, |\psi_m\rangle$ des états purs et soient $p_1, \ldots, p_m \in [0,1]$ des probabilités telles que $p_1 + \ldots + p_m = 1$. Alors, la matrice de densité d'un système où chaque état $|\psi_i\rangle$ a été préparé avec probabilité p_i est:

$$\rho = \sum_{i=1}^{m} p_i |\psi_i\rangle\langle\psi_i| . \tag{1.2}$$

Lors de l'interaction d'un système quantique S avec un environnement E, le nombre de qubits dans S peut changer. En effet, certains qubits de S peuvent être jetés et certains qubits de E peuvent être ajoutés à S. Formellement, n'importe quelle évolution d'un système avec n qubits vers un système avec n' qubits peut être représenté par une *application CPTP* $\mathcal{E}$ (completely positive trace preserving map) parfois appelée un *canal quantique* ou une *opération quantique*. Une application CPTP est définie par un ensemble de matrices complexes E_k de taille $2^{n'} \times 2^n$ appelées les *opérateurs de Kraus* satisfaisant $\sum_k E_k^\dagger E_k = \mathbb{1}$:

$$\mathcal{E} : \rho \mapsto \sum_{k=1}^{K} E_k \rho E_k^\dagger .$$

Par exemple, une évolution unitaire U est représentée par un unique opérateur de Kraus $E_1 = U$ où $n = n'$.

Comme indiqué précédemment, la mesure d'un opérateur de Pauli $P \in \mathcal{P}_n$ d'un état pur $|\psi\rangle$ est la mesure projective associée à $\Pi_0 := (\mathbb{1} + P)/2$ et $\Pi_1 := (\mathbb{1} - P)/2$. Soit $\mathcal{M}$ une application CPTP représentant cette mesure et soient:

$$p_0 := \langle\psi|\Pi_0|\psi\rangle , \qquad |\psi_0\rangle := \frac{\Pi_0|\psi\rangle}{\sqrt{\langle\psi|\Pi_0|\psi\rangle}},$$
$$p_1 := \langle\psi|\Pi_1|\psi\rangle , \qquad |\psi_1\rangle := \frac{\Pi_1|\psi\rangle}{\sqrt{\langle\psi|\Pi_1|\psi\rangle}}.$$

Alors $\mathcal{M}$ est équivalente à préparer l'état $|\psi_0\rangle$ avec probabilité p_0 et l'état $|\psi_1\rangle$ avec probabilité p_1. En utilisant eq. (1.2), nous avons:

$$\mathcal{M} : |\psi\rangle\langle\psi| \mapsto |0\rangle\langle 0| \otimes \Pi_0|\psi\rangle\langle\psi|\Pi_0 + |1\rangle\langle 1| \otimes \Pi_1|\psi\rangle\langle\psi|\Pi_1.$$

Ici, pour simplifier, la sortie classique de la mesure $b \in \{0,1\}$ est stockée comme un qubit. Plus généralement, pour n'importe quelle matrice de densité ρ nous avons:

$$\mathcal{M} : \rho \mapsto |0\rangle\langle 0| \otimes \Pi_0\rho\Pi_0 + |1\rangle\langle 1| \otimes \Pi_1\rho\Pi_1. \tag{1.3}$$

Les opérateurs de Kraus de $\mathcal{M}$ sont $E_0 = |0\rangle \otimes \Pi_0$ et $E_1 = |1\rangle \otimes \Pi_1$ avec $n' = n + 1$.

1.3 Correction d'erreurs quantiques

Dans la Section 1.1, nous avons utilisé le scénario de communication présenté dans la Figure 1.4 pour introduire les codes correcteurs classiques. De manière similaire, un *code correcteur quantique* peut être utilisé dans un protocole où Alice souhaite envoyer à Bob un état quantique $|\varphi\rangle$ avec K qubits via un canal de communication bruité. L'idée générale est la même que dans le cas classique: elle commencent par encoder $|\varphi\rangle$ de manière redondante dans un état $|\psi\rangle$ avec N qubits, puis elle envoie $|\psi\rangle$ à travers le canal bruité et finalement Bob applique une opération quantique pour corriger ρ l'état mixte reçu. La communication est un succès quand l'état $|\hat{\varphi}\rangle$ que Bob obtient à la fin est égal à $|\varphi\rangle$. Dans ce protocole, chaque état $|\varphi\rangle$ est encodé par un *état du code* $|\psi\rangle$ et l'ensemble des états du code noté $\mathcal{Q}$ est appelé l'*espace de code* ou un *code correcteur d'erreurs quantiques*. Par définition, l'encodage est une application linéaire de $\left(\mathbb{C}^2\right)^{\otimes K}$ vers $\left(\mathbb{C}^2\right)^{\otimes N}$ et donc l'ensemble $\mathcal{Q}$ est un sous-espace vectoriel de $\left(\mathbb{C}^2\right)^{\otimes N}$ de dimension 2^K. Le nombre entier K est appelé le nombre de *qubits logiques* et N est appelé le nombre de *qubits physiques*.

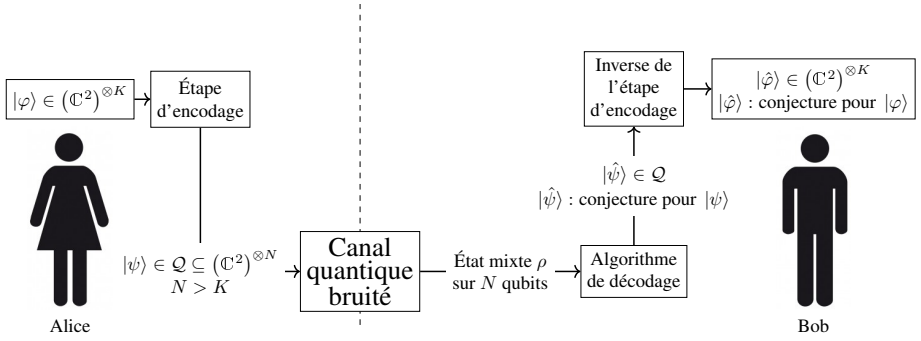

Figure 1.5: scénario de communication pour les codes correcteurs quantiques.

Dans ce travail, nous nous intéressons aux canaux quantiques qui satisfont la propriété *stochastique locale* de paramètre $p \in [0, 1]$. Un modèle de bruit est stochastique local quand la probabilité qu'un certain ensemble de qubits S soit dans le support de l'erreur décroît exponentiellement en $|S|$. Formellement, soit $V_{\mathcal{Q}}$ l'ensemble des qubits et soit $\mathcal{E}$ un canal quantique alors:

- $\mathcal{E}$ a la propriété d'être *stochastique* si $\mathcal{E}$ peut être décrit par un processus en deux étapes:
  - Lors de la première étape, un ensemble aléatoire $E \subseteq V_{\mathcal{Q}}$ appelé le *support* de l'erreur est choisi aléatoirement.
  - Lors de la deuxième étape, un canal quantique $\mathcal{E}_E$ est appliqué sur les qubits. Ici, pour chaque $F \subseteq V_{\mathcal{Q}}$, $\mathcal{E}_F$ est une application CPTP agissant sur les qubits de F.

- $\mathcal{E}$ est *stochastique local* de paramètre $p \in [0,1]$ s'il est stochastique et la distribution de probabilité de E satisfait pour tout $S \subseteq V_{\mathcal{Q}}$:

$$\mathbb{P}\Big[S \subseteq E\Big] \leq p^{|S|}. \tag{1.4}$$

En d'autres termes, un bruit stochastique local $\mathcal{E}$ est tel que:

$$\mathcal{E} : \rho \mapsto \sum_{F \subseteq V_{\mathcal{Q}}} \mathbb{P}\Big[E = F\Big] \mathcal{E}_F(\rho),$$

où le support $E \subseteq V_{\mathcal{Q}}$ est choisi aléatoirement de telle manière que eq. (1.4) soit vérifiée pour tout $S \subseteq V_{\mathcal{Q}}$.

Par exemple, le canal qui applique des bit-flips et des phase-flips de manière indépendante et identiquement distribuée a la propriété stochastique locale. Un autre exemple est le *canal dépolarisant* qui applique sur chaque qubit indépendamment le canal $\mathcal{D}$ défini ci-dessous:

$$\mathcal{D} : \rho \mapsto (1-p)\rho + \frac{p}{3}\Big(X\rho X + Y\rho Y + Z\rho Z\Big).$$

Le canal dépolarisant pour n qubits est égal à $\mathcal{D}^{\otimes n}$ et est stochastique local de paramètre $p/3$.

Lorsque nous parlerons de *probabilité de succès* (ou de probabilité pour l'erreur d'être corrigée) pour un bruit stochastique local, cela fera référence à la probabilité sur E pour la communication d'être un succès:

$$\mathbb{P}\Big[\text{Succès}\Big] = \sum_{F \subseteq V_{\mathcal{Q}}} \mathbb{P}\Big[E = F\Big] \mathbb{P}\Big[\, |\hat{\varphi}\rangle = |\varphi\rangle \;\Big|\; E = F\Big].$$

Notons que si l'algorithme de décodage est déterministe, on a $\mathbb{P}\Big[\, |\hat{\varphi}\rangle = |\varphi\rangle \;\Big|\; E = F\Big] \in \{0,1\}$.

Dans le cas classique, nous nous sommes concentrés sur le modèle de bit-flips qui est assez réaliste. En revanche dans le cas quantique, l'ensemble des erreurs possibles est infini: n'importe quelle unitaire ou même n'importe quelle application CPTP peut potentiellement être appliquée sur l'état que l'on essaye de protéger. De manière surprenante, il suffit de corriger l'ensemble fini des erreurs de Pauli pour corriger les erreurs générales (voir [8] ou [82]). Par exemple, si une procédure de correction d'erreurs est capable de corriger n'importe quelle erreur de Pauli qui agit sur les qubits de $F \subseteq V_{\mathcal{Q}}$, alors la même procédure corrige une application CPTP arbitraire agissant sur F. En conséquence, pour un modèle de bruit stochastique local, nous pouvons supposer sans perte de généralité que les applications CPTP $\mathcal{E}_F$ sont des canaux de Pauli (c'est à dire qu'ils appliquent des erreurs de Pauli sur les qubits de F).

Dans les années 1990, la question de savoir si la correction d'erreurs quantiques est possible était débattue jusqu'à ce que Shor [96] et Steane [102] montrent que c'est en effet possible. Suite à ces travaux fondateurs, la théorie de la correction d'erreurs

quantiques a été développée, par exemple dans [63, 20, 101, 43]. En particulier dans [43], Gottesman a introduit les outils mathématiques utilisés pour définir et étudier une large classe de codes quantiques appelés les *codes stabilisateurs*. Un code stabilisateur $\mathcal{Q}$ sur N qubits physiques est défini à partir d'un ensemble fini d'opérateurs de Pauli appelés les *générateurs du stabilisateur*. Ces générateurs $g_1, \ldots, g_M \in \mathcal{P}_N$ doivent commuter et $\mathcal{Q}$ est alors défini de la manière suivante:

$$\mathcal{Q} := \Big\{ |\psi\rangle \in \left(\mathbb{C}^2\right)^{\otimes N} : g_1 |\psi\rangle = \ldots = g_M |\psi\rangle = |\psi\rangle \Big\}.$$

Le code $\mathcal{Q}$ est appelé un $[[N; K]]$ code stabilisateur où K est le nombre de qubits logiques.

Dans cette thèse, nous nous concentrons sur un groupe particulier de codes stabilisateurs appelés les codes CSS LDPC. La terminologie "CSS" vient de Calderbank, Shor et Steane qui ont introduit la construction CSS en 1996 [20, 102]. Un code CSS est construit à partir de deux codes classiques $\mathcal{C}_X$ et $\mathcal{C}_Z$ avec $\mathcal{C}_Z^{\perp} \subseteq \mathcal{C}_X$ où $\mathcal{C}_X$ est utilisé pour corriger les erreurs de bit-flips et $\mathcal{C}_Z$ est utilisé pour corriger les erreurs de phase-flips. La première étape de la procédure de correction d'erreurs est de mesurer les générateurs du stabilisateur. La sortie de la mesure est un mot binaire appelé *syndrome*. D'un point de vue pratique, pour être capable de mesurer les générateurs du stabilisateur, il est important que chacun implique un petit nombre de qubits et que chaque qubit soit impliqué dans un petit nombre de générateurs. Ces propriétés sont satisfaites quand les deux codes classiques utilisés pour construire le code CSS sont LDPC et dans ce cas le code est dit CSS LDPC.

La distance minimale d'un code stabilisateur est le poids minimum d'une erreur de Pauli qui envoie un état du code sur un autre état du code orthogonal au premier. Cette quantité est une bonne indication de la performance du code et c'est une question ouverte de savoir si des codes avec distance minimale $D = \Theta(N)$ existent. Sans la contrainte LDPC, cette question a été résolue par l'affirmative lorsque les codes CSS ont été introduits dans [20, 102]. Pour les codes LDPC, la meilleure distance minimale connue est $D = \Theta(\sqrt{N}\ \sqrt[4]{\log(N)})$ pour une famille de codes avec $K = 1$ qubit logique, dont la construction s'appuie sur une variété en quatre dimensions [41]. Avec la contrainte additionnelle d'un rendement constant $K = \Theta(N)$, la meilleure distance minimale connue est $D = \Theta(\sqrt{N})$ et est atteinte par le produit d'hypergraphes de Tillich et Zémor [108]. Quand la distance minimale D d'un $[[N; K]]$ code stabilisateur est connue, on dit que le code est un $[[N; K; D]]$ code stabilisateur.

Un *décodeur* ou *algorithme de décodage* pour un code stabilisateur est un algorithme classique qui prend en entrée un syndrome et renvoie une erreur de Pauli. L'algorithme réussit lorsqu'appliquer cette Pauli sur l'état quantique reçu par Bob, lui fait retrouver l'état originel qu'Alice a envoyé par le canal quantique. Pour les codes CSS, le but du décodeur est de corriger les deux codes classiques initiaux en prenant en compte la *dégénérescence* du code CSS. Le terme "dégénérescence" fait référence au fait qu'il existe des erreurs *équivalentes*, *i.e.* des opérateurs de Pauli différents qui agissent de la même façon sur les états du code. Par exemple, un générateur du stabilisateur d'un code envoie n'importe quel état du code sur lui-même. Il est donc équivalent à l'identité et aux autres générateurs du stabilisateur. Par comparaison, dans le cas classique, deux

erreurs agissent de la même façon sur un mot de code si et seulement si elles sont égales. À cause de la dégénérescence, le décodeur d'un code quantique peut avoir corrigé l'erreur même s' il n'a pas retrouvé l'erreur physique qui est advenue sur l'état initial. Ceci est crucial pour les code CSS LDPC puisque les codes classiques utilisés dans la construction doivent contenir des mots de code de poids constant. Les codes CSS LDPC sont donc fortement dégénérés. Un exemple de décodage avec de bonnes performances est le *décodeur de poids minimum*: il renvoie une erreur de Pauli de poids minimum dont le syndrome est égal au syndrome d'entrée. Un *seuil* pour une famille de codes quantiques est une probabilité non nulle $p_{\text{th}} \in (0, 1]$ telle que l'erreur générée par un canal stochastique local de paramètre $p < p_{\text{th}}$ est corrigée avec une probabilité qui tend vers 1 quand la taille de bloc tend vers l'infini.

Parmi les codes CSS LDPC, le *code torique* introduit par Kitaev est le plus connu et a été abondamment étudié [61, 62, 30, 86, 28, 68]. Le code torique de paramètre $L \in \mathbb{N}^*$ est défini à partir du pavage d'un tore en deux dimensions, ce qui conduit à un $[[2L^2, 2, L]]$ CSS LDPC code stabilisateur. Le code torique a beaucoup d'avantages, par exemple sa distance minimale $D = \Theta(\sqrt{N})$ est large et les générateurs du stabilisateur impliquent seulement des interactions plus proches voisins. Ceci est approprié pour une implémentation puisqu'un tore en deux dimensions peut être plongé dans notre espace euclidien à trois dimensions. Ainsi, une implémentation du code torique nécessiterait de ne faire interagir que des qubits proches dans l'espace. De plus, les performances du code torique sont vraiment bonnes même pour de petites tailles de blocs. Enfin, le décodage de plus petit poids peut être implémenté en temps polynomial avec l'algorithme de Edmonds [30, 34].

Au-delà du code torique, n'importe quel pavage d'une variété définit un code CSS LDPC appelé *code topologique*. Ce principe permet d'utiliser de puissants arguments issus de la topologie pour étudier de tels codes et pour concevoir des décodeurs. Voir par exemple le code de surface [15], les codes hyperboliques de dimension deux [16], les codes semi-hyperboliques [17] et les codes hyperboliques de dimension quatre [51, 69, 55]. Il existe deux résultats qui imposent des contraintes fortes sur le compromis entre les paramètres N, K et D d'un code topologique en deux dimensions [14, 27]. Le premier établi que $KD^2 = \mathcal{O}(N)$ et est satisfait pour n'importe quel code topologique construit à partir d'une variété euclidienne de dimension deux [14]. Le second résultat est $KD^2 = \mathcal{O}(N \log^2(N))$ et doit être satisfait même si l'espace sous-jacent n'est pas euclidien [27]. Par conséquent, l'avantage d'utiliser des codes topologiques construits à partir d'espaces de dimension quatre est d'outrepasser ces deux résultats négatifs. En particulier, les paramètres $K = \Theta(N)$ et $D = \Omega(N^{0.2})$ sont atteints pour des codes hyperboliques en quatre dimensions [51, 69]. Notons en revanche qu'il n'est pas possible de plonger un code hyperbolique ou un code de dimension quatre dans notre monde réel en conservant des interactions plus proches voisins.

Une autre généralisation du code torique est le *code produit d'hypergraphes* introduit par Tillich et Zémor dans [108]. Cette construction combinatoire a l'avantage de construire des codes quantiques à partir de bons codes classiques et donc des arguments de théorie des codes peuvent être utilisés pour les étudier. Si les codes classiques initiaux sont LDPC, ont un rendement constant et ont une distance minimale linéaire, alors le produit d'hypergraphes correspondant est également LDPC, a un rendement constant et

sa distance minimale croît comme la racine carrée de la taille de bloc. Ce comportement asymptotique des paramètres est l'un des meilleurs parmi les constructions connues de codes quantiques. Le principal objet d'intérêt dans cette thèse de doctorat est une famille spécifique de produits d'hypergraphes appelés *codes expanseurs quantiques* [67]. Contrairement aux codes produits d'hypergraphes plus généraux, les codes expanseurs quantiques ont l'avantage d'être équipés d'un décodeur efficace appelé le *décodeur small-set-flip*.

Dans Figure 1.6, nous donnons des exemples de codes CSS LDPC. Les colonnes "Dimension" et "Distance minimale" contiennent les paramètres K et D des codes. Par définition de la distance minimale, n'importe quelle erreur de poids jusqu'à $\lfloor (D-1)/2 \rfloor$ est corrigée par le décodeur de poids minimal qui (en général) fonctionne en temps exponentiel. Toutefois, les décodeurs connus fonctionnant en temps polynomial ne corrigent pas nécessairement toutes les erreurs jusqu'à une fraction de la distance minimale. C'est pourquoi, nous reportons dans la colonne "Poids maximum d'une erreur adverse corrigée par un décodeur efficace", la meilleure valeur connue T telle que n'importe quelle erreur de poids jusqu'à T soit corrigée par un algorithme de décodage en temps polynomial.

	Dimension	Distance minimale	Poids maximum d'une erreur adverse corrigée par un décodeur efficace
Code torique [62]	2	$\Theta(\sqrt{N})$	$\Theta(\sqrt{N})$
À partir de variété 4D [41]	1	$\Theta(\sqrt{N}\sqrt[4]{\log(N)})$	Pas de décodeur efficace
Hyperbolique 2D [41]	$\Theta(N)$	$\Theta(\log N)$	$\Theta(\log N)$
Hyperbolique 4D [51, 55, 69]	$\Theta(N)$	$\Omega(N^{0.2}), \mathcal{O}(N^{0.3})$	$\Theta(\log N)$
Code produit d'hypergraphes [108]	$\Theta(N)$	$\Theta(\sqrt{N})$	Pas de décodeur efficace
Codes expanseurs quantiques [67]	$\Theta(N)$	$\Theta(\sqrt{N})$	$\Theta(\sqrt{N})$

Figure 1.6: exemples de codes CSS LDPC.

1.4 Codes produits d'hypergraphes

La question de savoir s'il existe des codes CSS LDPC avec rendement constant et distance minimale linéaire est toujours ouverte. Une approche naïve serait de construire un code CSS à partir de deux codes LDPC $\mathcal{C}_X$ et $\mathcal{C}_Z$ avec une bonne distance minimale. Malheureusement, si $\mathcal{C}_Z$ est LDPC alors l'espace vectoriel $\mathcal{C}_Z^\perp$ contient des éléments de poids constant. Ainsi, l'inclusion $\mathcal{C}_Z^\perp \subseteq \mathcal{C}_X$ implique que le distance minimale de $\mathcal{C}_X$ est constante. C'est pourquoi, de bons codes LDPC classiques ne peuvent pas être utilisés directement pour construire de bons codes CSS LDPC.

Un code produit d'hypergraphes est un code CSS $\mathcal{Q}$ issu de deux codes classiques $\mathcal{C}_1$ et $\mathcal{C}_2$ sans condition supplémentaire telle que $\mathcal{C}_2^\perp \subseteq \mathcal{C}_1$. Plus précisément, deux codes classiques $\mathcal{C}_X$ et $\mathcal{C}_Z$ sont construits à partir de $\mathcal{C}_1$ et $\mathcal{C}_2$, et le code produit d'hypergraphes est le code CSS associé à $\mathcal{C}_X$ et $\mathcal{C}_Z$. Ceci est particulièrement intéressant pour les codes

LDPC parce que si $\mathcal{C}_1$ et $\mathcal{C}_2$ sont des codes LDPC de rendement constant alors $\mathcal{Q}$ l'est aussi. De plus, quand $\mathcal{C}_1$ et $\mathcal{C}_2$ ont des distances minimales linéraires, le produit d'hypergraphes a une distance minimale égale à $D = \Theta(\sqrt{N})$ où N est le nombre de qubits physiques. Grâce à ces paramètres favorables, il est attendu que les codes produits d'hypergraphes se comportent favorablement pour la correction d'erreurs quantiques et le calcul quantique tolérant aux fautes.

Une idée naturelle pour décoder les codes produits d'hypergraphes est d'utiliser les décodeurs des codes classiques. Toutefois, cette stratégie ne marche pas dans le cas général. Par exemple, le décodage itératif est utilisé de manière intensive dans le cas classique mais ne fonctionne pas bien pour les codes produits d'hypergraphes [86, 78]. Heureusement, il peut être amélioré en utilisant des réseaux de neurones [68] ou l'amélioration OSD (ordered statistics decoding) [85]. Ref. [85] montre que les produits d'hypergraphes ont de bonnes performances en pratique: pour un canal dépolarisant avec un taux d'erreurs physiques en dessous de 10%, un code produit d'hypergraphes bien choisi avec 28 qubits logiques a de meilleures performances qu'un code de surface avec 1 qubit logique décodé avec le décodeur de poids minimum (voir la figure 3 de [85]).

1.5 Codes expanseurs quantiques

En 1996, Sipser et Spielman ont introduit le concept de *graphes expanseurs* et ont défini un *code expanseur classique* comme étant un code dont le graphe de Tanner est un expanseur [98]. La propriété d'expansion est utile pour montrer certaines propriétés du code, comme par exemple le fait que la distance minimale d'un code expanseur classique est linéaire en la taille du bloc. Pour cette thèse, nous nous intéressons en particulier à la manière dont une expansion suffisante implique que n'importe quelle erreur de poids jusqu'à une fraction de la distance minimale est corrigée par un décodeur appelé l'algorithme de *bit-flip*. L'algorithme de bit-flip est l'un des algorithmes les plus simples qu'il soit possible de concevoir, et c'est pourquoi il est intéressant pour une analyse théorique. Toutefois, les simulations numériques montrent que le décodage itératif est le meilleur décodage pour les codes LDPC classiques et qu'il bat largement l'algorithme de bit-flip. De plus, le décodage itératif peut également être analysé avec des arguments d'expansion [19].

Le problème de trouver les paramètres d'expansion d'un graphe est un problème co-NP-difficile [10] et la construction de codes expanseurs n'est pas aisée. Dans ce travail, nous nous appuyons sur une construction probabiliste appelée le *modèle de configuration*, qui permet de construire avec grande probabilité un code avec n'importe quel paramètre d'expansion et n'importe quel rendement [98, 10, 80]. La première construction déterministe de graphes expanseurs est basée sur des arguments algébriques de Margulis [77] et a été améliorée par Barg et Zémor [5, 7, 6]. Il est également possible de construire des graphes expanseurs avec la méthode du produit zig-zag [22]. Voir [56] pour une veille sur les graphes expanseurs.

Par définition, un *code expanseur quantique* est un $[[\Theta(n^2), \Theta(n^2), \Theta(n)]]$ code stabilisateur défini comme le produit d'hypergraphes d'un $[n, \Theta(n), \Theta(n)]$ code expanseur

classique avec lui-même [67]. Ref. [67] a également introduit le *décodeur small-set-flip* et a montré que n'importe quelle erreur de poids jusqu'à une fraction de la distance minimale est corrigée. La principale motivation pour étudier ces codes est le résultat de Daniel Gottesman qui montre que le calcul tolérant aux fautes avec coût constant en espace est possible [46]. Comme discuté au début de ce résumé, ce résultat est soumis à la conjecture qu'il existe des codes quantiques avec les propriétés appropriées. Grâce à [67], il est déjà connu que les codes expanseurs quantiques satisfont plusieurs de ces propriétés: il sont LDPC, ils ont un rendement constant et une bonne distance minimale. Le principal résultat de cette thèse est de montrer que les codes expanseurs quantiques satisfont aussi les autres propriétés nécessaires, et qu'ils peuvent donc être utilisés pour implémenter le schéma de tolérance aux fautes de Gottesman.

De par la construction produit d'hypergraphes, un code expanseur quantique est un code CSS associé à deux codes classiques $\mathcal{C}_X$ et $\mathcal{C}_Z$ construits à partir du code expanseur classique initial. Modulo une permutation des bits, le code $\mathcal{C}_X$ est égal à $\mathcal{C}_Z$. Ainsi, comme c'est le cas pour beaucoup de codes CSS, il est suffisant de décrire et d'analyser seulement le décodeur pour les erreurs de type bit-flip. Sous cette hypothèse, le support de l'erreur détermine l'erreur sur les qubits et donc nous supposerons souvent qu'une "erreur" et un "support d'erreur" sont les mêmes objets. Le décodeur small-set-flip est un algorithme de décodage à décisions dures. Cela qui signifie que son exécution est divisée en plusieurs tours et qu'à chaque tour, les qubits appartenant à un certain ensemble F sont flippés. L'implémentation de l'algorithme small-set-flip est assez simple: à chaque tour, un *petit ensemble* $F \in \mathcal{F}$ est sélectionné de telle manière que le poids du syndrome diminue suffisamment lorsque les qubits de F sont flippés (le poids du syndrome est le nombres de bits du syndrome égaux à 1).

1.6 Calcul quantique tolérant aux fautes

Le but du calcul tolérant aux fautes est de concevoir des circuits robustes contre le bruit [44]. Un circuit C est décrit par des fils (qui contiennent des états quantiques) et par des opérations élémentaires telles qu'une préparation d'état, une mesure ou une porte unitaire. On suppose souvent que C a une sortie classique $r \in \mathbb{F}_2^m$: pendant la dernière étape, tous les fils quantiques sont mesurés et un post-traitement classique est appliquée pour obtenir r. Dans cette thèse, une préparation d'état crée un état $|0\rangle$, une mesure correspond à la mesure d'une matrice de Pauli Z et les portes unitaires appartiennent à l'ensemble fini $\mathcal{G} := \{X, Z, H, S, T, C_X\}$ où:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

$$S = \begin{pmatrix} e^{-i\pi/4} & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \quad T = \begin{pmatrix} e^{-i\pi/8} & 0 \\ 0 & e^{i\pi/8} \end{pmatrix}, \quad C_X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Notons que $\mathcal{G}$ n'est pas minimal car $Z = S^2 = T^4$ et $X = H \circ Z \circ H$. Par contre, $\mathcal{G}$ est *universel* dans le sens où n'importe quelle unitaire peut être approchée par des portes appartenant à $\mathcal{G}$. Formellement, pour tout $\eta > 0$ et pour toute unitaire U, il existe une unitaire V construite à partir des portes de $\mathcal{G}$ telle que:

$$\left\| (U - V) \left| \psi \right\rangle \right\| \leq \eta \qquad \text{pour n'importe quel état pur } \left| \psi \right\rangle.$$

De plus, le théorème de Solovay-Kitaev [61] affirme que n'importe quel circuit fait de m portes C_X et de portes arbitraires sur un qubit peut être approché par $\mathcal{O}\big(m \log^2(m/\eta)\big)$ portes de $\mathcal{G}$.

Un *site* est un point du circuit où une erreur peut se produire. Par exemple, chaque préparation d'état, chaque mesure et chaque porte unitaire est un site. Le modèle de bruit que nous utiliserons est le modèle de bruit *stochastique local* de paramètre $p \in [0, 1]$. Soit L l'ensemble des sites d'un circuit arbitraire C. Alors l'erreur est représentée par une variable aléatoire $F \subseteq L$ appelée l'ensemble des *sites défaillants* qui satisfait pour tout $R \subseteq L$:

$$\mathbb{P}\Big[R \subseteq F\Big] \leq p^{|R|}.$$

Une fois que F a été choisi, un site de $L \setminus F$ se comporte normalement, mais un site défaillant $l \in F$ est remplacé par une application CPTP arbitraire dont les espaces d'entrée et de sortie sont les mêmes que pour l. Soit r la sortie de C dans le cas sans bruit $F = \varnothing$, alors le but du calcul tolérant aux fautes est, étant donné un paramètre $\epsilon > 0$, de concevoir un circuit C$'$ dont la sortie est r avec probabilité supérieure à $1 - \epsilon$ quand les sites sont soumis à un bruit stochastique local.

Dans ref. [1], le circuit tolérant aux fautes C$'$ est construit en utilisant la *concaténation* de codes. L'idée est de définir une fonction Φ qui envoie un circuit C sur un autre circuit $\Phi(\mathrm{C})$ qui est plus robuste contre le bruit. Finalement, le circuit C$'$ est défini par $\mathrm{C}' = \Phi^k(\mathrm{C})$ pour un $k \in \mathbb{N}$ bien choisi. Dans $\Phi(\mathrm{C})$, chaque fil de C est encodé avec N fils en utilisant un code stabilisateur $[[N, 1]]$. De plus, de la correction d'erreurs est régulièrement effectuée sur les fils de $\Phi(\mathrm{C})$. Dans ce protocole de tolérance aux fautes, les qubits sont encodés un par un ce qui résulte en un coût en temps et en espace polylogarithmique comme décrit dans eq. (1.1).

Dans ref. [46] où de la tolérance aux fautes avec coût constant en espace est atteinte, les fils de C sont divisés en blocs de $K \in \mathbb{N}$ fils et chaque bloc est encodé en utilisant un $[[N, K]]$ code stabilisateur $\mathcal{Q}$. Contrairement au cas de la concaténation de codes, la taille du bloc de $\mathcal{Q}$ dépend du nombre de fils dans C, et $\mathcal{Q}$ est choisi parmi une famille de codes LDPC quantiques avec rendement constant. Dans cette thèse, nous montrons que les codes expanseurs quantiques peuvent être utilisés dans cette construction pour corriger les erreurs qui apparaissent dans le circuit C$'$.

1.7 Résumé des contributions

Les auteurs de ref. [67] ont étudié comment le décodeur small-set-flip corrige un code expanseur quantique dans le cas d'erreurs adverses. L'inconvénient de ce cadre est

que la distance minimale du code est une limite fondamentale sur le poids de l'erreur que n'importe algorithme peut corriger. Lorsque l'erreur est générée avec un canal quantique (par exemple avec le canal dépolarisant), le poids de l'erreur est en général linéaire en la taille du bloc bien au delà de la meilleure distance minimale connue $\Theta\left(\sqrt{N}\sqrt[4]{\log(N)}\right)$ pour un code LDPC quantique [41]. Malgré cette borne supérieure, il est connu que certains codes LDPC quantiques ont un *seuil* et donc corrigent un bruit dépolarisant (voir [30, 17] pour des simulations numériques et [30, 66] pour des arguments théoriques). Dans cette thèse, nous montrons que le décodeur small-set-flip a un seuil pour n'importe quel modèle de bruit qui satisfait la propriété *stochastique locale* que nous avons défini dans la Section 1.3.

Kovalev et Pryadko ont déjà montré l'existence d'un seuil quand un bruit stochastique local est corrigé avec le décodeur de poids minimum [66]. Comme expliqué ci-dessous, nous avons montré dans ref. [38] que des techniques similaires peuvent être utilisées pour le décodeur small-set-flip. Soit $V_{\mathcal{Q}}$ l'ensemble des qubits et soit G_X le graphe de Tanner de $\mathcal{C}_X$. On dira qu'un ensemble de qubits est un *petit groupe* lorsque n'importe quelle erreur adverse dont le support est inclus dans ce groupe est corrigée par le décodeur. Par exemple pour les codes expanseurs quantiques, un "petit groupe" est un ensemble de qubits $K \subseteq V_{\mathcal{Q}}$ de taille $\mathcal{O}(\sqrt{N})$. Ici, nous utilisons le mot "groupe" comme un synonyme d'"ensemble" pour éviter la confusion entre "petit groupe" et "petit ensemble". De plus, pour une erreur initiale $E \subseteq V_{\mathcal{Q}}$, deux ensembles de qubits K_1, K_2 sont dits *indépendants* lorsque le comportement du décodeur sur l'erreur $K_1 \cap E$ ne dépend pas de l'erreur $K_2 \cap E$. Dans ref. [66], les qubits sont divisés en petits groupes indépendants de telle manière que le décodeur corrige les erreurs de chaque groupe et donc corrige l'erreur globale.

Afin de décomposer les qubits en petits groupes indépendants, nous dirons que le décodeur est *local* lorsque deux ensembles de qubits sont indépendants dès que l'intersection de leur voisinage dans G_X est vide (pour rappel, le voisinage d'un ensemble de qubits dans G_X est un ensemble de bits de parité). Par exemple, le décodeur small-set-flip et le décodeur de poids minimum sont locaux. On définit le *graphe d'adjacence* du code comme étant le graphe dont l'ensemble des sommets est $V_{\mathcal{Q}}$ et tel que deux qubits sont reliés si et seulement si ils partagent un bit de parité dans G_X. En particulier, si deux ensembles $K_1, K_2 \subseteq V_{\mathcal{Q}}$ ne sont pas adjacents dans le graphe d'adjacence alors aucun bit de parité ne peut être adjacent à la fois à K_1 et à K_2 dans G_X. Dans ce cas, K_1 et K_2 sont donc indépendants.

Lorsque le décodeur est utilisé, on appelle *erreur résiduelle* l'erreur physique qu'il reste sur les qubits après avoir appliqué la correction donnée par le décodeur. On définit aussi le *support d'exécution* comme étant l'ensemble de tous les qubits qui appartiennent au support de l'erreur à un moment quelconque de l'algorithme. Par exemple avec le décodeur small-set-flip, le support d'exécution contient les qubits de l'erreur initiale ainsi que les qubits de tous les petits ensembles qui ont été flippés par l'algorithme. Ensuite, on décompose les qubits en groupes indépendants $K_1, K_2, \ldots$ en définissant K_i comme étant les composantes connexes du support d'exécution dans le graphe d'adjacence. Finalement, on utilisera des arguments de percolation pour montrer que K_i est un petit groupe avec grande probabilité [48, 74, 58].

Afin de résumer les arguments ci-dessus, donnons une esquisse de la preuve de [38]. On

applique le décodeur small-set-flip sur une erreur générée par un bruit stochastique local. Soit K une composante connexe du support d'exécution U dans le graphe d'adjacence. La propriété de localité assure que la façon dont le décodeur agit sur K ne dépend pas du fait qu'il y ait des erreurs ou non en dehors de K. En particulier, l'erreur résiduelle sur les qubits de K est égale à l'erreur résiduelle que nous obtiendrions en utilisant le décodeur small-set-flip sans erreur initiale à l'extérieur de K. En utilisant des arguments venant de la théorie de la percolation, l'ensemble K satisfait $\left|K\right| = \mathcal{O}(\sqrt{N})$ avec grande probabilité et donc les erreurs initiales qui appartiennent à K sont corrigées par le décodeur small-set-flip. Ceci est valable pour chaque composante connexe K de U donc l'erreur globale est corrigée.

La théorie de la percolation est un domaine à part entière de la théorie des probabilités [48]. Dans ce travail, nous sommes intéressés par le cas particulier de la *percolation de sites* avec un bruit stochastique local sur un graphe fini de degré borné. Soit $\mathcal{V}$ l'ensemble des sommets d'un graphe fini $\mathcal{G}$ dont le degré maximum est $d_{\mathcal{G}}$ (les sommets sont appelés "sites" dans un contexte de percolation) et choisissons un sous-ensemble aléatoire de sommets $E \subseteq \mathcal{V}$. Alors la question centrale en théorie de la percolation est de comprendre comment se comporte la taille des composantes connexes de E. Dans le cas des codes correcteurs, le graphe $\mathcal{G}$ est le graphe d'adjacence et l'ensemble E représente le support de l'erreur. Habituellement pour la percolation, la loi de probabilité sur E est un *modèle iid* (chaque site est dans E avec probabilité p indépendamment des autres sites) mais dans ce travail, on s'intéresse au cas plus général d'un modèle de bruit *stochastique local*. Par définition, un modèle de bruit a la propriété stochastique locale quand l'ensemble aléatoire E satisfait eq. (1.4) pour tout $S \subseteq \mathcal{V}$. En suivant ref. [66], on peut étendre des résultats utiles de percolation à ce cas.

Une autre différence majeure avec les résultats standards de percolation est que nous réalisons un processus généralisé que nous appelons α-*percolation*. Au lieu de s'intéresser à la taille maximale des sous-ensembles connectés de E, le but de l'α-percolation est de regarder les α-sous-ensembles connectés. Un α-*sous-ensemble* est un ensemble $X \subseteq \mathcal{V}$ tel qu'au moins $\alpha|X|$ éléments de X appartiennent à E. Par exemple, un 1-sous-ensemble est simplement un sous-ensemble et la 1-percolation est la percolation dans le sens habituel. S'intéresser aux α-sous-ensembles est pertinent car le support d'exécution du décodeur small-set-flip est un α-sous-ensemble de l'erreur initiale pour un certain $\alpha \in (0, 1]$. Le principal théorème (qui a déjà été prouvé dans ref. [66] pour le cas particulier $\alpha = 1/2$) établit qu'avec grande probabilité, tout α-sous-ensemble connecté de E a pour taille $\mathcal{O}(\sqrt{|\mathcal{V}|})$. Si on revient sur la discussion à propos des décodeurs locaux, l'ensemble K (défini comme étant n'importe quel sous-ensemble du support d'exécution) est un α-sous-ensemble de E et donc l'égalité $\left|K\right| = \mathcal{O}(\sqrt{N})$ est vraie.

Pour un bruit iid et $\alpha = 1$, il est connu qu'avec grande probabilité, si $p < 1/(d_{\mathcal{G}} - 1)$ alors les composantes connexes de E ont pour taille $\mathcal{O}\big(\log(|\mathcal{V}|)\big)$. À l'inverse, si $p > 1/(d_{\mathcal{G}} - 1)$, il existe une unique composante dont la taille est linéaire appelée "composante géante" [58]. Dans ce travail, nous renforçons ce résultat pour n'importe quel bruit stochastique local et $\alpha \in (0, 1]$ arbitraire. Plus précisément, nous déterminons un réel non-nul $p_{\mathrm{th}} = p_{\mathrm{th}}(\alpha)$ tel que si $p < p_{\mathrm{th}}$, alors la probabilité qu'il existe

un α-sous-ensemble de E de taille au dessus de $t \in \mathbb{N}$ est au plus $\Theta(|\mathcal{V}|\beta^{\alpha t})$ où $\beta = p/p_{\text{th}} < 1$. Une valeur seuil détermine en général une transition de phase entre le régime où E a de petites composantes connexes et le régime où E a une composante géante unique. Toutefois pour la correction d'erreurs, la préoccupation est d'être en dessous du seuil et donc p_{th} est appelé un seuil même s'il s'agit seulement d'une borne inférieure sur le seuil de percolation. En particulier, la valeur p_{th} que nous obtenons n'est pas optimale puisque $p_{\text{th}}(1) < 1/(d_{\mathcal{G}} - 1)$. Il serait intéressant de trouver le véritable seuil pour $\alpha < 1$ et d'étendre les autres résultats venant de la théorie de la percolation sur graphes finis [58].

Une des hypothèses délicates pour appliquer le schéma de tolérance aux fautes de Gottesman est de montrer que les codes expanseurs quantiques permettent de gérer des erreurs sur le syndrome. Formellement dans ce modèle d'erreurs, un ensemble de bits D est aléatoirement choisi suivant un modèle de bruit stochastique local et l'entrée de l'algorithme small-set-flip est le syndrome où les bits de D ont été flippés. Cette hypothèse est nécessaire dans le contexte du calcul quantique tolérant aux fautes, parce que les bits du syndrome sont produits par une mesure quantique effectuée à l'aide de composants physiques bruités. Toutefois à cause la propriété LDPC, il n'est pas possible que le décodeur corrige entièrement les qubits lorsque le syndrome est bruité. À la place, nous montrons que l'erreur résiduelle est équivalente à une erreur stochastique locale avec un paramètre sous contrôle.

De manière similaire au cas où le syndrome est parfait, la première étape de l'analyse est de considérer des erreurs adverses dont le poids est en dessous de la distance minimale. Dans ce cas, nous montrons que le poids de l'erreur résiduelle est bornée supérieurement par une fonction linéaire en $|D|$. La stratégie de preuve consiste à retourner au cas où la mesure du syndrome est non bruité et de montrer que l'algorithme small-set-flip peut flipper plusieurs petits ensembles à chaque tour. Comme nous en discuterons plus tard, cette propriété implique également que le décodeur peut être parallélisé dans le cas d'un syndrome bruité ou non bruité. Finalement, afin de traiter des erreurs stochastiques locales, nous appliquerons un processus d'α-percolation sur le *graphe d'adjacence du syndrome* pour réduire le problème au cas d'erreurs adverses. Le graphe d'adjacence du syndrome est le graphe de Tanner de $\mathcal{C}_X$ avec des arrêtes additionnelles entre les qubits reliés dans le graphe d'adjacence [46].

De plus, notre analyse montre que l'algorithme de small-set-flip a la propriété d'être *single-shot*. Un code quantique est dit single-shot quand un unique tour de mesures bruitées du syndrome est suffisant pour que le décodeur ait un seuil. Par comparaison, le code torique n'est pas single-shot, ce qui signifie que le syndrome doit être mesuré $\Theta(D)$ fois afin d'obtenir suffisamment d'information pour corriger l'erreur. Cette propriété est très avantageuse dans le contexte de la tolérance aux fautes où de nombreuses étapes de corrections d'erreurs sont nécessaires.

La correction d'erreurs single-shot a été introduite par Hector Bombin dans [12]. Plusieurs familles de codes ont cette propriété: les codes couleurs de jauge en dimension trois [12], le code torique en dimension quatre [32] et les codes hyperboliques de dimension quatre [55]. Une théorie de la correction d'erreurs single-shot a été proposée dans ref. [21] où le principe est de corriger les erreurs de syndrome avant de corriger

les erreurs sur les qubits. Par [21] et les résultats de cette thèse, la propriété single-shot semble étroitement liée à la propriété de *robustesse*. Informellement, un code est robuste si en dessous de la distance minimale, le poids du syndrome peut être borné inférieurement par une fonction linéaire du poids de l'erreur.

Dans ref. [37], nous avons également montré que l'algorithme de small-set-flip peut être parallélisé pour fonctionner en profondeur constante. Lorsque le syndrome est bruité, si le nombre de tours est choisi suffisamment grand (mais constant) alors l'erreur résiduelle sera équivalente à une erreur stochastique locale avec petit paramètre. D'autre part, lorsque le syndrome est parfait, un nombre fixé de tours n'est pas suffisant pour corriger entièrement l'erreur. En revanche, lancer l'algorithme avec un nombre de tours logarithmique en la taille du syndrome corrigera l'erreur avec grande probabilité.

Dans ref. [49], nous avons fait des simulations pour étudier comment le décodeur small-set-flip se comporte en pratique. Pour simplifier, nous nous sommes concentrés sur l'algorithme séquentiel avec des mesures de syndromes parfaites. Les résultats numériques que nous avons obtenus sont prometteurs. Par exemple, la valeur du seuil que nous avons obtenu est largement au dessus de la borne inférieure assurée par les arguments théoriques. Le seuil est d'environ 4.5% pour une famille de codes produits d'hypergraphes avec rendement $1/61$ et d'environ 2% pour un rendement $1/5$.

1.8 Conclusion

Dans cette thèse, nous montrons que les codes expanseurs quantiques et le décodeur small-set-flip peuvent être utilisés dans la construction de ref. [46] pour réaliser du calcul quantique tolérant aux fautes avec coût constant en espace. Le décodeur small-set-flip peut être parallélisé pour fonctionner en temps constant et, par conséquent, chaque site du circuit tolérant aux fautes obtenu nécessite d'utiliser un circuit classique avec profondeur constante. Par comparaison, il n'est pas connu si la complexité temporelle nécessaire pour décoder les autres familles de codes est constante. Ainsi, pour ces autres familles de codes, chaque étape du circuit tolérant aux fautes contient un circuit classique dont la profondeur augmente avec le nombres de qubits.

Comme travail futur, il serait intéressant d'essayer de réduire le coût en temps de la construction présentée dans ref. [46]. En particulier, le coût en temps est large si le circuit initial est parallèle parce qu'il doit être transformé en un circuit séquentiel pour que la construction fonctionne.

Pour les codes expanseurs quantiques et plus généralement les codes produits d'hypergraphes, il est essentiel de trouver des décodeurs plus rapides avec de meilleures performances. Par exemple, en utilisant des décodeurs basés sur le small-set-flip ou le décodeur itératif (voir [85]). De plus, les simulations de ref. [49] nécessitent des codes avec une large taille de bloc et il serait intéressant d'étudier comment les codes produits d'hypergraphes modifiés de ref. [65, 78] se comportent en pratique. Finalement, des simulations numériques pour le décodage small-set-flip parallèle et pour le cas d'un syndrome bruité seraient intéressantes également.

Chapter 2

Introduction (English)

Quantum computers are physical objects taking advantage of the laws of quantum physics to perform some computational tasks faster than the best classical algorithms. For instance, Shor showed in ref. [97] how this technology would provide an exponential speedup for the integer factorization problem compared to the existing classical algorithms. Many further examples can be found in [25] such as Grover's algorithm leading to a quadratic speedup for unstructured search [50] or the HHL algorithm for solving linear systems [53]. On a practical point of view, one of the short-term targets is to reach the quantum supremacy by building a quantum device which would perform a specific task faster than the current super-computers [11, 54, 13]. In short, a quantum circuit uses *quantum gates* acting on *quantum states* which store quantum information whose basic unit is called a *qubit* (abbreviation for quantum bit).

Today, several architectures compete with each other such as quantum circuits based on superconductors or trapped ions, but despite many efforts and interesting progress, quantum computers are still in the experimental stage [60, 4, 75, 31, 114, 23, 33]. The main problem is that a quantum system is unavoidably subjected to noise and decoherence. Thus, quantum states are fragile and an active protection is required to store and process quantum information. Quantum supremacy could be achieved with noisy intermediate-scale quantum technology where the goal is to build devices with small enough noise to perform a non trivial computation out of reach for classical computers [87]. However, this strategy is not sustainable for time-consuming computations and contrarily to the case of classical hardwares, there is no hope to remove entirely the physical noise affecting quantum circuits. Hopefully, given a quantum circuit C, it is possible to design another circuit C' called a *fault-tolerant circuit*, which executes the same computation than C and works even though its basic components are noisy.

To formalize the idea of noise, a quantum circuit is split into elementary steps called *locations* that represent the spots where an error could occur. For example, a *gate location* represents a point in the circuit where a gate is applied and a *wait location* refers to a qubit waiting while some gates are applied on other locations. Since the components we use for building the circuit are noisy, some of the locations are *faulty* meaning that the action of this location is not the expected one. The simplest noise model

is the *iid* error model (independent and identically distributed) where the locations are faulty with some probability p independently. The fundamental result in fault-tolerant quantum computation is called the *threshold theorem* and has been proven by Aharonov and Ben-Or in ref. [1]. They showed that if the locations are subjected to an iid error model with parameter $p < p_{\text{th}}$ (p_{th} is a universal constant called the *threshold*), then for any quantum circuit C and for any target probability $\epsilon > 0$, there exists a fault-tolerant circuit simulating C and failing with probability at most ϵ. Beyond the iid error model, this PhD thesis is interested in the more general *local stochastic error model* [44, 46]. In addition, we would like to point out that fault-tolerance is also possible with other error models [1, 2, 105].

There are two natural ways to measure the efficiency of a fault-tolerant protocol, namely the *time overhead* and the *space overhead*. Let C be a quantum circuit and let C′ be the corresponding fault-tolerant circuit. We assume that C (resp. C′) has $|\mathrm{C}|$ locations (resp. $|\mathrm{C}'|$ locations), uses m qubits (resp. m' qubits) and requires t time steps to run (resp. t' time steps to run). Then, the time overhead is defined to be the ratio t'/t and the space overhead to be m'/m. For instance, with the usual threshold theorem of ref. [1], the time and space overheads are polylogarithmic in $|\mathrm{C}|$:

$$\frac{|\mathrm{C}'|}{|\mathrm{C}|} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right), \quad \frac{t'}{t} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right), \quad \frac{m'}{m} = \mathrm{polylog}\left(\frac{|\mathrm{C}|}{\epsilon}\right). \quad (2.1)$$

The main motivation of this PhD is ref. [46] where constant space overhead is proven to be achievable using well chosen *quantum error correcting codes*. Starting with a k-qubit state, the idea of quantum error correction is to encode it within $n > k$ qubits, adding some redundancy in such a way that an error on a small number of qubits is not harmful to recover the initial state. The procedure used to recover the initial state is called *decoder* and the time consumption of this classical algorithm is a bottleneck. Indeed, for each location of C, the fault-tolerant circuit provided by [46] requires to run the decoder. In this PhD thesis, we consider the so-called *quantum expander codes* showing that the associated *small-set-flip decoder* can be parallelized to run in constant time. Following ref. [46], we get a fault-tolerant protocol with constant space overhead. In addition, each time step of the resulting quantum circuit requires to run a constant depth classical circuit (this is a reasonable assumption if classical computation is fast enough).

The goal of this introduction is to describe and explain our result. Among other things, we will need to give some terminology from classical error correction and discuss about the concepts of hypergraph product codes and classical expander codes.

2.1 Classical error correction

The field of computer science which studies error correction is called *Coding Theory* and starts with Claude Shannon's work in 1948 [94]. Shannon showed that a transmission task can be decomposed into two steps: *source coding* and *channel coding*. Source coding is used for data compression [111, 26], however we do not discuss it in this PhD thesis. Instead, we focus on the channel coding step where error correcting codes are

used. In [94], Shannon defined the notion of *capacity* which is the maximum rate at which information can be transmitted reliably through a given noisy channel. He also proved that the capacity of any channel is attained by random error correcting codes which, unfortunately, cannot be implemented efficiently. At the same time as these breakthrough results were published, Richard Hamming introduced the first practical class of error correcting codes now called *the Hamming code* [52].

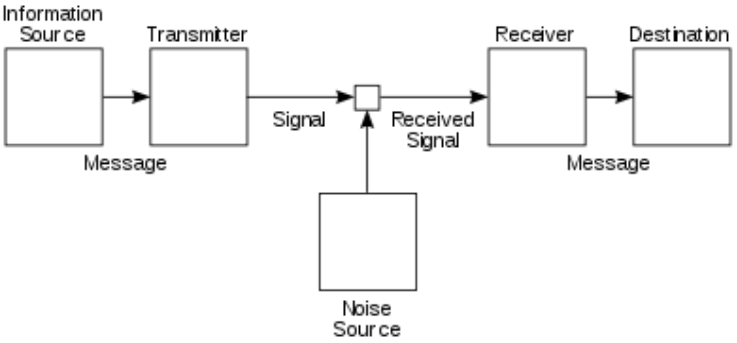

Figure 2.1: Shannon's diagram for communication [94].

Error correcting codes are often presented with the communication scenario where two protagonists, generally called Alice and Bob, would like to communicate. Classical information is represented as a sequence of *bits* (0 and 1) and hence a message of k bits is a binary vector $s \in \mathbb{F}_2^k = \{0,1\}^k$ (s stands for source message). Alice and Bob are far apart and the only communication channel they have access to is noisy, meaning that some of the bits that Alice sends to Bob are corrupted by the channel. When Alice sends a message $x \in \mathbb{F}_2^n$ through the noisy channel, Bob gets as output another message $y \in \mathbb{F}_2^n$ accordingly to a given probability distribution depending on x. Two important channel models are the *binary erasure channel* and the *binary symmetric channel*. Both of them take as input a single bit. For the binary erasure channel with parameter $p \in [0,1]$, the bit that Alice sends is correctly transmitted with probability $1-p$ and is erased with probability p. When Bob receives 0 or 1 he knows that the bit is correct and when the bit is erased, he receives the symbol '?'. For the binary symmetric channel, each transmitted bit is *flipped* ($0 \mapsto 1$ and $1 \mapsto 0$) with probability p and is correct with probability $1-p$. With a single use of these channels, Alice can send one bit to Bob but she can also use it $n \in \mathbb{N}$ times in a row to send n bits. In that case, we assume that the n uses of the channel are independent in term of probabilities.

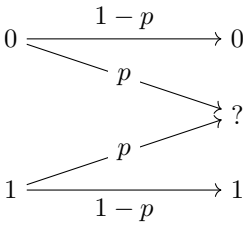

Figure 2.2: the binary erasure channel.

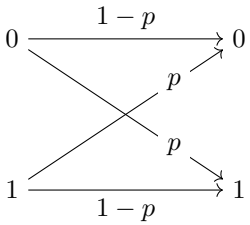

Figure 2.3: the binary symmetric channel.

In what follows we will be mainly interested in the case of the binary symmetric channel. Alice's goal is to send an arbitrary message $s \in \mathbb{F}_2^k$ to Bob in such a way that he can recover s with good probability. Obviously, since any message $s \in \mathbb{F}_2^k$ could be sent, there is no way for Bob to correct a single bit-flip under the assumption that Alice sends s directly through the channel. Instead, she can send $x \in \mathbb{F}_2^n$ an encoded version of s chosen in such a way that Bob can recover s even though the channel flipped a few bits of x. In particular, the bits of x must contain some redundancy and thus $n > k$ must hold. An error correcting code $\mathcal{C} \subseteq \mathbb{F}_2^n$ satisfies $\left|\mathcal{C}\right| = \left|\mathbb{F}_2^k\right| = 2^k$ and is defined as the set of possible bit-strings x, called the *codewords*, that Alice can send to Bob.

An error correcting code $\mathcal{C}$ is said to be an $[n, k]$ *linear code* when the set of codewords is a k dimensional subspace of $\mathbb{F}_2^n$. From now on, the codes we consider are linear. The *minimal distance* of a code is the minimum weight of a non-zero codeword. When the minimal distance d is known, the code is said to be an $[n, k, d]$ linear code. The bits of s are called the *logical bits*, the bits of x are called the *physical bits* and the number of physical bits n is called the *block length*.

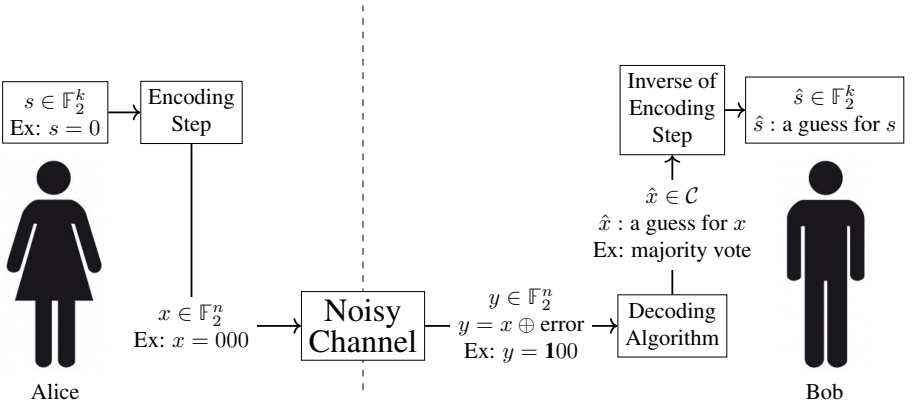

Figure 2.4: classical communication with error correcting codes.
For instance, Alice sends the bit $s \in \mathbb{F}_2$ three times to Bob who can recover it using a majority vote (this is called the 3-bit repetition code).

Figure 2.4 presents the protocol based on error correcting codes used by Alice and

Bob to communicate with a noisy channel:

- Alice encodes the message s into a codeword x and sends x to Bob through the noisy channel.
- Bob gets y as output of the channel and tries to infer the codeword x that Alice sent, its guess is denoted by $\hat{x}$.
- Bob decodes $\hat{x}$ to a message $\hat{s}$ by applying the inverse of the function Alice applied during the encoding step.

The above protocol is a success when $s = \hat{s}$ and a failure when $s \neq \hat{s}$. In this PhD thesis, we are interested in linear codes, for which the first and the third step can be done easily using linear algebra. Thus, we will mainly focus on the second step, called the *error correction step* or the *decoding step*, where Bob infers $\hat{x}$ from y using an algorithm called *decoding algorithm* or *decoder*. One of the main goals in coding theory is to find families of error correcting codes with large ratio k/n and which admit an efficient time decoder with low failure probability.

One breakthrough result of ref. [94] is the possibility to use a noisy channel to transmit information reliably at constant rate. Instead of using a single error correcting code, consider a family of codes $\mathcal{C}_1, \mathcal{C}_2, \ldots$ where $\mathcal{C}_i$ encodes k_i logical bits with n_i physical bits. When the ratio k_i/n_i converges and n_i goes to infinity, the limit $r \in [0, 1]$ is called the *asymptotic rate* of the family. In addition, the *threshold* of $\mathcal{C}_i$ is the maximum real number $p \in [0, 1]$ such that the probability of failure vanishes as i goes to infinity when using a binary symmetric channel with parameter p. We say that $\mathcal{C}_i$ allows for reliable transmission over the binary symmetric channel when the parameter p is below the threshold of the code family. A binary symmetric channel with fixed parameter $p \in (0, 1)$ cannot transmit information at rate arbitrarily close to 1. Indeed there is a limit called the *capacity* of the channel which is an upper bound on the asymptotic rate of any family of code allowing for reliable transmission. The notion of capacity was introduced by Shannon in [94] and is equal to $C_{\text{BEC}}(p) := 1 - p$ for the binary erasure channel and to $C_{\text{BSC}}(p) := 1 - h_2(p)$ for the binary symmetric channel where $h_2(p) := -p \log_2(p) - (1 - p) \log_2(1 - p)$ is the binary entropy function.

In his thesis of 1962 [42], Robert Gallager introduced the class of LDPC codes (Low Density Parity Check codes) and suggested to decode them with an algorithm called *belief propagation decoder* (also called *iterative decoder* or *message passing algorithm* or *probabilistic decoder*). However at this point, computers were not powerful enough to implement the belief propagation decoder and thus LDPC codes were not of practical interest. In the late 90's, there has been a renewed interest in LDPC codes with the papers [70, 71, 73, 72]. Nowadays they are intensively used in the communication networks [91, 95].
In order to define LDPC codes, we need to explain the concepts of *parity check matrices* and *Tanner graphs*. Let $\mathcal{C}$ be a linear code and H be a binary matrix such that $\mathcal{C} = \ker(H)$. Then H is said to be a *parity check matrix* for $\mathcal{C}$. A *Tanner graph* G for $\mathcal{C}$ is a bipartite graph built from H in the following way: the left vertices of G represent the columns of H, the right vertices represent the rows of H and two vertices are linked if

and only if the corresponding entry in H is equal to 1 [104]. By definition, a family of codes is LDPC if and only if the degrees in the Tanner graphs are upper bounded by a uniform constant.

In the early 90's, the *turbo codes* of Berrou, Glavieux and Thitimajshima [9] made the scientific community aware that some error correcting codes with high success probability and whose rate approaches channel capacity can be implemented in practice. The turbo codes use as building blocks the family of *convolutional codes* [59] introduced by Elias in [36]. A convolutional code encodes a bit-stream into another bit-stream. In this PhD thesis we are interested in *block codes* such as LDPC codes which, by opposition to a convolutional code, encode a block of bits into a larger block of bits (a block is a bit-string of fixed length).

For simplicity, the classical codes we will use are *regular LDPC codes* as defined by Gallager in his thesis [42]. By definition, a regular code is such that the left vertices of the Tanner graph have degree d_V and the right vertices have degree d_C where $d_V, d_C \in \mathbb{N}$ are fixed integers. Note however, that the performance of regular LDPC codes is not as good as the performance of other families of LDPC codes. For example, ref. [83] shows that no family of regular codes achieves the capacity of the binary erasure channel. It has also been demonstrated that irregular LDPC codes lead to better thresholds [72, 73, 71, 70] and better finite-length performance [90, 110, 113, 115]. More generally, many families of codes have been introduced with the objective to increase the success probability and to speed up the encoding and decoding steps. See for example the concatenated codes [40], the algebraic Reed-Solomon codes [103, 64, 89] and more recently the spatially-coupled LDPC codes [39], the polar codes [3] and the irregular LDPC codes based on protographs [106]. In addition, there exist many algorithmic techniques for optimizing LDPC codes [24, 93, 57, 107].

For more details about classical error correction, see [111, 92, 59].

2.2 Quantum information

This section introduces the fundamental concepts of quantum information theory we will need to explain our results. See [82] for more details about quantum information and quantum computation.

A *pure state* on n qubits $|\psi\rangle \in \mathbb{C}^{2^n}$ ("ket psi") is mathematically defined by a complex vector normalized for the Euclidean norm with 2^n entries. It is convenient to see $|\psi\rangle$ as an element of $\left(\mathbb{C}^2\right)^{\otimes n}$ where all along this manuscript, $\otimes$ is the *tensor product* (or *Kronecker product*). The row vector $\langle\psi| := |\psi\rangle^\dagger$ ("bra psi") is defined to be the transpose conjugate of $|\psi\rangle$. For $i \in [\![0; 2^n - 1]\!]$, the state $|i\rangle$ is the column vector whose only non-zero entry is a 1 at i^{th} position.

Dynamics of pure quantum states is split into two kinds of operations: *unitary evolution* and *measurements*. A unitary evolution has the form $U : |\psi\rangle \mapsto U|\psi\rangle$ where U is any complex unitary matrix of dimension $2^n \times 2^n$. Among the unitary matrices,

the *Pauli group* is of particular interest. It is defined by:

$$\mathcal{P}_n := \left\{ \alpha P : \alpha \in \left\{1, -1, i, -i\right\} \text{ and } P \in \left\{\mathbb{1}, X, Y, Z\right\}^{\otimes n} \right\}$$

where X is called the *bit-flip* and Z is called the *phase-flip*:

$$X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Y := iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

For a Pauli error $P \in \mathcal{P}_n$, the *weight* of P denoted by $|P|$ is defined to be the number of qubits on which P acts non trivially. Formally, if we write $P = \alpha P_1 \otimes \ldots \otimes P_n$ where $\alpha \in \{1, -1, i, -i\}$ is a global phase and $P_1, \ldots, P_n \in \{\mathbb{1}, X, Y, Z\}$ are one-qubit Pauli matrices, then $|P|$ is defined by:

$$|P| := \#\big\{i \in [\![1; n]\!] : P_i \neq \mathbb{1}\big\}.$$

A matrix Π is said to be an *orthogonal projector* when $\Pi^2 = \Pi$ and $\Pi = \Pi^\dagger$ where $\Pi^\dagger$ is the Hermitian adjoint of Π (the Hermitian adjoint is the transpose conjugate matrix).
In general, the measurement of a pure state $|\psi\rangle \in \mathbb{C}^{2^n}$ outputs a classical bit $b \in \mathbb{F}_2$ and modifies $|\psi\rangle$ (we say that $|\psi\rangle$ *collapses* to another quantum state). For error correction, we are interested in *projective measurements* defined with two orthogonal projectors Π_0, Π_1. These projectors must be of size $2^n \times 2^n$ and must satisfy $\Pi_0 + \Pi_1 = \mathbb{1}$ where $\mathbb{1}$ is the identity matrix.

- With probability $\langle\psi| \Pi_0 |\psi\rangle$: the measurement output is $b = 0$ and the state collapses to $\dfrac{\Pi_0 |\psi\rangle}{\sqrt{\langle\psi| \Pi_0 |\psi\rangle}}$.
- With probability $\langle\psi| \Pi_1 |\psi\rangle$: the measurement output is $b = 1$ and the state collapses to $\dfrac{\Pi_1 |\psi\rangle}{\sqrt{\langle\psi| \Pi_1 |\psi\rangle}}$.

In particular, the *Pauli measurement* associated to a Pauli operator $P \in \mathcal{P}_n$ is the projective measurement defined with $\Pi_0 := (\mathbb{1} + P)/2$ (the orthogonal projector onto the $+1$ eigenspace of P) and $\Pi_1 := (\mathbb{1} - P)/2$ (the orthogonal projector onto the -1 eigenspace of P).

In general, a quantum system S is not isolated in the sense that it interacts with other systems called *environment*. In that case, the state of S is not necessarily a pure state but can always be seen as a probabilistic mixture of pure states. A *density matrix* is a convenient way to represent such a mixture as a $2^n \times 2^n$ positive semi-definite operator with complex coefficients and trace 1. For example, the density matrix associated to a pure state $|\psi\rangle$ is $|\psi\rangle\langle\psi|$ where $\langle\psi| = |\psi\rangle^\dagger$ as previously. More generally, let $|\psi_1\rangle, \ldots, |\psi_m\rangle$ be pure states and let $p_1, \ldots, p_m \in [0, 1]$ be probabilities such that

$p_1 + \ldots + p_m = 1$. Then, the density matrix of a system where each state $|\psi_i\rangle$ has been prepared with probability p_i is:

$$\rho = \sum_{i=1}^{m} p_i \left|\psi_i\right\rangle \left\langle\psi_i\right| . \tag{2.2}$$

During the interaction of a quantum system S with an environment E, the number of qubits in S can change. Indeed, some of the qubits of S can be discarded and some qubits of E can be added to S. Formally, any valid evolution of an n-qubit system to an n'-qubit system is represented by a *CPTP map* $\mathcal{E}$ (completely positive trace preserving map) sometimes called a *quantum channel* or a *quantum operation*. A CPTP map is defined by a set of $2^{n'} \times 2^n$ complex matrices E_k called *Kraus operators* and satisfying $\sum_k E_k^\dagger E_k = \mathbb{1}$. $\mathcal{E}$ is then equal to:

$$\mathcal{E} : \rho \mapsto \sum_{k=1}^{K} E_k \rho E_k^\dagger .$$

For instance, a unitary evolution U is represented by the single Kraus operator $E_1 = U$ where $n = n'$.

As stated above, the measurement of a Pauli operator $P \in \mathcal{P}_n$ on a pure state $|\psi\rangle$ is the projective measurement associated to $\Pi_0 := (\mathbb{1} + P)/2$ and $\Pi_1 := (\mathbb{1} - P)/2$. Let $\mathcal{M}$ be the CPTP map representing this measurement and let:

$$p_0 := \langle\psi| \Pi_0 |\psi\rangle , \qquad |\psi_0\rangle := \frac{\Pi_0 |\psi\rangle}{\sqrt{\langle\psi| \Pi_0 |\psi\rangle}},$$
$$p_1 := \langle\psi| \Pi_1 |\psi\rangle , \qquad |\psi_1\rangle := \frac{\Pi_1 |\psi\rangle}{\sqrt{\langle\psi| \Pi_1 |\psi\rangle}}.$$

Then $\mathcal{M}$ is equivalent to prepare the state $|\psi_0\rangle$ with probability p_0 and the state $|\psi_1\rangle$ with probability p_1. Using eq. (2.2), we have:

$$\mathcal{M} : |\psi\rangle\langle\psi| \mapsto |0\rangle\langle 0| \otimes \Pi_0 |\psi\rangle\langle\psi| \Pi_0 + |1\rangle\langle 1| \otimes \Pi_1 |\psi\rangle\langle\psi| \Pi_1 .$$

Note that for simplicity, the classical outcome of the measurement $b \in \{0, 1\}$ is stored as a qubit. More generally, for any density matrix ρ we have:

$$\mathcal{M} : \rho \mapsto |0\rangle\langle 0| \otimes \Pi_0 \rho \Pi_0 + |1\rangle\langle 1| \otimes \Pi_1 \rho \Pi_1 . \tag{2.3}$$

The Kraus operators of $\mathcal{M}$ are $E_0 = |0\rangle \otimes \Pi_0$ and $E_1 = |1\rangle \otimes \Pi_1$ with $n' = n + 1$.

2.3 Quantum error correction

In Section 2.1, we used the communication scenario presented in Figure 2.4 to introduce classical error correcting codes. Similarly, a *quantum error correcting code* can be used in a protocol where Alice wishes to send a quantum state $|\varphi\rangle$ with K qubits through a

noisy quantum channel to Bob. The basic idea is the same as in the classical setting: she starts by encoding $|\varphi\rangle$ redundantly in a state $|\psi\rangle$ with N qubits, then she sends $|\psi\rangle$ through the noisy channel and finally Bob applies a quantum operation to correct the received mixed state ρ. The communication is a success when the state $|\hat{\varphi}\rangle$ that Bob gets at the end is equal to $|\varphi\rangle$. In this protocol, each state $|\varphi\rangle$ is mapped to its encoded version $|\psi\rangle$ called a *code state*. The set of code states denoted by $\mathcal{Q}$ is called the *code space* or a *quantum error correcting code*. By definition, the encoding is a linear map from $\left(\mathbb{C}^2\right)^{\otimes K}$ to $\left(\mathbb{C}^2\right)^{\otimes N}$ and thus the set $\mathcal{Q}$ is a 2^K-dimensional linear subspace of $\left(\mathbb{C}^2\right)^{\otimes N}$. The integer K is called the number of *logical qubits* and the integer N is called the number of *physical qubits*.

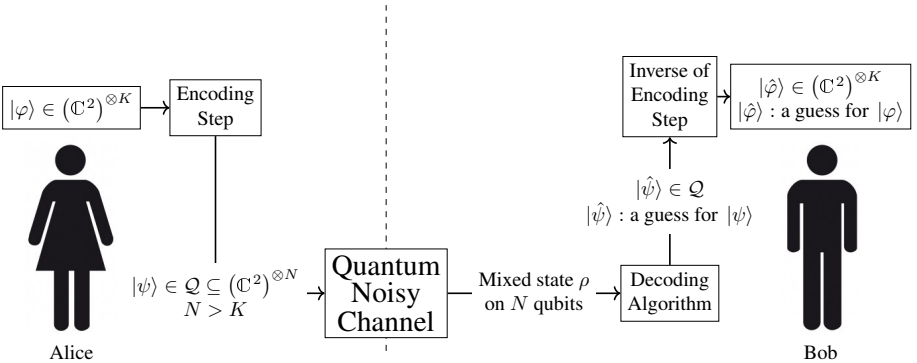

Figure 2.5: communication scenario for quantum error correcting codes.

In this work, we are interested in quantum channels satisfying the *local stochastic* property with parameter $p \in [0, 1]$. A noise model is local stochastic when the probability for a given set of qubits S to be in the support of the error decays exponentially in $|S|$. Formally, let $V_{\mathcal{Q}}$ be the set of qubits and let $\mathcal{E}$ be a quantum channel. Then:

- $\mathcal{E}$ has the *stochastic* property if $\mathcal{E}$ can be described by the following two steps process:
  - In the first step, a random set $E \subseteq V_{\mathcal{Q}}$ called the *support* of the error is randomly chosen.
  - In the second step, a quantum channel $\mathcal{E}_E$ is applied on the qubits. Here, for each $F \subseteq V_{\mathcal{Q}}$, $\mathcal{E}_F$ is a CPTP map acting on the qubits of F.
- $\mathcal{E}$ is *local stochastic* with parameter $p \in [0, 1]$ if it has the stochastic property and the probability distribution of E satisfies for any $S \subseteq V_{\mathcal{Q}}$:

$$\mathbb{P}\Big[S \subseteq E\Big] \leq p^{|S|}. \tag{2.4}$$

In other words, a local stochastic noise $\mathcal{E}$ is such that:

$$\mathcal{E} : \rho \mapsto \sum_{F \subseteq V_{\mathcal{Q}}} \mathbb{P}\Big[E = F\Big] \mathcal{E}_F(\rho),$$

where the support $E \subseteq V_{\mathcal{Q}}$ is randomly chosen with eq. (2.4) holding for any $S \subseteq V_{\mathcal{Q}}$.

For instance, the channel applying independent and identically distributed bit-flip and phase-flip errors has the local stochastic property. Another example is the *depolarizing channel* which applies the channel $\mathcal{D}$ defined below on each qubit independently:

$$\mathcal{D} : \rho \mapsto (1-p)\rho + \frac{p}{3}\Big(X\rho X + Y\rho Y + Z\rho Z\Big).$$

The depolarizing channel for n qubits is equal to $\mathcal{D}^{\otimes n}$ and is local stochastic with parameter $p/3$.

When we will talk about the *success probability* (or the probability for the error to be corrected) for a local stochastic error model, we refer to the probability on E for the communication to be a success:

$$\mathbb{P}\Big[\text{Success}\Big] = \sum_{F \subseteq V_{\mathcal{Q}}} \mathbb{P}\Big[E = F\Big]\mathbb{P}\Big[\, |\hat{\varphi}\rangle = |\varphi\rangle \;\Big|\; E = F\Big].$$

Note that when the decoding algorithm is deterministic, we have $\mathbb{P}\Big[\, |\hat{\varphi}\rangle = |\varphi\rangle \;\Big|\; E = F\Big] \in \{0, 1\}$.

In the classical case, we focused on the bit-flip error model which is quite realistic. However, in the quantum setting the set of possible errors is infinite: any unitary or even any CPTP map could potentially happen on the quantum state we try to protect. Surprisingly, it is sufficient to correct the finite set of Pauli errors, to be able to correct general errors (see for example [8] or [82]). For instance, if a given error correcting procedure is able to correct any Pauli error acting on the qubits of $F \subseteq V_{\mathcal{Q}}$, then the same procedure corrects an arbitrary CPTP map acting on F. As a consequence, for a local stochastic error model, we can assume without loss of generality that the CPTP maps $\mathcal{E}_F$ are Pauli channels (*i.e.* apply Pauli errors on the qubits of F).

In the '90s, the question to know whether quantum error correction was possible was discussed until Shor [96] and Steane [102] proved it is indeed feasible. Just after these seminal works, the theory of quantum error correction was developed, for example in [63, 20, 101, 43]. In particular in [43], Gottesman introduced the mathematical tools used to define and to study a wide variety of quantum codes called *stabilizer codes*. A stabilizer code $\mathcal{Q}$ on N physical qubits is defined from a finite set of commuting Pauli operators $g_1, \ldots, g_M \in \mathcal{P}_N$. These operators are called *stabilizer generators* and $\mathcal{Q}$ is defined in the following way:

$$\mathcal{Q} := \Big\{\, |\psi\rangle \in \big(\mathbb{C}^2\big)^{\otimes N} : g_1 |\psi\rangle = \ldots = g_M |\psi\rangle = |\psi\rangle \,\Big\}.$$

The code $\mathcal{Q}$ is said to be an $[[N; K]]$ stabilizer code where K is the number of logical qubits.

In this thesis, we focus on a particular class of stabilizer codes called CSS LDPC codes. The terminology "CSS" stands for Calderbank, Shor and Steane who introduced the CSS construction in 1996 [20, 102]. A CSS code is constructed from two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ with $\mathcal{C}_Z^{\perp} \subseteq \mathcal{C}_X$. The code $\mathcal{C}_X$ is used to correct bit-flip errors and $\mathcal{C}_Z$

is used to correct phase-flip errors. The first step of the error correction procedure is to measure the stabilizer generators. The output of the measurement is a bit-string called *syndrome*. From a practical perspective, if we want to be able to measure the stabilizer generators, it is important that each one involves a small number of qubits and each qubit is involved in a small number of operators. These properties hold when the two classical codes used to construct the CSS code are LDPC and in that case the code is said to be a CSS LDPC code.

The minimal distance of a stabilizer code is the minimal weight of a Pauli error which maps a code state to another orthogonal code state. This quantity is a good indication of the performance of the code and it not known whether CSS LDPC codes with minimal distance $D = \Theta(N)$ exist. Without the LDPC constraint, this question was resolved in the affirmative when the CSS codes were introduced [20, 102]. For LDPC codes, the best known minimal distance is $D = \Theta(\sqrt{N}\ \sqrt[4]{\log(N)})$ for a family of codes with $K = 1$ logical qubit whose construction is based on an 4-dimensional manifold [41]. With the additional constraint of constant rate $K = \Theta(N)$, the best known minimal distance is $D = \Theta(\sqrt{N})$ and is achieved by the hypergraph product codes of Tillich and Zémor [108]. When the minimal distance D of an $[[N; K]]$ stabilizer code is known, the code is said to be an $[[N; K; D]]$ stabilizer code.

A *decoder* or *decoding algorithm* for a stabilizer code is a classical algorithm taking the syndrome as input and outputting a Pauli error. The algorithm succeeds when applying this Pauli on the quantum state received by Bob turns it back to the original code state Alice sent through the noisy channel. For CSS codes, the goal of the decoder is to correct the two initial classical codes taking into account the *degeneracy* of the CSS code. The terminology "degeneracy" refers to the property that there exist *equivalent errors*, *i.e.* different Pauli operators which act in the same way on the code states. For instance, a stabilizer generator of a code maps any code state to itself. Thus, it is equivalent to the identity and to the other stabilizer generators. By contrast in the classical setting, two errors act in the same way on the codewords if and only if they are equal. Because of the degeneracy, the decoder for a quantum code may have corrected the error even though it did not find the error that physically happened on the initial state. This is crucial for CSS LDPC codes since the classical codes used in the construction must contain constant weight codewords. Thus, CSS LDPC codes are highly degenerate. An example of a decoder with good performance is the *minimum weight decoder*. It returns a minimum weight Pauli error whose syndrome is equal to the input syndrome. A *threshold* for a family of quantum codes is a non-zero probability $p_{\text{th}} \in (0, 1]$ such that the error generated by a local stochastic channel with parameter $p < p_{\text{th}}$ is corrected with probability going to 1 in the limit of large block length.

Among CSS LDPC codes, the *toric code* introduced by Kitaev is the most famous one and has been widely studied [61, 62, 30, 86, 28, 68]. The toric code of parameter $L \in \mathbb{N}^*$ is defined using a tessellation of a 2-dimensional torus leading to a $[[2L^2, 2, L]]$ stabilizer CSS LDPC code. The toric code has many advantages. For example, its minimal distance $D = \Theta(\sqrt{N})$ is large and its stabilizer generators involve only nearest neighbor interactions. This is convenient for implementation since a 2-dimensional torus can be embedded in our 3 dimensional Euclidean space. Thus, a real device implementing the toric code would require only interactions between qubits close to

each other in space. In addition, the performance of the toric code is really good even for small block size and the minimum weight decoder can be implemented in polynomial time with the minimum weight perfect matching algorithm of Edmonds [30, 34].

Beyond the toric code, any tessellation of a manifold defines a CSS LDPC code called a *topological code*. This principle allows to use powerful arguments from topology to study such codes and to design decoders. See for example the surface code [15], the 2-dimensional hyperbolic codes [16], the semi-hyperbolic codes [17] and the 4-dimensional hyperbolic codes [51, 69, 55]. There exist two results which put strong constraints on the trade-off between the parameters N, K and D of a 2-dimensional topological code [14, 27]. The first one states that $KD^2 = \mathcal{O}(N)$ and holds for any topological code constructed from a 2-dimensional Euclidean manifold [14]. The second bound is $KD^2 = \mathcal{O}(N \log^2(N))$ and holds even if the underlying space is not Euclidean [27]. Thereby, the advantage of using topological codes constructed from 4-dimensional spaces is to go beyond these two no-go results. In particular, the parameters $K = \Theta(N)$ and $D = \Omega(N^{0.2})$ are achieved for 4-dimensional hyperbolic codes [51, 69]. Note however that there is no way to embed an hyperbolic code or a 4-dimensional code into the real world keeping nearest neighbors interactions.

Another generalization of the toric code is the *hypergraph product code* introduced by Tillich and Zémor in [108] and described in Section 4.2. This combinatorial construction has the advantage to build quantum codes from good classical ones. Thus, some arguments from classical coding theory can be imported to study them. If the initial classical codes are LDPC, have constant rate and have linear minimal distances then the resulting hypergraph product code is also LDPC, has constant rate and its minimal distance grows like the square root of the block length. This asymptotic scaling of the parameters is one of the best among the known constructions of quantum codes. The main object of interest in this PhD thesis described in Section 2.5 is a particular family of hypergraph product codes called the *quantum expander codes* [67]. Compared to general hypergraph product codes, the quantum expander codes have the advantage to come up with an efficient decoder called the *small-set-flip decoder*.

In Figure 2.6, we provide examples of CSS LDPC codes. The columns "Dimension" and "Minimal distance" contain the parameters K and D of the codes. By definition of the minimal distance, any error of weight up to $\lfloor (D-1)/2 \rfloor$ is corrected by the minimum weight decoder which runs in exponential time in general. However, the known polynomial time decoder do not necessarily correct all the errors of weight up to a fraction of the minimal distance. Hence, we report in the column "Maximum weight of adversarial errors corrected with an efficient decoder" the best known value of T such that any error of weight up to T is corrected by a polynomial time decoding algorithm.

	Dimension	Minimal distance	Maximum weight of adversarial errors corrected with an efficient decoder
Toric code [62]	2	$\Theta(\sqrt{N})$	$\Theta(\sqrt{N})$
From a 4D manifold [41]	1	$\Theta(\sqrt{N}\sqrt[4]{\log(N)})$	No efficient decoder
Hyperbolic 2D [41]	$\Theta(N)$	$\Theta(\log N)$	$\Theta(\log N)$
Hyperbolic 4D [51, 55, 69]	$\Theta(N)$	$\Omega(N^{0.2}), \mathcal{O}(N^{0.3})$	$\Theta(\log N)$
Hypergraph product codes [108]	$\Theta(N)$	$\Theta(\sqrt{N})$	No efficient decoder
Quantum expander codes [67]	$\Theta(N)$	$\Theta(\sqrt{N})$	$\Theta(\sqrt{N})$

Figure 2.6: some examples of CSS LDPC codes.

2.4 Hypergraph product codes

It is still an open question to know whether LDPC CSS codes with constant rate and linear minimal distance exist. A naive approach would be to build a CSS code from two LDPC codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ with good minimal distances. Unluckily, if $\mathcal{C}_Z$ is LDPC then the vector space $\mathcal{C}_Z^\perp$ contains constant weight elements and the required inclusion $\mathcal{C}_Z^\perp \subseteq \mathcal{C}_X$ implies that the minimal distance of $\mathcal{C}_X$ is constant. Accordingly, good LDPC classical codes cannot be used directly to build a good LDPC CSS code.

The hypergraph product construction builds a CSS code $\mathcal{Q}$ starting from two classical codes $\mathcal{C}_1$ and $\mathcal{C}_2$ without any extra condition such as $\mathcal{C}_2^\perp \subseteq \mathcal{C}_1$. More precisely, two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ are constructed from $\mathcal{C}_1$ and $\mathcal{C}_2$ and then the hypergraph product code is the CSS code associated to $\mathcal{C}_X$ and $\mathcal{C}_Z$. It is particularly interesting for the LDPC case because if $\mathcal{C}_1$ and $\mathcal{C}_2$ are constant rate LDPC codes then so is $\mathcal{Q}$. In addition, when $\mathcal{C}_1$ and $\mathcal{C}_2$ have linear minimal distances, the hypergraph product has minimal distance equal to $D = \Theta(\sqrt{N})$ where N is the number of physical qubits. Thanks to these favorable parameters, it is expected that hypergraph product codes perform well for quantum error correction and fault-tolerant quantum computation.

A natural idea for decoding hypergraph product codes is to bring the decoders of classical codes. Unfortunately, this strategy does not work in general. For example, the belief propagation decoder is intensively used in the classical setting but does not perform well for hypergraph product codes [86, 78]. Hopefully, it can be improved using neural networks [68] or an ordered statistics decoding post-processing [85]. Ref. [85] shows that hypergraph product codes have really good performance in practice: for a depolarizing channel with physical error rate below 10%, a well chosen hypergraph product code with 28 logical qubits performs better than the minimum weight decoding of a surface code with 1 logical qubit (see Figure 3 of [85]).

2.5 Quantum expander codes

In 1996, Sipser and Spielman introduced the concept of *expander graphs* and defined a *classical expander code* as being a code whose Tanner graph is an expander [98].

The expansion property is really convenient to show some properties of the code. For example, the minimal distance of a classical expander code is linear in the block length. For this PhD thesis, we are especially interested in the manner in which sufficient expansion implies that any error of weight up to a fraction of the minimal distance is corrected by a decoder called the *bit-flip* algorithm. The bit-flip algorithm is one of the simplest decoders it is possible to design and this is why it is convenient for a theoretical analysis. Nevertheless, numerical simulations show that the belief propagation decoder is the best known decoder for classical LDPC codes and widely outperforms the bit-flip algorithm. In addition, the belief propagation decoder can also be analyzed with expansion based arguments [19].

Finding the expansion parameters of a graph is a co-NP-hard problem [10] and constructing expander graphs is not straightforward. In this work, we rely on a probabilistic construction called the *configuration model* allowing to build with high probability a code with any desired expansion parameter and any desired rate [98, 10, 80]. The first deterministic construction of expander graphs is based on algebraic arguments by Margulis [77] and was latter improved by Barg and Zémor [5, 7, 6]. It is also possible to construct good expander graphs with the zig-zag product method [22]. See [56] for a survey on expander graphs.

By definition, a *quantum expander code* is a $[[\Theta(n^2), \Theta(n^2), \Theta(n)]]$ stabilizer code defined as the hypergraph product of a classical $[n, \Theta(n), \Theta(n)]$ expander code with itself [67]. Ref. [67] also introduced the *small-set-flip decoder* and proved that any error of weight up to a fraction of the minimal distance is corrected. The main motivation for studying these codes is the result of Daniel Gottesman showing that fault-tolerant quantum computation with constant space overhead is possible [46]. As discussed at the beginning of this chapter, this result is subjected to the conjecture that quantum codes with suitable properties exist. Thanks to [67], it is already known that quantum expander codes satisfy many of these properties: they are LDPC, they have constant rate and a good minimal distance. The main result of my PhD is to show that quantum expander codes satisfy also the other desired properties and thus can be used to implement Gottesman's fault-tolerant scheme.

Remember from the hypergraph product construction that a quantum expander code is a CSS code associated to two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ constructed from the initial classical expander code. It turns out that up to a permutation on the bits, the code $\mathcal{C}_X$ is equal to $\mathcal{C}_Z$. Thus, as it is usual for CSS codes, it is sufficient to describe and analyze the decoder for bit-flip errors only. Under this hypothesis, the error support determines the error on the qubits and thus we will often assume that an "error" and an "error support" are the same objects. The small-set-flip decoder is a hard decoding algorithm. This means that its execution is divided into several rounds and at each round, the qubits belonging to some set F are flipped. The set F is called a *small-set* and we denote by $\mathcal{F}$ the ensemble of all the possible small-sets that can be flipped. The implementation of the small-set-flip algorithm is quite simple: at each round, a small-set $F \in \mathcal{F}$ is selected in such a way that the syndrome weight decreases sufficiently when the qubits of F are flipped (the syndrome weight is the number of syndrome bits equal to 1).

2.6 Fault-tolerant quantum computation

The goal of fault-tolerant quantum computation is to design circuits which are robust against noise [44]. A circuit C is described by wires (containing quantum states) and elementary operations such as state preparations, measurements or unitary gates. We often assume that C has a classical output $r \in \mathbb{F}_2^m$. During the last step, all the quantum wires are measured and r is equal to the measurement output on which is applied a classical post-processing. In this PhD thesis, a state preparation creates a $|0\rangle$ state, a measurement corresponds to the measurement of a Z-Pauli matrix and the unitary gates belong to the finite gate set $\mathcal{G} := \{X, Z, H, S, T, C_X\}$ where:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

$$S = \begin{pmatrix} e^{-i\pi/4} & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \quad T = \begin{pmatrix} e^{-i\pi/8} & 0 \\ 0 & e^{i\pi/8} \end{pmatrix}, \quad C_X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Note that $\mathcal{G}$ is not minimal since $Z = S^2 = T^4$ and $X = H \circ Z \circ H$. However, it is *universal* in the sense that any unitary can be approximated with gates belonging to $\mathcal{G}$. Formally, for all $\eta > 0$ and for all unitary U, there exists a unitary V constructed from the gates of $\mathcal{G}$ such that:

$$\big\| (U - V) |\psi\rangle \big\| \leq \eta \qquad \text{for any pure state } |\psi\rangle.$$

In addition, the Solovay-Kitaev theorem [61] asserts that any circuit made of m controlled-not gates and arbitrary single qubit gates can be approximated with $\mathcal{O}\big(m \log^2(m/\eta)\big)$ gates of $\mathcal{G}$.

A *location* is a point of the circuit where an error can occur. For example, each state preparation, each measurement and each unitary gate is a location. The noise model we will use is the *local stochastic error model* with parameter $p \in [0, 1]$. Let L be the set of locations of an arbitrary circuit C, then the error is represented by a random variable $F \subseteq L$ called the set of *faulty locations*. F satisfies for all $R \subseteq L$:

$$\mathbb{P}\Big[R \subseteq F\Big] \leq p^{|R|}.$$

Once F has been chosen, a location of $L \setminus F$ behaves normally but a faulty location $l \in F$ is replaced by an arbitrary CPTP map with the same input and output spaces than l. Let r be the output that C would yield in the noiseless case $F = \varnothing$. Then, the aim of fault-tolerant quantum computation is, given a parameter $\epsilon > 0$, to design a circuit C$'$ whose output is r with probability at least $1 - \epsilon$ when the locations are subjected to a local stochastic noise.

In ref. [1], the fault-tolerant circuit C$'$ is built using *code concatenation*. The idea is to define a function Φ mapping a circuit C to another circuit $\Phi(\mathrm{C})$ which is more

robust against noise. Then, the circuit C' is defined by $\mathrm{C}' = \Phi^k(\mathrm{C})$ for a well chosen $k \in \mathbb{N}$. In $\Phi(\mathrm{C})$, each wire of C is encoded into N wires using an $[[N, 1]]$ stabilizer code and error correction is regularly performed on the wires of $\Phi(\mathrm{C})$. In this fault-tolerant protocol, the qubits are encoded one by one leading to polylogarithmic space and time overheads as stated in eq. (2.1).

In ref. [46] where fault-tolerance with constant space overhead is achieved, the wires of C are divided into blocks of $K \in \mathbb{N}$ wires and each block is encoded using an $[[N, K]]$ stabilizer code $\mathcal{Q}$. Contrarily to the case of code concatenation, the block length of $\mathcal{Q}$ depends on the number of wires in C and $\mathcal{Q}$ is chosen among a family of constant rate LDPC codes. In this PhD thesis, we show that quantum expander codes can be used in this construction to correct the errors appearing in the resulting circuit C'.

2.7 Summary of contributions

The authors of ref. [67] studied how the small-set-flip decoder corrects a quantum expander code in the case where the error is adversarial. The drawback of this setting is that the minimal distance of the code is a fundamental limit on the error weight that any algorithm can correct. When the error is generated with a quantum noisy channel (for example with the depolarizing channel) the error weight is generally linear in the block size N. This is way above the best known minimal distance $\Theta\left(\sqrt{N}\ \sqrt[4]{\log(N)}\right)$ for LDPC quantum codes [41]. Despite this upper bound on the minimal distance, it is well known that some quantum LDPC codes have a *threshold* and thus successfully correct a depolarizing noise (see [30, 17] for numerical simulations and [30, 66] for theoretical arguments). In this PhD thesis, we show that the small-set-flip decoder has a threshold for any noise model satisfying the *local stochastic* property that we defined and discussed in Section 2.3.

Kovalev and Pryadko already proved the existence of a threshold when a local stochastic noise is corrected with the minimum weight decoder [66]. As explained below, we showed in ref. [38] that similar techniques can be used for the small-set-flip decoder. We denote by $V_{\mathcal{Q}}$ the set of qubits and by G_X the Tanner graph of $\mathcal{C}_X$. We will say that a set of qubits is a *small ensemble* when any adversarial error whose support is included in this ensemble is corrected by the decoder. For instance for quantum expander codes, a "small ensemble" is a set of qubits $K \subseteq V_{\mathcal{Q}}$ with size $\mathcal{O}(\sqrt{N})$. Here, we use the word "ensemble" as a synonym of "set" to avoid a mix-up between "small ensemble" and "small-set". In addition, for a given initial error $E \subseteq V_{\mathcal{Q}}$, two sets of qubits K_1, K_2 are said to be *independent* when the behavior of the decoder on the error $K_1 \cap E$ does not depend on the error $K_2 \cap E$. In ref. [66], the qubits are decomposed into small independent ensembles so that the decoder corrects the error included in each ensemble and thus corrects the entire error as well.

In order to decompose the qubits into small independent ensembles, we will say that a decoder is *local* when two sets of qubits are independent as soon as the intersection of their neighborhoods in G_X is empty (as a reminder, the neighborhood of a set of qubits in G_X is a set of check-nodes). For example, the small-set-flip decoder and the minimum weight decoder are local. We define the *adjacency graph* of the code to be the

graph with vertex set $V_{\mathcal{Q}}$ and such that two qubits are linked if and only if they share a check-node in G_X. In particular, if two sets $K_1, K_2 \subseteq V_{\mathcal{Q}}$ are not adjacent in the adjacency graph, then no check-node can be adjacent to both K_1 and K_2 in G_X, and thus K_1 and K_2 are independent.
When we run a decoder, we call *residual error* the physical error remaining on the qubits after applying the correction the decoder guessed. We also define the *execution support* to be the set of all qubits belonging to the error support at some point of the algorithm. For instance with the small-set-flip decoder, the execution support contains the qubits of the initial error together with the qubits of all the small-sets which have been flipped by the algorithm. Then, we can decompose the qubits into independent ensembles $K_1, K_2, \ldots$ by defining K_i to be the connected components of the execution support in the adjacency graph. Finally, we will use percolation arguments to show that K_i is a small ensemble with high probability [48, 74, 58].
To summarize the arguments above, we give a sketch of the proof of [38]: we run the small-set-flip decoder on an error generated with a local stochastic noise. Let K be a connected component of the execution support U in the adjacency graph. Then, the locality property ensures that the way the decoder acts on K does not depend on whether or not there are errors outside K. In particular, the residual error on the qubits of K is equal to the residual error we would get by running the small-set-flip decoder without initial error outside K. Using arguments from percolation theory, the set K satisfies $\left|K\right| = \mathcal{O}(\sqrt{N})$ with high probability and thus the initial errors belonging to K are corrected by the small-set-flip decoder. This is true for each connected component K of U thus the entire error is corrected.

Percolation theory by itself is a whole field of probability theory [48]. However, in this work we are interested in the special case of *site percolation* with local stochastic noise on a finite graph with bounded degree. Let $\mathcal{V}$ be the set of vertices of a finite graph $\mathcal{G}$ with maximum degree $d_{\mathcal{G}}$ (the vertices are called "sites" in the context of percolation) and choose a random subset of the vertices $E \subseteq \mathcal{V}$. Then, the central question in percolation theory is to understand what is the size of the connected components of E. For error correcting codes, the graph $\mathcal{G}$ is the adjacency graph and the set E represents the support of the error. Usually for percolation, the probabilistic law on E is an *iid error model* (each site is in E with probability p independently from the other sites) but in this work we deal with the more general *local stochastic* error model. By definition, a noise model has the local stochastic property when the random set E satisfies eq. (2.4) for all $S \subseteq \mathcal{V}$. Hence, following ref. [66], we extend some useful percolation results to this case.

The other significant difference with the results from standard percolation is that we perform a generalized process that we call α-*percolation*. Instead of being interested in the maximum size of the connected subsets of E, the aim of α-percolation is to look at the connected α-subsets where an α-*subset* is a set $X \subseteq \mathcal{V}$ such that at least $\alpha|X|$ elements of X belong to E. For instance, a 1-subset is simply a subset and 1-percolation is percolation in the usual sense. Looking at α-subsets is relevant because the execution support of the small-set-flip decoder is an α-subset of the initial error for some $\alpha \in (0, 1]$. The main theorem (already proved in the particular case $\alpha = 1/2$ in ref. [66]) states that

with high probability, any connected α-subset of E has size $\mathcal{O}(\sqrt{|\mathcal{V}|})$. If we go back to the discussion about local decoders, the set K (defined to be any connected subset of the execution support) is an α-subset of E and thus $\left|K\right| = \mathcal{O}(\sqrt{N})$ must hold.

For an iid noise and $\alpha = 1$, it is well known that with high probability, if $p < 1/(d_{\mathcal{G}} - 1)$ then the connected components of E have size $\mathcal{O}\big(\log(|\mathcal{V}|)\big)$. On the other hand, if $p > 1/(d_{\mathcal{G}} - 1)$, there is a unique component whose size is linear (called "giant component") [58]. In this work, we extend this result to any local stochastic noise and arbitrary $\alpha \in (0, 1]$. More precisely, we determine a non-zero value $p_{\mathrm{th}} = p_{\mathrm{th}}(\alpha)$ such that if $p < p_{\mathrm{th}}$, then the probability for the existence of a connected α-subset for E with size above $t \in \mathbb{N}$ is at most $\Theta(|\mathcal{V}|\beta^{\alpha t})$ where $\beta = p/p_{\mathrm{th}} < 1$. A threshold value usually identifies a phase transition between the regime where E has small connected components and the regime where E has a unique giant component. However for error correction, the concern is to be below threshold and thus p_{th} is said to be a threshold even though it is only a lower bound on the percolation threshold. In particular, the value p_{th} we get is not tight for an iid noise since $p_{\mathrm{th}}(1) < 1/(d_{\mathcal{G}} - 1)$. It could be interesting to find the actual threshold for $\alpha < 1$ and to extend the other results from percolation on finite graphs [58].

A challenging assumption to apply Gottesman's fault-tolerant scheme is to show that the quantum expander codes are not defeated when the syndrome measurement is noisy. Formally in this error model, a set of syndrome bits D is randomly chosen according to a local stochastic noise and the input of the small-set-flip algorithm is the syndrome where the bits of D have been flipped. This hypothesis is necessary in the context of fault-tolerant quantum computation, because the syndrome bits are produced by a quantum measurement performed with physical noisy components. However because of the LDPC property, we cannot hope for the decoder to correct entirely the error on the qubits when the syndrome is noisy. Instead, we show that the residual error is equivalent to a local stochastic error with controlled parameter.

Similarly to the case where the syndrome measurement is perfect, the first step of the analysis is to consider adversarial errors whose weight is below the minimal distance. In that case, the weight of the residual error is shown to be upper bounded by a linear function of $|D|$. The proof strategy is to go back to the case of noiseless syndrome and to show that the small-set-flip algorithm can flip many small-sets in each round. As we discuss later, this property also implies that the decoder can be parallelized for noiseless and noisy syndrome measurements. Finally, to deal with local stochastic errors, we will apply an α-percolation process to the *syndrome adjacency graph* to reduce the problem to the case of adversarial errors. The syndrome adjacency graph is the Tanner graph of $\mathcal{C}_X$ with additional edges between the qubits linked in the adjacency graph [46].

In addition, our analysis shows that the small-set-flip algorithm has the *single-shot* property. A quantum code is said to be single-shot when one round of noisy syndrome measurement is sufficient for the decoder to have a threshold. By contrast, the toric code and other 2-dimensional quantum codes are not single-shot. This means that the syndrome has to be measured $\Theta(D)$ times to get enough information to correct the error. This property is really favorable in the context of fault-tolerance where many error correction steps have to be performed.

Single-shot error correction was introduced by Hector Bombin in [12] and various code families have been shown to be single-shot: the three-dimensional gauge color codes [12], the four-dimensional toric code [32] and the four-dimensional hyperbolic codes [55]. A theory of single-shot error correction has been proposed in ref. [21] where the main idea was to correct the syndrome errors before trying to correct the qubit errors. Taking into account [21] and Lemma 4.21 reported in this manuscript, the single-shot property seems closely related to the *soundness* property. Informally, a code is sound if below minimal distance, the syndrome weight can be lower bounded by a linear function of the error weight.

In ref. [37] we have also shown that the small-set-flip algorithm can be parallelized to run in constant depth. When the syndrome is noisy, if the number of rounds is chosen to be big enough (but constant) then the residual error will be equivalent to a local stochastic error with a small parameter. On the other hand, when the syndrome is perfect, a fixed number of rounds will not be sufficient to correct entirely the error. However, running the algorithm with a number of steps logarithmic in the syndrome weight will correct the error with high probability.

In ref. [49], we have done some simulations to study how the small-set-flip decoder performs in practice. For simplicity, we have restricted our attention to the sequential algorithm and perfect syndrome measurements. The numerical results we get are promising. For instance, the value we derived for the threshold is way above the lower bound provided by theoretical arguments. The threshold value is near 4.5% for a family of hypergraph product codes with rate $1/61$ and near 2% for rate $1/5$.

Chapter 3

Classical error correction

The goal of this chapter is to provide the notions of classical coding we did not introduced in Section 2.1. Section 3.1 summarizes standard terminology and definitions. Section 3.2 is a detailed study of classical expander codes, many of the ideas of this section will be used in Chapter 4 for quantum expander codes.

3.1 Background

As stated in Section 2.1, a classical message on n bits is represented with a binary vector of $\mathbb{F}_2^n$. The group operation on $\mathbb{F}_2 = \{0, 1\}$ is the addition modulo 2 denoted by $\oplus$ and defined by:

$$0 \oplus 0 = 1 \oplus 1 = 0, \qquad 0 \oplus 1 = 1 \oplus 0 = 1.$$

The metric on $\mathbb{F}_2^n$ used for error correcting codes is called the *Hamming distance*:

Definition 3.1 (Hamming weight and Hamming distance)**.** The Hamming weight of a binary vector $e \in \mathbb{F}_2^n$ is the number of 1s in e:

$$|e| := \#\Big\{i \in [\![1; n]\!] : e_i = 1\Big\}.$$

The Hamming distance between $e_1 \in \mathbb{F}_2^n$ and $e_2 \in \mathbb{F}_2^n$ is equal to $\big|e_1 \oplus e_2\big|$, it is the number of indices where the bits of e_1 and e_2 are different.

For example, when Alice sends a bit-string $x \in \mathbb{F}_2^n$ to Bob through the binary symmetric channel, the probability he gets some message $y \in \mathbb{F}_2^n$ is equal to:

$$\mathbb{P}\Big[\text{Bob gets } y \;\Big|\; \text{Alice sent } x\Big] = p^{|x \oplus y|}(1-p)^{n-|x \oplus y|}.$$

The binary vector $e := x \oplus y$ is called the *error*. When the i^{th} bit of x is flipped by the channel, we have $e_i = 1$ and $e_i = 0$ otherwise.

By definition, the set of codewords of an $[n, k, d]$-linear error correcting code $\mathcal{C}$ is a k dimensional linear subspace of $\mathbb{F}_2^n$. The integer k is called the number of logical bits, n is called the number of physical bits and d is the minimal distance defined by:

$$d := \min\Big\{|c| : c \in \mathcal{C}, c \neq 0^n\Big\}.$$

Representing a linear code can be done using either a *generator matrix* or a *parity check matrix*.
A generator matrix G is an $n \times k$ binary matrix whose columns span the set of codewords of $\mathcal{C}$, in particular the rank of G is equal to k the dimension of $\mathcal{C}$. In the communication protocol of Figure 2.4, Alice encodes a message $s \in \mathbb{F}_2^k$ into a codeword $x \in \mathbb{F}_2^n$, then she sends x through the noisy channel to Bob who gets $y \in \mathbb{F}_2^n$ as channel output, he corrects y to $\hat{x} \in \mathbb{F}_2^n$ with the decoding algorithm and finally he computes $\hat{s} \in \mathbb{F}_2^k$ the message whose corresponding codeword is $\hat{x}$. With the generator matrix G, Alice can perform efficiently the encoding step using the formula $x = Gs$ and Bob can deduce $\hat{s}$ from $\hat{x}$ using a Gaussian elimination on the linear system $\hat{x} = G\hat{s}$.
A parity check matrix for $\mathcal{C}$ is any binary matrix H with n columns satisfying $\mathcal{C} = \ker(H)$. In particular, each column of G belongs to the kernel of H thus $HG = 0$. Moreover, the rank of H is $n - k$ thus $m \geq n - k$ with equality when H is full rank. The rows of H span $\mathcal{C}^\perp$ the orthogonal space of $\mathcal{C}$ defined by

$$\mathcal{C}^\perp := \Big\{d \in \mathbb{F}_2^n : d^T c = 0 \text{ for all } c \in \mathcal{C}\Big\}$$

where d^T is the transpose vector of d. The rows of H can also be seen as linear constraints on the bits of $c \in \mathcal{C}$ called *parity check equations* as illustrated in Figure 3.1 with the Hamming code.

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

$$x \in \mathcal{C} \Leftrightarrow \begin{cases} x_1 \oplus x_2 \oplus x_3 \oplus x_5 = 0 \\ x_2 \oplus x_3 \oplus x_4 \oplus x_6 = 0 \\ x_3 \oplus x_5 \oplus x_6 \oplus x_7 = 0 \end{cases}$$

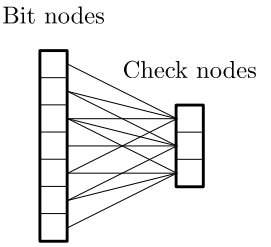

Figure 3.1: parity check matrix, parity check equations and factor graph for the $[7, 3, 3]$ Hamming code.
The columns of the parity check matrix are the 7 possible non-zero strings with 3 bits.

For our purpose, there is a convenient way to represent a parity check matrix H with n columns and m rows using a bipartite graph called the *factor graph* or the *Tanner graph* of the code (see Figure 3.1 for an example). Let G be a bipartite graph whose left set of vertices V satisfies $\big|V\big| = n$ and whose right set of vertices C satisfies $\big|C\big| = m$.

In this context, an element $v \in V$ is called a *bit-node* or simply a *bit* and will represent a column of H (*i.e.* a physical bit of the code). Similarly, an element $c \in C$ is called a *check-node* or simply a *check* and will represent a row of H (*i.e.* a parity check equation). The graph G is said to be the Tanner graph of H (or a Tanner graph for the associated code) when the edges in G correspond to the 1 in H: if $H_{i,j} = 1$ then the i^{th} bit-node and the j^{th} check-node are connected and if $H_{i,j} = 0$ then their are not connected.

The bit-string $\sigma(y) := Hy \in \mathbb{F}_2^m$ is called the *syndrome* of y or the *syndrome* of the error, for example $y \in \mathcal{C}$ holds if and only if $\sigma(y) = (0, \dots, 0)$. The terminology "syndrome of the error" is justified by the fact that when Alice sends a codeword x to Bob who receives y, the syndrome of y does not depend on x but exclusively depends on the error $e = x \oplus y$ that happened on the channel. Indeed, for all $x \in \mathcal{C}$, $Hx = 0$ holds and:

$$\sigma(y) = Hy \oplus Hx = \sigma(e).$$

As shown in Figure 2.4, the error correction step is done with an algorithm which infers some $\hat{x}$ from the output of the channel y. In this PhD thesis we use the *syndrome decoding* strategy where the decoder takes as input the syndrome $\sigma(y) = \sigma(e) \in \mathbb{F}_2^m$, returns $\hat{e} \in \mathbb{F}_2^n$ a guessed for the error and finally sets $\hat{x} = y \oplus \hat{e}$. Syndrome decoding is particularly relevant for quantum error correction since we do not have a direct access to the value of a quantum state.
For example, on the input $\sigma \in \mathbb{F}_2^m$, the *minimum weight decoder* returns:

$$\hat{e} := \underset{e \in \mathbb{F}_2^n : \sigma(e) = \sigma}{\arg\min} |e|. \tag{3.1}$$

Notation 3.2. *It will be convenient to see an error pattern $e \in \mathbb{F}_2^n$ as a subset of $[\![1; n]\!]$ denoted by an upper letter:*

$$E := \Big\{ i \in [\![1; n]\!] : e_i = 1 \Big\}.$$

Similarly, an error is often seen as a subset of the bit-nodes and the syndrome is often seen as a subset of the check-nodes: $E \subseteq V$ and $\sigma \subseteq C$.
When a bit-string is seen as a subset, the bit-wise addition $\oplus$ is replaced by the symmetric difference of sets and the Hamming weight defined in Definition 3.1 becomes the cardinality.

On a Tanner graph, the syndrome is the set of check-nodes incident to an odd number of faulty bits, see Figure 3.2 for some examples on the Hamming code. A check-node is said to be *unsatisfied* when it belongs to the syndrome and is said to be *satisfied* otherwise.

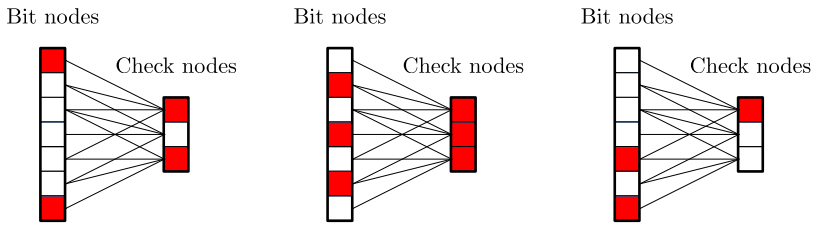

Figure 3.2: examples of errors and the associated syndrome for the Hamming code. The faulty bits and the unsatisfied checks are in red.

When we will talk about the *degree* of a physical bit or the *degree* of a check, we refer to the degree of the corresponding node in the Tanner graph. The degree of a bit is the number of parity check equations in which it appears and the degree of a check is the number of bits in the corresponding parity check equation.

In this PhD thesis, we are interested in the notion of *low-density parity check* (LDPC) codes. A linear code is (l, r)-LDPC when the bit-nodes in the Tanner graph have degree at most l and the check-nodes have degree at most r. A family of linear codes $\mathcal{C}_1, \mathcal{C}_2, \ldots$ is said to be LPDC when there exist two integers $l, r \in \mathbb{N}$ such that each code of the family is (l, r)-LDPC. For a family of LDPC codes, the parity check matrices are sparse because the weight of the columns of the matrices is upper bounded by l and the weight of the rows of the matrices is upper bounded by r.

3.2 Classical expander codes

Remember that the syndrome decoding task consists in finding the bit errors which have led to a given set of unsatisfied check-nodes. By definition, the neighborhood of any unsatisfied check-node must contain at least one bit in the error. It would be very convenient if, on the one hand, all the bits in the error have a neighborhood containing a majority of unsatisfied check-nodes and, on the other hand, the neighborhood of all the bits not in the error was mainly composed of satisfied check-nodes. Under these two assumptions, the error would be corrected by flipping all the bits adjacent to a majority of unsatisfied check-nodes. Actually, we cannot hope these two properties to hold for any adversarial error but classical expander codes are defined so that the bits mostly satisfy these properties. In order to avoid a situation where many satisfied check-nodes are in the neighborhood of a bit in the error, the expansion property ensures that there are few check-nodes adjacent to two bits in the error. Equivalently, the number of check-nodes in the neighborhood of the error must be large.

An expander code can be decoded with the *bit-flip* algorithm which, while it is possible to do so, finds and flips a bit such that this flip decreases the syndrome weight. The bit-flip algorithm is sequential in the sense that a single bit is flipped at each round of the procedure. It is also possible to design a parallel version of the algorithm where at each round, all the bits having a majority of unsatisfied neighbors are flipped. Even though the sequential algorithm has better performance than the parallel version, it is also way slower since it runs essentially in linear time compared to logarithmic time when parallelized, see for example the table in [98]. In this chapter, we focus on the sequential bit-flip algorithm trying to highlight the ideas that can be reused for the analysis of the *small-set-flip* decoder for quantum expander codes (see Section 4.3). In particular, we introduce some concepts and show some properties that are not essential for classical expander codes but will be useful in the quantum setting. Note that the constants reported in the statements of this chapter can be improved with the arguments of [18].

3.2.1 Definition

All along this PhD thesis, we fix a Tanner graph G whose set of bit-nodes is V, whose set of check-nodes is C and we denote by Γ the neighborhood in G. In addition, G is supposed to be a regular graph: the bit-nodes have degree d_V and the check-nodes have degree d_C. Note that for any $E \subseteq V$: $\big|\Gamma(E)\big| \leq d_V\big|E\big|$ and for any $D \subseteq C$: $\big|\Gamma(D)\big| \leq d_C\big|D\big|$. An expander graph is such that for any set E or D sufficiently small, the neighborhood is large in the sense that the previous upper bounds are nearly reached.

Definition 3.3 (Expander graph [98])**.** Let G be a regular Tanner graph with V the set of bit-nodes and C the set of check-nodes. The left degree of G is denoted by d_V and its right degree is denoted by d_C. We say that G is an expander graph with parameter

$\delta > 0$ if there exists $\gamma > 0$ such that:

$$\forall E \subseteq V : \big|E\big| \leq \gamma \big|V\big| \Rightarrow \big|\Gamma(E)\big| \geq (1-\delta) d_V \big|E\big|,$$
$$\forall D \subseteq C : \big|D\big| \leq \gamma \big|C\big| \Rightarrow \big|\Gamma(D)\big| \geq (1-\delta) d_C \big|D\big|.$$

When we want to specify the value of γ, we say that a graph is a (γ, δ)-expander graph. A *classical expander code* with parameters $\gamma, \delta > 0$ is a code associated to a (γ, δ)-expander Tanner graph.
A code family is a *classical expander code family* with parameter $\delta > 0$ when there exists $\gamma > 0$ (γ independent of the code) such that each code of the family is a (γ, δ)-expander code.

Following [67] we could distinguish between the notions of left expansion and right expansion.
G is a left expander graph with parameter $\delta_V > 0$ if there exists $\gamma_V > 0$ such that:

$$\forall E \subseteq V : \big|E\big| \leq \gamma_V \big|V\big| \Rightarrow \big|\Gamma(E)\big| \geq (1-\delta_V) d_V \big|E\big|. \tag{3.2}$$

G is a right expander graph with parameter $\delta_C > 0$ if there exists $\gamma_C > 0$ such that:

$$\forall D \subseteq C : \big|D\big| \leq \gamma_C \big|C\big| \Rightarrow \big|\Gamma(D)\big| \geq (1-\delta_C) d_C \big|D\big| \tag{3.3}$$

For simplicity, we consider graphs with both right and left expansion where $\gamma = \gamma_V = \gamma_C$ and $\delta = \delta_V = \delta_C$. Note that the left expansion property is sufficient for the analysis done in Section 3.2 for classical expander codes but we will need left and right expansion for quantum expander codes.

3.2.2 Analysis of classical expander codes

In [98], Sipser and Spielman showed that an expander code with parameter $\delta < 1/2$ has a linear minimum distance and if $\delta < 1/4$ then any error of weight up to a fraction of the minimal distance is corrected by a decoder called the *bit-flip algorithm*.

In this section, we establish useful properties of expander graphs and deduce a lower bound on the minimal distance.

Notation 3.4. *Let G be a bipartite expander graph with parameters $\gamma, \delta > 0$, with left degree d_V and right degree d_C. Let V be the set of bit-nodes and let C be the set of check-nodes. The graph G is interpreted as a Tanner graph and we denote by $\mathcal{C}$ the corresponding classical error correcting code.*
For $E \subseteq V$, let $\Gamma_{\mathrm{u}}(E) \subseteq \Gamma(E)$ be the set of unique neighbors of E:

$$\Gamma_{\mathrm{u}}(E) := \Big\{ c \in C : \text{ there exists a unique } e \in E \text{ such that } c \in \Gamma(e) \Big\}$$

and let $\Gamma_{\mathrm{m}}(E)$ be the set of multiple neighbors of E:

$$\Gamma_{\mathrm{m}}(E) := \Gamma(E) \setminus \Gamma_{\mathrm{u}}(E) = \Big\{ c \in C : \exists e \neq e' \in E, c \in \Gamma(e) \cap \Gamma(e') \Big\}.$$

The key property of bipartite expander graphs is that the size of $\Gamma_{\mathrm{u}}(E)$ is large and the size of $\Gamma_{\mathrm{m}}(E)$ is small (see Lemma 3.5).

Lemma 3.5. *We use Notation 3.4. If $E \subseteq V$ is such that $|E| \leq \gamma|V|$ then:*

$$|\Gamma_{\mathrm{u}}(E)| \geq d_V|E|(1-2\delta) \qquad \text{and} \qquad |\Gamma_{\mathrm{m}}(E)| \leq d_V|E|\delta.$$

Proof. First of all, we have:

$$|\Gamma(E)| = |\Gamma_{\mathrm{u}}(E)| + |\Gamma_{\mathrm{m}}(E)|. \tag{3.4}$$

Moreover, $d_V|E| = \sum_{e\in E}|\Gamma(e)|$. On the right hand side of the latter equality, the elements of $\Gamma_{\mathrm{u}}(E)$ are counted once and the element of $\Gamma_{\mathrm{m}}(E)$ are counted at least twice. Thus:

$$d_V|E| \geq |\Gamma_{\mathrm{u}}(E)| + 2|\Gamma_{\mathrm{m}}(E)|. \tag{3.5}$$

Subtracting eq. (3.4) from eq. (3.5) and using the expansion property $\Gamma(E) \geq (1-\delta)d_V|E|$ from Definition 3.3, we get:

$$|\Gamma_{\mathrm{m}}(E)| \leq d_V|E| - |\Gamma(E)| \leq d_V(1-1+\delta)|E| = d_V|E|\delta. \tag{3.6}$$

By eq. (3.4), eq. (3.6) and the expansion property:

$$|\Gamma_{\mathrm{u}}(E)| = |\Gamma(E)| - |\Gamma_{\mathrm{m}}(E)| \geq |\Gamma(E)| - d_V|E|\delta \geq (1-2\delta)d_V|E|.$$

□

By Lemma 3.5 and the following argument, the minimal distance of an expander code with $\delta < 1/2$ is linear in the block length $|V|$. Indeed, if an error $E \subseteq V$ satisfies $0 < |E| \leq \gamma|V|$ then $\sigma(E) \neq \varnothing$ because $\Gamma_{\mathrm{u}}(E) \subseteq \sigma(E)$ and thus:

$$|\sigma(E)| \geq |\Gamma_{\mathrm{u}}(E)| \geq d_V\delta(1-2\delta) > 0.$$

Hence we get $d(\mathcal{C}) > \gamma|V|$ since the minimal distance $d(\mathcal{C})$ is the minimal weight of a non-zero error with empty syndrome. Actually, the more careful analysis of Lemma 3.6 provides a better lower bound on the minimal distance of $\mathcal{C}$.

Lemma 3.6. *We use Notation 3.4. If $\delta < 1/2$ then:*

$$d(\mathcal{C}) \geq 2(1-\delta)\lfloor\gamma|V|\rfloor.$$

Proof. We show that for any $E \subseteq V$ with $0 < |E| < 2(1-\delta)\left\lfloor\gamma|V|\right\rfloor$, we have $\sigma(E) \neq \varnothing$.
First, using $\Gamma_{\mathrm{u}}(E) \subseteq \sigma(E)$ and Lemma 3.5, we know that in the particular case where $|E| \leq \gamma|V|$:

$$|\sigma(E)| \geq |\Gamma_{\mathrm{u}}(E)| \geq d_V|E|(1-2\delta) \neq 0 \tag{3.7}$$

and thus $\sigma(E) \neq \varnothing$.
In the other case where $\gamma|V| < |E| < 2(1-\delta)\left\lfloor \gamma|V| \right\rfloor$, we choose $E_1, E_2 \subseteq E$ such that $E = E_1 \cup E_2$, $E_1 \cap E_2 = \varnothing$ and $|E_1| = \left\lfloor \gamma|V| \right\rfloor$. We have $\sigma(E) = \sigma(E_1) \oplus \sigma(E_2)$ thus $|\sigma(E)| \geq |\sigma(E_1)| - |\sigma(E_2)|$ and showing $|\sigma(E_1)| > |\sigma(E_2)|$ is sufficient to conclude that $\sigma(E) \neq \varnothing$. Similarly to eq. (3.7), we have:

$$|\sigma(E_1)| \geq d_V(1-2\delta)|E_1| = d_V(1-2\delta)\left\lfloor \gamma|V| \right\rfloor$$

and:

$$\begin{aligned} |\sigma(E_2)| &\leq d_V|E_2| && \text{because } G \text{ has left degree } d_V, \\ &= d_V(|E| - |E_1|) \\ &< d_V(1-2\delta)\left\lfloor \gamma|V| \right\rfloor && \text{because } |E| < 2(1-\delta)\left\lfloor \gamma|V| \right\rfloor \text{ and } |E_1| = \left\lfloor \gamma|V| \right\rfloor. \end{aligned}$$

Hence $|\sigma(E_1)| > |\sigma(E_2)|$ which concludes the proof. □

3.2.3 Bit-flip algorithm

In this section we show that expander codes can be decoded using the *bit-flip algorithm*. The definition of expander graphs (Definition 3.3) implies that for an error E sufficiently small $|E| \leq \gamma|V|$, the bits $e \in V$ whose neighborhood $\Gamma(e)$ is mainly composed of unsatisfied checks are generally in the error. The bit-flip algorithm takes advantage of this remark by flipping the bits whose neighborhood contains more unsatisfied check-nodes than satisfied ones. When we pay attention to the syndrome, the bit-flip algorithm flips a bit when this leads to a decrease in the syndrome weight and it iterates this process while such a bit exists.

The bit-flip algorithm was introduced by Gallager in [42] and the analysis for the expander codes was done by Sipser and Spielman in [98]. In Algorithm 1 we present a slight generalization of the bit-flip where a bit is flipped when this decreases the syndrome weight by at least $B \in \mathbb{N}^*$ units. This parameter B is not really necessary (for example [42] and [98] set $B = 1$) but we allow for B being arbitrary because our objective is to extend the ideas of the analysis to the quantum case where taking $B \neq 1$ is relevant.

It is great for understanding the algorithm to think about the physical error on the bits as shown in Figure 3.3. Indeed, let E_i be the set of bits in error at round number i of the while loop. Of course, the decoder does not have access to E_i but we make it appear on the comments of Algorithm 1 for clarity. The variables used by the algorithm are $\sigma_i = \sigma(E_i)$ and $\hat{E}_i = E \oplus E_i$. Let f be the number of rounds in the while loop then the output of the algorithm $\hat{E} = \hat{E}_f$ represents a guess for the error E. Hence, the procedure is a success when $\hat{E} = E$, *i.e.* when the physical error at the end is empty: $E_f = \varnothing$.

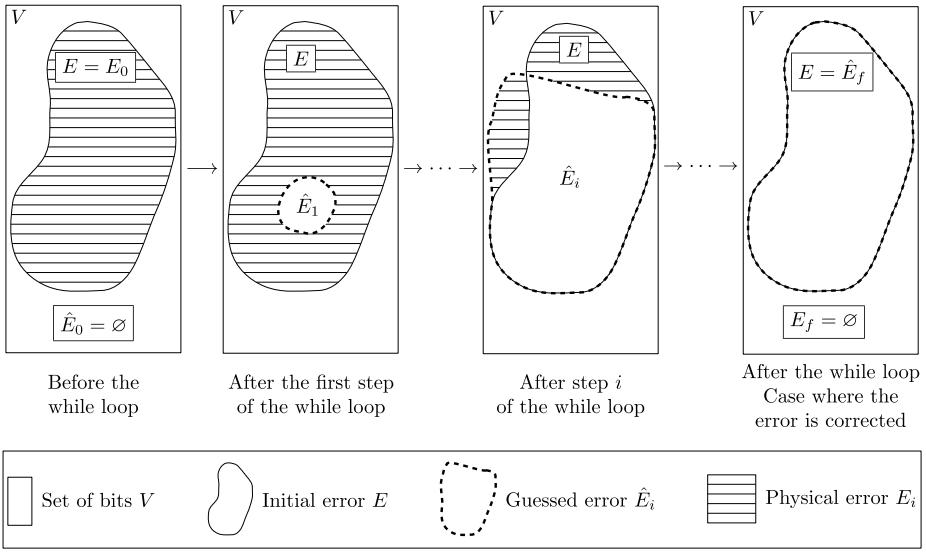

Figure 3.3: schematic representation of the sets E, E_i and $\hat{E}_i$ used in the bit-flip algorithm (Algorithm 1).

Algorithm 1 : the bit-flip algorithm with parameter $B \in \mathbb{N}^*$

Input: a syndrome $\sigma \subseteq C$. // $\sigma = \sigma(E)$ for some $E \subseteq V$
Output: a guess for the error $\hat{E} \subseteq V$.

$\hat{E}_0 = \varnothing$; $\sigma_0 = \sigma$; $i = 0$ // $E_0 = E, \sigma_0 = \sigma(E_0)$
while $\left[\exists e_i \in V : \left|\sigma_i \oplus \Gamma(e_i)\right| \leq \left|\sigma_i\right| - B\right]$ **do**
  Pick such a e_i arbitrarily. // $\left|\sigma(E_i \oplus \{e_i\})\right| \leq \left|\sigma(E_i)\right| - B$
  $\hat{E}_{i+1} = \hat{E}_i \oplus \{e_i\}$ // $E_{i+1} = E_i \oplus \{e_i\}$
  $\sigma_{i+1} = \sigma_i \oplus \Gamma(e_i)$ // $\sigma_{i+1} = \sigma(E_{i+1})$
  $i = i + 1$
end while
return $\hat{E}_i$

For simplicity we will say that we run Algorithm 1 on an input $E \subseteq V$ when we run it on the input $\sigma(E)$. In addition, we say that Algorithm 1 corrects the error E when its output $\hat{E}$ is equal to E.

Theorem 3.7. *Let $B \in \mathbb{N}^*$. We use Notation 3.4 assuming $\delta < \frac{1}{4}\left(1 - \frac{B-1}{d_V}\right)$. Any error $E \subseteq V$ with $\left|E\right| \leq \gamma\left|V\right|\left(1 + \frac{d_V}{B}\right)^{-1}$ is corrected by Algorithm 1.*

The proof of Theorem 3.7 has two main steps that we describe below in Lemma 3.9 and Lemma 3.10.
The first part of the analysis is independent of the fact that we consider an expander code and uses the notion of *execution support* defined in Notation 3.8.

Notation 3.8. *We run Algorithm 1 with parameter $B \in \mathbb{N}^*$ on the input error $E \subseteq V$ and we denote by f the number of rounds of this execution. The execution support $U \subseteq V$ is defined to be the set:*

$$U := E \cup \{e_0, \dots, e_{f-1}\}.$$

The execution support contains all the bits which are in error at some point during the execution of the algorithm. Lemma 3.9 below shows that $\big|U\big| = \mathcal{O}(\big|E\big|)$.

Lemma 3.9. *Using Notation 3.4 and Notation 3.8, we have:*

$$\big|U\big| \leq \big|E\big| \left(1 + \frac{d_V}{B}\right).$$

Proof. Let $\sigma_i \subseteq C$ be the variables from the body of Algorithm 1, we have:

$$\big|\sigma_0\big| \geq \big|\sigma_0\big| - \big|\sigma_f\big| = \sum_{i=0}^{f-1} \big|\sigma_i\big| - \big|\sigma_{i+1}\big| \geq Bf.$$

But $\big|\sigma_0\big| \leq d_V \big|E\big|$ because the bit-nodes have degree d_V thus:

$$\big|U\big| \leq \big|E\big| + f \leq \big|E\big| + \frac{\big|\sigma_0\big|}{B} \leq \big|E\big| \left(1 + \frac{d_V}{B}\right).$$

□

Lemma 3.9 does not assume the code to be an expander but this hypothesis will be used in Lemma 3.10 below. Using the notations of Figure 3.3, while the set of physical errors E_i is not empty, we do not wish the decoder to stop at round i. If $|E_i|$ is sufficiently small, the bit-flip will indeed not stop because, as stated in Lemma 3.10 where E_i is called F, there is at least one bit satisfying the while loop condition.

Lemma 3.10. *We use Notation 3.4 assuming $\delta < \frac{1}{4}$.*
Let $F \subseteq V$ be an error with $0 < \big|F\big| \leq \gamma \big|V\big|$ then there exists $e \in F$ such that flipping the bit e decreases the syndrome weight by at least $d_V(1 - 4\delta)$:

$$\big|\sigma(F \setminus \{e\})\big| \leq \big|\sigma(F)\big| - d_V(1 - 4\delta).$$

Proof. Using Lemma 3.5 applied for $E = F$:

$$\big|\Gamma_{\mathrm{u}}(F)\big| \geq d_V \big|F\big|(1 - 2\delta). \tag{3.8}$$

Hence the mean value of $\left|\Gamma_{\mathrm{u}}(F) \cap \Gamma(e)\right|$ over $e \in F$ is lower bounded in the following way:

$$\frac{1}{|F|}\sum_{e\in F}\left|\Gamma_{\mathrm{u}}(F)\cap\Gamma(e)\right| \geq \frac{1}{|F|}\left|\Gamma_{\mathrm{u}}(F)\cap\bigcup_{e\in F}\Gamma(e)\right| = \frac{\left|\Gamma_{\mathrm{u}}(F)\right|}{|F|} \geq d_V(1-2\delta).$$

Thus there exists at least one bit $e \in F$ with $\left|\Gamma_{\mathrm{u}}(F)\cap\Gamma(e)\right| \geq d_V(1-2\delta)$. We conclude that:

$$\begin{aligned}
\left|\sigma(F\setminus\{e\})\right| &= \left|\sigma(F)\oplus\Gamma(e)\right| \\
&= \left|\sigma(F)\right| + \left|\Gamma(e)\right| - 2\left|\sigma(F)\cap\Gamma(e)\right| \\
&\leq \left|\sigma(F)\right| + \left|\Gamma(e)\right| - 2\left|\Gamma_{\mathrm{u}}(F)\cap\Gamma(e)\right| \quad \text{because } \Gamma_{\mathrm{u}}(F)\subseteq\sigma(F), \\
&\leq \left|\sigma(F)\right| + d_V - 2d_V(1-2\delta) \\
&= \left|\sigma(F)\right| - d_V(1-4\delta).
\end{aligned}$$

□

We are now ready to prove Theorem 3.7 using Lemma 3.9 and Lemma 3.10.

Proof of Theorem 3.7. We run Algorithm 1 on the input E and denote by $\hat{E}$ the output. In this proof we set $F := E \oplus \hat{E}$ to be the physical error on the bits after applying the correction $\hat{E}$. Note that $F = E_f$ where f is the number of rounds in the while loop and E_i is defined as in Figure 3.3. By the contraposition of Lemma 3.10, showing items (i) and (ii) below is sufficient to prove $\left|F\right| = 0$ and Theorem 3.7.

(i) $\left|F\right| \leq \gamma\left|V\right|$.

(ii) $\forall e \in F : \left|\sigma(F\setminus\{e\})\right| > \left|\sigma(F)\right| - d_V(1-4\delta)$.

Let's show items (i) and (ii).
By definition of the execution support U defined in Notation 3.8, we have $F \subseteq U$. Using Lemma 3.9 and the hypothesis $\left|E\right| \leq \gamma\left|V\right|\left(1+\frac{d_V}{B}\right)^{-1}$, we get item (i):

$$\left|F\right| \leq \left|U\right| \leq \left|E\right|\left(1+\frac{d_V}{B}\right) \leq \gamma\left|V\right|.$$

For item (ii), we remark that Algorithm 1 stopped because the while loop condition was not satisfied at round $i = f$ for $\sigma_i = \sigma(F)$:

$$\forall e \in V : \left|\sigma(F)\oplus\Gamma(e)\right| > \left|\sigma(F)\right| - B.$$

We are dealing with integers and thus:

$$\begin{aligned}
\forall e \in V : \left|\sigma(F)\oplus\Gamma(e)\right| &\geq \left|\sigma(F)\right| - B + 1 \\
&> \left|\sigma(F)\right| - d_V(1-4\delta) \quad \text{because } \delta < \frac{1}{4}\left(1-\frac{B-1}{d_V}\right).
\end{aligned}$$

Finally, item (ii) holds because for $e \in F$:

$$\sigma(F) \oplus \Gamma(e) = \sigma(F \oplus \{e\}) = \sigma(F \setminus \{e\}).$$

□

3.2.4 Existence of expander graphs

The goal of this section is to discuss the existence and the construction of expander graphs, we do not give the proofs which can be found in ref. [92]. Note that the expansion property is equivalent to both left expansion and right expansion as defined in eqs. (3.2) and (3.3). Ref. [92] is only interested in the left expansion property but their results also hold for right expansion by exchanging the bit-nodes and the check-nodes.

In this PhD thesis, we rely on a probabilistic construction called *configuration model* which produces a bipartite graph having the expansion property with high probability. For any desired rate $r \in [0, 1)$ and any desired expansion parameter $\delta > 0$, this construction shows the existence of regular LDPC code families with expansion parameter δ and asymptotic rate r (the asymptotic rate is the limit of the rates when the number of bits goes to infinity). Explicit constructions of expander graphs are possible but are more complicated thus we focus on the configuration model [77, 5, 7, 6, 22].

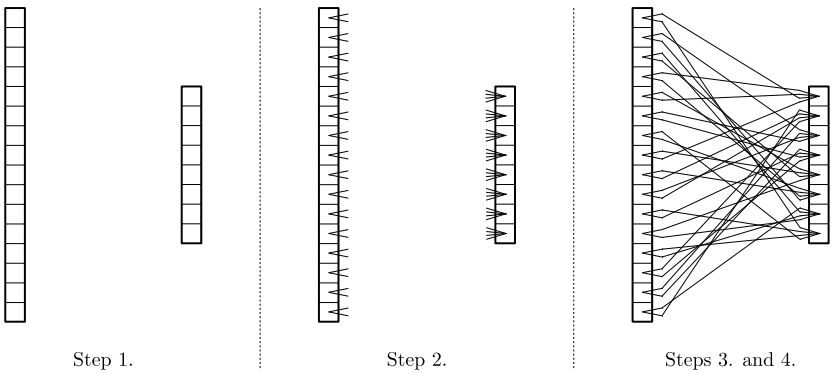

Figure 3.4: configuration model with $n = 16$, $m = 8$, $d_V = 2$ and $d_C = 4$.

The first constraint one needs to know when trying to build a family of expander codes with the LDPC property is given in Lemma 3.11 below.

Lemma 3.11. *Let G be a bipartite graph with left degree d_V, right degree d_C and expansion parameter δ then:*

$$\delta > \max\left(\frac{1}{d_V}, \frac{1}{d_C}\right).$$

Proof. The proof is presented in ref. [92] Problem 8.2. □

Since the bipartite graphs represent error correcting codes, the inequality $d_V \leq d_C$ generally holds. By Lemma 3.11, in order to build a Tanner graph with target parameter $\delta > 0$, the left degree must satisfy $d_V > 1/\delta$. For instance, the hypothesis $\delta < 1/4$ of Lemma 3.10 requires $d_V \geq 5$.

The configuration model presented below allows to construct expander graphs with high probability as soon as the constraint of Lemma 3.11 holds. The goal is to build a regular Tanner graph with $n \in \mathbb{N}$ bit-nodes, $m \in \mathbb{N}$ check-nodes, left-degree $d_V \in \mathbb{N}$ and right degree $d_C \in \mathbb{N}$ where $nd_V = md_C$. Note that $nd_V = md_C$ is required since both integers nd_V and md_C are equal to the number of edges in the graph. The configuration model proceed in the following way (see Figure 3.4 for a graphical representation):

1. Build n bit-nodes and m check-nodes.
2. Create d_V sockets per bit-node called *left-half-edges* and create d_C sockets per check-node called *right-half-edges*.
3. Pick a random permutation σ of $[\![1; nd_V]\!] = [\![1; md_C]\!]$.
4. For each $i \in [\![1; nd_V]\!]$, create an edge between the i^{th} left-half-edge and the $\sigma(i)^{\text{th}}$ right-half-edge (by convention, an edge is created only once).

A graph constructed with the configuration model is not always a good expander since any permutation σ can be picked. However, as stated in Lemma 3.12, if the necessary condition $\delta > 1/d_V$ of Lemma 3.11 holds then the resulting graph is an expander of parameter δ with high probability.

Lemma 3.12. *Let $\delta > 1/d_V$ and let G be a bipartite graph chosen at random with the configuration model as described above, then there exists $\gamma > 0$ such that:*

$$\mathbb{P}\Big[G \text{ is an expander graph with parameters } (\gamma, \delta)\Big] \geq 1 - \mathcal{O}\big(n^{-\beta}\big)$$

where $\beta = d_V\delta - 1 > 0$.

Proof. The proof is presented in ref. [92] Theorem 8.7. □

For a regular LDPC code $\mathcal{C}$ with n bit-nodes and m check-nodes, we call *design rate* the real number $1 - m/n = 1 - d_V/d_C$. The code $\mathcal{C}$ has dimension at most $n - m$ and rate at most equal to the design rate with equalities if and only if the parity check equations are independent from each other. If the left degree d_V is even then each bit appears d_V times in the parity check equations, thus they cannot be independent since their binary sum is equal to zero. Lemma 3.13 below asserts that with high probability, there is no other dependency between the parity check equations.

Lemma 3.13. *With high probability, the rate of a random code constructed with the configuration model as described above is equal to:*

$$\begin{cases} 1 - \dfrac{d_V}{d_C} & \text{if } d_V \text{ is odd} \\ 1 - \dfrac{d_V}{d_C} + \dfrac{1}{n} & \text{if } d_V \text{ is even.} \end{cases} \tag{3.9}$$

Proof. The proof is presented in ref. [92] Lemma 3.27. □

To summarize this section, we apply the previous results for the construction of a code family $\mathcal{C}_i$ with any desired expansion parameter and any minimum asymptotic rate. Let $\delta > 0$ and $r \in [0, 1)$, we define:

$$d_V := \left\lceil \frac{1}{\delta} \right\rceil + 1, \qquad d_C := \left\lceil \frac{d_V}{1-r} \right\rceil .$$

By Lemmas 3.12 and 3.13, there exists $\gamma > 0$ such that for n sufficiently large, there exist codes with expansion parameters (γ, δ) and rate as in eq. (3.9). Hence, for $i \in \mathbb{N}^*$ sufficiently large, there exists a code $\mathcal{C}_i$ with $n_i = id_C$ bit-nodes, $m_i = id_V$ check-nodes, left degree d_V, right degree d_C, expansion parameters (γ, δ) and rate as in eq. (3.9). The code family we get has asymptotic rate $1 - d_V/d_C \geq r$ and expansion parameter δ.

Chapter 4

Quantum error correction

4.1 Background

In this section, we pursue the presentation of quantum error correcting codes we started in Section 2.3.

4.1.1 Definition of stabilizer codes

In full generality, a quantum error correcting code $\mathcal{Q}$ can be defined as an arbitrary 2^K dimensional linear subspace of the Hilbert space $\left(\mathbb{C}^2\right)^{\otimes N}$. However in this work, we are interested in the class of *stabilizer codes* that we define below [43].

For a stabilizer code $\mathcal{Q}$, the code states are defined using a group $\mathcal{S} \subseteq \pm\{\mathbb{1}, X, Y, Z\}^{\otimes N} \subseteq \mathcal{P}_N$ of Pauli operators called the *stabilizer group* or simply the *stabilizer* of the code. By definition, the code states are eigenstates with eigenvalue $+1$ for each operator in $\mathcal{S}$:

$$\mathcal{Q} := \Big\{ |\psi\rangle : s\,|\psi\rangle = |\psi\rangle\,, \forall s \in \mathcal{S} \Big\}. \tag{4.1}$$

In order to prevent the degenerate case $\mathcal{Q} = \{0\}$, we require the group $\mathcal{S}$ to be abelian and to satisfy $-\mathbb{1} \notin \mathcal{S}$.

One advantage of stabilizer codes is the possibility to provide a compact representation: it is sufficient to write down a generating set of $\mathcal{S}$ to describe $\mathcal{Q}$. The elements of this generating set are called the *stabilizer generators* or simply the *generators* of the code and are the counter-part of the parity check equations of classical linear codes. For example, the quantum code called "the 5-qubit code" is stabilized by $\mathcal{S} = \langle g_1, g_2, g_3, g_4 \rangle$ where:

$$\begin{array}{rcccccccccc} g_1 & = & X & \otimes & Z & \otimes & Z & \otimes & X & \otimes & \mathbb{1} \\ g_2 & = & \mathbb{1} & \otimes & X & \otimes & Z & \otimes & Z & \otimes & X \\ g_3 & = & X & \otimes & \mathbb{1} & \otimes & X & \otimes & Z & \otimes & Z \\ g_4 & = & Z & \otimes & X & \otimes & \mathbb{1} & \otimes & X & \otimes & Z \end{array} \tag{4.2}$$

The dimension of the 5-qubits code is equal to 2 and an orthonormal basis of the code space is given by the states $|0_L\rangle$ and $|1_L\rangle$ defined by:

$$|0_L\rangle = \frac{1}{4}[|00000\rangle + |10010\rangle + |01001\rangle + |10100\rangle + |01010\rangle - |11011\rangle - |00110\rangle - |11000\rangle$$
$$- |11101\rangle - |00011\rangle - |11110\rangle - |01111\rangle - |10001\rangle - |01100\rangle - |10111\rangle + |00101\rangle]$$

$$|1_L\rangle = X^{\otimes 5}\,|0_L\rangle$$
$$= \frac{1}{4}[|11111\rangle + |01101\rangle + |10110\rangle + |01011\rangle + |10101\rangle - |00100\rangle - |11001\rangle - |00111\rangle$$
$$- |00010\rangle - |11100\rangle - |00001\rangle - |10000\rangle - |01110\rangle - |10011\rangle - |01000\rangle + |11010\rangle]$$

In particular, we can check that $|0_L\rangle$ and $|1_L\rangle$ are $+1$ eigenstates for g_1, g_2, g_3 and g_4 and form an orthonormal family.
The fact that the dimension of the 5-qubits code is equal to 2 is a consequence of Proposition 4.5: an N-qubit stabilizer code defined with M independent generators has dimension 2^K where $K = N - M$ is called the *number of logical qubits*. A code with N physical qubits and k logical qubits is said to be an $[[N, K]]$ stabilizer code or simply an $[[N, K]]$ code, for example the 5-qubit code is a $[[5, 1]]$ stabilizer code.

4.1.2 Parity check matrices and the symplectic representation of Pauli operators

Let $\mathcal{Q}$ be an $[[N, K]]$ code defined by $M \in \mathbb{N}$ stabilizer generators. These generators can be represented with a binary matrix H of size $M \times 2N$ called the *parity check matrix* of the code where, by definition, each row is the *symplectic representation* of one stabilizer generator as defined in Notation 4.1.

Notation 4.1 (Sympleptic representation of Pauli operators)**.** *Let $P = \alpha \bigotimes_{i=1}^{N} P_i \in \mathcal{P}_N$ be a Pauli operator on N qubits where $\alpha \in \{1, -1, i, -i\}$ and $P_i \in \{\mathbb{1}, X, Y, Z\}^{\otimes N}$. The symplectic representation of P denoted by $r(P) \in \mathbb{F}_2^{2N}$ is the concatenation of the two bit-strings $x \in \mathbb{F}_2^N$ and $z \in \mathbb{F}_2^N$ defined by:*

$$\begin{aligned} x_i = 0 \text{ and } z_i = 0 \quad &\text{if } P_i = \mathbb{1}, \\ x_i = 1 \text{ and } z_i = 0 \quad &\text{if } P_i = X, \\ x_i = 1 \text{ and } z_i = 1 \quad &\text{if } P_i = Y, \\ x_i = 0 \text{ and } z_i = 1 \quad &\text{if } P_i = Z. \end{aligned}$$

For instance, the four generators of the 5-qubit code shown in eq. (4.2) lead to the following parity check matrix:

$$\left(\begin{array}{ccccc|ccccc} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right).$$

When we define a stabilizer code with its parity check matrix, we implicitly state that the stabilizer generators belong to $\{\mathbb{1}, X, Y, Z\}^{\otimes N}$. Since, the symplectic representation is a bijection from $\{\mathbb{1}, X, Y, Z\}^{\otimes N}$ to $\mathbb{F}_2^{2N}$, each parity check matrix represents a unique stabilizer code. Note that the generators belong to $\{\mathbb{1}, X, Y, Z\}^{\otimes N}$ but the whole stabilizer group is included in $\pm\{\mathbb{1}, X, Y, Z\}^{\otimes N}$ and not necessarily in $\{\mathbb{1}, X, Y, Z\}^{\otimes N}$. The symplectic representation is not bijective from $\mathcal{P}_N$ to $\mathbb{F}_2^{2N}$: each $r \in \mathbb{F}_2^{2N}$ represents four matrices P, $-P$, iP and $-iP$. This ambiguity is not problematic because we will use the symplectic representation when the global phase in front of P is not relevant, for example when P is an error applied on a quantum state or when we wish to know whether P commutes with another Pauli operator.

Notation 4.2. *Let $P_1, P_2 \in \mathcal{P}_N$ be two Pauli matrices. We denote by $c(P_1, P_2) \in \mathbb{F}_2$ the bit defined by:*

$$c(P_1, P_2) = 0 \quad \textit{if } P_1 \textit{ and } P_2 \textit{ commute},$$
$$c(P_1, P_2) = 1 \quad \textit{if } P_1 \textit{ and } P_2 \textit{ anti-commute}.$$

Here the letter c stands for "commute".

In Notation 4.2, we emphasize the useful fact that two Pauli matrices either commute or anti-commute. Let $x \in \mathbb{F}_2^N$ and $z \in \mathbb{F}_2^N$ be two row vectors such that $r(P_2)$ is the concatenation of x and z, then $c(P_1, P_2)$ is equal to the inner product $r(P_1) \cdot r^T$ where r is the concatenation of z and x. In mathematics, the function c is called a *symplectic form*.

The symplectic representation is a surjective group homomorphism:

$$r(P_1 P_2) = r(P_1) \oplus r(P_2).$$

A family of stabilizer codes is said to be *LDPC* when there exist two integers $l, r \in \mathbb{N}$ such that the weight of the columns of the parity check matrices is upper bounded by l and the weight of their rows is upper bounded by r. LDPC codes are particularly advantageous for implementations because the measurement of a generator with weight r is done with a quantum measurement on r particles, thus r must be as small as possible.

4.1.3 Decoding algorithm

In this section, we discuss the way Pauli errors are corrected for stabilizer codes and in Section 4.1.4 we will address the case where the error is any CPTP map. Informally, a CPTP map can be written as a linear combination of Pauli errors. Section 4.1.4

formalizes this idea to show that an error correcting procedure which corrects any Pauli error acting on $T \in \mathbb{N}$ qubits corrects an arbitrary CPTP map acting on these qubits. Hence, as it is usual in quantum error correction, this manuscript will mostly focus on the case where the error is Pauli.

By the definition of a stabilizer code given in eq. (4.1), a code state $|\psi\rangle \in \mathcal{Q}$ is an eigenvector with eigenvalue $+1$ for the stabilizer generators. In particular, when we measure one of the generators on the state $|\psi\rangle$, we get $+1$ with probability 1. On the other hand, when we measure the generators $g_1, \dots, g_M$ on $U|\psi\rangle$ where U is some unitary representing an error, we get some sequence of eigenvalues in $\{-1, +1\}^M$. These M eigenvalues are generally reported as a sequence of bits called the *syndrome* $\sigma \in \mathbb{F}_2^M$, where a $+1$ eigenvalue is represented by the bit 0 and a -1 eigenvalue is represented by the bit 1. The syndrome contains information about the error applied on the code state and the goal of the error correction procedure is to infer the error from the syndrome.

When U is a general unitary, the quantum state $U|\psi\rangle$ is not necessarily an eigenvector of the generators, thus the value of the syndrome is not deterministic and the state collapses during the measurements. Nevertheless, in this section we focus on the case where U is a Pauli matrix. In this particular case, the state $U|\psi\rangle$ is indeed an eigenvector of the generators, see eq. (4.3).

Let $P \in \{\mathbb{1}, X, Y, Z\}^{\otimes N} \subseteq \mathcal{P}_N$ be a Pauli error, let $g \in \{g_1, \dots, g_M\}$ be a generator and let $|\psi\rangle \in \mathcal{Q}$ be a code state then $P|\psi\rangle$ is either a $+1$ or a -1 eigenstate of g:

$$\begin{aligned} &\text{If } gP = Pg \qquad \text{then} \qquad g\Big(P|\psi\rangle\Big) = P\Big(g|\psi\rangle\Big) = P|\psi\rangle\,. \\ &\text{If } gP = -Pg \qquad \text{then} \qquad g\Big(P|\psi\rangle\Big) = -P\Big(g|\psi\rangle\Big) = -P|\psi\rangle\,. \end{aligned} \tag{4.3}$$

When we measure the generators $g_1, \dots, g_M$ on the state $|\widetilde{\psi}\rangle := P|\psi\rangle$, the syndrome we get is called the *syndrome* of the error P (or the syndrome of the state $|\widetilde{\psi}\rangle$), it is denoted by $\sigma(P) \in \mathbb{F}_2^M$ or $\sigma\left(|\widetilde{\psi}\rangle\right)$ and its i^{th} bit is equal to:

$$\sigma(P)_i := \begin{cases} 0 & \text{if } Pg_i = g_iP, \\ 1 & \text{if } Pg_i = -g_iP. \end{cases} \quad \text{equivalently} \quad \sigma\left(|\widetilde{\psi}\rangle\right)_i := \begin{cases} 0 & \text{if } g_i|\widetilde{\psi}\rangle = |\widetilde{\psi}\rangle, \\ 1 & \text{if } g_i|\widetilde{\psi}\rangle = -|\widetilde{\psi}\rangle. \end{cases} \tag{4.4}$$

We emphasize that the syndrome of $|\widetilde{\psi}\rangle$ depends on P but does not depend on the initial code state $|\psi\rangle$. Let ρ be a density matrix then measuring the operator P corresponds to the CPTP map of eq. (2.3) implemented with the circuit on the left part of Figure 4.1. The circuit on the left part of Figure 4.1 uses the Hadamard gate H and the controlled-P gate C-P defined by:

$$H : \begin{cases} |0\rangle \mapsto \frac{1}{\sqrt{2}}\Big(|0\rangle + |1\rangle\Big) \\ |1\rangle \mapsto \frac{1}{\sqrt{2}}\Big(|0\rangle - |1\rangle\Big) \end{cases} \qquad C\text{-}P : \begin{cases} |0\rangle \otimes |\psi\rangle \mapsto |0\rangle \otimes |\psi\rangle \\ |1\rangle \otimes |\psi\rangle \mapsto |1\rangle \otimes \big(P|\psi\rangle\big) \end{cases}$$

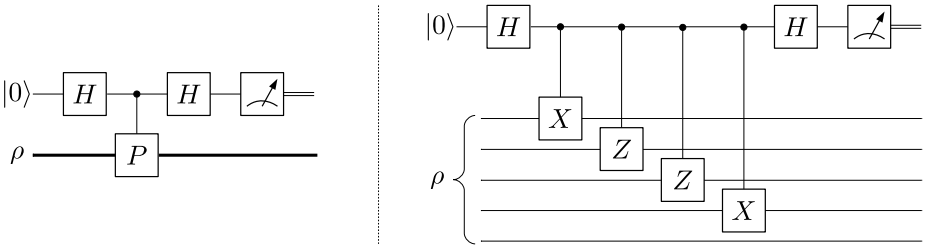

Figure 4.1: the circuit to measure a Pauli matrix P (on the left) and the circuit to measure the generator $g_1 = X \otimes Z \otimes Z \otimes X \otimes \mathbb{1}$ of the 5-qubit code (on the right).

The circuit proceeds in the following way:

- An ancilla qubit is prepared in the state $|0\rangle$.
- The Hadamard gate H is applied on the ancilla qubit.
- The controlled-P gate is applied.
- The Hadamard gate is applied on the ancilla qubit.
- The ancilla qubit is measured in the Z-basis (*i.e.* we perform the Pauli measurement associated to the Pauli matrix Z).

A *decoding algorithm* or *decoder* is a classical algorithm that takes as input a syndrome $\sigma \in \mathbb{F}_2^M$ and tries to guess the error P. Once the decoder has provided its output $\hat{P} \in \{\mathbb{1}, X, Y, Z\}^{\otimes N}$, the correction $\hat{P} = \hat{P}^\dagger$ is applied on the quantum state with the hope that $\hat{P}\,|\widetilde{\psi}\rangle = |\psi\rangle$ holds.

In the classical setting, a decoder succeeds when it identifies exactly the error, *i.e.* when $P = \hat{P}$. However, for a stabilizer code the success condition is relaxed: suppose we start from a code state $|\psi\rangle \in \mathcal{Q}$, that the error $P \in \mathcal{P}_N$ yields $|\widetilde{\psi}\rangle := P\,|\psi\rangle$ and that the decoder outputs $\hat{P} = sP$ where $s \in \mathcal{S}$ is in the stabilizer group. Under these assumptions, when we apply the correction $\hat{P}$ to $|\widetilde{\psi}\rangle$, we end up with the state $\hat{P}\,|\widetilde{\psi}\rangle = s\,|\psi\rangle = |\psi\rangle$. In this example, the decoder did not find the error but it was sufficient to correct the state. We say that two Pauli errors $P_1, P_2 \in \mathcal{P}_N$ are *equivalent* when they are equal up to a multiplication by a matrix of the stabilizer group $\mathcal{S}$, or equivalently when $P_1 P_2 \in \mathcal{S}$. In that case, P_1 and P_2 act in the same way on the code states:

$$\forall\,|\psi\rangle \in \mathcal{Q} : P_1\,|\psi\rangle = P_2\,|\psi\rangle\,.$$

We say that a stabilizer code is *degenerate* to mention the existence of two different Pauli errors which are equivalent.

If two Pauli errors P_1 and P_2 are equivalent then the syndrome of P_1 is equal to the syndrome of P_2 since these syndromes are both equal to the syndrome of $|\widetilde{\psi}\rangle = P_1\,|\psi\rangle = P_2\,|\psi\rangle$. However, the contraposition is false: there exist errors with the same syndrome but non equivalent. In fact, two Pauli errors P_1 and P_2 have the same

syndrome if and only if they satisfy $P_1 P_2^\dagger \in \mathcal{N}$ where $\mathcal{N} = \mathcal{N}(\mathcal{Q})$ is called the *normalizer group* (or simply the *normalizer*) of the code defined by:

$$\mathcal{N} := \Big\{ P \in \mathcal{P}_N : \sigma(P) = 0^M \Big\}.$$

In mathematics, for a group G, the normalizer of a subgroup $S \subseteq G$ is the subgroup $N \subseteq G$ containing the elements $g \in G$ which commute with all $s \in S$. Using eq. (4.4), a Pauli matrix is indeed in $\mathcal{N}$ if and only if it commutes with each stabilizer generators $g_1, \dots, g_M$. In addition, since $g_1, \dots, g_M$ generate the stabilizer group $\mathcal{S}$, a Pauli matrix is in $\mathcal{N}$ if and only if it commutes with each element of $\mathcal{S}$:

$$\mathcal{N} = \Big\{ P \in \mathcal{P}_N : \forall s \in \mathcal{S}, Ps = sP \Big\}, \quad \mathcal{N} = \Big\{ P \in \mathcal{P}_N : \forall i \in [\![1; m]\!], Pg_i = g_i P \Big\}.$$

Another interesting property is that the normalizer group fixes the code space and the stabilizer group fixes the code states:

$$\begin{aligned} &\forall n \in \mathcal{N} : && n\mathcal{Q} = \mathcal{Q}, \\ &\forall s \in \mathcal{S}, \forall \left|\psi\right\rangle \in \mathcal{Q} : && s \left|\psi\right\rangle = \left|\psi\right\rangle. \end{aligned}$$

For the 5-qubit code defined in eq. (4.2), the normalizer group is equal to:

$$\mathcal{N} = \langle \mathcal{S}, X_L, Z_L \rangle \text{ where } X_L := X^{\otimes 5} \text{ and } Z_L := Z^{\otimes 5}.$$

The operators X_L and Z_L satisfy:

$$X_L \left|0_L\right\rangle = \left|1_L\right\rangle, \quad X_L \left|1_L\right\rangle = \left|0_L\right\rangle, \quad Z_L \left|0_L\right\rangle = \left|0_L\right\rangle, \quad Z_L \left|1_L\right\rangle = -\left|1_L\right\rangle,$$

where $\left|0_L\right\rangle$ and $\left|1_L\right\rangle$ are the logical states of the 5-qubit code defined below eq. (4.2). The effect of the matrices X_L and Z_L on the logical states is similar to the effect of the usual Pauli matrices X and Z on the computational basis $\left|0\right\rangle$, $\left|1\right\rangle$ thus they are called *logical Pauli operators* of the code. In fact, we can talk about other *logical operations* such as a logical Hadamard or a logical controlled-not or a logical measurement. The logical operations allow us to process the information contained in the code states while keeping the protection of the error correcting code. This is essential to perform *fault-tolerant quantum computation* (see Chapter 6).

The minimal distance of an $[[N; K]]$ stabilizer code is the minimum weight of an error in $\mathcal{N} \setminus \mathcal{S}$.

$$D := \min \Big\{ \big|P\big| : P \in \mathcal{N} \setminus \mathcal{S} \Big\}. \tag{4.5}$$

This definition is meaningful because $\mathcal{N}$ is the set of undetectable Pauli errors and $\mathcal{S}$ is the set of Pauli errors which leave the code states invariant. Thus D is the minimum weight of an undetectable error which does not act trivially on the code space. When the minimal distance is specified, we say that the code is an $[[N, K, D]]$ stabilizer code.

To conclude this section, we provide the example of the *minimum weight decoder*, which, given a syndrome σ, returns a minimum weight error among the errors with syndrome σ:

$$\hat{P} := \underset{P\in\mathcal{P}_N:\sigma(P)=\sigma}{\arg\min} \big|P\big|. \tag{4.6}$$

This decoder have the advantage to perform well, but in general it cannot be implemented efficiently. A major exception is the polynomial time implementation for the 2-dimensional topological codes [30, 16, 17].

In this PhD thesis, we are interested in the hypergraph product codes, see Section 4.2, and for this specific code family we do not know how to implement the minimum weight decoding efficiently. In Section 4.3.1, we define the family of *quantum expander codes* which is a particular case of hypergraph product and can be decoded with the efficient *small-set-flip decoder*.

4.1.4 Error correction of non Pauli errors

In Section 4.1.3, we discussed the way Pauli errors can be corrected with stabilizer codes. In fact, the error correction procedure we used in that case also works for more general errors.

Proposition 4.3 (Theorem 10.2 of [82])**.** *Let $\mathcal{Q}$ be an $[[N,K]]$ stabilizer code such that any Pauli error of weight at most $T\in\mathbb{N}$ is corrected by a given error correction procedure EC:*

$$\forall\,|\psi\rangle\in\mathcal{Q}, \forall P\in\mathcal{P}_N \text{ with } \big|P\big|\leq T : EC\Big(P\,|\psi\rangle\Big) = |\psi\rangle\,.$$

If we consider a CPTP map with Kraus operators $\{E_1,\dots,E_K\}$ such that E_k acts on at most T qubits (the operators E_k may act on different qubits but each one acts on at most T qubits) then this CPTP map is corrected by EC:

$$\forall\,|\psi\rangle\in\mathcal{Q} : EC\left(\sum_{k=1}^{K} E_k\,|\psi\rangle\,\langle\psi|\,E_k^\dagger\right) = |\psi\rangle\,\langle\psi|\,.$$

The full proof of Proposition 4.3 can be found in Theorem 10.2 of [82]. Here, we focus on the particular case where the error correction procedure EC is the one described in Section 4.1.3 and we will only prove the case where the error is a unitary matrix acting on the first T qubits. Up to technical details, our proof can be generalized to the case where the error is a CPTP map with each Kraus operator acting on T qubits. However, our arguments are a bit different from the one used in [82] for the case where EC is not supposed to be the procedure described in Section 4.1.3.

Let $g_1,\dots,g_M$ be the stabilizer generators of $\mathcal{Q}$. The error correction procedure presented in Section 4.1.3 proceeds in three steps:

(i) Measure the stabilizer generators to get a syndrome $\sigma\in\mathbb{F}_2^M$.

(ii) Use a decoder to guess a Pauli correction $\hat{E}\in\mathcal{P}_N$.

(iii) Apply the correction $\hat{E}$ to the quantum state.

We denote by $\mathcal{M}$ the quantum operation corresponding to item (i) and by $\mathcal{C}$ the quantum operation corresponding to items (ii) and (iii). $\mathcal{M}$ takes as input a quantum register on N qubits and its output is composed of one classical register with M bits and one quantum register with N qubits. $\mathcal{C}$ takes as input a classical register on M bits and a quantum register on N qubits and its output is a quantum register on N qubits.
In order to measure one bit of the syndrome in the operation $\mathcal{M}$, we need to measure a generator which is a Pauli matrix $P \in \{\mathbb{1}, X, Y, Z\}^{\otimes N}$. This measurement can be done with the circuit presented in the left part of Figure 4.1. This circuit maps an arbitrary density matrix ρ to:

$$|0\rangle\langle 0| \otimes \left(\Pi_0 \rho \Pi_0^\dagger\right) + |1\rangle\langle 1| \otimes \left(\Pi_1 \rho \Pi_1^\dagger\right)$$

where $\Pi_0 = \frac{1}{2}(\mathbb{1} + P)$ is the orthogonal projector onto the $+1$ eigenspace of P and $\Pi_1 = \frac{1}{2}(\mathbb{1} - P)$ is the orthogonal projector onto the -1 eigenspace of P. More generally, when we measure the generators $g_1, \ldots, g_M$ with M copies of the circuit presented in Figure 4.1, the syndrome $\sigma \in \mathbb{F}_2^M$ is stored in a classical register and a projection Π_σ is applied on the quantum register:

$$\rho \mapsto \sum_{\sigma \in \mathbb{F}_2^M} |\sigma\rangle\langle\sigma| \otimes \left(\Pi_\sigma \rho \Pi_\sigma^\dagger\right) \qquad \text{where} \qquad \Pi_\sigma := \frac{1}{2^M} \prod_{i=1}^{M} \left(\mathbb{1} + (-1)^{\sigma_i} g_i\right).$$

The matrix Π_σ is an orthogonal projector onto the states with syndrome σ (for a proof of this point, see the proof of Proposition 4.5). We can check that for all $|\psi\rangle \in \mathcal{Q}$, $\sigma \in \mathbb{F}_2^M$ and $P \in \{\mathbb{1}, X, Y, Z\}^{\otimes N}$:

$$\begin{aligned} &\Pi_\sigma P = P \Pi_{\sigma \oplus \sigma(P)} \\ &\Pi_\sigma |\psi\rangle = 0 && \text{if } \sigma \neq 0^M \\ &\Pi_\sigma |\psi\rangle = |\psi\rangle && \text{if } \sigma = 0^M. \end{aligned} \tag{4.7}$$

Thus:

$$\begin{aligned} &\Pi_\sigma P |\psi\rangle = P |\psi\rangle && \text{if } \sigma(P) = \sigma \\ &\Pi_\sigma P |\psi\rangle = 0 && \text{if } \sigma(P) \neq \sigma. \end{aligned} \tag{4.8}$$

As a summary, the operations $\mathcal{M}$ and $\mathcal{C}$ satisfy:

$$\mathcal{M} : \rho \mapsto \sum_{\sigma \in \mathbb{F}_2^M} |\sigma\rangle\langle\sigma| \otimes \left(\Pi_\sigma \rho \Pi_\sigma^\dagger\right), \tag{4.9}$$

$$\mathcal{C} : |\sigma\rangle\langle\sigma| \otimes \rho \mapsto \hat{P}_\sigma \rho \hat{P}_\sigma^\dagger \tag{4.10}$$

where $\hat{P}_\sigma \in \{\mathbb{1}, X, Y, Z\}^{\otimes N}$ is the Pauli matrix computed by the decoder on input σ.

We are now ready to show that if $\mathcal{C} \circ \mathcal{M}$ corrects any Pauli error acting on the first $T \in \mathbb{N}$ qubits then $\mathcal{C} \circ \mathcal{M}$ also corrects an arbitrary unitary acting on the first T qubits.

We denote by $\mathcal{P}_{N,T} := \{\mathbb{1}, X, Y, Z\}^{\otimes T} \otimes \mathbb{1}^{\otimes N-T}$ the set of Pauli matrices acting on the first T qubits.
We assume that these Pauli errors are corrected by the error correction procedure, *i.e.* for all $P \in \mathcal{P}_{N,T}$ and for all $|\psi\rangle \in \mathcal{Q}$:

$$\mathcal{C} \circ \mathcal{M}\Big[P |\psi\rangle \langle\psi| P^\dagger\Big] = |\psi\rangle \langle\psi| .$$

For simplicity, we replace the density matrices with pure states:

$$\mathcal{C} \circ \mathcal{M}\Big[P |\psi\rangle\Big] = |\psi\rangle . \tag{4.11}$$

First, eq. (4.11) implies that two Pauli errors with the same syndrome act in the same way on the code states, *i.e.* for all $P_1, P_2 \in \mathcal{P}_{N,T}$:

$$\sigma(P_1) = \sigma(P_2) \Rightarrow \forall |\psi\rangle \in \mathcal{Q} : P_1 |\psi\rangle = P_2 |\psi\rangle . \tag{4.12}$$

Indeed, let $\sigma = \sigma(P_1) = \sigma(P_2)$ then using eqs. (4.8) to (4.11):

$$\hat{P}_\sigma P_1 |\psi\rangle = \mathcal{C} \circ \mathcal{M}\Big[P_1 |\psi\rangle\Big] = |\psi\rangle = \mathcal{C} \circ \mathcal{M}\Big[P_2 |\psi\rangle\Big] = \hat{P}_\sigma P_2 |\psi\rangle .$$

Hence we have $P_1 |\psi\rangle = P_2 |\psi\rangle$ because $\hat{P}_\sigma$ is invertible.

When we deal with general errors, the key observation is that the operators Π_σ collapse arbitrary errors onto Pauli errors. For instance, let $\rho = U |\psi\rangle \langle\psi| U^\dagger$ where $U = U_0 \otimes \mathbb{1}^{\otimes N-T}$ and U_0 is a unitary complex matrix of size $2^T \times 2^T$ acting on the first T qubits. The family $\{\mathbb{1}, X, Y, Z\}^{\otimes T}$ is a basis for the linear space of complex matrices thus we can write U as:

$$U = \sum_{P \in \mathcal{P}_{N,T}} \alpha_P P \qquad \text{where } \alpha_P \in \mathbb{C}.$$

As a consequence, the state ρ can be written as:

$$\rho = \sum_{P_1, P_2 \in \mathcal{P}_{N,T}} \alpha_{P_1} \alpha^*_{P_2} P_1 |\psi\rangle \langle\psi| P_2^\dagger \tag{4.13}$$

We have:

$$\begin{aligned}
\mathcal{M}[\rho] &= \sum_{\sigma \in \mathbb{F}_2^M} |\sigma\rangle \langle\sigma| \otimes \left[\sum_{P_1, P_2 \in \mathcal{P}_{N,T}} \alpha_{P_1} \alpha^*_{P_2} \Pi_\sigma P_1 |\psi\rangle \langle\psi| P_2^\dagger \Pi_\sigma^\dagger \right] && \text{by eqs. (4.9) and (4.13)} \\
&= \sum_{\sigma \in \mathbb{F}_2^M} |\sigma\rangle \langle\sigma| \otimes \left[\sum_{\substack{P_1, P_2 \in \mathcal{P}_{N,T}: \\ \sigma = \sigma(P_1) = \sigma(P_2)}} \alpha_{P_1} \alpha^*_{P_2} P_1 |\psi\rangle \langle\psi| P_2^\dagger \right] && \text{by eq. (4.8)} \\
&= \sum_{\sigma \in \mathbb{F}_2^M} \beta_\sigma |\sigma\rangle \langle\sigma| \otimes \left[P_\sigma |\psi\rangle \langle\psi| P_\sigma^\dagger \right] && \text{by eq. (4.12)}
\end{aligned}$$

where $P_\sigma \in \mathcal{P}_{N,T}$ is chosen such that $\sigma(P_\sigma) = \sigma$ and:

$$\beta_\sigma = \sum_{\substack{P_1,P_2\in\mathcal{P}_{N,T}:\\ \sigma=\sigma(P_1)=\sigma(P_2)}} \alpha_{P_1}\alpha^*_{P_2}$$

Note that P_σ is undefined when $\left\{P \in \mathcal{P}_{N,T} : \sigma(P) = \sigma\right\}$ is empty. However in that case, $\beta_\sigma = 0$ thus we can set P_σ arbitrarily, for example $P_\sigma = 0$.
By hypothesis, $P_\sigma \in \mathcal{P}_{N,T}$ is corrected by $\mathcal{C} \circ \mathcal{M}$ and thus we have:

$$\mathcal{C} \circ \mathcal{M}(\rho) = \left(\sum_{\sigma\in\mathbb{F}_2^M} \beta_\sigma\right) |\psi\rangle \langle\psi| .$$

The CPTP map $\mathcal{C} \circ \mathcal{M}$ is trace preserving thus:

$$\mathcal{C} \circ \mathcal{M}(\rho) = |\psi\rangle \langle\psi| .$$

The conclusion is that $\mathcal{C} \circ \mathcal{M}$ corrects the error U.

4.1.5 Dimension of a stabilizer code

The goal of this section is to prove the formula for the dimension of a stabilizer code given in Proposition 4.5. Let $g_1, \ldots, g_M \in \mathcal{P}_N$ be commuting Pauli operators. We define a stabilizer group $\mathcal{S} = \langle g_1, \ldots, g_M\rangle$ and $\mathcal{Q}$ the corresponding quantum code as in eq. (4.1). First, since we have assumed $-\mathbb{1} \notin \mathcal{S}$ to avoid the degenerate case $\mathcal{Q} = \{0\}$, we must have $g_i \in \pm\{\mathbb{1}, X, Y, Z\}^{\otimes N}$ to ensure $g_i^2 \neq -\mathbb{1}$. Second, if one of the generators is equal to the product of other generators then we can freely remove it without changing $\mathcal{S}$. Hence we assume that the generators are independent:

$$\forall v \in \mathbb{F}_2^M : \prod_{i=1}^M g_i^{v_i} = \mathbb{1} \Rightarrow v = 0^M. \tag{4.14}$$

As a summary, without loss of generality we assume that the stabilizer generators $g_1, \ldots, g_M$ belong to $\pm\{\mathbb{1}, X, Y, Z\}^{\otimes N}$, are independent in the sense of eq. (4.14) and satisfy:

$$g_i^2 = \mathbb{1}, \qquad g_i = g_i^\dagger. \tag{4.15}$$

The first step to compute the dimension of a stabilizer code is to understand how $\mathcal{Q}$ is modified when g_i is replaced by $-g_i$. Let $\sigma \in \mathbb{F}_2^M$ be a syndrome. Then the set of quantum states with syndrome σ is the code space of the quantum code whose stabilizer generators are $(-1)^{\sigma_1} g_1, \ldots, (-1)^{\sigma_M} g_M$:

$$\mathcal{Q}_\sigma := \left\{ |\psi\rangle : (-1)^{\sigma_i} g_i |\psi\rangle = |\psi\rangle , i = 1, \ldots, M\right\}. \tag{4.16}$$

For example we have $\mathcal{Q} = \mathcal{Q}_\sigma$ when $\sigma = 0^M$. The stabilizer code $\mathcal{Q}_\sigma$ does depend on σ but all these codes are similar because, as shown in Lemma 4.4, they are equal up to a multiplication by a Pauli matrix.

Lemma 4.4. *Let $g_1, \dots, g_M \in \mathcal{P}_N$ be independent, i.e. satisfying eq. (4.14), and commuting Pauli matrices such that $-\mathbb{1} \notin \langle g_1, \dots, g_M \rangle$. Let $\sigma, \sigma' \in \mathbb{F}_2^M$ be two syndromes and let $\mathcal{Q}_\sigma, \mathcal{Q}_{\sigma'}$ as defined in eq. (4.16). Then:*

$$\mathcal{Q}_{\sigma'} = P\mathcal{Q}_\sigma \text{ where } P \in \mathcal{P}_N.$$

Proof. First, using the independence of the generators, we show that the parity check matrix H of $\mathcal{Q}$ is full rank. For that, we show that H^T is injective using the symplectic representation r defined in Notation 4.1. Let $v \in \mathbb{F}_2^M$ be such that $H^T v = 0^{2N}$ then:

$$r(\mathbb{1}) = 0^{2N} = H^T v = \bigoplus_{i=1}^{M} v_i r(g_i) = r\left(\prod_{i=1}^{M} g_i^{v_i}\right).$$

The homomorphism r is not injective but nonetheless $\prod_{i=1}^M g_i^{v_i} = \lambda \mathbb{1}$ where $\lambda \in \{1, -1, i, -i\}$ and we get $\lambda = 1$ using the hypothesis $-\mathbb{1} \notin \langle g_1, \dots, g_M \rangle$. By eq. (4.14), we conclude that $v = 0^M$ and thus H is full rank.

Since H is full rank, there exist $x, z \in \mathbb{F}_2^N$ such that $He = \sigma \oplus \sigma'$ where $e = (z, x) \in \mathbb{F}_2^{2N}$ is the concatenation of z and x. We define the Pauli matrix P promised by Lemma 4.4 to be such that $r(P) = (x, z)$.
For $i \in [\![1, M]\!]$, the i^{th} bit of He is equal to the inner product $r(g_i) \cdot e^T$ and thus we have $c(g_i, P) = r(g_i) \cdot e^T = \sigma_i \oplus \sigma'_i$ where $c(g_i, P)$ defined in Notation 4.2 satisfies:

$$g_i P = (-1)^{c(g_i, P)} P g_i.$$

By the definition of $\mathcal{Q}_\sigma$ and $\mathcal{Q}_{\sigma'}$, we have $P\mathcal{Q}_\sigma \subseteq \mathcal{Q}_{\sigma'}$ and $P^\dagger \mathcal{Q}_{\sigma'} = P\mathcal{Q}_{\sigma'} \subseteq \mathcal{Q}_\sigma$. □

We conclude this section with Proposition 4.5 which states that the dimension of a stabilizer code is 2^{N-M} where M is the number of independent generators.

Proposition 4.5. *Let $g_1, \dots, g_M \in \mathcal{P}_N$ be independent,i.e. satisfying eq. (4.14), and commuting Pauli matrices such that $-\mathbb{1} \notin \langle g_1, \dots, g_M \rangle$.*
The associated quantum code $\mathcal{Q}$ has $K := N - M$ logical qubits and the physical space $\left(\mathbb{C}^2\right)^{\otimes N}$ is equal to the following orthogonal direct sum ($\oplus$ is the orthogonal direct sum of linear spaces):

$$\left(\mathbb{C}^2\right)^{\otimes N} = \bigoplus_{\sigma \in \mathbb{F}_2^M} \mathcal{Q}_\sigma.$$

Proof. For $\sigma \in \mathbb{F}_2^M$, we define Π_σ by:

$$\Pi_\sigma := \frac{1}{2^M} \prod_{i=1}^{M} \left(\mathbb{1} + (-1)^{\sigma_i} g_i\right).$$

With eq. (4.15) and the definition of Π_σ above, we have:

$$\Pi_\sigma \Pi_\sigma = \Pi_\sigma \qquad \Pi_\sigma^\dagger = \Pi_\sigma \qquad \mathrm{Im}(\Pi_\sigma) = \mathcal{Q}_\sigma.$$

Hence Π_σ is the orthogonal projector onto $\mathcal{Q}_\sigma$. Similarly, for any $i \in [\![1, M]\!]$:

$$\Big(\mathbb{1} + g_i\Big)\Big(\mathbb{1} - g_i\Big) = \Big(\mathbb{1} - g_i\Big)\Big(\mathbb{1} + g_i\Big) = 0.$$

Thus for $\sigma \neq \sigma'$, we get $\Pi_\sigma \Pi_{\sigma'} = 0$ and the two subspaces $\mathcal{Q}_\sigma$ and $\mathcal{Q}_{\sigma'}$ are orthogonal. Finally, as stated in Proposition 4.5, we have $\left(\mathbb{C}^2\right)^{\otimes N} = \bigoplus_\sigma \mathcal{Q}_\sigma$ because:

$$\sum_{\sigma \in \mathbb{F}_2^M} \Pi_\sigma = \frac{1}{2^M} \prod_{i=1}^{M} \sum_{\sigma_i \in \mathbb{F}_2} \Big(\mathbb{1} + (-1)^{\sigma_i} g_i\Big) = \mathbb{1}$$

and thus for all $|\psi\rangle \in \left(\mathbb{C}^2\right)^{\otimes N}$, $|\psi\rangle = \sum_\sigma \Pi_\sigma |\psi\rangle$.

For the dimension of $\mathcal{Q}$, we already know that $2^N = \dim\left(\left(\mathbb{C}^2\right)^{\otimes N}\right) = \sum_\sigma \dim(\mathcal{Q}_\sigma)$. By Lemma 4.4, the dimension of $\mathcal{Q}_\sigma$ does not depend on σ and thus $\dim(\mathcal{Q}) = 2^{N-M}$. □

4.1.6 CSS codes

In 1996, a broad class of quantum codes called the Calderbank-Shor-Steane codes, or *CSS* codes for short, was introduced independently by Calderbank and Shor in [20], and by Steane in [102]. Even though the CSS codes were defined before the stabilizer codes, it is convenient to use the stabilizer formalism to describe them.
A CSS code is a stabilizer code where each stabilizer generator is either a product of X-Pauli matrices or a product of Z-Pauli matrices:

$$\begin{aligned} &g_i \in \{\mathbb{1}, X\}^{\otimes N} && \text{for } i = 1, \dots, M_Z, \\ &g_{i+M_Z} \in \{\mathbb{1}, Z\}^{\otimes N} && \text{for } i = 1, \dots, M_X. \end{aligned}$$

The generators equal to a product of X-Pauli matrices are called the *X-type generators* and those equal to a product of Z-Pauli matrices are called the *Z-type generators*.
Let $\mathcal{C}_X$ and $\mathcal{C}_Z$ be two classical codes satisfying $\mathcal{C}_Z^\perp \subseteq \mathcal{C}_X$ (or equivalently $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$) then we can define a CSS code denoted by $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ in the following manner. Let H_X be a parity check matrix for $\mathcal{C}_X$ with size $M_X \times N$ and let H_Z be a parity check matrix for $\mathcal{C}_Z$ with size $M_Z \times N$. Then $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ is the stabilizer code with $M = M_X + M_Z$ generators given by the following parity check matrix:

$$H = \left(\begin{array}{c|c} H_Z & 0 \\ \hline 0 & H_X \end{array}\right). \tag{4.17}$$

When we talk about CSS codes in this PhD thesis, the subscript X is used to indicate that the code $\mathcal{C}_X$ corrects the X-Pauli errors and the subscript Z is used to indicate that $\mathcal{C}_Z$ corrects the Z-Pauli errors. Hence the generators associated to $\mathcal{C}_X$ are Z-type generators and the generators associated to $\mathcal{C}_Z$ are X-type generators. Some authors take the converse convention.

One nice feature of CSS codes is the possibility to handle them with the formalism of classical error correcting codes. Since the generators of a CSS code commute and are either X-type or Z-type generators, the stabilizer group $\mathcal{S}$ is included in $\mathcal{S}_0$ where:

$$\mathcal{S}_0 := \Big\{ g_X g_Z : g_X \in \{\mathbb{1}, X\}^{\otimes N}, g_Z \in \{\mathbb{1}, Z\}^{\otimes N} \Big\}. \tag{4.18}$$

The symplectic representation defined in Notation 4.1 restricted to $\mathcal{S}_0$ is a bijection $r : \mathcal{S}_0 \to \mathbb{F}_2^{2N}$. Hence for all $P \in \mathcal{S}_0$ and for all $v \in \mathbb{F}_2^M$, we have the equivalence:

$$P = \prod_{i=1}^{M} g_i^{v_i} \Leftrightarrow r(P) = \bigoplus_{i=1}^{M} v_i r(g_i). \tag{4.19}$$

As a consequence, the elements of the stabilizer group can be seen as binary vectors and CSS codes are described with linear algebra. For example, a product of X-type generators has symplectic representation $(e, 0^N)$ where $e \in \mathcal{C}_Z^\perp$:

$$\mathcal{C}_Z^\perp \times \{0^N\} = \Big\{ r(s) : s \in \mathcal{S} \cap \{\mathbb{1}, X\}^{\otimes N} \Big\}. \tag{4.20}$$

As discussed in Section 4.1.1 and Proposition 4.5, defining a stabilizer code with commuting generators and $-\mathbb{1} \notin \langle g_1, \dots, g_M \rangle$ is necessary and sufficient to avoid the degenerate case $\mathcal{Q} = \{0\}$. For a CSS code, the condition that the generators commute is equivalent to $H_X H_Z^T = 0$ and is ensured by the requirement $\mathcal{C}_Z^\perp \subseteq \mathcal{C}_X$. The other condition $-\mathbb{1} \notin \langle g_1, \dots, g_M \rangle$ always holds for a CSS code because $-\mathbb{1} \notin \mathcal{S}_0$.

When the Z-type generators of a CSS code are measured, we get the first part of the syndrome $\sigma_X \in \mathbb{F}_2^{M_X}$ which gives information about the X-type errors on the state. Similarly, the Z-type errors are detected with the syndrome $\sigma_Z \in \mathbb{F}_2^{M_Z}$ obtained when we measure the X-type generators. The entire syndrome $\sigma \in \mathbb{F}_2^M$ is the concatenation of these two bit-strings $\sigma = (\sigma_X, \sigma_Z)$. In this PhD thesis, we focus on CSS codes and we correct X-type and Z-type errors independently. Therefore we need two decoders: the first one takes as input the syndrome σ_X and uses the code $\mathcal{C}_X$ to infer an X-type correction $\hat{P} \in \{\mathbb{1}, X\}^{\otimes N}$ often seen as a bit-string $\hat{e}_X \in \mathbb{F}_2^N$, and the second decoder proceeds in the same way for Z-type errors. Note that if a decoder corrects both X and Z errors on some qubit then the error $Y = iXZ$ is also corrected. Thereby, the decoding problem for CSS codes is close to the one for classical codes and thus we will often use the terminology from classical error correction in the quantum setting. For example, the Z-type generators will be sometimes called the *check-nodes*. Note that for many error models such as the depolarizing channel, the correlations between X and Z errors can be used to improve the decoders [29, 81, 88].

Nonetheless, since some CSS codes are degenerate, they cannot be corrected in the same way as classical codes. The degeneracy of a stabilizer code implies that it is sufficient to correct the error up to an element of the stabilizer group, for example the X-type error can be corrected up to a product of X-type generators. Let $e_X \in \mathbb{F}_2^N$ be an X-type error and let $\hat{e}_X \in \mathbb{F}_2^N$ be the correction that returns the decoder. Using the bijection given in eq. (4.20) between product of X-type generators and the vector space $\mathcal{C}_Z^\perp$, the decoder succeeds in correcting the X-error if and only if $e_X \oplus \hat{e}_X \in \mathcal{C}_Z^\perp$.

Taking into account the degeneracy is particularly important for LDPC codes since the vector space $\mathcal{C}_Z^\perp$ contains constant weight elements.

For a stabilizer code, remember from Section 4.1.3 that the minimal distance is the minimal weight of an element in $\mathcal{N} \setminus \mathcal{S}$, where $\mathcal{N} \setminus \mathcal{S}$ is the set of a Pauli errors with an empty syndrome acting non-trivially on the code state. For a CSS code, there is always an X-type error or a Z-type error in $\mathcal{N} \setminus \mathcal{S}$ with weight equal to the minimal distance. We define D_X to be the minimal weight of an X-type error in $\mathcal{N} \setminus \mathcal{S}$. Using the terminology of classical error correction, D_X is the minimal weight of a codeword in $\mathcal{C}_X$ which does not belong to $\mathcal{C}_Z^\perp$. The minimal distance for Z-type errors is defined in a similar manner:

$$D_X := \min\left\{\left|e_X\right| : e_X \in \mathcal{C}_X \setminus \mathcal{C}_Z^\perp\right\}, \qquad D_Z := \min\left\{\left|e_Z\right| : e_Z \in \mathcal{C}_Z \setminus \mathcal{C}_X^\perp\right\}.$$

Note that D_X and D_Z are not the minimal distances of the codes $\mathcal{C}_X$ and $\mathcal{C}_Z$. Finally, the minimal distance of $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ is equal to:

$$D = \min(D_X, D_Z). \tag{4.21}$$

The classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ can be represented by their Tanner graphs which must have the same number of bit-nodes. The *Tanner graph* for the code $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ has three types of nodes: the bits-nodes, the check-nodes of $\mathcal{C}_X$ (which represent the Z-type generators) and the check-nodes of $\mathcal{C}_Z$ (which represent the X-type generators). A bit-node and an Z-type generator (resp. X-type generator) are connected if and only if they are connected in the Tanner graph of $\mathcal{C}_X$ (resp. $\mathcal{C}_Z$). For example when both $\mathcal{C}_X$ and $\mathcal{C}_Z$ are equal to the $[7, 3, 3]$ Hamming code defined in Figure 3.1, we get a CSS code called *the 7-qubit code* or *the Steane code* whose parity check matrix is given in Figure 4.2 and whose factor graph is given in Figure 4.3.

$$\left(\begin{array}{ccccccc|ccccccc}
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right)$$

Figure 4.2: Parity check matrix for the 7-qubit code.

The CSS codes we are interested in are the *quantum expander codes* which can be decoded with the *small-set-flip decoder* (see Section 4.3.1). In that case, the Tanner graphs of $\mathcal{C}_X$ and $\mathcal{C}_Z$ are isomorphic and thus the correction of X-errors is similar to the correction of Z-errors. In particular, the decoding algorithm is the same for both types of errors and we will describe it only for X-errors. An X-type error can be seen either as a classical error $e_X \in \mathbb{F}_2^N$ or as a subset of the qubits $E_X \subseteq V_\mathcal{Q}$ where $V_\mathcal{Q}$ is the set of qubits. For quantum expander codes, it is convenient to use the representation E_X and to see the syndrome as a subset $\sigma_X \subseteq C_X$ where C_X is the set of Z-type generators.

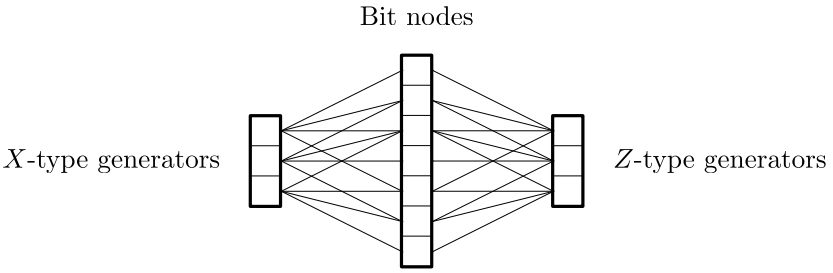

Figure 4.3: Factor graph for the 7-qubit code.

To summarize, the small-set-flip decoder is a classical algorithm which takes as input a syndrome $\sigma \subseteq C_X$ and returns a subset of the qubits $\hat{E} \subseteq V_{\mathcal{Q}}$.

Proposition 4.6 provides the formula to compute the dimension of a CSS code from the dimension of the underlying classical codes.

Proposition 4.6. *Let $\mathcal{C}_X$ be a classical $[N, k_X]$ code and $\mathcal{C}_Z$ be an $[N, k_Z]$ code with $\mathcal{C}_Z^\perp \subseteq \mathcal{C}_X$.*
The quantum code $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ is an $[[N, K]]$ stabilizer code where $K = k_X + k_Z - N$.

Proof. We would like to use Proposition 4.5 but the generators $g_1, \ldots, g_M$ are not necessarily independent. A consequence of eq. (4.20) is that the number of independent generators in $g_1, \ldots, g_M$ is equal to the number of independent rows in the parity check matrix H. By the rank nullity theorem, the parity check matrix H_Z has $M'_Z = N - k_Z$ independent rows and the parity check matrix H_X has $M'_X = N - k_X$ independent rows. Thus the number of independent rows in H is equal to $M' = M'_X + M'_Z = 2N - k_X - k_Z$.
By Proposition 4.5: $K = N - M' = k_X + k_Z - N$. □

In [20] and [102], the CSS codes were introduced by giving an explicit orthogonal basis for the code space. For $x \in \mathcal{C}_X$, we define:

$$|x \oplus \mathcal{C}_Z^\perp\rangle := \frac{1}{\sqrt{\mathcal{C}_Z^\perp}} \sum_{y \in \mathcal{C}_Z^\perp} |x \oplus y\rangle \,.$$

Lemma 4.7. *The set $\left\{ |x + \mathcal{C}_Z^\perp\rangle : x \in \mathcal{C}_X \right\}$ is an orthogonal basis for the code space of $CSS(\mathcal{C}_X, \mathcal{C}_Z)$.*

Proof. Let $\mathcal{Q}$ be the quantum code spanned by the states $|x \oplus \mathcal{C}_Z^\perp\rangle$:

$$\mathcal{Q} := \text{span}\Big\{ |x \oplus \mathcal{C}_Z^\perp\rangle : x \in \mathcal{C}_X \Big\}.$$

The strategy to show the equality $CSS(\mathcal{C}_X, \mathcal{C}_Z) = \mathcal{Q}$ is to prove that $\mathcal{Q} \subseteq CSS(\mathcal{C}_X, \mathcal{C}_Z)$ and then that these two vector spaces have both dimension equal to $N - k_X - k_Z$.

The statement $\mathcal{Q} \subseteq CSS(\mathcal{C}_X, \mathcal{C}_Z)$ means that the states $|x \oplus \mathcal{C}_Z^\perp\rangle$ are $+1$ eigenstates for the generators of $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ specified by the parity check matrix of eq. (4.17). Let $h \in \mathbb{F}_2^N$ be a row of H_Z then the corresponding X-type generator is

$$g = \bigotimes_{i=1}^{N} X^{h_i}$$

and:

$$g\,|x \oplus \mathcal{C}_Z^\perp\rangle = \sum_{y \in \mathcal{C}_Z^\perp} g\,|x \oplus y\rangle = \sum_{y \in \mathcal{C}_Z^\perp} |x \oplus y \oplus h\rangle = \sum_{y' \in \mathcal{C}_Z^\perp} g\,|x \oplus y'\rangle = |x \oplus \mathcal{C}_Z^\perp\rangle\,.$$

Note that in the previous equalities and in what follows, we omit the global phase $1/\sqrt{\mathcal{C}_Z^\perp}$ of the definition of $|x \oplus \mathcal{C}_Z^\perp\rangle$.
Let $h \in \mathbb{F}_2^N$ be a row of H_X then the corresponding Z-type generator is:

$$g = \bigotimes_{i=1}^{N} Z^{h_i}$$

and:

$$g\,|x \oplus \mathcal{C}_Z^\perp\rangle = \sum_{y \in \mathcal{C}_Z^\perp} g\,|x \oplus y\rangle = \sum_{y \in \mathcal{C}_Z^\perp} (-1)^{(x \oplus y)\cdot h}\,|x \oplus y\rangle = \sum_{y \in \mathcal{C}_Z^\perp} |x \oplus y\rangle = |x \oplus \mathcal{C}_Z^\perp\rangle\,.$$

Hence $\mathcal{Q} \subseteq CSS(\mathcal{C}_X, \mathcal{C}_Z)$ holds.

In order to compute the dimension of $\mathcal{Q}$, we show that $\left\{\, |x \oplus \mathcal{C}_Z^\perp\rangle : x \in \mathcal{C}_X \right\}$ is an orthonormal basis of $\mathcal{Q}$. Let $x, x' \in \mathcal{C}_X$, we distinguish between two cases:

- If $x \oplus x' \in \mathcal{C}_Z^\perp$ then $|x \oplus \mathcal{C}_Z^\perp\rangle = |x' \oplus \mathcal{C}_Z^\perp\rangle$ because:

$$\begin{aligned} |x \oplus \mathcal{C}_Z^\perp\rangle &= \sum_{y \in \mathcal{C}_Z^\perp} |x \oplus y\rangle = \sum_{y' \in \mathcal{C}_Z^\perp} |x \oplus y' \oplus x \oplus x'\rangle \\ &= \sum_{y' \in \mathcal{C}_Z^\perp} |x' \oplus y'\rangle = |x' \oplus \mathcal{C}_Z^\perp\rangle\,. \end{aligned}$$

- If $x \oplus x' \notin \mathcal{C}_Z^\perp$ then for any $y, y' \in \mathcal{C}_Z^\perp : x \oplus y \oplus x' \oplus y' \notin \mathcal{C}_Z^\perp$. In particular $x \oplus y \neq x' \oplus y'$ and thus the vectors $|x \oplus y\rangle$ and $|x' \oplus y'\rangle$ are orthogonal. Finally, the states $|x \oplus \mathcal{C}_Z^\perp\rangle$ and $|x' \oplus \mathcal{C}_Z^\perp\rangle$ are orthogonal since the two sums that define them are orthogonal term by term.

The dimension of $\mathcal{Q}$ is equal to:

$$\dim(\mathcal{Q}) = \#\left\{\, |x \oplus \mathcal{C}_Z^\perp\rangle : x \in \mathcal{C}_X \right\} = \frac{|\mathcal{C}_X|}{|\mathcal{C}_Z^\perp|} = \frac{2^{k_X}}{2^{N-k_Z}} = 2^{k_X + k_Z - N}.$$

Using Proposition 4.6, we conclude that $\mathcal{Q}$ and $CSS(\mathcal{C}_X, \mathcal{C}_Z)$ have the same dimension and thus they are equal. □

4.2 Hypergraph product codes

The *hypergraph product* construction introduced by Tillich and Zémor in ref. [108] is a generic way to build a quantum code $\mathcal{Q}$ from two classical codes $\mathcal{C}_1$ and $\mathcal{C}_2$. Let G_1, G_2 be the Tanner graphs of $\mathcal{C}_1, \mathcal{C}_2$ then the associated hypergraph product is a CSS code whose Tanner graph is equal to the so-called *Cartesian product* of G_1 and G_2. This construction provides quantum LDPC codes with constant rate and a minimal distance proportional to the square root of the block length. For instance, when $\mathcal{C}_1$ and $\mathcal{C}_2$ are both equal to the same $[n, \Theta(n), \Theta(n)]$ LDPC code, $\mathcal{Q}$ is an $[[N, \Theta(N), \Theta(\sqrt{N})]]$ LDPC CSS code where $N = \Theta(n^2)$.

As shown in Figure 2.6, the asymptotic scaling for the hypergraph product code parameters is good by comparison to the other known families of LDPC codes. The technical idea to compute the dimension and the minimal distance of an hypergraph product code is to rely on two notions from classical coding called *transpose code* and *classical product code*.

Let $\mathcal{C}$ be a classical code with parity check matrix H. Then, by definition, the transpose code of $\mathcal{C}$ denoted by $\mathcal{C}^T$ has parity check matrix H^T. In other words, the Tanner graph of $\mathcal{C}^T$ is equal to the Tanner graph of $\mathcal{C}$ where the bit-nodes are interpreted as check-nodes and vice-versa.

Classical product codes have been introduced by Elias in ref. [35]. Given $\mathcal{C}_1$ an $[n_1, k_1, d_1]$ code and $\mathcal{C}_2$ an $[n_2, k_2, d_2]$ code, the associated product code is an $[n_1 n_2, k_1 k_2, d_1 d_2]$ that we denote by $\mathcal{C}$. Informally, the physical bits of $\mathcal{C}$ are organized accordingly to an $n_1 \times n_2$ rectangle and a bit-string $c \in \mathbb{F}_2^{n_1 \times n_2}$ is a codeword of $\mathcal{C}$ if and only if each column of c is a codeword of $\mathcal{C}_1$ and each row of c is a codeword of $\mathcal{C}_2$. For completeness, we provide a quick survey on classical product codes in Chapter 7.

Finally, the hypergraph product associated to $\mathcal{C}_1$ and $\mathcal{C}_2$ is the CSS code built from $\mathcal{C}_X$, $\mathcal{C}_Z$ where $\mathcal{C}_X^T$ is the classical product of $\mathcal{C}_1, \mathcal{C}_2^T$ and $\mathcal{C}_Z^T$ is the classical product of $\mathcal{C}_1^T$, $\mathcal{C}_2$.

The organization of this section is the following: Section 4.2.1 defines the hypergraph product construction and Section 4.2.2 is a review on their main properties.

4.2.1 Definition of hypergraph product codes

Let $\mathcal{C}_1$ be an $[n_1, k_1, d_1]$ classical code with parity check matrix $H_1 \in \mathbb{F}_2^{m_1 \times n_1}$ and let $\mathcal{C}_2$ be an $[n_2, k_2, d_2]$ classical code with parity check matrix $H_2 \in \mathbb{F}_2^{m_2 \times n_2}$. The hypergraph product of $\mathcal{C}_1, \mathcal{C}_2$ denoted by $\mathcal{Q}$ is defined to be the CSS code associated to the classical codes $\mathcal{C}_X, \mathcal{C}_Z$ described by the following parity check matrices:

$$H_X = \left(\mathbb{1}_{n_1} \otimes H_2, H_1^T \otimes \mathbb{1}_{m_2}\right), \qquad H_Z = \left(H_1 \otimes \mathbb{1}_{n_2}, \mathbb{1}_{m_1} \otimes H_2^T\right), \tag{4.22}$$

where $\mathbb{1}_n$ is the $n \times n$ identity matrix. Note that H_X is an $M_X \times N$ matrix and H_Z is an $M_Z \times N$ matrix where:

$$N := n_1 n_2 + m_1 m_2, \qquad M_X := n_1 m_2, \qquad M_Z := m_1 n_2.$$

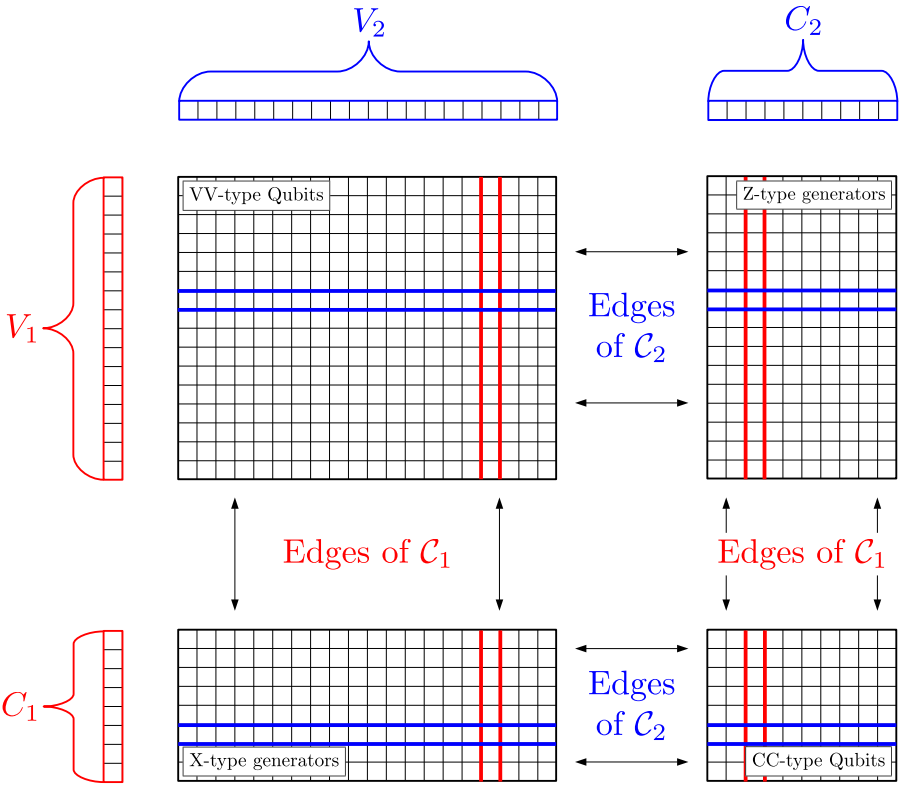

Figure 4.4: Tanner graph of an hypergraph product code.

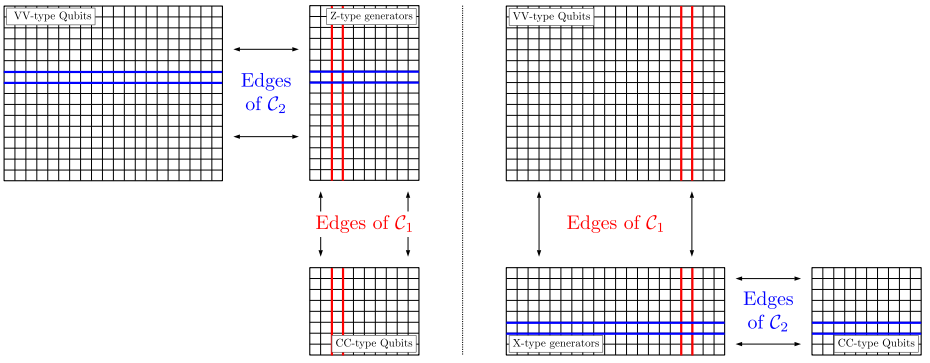

Figure 4.5: Tanner graph of $\mathcal{C}_X$ (on the left) and Tanner graph of $\mathcal{C}_Z$ (on the right).

Furthermore, it is straightforward to check that the orthogonality property $H_X H_Z^T = 0$ holds:

$$\begin{aligned} H_X H_Z^T &= \left(\mathbb{1}_{n_1} \otimes H_2, H_1^T \otimes \mathbb{1}_{m_2}\right) \cdot \begin{pmatrix} H_1^T \otimes \mathbb{1}_{n_2} \\ \mathbb{1}_{m_1} \otimes H_2 \end{pmatrix} \\ &= \left(\mathbb{1}_{n_1} \otimes H_2\right) \cdot \left(H_1^T \otimes \mathbb{1}_{n_2}\right) + \left(H_1^T \otimes \mathbb{1}_{m_2}\right) \cdot \left(\mathbb{1}_{m_1} \otimes H_2\right) \\ &= H_1^T \otimes H_2 + H_1^T \otimes H_2 \\ &= 0. \end{aligned}$$

The Tanner graph $G_{\mathcal{Q}}$ of an hypergraph product code is equal to the *Cartesian product* of the classical Tanner graphs as represented in Figure 4.4. Formally, let V_1 (resp. V_2) be the set of bit-nodes of $\mathcal{C}_1$ (resp. $\mathcal{C}_2$), let C_1 (resp. C_2) be its set of check-nodes and let G_1 (resp. G_2) be its Tanner graph. Then, the graph $G_{\mathcal{Q}}$ is the Cartesian product of G_1 and G_2 defined in the following way:

- The qubits are indexed by the set $V_{\mathcal{Q}} := (V_1 \times V_2) \uplus (C_1 \times C_2)$ ($\uplus$ is the disjoint union).
- The X-type generators are indexed by the set $C_Z := C_1 \times V_2$.
- The Z-type generators are indexed by the set $C_X := V_1 \times C_2$.
- Two nodes $u_1, u_2 \in V_{\mathcal{Q}} \uplus C_X \uplus C_Z$ are connected in $G_{\mathcal{Q}}$ in two cases. Either the first components of u_1 and u_2 are equal and their second components are connected in G_2; or their second components are equal and their first components are connected in G_1.

The qubits belonging to $V_1 \times V_2$ are called the *VV-type qubits* and the qubits belonging to $C_1 \times C_2$ are called the *CC-type qubits*. The incidence relation of $G_{\mathcal{Q}}$ defined above can be split in four cases:

- A VV-type qubit $q \in V_1 \times V_2$ is in the support of an X-type generator $(c_1, v_2) \in C_1 \times V_2$ if and only if $q = (v_1, v_2)$ where v_1 is connected to c_1 in G_1.
- A VV-type qubit $q \in V_1 \times V_2$ is in the support of a Z-type generator $(v_1, c_2) \in V_1 \times C_2$ if and only if $q = (v_1, v_2)$ where v_2 is connected to c_2 in G_2.
- A CC-type qubit $q \in C_1 \times C_2$ is in the support of an X-type generator $(c_1, v_2) \in C_1 \times V_2$ if and only if $q = (c_1, c_2)$ where c_2 is connected to v_2 in G_2.
- A CC-type qubit $q \in C_1 \times C_2$ is in the support of a Z-type generator $(v_1, c_2) \in V_1 \times C_2$ if and only if $q = (c_1, c_2)$ where c_1 is connected to v_1 in G_1.

In Figure 4.4 where $G_{\mathcal{Q}}$ is depicted, each column is a copy of G_1 and each row is a copy of G_2. In particular, there is no edge between a qubit and a generator if they are not in the same row or in the same column. Therefore in $G_{\mathcal{Q}}$, there are $\left|V_2\right| + \left|C_2\right| = n_2 + m_2$ copies of G_1 and $\left|V_1\right| + \left|C_1\right| = n_1 + m_1$ copies of G_2.

The Tanner graph of the classical code $\mathcal{C}_X$ used to correct X-type errors is the sub-graph of $G_{\mathcal{Q}}$ induced by the set of VV-type qubits, the set of CC-type qubits and the set of Z-type generators (see the left part of Figure 4.5). Similarly, the qubits and

the X-type generators induce the Tanner graph of $\mathcal{C}_Z$ used to correct Z-type errors (see the right part of Figure 4.5).

Remark 4.8. If the initial codes $\mathcal{C}_1$ and $\mathcal{C}_2$ have the LDPC property then the resulting hypergraph product code $\mathcal{Q}$ is also LDPC.

Proof. In this proof, a matrix A is said to be (r, c)-LDPC when each row has weight upper bounded by r and each column has weight upper bounded by c. If a matrix A is (r, c)-LDPC then the tensor product $A \otimes \mathbb{1}$ is also (r, c)-LDPC. By eq. (4.22), if $\mathcal{C}_1, \mathcal{C}_2$ are LDPC then $\mathcal{C}_X, \mathcal{C}_Z$ and $\mathcal{Q}$ are also LDPC. □

In Section 4.3, we will assume that the Tanner graph G_1 (resp. G_2) of $\mathcal{C}_1$ (resp. $\mathcal{C}_2$) is regular with left degree d_{V_1} (resp. d_{V_2}) and right degree d_{C_1} (resp. d_{C_2}). In that case, the VV-type qubits have degree $d_{V_1} + d_{V_2}$, the CC-type qubits have degree $d_{C_1} + d_{C_2}$, the X-type generators have degree $d_{C_1} + d_{V_2}$ and the Z-type generators have degree $d_{V_1} + d_{C_2}$.

4.2.2 Parameters of an hypergraph product code

The goal of this section is to prove Theorem 4.9 below where the dimension and the minimal distance of an hypergraph product code are computed.
Let $\mathcal{C}$ be a classical error correcting code with parity check matrix H. Then the *transposed code* $\mathcal{C}^T$ is defined to be the classical code whose parity check matrix is H^T. The code $\mathcal{C}^T$ has the same Tanner graph as $\mathcal{C}$ where the bit-nodes and the check-nodes have been switched. In particular in Theorem 4.9, m_1 is the number of check-nodes of $\mathcal{C}_1$ and m_2 is the number of check-nodes of $\mathcal{C}_2$.
In the degenerate case where a code $\mathcal{C} = \{0^N\}$ is trivial, Theorem 4.9 uses the convention that $\mathcal{C}$ is an $[n, 0, d]$ code with $d = +\infty$.

Theorem 4.9 ([108])**.** *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two classical error correcting codes and let n_1, k_1, d_1, n_2, k_2, d_2, m_1, k_1^T, d_1^T, m_2, k_2^T, $d_2^T \in \mathbb{N}$ be such that:*

$$\begin{array}{ll} \mathcal{C}_1 \text{ is an } [n_1, k_1, d_1] \text{ code,} & \mathcal{C}_2 \text{ is an } [n_2, k_2, d_2] \text{ code,} \\ \mathcal{C}_1^T \text{ is an } [m_1, k_1^T, d_1^T] \text{ code,} & \mathcal{C}_2^T \text{ is an } [m_2, k_2^T, d_2^T] \text{ code.} \end{array}$$

Then $\mathcal{Q}$ the hypergraph product of $\mathcal{C}_1$ and $\mathcal{C}_2$ is an $[[N, K, D]]$ stabilizer code where:

$$\textit{(i)}\ N = n_1 n_2 + m_1 m_2, \quad \textit{(ii)}\ K = k_1 k_2 + k_1^T k_2^T, \quad \textit{(iii)}\ D = \min(D_X, D_Z),$$

$$\textit{(iv)}\ D_X \geq \min(d_1^T, d_2), \qquad \textit{(v)}\ D_Z \geq \min(d_1, d_2^T).$$

The integers D_X and D_Z are the minimal distances associated to X-type and Z-type errors and moreover:

$$\begin{array}{ll} \textit{(vi)} & \textit{If } k_1 \neq 0 \textit{ and } k_2 \neq 0 \textit{ then } D_X \leq d_2 \textit{ and } D_Z \leq d_1. \\ \textit{(vii)} & \textit{If } k_1^T \neq 0 \textit{ and } k_2^T \neq 0 \textit{ then } D_X \leq d_1^T \textit{ and } D_Z \leq d_2^T. \end{array}$$

(viii) Otherwise $K = 0$ and $\mathcal{Q}$ is trivial.

We split the proof of Theorem 4.9 into three parts: the proof of item (ii), the proof of item (iv) and the proof of item (vi). Item (i) is a direct consequence of the hypergraph product definition, item (iii) has already been proven in eq. (4.21), the proof of item (v) is similar to the proof of item (iv), the proof of item (vii) is similar to the proof of item (vi) and item (viii) is a consequence of item (ii).

Oftentimes, the parity check matrices of $\mathcal{C}_1$ and $\mathcal{C}_2$ are full rank with $n_1 < m_1$ and $n_2 < m_2$. In that case, the codes $\mathcal{C}_1^T$ and $\mathcal{C}_2^T$ are reduced to the all-zero codeword and:

$$k_1 = n_1 - m_1, \qquad k_2 = n_2 - m_2, \qquad k_1^T = k_2^T = 0, \qquad d_1^T = d_2^T = +\infty.$$

Moreover in the next sections, we will only consider the hypergraph product of a code $\mathcal{C}$ with itself. If $\mathcal{C}$ is an $[n, k, d]$ code with $m < n$ check-nodes and full rank parity check matrix then Theorem 4.9 becomes:

$$N = n^2 + m^2 = n^2 + (n-k)^2, \qquad K = k^2, \qquad D = D_X = D_Z = d.$$

When $\mathcal{C}$ defines a family of classical codes with rate $r = k/n$ and linear minimal distance $d = \Theta(n)$, the corresponding hypergraph product is a family of CSS codes with rate $r_{\mathcal{Q}}$ where:

$$N = \Theta(n^2), \qquad \frac{1}{r_{\mathcal{Q}}} = \left(\frac{1}{r}\right)^2 + \left(\frac{1}{r} - 1\right)^2, \qquad D = \Theta\Big(\sqrt{N}\Big).$$

An interesting example where the parity check matrices are not full rank is given in Example 4.10 below.

Example 4.10 (Toric code)**.** *The toric code of length $L \in \mathbb{N}^*$ can be defined as the hypergraph product of an L-bit repetition code $\mathcal{C}$ with itself. By definition, the L-bit repetition code $\mathcal{C} = \mathcal{C}^T$ is an $[L, 1, L]$ linear code whose Tanner graph is a cycle of length $2L$ with L bit-nodes and L check-nodes. By Theorem 4.9, the toric code is a $[[2L^2, 2, L]]$ stabilizer code.*

We now prove the statements of Theorem 4.9.

Proof of Theorem 4.9 (ii). First, we show using linear algebra that computing K is equivalent to computing k_X^T and k_Z^T the dimensions of $\mathcal{C}_X^T$ and $\mathcal{C}_Z^T$. Let $\mathcal{C}$ be an $[n, k]$ classical error correcting code (later we will take $\mathcal{C} \in \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_X, \mathcal{C}_Z\}$) with m check-nodes and with parity check matrix H. The rank–nullity theorem asserts that:

$$k = n - \operatorname{rank}(H) \qquad \text{and} \qquad k^T = m - \operatorname{rank}(H^T).$$

Since $\operatorname{rank}(H) = \operatorname{rank}(H^T)$, we have $k = n - m + k^T$ and for $\mathcal{C} \in \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_X, \mathcal{C}_Z\}$ we get:

$$\begin{aligned} k_1 &= n_1 - m_1 + k_1^T, & k_X &= n_1 n_2 + m_1 m_2 - n_1 m_2 + k_X^T, \\ k_2 &= n_2 - m_2 + k_2^T, & k_Z &= n_1 n_2 + m_1 m_2 - m_1 n_2 + k_Z^T. \end{aligned} \tag{4.23}$$

By Proposition 4.6, we have $K = k_X + k_Z - n_1 n_2 - m_1 m_2$ and thus computing k_X^T and k_Z^T is sufficient to compute K.

The integer k_X^T is the dimension of the code $\mathcal{C}_X^T$ whose Tanner graph is given in the left part of Figure 4.5 where the Z-type generators are interpreted as bits and the qubits are interpreted as check-nodes. In fact $\mathcal{C}_X^T = \mathcal{C}_1 \otimes \mathcal{C}_2^T$ is a classical product code as defined in Section 7.2. By Proposition 7.2:

$$k_X^T = k_1 k_2^T, \qquad \text{and similarly} \qquad k_Z^T = k_1^T k_2. \tag{4.24}$$

Finally, using eqs. (4.23) and (4.24) we get:

$$\begin{aligned}
K &= k_X + k_Z - n_1 n_2 - m_1 m_2 \\
&= n_1 n_2 + m_1 m_2 - n_1 m_2 + k_X^T - m_1 n_2 + k_Z^T \\
&= n_1 n_2 + m_1 m_2 - n_1 m_2 + k_1 k_2^T - m_1 n_2 + k_1^T k_2 \\
&= n_1 n_2 + m_1 m_2 - n_1 m_2 + (n_1 - m_1 + k_1^T) k_2^T - m_1 n_2 + k_1^T (n_2 - m_2 + k_2^T) \\
&= (n_1 - m_1 + k_1^T)(n_2 - m_2 + k_2^T) + k_1^T k_2^T \\
&= k_1 k_2 + k_1^T k_2^T .
\end{aligned}$$

□

Surprisingly, the formula for the dimension of an hypergraph product code given in Theorem 4.9 (ii) can be used to determine its minimal distance.

Proof of Theorem 4.9 (iv). As discussed in Section 4.1.6, D_X is the minimal weight of an X-type Pauli error outside the stabilizer group with an empty syndrome. To prove $D_X \geq \min(d_1^T, d_2)$, we show that if an X-type error $E \subseteq V_{\mathcal{Q}}$ satisfies $\left|E\right| < \min(d_1^T, d_2)$ and has an empty syndrome then E is equal to a product of generators. In order to write E as a product of generators, we are going to construct $\mathcal{Q}'$ another hypergraph product code with zero logical qubit and whose Tanner graph is a sub-graph of the Tanner graph of $\mathcal{Q}$. The way $\mathcal{Q}'$ is constructed ensures that the error E has also an empty syndrome with respect to $\mathcal{Q}'$. Finally, since $\mathcal{Q}'$ has zero logical qubit, any error with an empty syndrome is a product of generators.

In this proof, we denote by G_1 and G_2 the Tanner graphs of $\mathcal{C}_1$ and $\mathcal{C}_2$. We will also define a classical code $\mathcal{C}_1'$ (resp. $\mathcal{C}_2'$) by its Tanner graph G_1' (resp. G_2') which is a sub-graph of G_1 (resp. G_2). The hypergraph product of $\mathcal{C}_1$ and $\mathcal{C}_2$ (resp. $\mathcal{C}_1'$ and $\mathcal{C}_2'$) is denoted by $\mathcal{Q}$ (resp. $\mathcal{Q}'$) and its Tanner graph by $G_{\mathcal{Q}}$ (resp. $G_{\mathcal{Q}'}$).

Let $E \subseteq V_{\mathcal{Q}} = V_1 \times V_2 \uplus C_1 \times C_2$ be an X-type error seen as a subset of the qubits with $\sigma(E) = \varnothing$ and $\left|E\right| < \min(d_1^T, d_2)$. As illustrated in Figure 4.6, we define the set $V_2' \subseteq V_2$ to be the projection of $E \cap (V_1 \times V_2)$ onto its second coordinate and $C_1' \subseteq C_1$ to be the projection of $E \cap (C_1 \times C_2)$ onto its first coordinate:

$$\begin{aligned}
V_2' &:= \Big\{ v_2 \in V_2 : \exists v_1 \in V_1, (v_1, v_2) \in E \Big\}, \\
C_1' &:= \Big\{ c_1 \in C_1 : \exists c_2 \in C_2, (c_1, c_2) \in E \Big\}.
\end{aligned}$$

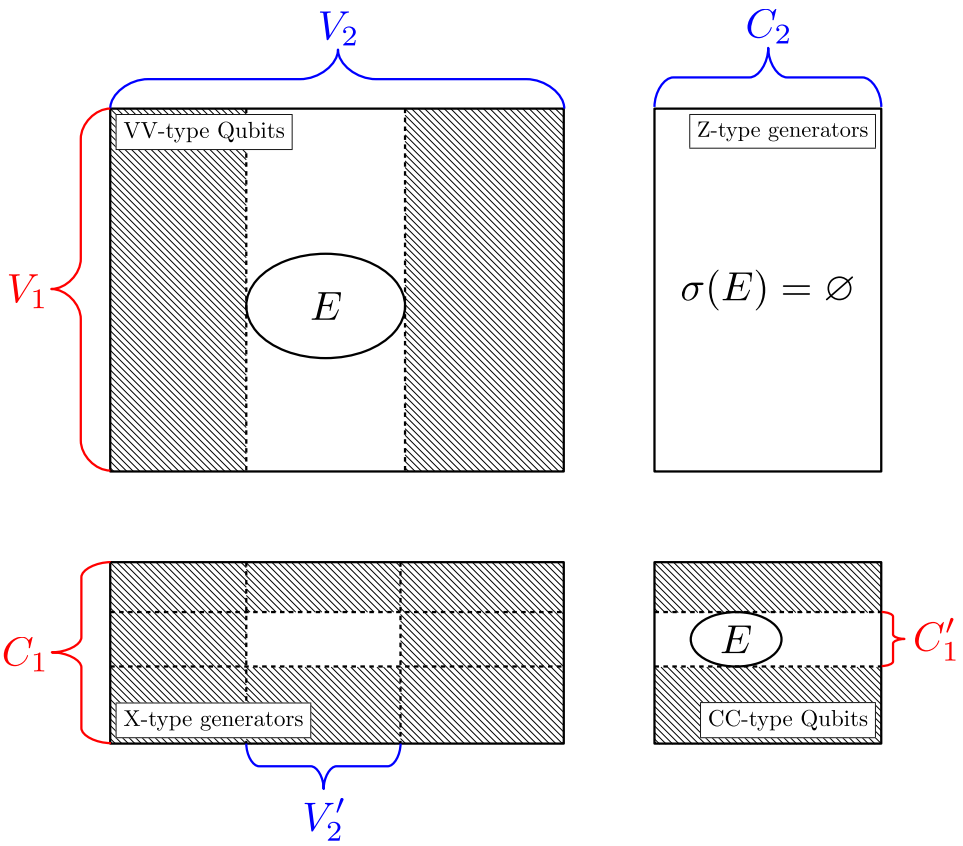

Figure 4.6: Tanner graph of $\mathcal{Q}'$.

Let G_1' be the sub-graph of G_1 induced by $V_1 \uplus C_1'$ and let G_2' be the sub-graph of G_2 induced by $V_2' \uplus C_2$. Then the Tanner graph $G_{\mathcal{Q}'}$ is a sub-graph of $G_{\mathcal{Q}}$.

In a nutshell, the error E is equal to a product of X-type generators because its syndrome is empty for $\mathcal{Q}'$ which has zero logical qubit. We provide more details below: Firstly, some of the Z-type generators have fewer neighbors in $G_{\mathcal{Q}'}$ than in $G_{\mathcal{Q}}$. However, for any X-type error $E' \subseteq V_1 \times V_2' \uplus C_1' \times C_2$ included in the qubits of $\mathcal{Q}'$, the syndrome of E' does not depend on whether we are talking about $\mathcal{Q}$ or $\mathcal{Q}'$. In particular, the error E has an empty syndrome for both codes.
Secondly, any error $E_2 \subseteq V_2'$ included in the bits-nodes of $\mathcal{C}_2'$ has the same syndrome for $\mathcal{C}_2$ and for $\mathcal{C}_2'$. In particular, $\left|E_2\right| \leq \left|V_2'\right| \leq \left|E\right| < d_2$ thus E_2 cannot have an empty syndrome except if $E_2 = \varnothing$. Hence the only codeword of $\mathcal{C}_2'$ is the all zero bit-string and thus the dimension of $\mathcal{C}_2'$ is zero. Similarly, the dimension of $\mathcal{C}_1^T$ is zero and using Theorem 4.9 (ii) we conclude that $\mathcal{Q}'$ has zero logical qubit. Since the error E has an empty syndrome for $\mathcal{Q}'$, it can be written as a product of X-type generators.
Thirdly, an X-type generator appearing in $G_{\mathcal{Q}'}$ has the same neighborhood in $G_{\mathcal{Q}}$ and

in $G_{\mathcal{Q}'}$. In particular, E is a product of X-type generator for $\mathcal{Q}'$ thus it is a product of X-type generator for $\mathcal{Q}$. □

Finally, the proof of item (vi) lies in finding a non trivial X-type error with empty syndrome and weight below d_2 under the assumptions $k_1, k_2 \neq 0$.

Proof of Theorem 4.9 (vi). To prove the inequality $D_X \leq d_2$, we show that if $k_1 \neq 0$ and $k_2 \neq 0$ then there exists an X-type error $E \subseteq V_{\mathcal{Q}}$ of weight d_2 with an empty syndrome and which cannot be written as a product of X-type generators. The inequality $D_Z \leq d_1$ could be proven in the same way.

We emphasize that below, we use $\mathcal{C}_1^{\perp}$ the dual space of $\mathcal{C}_1$ not to be confused with $\mathcal{C}_1^T$. First, the inequality $k_1 \neq 0$ implies $\dim(\mathcal{C}_1^{\perp}) = n - k_1 < n$ and thus there exists at least one binary vector $e_1 \notin \mathcal{C}_1^{\perp}$ with Hamming weight equal to 1. Second, the inequality $k_2 \neq 0$ implies that there exists a codeword $e_2 \in \mathcal{C}_2$ whose support $E_2 \subseteq V_2$ satisfies $E_2 \neq \varnothing$.

Let $v_1 \in V_1$ be the unique element in the support of e_1, let $v_2 \in E_2$ and let $E := \{v_1\} \times E_2$. To conclude the proof, we show that E is not a product of X-type generators and has an empty syndrome.

If the error E was equal to a product of X-type generators then the error $\{v_1\} \times \{v_2\}$ would be equal to a product of X-type generators belonging to $C_1 \times \{v_2\}$. The X-type generators in $C_1 \times \{v_2\}$ belong to the red column of Figure 4.7 which is a copy of the Tanner graph of $\mathcal{C}_1$. Hence $e_1 \notin \mathcal{C}_1^{\perp}$ implies that E cannot be a product of X-type generators.

The syndrome of E is included in the Z-type generators of $\{v_1\} \times C_2$. These generators are represented by the blue row in Figure 4.7 which is a copy of the Tanner graph of $\mathcal{C}_2$ and thus the statement $e_2 \in \mathcal{C}_2$ implies that E has an empty syndrome for the hypergraph product code.

□

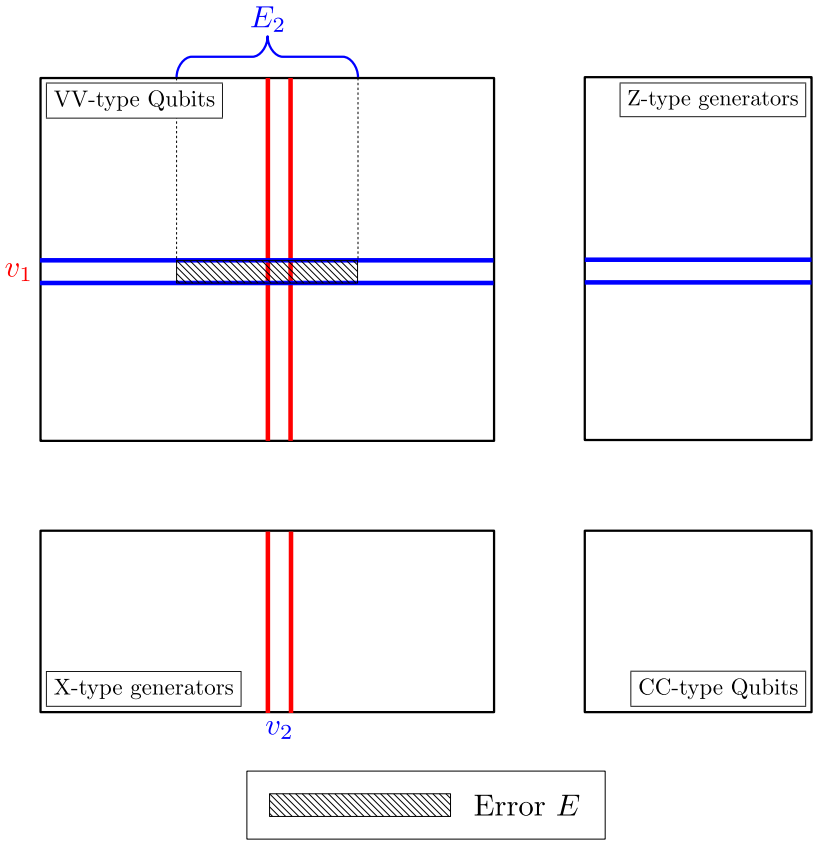

Figure 4.7: Example of a non trivial error E with empty syndrome.

4.3 Quantum expander codes

Given $\mathcal{C}_1, \mathcal{C}_2$, two classical expander codes as defined in Section 3.2, the hypergraph product of $\mathcal{C}_1$ and $\mathcal{C}_2$ is called a *quantum expander code* [67]. For simplicity in this manuscript, we focus on the particular case $\mathcal{C} = \mathcal{C}_1 = \mathcal{C}_2$ where $\mathcal{C}$ is an $[n, \Theta(n), \Theta(n)]$ expander code with a regular Tanner graph (see Section 3.2.4 for the existence of such codes). By Remark 4.8 and Theorem 4.9, the resulting quantum code is LDPC (but not regular) with parameters $[[N, \Theta(N), \Theta(\sqrt{N})]]$ where $N = \Theta(n^2)$.

The main advantage of a quantum expander code compared to other families of hypergraph product codes is the possibility to analyze the performance of the *small-set-flip decoder* [67]. The small-set-flip decoder is similar to the bit-flip algorithm discussed in Section 3.2.3: the execution is divided into several rounds where the algorithm tries to reduce the syndrome weight. In the classical setting, a single bit is flipped at each round, but in the quantum case, a set of several qubits F has to be flipped if we wish to ensure the syndrome weight to decrease. When the small-set-flip decoder is used to

correct X-type errors, the check-nodes are represented by the Z-type generators and the set F is selected among all possible subsets of an X-type generator support. A subset of an X-type generator support is called a *small-set* and the set of small-sets is denoted by $\mathcal{F}$. As a summary, at each round, the small-set-flip algorithm decreases the syndrome weight by flipping a set of qubits $F \in \mathcal{F}$.

This section is a review on the results of ref. [67], in particular the small-set-flip algorithm is shown to correct any adversarial error with weight up to a fraction of the minimal distance. This section is organized as follows: in Section 4.3.1 we define the quantum expander codes and the small-set-flip decoder, and in Section 4.3.3 we analyze their behavior against adversarial errors. In Chapter 5, we will rely on the results presented in this section to show that quantum expander codes have a threshold for any local stochastic error model as defined in Section 2.3.

4.3.1 Definition

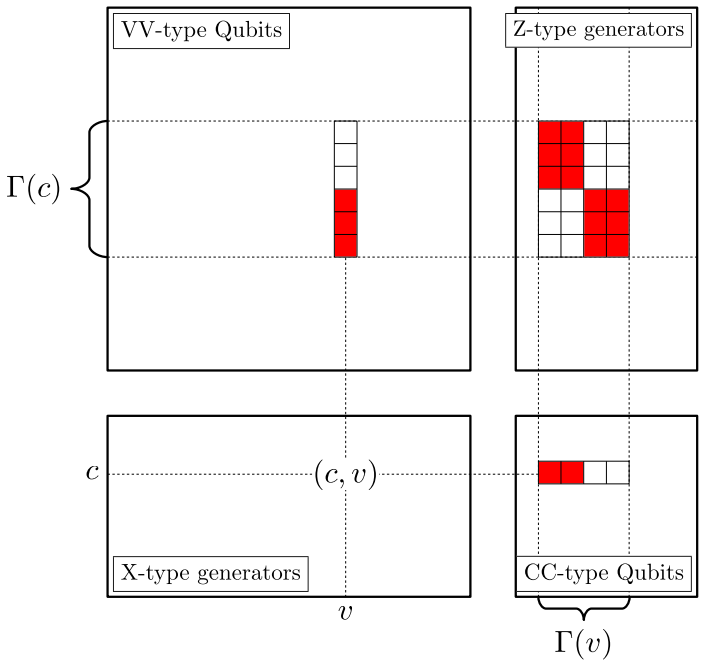

Figure 4.8: constant weight error on which the bit-flip decoder is blocked. The qubits in red represent the error and the Z-type generators in red represent the unsatisfied check-nodes.

Definition 4.11 (Quantum expander code [67])**.** A quantum expander code is the hypergraph product of a classical expander code with itself.

In what follows, we describe the error correction procedure for X-type errors. In fact, for the hypergraph product of a code with itself, the Tanner graph of $\mathcal{C}_X$ and $\mathcal{C}_Z$ presented in Figure 4.5 are equal up to a permutation on the qubits. Hence, X-type errors and Z-type errors are corrected in the same way and without loss of generality, we deal with X-type errors only.
All along this section we use the notations defined below.

Notation 4.12. *Let G be the Tanner graph of a classical expander code, let V be the set of bit-nodes, let C be the set of check-nodes and let Γ be the neighborhood in G. We assume that G has left degree d_V, right degree d_C and is a (γ, δ)-expander graph with $\gamma > 0$ and $\delta < 1/8$. We denote by $\mathcal{Q}$ the associated quantum expander code, by $V_\mathcal{Q}$ the set of qubits, by C_X the set of Z-type stabilizer generators, by C_Z the set of X-type stabilizer generators and by N the number of physical qubits:*

$$V_\mathcal{Q} := V^2 \uplus C^2, \qquad C_X := V \times C, \qquad C_Z := C \times V, \qquad N := \big|V_\mathcal{Q}\big|.$$

For an X-type error $E \subseteq V_\mathcal{Q}$, the syndrome of E is denoted $\sigma_X(E) \subseteq C_X$.
In addition we define the following positive constants:

$$r := \frac{d_V}{d_C} = \frac{\big|C\big|}{\big|V\big|}, \qquad \gamma_0 = \frac{r^2}{\sqrt{1+r^2}}\gamma, \qquad \beta_0 := 1 - 8\delta, \qquad \alpha_0 := \frac{r\beta_0}{4 + 2r\beta_0}.$$

When a quantum expander code is decoded with the bit-flip algorithm used in the classical case, there exist constant weight errors which are not corrected. Figure 4.8 provides an example where the error is made of three VV-type qubits and two CC-type qubits in the support of an X-type generator $g = (c, v) \in C_X$. In this example, the error weight is 5, the syndrome weight is 12, the Tanner graph of the initial classical code has left degree $d_V = 4$ and right degree $d_C = 6$, and the X-type generators have degree 10. The bit-flip decoder fails in correcting this error since no bit-flip decreases the syndrome weight. In particular, we can check that the syndrome weight is not modified when a single qubit belonging to the support of g is flipped.

For the example shown in Figure 4.8, the *small-set-flip decoder* proposed by ref. [67] flips the five red qubits in one round. More generally, the decoding strategy is to decrease the syndrome weight by flipping a set of qubits called a *small-set*. By definition, a set $F \subseteq V_\mathcal{Q}$ is a small-set if and only if it is included in the support of an X-type generator, *i.e.* if and only if $F \in \mathcal{F}_0$ where:

$$\mathcal{F}_0 := \Big\{ F \subseteq \Gamma_Z(g) : g \in C_Z \Big\}. \tag{4.25}$$

Given a syndrome $\sigma \subseteq C_X$ and a small-set $F \in \mathcal{F}_0$, we denote by $\Delta(\sigma, F)$ the diminution of the syndrome weight when F is flipped:

$$\Delta(\sigma, F) := \big|\sigma\big| - \big|\sigma \oplus \sigma_X(F)\big|. \tag{4.26}$$

In the original paper [67], at each round, the small-set-flip algorithm flips a set of qubits F chosen to be the small-set $F \in \mathcal{F}_0$ maximizing $\Delta(\sigma, F)/\big|F\big|$. In order to simplify the proofs in this manuscript, the selected set F is an arbitrary $F \in \mathcal{F}_0$ satisfying

$\left|\sigma_X(F)\right| \geq \frac{d_V}{2}\left|F\right|$ and $\Delta(\sigma, F) \geq \beta_0\left|\sigma_X(F)\right|$ (β_0 is defined in Notation 4.12). Our condition is more convenient for the case where the syndrome is noisy but the algorithm of [67] and many other variants can be analyzed in the same way. Let:

$$\mathcal{F} := \left\{ F \in \mathcal{F}_0 : \left|\sigma_X(F)\right| \geq \frac{d_V}{2}\left|F\right| \right\}. \tag{4.27}$$

We are now ready to define the small-set-flip decoder (Algorithm 2).

Algorithm 2 : the small-set-flip decoder.

Input: the syndrome $\sigma \subseteq C_X$. // $\sigma = \sigma(E)$ for some $E \subseteq V_{\mathcal{Q}}$
Output: a guess for the error $\hat{E} \subseteq V_{\mathcal{Q}}$.

$\hat{E}_0 = \varnothing$; $\sigma_0 = \sigma$; $i = 0$ // $E_0 = E, \sigma_0 = \sigma(E_0)$
while $\exists F_i \in \mathcal{F} : \Delta(\sigma_i, F_i) \geq \beta_0\left|\sigma_X(F_i)\right|$ **do**
  Pick such an F_i arbitrarily. // $\left|\sigma_X(E_i)\right| - \left|\sigma_X(E_i \oplus F_i)\right| \geq \beta_0\left|\sigma_X(F_i)\right|$
  $\hat{E}_{i+1} = \hat{E}_i \oplus F_i$ // $E_{i+1} = E_i \oplus F_i$
  $\sigma_{i+1} = \sigma_i \oplus \sigma_X(F_i)$ // $\sigma_{i+1} = \sigma(E_{i+1})$
  $i = i + 1$
end while
return $\hat{E}_i$

Similarly to the bit-flip algorithm (Algorithm 1) it is insightful to think about the set $E_i := E \oplus \hat{E}_i$ which represents the physical errors at each round of the algorithm (see Figure 3.3 for a graphical representation of E, E_i and $\hat{E}_i$). The comments of Algorithm 2 mention the set E_i and its relationship with the variables of the algorithm.

4.3.2 Weighted cardinality, reduced weight and reduced set

Before analyzing Algorithm 2, we introduce in Definition 4.13 below three convenient tools: the *weighted cardinality*, the *reduced weight* and the concept of *reduced set*.

Definition 4.13. We use Notation 4.12. Let $E \subseteq V_{\mathcal{Q}} = V^2 \uplus C^2$ be an error set. The weighted cardinality $\left\|E\right\|$ is defined by:

$$\left\|E\right\| := \frac{\left|E \cap V^2\right|}{d_C} + \frac{\left|E \cap C^2\right|}{d_V}.$$

The reduced weight of E denoted by $\left|E\right|_{\mathrm{R}}$ is defined to be the minimum weight of an error equivalent to E:

$$\left|E\right|_{\mathrm{R}} := \min_{E' \in \mathcal{C}_Z^{\perp}} \left|E + E'\right|.$$

A set $E \subseteq V_{\mathcal{Q}}$ is said to be reduced when $|E|$ is minimal over the errors equivalent to E:

$$E \text{ is reduced } \Leftrightarrow \left|E\right| = \left|E\right|_{\mathrm{R}}.$$

Similarly, a set $E \subseteq V_{\mathcal{Q}}$ is said to be $\|\ \|$-reduced when $\|E\|$ is minimal over the errors equivalent to E:

$$E \text{ is } \|\ \|\text{-reduced} \Leftrightarrow \|E\| = \min_{E' \in \mathcal{C}_Z^\perp} \|E + E'\|.$$

As shown in Lemma 4.14 below, the weighted cardinality $\|\cdot\|$ shares a couple of properties with the usual the cardinality $|\cdot|$.

Lemma 4.14. *We use Notation 4.12. Let $E, E_1, E_2 \subseteq V_{\mathcal{Q}}$ then:*

(i) $\|E\| \geq 0$, *(ii)* $\|E\| = 0 \Leftrightarrow E = \varnothing$,
(iii) $E_1 \subseteq E_2 \Rightarrow \|E_1\| \leq \|E_2\|$, *(iv)* $d_V\|E\| \leq |E| \leq d_C\|E\|$,
(v) $|\sigma_X(E)| \leq d_V d_C \|E\|$, *(vi)* $\|E_1 \cup E_2\| \leq \|E_1\| + \|E_2\|$,
(vii) $\|E_1 \uplus E_2\| = \|E_1\| + \|E_2\|$,

(viii) $$\|E_1 \oplus E_2\| = \|E_1\| + \|E_2\| - 2\|E_1 \cap E_2\|.$$

Proof. For the usual cardinality, we already have:

$$|E| \geq 0, \qquad |E| = 0 \Leftrightarrow E = \varnothing,$$
$$E_1 \subseteq E_2 \Rightarrow |E_1| \leq |E_2|, \qquad |E_1 \cup E_2| \leq |E_1| + |E_2|,$$
$$|E_1 \uplus E_2| = |E_1| + |E_2|, \qquad |E_1 \oplus E_2| = |E_1| + |E_2| - 2|E_1 \cap E_2|.$$

It is a direct consequence to show items (i), (ii), (iii), (vi), (vii), (viii) where $|\cdot|$ is replaced by $\|\cdot\|$.
The inequality $d_V \leq d_C$ implies item (iv).
Finally, by linearity, it is sufficient to show item (v) in the two particular cases $E \subseteq V^2$ and $E \subseteq C^2$. For example when $E \subseteq V^2$:

$$|\sigma_X(E)| = \Big|\bigoplus_{e \in E} \sigma_X(\{e\})\Big| \leq \sum_{e \in E} \big|\sigma_X(\{e\})\big| = d_V|E| = d_V d_C \|E\|.$$

□

In addition, all along this manuscript we will use the handy property of Lemma 4.15 below.

Lemma 4.15. *We use Notation 4.12.*
Let $E_1 \subseteq E_2 \subseteq V_{\mathcal{Q}}$ be two errors. If E_2 is $\|\ \|$-reduced (resp. reduced) then E_1 is $\|\ \|$-reduced (resp. reduced).

Proof. Let's prove that $\|E_1\| \leq \|E_1 \oplus E\|$ holds for any $E \in \mathcal{C}_Z^\perp$. By Lemma 4.14 (viii):

$$\|E_1 \oplus E\| - \|E_1\| = \|E_1\| + \|E\| - 2\|E_1 \cap E\| - \|E_1\| = \|E\| - 2\|E_1 \cap E\|.$$

Similarly:

$$\big\|E_2 \oplus E\big\| - \big\|E_2\big\| = \big\|E\big\| - 2\big\|E_2 \cap E\big\|,$$

and by Lemma 4.14 (iii):

$$\big\|E_1 \cap E\big\| \leq \big\|E_2 \cap E\big\|.$$

Since E_2 is $\|\ \|$-reduced, for all $E \in \mathcal{C}_Z^{\perp}$:

$$\begin{aligned} 0 \leq \big\|E_2 \oplus E\big\| - \big\|E_2\big\| &= \big\|E\big\| - 2\big\|E_2 \cap E\big\| \\ &\leq \big\|E\big\| - 2\big\|E_1 \cap E\big\| = \big\|E_1 \oplus E\big\| - \big\|E_1\big\|. \end{aligned}$$

Hence E_1 is $\|\ \|$-reduced.
The same proof also works when replacing $\|\cdot\|$ with $|\cdot|$. □

4.3.3 Errors in the adversarial setting [67]

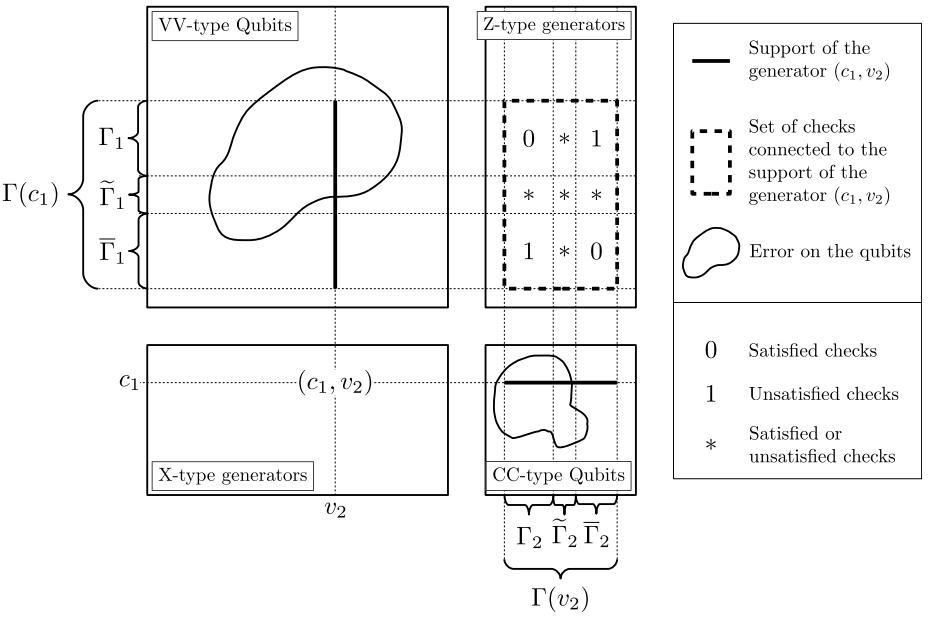

Figure 4.9: schematic representation of a critical generator (c_1, v_2).

In this section, we show in Proposition 4.16 that any error with weight up to $\Theta(\sqrt{N})$ is corrected by Algorithm 2. For simplicity, we assume that the expansion parameter δ of Notation 4.12 satisfies $\delta < 1/8$, but as stated in Theorem 2 of ref. [67], the result still hold under the weaker hypothesis $\delta < 1/6$.

Proposition 4.16 (Lemma 10 of [67]). *We use Notation 4.12.*
Let $E \subseteq V_{\mathcal{Q}}$ be an error satisfying $\left|E\right| \leq 2\alpha_0\gamma_0\sqrt{N}$. Then E is corrected by the small-set-flip algorithm. In other words, if we run Algorithm 2 on input $\sigma_X(E)$ then the output $\hat{E}$ is equivalent to E.

The proof of Proposition 4.16 is given at the end of this section which is organized in such a manner that we will be able to reuse the intermediate results in Chapter 5.

We start with a quick proof sketch for Proposition 4.16. Assume Algorithm 2 runs on input $\sigma_X(E)$ where $E \subseteq V_{\mathcal{Q}}$ is an error and $\left|E\right| = \mathcal{O}(\sqrt{N})$. For each round i, let $E_i := E \oplus \hat{E}_i$ be the physical error on the qubits as defined in the comments of Algorithm 2.
First, we show that $\left|E_i\right| = \mathcal{O}(\sqrt{N})$ holds for all i. We already know that $\left|E_0\right| = \left|E\right| = \mathcal{O}(\sqrt{N})$ and by the LDPC property, the weight of the initial syndrome $\sigma_0 = \sigma_X(E_0)$ satisfies $|\sigma_0| = \mathcal{O}(\left|E_0\right|) = \mathcal{O}(\sqrt{N})$. At each round of the algorithm, a small-set F_i is flipped and the syndrome weight decreases. In other words, the weight of the physical error changes by the quantity $|E_{i+1}| - |E_i| \leq |F_i| = \mathcal{O}(1)$ and the syndrome weight decrease at least by one. Finally:

$$|E_i| = |E_0| + \mathcal{O}(|\sigma_0|) = \mathcal{O}(\sqrt{N}).$$

By definition, the error is corrected if the algorithm stops when the physical error E_i is trivial (equivalent to the empty error). By contraposition, to prove Proposition 4.16, it is sufficient to show the following assertion: if the physical error E_i is not trivial and satisfies $|E_i| = \mathcal{O}(\sqrt{N})$ then the algorithm does not stop at round i.

In fact, showing the last statement is the main difficulty in the proof of Proposition 4.16. The solution proposed by the authors of ref. [67] is to use the notion of *critical generator* defined in Definition 4.17 below. In a nutshell, if an X-type generator is critical then flipping the appropriate small-set in its support will decrease the syndrome weight.

The definition of critical generator is a bit technical thus we provide a schematic representation in Figure 4.9. The idea is to rely on the observation that, for an hypergraph product code, the support of any X-type generator $(c_1, v_2) \in C \times V$ has the following shape:

- The VV-type qubits in the support of (c_1, v_2) are the elements of $\Gamma(c_1) \times \{v_2\}$.
- The CC-type qubits in the support of (c_1, v_2) are the elements of $\{c_1\} \times \Gamma(v_2)$.
- The Z-type generators connected to the support of (c_1, v_2) are the elements of $\Gamma(c_1) \times \Gamma(v_2)$.

The latter point means that the rectangle $\Gamma(c_1) \times \Gamma(v_1)$ in the Tanner graph of Figure 4.9 represents the set of check-nodes affected by at least one qubit of the support of (c_1, v_2). Informally, (c_1, v_2) is a critical generator when:

- The VV-type qubits in $\Gamma(c_1) \times \{v_1\}$ can be partitioned using three sets $\Gamma_1, \overline{\Gamma}_1, \widetilde{\Gamma}_1 \subseteq V$ (see item (1) in Definition 4.17).
  In what follows, we say "a qubit of Γ_1" (resp. $\overline{\Gamma}_1$, resp. $\widetilde{\Gamma}_1$) to talk about the qubits of $\Gamma_1 \times \{v_1\}$ (resp. $\overline{\Gamma}_1 \times \{v_1\}$, resp. $\widetilde{\Gamma}_1 \times \{v_1\}$).

- The CC-type qubits in $\{c_1\}\times\Gamma(v_1)$ can be partitioned using three sets $\Gamma_2, \overline{\Gamma}_2, \widetilde{\Gamma}_2 \subseteq C$ (see item (2) in Definition 4.17).
  In what follows, we say "a qubit of Γ_2" (resp. $\overline{\Gamma}_2$, resp. $\widetilde{\Gamma}_2$) to talk about the qubits of $\{c_1\} \times \Gamma_2$ (resp. $\{c_1\} \times \overline{\Gamma}_2$, resp. $\{c_1\} \times \widetilde{\Gamma}_2$).
- The qubits of Γ_1 and Γ_2 are in the error (see items (3) and (4) in Definition 4.17)
- The qubits of $\overline{\Gamma}_1$ and $\overline{\Gamma}_2$ are not in the error (see items (5) and (6) in Definition 4.17).
- The qubits of $\widetilde{\Gamma}_1$ and $\widetilde{\Gamma}_2$ can be either in the error or outside the error but there are few such qubits (see items (8) and (9) in Definition 4.17).
- For any qubit q of Γ_1, $\overline{\Gamma}_1$, Γ_2 or $\overline{\Gamma}_2$ and for any check-node c connected to q, if a qubit $q' \neq q$ is in the support of c then q' is not in the error (see items (10) to (17) in Definition 4.17).

As shown in the proof of Lemma 4.19 below, when a critical generator exists, the while loop condition of Algorithm 2 is satisfied by the small-set whose elements are the qubits of Γ_1 and Γ_2. As a consequence, showing the existence of a critical generator is sufficient to prove that the algorithm does not stop.

Definition 4.17 (Critical generator, Definition 6 of [67])**.** We use Notation 4.12.
A critical generator for an error $E \subseteq V_{\mathcal{Q}}$ is an X-type generator $(c_1, v_2) \in C_Z$ such that there exist $\Gamma_1, \overline{\Gamma}_1, \widetilde{\Gamma}_1 \subseteq V$ and $\Gamma_2, \overline{\Gamma}_2, \widetilde{\Gamma}_2 \subseteq C$ with:

$$(1)\ \Gamma(c_1) = \Gamma_1 \uplus \overline{\Gamma}_1 \uplus \widetilde{\Gamma}_1, \qquad (2)\ \Gamma(v_2) = \Gamma_2 \uplus \overline{\Gamma}_2 \uplus \widetilde{\Gamma}_2,$$

$$(3)\ \Gamma_1 \times \{v_2\} \subseteq E, \qquad (4)\ \{c_1\} \times \Gamma_2 \subseteq E,$$
$$(5)\ \overline{\Gamma}_1 \times \{v_2\} \subseteq V^2 \setminus E, \qquad (6)\ \{c_1\} \times \overline{\Gamma}_2 \subseteq C^2 \setminus E,$$
$$(7)\ \left|\Gamma_1\right| + \left|\Gamma_2\right| \neq 0, \qquad (8)\ \left|\widetilde{\Gamma}_1\right| \leq 2\delta d_C, \qquad (9)\ \left|\widetilde{\Gamma}_2\right| \leq 2\delta d_V,$$

and for all $v_1 \in \Gamma_1$, $\overline{v}_1 \in \overline{\Gamma}_1$, $c_2 \in \Gamma_2$ and $\overline{c}_2 \in \overline{\Gamma}_2$:

$$(10)\ E \cap \left[\{v_1\} \times \Gamma(c_2)\right] = \left\{(v_1, v_2)\right\} \qquad (11)\ E \cap \left[\Gamma(v_1) \times \{c_2\}\right] = \left\{(c_1, c_2)\right\}$$
$$(12)\ E \cap \left[\{\overline{v}_1\} \times \Gamma(c_2)\right] = \varnothing \qquad (13)\ E \cap \left[\Gamma(\overline{v}_1) \times \{c_2\}\right] = \left\{(c_1, c_2)\right\}$$
$$(14)\ E \cap \left[\{v_1\} \times \Gamma(\overline{c}_2)\right] = \left\{(v_1, v_2)\right\} \qquad (15)\ E \cap \left[\Gamma(v_1) \times \{\overline{c}_2\}\right] = \varnothing$$
$$(16)\ E \cap \left[\{\overline{v}_1\} \times \Gamma(\overline{c}_2)\right] = \varnothing \qquad (17)\ E \cap \left[\Gamma(\overline{v}_1) \times \{\overline{c}_2\}\right] = \varnothing$$

Using expansion based arguments, we can show the existence of critical generators for errors of weight $\mathcal{O}(\sqrt{N})$.

Lemma 4.18 (Lemma 7 of [67])**.** *We use Notation 4.12.*
Let $E \subseteq V_{\mathcal{Q}}$ be an error with $0 < \left|E\right| \leq \gamma\left|C\right|$ then there exists a critical generator for E.

Proof. We denote by $E_V := E \cap V^2$ the error on the VV-type qubits and by $E_C := E \cap C^2$ the error on the CC-type qubits. In this proof, we find a critical generator (c_1, v_2) in the case where $E_V \neq \varnothing$. If $E_V = \varnothing$ then $E_C \neq \varnothing$ and the proof works in the same way exchanging the roles of V and C. Hence we assume $E_V \neq \varnothing$ and we denote by $E_V^1, E_V^2 \subseteq V$ and $E_C^1, E_C^2 \subseteq C$ the projections of E_V and E_C on their first and second coordinates:

$$\begin{aligned}
E_V^1 &:= \Big\{ v_1 \in V : \exists v_2 \in V, (v_1, v_2) \in E_V \Big\}, \\
E_V^2 &:= \Big\{ v_2 \in V : \exists v_1 \in V, (v_1, v_2) \in E_V \Big\}, \\
E_C^1 &:= \Big\{ c_1 \in C : \exists c_2 \in C, (c_1, c_2) \in E_C \Big\}, \\
E_C^2 &:= \Big\{ c_2 \in C : \exists c_1 \in C, (c_1, c_2) \in E_C \Big\}.
\end{aligned}$$

Since $0 < \big|E_V\big| \leq \big|E\big| \leq \gamma\big|C\big| \leq \gamma\big|V\big|$, we also have $0 < \big|E_V^2\big| \leq \gamma\big|V\big|$. By Lemma 3.5:

$$\big|\Gamma_{\mathrm{u}}(E_V^2)\big| \geq d_V \big|E_V^2\big|(1 - 2\delta), \tag{4.28}$$

where the notation Γ_{u} is defined in Notation 3.4 and refers to Γ the neighborhood in the initial expander graph. Hence the mean value of $\big|\Gamma_{\mathrm{u}}(E_V^2) \cap \Gamma(v_2)\big|$ over $v_2 \in E_V^2$ satisfies:

$$\begin{aligned}
\frac{1}{\big|E_V^2\big|} \sum_{v_2 \in E_V^2} \Big|\Gamma_{\mathrm{u}}(E_V^2) \cap \Gamma(v_2)\Big| &\geq \frac{1}{\big|E_V^2\big|} \left|\Gamma_{\mathrm{u}}(E_V^2) \cap \bigcup_{v_2 \in E_V^2} \Gamma(v_2)\right| \\
&= \frac{\big|\Gamma_{\mathrm{u}}(E_V^2)\big|}{\big|E_V^2\big|} \\
&\geq d_V(1 - 2\delta).
\end{aligned}$$

Thus there exists at least one vertex $v_2 \in E_V^2$ with $\big|\Gamma_{\mathrm{u}}(E_V^2) \cap \Gamma(v_2)\big| \geq d_V(1 - 2\delta)$. We define the set $A_1 \subseteq C$ by:

$$A_1 := \Gamma_{\mathrm{u}}(E_V^2) \cap \Gamma(v_2).$$

- In the case where $A_1 \cap E_C^2 = \varnothing$, we take an arbitrary $v_0 \in V$ such that $(v_0, v_2) \in E_V$ (v_0 exists because $v_2 \in E_V^2$) and we pick an arbitrary $c_1 \in \Gamma(v_0)$. The two elements c_1 and v_2 define the critical generator (c_1, v_2) and we set:

$$\begin{aligned}
\Gamma_1 &:= \Big\{ v_1 \in \Gamma(c_1) : (v_1, v_2) \in E_V \Big\} & \overline{\Gamma}_1 &:= \Gamma(c_1) \setminus \Gamma_1 \\
\widetilde{\Gamma}_1 &= \varnothing & \Gamma_2 &= \varnothing \\
\overline{\Gamma}_2 &:= A_1 & \widetilde{\Gamma}_2 &:= \Gamma(v_2) \setminus \overline{\Gamma}_2
\end{aligned}$$

  It remains to show that these sets satisfy the properties of Definition 4.17.

- In the second case where $A_1 \cap E_C^2 \neq \varnothing$, we define $A_2 \subseteq C$ by:

$$A_2 := \Big\{ c_2 \in C : \exists c_1 \in A_1, (c_1, c_2) \in E_C \Big\}.$$

We have $0 < \big|A_2\big| \leq \big|E_C^2\big| \leq \gamma |C|$ and the same argument as the one used below eq. (4.28) ensures that there exists $c_1 \in A_2$ such that $\big|\Gamma_{\rm u}(A_2) \cap \Gamma(c_1)\big| \geq d_C(1-2\delta)$. We define $A_3 := \Gamma_{\rm u}(A_2) \cap \Gamma(c_1)$ and:

$$\begin{aligned} &\Gamma_1 := E_V^1 \cap A_3 \qquad &&\overline{\Gamma}_1 := (V \setminus E_V^1) \cap A_3 \qquad &&\widetilde{\Gamma}_1 := \Gamma(c_1) \setminus A_3 \\ &\Gamma_2 := E_C^2 \cap A_1 \qquad &&\overline{\Gamma}_2 := (C \setminus E_C^2) \cap A_1 \qquad &&\widetilde{\Gamma}_2 := \Gamma(v_2) \setminus A_1 \end{aligned}$$

□

Lemma 4.18 establishes the existence of a critical generator when the error is small enough. The next step provided by Lemma 4.19 is to use this critical generator to construct a small-set F such that flipping F decreases the syndrome weight.

Lemma 4.19 (Lemma 8 of [67])**.** *We use Notation 4.12.*
Let $E_{\rm R} \subseteq V_{\mathcal{Q}}$ be a $\|\ \|$-reduced error such that $0 < \big\|E_{\rm R}\big\| \leq \gamma_0 \sqrt{N}/d_V$, then there exists a small-set $F \in \mathcal{F}$ with $F \subseteq E_{\rm R}$ and:

$$\textit{(i)}\ \ \big|\sigma_X(F)\big| \geq \frac{1}{2} d_V d_C \big\|F\big\|, \quad \textit{(ii)}\ \ \Delta(\sigma_X(E_{\rm R}), F) \geq \big|\sigma_X(F)\big| - 4\delta d_V d_C \big\|F\big\|.$$

Proof. We have:

$$\big|E_{\rm R}\big| \leq d_C \big\|E_{\rm R}\big\| \leq \gamma_0 \sqrt{N}/r = \gamma \frac{r}{\sqrt{1+r^2}} \sqrt{N} = \gamma \frac{r}{\sqrt{1+r^2}} \sqrt{\big|V\big|^2 + \big|C\big|^2} = \gamma |C|.$$

By Lemma 4.18 there exists a critical generator $(c_1, v_2) \in C_Z$ for $E_{\rm R}$ and the associated sets $\Gamma_1, \overline{\Gamma}_1, \widetilde{\Gamma}_1, \Gamma_2, \overline{\Gamma}_2, \widetilde{\Gamma}_2$ defined in Definition 4.17. Let $F := \big(\Gamma_1 \times \{v_2\}\big) \uplus \big(\{c_1\} \times \Gamma_2\big)$. Then $F \subseteq E_{\rm R}$.
For this proof, we extend the notation $\|\cdot\|$ of Definition 4.13 to any set $A_1 \subseteq V$ and to any set $A_2 \subseteq C$ by the formulas:

$$\big\|A_1\big\| := \frac{\big|A_1\big|}{d_C}, \qquad\qquad \big\|A_2\big\| := \frac{\big|A_2\big|}{d_V}.$$

We have:

$$\begin{aligned} \sigma_X(F) &= \Big[\Gamma_1 \times \Big(\Gamma(v_2) \setminus \Gamma_2\Big)\Big] \uplus \Big[\Big(\Gamma(c_1) \setminus \Gamma_1\Big) \times \Gamma(c_1)\Big] \\ &= \Big[\Gamma_1 \times \Big(\overline{\Gamma}_2 \uplus \widetilde{\Gamma}_2\Big)\Big] \uplus \Big[\Big(\overline{\Gamma}_1 \uplus \widetilde{\Gamma}_1\Big) \times \Gamma(c_1)\Big]. \end{aligned}$$

Thus:

$$\begin{aligned} \big|\sigma_X(F)\big| &= \big|\Gamma_1\big|\Big(\big|\overline{\Gamma}_2\big| + \big|\widetilde{\Gamma}_2\big|\Big) + \big|\Gamma_2\big|\Big(\big|\overline{\Gamma}_1\big| + \big|\widetilde{\Gamma}_1\big|\Big) \\ &= d_V d_C \Big[\big\|\Gamma_1\big\|\Big(\big\|\overline{\Gamma}_2\big\| + \big\|\widetilde{\Gamma}_2\big\|\Big) + \big\|\Gamma_2\big\|\Big(\big\|\overline{\Gamma}_1\big\| + \big\|\widetilde{\Gamma}_1\big\|\Big)\Big] \\ &= d_V d_C \Big[\big\|\Gamma_1\big\|\Big(1 - \big\|\Gamma_2\big\|\Big) + \big\|\Gamma_2\big\|\Big(1 - \big\|\Gamma_1\big\|\Big)\Big]. \end{aligned}$$

But $\|F\| = \|\Gamma_1\| + \|\Gamma_2\|$, thus:

$$\frac{|\sigma_X(F)|}{d_V d_C \|F\|} = 1 - \frac{2\|\Gamma_1\|\|\Gamma_2\|}{\|\Gamma_1\| + \|\Gamma_2\|}. \tag{4.29}$$

Note that $0 < \|\Gamma_1\| + \|\Gamma_2\|$ because $0 < |\Gamma_1| + |\Gamma_2|$.
Let $S := \big(\Gamma(c_1) \times \{v_2\}\big) \uplus \big(\{c_1\} \times \Gamma(v_2)\big)$ be the support of the generator (c_1, v_2) and let $F' := S \setminus F$. The sets F and F' are equivalent and F is $\|\ \|$-reduced since it is a subset of the $\|\ \|$-reduced set E_{R} (see Lemma 4.15) hence the following two properties hold:

$$\|F\| \leq \|F'\|, \qquad\qquad \|F\| + \|F'\| = \|S\| = 2.$$

As a consequence:

$$\|F\| \leq 1, \qquad\qquad \|\Gamma_1\| \leq 1 - \|\Gamma_2\|.$$

A function analysis shows that $\|\Gamma_1\| \mapsto \dfrac{\|\Gamma_1\|\|\Gamma_2\|}{\|\Gamma_1\| + \|\Gamma_2\|}$ is non-decreasing and thus:

$$\frac{\|\Gamma_1\|\|\Gamma_2\|}{\|\Gamma_1\| + \|\Gamma_2\|} \leq \Big(1 - \|\Gamma_2\|\Big)\|\Gamma_2\| \leq \frac{1}{4}.$$

Lemma 4.19 (i) is a consequence of the latter upper bound and eq. (4.29). We also have $F \in \mathcal{F}$ by Lemma 4.14 (iv):

$$|\sigma_X(F)| \geq \frac{1}{2} d_V d_C \|F\| \geq \frac{d_V}{2}|F|.$$

For Lemma 4.19 (ii), we lower bound $\Delta\big(\sigma_X(E_{\mathrm{R}}), F\big)$ using the inequalities $\|\widetilde{\Gamma}_1\| \leq 2\delta$ and $\|\widetilde{\Gamma}_2\| \leq 2\delta$:

$$\begin{aligned}
\Delta\big(\sigma_X(E_{\mathrm{R}}), F\big) &\geq |\Gamma_1||\overline{\Gamma}_2| + |\Gamma_2||\overline{\Gamma}_1| - |\Gamma_1||\widetilde{\Gamma}_2| - |\Gamma_2||\widetilde{\Gamma}_1| \\
&= |\sigma_X(F)| - 2 d_V d_C \Big(\|\Gamma_1\|\|\widetilde{\Gamma}_2\| + \|\Gamma_2\|\|\widetilde{\Gamma}_1\|\Big) \\
&\geq |\sigma_X(F)| - 4\delta d_V d_C \Big(\|\Gamma_1\| + \|\Gamma_2\|\Big) \\
&= |\sigma_X(F)| - 4\delta d_V d_C \|F\|.
\end{aligned}$$

□

By Lemma 4.19, for an error $E_R \subseteq V_{\mathcal{Q}}$ satisfying the right hypothesis, there exists a small-set $F \subseteq E_{\mathrm{R}}$ such that flipping F decreases the syndrome weight. When we apply Lemma 4.19 iteratively, we can write E_{R} as a disjoint union of small-sets as shown in Lemma 4.20 below.

Lemma 4.20. *We use Notation 4.12.*
Let $E_{\mathrm{R}} \subseteq V_{\mathcal{Q}}$ be a $\|\ \|$-reduced error such that $\big\|E_{\mathrm{R}}\big\| \leq \gamma_0\sqrt{N}/d_V$. Then there exists an integer $f' \in \mathbb{N}$ and small-sets $F_0, \ldots, F_{f'-1} \in \mathcal{F}$ such that:

$$\textit{(i)}\ \ E_{\mathrm{R}} = \biguplus_{i=0}^{f'-1} F_i, \qquad \textit{(ii)}\ \ \big|\sigma_X(E_{\mathrm{R}})\big| \geq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - \sum_{i=0}^{f'-1} 4\delta d_V d_C \big\|F_i\big\|.$$

In addition we have:

$$\textit{(iii)}\ \ \big|\sigma_X(E_{\mathrm{R}})\big| \geq \frac{\beta_0 d_V d_C}{2}\big\|E_{\mathrm{R}}\big\|.$$

Proof. We use an induction to define the small-sets $F_0, \ldots, F_{f'-1} \in \mathcal{F}$ as well as some $\|\ \|$-reduced sets $E_0, \ldots, E_{f'} \subseteq V_{\mathcal{Q}}$ with $\big\|E_i\big\| \leq \gamma_0\sqrt{N}/d_V$.
Let $E_0 := E_{\mathrm{R}}$. By hypothesis, E_0 is $\|\ \|$-reduced and $\big\|E_0\big\| \leq \gamma_0\sqrt{N}/d_V$.
Assume by the induction hypothesis that the $\|\ \|$-reduced set E_i has already been defined and satisfies $\big\|E_i\big\| \leq \gamma_0\sqrt{N}/d_V$. Lemma 4.19 applied to E_i provides the desired small-set $F_i \subseteq E_i$ and we define $E_{i+1} := E_i \oplus F_i = E_i \setminus F_i$. The set E_{i+1} is $\|\ \|$-reduced as a subset of the $\|\ \|$-reduced set E_i (see Lemma 4.15) and satisfies $\big\|E_{i+1}\big\| \leq \big\|E_i\big\| \leq \gamma_0\sqrt{N}/d_V$.
Let $f' \in \mathbb{N}$ be the number of rounds after which Lemma 4.19 cannot be applied anymore. The set $E_{f'}$ is also $\|\ \|$-reduced and satisfies $\big\|E_{f'}\big\| \leq \gamma_0\sqrt{N}/d_V$. Hence we necessarily have $\big\|E_{f'}\big\| = 0$ and thus item (i) holds. Item (ii) holds by Lemma 4.19 (ii):

$$\big|\sigma_X(E_{\mathrm{R}})\big| = \big|\sigma_X(E_0)\big| = \sum_{i=0}^{f'-1} \Delta(\sigma_X(E_i), F_i) \geq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - \sum_{i=0}^{f'-1} 4\delta d_V d_C \big\|F_i\big\|.$$

Finally, when we combine items (i) and (ii) with Lemma 4.19 (i) we get item (iii):

$$\begin{aligned}\big|\sigma_X(E_{\mathrm{R}})\big| \geq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - \sum_{i=0}^{f'-1} 4\delta d_V d_C \big\|F_i\big\| &\geq \frac{\beta_0 d_V d_C}{2} \sum_{i=0}^{f'-1} \big\|F_i\big\| \\ &= \frac{\beta_0 d_V d_C}{2}\big\|E_{\mathrm{R}}\big\|.\end{aligned}$$

□

Item (iii) of Lemma 4.20 asserts that the syndrome weight of a small $\|\ \|$-reduced error E_{R} can be lower bounded by a linear function of $\big\|E_{\mathrm{R}}\big\|$. This property is called *soundness* and, as stated in Lemma 4.21, a similar lower bound can be derived using the usual cardinality $|\cdot|$ instead of $\|\cdot\|$.

Lemma 4.21 (Soundness, Corollary 9 of [67])**.** *We use Notation 4.12 and $|\cdot|_{\mathrm{R}}$ from Definition 4.13.*
If $E \subseteq V_{\mathcal{Q}}$ satisfies $\big|E\big|_{\mathrm{R}} \leq \gamma_0\sqrt{N}$ then:

$$\big|\sigma_X(E)\big| \geq \frac{\beta_0 d_V}{2}\big|E\big|_{\mathrm{R}}.$$

Proof. Let E_1 be a reduced error equivalent to E and let E_2 be a $\|\ \|$-reduced error equivalent to E then:

$$\|E_2\| \leq \|E_1\| \leq \frac{1}{d_V}|E_1| = \frac{1}{d_V}|E|_{\mathrm{R}} \leq \frac{\gamma_0}{d_V}\sqrt{N}.$$

We can conclude using Lemma 4.20 (iii):

$$|\sigma_X(E)| = |\sigma_X(E_2)| \geq \frac{\beta_0 d_V d_C}{2}\|E_2\| \geq \frac{\beta_0 d_V}{2}|E_2| \geq \frac{\beta_0 d_V}{2}|E_1| = \frac{\beta_0 d_V}{2}|E|_{\mathrm{R}}.$$

□

To conclude this section, we prove Proposition 4.16.

Proof of Proposition 4.16. We run the small-set-flip algorithm (Algorithm 2) on input E, we denote by $\hat{E} \subseteq V_{\mathcal{Q}}$ the output of the algorithm, by $f \in \mathbb{N}$ the number of rounds and we use the notations σ_i and F_i from the body of Algorithm 2.
We also define $U := E \cup F_0 \cup \ldots \cup F_{f-1}$ called *the execution support*. This set will be a key notion when we will deal with stochastic noise in Section 5.1. Informally, a qubit is an element of $U \subseteq V_{\mathcal{Q}}$ if and only if there is an error on that qubit at some point of the algorithm.
Each F_i belongs to the set $\mathcal{F} := \left\{F \in \mathcal{F}_0 : |\sigma_X(F)| \geq \frac{d_V}{2}|F|\right\}$ thus we have:

$$|\sigma_i| - |\sigma_{i+1}| = \Delta(\sigma_i, F_i) \geq \beta_0|\sigma_X(F_i)| \geq \frac{\beta_0 d_V}{2}|F_i|.$$

Hence we can lower bound the weight of the initial syndrome by:

$$|\sigma_X(E)| = |\sigma_0| \geq |\sigma_0| - |\sigma_f| = \sum_{i=0}^{f-1}|\sigma_i| - |\sigma_{i+1}| \geq \frac{\beta_0 d_V}{2}\sum_{i=0}^{f-1}|F_i|.$$

But $|\sigma_X(E)| \leq d_C|E|$ thus $\sum_i |F_i| \leq \frac{2}{\beta_0 r}|E|$ and the size of the execution support is upper bounded linearly with the size of the initial error:

$$|U| \leq |E| + \sum_{i=0}^{f-1}|F_i| \leq \frac{1}{2\alpha_0}|E|.$$

Let E_{R} be a $\|\ \|$-reduced error equivalent to $E \oplus \hat{E}$ then:

$$\|E_{\mathrm{R}}\| \leq \|E \oplus \hat{E}\| \leq \frac{1}{d_V}|E \oplus \hat{E}| \leq \frac{1}{d_V}|U| \leq \frac{1}{2d_V\alpha_0}|E| \leq \frac{\gamma_0\sqrt{N}}{d_V}.$$

To show Proposition 4.16, it is sufficient to prove that $E_{\mathrm{R}} = \varnothing$. Suppose by contradiction that $\|E_{\mathrm{R}}\| \neq 0$ then all the hypotheses of Lemma 4.19 hold and thus there exists $F \in \mathcal{F}$ such that:

$$\Delta\big(\sigma_X(E_{\mathrm{R}}), F\big) \geq |\sigma_X(F)| - 4\delta d_V d_C\|F\| \geq \beta_0|\sigma_X(F)|.$$

In other words, the while loop condition of the small-set-flip algorithm (Algorithm 2) is satisfied for $\sigma_i = \sigma_X(E_{\mathrm{R}})$. But since $\hat{E}$ is the output of the algorithm, the while loop condition is not satisfied for $\sigma_i = \sigma_X(E \oplus \hat{E}) = \sigma_X(E_{\mathrm{R}})$ and we get a contradiction. □

Chapter 5

Small-set-flip algorithm with noisy syndrome measurements

In this chapter are presented the results we get during this PhD concerning the decoding of quantum expander codes with the small-set-flip algorithm. These results are summarized by the following points:

- Analysis of the decoder when the error on the qubits is generated with a local stochastic noise model.
- Analysis of the decoder when the syndrome measurements are noisy.
- Showing that the decoder has the *single-shot* property.
- Parallelization of the decoder.
- Numerical simulations of the decoder.

We start this introduction with a discussion about local stochastic error models. We use the small-set-flip algorithm to correct a quantum expander code with block-length N subjected to bit-flip errors. In Section 4.3, we have already reported the results of ref. [67] where any adversarial error with weight $\mathcal{O}(\sqrt{N})$ is shown to be corrected. However, in a practical situation such as fault-tolerant quantum computation (where the goal is to build quantum circuits working even if their basic components are noisy), a natural hypothesis is to assume that each qubit has a non-zero probability to be in the error support. For such an error model, the results of Section 4.3 are not sufficient because a typical error has linear weight.

One of the noise models we could think about is the iid error model, where each qubit belongs to the error support with some probability $p_{\text{phys}} \in (0, 1]$ independently from the other qubits. However, the independence assumption is not suitable for fault-tolerant quantum computation. For instance, a controlled-not gate acts on two qubits and, when this gate is noisy, it will probably create a correlated error on the qubits (here we are

talking about correlation in the probability distribution of the support, not about quantum correlations). On the other hand, we cannot ask a family of quantum codes to correct an error with arbitrary correlations. Take as an example the noise model where the error is empty with probability $1 - p_{\text{phys}}$ and a logical error occurs with probability p_{phys}. Even though each qubit belongs to the error support with probability at most p_{phys}, no code family can correct the resulting error with high probability.

In this manuscript, the only constraint we ask for a noise model to be valid is to satisfy the *local stochastic* property defined below. In addition, we will also assume that the small-set-flip decoder takes as input a noisy syndrome. This hypothesis is necessary in the context of fault-tolerant quantum computation because the circuit used to measure the stabilizer generators is noisy. Let $V_{\mathcal{Q}}$ be the set of physical qubits and let C_X be the set of check-nodes, then the error is described by two random sets $E \subseteq V_{\mathcal{Q}}$ and $D \subseteq C_X$. The set E contains the qubits affected by an X-Pauli error and D is the set of faulty syndrome bits. A noise model is *local stochastic* with parameter $(p_{\text{phys}}, p_{\text{synd}})$ when for all $S \subseteq V_{\mathcal{Q}}$ and for all $T \subseteq C_X$:

$$\mathbb{P}\Big[S \subseteq E \text{ and } T \subseteq D\Big] \leq p_{\text{phys}}^{|S|} p_{\text{synd}}^{|T|}. \tag{5.1}$$

Informally, eq. (5.1) means that the probability for an error pattern to be in the error support decreases exponentially with its size.

Let $E \subseteq V_{\mathcal{Q}}$ and $D \subseteq C_X$ be generated with a local stochastic error model with parameter $(p_{\text{phys}}, p_{\text{synd}})$ and let $\sigma_X(E)$ be the syndrome of E. The small-set-flip algorithm takes as input the observed syndrome $\sigma_X(E) \oplus D$ and outputs some set $\hat{E} \subseteq V_{\mathcal{Q}}$ on which X-Pauli matrices are applied to try to correct E. The set $E \oplus \hat{E}$ is called the *residual error* and represents the qubits on which a physical error remains after correction. We will show the existence of a threshold $p_0 > 0$ such that if $p_{\text{phys}} < p_0$ and $p_{\text{synd}} < p_0$ then the residual error is described by a local stochastic noise with controlled parameter. In addition, the correction is performed with a single round of syndrome measurement, this feature is called *single-shot error correction* [12].

We will also explain how the small-set-flip decoder can be parallelized. On the one hand, if the syndrome measurement is perfect, running the parallel algorithm for a logarithmic number of rounds is sufficient to correct entirely the error with high probability (i.e. the residual error is equivalent to the empty error with high probability). On the other hand, if the syndrome measurement is noisy then, with high probability, running the parallel algorithm for a constant number of rounds leads to a residual error described by a local stochastic error model with small parameter.

When we compute numerical lower bounds on the threshold p_0 by theoretical arguments, the value we get is not reasonable ($\sim 10^{-58}$). Hence, in order to get an idea on the real value of p_0, we have performed some simulations for the sequential algorithm with perfect syndrome measurements. Our results yield a threshold of 4.5% for a family of quantum expander codes with rate $1/61$ and 2% for a family with rate 20%.

As a conclusion, using ref. [46], our results allow to design fault-tolerant protocols with constant space overhead as described in Chapter 6. For more details and a technical

discussion about the content of this introduction, see Section 2.7. This chapter is organized as follows: Section 5.1 analyses the sequential small-set-flip decoder, Section 5.2 analyses the parallel version and Section 5.3 summarizes the numerical simulations.

5.1 Sequential decoding

The goal of this section is to show the following: when a local stochastic error is corrected by the small-set-flip algorithm with noisy syndrome measurements, the residual error $E \oplus \hat{E}$ is equivalent to a local stochastic error E_{ls}. By the definition given in eq. (5.1), E_{ls} is local stochastic if for all $S \subseteq V_{\mathcal{Q}}$, we can upper bound the probability of the event $\{S \subseteq E_{\text{ls}}\}$ with a quantity exponentially small in $|S|$. By assumption, for all $T \subseteq C_X$, we already have a similar upper bound on the probability for the event $\{T \subseteq D\}$ to happen. As shown in this section, for all $S \subseteq V_{\mathcal{Q}}$, the hypothesis $S \subseteq E \oplus \hat{E}$ implies the existence of a large set $T \subseteq D$. More precisely, we will find a set of qubits W that we will call a *witness* and the set T will be the check-nodes of D whose support contains a qubit of W. Finally, $\mathbb{P}[S \subseteq E_{\text{ls}}]$ is upper bounded by the probability for a witness to exist and, since the corresponding set T is large and satisfies $T \subseteq D$, we will show that this probability is exponentially small.

Before looking at local stochastic noise, the first part of the analysis deals with adversarial errors whose weight is below the minimal distance. In that case, the size of the residual error is shown to be upper bounded by a linear function of $|D|$. The proof for local stochastic noise will rely on this result and on arguments from percolation theory.

From now on, the small-set-flip decoder presented in Section 4.3 (Algorithm 2) will not be used anymore, it is replaced by Algorithm 3 below. The only difference between these two algorithms is the while loop condition where the constant $\beta_0 = 1 - 8\delta$ is replaced by $\beta_1 = 1/2 - 8\delta < \beta_0$ (the real number δ is the expansion parameter of the classical Tanner graph). In particular, since $\beta_1 > 0$ is required for the analysis of this chapter to work, δ must satisfy $\delta < 1/16$ whereas $\delta < 1/8$ was sufficient in the case of noiseless syndrome measurement. We also change the notation σ to $\widetilde{\sigma}$ to remind the reader that $\widetilde{\sigma}$ represents a faulty syndrome.

At each round of Algorithm 3, an element of $\mathcal{F}$ is flipped where $\mathcal{F}$ is the small-set ensemble as defined in eq. (4.27):

$$\mathcal{F} := \left\{ F \subseteq \Gamma_Z(g) : g \in C_Z \text{ and } \left|\sigma_X(F)\right| \geq \frac{d_V}{2}\left|F\right| \right\}.$$

In addition, for all $\widetilde{\sigma} \subseteq C_X$ and $F \in \mathcal{F}$, the quantity $\Delta(\widetilde{\sigma}, F)$ used in Algorithm 3 has been defined in eq. (4.26) by:

$$\Delta(\widetilde{\sigma}, F) := \left|\widetilde{\sigma}\right| - \left|\widetilde{\sigma} \oplus \sigma_X(F)\right|.$$

When the observed syndrome is $\widetilde{\sigma}$ and the small-set F is flipped, the syndrome weight decreases by $\Delta(\widetilde{\sigma}, F)$ units.

Algorithm 3 : the small-set-flip decoder for noisy syndrome measurements.

Input: the observed syndrome $\widetilde{\sigma} \subseteq C_X$. // $\widetilde{\sigma} = \sigma(E) \oplus D$ for some $E \subseteq V_{\mathcal{Q}}, D \subseteq C_X$
Output: a guess for the error $\hat{E} \subseteq V_{\mathcal{Q}}$.

$\hat{E}_0 = \varnothing$; $\widetilde{\sigma}_0 = \widetilde{\sigma}$; $i = 0$
while $\exists F_i \in \mathcal{F} : \Delta(\widetilde{\sigma}_i, F_i) \geq \beta_1 \left|\sigma_X(F_i)\right|$ **do**
  Pick such an F_i arbitrarily.
  $\hat{E}_{i+1} = \hat{E}_i \oplus F_i$
  $\widetilde{\sigma}_{i+1} = \widetilde{\sigma}_i \oplus \sigma_X(F_i)$
  $i = i + 1$
end while
return $\hat{E}_i$

This section is organized as follows: we focus on adversarial errors with weight below the minimal distance in Section 5.1.2 and we analyze the local stochastic error model in Section 5.1.3.

5.1.1 Notations

All the statements presented in Section 5.1 will use the notations given below.

The error correcting code. Let G be a (γ, δ)-expander graph with $\gamma > 0$ and $\delta < 1/16$ seen as a Tanner graph. We denote by V its set of bit-nodes, by C its set of check-nodes, by d_V its left degree, by d_C its right degree and by $\mathcal{Q}$ the associated quantum expander code. Let $V_{\mathcal{Q}}$ be the set of qubits, let C_X be the set of Z-type stabilizer generators, let C_Z be the set of Z-type stabilizer generators and let $N := \left|V_{\mathcal{Q}}\right|$ be the number of physical qubits of $\mathcal{Q}$.
We also use the notations $\|\cdot\|$ and $|\cdot|_{\mathrm{R}}$ from Definition 4.13.

The decoder. We run Algorithm 3 with the observed syndrome $\widetilde{\sigma} := \sigma_X(E) \oplus D$ as input where $E \subseteq V$ represents the error on the qubits and $D \subseteq C_X$ represents the error on the syndrome bits.

For simplicity, we say that the input of the decoder is (E, D) instead of $\sigma(E) \oplus D$. We denote by $\hat{E}$ its output, by f the number of rounds and by $F_0, \ldots, F_{f-1}$ the small-sets flipped by Algorithm 3. We will also use the following variables from the body of Algorithm 3:

$$\hat{E}_i := F_0 \oplus \ldots \oplus F_{i-1}, \qquad \widetilde{\sigma}_i := \sigma_X\left(E \oplus \hat{E}_i\right) \oplus D.$$

The set $U := E \cup F_0 \cup \ldots \cup F_{f-1}$ is called the *execution support* and $E \oplus \hat{E}$ is called the *residual error*.

Useful constants. We define the following positive constants:

$$r := d_V/d_C, \qquad \gamma_0 = \frac{r^2}{\sqrt{1+r^2}}\gamma, \qquad \beta_0 := 1 - 8\delta, \qquad \beta_1 := \frac{1-16\delta}{2},$$

$$c_0 := \frac{4}{d_V \beta_1}, \qquad c_1 := \frac{\beta_1}{\beta_0(1-\beta_1)}, \qquad c_2 := \frac{2}{1-\beta_1}, \qquad \alpha_0 := \frac{r\beta_0}{4+2r\beta_0},$$

$$\alpha_1 := \frac{r\beta_1}{4+2r\beta_1}, \quad c_3 := \frac{1}{2\alpha_1}, \quad c_4 := \frac{1}{2\alpha_1} - 1, \quad d_{\mathcal{G}} := d_C(d_C + 2d_V - 2).$$

The syndrome adjacency graph. The notion of syndrome adjacency graph will be used in Section 5.1.3. We will perform a percolation process on its vertices to study the typical shape of a random error.
We define $\mathcal{G}$ called the *syndrome adjacency graph* of the code in the following way: $\mathcal{G}$ is equal to the Tanner graph of $\mathcal{C}_X$ (see the left part of Figure 4.5) with additional edges between the qubits which share an X-type or a Z-type generator. In other words, the set of vertices of $\mathcal{G}$ is indexed by $\mathcal{V} := V \cup C_X$, a Z-type generator is incident to the qubits in its support and two qubits are linked when they are both in the support of the same generator. We denote by $\Gamma_{\mathcal{Q}}$ the neighborhood in the Tanner graph of $\mathcal{Q}$, by Γ_X the neighborhood in the Tanner graph of $\mathcal{C}_X$, by $\Gamma_{\mathcal{G}}$ the neighborhood in the graph $\mathcal{G}$ and we point out that the degree of $\mathcal{G}$ is upper bounded by the integer $d_{\mathcal{G}}$ defined above.

5.1.2 Errors below minimal distance

The first step in the analysis of Algorithm 3 is to address the case where the error has weight $\mathcal{O}(\sqrt{N})$. In Section 4.3.3, we assumed perfect syndrome measurement and in this section, we tackle the case where the syndrome is noisy. The main result is Corollary 5.4 where we show that the residual error $E \oplus \hat{E}$ satisfies $|E \oplus \hat{E}|_{\mathrm{R}} \leq c_0|D|$ where $|\cdot|_{\mathrm{R}}$ is the reduced weight of Definition 4.13. In other words, the residual error is equivalent to an error E' satisfying $|E'| \leq c_0|D|$.

Let $E_i := E \oplus \hat{E}_i$ be the physical error on the qubits at round i. In what follows, a small-set is said to be *valid* for Algorithm 2 (resp. Algorithm 3) when it satisfies the while loop condition of Algorithm 2 (resp. Algorithm 3). Hence, a small-set $F \in \mathcal{F}$ is valid if and only if the difference between $|\sigma_X(E_i)|$ and $|\sigma_X(E_i \oplus F)|$ is at least $\beta_0|\sigma_X(F)|$ (resp. $\beta_1|\sigma_X(F)|$). In Section 4.3.3 where we wanted to show that the residual is trivial (*i.e.* equivalent to an empty error), the critical point was to prove that Algorithm 2 does not stop as long as E_i is not trivial. It is indeed the case by Lemma 4.19: if E_i is not trivial then a valid small-set for Algorithm 2 exists. Similarly here, we will show that the condition $|E_i|_{\mathrm{R}} > c_0|D|$ implies the existence of a valid small-set for Algorithm 3. When $|E_i|_{\mathrm{R}} > c_0|D|$, the set E_i is not trivial and we have a valid small-set F for Algorithm 2, but unfortunately, since the check-nodes adjacent to F could be affected by the syndrome error, F is not guaranteed to be valid for Algorithm 3 as well.

The solution is to strengthen the statement of Lemma 4.19: we decompose the error as a disjoint union of small-sets $F_0, \ldots, F_{f'-1}$ (as we did in Lemma 4.20) and we show that several small-sets among $F_0, \ldots, F_{f'-1}$ are valid for Algorithm 3. The technical idea is to compute the average of $\left|\widetilde{\sigma} \cap \sigma_X(F_i)\right|$ over the F_i to give a lower bound on the number of F_i which are valid for Algorithm 3. Finally we get Lemma 5.1 below:

there exists a set of small-sets $\mathcal{F}^* \subseteq \mathcal{F}$ which are valid for Algorithm 3. More precisely, the small-sets belonging to $\mathcal{F}^*$ are valid for Algorithm 3 by item (i), these small-sets do not intersect by item (ii) and item (iii) provides a lower bound on the size of $\mathcal{F}^*$. Note that if $c_1\big|\sigma_X(E)\big| < c_2\big|D\big|$ then the lemma is pointless since $\mathcal{F}^*$ could be empty, but otherwise the lower bound of item (iii) is non-trivial. As a summary, Lemma 5.1 improves Lemma 4.19 in two ways: firstly, it can be applied even if the syndrome is noisy and secondly, it provides many valid small-sets instead of a single one. This second point will be useful to parallelize the algorithm in Section 5.2.

Lemma 5.1. *We use the notations of Section 5.1.1.*
If $\big|E\big| \leq \gamma_0\sqrt{N}$ *then there exists* $\mathcal{F}^* \subseteq \mathcal{F}$ *such that:*

(i) $\Delta(\widetilde{\sigma}, F) \geq \beta_1\big|\sigma_X(F)\big|$ *for all* $F \in \mathcal{F}^*$,

(ii) $F \cap F' = \varnothing$ *for all* $F, F' \in \mathcal{F}^*$ *with* $F \neq F'$,

(iii) $\displaystyle\sum_{F\in\mathcal{F}^*} \big|\sigma_X(F)\big| \geq c_1\big|\sigma_X(E)\big| - c_2\big|D\big|$.

Proof. Let E_{R} be the $\|\ \|$-reduced error equivalent to E then:

$$\big\|E_{\mathrm{R}}\big\| \leq \big\|E\big\| \leq \frac{1}{d_V}\big|E\big| \leq \frac{\gamma_0}{d_V}\sqrt{N}.$$

Applying Lemma 4.20 to E_{R} provides some small-sets $F_0, \ldots, F_{f'-1}$.
Let us start by providing an overview of how we will proceed in this proof. By Lemma 4.20 (i) we have:

$$\sigma_X(E) = \sigma_X(E_{\mathrm{R}}) = \bigoplus_{i=0}^{f'-1} \sigma_X(F_i)$$

and a union bound yields

$$\big|\sigma_X(E)\big| \leq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big|.$$

In fact, we will prove in eq. (5.2) below that that this upper bound is nearly tight: $\big|\sigma_X(E)\big| \geq \beta_0 \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big|$ where β_0 defined in Section 5.1.1 is arbitrarily close to 1 when δ is small. Intuitively, this means that the intersection of the sets $\sigma_X(F_i)$ is small and thus $\big|\sigma_X(E) \cap \sigma_X(F_i)\big|$ is generally large. This is still true if the size of the syndrome error D is small, *i.e.*, it holds that $\big|\widetilde{\sigma} \cap \sigma_X(F_i)\big|$ is generally large. Hence, we will obtain Lemma 5.1 by computing the average of the quantity $\big|\widetilde{\sigma} \cap \sigma_X(F_i)\big|$ over the

sets F_i. We now provide the details:

$$\begin{aligned}
\big|\sigma_X(E)\big| = \big|\sigma_X(E_{\mathrm{R}})\big| &\geq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - \sum_{i=0}^{f'-1} 4\delta d_V d_C \big\|F_i\big\| && \text{by Lemma 4.20 (ii),}\\
&= \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - 4\delta d_V d_C \big\|E_{\mathrm{R}}\big\| && \text{by Lemma 4.20 (i),}\\
&\geq \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| - \frac{8\delta}{\beta_0}\big|\sigma_X(E)\big| && \text{by Lemma 4.20 (iii).}
\end{aligned}$$

Hence we have:

$$\big|\sigma_X(E)\big| \geq \left(1 + \frac{8\delta}{\beta_0}\right)^{-1} \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big| = \beta_0 \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big|. \tag{5.2}$$

The relation between $\Delta(\widetilde{\sigma}, F)$ and $\big|\widetilde{\sigma} \cap \sigma_X(F)\big|$ is given by:

$$\Delta(\widetilde{\sigma}, F) = \big|\widetilde{\sigma}\big| - \big|\widetilde{\sigma} \oplus \sigma_X(F)\big| = 2\big|\widetilde{\sigma} \cap \sigma_X(F)\big| - \big|\sigma_X(F)\big| \tag{5.3}$$

where we have used the equality $\big|A_1 \oplus A_2\big| = \big|A_1\big| + \big|A_2\big| - 2\big|A_1 \cap A_2\big|$.
We define the set $\mathcal{F}^*$ promised in Lemma 5.1 by:

$$\mathcal{F}^* := \Big\{F \in \{F_0, \ldots, F_{f'-1}\} : \Delta(\widetilde{\sigma}, F) \geq \beta_1 \big|\sigma_X(F)\big|\Big\}.$$

Lemma 5.1 (i) holds by definition of $\mathcal{F}^*$. Lemma 5.1 (ii) holds because the property $E_{\mathrm{R}} = \biguplus_i F_i$ from Lemma 4.20 (i) implies that the sets F_i are disjoint. Moreover by eq. (5.3), when $F \in \{F_0, \ldots, F_{f'-1}\} \setminus \mathcal{F}^*$:

$$\big|\widetilde{\sigma} \cap \sigma_X(F)\big| \leq \frac{1 + \beta_1}{2}\big|\sigma_X(F)\big|. \tag{5.4}$$

On the one hand, eqs. (5.2) and (5.4) give an upper bound on the sum $S := \sum_{i=0}^{f'-1} \big|\widetilde{\sigma} \cap \sigma_X(F_i)\big|$:

$$\begin{aligned}
S &= \sum_{F_i \in \mathcal{F}^*} \big|\widetilde{\sigma} \cap \sigma_X(F_i)\big| + \sum_{F_i \notin \mathcal{F}^*} \big|\widetilde{\sigma} \cap \sigma_X(F_i)\big|\\
&\leq \sum_{F_i \in \mathcal{F}^*} \big|\sigma_X(F_i)\big| + \frac{1 + \beta_1}{2} \sum_{F_i \notin \mathcal{F}^*} \big|\sigma_X(F_i)\big| && \text{by eq. (5.4),}\\
&= \frac{1 - \beta_1}{2} \sum_{F_i \in \mathcal{F}^*} \big|\sigma_X(F_i)\big| + \frac{1 + \beta_1}{2} \sum_{i=0}^{f'-1} \big|\sigma_X(F_i)\big|\\
&\leq \frac{1 - \beta_1}{2} \sum_{F_i \in \mathcal{F}^*} \big|\sigma_X(F_i)\big| + \frac{1 + \beta_1}{2\beta_0}\big|\sigma_X(E)\big| && \text{by eq. (5.2).}
\end{aligned}$$

On the other hand, the property $E_{\mathrm{R}} = \biguplus_i F_i$ from Lemma 4.20 (i) implies that $\sigma_X(E) = \sigma_X(E_{\mathrm{R}}) = \bigoplus_{i=0}^{f'-1} \sigma_X(F_i)$ and thus S is lower bounded by:

$$\begin{aligned} S &\geq \big|\widetilde{\sigma} \cap \sigma_X(E)\big| \\ &= \big|(\sigma_X(E) \oplus D) \cap \sigma_X(E)\big| \\ &= \big|\sigma_X(E) \oplus (D \cap \sigma_X(E))\big| \\ &= \big|\sigma_X(E)\big| - \big|D \cap \sigma_X(E)\big| \\ &\geq \big|\sigma_X(E)\big| - \big|D\big|. \end{aligned}$$

Combining both inequalities we get Lemma 5.1 (iii).

□

The main statement of this section is presented in Proposition 5.2 and asserts that the weight of the residual error after correction is upper bounded linearly in the syndrome error weight. In Corollary 5.4, we will give another formulation of Proposition 5.2 where the hypothesis $\big|U\big| = \mathcal{O}(\sqrt{N})$ is replaced by the more natural one $\big|E\big|, \big|D\big| = \mathcal{O}(\sqrt{N})$.

Proposition 5.2. *We use the notations of Section 5.1.1.*
Suppose that $E \oplus \hat{E}$ is reduced with $\big|U\big| \leq \gamma_0\sqrt{N}$ then:

$$\big|E \oplus \hat{E}\big| \leq c_0\big|D\big|.$$

Proof. With the notations of Section 5.1.1, the value of the syndrome at the end of the algorithm is $\widetilde{\sigma}_f = \sigma_X(E \oplus \hat{E}) \oplus D$. Lemma 5.1 applied when the input of Algorithm 3 is $(E \oplus \hat{E}, D)$ provides a set $\mathcal{F}^*$ and since the while loop condition is not satisfied for $\widetilde{\sigma}_f$, the set $\mathcal{F}^*$ must be empty. By Lemma 5.1 (iii):

$$c_1\big|\sigma_X(E \oplus \hat{E})\big| \leq c_2\big|D\big|.$$

By Lemma 4.21:

$$\big|E \oplus \hat{E}\big| \leq \frac{2}{\beta_0 d_V}\big|\sigma_X(E \oplus \hat{E})\big| \leq \frac{2c_2}{c_1\beta_0 d_V}\big|D\big| = c_0\big|D\big|.$$

□

As shown in Lemma 5.3 below, the hypothesis of Proposition 5.2 $\big|U\big| = \mathcal{O}(\sqrt{N})$ holds as soon as the initial errors E and D are sufficiently small. More precisely, the size of the execution support U is a linear function of $\big|E\big|$ and $\big|D\big|$.

Lemma 5.3. *We use the notations of Section 5.1.1 then:*

$$\big|U\big| \leq c_3\big|E\big| + c_4\big|D\big|.$$

Proof. The sets $\widetilde{\sigma}_i$ and F_i defined in Section 5.1.1 satisfy:

$$\big|\widetilde{\sigma}_i\big| - \big|\widetilde{\sigma}_{i+1}\big| \geq \beta_1\big|\sigma_X(F_i)\big| \geq \frac{\beta_1 d_V}{2}\big|F_i\big|.$$

Thus we can lower bound the weight of $\widetilde{\sigma}_0 = \sigma_X(E) \oplus D$ by:

$$\big|\sigma_X(E) \oplus D\big| \geq \big|\widetilde{\sigma}_0\big| - \big|\widetilde{\sigma}_f\big| = \sum_{i=0}^{f-1} \big|\widetilde{\sigma}_i\big| - \big|\widetilde{\sigma}_{i+1}\big| \geq \frac{\beta_1 d_V}{2} \sum_{i=0}^{f-1} \big|F_i\big|.$$

But $|\sigma_X(E) \oplus D| \leq d_C|E| + |D| \leq d_C\big(|E| + |D|\big)$ thus $\sum_i |F_i| \leq \frac{2}{\beta_1 r}\big(|E| + |D|\big)$ and:

$$\big|U\big| + \big|D\big| \leq \big|E\big| + \big|D\big| + \sum_{i=0}^{f-1} \big|F_i\big| \leq \frac{\big|E\big| + \big|D\big|}{2\alpha_1}.$$

Hence:

$$\big|U\big| \leq \frac{1}{2\alpha_1}\big|E\big| + \left(\frac{1}{2\alpha_1} - 1\right)\big|D\big| = c_3\big|E\big| + c_4\big|D\big|.$$

□

Combining Proposition 5.2 and Lemma 5.3, we get Corollary 5.4.

Corollary 5.4. *We use the notations of Section 5.1.1.*
If $c_3\big|E\big| + c_4\big|D\big| \leq \gamma_0\sqrt{N}$ then the reduced weight of the residual error satisfies:

$$\big|E \oplus \hat{E}\big|_{\mathrm{R}} \leq c_0\big|D\big|.$$

Proof. Let E' be a reduced error equivalent to $E \oplus \hat{E}$ then:

$$\big|E'\big| \leq \big|E \oplus \hat{E}\big| \leq \big|U\big| \leq c_3\big|E\big| + c_4\big|D\big| \leq \gamma_0\sqrt{N}$$

where the third inequality holds by Lemma 5.3. The syndromes of the errors E' and $E \oplus \hat{E}$ are equal thus no flip is done by Algorithm 3 on the input (E', D). Hence, there is a valid execution of Algorithm 3 on the input (E', D) with output $\varnothing$, support E' and residual error E'. Applying Proposition 5.2 to the latter execution we get:

$$\big|E \oplus \hat{E}\big|_{\mathrm{R}} = \big|E'\big| \leq c_0\big|D\big|.$$

□

5.1.3 Random errors with linear size

The upper bound given by Corollary 5.4 can be applied for any error with weight $\mathcal{O}\big(\sqrt{N}\big)$, but in this work we are interested in random errors whose typical weight is $\Theta(n)$. The only assumption we do on the error is that it is generated by a noise model satisfying the *local stochastic* condition of Definition 5.5 (see the introduction of this chapter for a discussion about the local stochastic property).

Definition 5.5 (Local stochastic error model)**.** We use the notations of Section 5.1.1. A random error $E \subseteq V_{\mathcal{Q}}$, $D \subseteq C_X$ is local stochastic with parameter $(p_{\text{phys}}, p_{\text{synd}})$ when for all $S \subseteq V_{\mathcal{Q}}$ and for all $T \subseteq C_X$:

$$\mathbb{P}\Big[S \subseteq E \text{ and } T \subseteq D\Big] \leq p_{\text{phys}}^{|S|} p_{\text{synd}}^{|T|}.$$

In the particular case where we do not mind about the error on the syndrome bits, a random error $E \subseteq V_{\mathcal{Q}}$ is local stochastic with parameter p_{phys} when for all $S \subseteq V_{\mathcal{Q}}$:

$$\mathbb{P}\Big[S \subseteq E\Big] \leq p_{\text{phys}}^{|S|}.$$

a) Statement of the main theorem

The main theorem of this section is Theorem 5.6 below: if the initial errors E and D satisfy the local stochastic property with parameters sufficiently small, then there exists a local stochastic error $E_{\text{ls}} \subseteq V_{\mathcal{Q}}$ such that E_{ls} and the residual error are equivalent with high probability. In other words, the residual error on the qubits can be described by a local stochastic noise, and this description is valid except for an event whose probability vanishes in the limit of large block-length (this event corresponds to a decoding failure).

Theorem 5.6. *We use the notations of Section 5.1.1.*
There exists a non-zero constant $p_0 > 0$ such that the following holds. Suppose the error (E, D) satisfies a local stochastic noise model with parameter $(p_{\text{phys}}, p_{\text{synd}})$ where $p_{\text{phys}} < p_0$ and $p_{\text{synd}} < p_0$. If we run Algorithm 3 on the input (E, D) then there exists a random variable $E_{\text{ls}} \subseteq V_{\mathcal{Q}}$ with a local stochastic distribution with parameter $p_{\text{ls}} := p_{\text{synd}}^{1/(2c_0)}$ such that:

$$\mathbb{P}\Big[E_{\text{ls}} \textit{ and } E \oplus \hat{E} \textit{ are not equivalent}\Big] \leq e^{-\Theta(\sqrt{N})}.$$

The proof of Theorem 5.6 is given at the end of this section.
It is worthwhile to rewrite Theorem 5.6 with the additional assumption that the syndrome measurement is perfect ($p_{\text{synd}} = 0$ and $D = \varnothing$). In that case, a local stochastic error whose parameter is below threshold is corrected by the small-set-flip decoder with high probability:

Corollary 5.7. *We use the notations of Section 5.1.1.*
There exists a non-zero constant $p_0 > 0$ such that the following holds. Suppose $D = \varnothing$ and E satisfies a local stochastic noise model with parameter $p_{\text{phys}} < p_0$ then:

$$\mathbb{P}\Big[\textit{Algorithm 3 corrects } E\Big] \geq 1 - e^{-\Theta(\sqrt{N})}.$$

Proof. When we apply Theorem 5.6 with $p_{\text{synd}} = 0$, we get $p_{\text{ls}} = 0$ and $E_{\text{ls}} = \varnothing$. Hence, with probability $1 - e^{-\Theta(\sqrt{N})}$, $E \oplus \hat{E}$ is equivalent to $\varnothing$, *i.e.* Algorithm 3 corrects E. □

When a quantum expander code $\mathcal{Q}$ is used in the context of fault-tolerance, the goal is to process the information encoded in $\mathcal{Q}$ by applying logical quantum gates on the qubits. Each logical operation adds a local stochastic error on the qubits, thus a round of error correction is performed after each logical gate. The concern in this discussion is to understand how evolves the error on the qubits: we would like to use Theorem 5.6 to show that the logical state encoded with $\mathcal{Q}$ is preserved after applying $T \in \mathbb{N}$ logical gates. A simplified error model for this framework is the following:

- At the beginning, there is no error on the qubits: $A_0 := \varnothing$.
- Repeat for $t = 0, \dots, T-1$:
  - An error (E_t, D_t) is generated accordingly to a local stochastic noise model with parameter (p_E, p_D).
  - The error E_t is added to the error on the qubits:
$$B_t := A_t \oplus E_t.$$
  - We run Algorithm 3 on the input (B_t, D_t) and apply the output $\hat{E}_t$ to the qubits:
$$A_{t+1} := B_t \oplus \hat{E}_t.$$

Roughly speaking, E_t comes from the noise in the circuit used to implement the t^{th} logical gate and D_t comes from the noise in the circuit used to measure the syndrome bits for the t^{th} error correction procedure. In fault-tolerant quantum computation, $\hat{E}_t$ is applied with a circuit made of noisy physical gates, but for simplicity here, we assume that $\hat{E}_t$ is applied without fault.

This procedure is a success when the logical state is preserved throughout the T rounds. In practice, we can determine whether the logical state has been preserved by checking if running the decoder without syndrome error turns the state back to the expected state. As a summary, if we run Algorithm 3 on the input $(A_T, \varnothing)$ and apply the output $\hat{E}_T$ to the qubits:

$$A_{T+1} := A_T \oplus \hat{E}_T,$$

then the above procedure is a success if and only if A_{T+1} is trivial (equivalent to the empty error). As shown in Claim 5.8 below, A_{T+1} is indeed trivial with probability $1 - (T+1)e^{-\Theta(\sqrt{N})}$. In particular, if the number of logical gates T is polynomial in N then the procedure fails with vanishing probability.

Claim 5.8. *Let p_0 be the threshold of Theorem 5.6 and Corollary 5.7, and let p_D and p_E be such that:*

$$p_D < \left(\frac{p_0}{2}\right)^{2c_0} \qquad \text{and} \qquad p_E < \frac{p_0}{2}.$$

Then the above procedure fails with probability at most $(T+1)e^{-\Theta(\sqrt{N})}$.

Proof. In this proof, we say that an error A is local stochastic with parameter p and failure probability p' if there exists an error A' with local stochastic parameter p such that:

$$\mathbb{P}\Big[A' \text{ and } A \text{ are not equivalent}\Big] \leq p'.$$

With this terminology, Theorem 5.6 asserts that the residual error $E \oplus \hat{E}$ is local stochastic with parameter p_{ls} and failure probability $e^{-\Theta(\sqrt{N})}$.

When $t = 0$, the error $A'_0 = E_0$ is local stochastic with parameter $p_E < p_0$ and failure probability 0. By Theorem 5.6, the residual error $A_1 = A'_0 \oplus \hat{E}_0$ is local stochastic with parameter $p_D^{1/(2c_0)}$ and failure probability $e^{-\Theta(\sqrt{N})}$.
When $t = 1$, the error $A'_1 = A_1 \oplus E_1$ is local stochastic with parameter $p_0/2 + p_E < p_0$ and failure probability $e^{-\Theta(\sqrt{N})}$. By Theorem 5.6, the residual error $A_2 = A'_1 \oplus \hat{E}_1$ is local stochastic with parameter $p_D^{1/(2c_0)}$ and failure probability $2e^{-\Theta(\sqrt{N})}$.
More generally, an induction on $t \in [\![1, T]\!]$ shows that the error E_t is local stochastic with parameter $p_D^{1/(2c_0)}$ and failure probability $te^{-\Theta(\sqrt{N})}$.
In particular for $t = T$, the error A_T is local stochastic with parameter $p_D^{1/(2c_0)}$ and failure probability $Te^{-\Theta(\sqrt{N})}$. By Corollary 5.7, the residual error $A_{T+1} = A_T \oplus \hat{E}_T$ is non trivial with probability at most $(T+1)e^{-\Theta(\sqrt{N})}$. Since the procedure is a success when A_{T+1} is non trivial, we get Claim 5.8.

□

b) Proof ideas and useful definitions

Let's turn our attention to the proof of Theorem 5.6 which is inspired from the arguments of [66] and [46]. Before providing the formal statements, we describe the main ideas of the proof. When a random error (E, D) is generated with a local stochastic error model, percolation arguments will allow us with high probability to decompose E and D as disjoint unions of small error sets, each of which has size $\mathcal{O}(\sqrt{N})$:

$$E = \uplus_K E_K, \qquad D = \uplus_K D_K. \tag{5.5}$$

As we explain in more details later, the index $K \subseteq V_{\mathcal{Q}}$ of eq. (5.5) goes through the connected components of U in the syndrome adjacency graph $\mathcal{G}$. Let $\hat{E}_K$ be the output of Algorithm 3 on the input (E_K, D_K). Because $|E_K|, |D_K| = \mathcal{O}(\sqrt{N})$, Corollary 5.4 states that after correction, the residual error $E_K \oplus \hat{E}_K$ is equivalent to E'_K satisfying $|E'_K| = \mathcal{O}(|D_K|)$. Moreover, Algorithm 3 is *local* in the sense that two errors far away in the factor graph of the code will not interact during the decoding procedure. The consequence for the error E is that each error set E_K is corrected independently from the other ones and thus the residual error $E \oplus \hat{E}$ is equal to the disjoint union:

$$E \oplus \hat{E} = \uplus_K E'_K.$$

Finally, since D has been chosen with a local stochastic noise, the upper bounds $|E'_K| = \mathcal{O}(|D_K|)$ imply that there is an error equivalent to $E \oplus \hat{E}$ with a local stochastic distribution.

The locality property for the small-set-flip algorithm is formalized in Lemma 5.9 where $\Gamma_{\mathcal{Q}}$ denotes the neighborhood in the Tanner graph of the quantum expander code. In this lemma, the sets E_K and D_K we talked about above are equal to $E_K = E \cap K$ and $D_K = D \cap \Gamma_X(K)$.

Lemma 5.9 (Locality of Algorithm 3 [38])**.** *We use the notations of Section 5.1.1. For any set $K \subseteq U$ with $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$, there is a valid execution of Algorithm 3 on the input $\big(E \cap K, D \cap \Gamma_X(K)\big)$ whose output is $\hat{E} \cap K$ and whose support is $U \cap K$.*

The locality property is important to understand this section, however the proof of Lemma 5.9 is less useful and that's why we report it later in Section 5.1.4.

Lemma 5.9 formalizes the idea that the small-set-flip algorithm handles the errors in the set K independently from the errors outside K. In fact, if $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$ holds then the support of a check-node cannot contain simultaneously a qubit of K and a qubit of $U \setminus K$. Moreover, to know whether the small-set-flip algorithm can flip a given small-set $F \subseteq K$, the only required information is the value of the check-nodes belonging to $\Gamma_{\mathcal{Q}}(K)$. As a consequence, the error on the qubits of $U \setminus K$ does not affect the decisions of the decoder regarding the qubits of K. In particular, the behavior of the algorithm in K would have been the same if there were no error at all on the qubits of $U \setminus K$, and this is the statement of Lemma 5.9.

In Section 5.1.1, we have defined the *syndrome adjacency graph* $\mathcal{G}$ where two qubits are connected if and only if they are in the support of the same stabilizer generator. As shown in the proof of Corollary 5.10 below, for all set $K \subseteq U$, the hypothesis $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$ of Lemma 5.9 is equivalent to $K \cap \Gamma_{\mathcal{G}}(U \setminus K) = \varnothing$ where $\Gamma_{\mathcal{G}}$ is the neighborhood in $\mathcal{G}$. Consequently, if K is a connected component of U in $\mathcal{G}$ then the hypotheses of Lemma 5.9 are satisfied for K. To summarize, each connected component of U in the syndrome adjacency graph is corrected by the small-set-flip algorithm independently from the other connected components, see Figure 5.1 for a schematic representation.

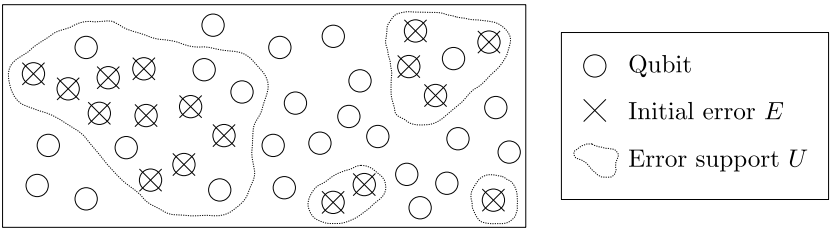

Figure 5.1: schematic representation of the error support in the syndrome adjacency graph. The small-set-flip decoder corrects the connected components of U independently.

In Corollary 5.10 below, we restate the locality property using the graph $\mathcal{G}$ instead of the Tanner graph.

Corollary 5.10. *We use the notations of Section 5.1.1.*
For any set $K \subseteq U$ with $K \cap \Gamma_{\mathcal{G}}(U \setminus K) = \varnothing$, there is a valid execution of Algorithm 3 on the input $\big(E \cap K, D \cap \Gamma_X(K)\big)$ whose output is $\hat{E} \cap K$ and whose support is $U \cap K$.

Proof. By Lemma 5.9, it is sufficient to prove that the statement $K \cap \Gamma_{\mathcal{G}}(U \setminus K) = \varnothing$ implies $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$. We show it by contraposition.
If $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) \neq \varnothing$ then there is a generator $g \in C_X \uplus C_Z$ such that $g \in \Gamma_{\mathcal{Q}}(K)$ and $g \in \Gamma_{\mathcal{Q}}(U \setminus K)$. Thus there are two qubits $q_1 \in K$ and $q_2 \in U \setminus K$ belonging to the support of g. Let q_3 be another qubit in the support of g with $q_3 \neq q_1$ and $q_3 \neq q_2$, then by the definition of the syndrome adjacency graph we have $q_3 \in \Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K)$. □

Using the execution support $U = E \cup F_0 \cup \ldots \cup F_{f-1}$, the error decomposition required in eq. (5.5) is done in the following way: each connected component K of U in $\mathcal{G}$ provides the error sets $E_K := K \cap E$ and $D_K := D \cap \Gamma_X(K)$. By Corollary 5.10, running Algorithm 3 on the input (E_K, D_K) leads to the residual error $E'_K = (E \oplus \hat{E}) \cap K$ and we will prove that the execution support K is small enough to apply Proposition 5.2. The key point is to remark that a fraction $2\alpha_1$ of the vertices in $X := K \cup D_K \subseteq \mathcal{V}$ belong to $E \cup D$ (see Lemma 5.3). We will say that X is a $2\alpha_1$-subset of $E \cup D$ (see Definition 5.11) and percolation arguments (see Lemma 5.18) will show that with high probability, any connected α-subset of a random error $E \cup D$ must be small enough to apply Proposition 5.2.

Definition 5.11 (α-subset and MaxConn$_\alpha$)**.** We use the notations of Section 5.1.1.
Let $\alpha \in (0; 1]$ and let $X, Y \subseteq \mathcal{V}$. X is said to be an α-subset of Y if $\big|X \cap Y\big| \geq \alpha \big|X\big|$.
We also define the integer $\mathrm{MaxConn}_\alpha(Y)$ by:

$$\mathrm{MaxConn}_\alpha(Y) = \max\Big\{ \big|X\big| : X \text{ is connected in } \mathcal{G} \text{ and is an } \alpha\text{-subset of } Y \Big\}.$$

As stated in Corollary 5.12 below, Lemma 5.3 implies that $U \cup D$ is a $2\alpha_1$-subset of $E \cup D$.

Corollary 5.12. *We use the notations of Section 5.1.1.*
The set $U \cup D$ is a $2\alpha_1$-subset of $E \cup D$.

Proof. By Lemma 5.3:

$$\big|U\big| + \big|D\big| \leq c_3\big|E\big| + (c_4 + 1)\big|D\big| = \frac{1}{2\alpha_1}\big|E \cup D\big| = \frac{1}{2\alpha_1}\big|(U \cup D) \cap (E \cup D)\big|.$$

thus $U \cup D$ is a $2\alpha_1$-subset of $E \cup D$. □

If we go back to Theorem 5.6, the error E_{ls} will be defined to be reduced and equivalent to $E \oplus \hat{E}$, and the local stochastic property for E_{ls} will be a consequence of the local stochastic property for D. More precisely, we already know for all $T \subseteq C_X$:

$$\mathbb{P}\Big[T \subseteq D\Big] \leq p_{\mathrm{synd}}^{|T|} \tag{5.6}$$

and we want to show for all $S \subseteq V_{\mathcal{Q}}$:

$$\mathbb{P}\Big[S \subseteq E_{\mathrm{ls}}\Big] \leq p_{\mathrm{ls}}^{|S|}.$$

Given a set $S \subseteq E_{\mathrm{ls}}$, we will find a large set $T \subseteq D$ to upper bound $\mathbb{P}\Big[S \subseteq E_{\mathrm{ls}}\Big]$ using eq. (5.6). Informally, the set T contains the faulty check-nodes which prevent the small-set-flip algorithm from correcting the qubits of S. Formally, $T = D \cap \Gamma_X(W)$ where $W \subseteq V_{\mathcal{Q}}$ is called a *witness* as defined in Definition 5.14. Let $S \subseteq E_{\mathrm{ls}}$, then a witness for S is a set $W \subseteq V_{\mathcal{Q}}$ satisfying $S \subseteq W$ with $\big|D \cap \Gamma_X(W)\big| = \Omega\big(\big|W\big|\big)$ and such that S intersects every connected component of W in $\mathcal{G}$. Finally, the main difficulty in the proof of Theorem 5.6 is to show the existence of a witness W for all $S \subseteq E_{\mathrm{ls}}$. Up to technical details, W will be the union of the connected components of the execution support which intersect S.

Definition 5.13 ([46])**.** We use the notations of Section 5.1.1.
For $S \subseteq V_{\mathcal{Q}}$, we define $\mathcal{M}(S)$ to be the set of all subsets $W \subseteq V_{\mathcal{Q}}$ with $S \subseteq W$ such that any connected component W' of W in $\mathcal{G}$ satisfies $W' \cap S \neq \varnothing$.

Lemma 2 of [46] provides an upper bound on the number of sets $W \in \mathcal{M}(S)$ of a given size:

$$\#\Big\{W \in \mathcal{M}(S) : \big|W\big| = t\Big\} \leq \frac{(ed)^t}{ed^{|S|}}. \tag{5.7}$$

Definition 5.14 (Witness)**.** We use the notations of Section 5.1.1.
$W \subseteq V_{\mathcal{Q}}$ is said to be a *witness for* $S \subseteq V_{\mathcal{Q}}$ if $W \in \mathcal{M}(S)$ and $\big|D \cap \Gamma_X(W)\big| \geq \big|W\big|/c_0$.

c) Formal proof

Besides Theorem 5.6, there are three main statements in this section:

- Lemma 5.15 shows how to find a witness for any $S \subseteq E \oplus \hat{E}$ under the assumptions that $E \oplus \hat{E}$ is reduced and $\big|E \oplus \hat{E}\big| \leq \gamma_0\sqrt{N}$.
- Lemma 5.17 shows how to find a witness for any $S \subseteq E \oplus \hat{E}$ under the assumption $\mathrm{MaxConn}_{\alpha_1}(E) \leq \gamma_0\sqrt{N}$.
- Lemma 5.18 shows that $\mathrm{MaxConn}_{\alpha_1}(E) \leq \gamma_0\sqrt{N}$ holds with high probability when E and D are local stochastic.

Finally, we will conclude the section with the proof of Theorem 5.6 where the existence of witnesses is used to establish the local stochastic property for the residual error.

Lemma 5.15. *We use the notations of Section 5.1.1.*
If the residual error $E \oplus \hat{E}$ is reduced and $\big|E \oplus \hat{E}\big| \leq \gamma_0\sqrt{N}$ then for all $S \subseteq E \oplus \hat{E}$, there is a witness W for S with the additional constraint $W \subseteq E \oplus \hat{E}$.

Proof. Let W be the union of all the connected components of $E \oplus \hat{E}$ in $\mathcal{G}$ that contain at least one element of S. The properties $W \in \mathcal{M}(S)$ and $W \subseteq E \oplus \hat{E}$ clearly hold and it remains to prove $\big|W\big| \leq c_0 \big|D \cap \Gamma_X(W)\big|$.
By locality (Corollary 5.10 applied with $K = W$), no flip is done by Algorithm 3 on the input $(W, D \cap \Gamma_X(W))$. Moreover, the error W is reduced as a subset of the reduced error $E \oplus \hat{E}$ (Lemma 4.15) and satisfies $\big|W\big| \leq \gamma_0\sqrt{N}$. As a summary, there is a valid execution of Algorithm 3 on the input $(W, D \cap \Gamma_X(W))$ with output $\varnothing$ leading to the residual error W. Proposition 5.2 applied for this execution provides:

$$\big|W\big| \leq c_0 \big|D \cap \Gamma_X(W)\big|.$$

□

Now, our goal is to prove Lemma 5.17 which is quite similar to Lemma 5.15 where the hypothesis $\big|E \oplus \hat{E}\big| \leq \gamma_0\sqrt{N}$ is replaced by $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$. There is a small technical difficulty for that: contrarily to the hypothesis of Lemma 5.15, the residual error is not reduced in general and thus we need to combine Lemma 5.15 with Lemma 5.16 below.

Lemma 5.16. *We use the notations of Section 5.1.1.*
Let $Y, X_1, X_2 \subseteq \mathcal{V}$ with $\big|X_2\big| \leq \big|X_1\big|$. If X_1 is an α-subset of Y then $X_1 \cup X_2$ is an $\frac{\alpha}{2}$-subset of Y.

Proof. By assumption $\big|X_1\big| \leq \frac{1}{\alpha}\big|X_1 \cap Y\big|$ thus:

$$\big|X_1 \cup X_2\big| \leq 2\big|X_1\big| \leq \frac{2}{\alpha}\big|X_1 \cap Y\big| \leq \frac{2}{\alpha}\big|(X_1 \cup X_2) \cap Y\big|.$$

□

Lemma 5.17. *We use the notations of Section 5.1.1.*
If $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ then there is a reduced error E_{ls} equivalent to the residual error $E \oplus \hat{E}$ such that there is a witness for all $S \subseteq E_{\mathrm{ls}}$.

Proof. We define E_{ls} as one of the minimal weight errors whose syndrome is the same as the syndrome of $E \oplus \hat{E}$. The first step is to show that E_{ls} and $E \oplus \hat{E}$ are equivalent. Let K be a connected component of $U \cup E_{\mathrm{ls}}$ in $\mathcal{G}$, let $E_K := E \cap K$ and let $D_K := D \cap \Gamma_X(K)$. By locality (Corollary 5.10), there is a valid execution of Algorithm 3 on the input (E_K, D_K) whose output is $\hat{E}_K := \hat{E} \cap K$ and whose support is $U_K := U \cap K$. Hence, the set $U_K \cup D_K$ is a $2\alpha_1$-subset of $E_K \cup D_K$ (Corollary 5.12) and the set $K \cup D_K$ is an α_1-subset of $E_K \cup D_K$ (Lemma 5.16). Finally, the set $K \cup D_K$ is an α_1-subset of $E \cup D$ and from the hypothesis $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$, we conclude that $\big|K\big| \leq \gamma_0\sqrt{N}$. In particular, $(E \oplus \hat{E}) \cap K$ and $E_{\mathrm{ls}} \cap K$ have the same syndrome and the weight of $(E \oplus \hat{E} \oplus E_{\mathrm{ls}}) \cap K$ is smaller than the minimal distance, thus $E_{\mathrm{ls}} \cap K$ is equivalent to $(E \oplus \hat{E}) \cap K$. Since this is true for all K then E_{ls} is equivalent to $E \oplus \hat{E}$.

Let $S \subseteq E_{\mathrm{ls}}$, let $E' = E_{\mathrm{ls}} \oplus \hat{E}$ and let $E'_K := E' \cap K$. Keeping the same notations than above, we will prove that for each K, there is a witness W_K for $S_K := S \cap K$

with the additional constraint $W_K \subseteq E_{\mathrm{ls}} \cap K$.
The errors E and E' have the same syndrome thus the behavior of Algorithm 3 is the same on input (E, D) and on input (E', D). Hence, there is a valid execution of Algorithm 3 on input (E', D) whose output is $\hat{E}$ and whose support is $E' \cup F_0 \cup \ldots \cup F_{f-1}$. By locality (Corollary 5.10), there is also a valid execution of Algorithm 3 on input (E'_K, D_K) whose output is $\hat{E}_K$ and whose support is $(E' \cup F_0 \cup \ldots \cup F_{f-1}) \cap K$. The residual error of the latter execution is $(E' \oplus \hat{E}) \cap K = E_{\mathrm{ls}} \cap K$ which is reduced and is of weight smaller than $\gamma_0\sqrt{N}$. By Lemma 5.15, there is a witness W_K for S_K with $W_K \subseteq E_{\mathrm{ls}} \cap K$.

Finally, $W = \biguplus_K W_K$ is a witness for S. □

The last ingredient before proving Theorem 5.6 is provided by Lemma 5.18 below: the condition $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ needed in Lemma 5.17 is verified with high probability for local stochastic errors.

Lemma 5.18 (α-percolation, [38])**.** *We use the notations of Section 5.1.1. Let $\alpha \in (0, 1]$ then there exists a threshold $p_{\mathrm{th}} = p_{\mathrm{th}}(\alpha) > 0$ such that for any $t \in \mathbb{N}^*$, for any $p_{\mathrm{phys}} < p_{\mathrm{th}}$ and for any $p_{\mathrm{synd}} < p_{\mathrm{th}}$ the following holds. If an error (E, D) is chosen according to a local stochastic noise with parameter $(p_{\mathrm{phys}}, p_{\mathrm{synd}})$ then:*

$$\mathbb{P}\Big[\mathit{MaxConn}_\alpha(E \cup D) \geq t\Big] \leq C|\mathcal{V}| \left(\frac{\max(p_{\mathrm{phys}}, p_{\mathrm{synd}})}{p_{\mathrm{th}}}\right)^{\alpha t},$$

where $C = C(p, \alpha)$ is independent of t.

Lemma 5.18 is a particular case of Theorem 5.24 where the graph $\mathcal{G}$ is the syndrome adjacency graph. The proof of Theorem 5.24 is the purpose of Section 5.1.5.
We conclude this section with the proof of Theorem 5.6.

Proof of Theorem 5.6. Let $p_{\mathrm{th}} > 0$ as defined in Lemma 5.18. The constant $p_0 > 0$ is defined to be a positive number such that:

$$p_0 \leq p_{\mathrm{th}}, \quad 2^{d_C c_0} p_0 < 1/2, \quad ed2^{d_C} p_0^{1/c_0} \leq 1/2, \quad 4 \cdot 2^{d_C} p_0^{1/(2c_0)} \leq 1. \tag{5.8}$$

Let $p_{\mathrm{phys}} < p_0$ and $p_{\mathrm{synd}} < p_0$. We assume that the pair (E, D) satisfies a local stochastic noise model with parameter $(p_{\mathrm{phys}}, p_{\mathrm{synd}})$ and we run Algorithm 3 on the input (E, D). We define the random error E_{ls} promised by Theorem 5.6 in the following way:

$$\begin{cases} \text{If } \mathrm{MaxConn}_{\alpha_1}(E \cup D) > \gamma_0\sqrt{N} \text{ then } E_{\mathrm{ls}} = \varnothing. \\ \text{If } \mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N} \text{ then } E_{\mathrm{ls}} \text{ is the error promised by Lemma 5.17.} \end{cases}$$

Lemma 5.17 ensures that if $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ then E_{ls} is equivalent to $E \oplus \hat{E}$ thus by Lemma 5.18:

$$\begin{aligned} \mathbb{P}\Big[E_{\mathrm{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] &\leq \mathbb{P}\Big[\mathrm{MaxConn}_{\alpha_1}(E \cup D) > \gamma_0\sqrt{N}\Big] \\ &\leq C|\mathcal{V}| \left(\frac{\max(p_{\mathrm{phys}}, p_{\mathrm{synd}})}{p_{\mathrm{th}}}\right)^{\alpha_1\gamma_0\sqrt{N}} \end{aligned}$$

We have $\left|\mathcal{V}\right| = \Theta(N)$ and $\max(p_{\text{phys}}, p_{\text{synd}}) < p_{\text{th}}$ hence as stated in Theorem 5.6:

$$\mathbb{P}\Big[E_{\text{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] = e^{-\Theta(\sqrt{N})}.$$

It remains to show that E_{ls} is local stochastic with parameter $p_{\text{ls}} = p_{\text{synd}}^{1/(2c_0)}$. For any $S \subseteq V_{\mathcal{Q}}$, if $S \subseteq E_{\text{ls}}$ then $E_{\text{ls}} \neq \varnothing$ and thus $\text{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$. In addition by Lemma 5.17, if $\text{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ then there is a witness W for any S. Hence we have:

$$\begin{aligned}\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] &= \mathbb{P}\Big[S \subseteq E_{\text{ls}} \text{ and } \text{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}\Big] \\ &\leq \mathbb{P}\Big[\text{There exists a witness } W \text{ for } S\Big].\end{aligned}$$

To conclude, we upper bound the probability that a witness exists.
For a fixed $W \subseteq V_{\mathcal{Q}}$, if W is a witness for S then there is a set $T \subseteq \Gamma_X(W)$ satisfying $\left|W\right| \leq c_0\left|T\right|$ and $T \subseteq D$. Thus the probability that W is a witness for S is upper bounded by:

$$\mathbb{P}\Big[W \text{ is a witness for } S\Big] \leq \sum_{\substack{T \subseteq \Gamma_X(W):\\ |W| \leq c_0|T|}} \mathbb{P}\Big[T \subseteq D\Big] \leq \sum_{\substack{T \subseteq \Gamma_X(W):\\ |W| \leq c_0|T|}} p_{\text{synd}}^{|T|}.$$

Since the cardinality of $\Gamma_X(W)$ is upper bounded by $d_C|W|$, we have:

$$\begin{aligned}\mathbb{P}\Big[W \text{ is a witness for } S\Big] &\leq \sum_{t \geq |W|/c_0} 2^{d_C|W|} p_{\text{synd}}^t \\ &\leq \sum_{t \geq |W|/c_0} \left(2^{d_C c_0} p_{\text{synd}}\right)^t.\end{aligned}$$

Equation (5.8) ensures $2^{d_C c_0} p_{\text{synd}} < 1/2$ thus:

$$\mathbb{P}\Big[W \text{ is a witness for } S\Big] \leq 2\left(2^{d_C} p_{\text{synd}}^{1/c_0}\right)^{|W|}.$$

Combining the previous inequalities we get:

$$\begin{aligned}\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] &\leq \mathbb{P}\Big[\text{There exists a witness } W \text{ for } S\Big] \\ &\leq \sum_{\substack{W \in \mathcal{M}(S)\\ |W| \geq |S|}} \mathbb{P}\Big[W \text{ is a witness for } S\Big] \\ &\leq 2 \sum_{\substack{W \in \mathcal{M}(S)\\ |W| \geq |S|}} \left(2^{d_C} p_{\text{synd}}^{1/c_0}\right)^{|W|} \\ &\leq \frac{2}{ed^{|S|}} \sum_{w \geq |S|} \left(ed2^{d_C} p_{\text{synd}}^{1/c_0}\right)^w \qquad \text{by eq. (5.7).}\end{aligned}$$

Equation (5.8) ensures $ed2^{d_C} p_{\text{synd}}^{1/c_0} \leq 1/2$ and $\frac{4}{e} e 2^{d_C} p_{\text{synd}}^{1/(2c_0)} \leq 1$ thus:

$$\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] \leq \frac{4}{e}\left(e2^{d_C} p_{\text{synd}}^{1/c_0}\right)^{|S|} \leq \left(p_{\text{synd}}^{1/(2c_0)}\right)^{|S|}.$$

□

5.1.4 Locality of the small-set-flip algorithm

The goal of this section is to prove the locality property of Lemma 5.9.
We start with Lemma 5.19 below showing that if $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$ then the syndrome of an error contained in K is independent of the qubits outside K. The variable A of this lemma will be set later to be either $A = E$ or $A = F_i$.

Lemma 5.19. *We use the notations of Section 5.1.1.*
Let $K \subseteq V_{\mathcal{Q}}$ be such that $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$. Then for all $A \subseteq U$:

$$\sigma_X(A \cap K) = \sigma_X(A) \cap \Gamma_X(K).$$

Proof. Let $\overline{K} := V_{\mathcal{Q}} \setminus K$ be the complement of K in $V_{\mathcal{Q}}$. We decompose $\sigma_X(A) \cap \Gamma_X(K)$ using the partition $A = (A \cap K) \uplus (A \cap \overline{K})$:

$$\begin{aligned}\sigma_X(A) \cap \Gamma_X(K) &= \Big(\sigma_X(A \cap K) \oplus \sigma_X(A \cap \overline{K})\Big) \cap \Gamma_X(K) \\ &= \Big(\sigma_X(A \cap K) \cap \Gamma_X(K)\Big) \oplus \Big(\sigma_X(A \cap \overline{K}) \cap \Gamma_X(K)\Big).\end{aligned}$$

On the one hand, we have $\sigma_X(A \cap K) \subseteq \Gamma_X(K)$ thus:

$$\sigma_X(A \cap K) \cap \Gamma_X(K) = \sigma_X(A \cap K).$$

On the other hand, we have $\sigma_X(A \cap \overline{K}) \subseteq \Gamma_X(A \cap \overline{K}) \subseteq \Gamma_{\mathcal{Q}}(A \cap \overline{K})$ and $\Gamma_X(K) \subseteq \Gamma_{\mathcal{Q}}(K)$ thus:

$$\begin{aligned}\sigma_X(A \cap \overline{K}) \cap \Gamma_X(K) &\subseteq \Gamma_{\mathcal{Q}}(A \cap \overline{K}) \cap \Gamma_{\mathcal{Q}}(K) \\ &\subseteq \Gamma_{\mathcal{Q}}(U \cap \overline{K}) \cap \Gamma_{\mathcal{Q}}(K) \qquad \text{because } A \subseteq U, \\ &= \Gamma_{\mathcal{Q}}(U \setminus K) \cap \Gamma_{\mathcal{Q}}(K) \\ &= \varnothing.\end{aligned}$$

Combining the three equalities, we get Lemma 5.19. □

In Lemma 5.20 below, we compare the way flipping a small-set $F \in \mathcal{F}$ affects a given syndrome $\widetilde{\sigma}$ and the way it affects the same syndrome restricted to the check-nodes which touch K.

Lemma 5.20. *We use the notations of Section 5.1.1.*
Let $\widetilde{\sigma} \subseteq C_X$ and let $K \subseteq V_{\mathcal{Q}}$ then for all $F \in \mathcal{F}$:

(i) $\Delta(\widetilde{\sigma} \cap \Gamma_X(K), F) \leq \Delta(\widetilde{\sigma}, F)$,
(ii) If $F \subseteq K$ then $\Delta(\widetilde{\sigma} \cap \Gamma_X(K), F) = \Delta(\widetilde{\sigma}, F)$.

Proof. We rewrite $\Delta(\widetilde{\sigma}, F)$ in the following way:

$$\begin{aligned}\Delta(\widetilde{\sigma}, F) &= \big|\widetilde{\sigma}\big| - \big|\widetilde{\sigma} \oplus \sigma_X(F)\big| \\ &= \big|\widetilde{\sigma}\big| - \Big(\big|\widetilde{\sigma}\big| + \big|\sigma_X(F)\big| - 2\big|\widetilde{\sigma} \cap \sigma_X(F)\big|\Big) \\ &= 2\big|\widetilde{\sigma} \cap \sigma_X(F)\big| - \big|\sigma_X(F)\big|.\end{aligned}$$

The proof of item (i) is straightforward using the latter equality:

$$\begin{aligned}\Delta(\widetilde{\sigma} \cap \Gamma_X(K), F) &= 2\big|\widetilde{\sigma} \cap \Gamma_X(K) \cap \sigma_X(F)\big| - \big|\sigma_X(F)\big| \\ &\leq 2\big|\widetilde{\sigma} \cap \sigma_X(F)\big| - \big|\sigma_X(F)\big| \\ &= \Delta(\widetilde{\sigma}, F).\end{aligned}$$

Under the assumption $F \subseteq K$ of item (ii), we have $\Gamma_X(K) \cap \sigma_X(F) = \sigma_X(F)$ hence the inequality above is an equality and thus item (ii) holds. □

We are now ready to prove Lemma 5.9.

Proof of Lemma 5.9. Let:

$$E' := E \cap K, \qquad D' := D \cap \Gamma_X(K), \qquad \hat{E}' := \hat{E} \cap K.$$

Lemma 5.9 asserts that there is a valid execution of Algorithm 3 with input (E', D'), output $\hat{E}'$ and support K. Hence we would like to find some small-sets $F'_0, \dots, F'_{f'-1} \in \mathcal{F}$ such that if we define

$$\begin{cases} \widetilde{\sigma}'_0 := \sigma_X\big(E'\big) \oplus D', \\ \widetilde{\sigma}'_{j+1} := \widetilde{\sigma}'_j \oplus \sigma_X\big(F'_j\big) \qquad \text{for } j \in [\![0; f'-1]\!]. \end{cases} \tag{5.9}$$

then we have the following four properties:

(i) $\hat{E}' = F'_0 \oplus \ldots \oplus F'_{f'-1}$,
(ii) $K = E' \cup F'_0 \cup \ldots \cup F'_{f'-1}$,
(iii) $\forall j \in [\![0; f'-1]\!] : \Delta(\widetilde{\sigma}'_j, F'_j) \geq \beta_1 \big|\sigma_X(F'_j)\big|$,
(iv) $\forall F \in \mathcal{F} : \Delta(\widetilde{\sigma}'_{f'}, F) < \beta_1 \big|\sigma_X(F)\big|$.

For any $i \in [\![0; f]\!]$, the set F_i is a subset of the support of an X-type generator $g \in C_Z$. If we assume by contradiction that $F_i \nsubseteq K$ and $F_i \cap K \neq \varnothing$ then we have $g \in \Gamma_{\mathcal{Q}}(F_i \setminus K) \subseteq \Gamma_{\mathcal{Q}}(U \setminus K)$ and $g \in \Gamma_{\mathcal{Q}}(F_i \cap K)$. This contradicts the hypothesis $\Gamma_{\mathcal{Q}}(K) \cap \Gamma_{\mathcal{Q}}(U \setminus K) = \varnothing$ and thus we must have either $F_i \subseteq K$ or $F_i \cap K = \varnothing$. Let $i_0 < \ldots < i_{f'-1} \in [\![0; f-1]\!]$ be the steps of the algorithm such that $F_{i_k} \subseteq K$ and let us prove that the sets $F'_0 = F_{i_0}, \dots, F'_{f'} = F_{i_{f'-1}}$ are appropriate. By definition we have:

$$\begin{aligned} F_i \cap K &= F_i \qquad &&\text{if } i \in \{i_0, \dots, i_{f'-1}\}, \\ F_i \cap K &= \varnothing \qquad &&\text{if } i \notin \{i_0, \dots, i_{f'-1}\}. \end{aligned} \tag{5.10}$$

Using the previous statements we can easily prove item (i):

$$\hat{E}' = \hat{E} \cap K = \left(\bigoplus_{i=0}^{f-1} F_i\right) \cap K = \bigoplus_{i=0}^{f-1} \Big(F_i \cap K\Big) = \bigoplus_{j=0}^{f'-1} F'_j.$$

Item (ii) is proved in the same manner:

$$K = U \cap K = \left(E \cup \bigcup_{i=0}^{f-1} F_i\right) \cap K = \Big(E \cap K\Big) \cup \bigcup_{i=0}^{f-1} \Big(F_i \cap K\Big) = E' \cup \bigcup_{j=0}^{f'-1} F'_j.$$

Before proving items (iii) and (iv), we need to show that the sets $\widetilde{\sigma}'_j$ defined by eq. (5.9) satisfy eq. (5.11) below:

$$\begin{aligned} &\widetilde{\sigma}'_j = \widetilde{\sigma}_{i_j} \cap \Gamma_X(K) \qquad \text{for } j \in [\![0; f'-1]\!], \\ &\widetilde{\sigma}'_{f'} = \widetilde{\sigma}_f \cap \Gamma_X(K). \end{aligned} \tag{5.11}$$

For $i \in [\![0; f]\!]$, let $\widetilde{\sigma}''_i := \widetilde{\sigma}_i \cap \Gamma_X(K)$ then we have:

$$\begin{aligned} \widetilde{\sigma}''_0 &= \widetilde{\sigma}_0 \cap \Gamma_X(K) \\ &= \Big(\sigma_X(E) \oplus D\Big) \cap \Gamma_X(K) \\ &= \Big(\sigma_X(E) \cap \Gamma_X(K)\Big) \oplus D' && \text{by distributivity,} \\ &= \sigma_X(E') \oplus D' && \text{by Lemma 5.19 applied with } W = E. \end{aligned}$$

Similarly, for $i \in [\![0; f-1]\!]$:

$$\begin{aligned} \widetilde{\sigma}''_{i+1} &= \widetilde{\sigma}_{i+1} \cap \Gamma_X(K) \\ &= \Big(\widetilde{\sigma}_i \oplus \sigma_X(F_i)\Big) \cap \Gamma_X(K) \\ &= \widetilde{\sigma}''_i \oplus \Big(\sigma_X(F_i) \cap \Gamma_X(K)\Big) && \text{by distributivity,} \\ &= \widetilde{\sigma}''_i \oplus \sigma_X(F_i \cap K) && \text{by Lemma 5.19 applied with } W = F_i. \end{aligned}$$

Using eq. (5.10), the sets $\widetilde{\sigma}''_i$ satisfy:

$$\begin{cases} \widetilde{\sigma}''_0 = \sigma_X(E') \oplus D', & \\ \widetilde{\sigma}''_{i+1} = \widetilde{\sigma}''_i \oplus \sigma_X(F_i) & \text{if } i \in \{i_0, \ldots, i_{f'-1}\}, \\ \widetilde{\sigma}''_{i+1} = \widetilde{\sigma}''_i & \text{if } i \notin \{i_0, \ldots, i_{f'-1}\}. \end{cases} \tag{5.12}$$

When we compare eq. (5.9) and eq. (5.12), an induction yields:

$$\begin{aligned} &\sigma'_0 = \widetilde{\sigma}''_0 = \ldots = \widetilde{\sigma}''_{i_0}, \\ &\sigma'_{j+1} = \widetilde{\sigma}''_{i_j+1} = \ldots = \widetilde{\sigma}''_{i_{j+1}} \qquad \text{for } j \in [\![0; f'-2]\!], \\ &\sigma'_{f'} = \widetilde{\sigma}''_{i_{f'-1}+1} = \ldots = \widetilde{\sigma}''_f. \end{aligned}$$

Hence eq. (5.11) hold.

We are now ready to prove items (iii) and (iv). By hypothesis, we have a valid execution of Algorithm 3 with input (E, D), output $\hat{E}$ and support U, thus:

$$\forall i \in [\![0; f-1]\!] : \Delta(\widetilde{\sigma}_i, F_i) \geq \beta_1 \big|\sigma_X(F_i)\big|, \tag{5.13}$$

$$\forall F \in \mathcal{F} : \Delta(\widetilde{\sigma}_f, F) < \beta_1 \big|\sigma_X(F)\big|. \tag{5.14}$$

Hence item (iii) holds because for all $j \in [\![0; f'-1]\!]$:

$$\begin{aligned}
\Delta(\widetilde{\sigma}'_j, F'_j) &= \Delta(\widetilde{\sigma}_{i_j} \cap \Gamma_X(K), F'_j) && \text{by eq. (5.11),} \\
&= \Delta(\widetilde{\sigma}_{i_j}, F'_j) && \text{by Lemma 5.20 (ii),} \\
&\geq \beta_1 \big|\sigma_X(F'_j)\big| && \text{by eq. (5.13).}
\end{aligned}$$

For item (iv), let $F \in \mathcal{F}$ then:

$$\begin{aligned}
\Delta(\widetilde{\sigma}'_{f'}, F) &= \Delta(\widetilde{\sigma}_f \cap \Gamma_X(K), F) && \text{by eq. (5.11),} \\
&\leq \Delta(\widetilde{\sigma}_f, F) && \text{by Lemma 5.20 (i),} \\
&< \beta_1 \big|\sigma_X(F)\big| && \text{by eq. (5.14).}
\end{aligned}$$

□

5.1.5 Percolation

The aim of this section is to prove Theorem 5.24 where a percolation process is performed on a graph $\mathcal{G}$ with degree upper bounded by $d_{\mathcal{G}} \geq 3$. In the particular case where $\mathcal{G}$ is the syndrome adjacency graph, Theorem 5.24 turns into Lemma 5.18 that we have used in the main text. Let $\mathcal{V}$ be the set of vertices of $\mathcal{G}$. Then a percolation process picks a random set $Y \subseteq \mathcal{V}$ and the question we are interested in is to compute the probability that Y contains a large connected α-subset in the sense of Definition 5.11. We assume that Y follows a local stochastic model, see Definition 5.5 that we rewrite in Definition 5.21 below.

Definition 5.21 (Local stochastic noise model)**.** Let $\mathcal{V}$ be a set of nodes and $p \in [0; 1]$. A random variable $Y \subseteq \mathcal{V}$ follows a local stochastic model with parameter p if and only if for all $S \subseteq \mathcal{V}$:

$$\mathbb{P}\Big[S \subseteq Y\Big] \leq p^{|S|}.$$

In this section we use the following upper bound holding for all integers $n, k \in \mathbb{N}$:

$$\binom{n}{k} \leq 2^{nh(k/n)}, \tag{5.15}$$

where $h(x) := -x \log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function.

The first part of this section is dedicated to derive an upper bound on the number of connected sets of a degree bounded graph as stated in Lemma 5.22 and Corollary 5.23. The proof of Lemma 5.22 uses the Raney numbers [84] and follows the proof of [109].

Lemma 5.22 ([84, 109]). *Let $\mathcal{G}$ be a graph with degree upper bounded by $d_{\mathcal{G}} \geq 3$ and let $\mathcal{V}$ be the set of vertices.*
For any integer $s \geq 1$, the number of connected sets of $\mathcal{G}$ with size s satisfies:

$$\left|\mathcal{C}_s(\mathcal{G})\right| \leq |\mathcal{V}| \frac{d_{\mathcal{G}}}{s\Big(s(d_{\mathcal{G}}-2)+2\Big)} \binom{s(d_{\mathcal{G}}-1)}{s-1}.$$

Proof. The main tool for this proof is the set $\mathcal{T}(s)$ of *labeled rooted trees* with maximum degree $d_{\mathcal{G}}$ and size s. An element of $\mathcal{T}(s)$ is a directed tree (a directed graph without cycle) with s vertices and a particular vertex called the *root* such that the edges are directed from the root to the leaves of the tree. Note that the in-degree is 0 for the root and 1 for the other vertices and, in addition, we require the out-degree to be at most $d_{\mathcal{G}}$ for the root and at most $d_{\mathcal{G}}-1$ for the other vertices. Finally, the trees are labeled meaning that for each vertex v, the directed edges whose tail is v are injectively labeled with labels in $[\![1; d_{\mathcal{G}}]\!]$ if v is the root and with labels in $[\![1; d_{\mathcal{G}}-1]\!]$ if v is not the root. By [84], we have:

$$\left|\mathcal{T}(s)\right| = R(d_{\mathcal{G}}-1, s-1, d_{\mathcal{G}}) := \frac{d_{\mathcal{G}}}{s(d_{\mathcal{G}}-2)+2} \binom{s(d_{\mathcal{G}}-1)}{s-1}.$$

where $R(a,b,c)$ are the Raney numbers.

From the graph $\mathcal{G}$, we construct the oriented graph $\mathcal{G}_0$ where each non-oriented edge has been replaced by two opposite oriented edges. Similarly than for the trees, we fix some labeling of $\mathcal{G}_0$: the directed edges whose tail is some node v are injectively labeled with labels in $[\![1; d_{\mathcal{G}}]\!]$.

Let $v \in \mathcal{V}$ and let $\mathcal{C}_s(v)$ be the set of connected sets $X \in \mathcal{C}_s(\mathcal{G})$ such that $v \in X$. Let $X \in \mathcal{C}_s(v)$ then there is at least one spanning tree T_0 of X in $\mathcal{G}_0$. From T_0, we get a labeled rooted tree $T \in \mathcal{T}(s)$ by fixing v as the root and using the labels in $\mathcal{G}_0$. We need to make a comment at this point: the labeling for T is not exactly the same as the one for T_0 as we explain below. The difficulty is that the labels in T are in $[\![1; d_{\mathcal{G}}-1]\!]$ (for edges whose tail is not the root) whereas the label $d_{\mathcal{G}}$ is allowed in T_0. Let v_1 be a vertex in T_0 which is not the root, let v_0 be the father of v_1 and let i be the label in T_0 of the oriented edge from v_1 to v_0. If j is the label in T_0 of another edge whose tail is v_1 then this edge in T has label j if $j < i$ and $j-1$ if $j > i$.

Once a vertex $v \in \mathcal{V}$ has been fixed, the above procedure builds at least one labeled rooted tree from a given $X \in \mathcal{C}_s(v)$ and conversely, any $T \in \mathcal{T}(s)$ can be obtained from at most one connected set $X \in \mathcal{C}_s(v)$. Hence $\left|\mathcal{C}_s(v)\right| \leq \left|\mathcal{T}(s)\right|$. In the sum $\sum_{v\in\mathcal{V}} \left|\mathcal{C}_s(v)\right|$, each connected set of $\mathcal{G}$ is counted s times. Therefore:

$$\left|\mathcal{C}_s(\mathcal{G})\right| = \frac{1}{s}\sum_{v\in\mathcal{V}} \left|\mathcal{C}_s(v)\right| \leq |\mathcal{V}| \frac{d_{\mathcal{G}}}{s\Big(s(d_{\mathcal{G}}-2)+2\Big)} \binom{s(d_{\mathcal{G}}-1)}{s-1}.$$

□

Corollary 5.23. *Let $\mathcal{G}$ be a graph with degree upper bounded by $d_{\mathcal{G}} \geq 3$ and let $\mathcal{V}$ be the set of vertices of $\mathcal{G}$.*
For any integer $s \geq 1$, the number of connected sets of $\mathcal{G}$ with size s satisfies:

$$\big|\mathcal{C}_s(\mathcal{G})\big| \leq \big|\mathcal{V}\big| K^s$$

where $K = K(d_{\mathcal{G}}) := (d_{\mathcal{G}} - 1)\left(1 + \dfrac{1}{d_{\mathcal{G}} - 2}\right)^{d_{\mathcal{G}}-2}$.

Proof. From Lemma 5.22, we have:

$$\begin{aligned}
\big|\mathcal{C}_s(\mathcal{G})\big| &\leq \big|\mathcal{V}\big| \frac{d_{\mathcal{G}}}{s\Big(s(d_{\mathcal{G}}-2)+2\Big)} \binom{s(d_{\mathcal{G}}-1)}{s-1} \\
&= \big|\mathcal{V}\big| \frac{d_{\mathcal{G}}}{\Big(s(d_{\mathcal{G}}-2)+1\Big)\Big(s(d_{\mathcal{G}}-2)+2\Big)} \binom{s(d_{\mathcal{G}}-1)}{s} \\
&\leq \big|\mathcal{V}\big| \binom{s(d_{\mathcal{G}}-1)}{s} && \text{since } s \geq 1 \text{ and } d_{\mathcal{G}} \geq 2, \\
&\leq \big|\mathcal{V}\big| 2^{s(d_{\mathcal{G}}-1)h(1/(d_{\mathcal{G}}-1))} && \text{by eq. (5.15)}, \\
&= \big|\mathcal{V}\big| \left((d_{\mathcal{G}}-1)\left(1+\frac{1}{d_{\mathcal{G}}-2}\right)^{d_{\mathcal{G}}-2}\right)^s .
\end{aligned}$$

□

Theorem 5.24 (α-percolation)**.** *Let $\mathcal{G}$ be a graph with degree upper bounded by $d_{\mathcal{G}} \geq 3$ and let $\mathcal{V}$ be the set of vertices of $\mathcal{G}$. Let $\alpha \in (0, 1]$, let $t \geq 1$ be an integer and let p_{th} be defined by:*

$$p_{\text{th}} = \left(\frac{2^{-h(\alpha)}}{\Big(d_{\mathcal{G}}-1\Big)\Big(1+\frac{1}{d_{\mathcal{G}}-2}\Big)^{d_{\mathcal{G}}-2}}\right)^{\frac{1}{\alpha}},$$

where $h(\alpha) = -\alpha \log_2 \alpha - (1-\alpha)\log_2(1-\alpha)$ is the binary entropy function. Then, for any random variable $Y \subseteq \mathcal{V}$ satisfying the local stochastic noise property of Definition 5.21 with parameter $p < p_{\text{th}}$, we have

$$\mathbb{P}\Big[\mathit{MaxConn}_\alpha(Y) \geq t\Big] \leq C\big|\mathcal{V}\big| \left(\frac{p}{p_{\text{th}}}\right)^{\alpha t}$$

where $1/C = (1 - 2^{h(\alpha)/\alpha} p)\left(1 - \left(\frac{p}{p_{\text{th}}}\right)^{\alpha}\right)$.

Proof. Let $\mathcal{C}_s(\mathcal{G})$ be the set of connected sets of vertices of size s in $\mathcal{G}$. Applying a union bound, we obtain

$$\begin{aligned}
\mathbb{P}\Big[\mathrm{MaxConn}_\alpha(Y) \geq t\Big] &= \mathbb{P}\Big[\exists s \geq t, \exists X \in \mathcal{C}_s(\mathcal{G}) : \big|X \cap Y\big| \geq \alpha\big|X\big|\Big] \\
&\leq \sum_{s \geq t} \sum_{X \in \mathcal{C}_s(\mathcal{G})} \mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big].
\end{aligned}$$

Let us first consider the quantity $\mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big]$ for a set $X \in \mathcal{C}_s(\mathcal{G})$:

$$\begin{aligned}
\mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big] &\leq \sum_{m \geq \alpha s} \sum_{\substack{X' \subseteq X: \\ \big|X'\big| = m}} \mathbb{P}\Big[X \cap Y = X'\Big] \\
&\leq \sum_{m \geq \alpha s} \sum_{\substack{X' \subseteq X: \\ \big|X'\big| = m}} \mathbb{P}\Big[X' \subseteq Y\Big] \\
&\leq \sum_{m \geq \alpha s} \sum_{\substack{X' \subseteq X: \\ \big|X'\big| = m}} p^m \\
&\leq \sum_{m \geq \alpha s} \binom{s}{m} p^m \,.
\end{aligned}$$

Using eq. (5.15):

$$\mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big] \leq \sum_{m \geq \alpha s} \left(2^{\frac{s}{m} h\left(\frac{m}{s}\right)} p\right)^m .$$

Since $(x \mapsto h(x)/x)$ is non increasing on $[\alpha, 1]$:

$$\begin{aligned}
\mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big] &\leq \sum_{m \geq \alpha s} \left(2^{h(\alpha)/\alpha} p\right)^m \\
&\leq \frac{(2^{h(\alpha)/\alpha} p)^{\alpha s}}{1 - 2^{h(\alpha)/\alpha} p}.
\end{aligned}$$

Let $K := \Big(d_{\mathcal{G}} - 2\Big)\Big(1 + \frac{1}{d_{\mathcal{G}} - 2}\Big)^{d_{\mathcal{G}} - 2}$ as defined in Corollary 5.23 then:

$$\begin{aligned}
\mathbb{P}\Big[\mathrm{MaxConn}_\alpha(Y) \geq t\Big] &\leq \sum_{s \geq t} \sum_{X \in \mathcal{C}_s(\mathcal{G})} \mathbb{P}\Big[\big|X \cap Y\big| \geq \alpha\big|X\big|\Big] \\
&\leq \frac{1}{1 - 2^{h(\alpha)/\alpha} p} \sum_{s \geq t} \big|\mathcal{C}_s(\mathcal{G})\big| 2^{h(\alpha) s} p^{\alpha s} \quad \text{by the union bound,} \\
&\leq \frac{|\mathcal{V}|}{1 - 2^{h(\alpha)/\alpha} p} \sum_{s \geq t} \Big(K 2^{h(\alpha)} p^\alpha\Big)^s \quad \text{by Corollary 5.23,} \\
&\leq \frac{|\mathcal{V}|}{1 - 2^{h(\alpha)/\alpha} p} \frac{(K 2^{h(\alpha)} p^\alpha)^t}{1 - K 2^{h(\alpha)} p^\alpha} \,.
\end{aligned}$$

Observing that $p_{\mathrm{th}} = \big(\frac{1}{K 2^{h(\alpha)}}\big)^{1/\alpha}$, we obtain the desired result. □

5.2 Parallel decoding

In this chapter, we introduce a parallel implementation for the small-set-flip algorithm where many small-sets are flipped in each round (Algorithm 4). Unlike the sequential decoder presented in Section 5.1, Algorithm 4 stops after a fixed number of rounds $f \in \mathbb{N}$ even though there still exists some small-sets that decrease the final syndrome weight when flipped.
In Section 5.2.4, f is set to be a constant integer independent of the initial syndrome weight and hence the decoder runs in constant time. This is particularly relevant in the context of fault-tolerant quantum computation where we do not want to give time for the errors to accumulate. The drawback is that the residual error satisfies the local stochastic condition with a constant parameter independent of the initial local stochastic parameters. In particular when the syndrome measurement is perfect, the errors on the qubits are not entirely corrected.
In Section 5.2.5, the integer f is set to be logarithmic in the initial syndrome weight. In that case, the residual error has local stochastic parameter $p_{\mathrm{ls}} := p_{\mathrm{synd}}^c$ where c is a constant and p_{synd} is the local stochastic parameter of the syndrome error. In particular, for perfect syndrome measurements where $p_{\mathrm{synd}} = 0$, we have $p_{\mathrm{ls}} = 0$ and thus there is no residual error on the qubits after correction.

5.2.1 Notations

In Section 5.2, we keep the notations of Section 5.1.1 running Algorithm 4 defined below instead of Algorithm 3. In particular, (E, D) is the input of Algorithm 4, $\hat{E}$ its output and $U = E \cup F_0 \cup \ldots \cup F_{f-1}$ its execution support. We also use the notation C_Z^k defined in Lemma 5.25 and define the additional positive constants:

$$\chi := \Big(d_C(d_V-1)+1\Big)\Big(d_V(d_C-1)+1\Big), \quad \eta := 1 - \frac{\beta_1 c_1}{d_V d_C \chi (d_V + d_C)} < 1,$$

$$c_5 := 1 - \eta + \frac{\beta_1 c_2}{d_V d_C \chi (d_V + d_C)}, \qquad c_6 := \frac{c_5}{1-\eta}.$$

For Section 5.2.4, we fix a parameter $c > 1$ and define:

$$f_0 = f_0(c) := \chi \left\lceil \frac{\log\left(\frac{8c}{\beta_0 d_V}\right)}{-\log(\eta)} \right\rceil, \qquad c_7 = c_7(c) := \frac{2}{\beta_0 d_V}\Big(1 + \eta^{f_0/\chi} + c_6\Big),$$

$$c_8 = c_8(c) := \Big(4cc_7c_3 + c_4\Big)^{-1}, \qquad c_9 = c_9(c) := \left(c_3 + \frac{c_4}{4cc_7}\right)^{-1}.$$

For Section 5.2.5 we define:

$$f_1 : s \mapsto \chi \left\lceil \frac{\log\left(\frac{2s}{\beta_0 d_V}\right)}{-\log(\eta)} \right\rceil + 1, \qquad c_0' := \left\lceil \frac{2\big(1+c_6\big)}{\beta_0 d_V} \right\rceil.$$

5.2.2 Definition of the decoder

The sequential small-set-flip algorithm (Algorithm 3) flips one small-set per step. It is possible to parallelize this procedure by flipping many small-sets belonging to the support of different X-type generators and such that their syndromes do not intersect. For that, we introduce in Lemma 5.25 a coloring of the X-type generators represented by the sets C_Z^k: if g_1 and g_2 have the same color then $\sigma_X(F_1) \cap \sigma_X(F_2) = \varnothing$ for any $F_1 \subseteq \Gamma_Z(g_1)$ and $F_2 \subseteq \Gamma_Z(g_2)$.

Lemma 5.25. *We use the notations of Section 5.1.1.*
There is a partition $C_Z = \biguplus_{k=1}^{\chi} C_Z^k$ for the X-type generators such that for all $k \in [\![1;\chi]\!]$ and all $g_1, g_2 \in C_Z^k$, we have:

$$\Gamma_X\Big(\Gamma_Z(g_1)\Big) \cap \Gamma_X\Big(\Gamma_Z(g_2)\Big) = \varnothing.$$

Proof. Let G_0 be a graph with vertex set C_Z where $g_1, g_2 \in C_Z$ are connected if and only if $\Gamma_X(\Gamma_Z(g_1)) \cap \Gamma_X(\Gamma_Z(g_2)) \neq \varnothing$. In other words the incidence relation Γ_0 is defined for all $g \in C_Z$ to be:

$$\Gamma_0(g) = \Gamma_Z(\Gamma_X(\Gamma_X(\Gamma_Z(g)))).$$

The sets $C_Z^1, \ldots, C_Z^\chi$ are defined as a coloring of the graph G_0. The degree of G_0 is $\chi - 1$ thus its chromatic number is upper bounded by χ. □

Lemma 5.25 leads to Algorithm 4 below which is a parallelized version of Algorithm 3. It is important to note a difference with Algorithm 3 though: instead of running until no flips reduce the syndrome weight, Algorithm 4 runs for a fixed number of steps f and may have some final steps that do not reduce the syndrome weight.

Algorithm 4 : the parallel small-set-flip decoder for noisy syndrome measurements with $f = f(|\widetilde{\sigma}|)$ steps.

Input: the observed syndrome $\widetilde{\sigma} \subseteq C_X$. // $\widetilde{\sigma} = \sigma(E) \oplus D$ for some $E \subseteq V_{\mathcal{Q}}, D \subseteq C_X$
Output: a guess for the error $\hat{E} \subseteq V_{\mathcal{Q}}$.

$\hat{E}_0 = \varnothing$; $\widetilde{\sigma}_0 = \widetilde{\sigma}$
for $i \in [\![0; f-1]\!]$ **do** // $f = f(|\widetilde{\sigma}|)$ is a parameter of the algorithm
  $k = i \mod \chi$ // Current color
  in parallel for $g \in C_Z^k$ **do**
    if $\mathcal{F}_g := \left\{ F \in \mathcal{F} : F \subseteq \Gamma_Z(g), \Delta(\widetilde{\sigma}_i, F) \geq \beta_1 \left|\sigma_X(F)\right| \right\} \neq \varnothing$ **then**
      Pick $F_g \in \mathcal{F}_g$ arbitrarily
    else
      $F_g = \varnothing$
  end if
 end parallel for
 $F_i = \bigoplus_{g \in C_Z^k} F_g$
 $\hat{E}_{i+1} = \hat{E}_i \oplus F_i$
 $\widetilde{\sigma}_{i+1} = \widetilde{\sigma}_i \oplus \sigma_X(F_i)$ // $\widetilde{\sigma}_{i+1} = \sigma_X(E \oplus \hat{E}_{i+1}) \oplus D$
end for
return $\hat{E}_i$

Remark 5.26. Assume we establish a list of all the small-sets that are flipped by Algorithm 4 on some input, then Algorithm 3 could flip these small-sets in the same order. As a consequence, if we define $U = E \cup F_0 \cup \ldots \cup F_{f-1}$ the execution support of Algorithm 4 then Corollary 5.12 implies that the set $U \cup D$ is a $2\alpha_1$-subset of $E \cup D$ and Lemma 5.3 provides:

$$\left|U\right| \leq c_3 \left|E\right| + c_4 \left|D\right|.$$

We will use Algorithm 4 in two cases: when the number of steps f is a fixed constant and when f grows logarithmically with the size of the input syndrome.

5.2.3 Analysis of the parallel decoder

In this section we show two useful properties we will use later in Section 5.2.4 and Section 5.2.5: Algorithm 4 is local (Lemma 5.27) and the weight of the observed syndrome decreases rapidly (Lemma 5.28 and Lemma 5.29).

Lemma 5.27 (Locality for Algorithm 4)**.** *We use the notations of Section 5.2.1. For any $K \subseteq U$ with $K \cap \Gamma_{\mathcal{G}}(U \setminus K) = \varnothing$, there is a valid execution of Algorithm 4 on the input $(E \cap K, D \cap \Gamma_X(K))$ whose output is $\hat{E} \cap K$ and whose support is $U \cap K$.*

Proof. The proof of Lemma 5.27 follows the same scheme than the proof of Lemma 5.9 presented in Section 5.1.4. □

Lemma 5.28. *We use the notations of Section 5.2.1. If we assume* $\left|U\right| \leq \gamma_0\sqrt{N}$ *then:*

$$\left|\widetilde{\sigma}_\chi\right| \leq \eta\left|\widetilde{\sigma}_0\right| + c_5\left|D\right|.$$

Proof. Our strategy is to find a step $i \in [\![0;\chi-1]\!]$ where the weight of the observed syndrome decreases a lot. Using Lemma 5.1, we are already able to build many small-sets (the elements of $\mathcal{F}^*$) which would decrease $\left|\widetilde{\sigma}_0\right|$, but we have to face two difficulties to prove that the syndrome decreasing stated in Lemma 5.28 holds.
The first difficulty is that two flips $F, F' \in \mathcal{F}^*$ can be subsets of the same generator $g \in C_Z$ and thus the decoder cannot flip both in one step. However, if for each generator $g \in C_Z$ we pick some $F_g^* \in \mathcal{F}^*$ then the number of F_g^* which are not empty is a fraction of $\left|\mathcal{F}^*\right|$. Thus, using the set $\{F_g^* : g \in C_Z\}$ instead of $\mathcal{F}^*$ will tackle the first difficulty.
Let $i \in [\![0;\chi-1]\!]$ be the step at which the small-set corresponding to some X-type generator g is flipped. The second difficulty is that even though flipping F_g^* decreases the weight of $\widetilde{\sigma}_0$, there is no guarantee that this flip decreases the weight of $\widetilde{\sigma}_i$ as well. Indeed, we could have $\Delta(\widetilde{\sigma}_i, F_g^*) \neq \Delta(\widetilde{\sigma}_0, F_g^*)$ due to the fact that we have flipped another small-set F in one of the prior steps. However, by the LDPC property, each flip F affects at most a constant number of F_g^* and as a consequence, either we flip many small-sets before flipping a given color $k \in [\![0;\chi-1]\!]$ or the majority of the flips with the color k are done.
Finally, if we identify $k \in [\![0;\chi-1]\!]$ the most represented color in $\mathcal{F}^*$ and $i \in [\![0;\chi-1]\!]$ the step where the decoder flips the generators with color k, then the weight of the observed syndrome will decrease a lot either before step i or at step i.

Here is the formal proof of the statement. We have $\left|E\right| \leq \left|U\right| \leq \gamma_0\sqrt{N}$ thus Lemma 5.1 provides a set $\mathcal{F}^* \subseteq \mathcal{F}$. For all X-type generator $g \in C_Z$, we define:

$$\mathcal{F}_g := \left\{F \in \mathcal{F} : F \subseteq \Gamma_Z(g), \Delta(\widetilde{\sigma}_0, F) \geq \beta_1\left|\sigma_X(F)\right|\right\}, \qquad \mathcal{F}_g^* := \mathcal{F}^* \cap \mathcal{F}_g.$$

In addition, we define a set $F_g^* \in \mathcal{F}_g^*$ in the following manner:

$$\begin{cases} \text{If } \mathcal{F}_g^* \neq \varnothing \text{ then } F_g^* := \underset{F \in \mathcal{F}_g^*}{\arg\max} \left|\sigma_X(F)\right|, \\ \text{If } \mathcal{F}_g^* = \varnothing \text{ then } F_g^* := \varnothing. \end{cases}$$

By Lemma 5.1 (ii), the small-sets in $\mathcal{F}_g^*$ are disjoint thus:

$$\left|\mathcal{F}_g^*\right| \leq \left|\Gamma_Z(g)\right| = d_V + d_C.$$

Hence:

$$\sum_{F \in \mathcal{F}_g^*} \left|\sigma_X(F)\right| \leq \left|\sigma_X(F_g^*)\right|\left|\mathcal{F}_g^*\right| \leq \left|\sigma_X(F_g^*)\right|\left(d_V + d_C\right). \tag{5.16}$$

The average value of $\sum_{g\in C_Z^k}\left|\sigma_X(F_g^*)\right|$ over $k\in[\![0;\chi-1]\!]$ satisfies:

$$\begin{aligned}\frac{1}{\chi}\sum_{k=0}^{\chi-1}\sum_{g\in C_Z^k}\left|\sigma_X(F_g^*)\right| &= \frac{1}{\chi}\sum_{g\in C_Z}\left|\sigma_X(F_g^*)\right| \\ &\geq \frac{1}{\chi(d_V+d_C)}\sum_{g\in C_Z}\sum_{F\in\mathcal{F}_g^*}\left|\sigma_X(F)\right| \quad \text{by eq. (5.16)}, \\ &= \frac{1}{\chi(d_V+d_C)}\sum_{F\in\mathcal{F}^*}\left|\sigma_X(F)\right| \\ &\geq \frac{c_1\left|\sigma_X(E)\right|-c_2\left|D\right|}{\chi(d_V+d_C)} \qquad \text{by Lemma 5.1 (iii)}.\end{aligned}$$

Thus there exists a color $k\in[\![0;\chi-1]\!]$ such that the set $\mathcal{F}_0:=\{F_g^*:g\in C_Z^k\}$ satisfies:

$$\sum_{F\in\mathcal{F}_0}\left|\sigma_X(F)\right|\geq\frac{c_1\left|\sigma_X(E)\right|-c_2\left|D\right|}{\chi(d_V+d_C)}. \tag{5.17}$$

Let i be the integer in $[\![1;\chi]\!]$ such that $i-1$ is a step of the algorithm where we flip the color k. In other words, $k=(i-1) \mod \chi$.
We define $\mathcal{F}_1\subseteq\mathcal{F}$ the set of all flips done during the steps $0,\dots,i-2$ and $\mathcal{F}_2\subseteq\mathcal{F}_k$ the set of all flips done at step $i-1$. By the coloring property of Lemma 5.25, we have $\sigma_X(F)\cap\sigma_X(F')=\varnothing$ for all $F,F'\in\mathcal{F}_0$, thus for all $F''\in\mathcal{F}_1$, each check-node of $\sigma_X(F'')$ is in the syndrome of at most one element of $\mathcal{F}_0$. In other words, for a given syndrome $\widetilde{\sigma}$ and for $F''\in\mathcal{F}_1$, the number of $F\in\mathcal{F}_0$ such that $\Delta(\widetilde{\sigma},F)\neq\Delta(\widetilde{\sigma}\oplus\sigma_X(F''),F)$ is at most $\left|\sigma_X(F'')\right|$ (as a reminder, $F_g^*\in\mathcal{F}_g^*$). As a consequence:

$$\left|\mathcal{F}_2\right|\geq\left|\mathcal{F}_0\right|-\sum_{F\in\mathcal{F}_1}\left|\sigma_X(F)\right|. \tag{5.18}$$

Moreover:

$$\left|\widetilde{\sigma}_0\right|-\left|\widetilde{\sigma}_{i-1}\right|\geq\sum_{F\in\mathcal{F}_1}\beta_1\left|\sigma_X(F)\right|, \tag{5.19}$$

and

$$\left|\widetilde{\sigma}_{i-1}\right|-\left|\widetilde{\sigma}_i\right|\geq\sum_{F\in\mathcal{F}_2}\beta_1\left|\sigma_X(F)\right|\geq\beta_1\left|\mathcal{F}_2\right|. \tag{5.20}$$

Combining eqs. (5.18) to (5.20) we get:

$$\sum_{F\in\mathcal{F}_0}\left|\sigma_X(F)\right|\leq d_Vd_C\left|\mathcal{F}_0\right|\leq d_Vd_C\left(\left|\mathcal{F}_2\right|+\sum_{F\in\mathcal{F}_1}\left|\sigma_X(F)\right|\right)\leq\frac{d_Vd_C}{\beta_1}\Big(\left|\widetilde{\sigma}_0\right|-\left|\widetilde{\sigma}_i\right|\Big).$$

By eq. (5.17):

$$\begin{aligned}\left|\widetilde{\sigma}_0\right| - \left|\widetilde{\sigma}_i\right| &\geq \frac{\beta_1}{d_V d_C} \sum_{F \in \mathcal{F}_0} \left|\sigma_X(F)\right| \\ &\geq \frac{\beta_1 \Big(c_1 \left|\sigma_X(E)\right| - c_2 \left|D\right|\Big)}{d_V d_C \chi (d_V + d_C)} \\ &= (1-\eta)\left|\sigma_X(E)\right| - (c_5 - 1 + \eta)\left|D\right|.\end{aligned}$$

Thus:

$$\left|\widetilde{\sigma}_\chi\right| \leq \left|\widetilde{\sigma}_i\right| \leq \left|\widetilde{\sigma}_0\right| - (1-\eta)\left|\sigma_X(E)\right| + (c_5 - 1 + \eta)\left|D\right| \leq \eta \left|\widetilde{\sigma}_0\right| + c_5 \left|D\right|.$$

□

When we apply Lemma 5.28 several times, we get Lemma 5.29 below: the weight of the observed syndrome decreases exponentially fast as a function of f.

Lemma 5.29. *We use the notations of Section 5.2.1 assuming $\left|U\right| \leq \gamma_0 \sqrt{N}$ then:*

$$\left|\widetilde{\sigma}_f\right| \leq \eta^{f/\chi} \left|\widetilde{\sigma}_0\right| + c_6 \left|D\right|.$$

Proof. By Lemma 5.28, for all i:

$$\left|\widetilde{\sigma}_{(i+1)\chi}\right| \leq \eta \left|\widetilde{\sigma}_{i\chi}\right| + c_5 \left|D\right|.$$

By an induction on i:

$$\left|\widetilde{\sigma}_{i\chi}\right| \leq \eta^i \left|\widetilde{\sigma}_0\right| + \left(c_5 \sum_{k=0}^{i-1} \eta^k\right) \left|D\right|.$$

Hence we can upper bound $\left|\sigma_f\right|$ by:

$$\left|\sigma_f\right| \leq \eta^{f/\chi} \left|\sigma_0\right| + c_6 \left|D\right|.$$

□

5.2.4 Constant time decoding

In this section, we fix $f = f_0$ where f_0 is defined in Section 5.2.1. As stated in Theorem 5.30 below, if the initial error is local stochastic with parameter sufficiently small then the residual error is also local stochastic with high probability.

Theorem 5.30. *We use the notations of Section 5.1.1.*
There exist two non-zero constants $p_1, p_2 > 0$ such that the following holds. Suppose the pair (E, D) satisfies a local stochastic noise model with parameter $(p_{\text{phys}}, p_{\text{synd}})$ where $p_{\text{phys}} < p_1$ and $p_{\text{synd}} < p_2$. If we run Algorithm 4 with f_0 steps on the input (E, D), then there exists a random variable $E_{\text{ls}} \subseteq V_{\mathcal{Q}}$ with a local stochastic distribution with parameter $p_{\text{ls}} = p_1^c$ and such that:

$$\mathbb{P}\Big[E_{\text{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] \leq e^{-\Theta(\sqrt{N})}.$$

Note that the constant integer f_0 depends on the parameter $c > 1$ appearing in the value of p_{ls} in Theorem 5.30. When c increases, the local stochastic parameter p_{ls} of the residual error gets better but the number of steps f_0 grows. On the other hand, when c is chosen close to 1, p_{ls} gets worse but the number of steps drops.

The proof of Theorem 5.30 proceeds in the same way as the proof of Theorem 5.6. Indeed, it is sufficient to find a reduced error E_{ls} equivalent to $E \oplus \hat{E}$ and such that for all $S \subseteq E_{\text{ls}}$ there is a *p-witness* for S in the sense of Definition 5.32 (the "p" in p-witness stands for parallel).
We start by establishing the existence of p-witnesses under the assumptions that the residual error $E \oplus \hat{E}$ is reduced with weight smaller than $\gamma_0\sqrt{N}$. This will be done in Lemma 5.33 below but we first show that the weight of the residual error is upper bounded by a linear function of the initial error weight (Lemma 5.31).

Lemma 5.31. *We use the notations of Section 5.2.1 running Algorithm 4 with f_0 steps. Suppose $|U| \leq \gamma_0\sqrt{N}$ and $E \oplus \hat{E}$ is reduced then*

$$|E \oplus \hat{E}| \leq \frac{1}{4c}|E| + c_7|D|.$$

Proof. The error $E \oplus \hat{E}$ is reduced and satisfies $|E \oplus \hat{E}| \leq |U| \leq \gamma_0\sqrt{N}$, hence:

$$\begin{aligned}
|E \oplus \hat{E}| &\leq \frac{2}{\beta_0 d_V}|\sigma_X(E \oplus \hat{E})| && \text{by Lemma 4.21,}\\
&\leq \frac{2}{\beta_0 d_V}\Big(|\widetilde{\sigma}_{f_0}| + |D|\Big)\\
&\leq \frac{2}{\beta_0 d_V}\Big(\eta^{f_0/\chi}|\widetilde{\sigma}_0| + (1 + c_6)|D|\Big) && \text{by Lemma 5.29,}\\
&\leq \frac{2}{\beta_0 d_V}\Big(\eta^{f_0/\chi}|\sigma_X(E)| + \big(1 + \eta^{f_0/\chi} + c_6\big)|D|\Big)\\
&\leq \frac{1}{4c}|\sigma_X(E)| + c_7|D|.
\end{aligned}$$

□

The definition of p-witness (Definition 5.32) is a bit different from the definition of witness we gave for the sequential algorithm (Definition 5.14).

Definition 5.32 (p-witness)**.** We use the notations of Section 5.2.1.
We say that $W \subseteq V$ is a *p-witness* for $S \subseteq V$ if $W \in \mathcal{M}(S)$ and:

$$\begin{aligned}
|W| &\leq c_3|E \cap W| + c_4|D \cap \Gamma_X(W)|,\\
|S| &\leq \frac{1}{4c}|E \cap W| + c_7|D \cap \Gamma_X(W)|.
\end{aligned}$$

It remains to show the existence of p-witnesses for the residual error as stated in Lemma 5.33 and Lemma 5.34.

Lemma 5.33. *We use the notations of Section 5.2.1 running Algorithm 4 with f_0 steps. Suppose $|U| + |D| \leq \gamma_0\sqrt{N}$ and the residual error $E \oplus \hat{E}$ is reduced then for all $S \subseteq E \oplus \hat{E}$, there is a p-witness W for S with $W \subseteq U$.*

Proof. Let W be the union of the connected components of U in $\mathcal{G}$ which contains at least one element of S. We have $W \cap \Gamma_{\mathcal{G}}(U \setminus W) = \varnothing$. Thus using Lemma 5.27 with $K = W$, there is a valid execution of Algorithm 4 on input $(E \cap W, D \cap \Gamma_X(W))$, with output $\hat{E} \cap W$ and support $U \cap W = W$. We have $|W| \leq \gamma_0\sqrt{N}$ and the residual error $(E \oplus \hat{E}) \cap W$ is reduced as a subset of the reduced error $E \oplus \hat{E}$ (see Lemma 4.15). Hence we can apply Lemma 5.31:

$$|S| \leq |(E \oplus \hat{E}) \cap W| \leq \frac{1}{4c}|E \cap W| + c_7|D \cap \Gamma_X(W)|.$$

By Remark 5.26:

$$|W| \leq c_3|E \cap W| + c_4|D \cap \Gamma_X(W)|.$$

□

Lemma 5.34. *We use the notations of Section 5.2.1 running Algorithm 4 with f_0 steps. If $\text{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ then there is a reduced error E_{ls} equivalent to the residual error $E \oplus \hat{E}$ such that for all $S \subseteq E_{\text{ls}}$ there is a p-witness for S.*

Proof. The proof is identical to the proof of Lemma 5.17 using Lemma 5.27 instead of Corollary 5.10 and Lemma 5.33 instead of Lemma 5.15. □

Finally, we can prove Theorem 5.30.

Proof of Theorem 5.30. Let $p_{\text{th}} > 0$ as defined in Lemma 5.18. We define $p_1, p_2 > 0$ to be some positive numbers such that:

$$\begin{aligned} &p_1 \leq p_{\text{th}}, && p_2 \leq p_{\text{th}}, && 2^{d_C/c_8}p_2 \leq 1/2, && 2^{1/c_9}p_1^{1/2} \leq 1/2, \\ &2^{d_C}edp_2^{c_8} \leq 1/2, && 2edp_1^{c_9/2} \leq 1/2, && \frac{4}{e}2^{d_C}ep_2^{c_8} \leq p_1^c/2, && \frac{4}{e}2ep_1^{c_9/2} \leq 1/2. \end{aligned} \tag{5.21}$$

Let $p_{\text{phys}} < p_2$ and $p_{\text{synd}} < p_2$, we assume that the pair (E, D) satisfies a local stochastic noise model with parameter $(p_{\text{phys}}, p_{\text{synd}})$ and we run Algorithm 3 on the input (E, D). We define the random error E_{ls} promised by Theorem 5.30 in the following way:

$$\begin{cases} \text{If } \text{MaxConn}_{\alpha_1}(E \cup D) > \gamma_0\sqrt{N} \text{ then } E_{\text{ls}} = \varnothing. \\ \text{If } \text{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N} \text{ then } E_{\text{ls}} \text{ is the error promised by Lemma 5.34.} \end{cases}$$

Lemma 5.34 ensures that if $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$ then E_{ls} is equivalent to $E \oplus \hat{E}$ thus by Lemma 5.18:

$$\begin{aligned}\mathbb{P}\Big[E_{\mathrm{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] &\leq \mathbb{P}\Big[\mathrm{MaxConn}_{\alpha_1}(E \cup D) > \gamma_0\sqrt{N}\Big] \\ &\leq C\big|\mathcal{V}\big| \left(\frac{\max(p_{\mathrm{phys}}, p_{\mathrm{synd}})}{p_{\mathrm{th}}}\right)^{\alpha_1\gamma_0\sqrt{N}}\end{aligned}$$

We have $\big|\mathcal{V}\big| = \Theta(N)$ and $\max(p_{\mathrm{phys}}, p_{\mathrm{synd}}) < p_{\mathrm{th}}$ hence as stated in Theorem 5.30:

$$\mathbb{P}\Big[E_{\mathrm{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] = e^{-\Theta(\sqrt{N})}.$$

It remains to show that E_{ls} is local stochastic with parameter $p_{\mathrm{ls}} = p_1^c$. We know by Lemma 5.34 that provided $\mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}$, there is a p-witness W for any $S \subseteq E_{\mathrm{ls}}$ thus for all $S \subseteq V_{\mathcal{Q}}$:

$$\begin{aligned}\mathbb{P}\Big[S \subseteq E_{\mathrm{ls}}\Big] &= \mathbb{P}\Big[S \subseteq E_{\mathrm{ls}} \text{ and } \mathrm{MaxConn}_{\alpha_1}(E \cup D) \leq \gamma_0\sqrt{N}\Big] \\ &\leq \mathbb{P}\Big[\text{There exists a p-witness } W \text{ for } S\Big]. \qquad (5.22)\end{aligned}$$

To conclude, we upper bound the probability that a p-witness exists.

For a fixed $W \subseteq V$, we distinguish between two cases:

$$\big|E \cap W\big| \leq 4cc_7\big|D \cap \Gamma_X(W)\big|, \qquad (5.23)$$

$$\big|E \cap W\big| \geq 4cc_7\big|D \cap \Gamma_X(W)\big|. \qquad (5.24)$$

Let us upper bound the probability that W is a p-witness for S in the case where eq. (5.23) holds. By Definition 5.32:

$$\big|W\big| \leq c_3\big|E \cap W\big| + c_4\big|D \cap \Gamma_X(W)\big| \leq \big|D \cap \Gamma_X(W)\big|/c_8.$$

In particular, the set $T := D \cap \Gamma_X(W)$ satisfies $T \subseteq \Gamma_X(W)$, $\big|W\big| \leq \big|T\big|/c_8$ and $T \subseteq D$. Thus:

$$\begin{aligned}&\mathbb{P}\Big[W \text{ is a p-witness for } S \text{ and eq. (5.23) holds}\Big] \\ &\qquad \leq \mathbb{P}\Big[\exists T \subseteq \Gamma_X(W) : \big|W\big| \leq \big|T\big|/c_8, T \subseteq D\Big] \\ &\qquad \leq \sum_{\substack{T \subseteq \Gamma_X(W): \\ |T| \geq c_8|W|}} p_{\mathrm{synd}}^{|T|} \\ &\qquad \leq \sum_{t \geq c_8|W|} 2^{d_C|W|} p_{\mathrm{synd}}^t \\ &\qquad \leq \sum_{t \geq c_8|W|} \Big(2^{d_C/c_8} p_{\mathrm{synd}}\Big)^t.\end{aligned}$$

By eq. (5.21), we have $2^{d_C/c_8} p_{\text{synd}} \leq 1/2$ thus:

$$\mathbb{P}\Big[W \text{ is a p-witness for } S \text{ and eq. (5.23) holds}\Big] \leq 2\left(2^{d_C} p_{\text{synd}}^{c_8}\right)^{|W|}. \tag{5.25}$$

In the second case where W is a p-witness for S and eq. (5.24) holds, we have:

$$\big|S\big| \leq \left(\frac{1}{4c} + \frac{1}{4c}\right)\big|E \cap W\big| \leq \frac{\big|E \cap W\big|}{2c}$$

and

$$\big|W\big| \leq \big|E \cap W\big|/c_9.$$

Thus the set $T := E \cap W$ satisfies $T \subseteq W$, $\big|S\big| \leq \big|T\big|/2c$, $\big|W\big| \leq \big|T\big|/c_9$ and $T \subseteq E$. Thus:

$$\begin{aligned}
&\mathbb{P}\Big[W \text{ is a p-witness for } Sandeq.\ (5.24) holds\Big] \\
&\qquad \leq \mathbb{P}\Big[\exists T \subseteq W : \big|S\big| \leq \big|T\big|/2c, \big|W\big| \leq \big|T\big|/c_9, T \subseteq E\Big] \\
&\qquad \leq \sum_{\substack{T \subseteq W: \\ |T| \geq c_9|W|, \\ |T| \geq 2c|S|}} \left(p_{\text{phys}}^{|T|/2}\right)^2 \\
&\qquad \leq p_{\text{phys}}^{c|S|} \sum_{\substack{T \subseteq W: \\ |T| \geq c_9|W|}} p_{\text{phys}}^{|T|/2} \\
&\qquad \leq p_{\text{phys}}^{c|S|} \sum_{t \geq c_9|W|} 2^{|W|} p_{\text{phys}}^{t/2} \\
&\qquad \leq p_{\text{phys}}^{c|S|} \sum_{t \geq c_9|W|} \left(2^{1/c_9} p_{\text{phys}}^{1/2}\right)^t.
\end{aligned}$$

By eq. (5.21), we have $2^{1/c_9} p_{\text{phys}}^{1/2} \leq 1/2$ thus:

$$\mathbb{P}\Big[W \text{ is a p-witness for } S \text{ and eq. (5.24) holds}\Big] \leq 2p_{\text{phys}}^{c|S|}\left(2p_{\text{phys}}^{c_9/2}\right)^{|W|}. \tag{5.26}$$

Thus:

$$\begin{aligned}
\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] &\leq \mathbb{P}\Big[\text{There exists a p-witness } W \text{ for } S\Big] \qquad \text{by eq. (5.22),} \\
&\leq \sum_{W \in \mathcal{M}(S)} \mathbb{P}\Big[W \text{ is a p-witness for } S\Big] \\
&\leq \sum_{W \in \mathcal{M}(S)} \mathbb{P}\Big[W \text{ is a p-witness for } S \text{ and eq. (5.23) holds}\Big] \\
&\qquad + \mathbb{P}\Big[W \text{ is a p-witness for } S \text{ and eq. (5.24) holds}\Big]
\end{aligned}$$

By eqs. (5.25) and (5.26):

$$\begin{aligned}\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] &\leq 2 \sum_{W \in \mathcal{M}(S)} \left[\left(2^{d_C} p_{\text{synd}}^{c_8}\right)^{|W|} + p_{\text{phys}}^{c|S|}\left(2p_{\text{phys}}^{c_9/2}\right)^{|W|}\right] \\ &\leq \frac{2}{ed^{|S|}} \sum_{w \geq |S|} \left[\left(2^{d_C} e d p_{\text{synd}}^{c_8}\right)^{w} + p_{\text{phys}}^{c|S|}\left(2ed p_{\text{phys}}^{c_9/2}\right)^{w}\right] \quad \text{by eq. (5.7).}\end{aligned}$$

By eq. (5.21), we have $2^{d_C} e d p_{\text{synd}}^{c_8} \leq 1/2$ and $2ed p_{\text{phys}}^{c_9/2} \leq 1/2$ thus:

$$\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] \leq \frac{4}{e}\left(2^{d_C} e p_{\text{synd}}^{c_8}\right)^{|S|} + \frac{4}{e} p_{\text{phys}}^{c|S|}\left(2ep_{\text{phys}}^{c_9/2}\right)^{|S|}.$$

By eq. (5.21), we have $\frac{4}{e} 2^{d_C} e p_{\text{synd}}^{c_8} \leq p_1^c/2$ and $\frac{4}{e} 2ep_{\text{phys}}^{c_9/2} \leq 1/2$ thus:

$$\mathbb{P}\Big[S \subseteq E_{\text{ls}}\Big] \leq \frac{p_1^{c|S|}}{2} + \frac{p_1^{c|S|}}{2} = p_1^{c|S|}.$$

□

5.2.5 Logarithmic time decoding

In this section, we run Algorithm 4 with $f_1(|\widetilde{\sigma}|)$ steps where the function f_1 is defined in Section 5.2.1. The main benefit of this implementation compared to Section 5.2.4 is the following: the smaller p_{synd} is, the smaller the local stochastic parameter of the residual error is. In particular, if the measurement is perfect ($D = \varnothing$ and $p_{\text{synd}} = 0$) then the error on the qubits is entirely corrected and this will be important for performing the gate teleportation technique in Chapter 6.

Lemma 5.35. *We use the notations of Section 5.2.1 running Algorithm 4 with $f = f_1(|\widetilde{\sigma}|)$ steps.*
Suppose that $E \oplus \hat{E}$ is reduced with $\big|U\big| \leq \gamma_0\sqrt{N}$ then:

$$\big|E \oplus \hat{E}\big| \leq c_0' \big|D\big|.$$

Proof. The error $E \oplus \hat{E}$ is reduced and satisfies $\big|E \oplus \hat{E}\big| \leq \big|U\big| \leq \gamma_0\sqrt{N}$ hence:

$$\begin{aligned}\big|E \oplus \hat{E}\big| &\leq \frac{2}{\beta_0 d_V}\big|\sigma_X(E \oplus \hat{E})\big| && \text{by Lemma 4.21,} \\ &\leq \frac{2}{\beta_0 d_V}\Big(\big|\widetilde{\sigma}_f\big| + \big|D\big|\Big) \\ &\leq \frac{2}{\beta_0 d_V}\Big(\eta^{f/\chi}\big|\widetilde{\sigma}_0\big| + (1+c_6)\big|D\big|\Big) && \text{by Lemma 5.29,} \\ &< 1 + \frac{2(1+c_6)}{\beta_0 d_V}\big|D\big| && \text{Because } f = f_1(|\widetilde{\sigma}_0|), \\ &\leq 1 + c_0'\big|D\big|.\end{aligned}$$

We have $\big|E \oplus \hat{E}\big| < 1 + c_0'\big|D\big|$ and all the values are integers thus Lemma 5.35 holds. □

Lemma 5.35 is the same statement as Proposition 5.2 except that c_0 is replaced by c_0' and Algorithm 3 is replaced by Algorithm 4. Hence, if we rewrite the lemmas and the proofs of Section 5.1 replacing c_0 by c_0' then we end up with Theorem 5.36 below.

Theorem 5.36. *We use the notations of Section 5.2.1 running Algorithm 4 with $f = f_1(|\widetilde{\sigma}|)$ steps.*
There exists a non-zero constant $p_3 > 0$ such that the following holds. Suppose the error (E, D) satisfies a local stochastic noise model with parameter $(p_{\text{phys}}, p_{\text{synd}})$ where $p_{\text{phys}} < p_3$ and $p_{\text{synd}} < p_3$. If we run Algorithm 3 with $f = f_1\big(|\sigma_X(E) \oplus D|\big)$ steps on the input (E, D) then there exists a random variable $E_{\text{ls}} \subseteq V_{\mathcal{Q}}$ with a local stochastic distribution with parameter $p_{\text{ls}} := p_{\text{synd}}^{1/(2c_0')}$ and such that:

$$\mathbb{P}\Big[E_{\text{ls}} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\Big] \leq e^{-\Theta(\sqrt{N})}.$$

Proof. We rewrite the statements and the proofs of Section 5.1 replacing c_0 by c_0' and Algorithm 3 by Algorithm 4. In more details:

- Proposition 5.2 is replaced by Lemma 5.35.
- Lemma 5.3 and Corollary 5.12 are replaced by Remark 5.26.
- Corollary 5.10 is replaced by Lemma 5.27.
- c_0 is replaced by c_0' in the definition of witness (Definition 5.14).
- Lemma 5.15 and Lemma 5.17 are proved in the same way.
- The proof of Theorem 5.36 is the same as the proof of Theorem 5.6.

□

An interesting consequence of Theorem 5.36 is the case where the syndrome is noiseless (see Corollary 5.37 below).

Corollary 5.37. *We use the notations of Section 5.2.1.*
There exists a non-zero constant $p_3 > 0$ such that the following holds. Suppose $D = \varnothing$ and E satisfies a local stochastic noise model with parameter $p_{\text{phys}} < p_3$ then if we run Algorithm 4 with $f_1(|\sigma_X(E)|)$ steps:

$$\mathbb{P}\Big[E \text{ is corrected}\Big] \geq 1 - e^{-\Theta(\sqrt{N})}.$$

Proof. Similar to the proof of Corollary 5.7. □

5.3 Numerical simulations for the small-set-flip algorithm

The results we have presented in Sections 5.1 and 5.2 are interesting from a theoretical perspective but are not of practical interest: as shown in Section 5.3.1 below, the lower bound we get on the threshold is small ($< 10^{-58}$) and the proofs hold when the weights of the stabilizer generators are large (≥ 35). In Section 5.3.2, we present simulation results showing that the real threshold value is near 4.5% for codes with rate $1/61$ and stabilizer weight 11, and near 2% for codes with rate $1/5$ and stabilizer weight 15.

5.3.1 Theoretical lower bound on the threshold

In Section 5.1.1, we assumed that the expansion parameter δ of the initial classical expander code satisfies $\delta < 1/16$, which requires left degree $d_V \geq 17$ (by Lemma 3.11) and right degree $d_C \geq d_V + 1$ (to get a code family with non-zero rate). The stabilizer generators of the resulting quantum expander code have weight $d_V + d_C \geq 35$, the VV-type qubits have weight $2d_V \geq 34$ and the CC-type qubits have weight $2d_C \geq 36$. Moreover, the thresholds given by Theorems 5.6, 5.30 and 5.36 are very low since they are smaller than p_{th} defined in Theorem 5.24:

$$p_{\text{th}} = \left(\frac{2^{-h(\alpha_1)}}{\left(d_{\mathcal{G}} - 1\right)\left(1 + \frac{1}{d_{\mathcal{G}}-2}\right)^{d_{\mathcal{G}}-2}} \right)^{\frac{1}{\alpha_1}},$$

with:

$$r := d_V/d_C, \quad d_{\mathcal{G}} := d_C(d_C + 2d_V - 2), \quad \beta_1 := \frac{1 - 16\delta}{2}, \quad \alpha_1 := \frac{r\beta_1}{4 + 2r\beta_1}.$$

By Lemma 3.11, we have $\delta > 1/d_V$ thus:

$$\beta_1 < \beta_1^* := \frac{1 - 16/d_V}{2}, \qquad \alpha_1 < \alpha_1^* := \frac{r\beta_1^*}{4 + 2r\beta_1^*},$$

$$p_{\text{th}} < p_{\text{th}}^* := \left(\frac{1}{\left(d_{\mathcal{G}} - 1\right)\left(1 + \frac{1}{d_{\mathcal{G}}-2}\right)^{d_{\mathcal{G}}-2}} \right)^{\frac{1}{\alpha_1^*}}.$$

For instance, if we set $d_C = d_V + 1$ then the maximum of the function $d_V \mapsto p_{\text{th}}^*$ is reached for $d_V = 66$ where $p_{\text{th}}^* \approx 10^{-58}$. This value is not physically reasonable, but the numerical simulations of Section 5.3.2 show that the real threshold is much better than the lower bound p_{th}.

5.3.2 Results

Algorithm 5 : the small-set-flip decoder of ref. [67].

Input: the syndrome $\sigma \subseteq C_X$. // $\sigma = \sigma(E)$ for some $E \subseteq V_{\mathcal{Q}}$
Output: a guess for the error $\hat{E} \subseteq V_{\mathcal{Q}}$.

$\hat{E}_0 = \varnothing$; $\sigma_0 = \sigma$; $i = 0$
while $\exists F \in \mathcal{F}_0 : \Delta(\sigma_i, F) > 0$ **do**
  $F_i := \underset{F \in \mathcal{F}_0}{\arg\max} \dfrac{\Delta(\sigma_i, F)}{|F|}$
  $\hat{E}_{i+1} = \hat{E}_i \oplus F_i$
  $\sigma_{i+1} = \sigma_i \oplus \sigma_X(F_i)$
  $i = i + 1$
end while
return $\hat{E}_i$

The simulations presented in this section have been done using the sequential small-set-flip algorithm of ref. [67] (Algorithm 5) with a perfect syndrome measurement and an iid bit-flip error model (each qubit is flipped with probability p independently). In Algorithm 5, $V_{\mathcal{Q}}$ is the set of qubits, C_X is the set of check-nodes, $\mathcal{F}_0$ is the set of small-sets as defined in eq. (4.25) and Δ is the function defined in eq. (4.26):

$$\mathcal{F}_0 := \Big\{ F \subseteq V_{\mathcal{Q}} \text{: } F \text{ is included in the support of an } X\text{-type generator} \Big\},$$

$$\Delta(\sigma, F) := \big|\sigma\big| - \big|\sigma \oplus \sigma_X(F)\big|.$$

At each round, Algorithm 5 selects the small-set $F \in \mathcal{F}_0$ which maximizes $\dfrac{\Delta(\sigma_i, F)}{|F|}$ and flips the qubits of F.

The classical codes used in the hypergraph product construction are regular LDPC codes generated with the configuration model and the *switching method* as described in Section 5.3.3 [79]. For short, we will say that a code family has degrees (d_V, d_C) or is a (d_V, d_C) family to indicate that the left degree of the classical Tanner graph is d_V and the right degree is d_C. The first hypergraph product family we consider has rate $1/61 \sim 1.6\%$ and is constructed from classical codes with rate $1/6$ and degrees $(5, 6)$ (see Figure 5.2). The second family has rate $1/5 = 20\%$ and is constructed from classical codes with rate $1/2$ and degrees $(5, 10)$ (see Figure 5.3).

For a given error probability $p \in [0, 1]$ called *physical error rate*, we have used Monte Carlo simulations to estimate the *block error rate*, which is the probability of failure when the small-set flip algorithm corrects an error where each qubit is flipped with probability p independently. Hence, the block error rate is the success rate of the following protocol:

– Each qubit is flipped with probability p independently.

- Algorithm 5 is used to correct the error.
- If the residual error on the qubits is equivalent to an empty error then the protocol is a success.

The block error rate as a function of the physical error rate is plotted in Figure 5.2 for the $(5, 6)$ family and in Figure 5.3 for the $(5, 10)$ family. The usual way to determine numerically the threshold is to identify the *pseudo-threshold* defined to be the physical error rate where the curves of Figures 5.2 and 5.3 cross. Using Figure 5.4, the pseudo-threshold is near 4.5% for the $(5, 6)$ family. Similarly, our numerical results show that the pseudo-threshold is near 2% for the $(5, 10)$ family. Unfortunately, near the pseudo-threshold, the block error rate is close to 1 meaning that these families cannot be used at physical error rate close to the threshold.

For the $(5, 6)$ family, we plot in Figure 5.5 the block error rate as a function of $\sqrt{K}$ where K the number of logical qubits. Using the results of Theorem 5.6, we expect to have $p \sim ae^{-b\sqrt{K}}$ but we have not been able to fit the curves in that way.

In Figures 5.6 and 5.7, we show how evolves the small-set-flip performance depending on the degrees of the classical codes. When $d_V = 4$, the results are not satisfying since increasing the block length of the codes does not clearly increase the success rate, see Figure 5.6. When $d_V = 6$, we get codes with smaller rate and the decoder performance is quite similar than for $d_V = 5$, see Figure 5.7.

As a conclusion, the lower bound on the threshold presented in Section 5.3.1 is very pessimistic since the small-set-flip decoder can be used at decent error rates (Figure 5.2 and Figure 5.3). Moreover, the $(5, 6)$ family seems to be optimal for the small-set-flip algorithm since this decoder does not work for a $(4, 5)$ family (Figure 5.6) and has a worse performance for a $(6, 7)$ family (Figure 5.7).

5.3.3 Code generation

The configuration model described in Section 3.2.4 builds regular Tanner graphs, however it is known that other techniques provide codes with better performance [57, 76, 100]. In this work, we rely on the *switching method* introduced by McKay and Brendan in ref. [79] to increase the *girth* of the Tanner graph (the girth is the length of the smallest cycle) [112]. We say that two edges (v_1, c_1) and (v_2, c_2) of a Tanner graph are switched when they are removed and replaced by (v_1, c_2) and (v_2, c_1) (note that this transformation does not change the node degrees and preserves the bipartite property). To generate our codes, we have built an initial Tanner graph using the configuration model and we have switched well chosen edges to increase the girth of the graph. As shown in Figure 5.2 and Figure 5.8, an $[[N, K]]$ stabilizer code generated with the switching method has similar performance than a $[[4N, 4K]]$ stabilizer code generated directly with the configuration model. Except for the codes of Figure 5.8, the classical Tanner graphs have been created with the switching method as described below.

Let n be the number of bit-nodes of a classical Tanner graph G and let v be a bit-node, then we define $l_v \in \{2, 4, 8, \ldots, n\}$ to be the size of the smallest cycle containing v (l_v is even since the graph is bipartite) and $m_v \in \mathbb{N}^*$ to be the number of cycles with

size l_v which contain v. For instance, $l_v = 2$ if and only if v belongs to a double edge in G. For $l \in \{2, 4, 8, \ldots, n\}$ and $m \in \mathbb{N}^*$, we define $s_1(l, m)$ to be the number of bit-nodes v such that $l_v = l$ and $m_v = m$:

$$s_1(l, m) = \#\Big\{v \in V_{\mathcal{Q}} : l_v = l \text{ and } m_v = m\Big\}.$$

For a given $l \in \{2, 4, 8, \ldots, n\}$, we define $s_2(l)$ to be the tuple:

$$s_2(l) = \Big(s_1(l, M), s_1(l, M-1), s_1(l, M-2), \ldots, s_1(l, 0)\Big)$$

where $M := \max\Big\{m : s_1(l, m) \neq 0\Big\}$. Finally, we define the score associated to a Tanner graph G by the tuple:

$$s_G = \Big(s_2(2), s_2(4), s_2(6), \ldots, s_2(n)\Big).$$

Two scores are compared using the lexicographic order on tuples.

The classical codes used in the simulations have been generated in the following way:

- Create a Tanner graph using the configuration model (keeping the double edges when they exist).
- Repeat many times:
  - Pick two edges randomly and switch them if the score of the Tanner graph decreases.

As a remark, we have used the score s_G because it leads to the best codes we have been able to produce, but there is no theoretical guarantee that this is the optimal quantity to consider.

5.3.4 Plots

For all plots, the error bars represent the 99% confidence intervals (≈ 2.58 standard deviation).

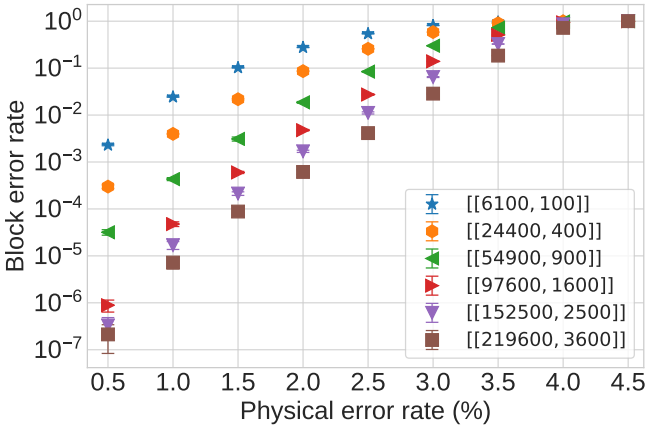

Figure 5.2: simulation results for the $(5, 6)$ family.

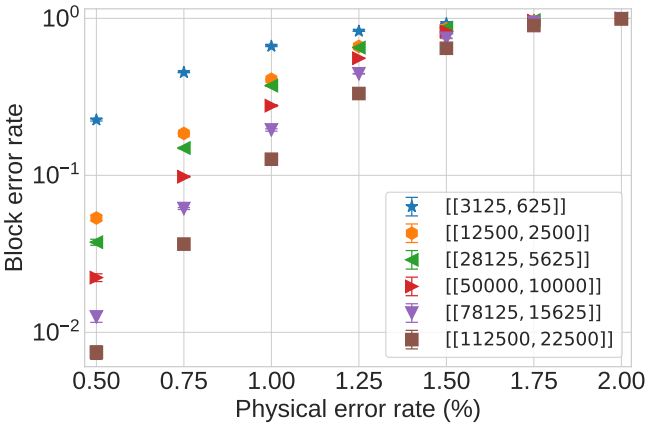

Figure 5.3: simulation results for the $(5, 10)$ family.

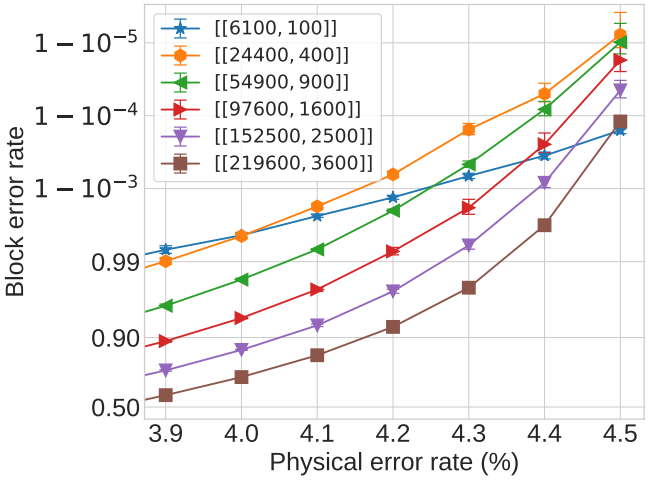

Figure 5.4: simulation results for the $(5, 6)$ family.

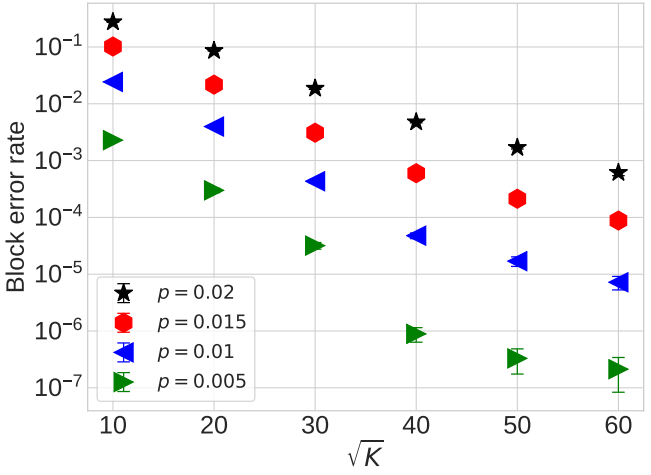

Figure 5.5: simulation results for the $(5, 6)$ family.

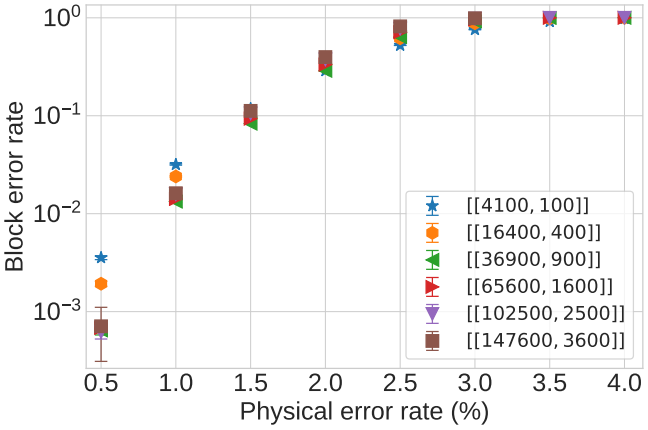

Figure 5.6: simulation results for the $(4, 5)$ family.

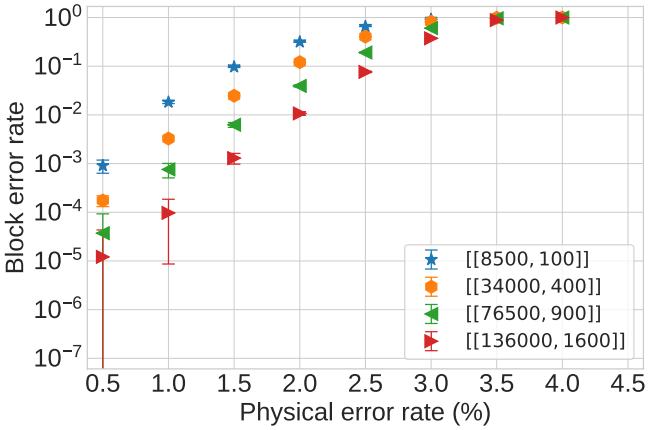

Figure 5.7: simulation results for the $(6, 7)$ family.

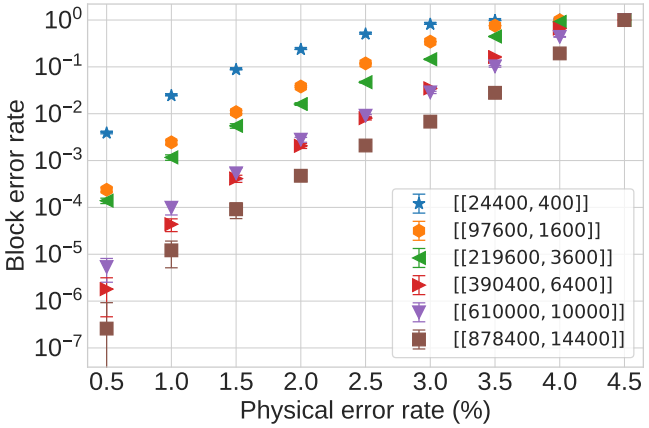

Figure 5.8: simulation results for a $(5, 6)$ family generated without the switching method.

Chapter 6

Fault-tolerant quantum computation

In fault-tolerant quantum computation, the objective is to perform quantum computations with circuits built from noisy gates. The *threshold theorem* asserts this is indeed possible if the noise on the gates is below a constant threshold value: given an arbitrary circuit C, there exists an equivalent circuit C′ such that the output of C′ is not modified by a small number of faulty gates [1]. The circuit C′ is divided into error correction cycles where an $[[N, K]]$ stabilizer code $\mathcal{Q}$ is used to remove the errors, and simulation cycles where the information encoded into $\mathcal{Q}$ is processed with *logical gates*. Formally, let U be a unitary on K qubits, then the associated logical unitary U^{L} satisfies for all $|\varphi\rangle \in \left(\mathbb{C}^2\right)^{\otimes K}$:

$$U^{\mathrm{L}}\mathcal{E}\big(\,|\varphi\rangle\,\big) = \mathcal{E}\big(U\,|\varphi\rangle\,\big)$$

where $\mathcal{E}\big(\,|\varphi\rangle\,\big) \in \left(\mathbb{C}^2\right)^{\otimes N}$ is the code state associated to $|\varphi\rangle$ and $\mathcal{E}\big(U\,|\varphi\rangle\,\big) \in \left(\mathbb{C}^2\right)^{\otimes N}$ is the code state associated to $U\,|\varphi\rangle \in \left(\mathbb{C}^2\right)^{\otimes K}$. A circuit used to implement a logical gate is said to be *fault-tolerant* if the error resulting from a faulty location hits a small number of qubits. Finally, the circuit C′ is built in the following way: the qubits of C are encoded using the code $\mathcal{Q}$, each gate U of C is replaced by a fault-tolerant implementation of U^{L} and the errors are regularly corrected using the decoding algorithm.

In ref. [1], the code $\mathcal{Q}$ used to prove the threshold theorem is a *concatenated code*. By definition, given an $[[N_0, 1]]$ code $\mathcal{Q}_0$ and an integer $k \in \mathbb{N}$, the associated concatenated code is an $[[N_0^k, 1]]$ code where each qubit is encoded k times in $\mathcal{Q}_0$. The *space overhead* is defined to be the ratio between the number of qubits in C′ and C; for instance, with fault-tolerance based on concatenated codes, the space overhead is polylogarithmic in the size of C. In this chapter, we present the protocol of ref. [46] where constant space overhead is reached using quantum expander codes (in fact, other families of constant rate LDPC stabilizer codes could be used). The implementation of the logical gates for quantum expander codes is done with the *gate teleportation* trick of ref. [47] where the required ancilla states are prepared using concatenated codes.

In Section 6.1 we provide a formal definition of a circuit and describe the noise model we will use. In Section 6.2 we present the threshold theorem using concatenated codes and in Section 6.3 we explain the protocol of ref. [46].

6.1 Background

A quantum circuit C defines a discrete time computation on classical bits and quantum states. In this model a bit is called a *classical wire*, a qubit is called a *quantum wire* and we will talk about *registers* to refer to a set of wires. For each time step $t \in [\![1; T]\!]$, some of the quantum wires are said to be *inactive*, the other one are said to be *active* which informally means that they are used for the computation. The state of the circuit at time step t is given by the value of the bits stored in the classical wires and by the quantum state stored in the active quantum wires. The time step t is also described by a collection of *locations* applied on wires: some of the inactive wires are associated with a *state preparation location* and each active wire is associated with either a *wait location*, a *gate location* or a *measurement location*.

If a state preparation location is applied on an inactive wire, then the wire is initialized as a $|0\rangle$ state and becomes active at time step $t+1$; otherwise it stays inactive. A wait location does nothing on the wire it acts on, this type of location will be used to model memory errors in noisy circuits. Gate locations represent unitary transformations applied to the quantum state and we will use six types of gate locations associated with the bit-flip gate X, the phase-flip gate Z, the Hadamard gate H, the controlled-not gate C-X (in that case, the location acts on two active wires), the S gate and the T gate:

$$X : \begin{cases} |0\rangle \mapsto |1\rangle \\ |1\rangle \mapsto |0\rangle \end{cases} \qquad Z : \begin{cases} |0\rangle \mapsto |0\rangle \\ |1\rangle \mapsto -|1\rangle \end{cases} \qquad H : \begin{cases} |0\rangle \mapsto \dfrac{1}{\sqrt{2}}\Big(|0\rangle + |1\rangle\Big) \\ |1\rangle \mapsto \dfrac{1}{\sqrt{2}}\Big(|0\rangle - |1\rangle\Big) \end{cases}$$

$$S : \begin{cases} |0\rangle \mapsto e^{-i\pi/4}|0\rangle \\ |1\rangle \mapsto e^{i\pi/4}|1\rangle \end{cases} \qquad T : \begin{cases} |0\rangle \mapsto e^{-i\pi/8}|0\rangle \\ |1\rangle \mapsto e^{i\pi/8}|1\rangle \end{cases} \qquad C\text{-}X : \begin{cases} |00\rangle \mapsto |00\rangle \\ |01\rangle \mapsto |01\rangle \\ |10\rangle \mapsto |11\rangle \\ |11\rangle \mapsto |10\rangle \end{cases}$$

When a measurement location is applied on an active wire, a Z-type measurement is performed on the corresponding qubit, the measurement result is stored in a classical wire and the quantum wire becomes inactive at time step $t+1$. Since the classical wires are not subjected to noise, it is not necessary to use the location formalism to describe their evolution; instead we will explain with words the algorithm we apply on them.

The output of the circuit is given by the value of the classical bits and by the quantum state stored in the active wires at time step $t = T$. We will assume that all the quantum wires are inactive at time step $t = 1$ and we will say that the circuit has classical output when all the quantum wires are inactive at the final time step.

A circuit will often be represented graphically, see the examples given in Figure 6.1. An active quantum wire is represented by a line, a quantum register (multiple active quantum wires) is represented by a thick line and a classical wire is represented by a double line. In Figure 6.1, the controlled-Z gate is defined to be $C\text{-}Z := (\mathbb{1} \otimes H) \circ C\text{-}X \circ (\mathbb{1} \otimes H)$ and the bit-flip acting on a classical wire maps 0 to 1 and 1 to 0.

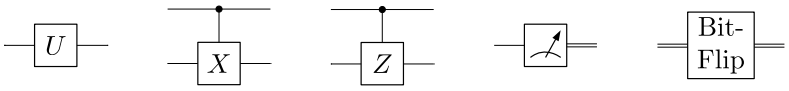

Figure 6.1: representation of a unitary U, a controlled-not gate, a controlled-Z gate, a measurement and a bit-flip.

The error models for circuits are the *local stochastic error model* and the *local stochastic Pauli error model* as defined below [44].

Definition 6.1 (Stochastic error model, local stochastic error model and local stochastic Pauli error model for circuits)**.** Let L be the set of locations of a circuit, then a *stochastic noise model* on L picks a random set of *faulty locations* $F \subseteq L$ and chooses a CPTP map for each faulty location. The CPTP map associated to a location $l \in F$ represents an error applied before or after l:

- A preparation location prepares the wire in the $|0\rangle$ state. When the location is in F, the CPTP map representing the error is applied on the wire once the $|0\rangle$ has been prepared.
- A measurement location measures the wire in the Z-basis and stores the measurement result in a classical wire. When the location is in F, the CPTP map representing the error is applied on the quantum wire before the measurement is performed.
- A gate location applies a unitary gate on the wire. When the location is in F, it applies the CPTP map representing the error after the unitary. Note that the CPTP map acts on two qubits when the gate is a controlled-not gate and it acts on one qubit otherwise.
- A wait location does not modify the quantum state of the circuit. When the location is in F, it applies the CPTP map representing the error on the wire.

A *local stochastic error model* on L with parameter $p_{\text{loc}} \in [0, 1]$ is a stochastic noise model such that the random set of faulty locations F satisfies for all $T \subseteq L$:

$$\mathbb{P}\Big[T \subseteq F\Big] \leq p_{\text{loc}}^{|T|}.$$

A local stochastic error model on L is Pauli when the errors applied on the faulty locations are Pauli matrices.

In the previous chapter, we already used the local stochastic error model in an another context to talk about the error applied on a set of qubits and bits. This error model will be used in this chapter as well, thus we recall the definition below.

Definition 6.2 (Stochastic error model, local stochastic error model and local stochastic Pauli error model for a set of wires)**.** Let V be a set of quantum wires and let C be a set of classical wires, then a *stochastic noise model* on V and C produces an error in the following way:

- A random set of faulty qubits $E \subseteq V$ is picked and a CPTP map $\mathcal{E}_E$ is applied on the qubits belonging to E. The error $\mathcal{E}_E$ can be any channel which maps $|E|$ qubits to $|E|$ qubits.
- A random set of faulty bits $D \subseteq C$ is picked and the bits belonging to D are flipped.

A *local stochastic error model* on V and C with parameter $p_{\text{wire}} \in [0, 1]$ is a stochastic noise model such that the sets E and D satisfy for all $S \subseteq V$ and $T \subseteq C$:

$$\mathbb{P}\Big[S \subseteq E, T \subseteq D\Big] \leq p_{\text{wire}}^{|S|+|T|}.$$

A local stochastic error model on V and C is Pauli when the error applied on the faulty qubits are Pauli matrices.

6.2 Fault-tolerance with poly-logarithmic overhead

The most famous result in fault-tolerant quantum computation is the *threshold theorem* of Aharonov and Ben-Or published in ref. [1]. We quickly present it in Theorem 6.3, see [1, 45] for more details. Informally, if we are able to build gates whose noise is below a threshold $p_{\text{th}} > 0$ (which is a universal constant), then for any circuit D, the threshold theorem provides a circuit D′ with the same output as D but working even with noisy gates.

In Section 6.3, the result of this section will be used as a black-box to prepare the ancilla quantum states needed to perform fault-tolerant quantum computation with constant space overhead.

Theorem 6.3 (Threshold theorem, [1])**.** *There exists a threshold $p_{\text{th}} > 0$ such that the following holds. Let $p_{\text{loc}} < p_{\text{th}}$, let $\delta > 0$ and let* D *be a circuit with classical output, with m qubits, with T time steps and $|\mathrm{D}|$ locations.*
Then there exists another circuit D′ *such that* D′ *has the same output as* D *and fails with probability at most δ when its locations are subjected to a local stochastic noise model with parameter p_{loc}. In addition,* D′ *has m' qubits, T' time steps and $|\mathrm{D}'|$ locations where:*

$$m' = m \operatorname{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right), \quad T' = T \operatorname{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right), \quad |\mathrm{D}'| = |\mathrm{D}| \operatorname{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right).$$

Proof sketch. The circuit D' is constructed from D using usual fault-tolerant techniques based on concatenated codes, see [1] or [45] for a complete description of the procedure. For example, using the $[[7,1]]$ Steane code, we define below a transformation Φ_0 on circuits such that $\Phi_0(\mathrm{D})$ is more robust than D when the gates are noisy. The circuit D' promised by Theorem 6.3 will be defined as $\mathrm{D}' := \Phi_0^k(\mathrm{D})$ for a well chosen k.

In $\Phi_0(\mathrm{D})$, each qubit of the circuit D is replaced by 7 qubits whose state is the encoded version of the original one-qubit state, and each location of D is replaced by the corresponding fault-tolerant gadget followed by a fault-tolerant error correction cycle. The 7-qubit code is particularly pleasant because the gadget corresponding to the Clifford gates are transversal:

$$X^{\mathrm{L}} = X^{\otimes 7}, \quad Z^{\mathrm{L}} = Z^{\otimes 7}, \quad H^{\mathrm{L}} = H^{\otimes 7}, \quad S^{\mathrm{L}} = \big(ZS\big)^{\otimes 7}, \quad C_X^{\mathrm{L}} = (C_X)^{\otimes 7}.$$

In $\Phi_0(\mathrm{D})$, a single error somewhere in a gadget will be corrected by the next error correction cycle. Hence, when the output of the circuit is wrong, two faulty locations happened in the same gadget, an event whose probability is upper bounded by $c_1\, p_{\mathrm{loc}}^2\, |\mathrm{D}|$ where c_1 is a constant. The number of gates and the probability of failure for D, $\Phi(\mathrm{D})$ and $\Phi^k(\mathrm{D})$ are summarized in Figure 6.2 where c_0 and c_1 are two universal constants. The second and third columns in this table show that the number of qubits and the number of gates grow exponentially with k. The fourth column provides the probability for the circuit to have a wrong output when the circuit is subjected to a local stochastic noise with parameter p_{loc}. If $p_{\mathrm{loc}} < p_{\mathrm{th}} := 1/c_1$, then this probability decreases as a double exponential in k.

Circuit	#Qubits	#Gates	$\mathbb{P}$[wrong output]
D	m	$\lvert\mathrm{D}\rvert$	$\leq p_{\mathrm{loc}}\, \lvert\mathrm{D}\rvert$
$\Phi_0(\mathrm{D})$	$7m$	$\leq c_0\, \lvert\mathrm{D}\rvert$	$\leq c_1\, p_{\mathrm{loc}}^2\, \lvert\mathrm{D}\rvert$
$\Phi_0^k(\mathrm{D})$	$7^k m$	$\leq c_0^k\, \lvert\mathrm{D}\rvert$	$\leq \dfrac{(c_1\, p_{\mathrm{loc}})^{2^k}}{c_1}\, \lvert\mathrm{D}\rvert$

Figure 6.2

For a target probability $\delta > 0$, we define $\mathrm{D}' := \Phi_0^k(\mathrm{D})$ where $k = \Theta\left(\log\, \log\, \dfrac{|\mathrm{D}|}{\delta}\right)$ is such that:

$$\frac{(c_1\, p_{\mathrm{loc}})^{2^k}}{c_1}\, |\mathrm{D}| \leq \delta.$$

Finally, the circuit D' fulfills the requirements of Theorem 6.3. □

Theorem 6.3 holds when the output of the circuit D is classical, but in this work, the fault-tolerant protocol based on concatenated codes will be used to create ancilla

quantum states. Let D be a circuit whose output is a quantum state $|\psi\rangle$. When D is subjected to a local stochastic noise, there is a non zero probability that the last location of D is faulty, thus we cannot hope to get $|\psi\rangle$ with high probability. Instead, we can build a fault-tolerant circuit D′ such that when D′ is subjected to a local stochastic noise, the output is $|\psi\rangle$ on which has been applied a local stochastic error in the sense of Definition 6.2.

Theorem 6.4. *There exists a threshold $p_{\rm th} > 0$ such that the following holds. Let $p_{\rm loc} < p_{\rm th}$, let $\delta > 0$ and let* D *be a circuit with m qubits, with T time steps and $|\mathrm{D}|$ locations. We assume that the output of* D *is a quantum state $|\psi\rangle$.*
Then there exists another circuit D′ *whose output is $|\psi\rangle$ and such that when* D′ *is subjected to a local stochastic noise model with parameter $p_{\rm loc}$, there exists $\mathcal{N}$ a local stochastic noise on the qubits of $|\psi\rangle$ with parameter $p_{\rm wire} = c\, p_{\rm loc}$ such that:*

$$\mathbb{P}\Big[\textit{output of } \mathrm{D}' \textit{ is not } \mathcal{N}\big(\,|\psi\rangle\,\big)\Big] \leq \delta.$$

In addition, D′ *has m' qubits, T' time steps and $|\mathrm{D}'|$ locations where:*

$$m' = m\ \mathrm{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right), \quad T' = T\ \mathrm{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right), \quad |\mathrm{D}'| = |\mathrm{D}|\ \mathrm{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right).$$

Proof sketch. Let m_0 be the number of qubits of $|\psi\rangle$ and let Φ_0 be the function defined in the proof of Theorem 6.3. The output of $\Phi_0^k(\mathrm{D})$ is $|\psi\rangle$ encoded in the concatenated code, thus we need to decode the output of $\Phi_0^k(\mathrm{D})$ in a fault-tolerant manner. We fix $\mathcal{E}^{-1}$ some decoding circuit for the Steane code and we denote by $\Phi(\mathrm{D})$ the circuit $\Phi_0(\mathrm{D})$ followed by m_0 copies of $\mathcal{E}^{-1}$, one per block of the Steane code. In particular, the output of $\Phi(\mathrm{D})$ is an m_0-qubit state. Note that when the circuit $\Phi^2(\mathrm{D})$ is created, the transformation Φ_0 is also applied to the m_0 decoding circuits contained in $\Phi(\mathrm{D})$. In other words, the circuit $\Phi^k(\mathrm{D})$ is the circuit $\Phi_0^k(\mathrm{D})$ followed by k layers of decoding: the first layer uses the circuit $\Phi_0^{k-1}(\mathcal{E}^{-1})$, the second layer uses $\Phi_0^{k-2}(\mathcal{E}^{-1})$ and the last layer uses $\mathcal{E}^{-1}$.

For a given $\delta > 0$, we choose again $k = \Theta\left(\log\ \log\ \frac{|\mathrm{D}|}{\delta}\right)$ so that the probability that $\Phi_0^k(\mathrm{D})$ fails is upper bounded by δ. The output of the circuit is equal to $\mathcal{N}_0(|\psi\rangle)$ where $\mathcal{N}_0$ represents the physical noise on the qubits. The local stochastic noise $\mathcal{N}$ promised in Theorem 6.4 is defined to be $\mathcal{N}_0$ except when $\Phi_0^k(\mathrm{D})$ fails:

- If the circuit $\Phi_0^k(\mathrm{D})$ does not fail, the noise $\mathcal{N}$ is equal to $\mathcal{N}_0$.
- If the circuit $\Phi_0^k(\mathrm{D})$ fails then $\mathcal{N}$ applies the identity on the qubits.

□

6.3 Fault-tolerance with constant space overhead

In this section, we describe the protocol of ref. [46] in the particular case where quantum expander codes are used. Given any circuit C with n qubits and size $|C| = \mathrm{poly}(n)$, the fault-tolerant circuit provided by [46] uses $\Theta(n)$ qubits and fails with vanishing probability when n goes to infinity. Contrarily to Theorem 6.3 where a single circuit was considered, we are interested here in the asymptotic behavior for a family of circuits. The main theorem of ref. [46] is very general and could be used with other constant rate LDPC stabilizer codes such as the 4-dimensional hyperbolic code of ref. [51]. However, for adversarial errors with weight up to a fraction of the minimal distance, the known algorithms to decode a 4-dimensional hyperbolic code run in exponential time. Hence, applying ref. [46] with 4-dimensional hyperbolic codes is possible only if classical computation is instantaneous, indeed it would require to run an exponential time classical circuit between some time steps of the fault-tolerant circuit. By contrast, for quantum expander codes, we do not require that classical computation is instantaneous since the small-set-flip decoder runs in constant depth. If we set aside the constant time requirement, there exists an efficient decoder for 4-dimensional hyperbolic codes but this algorithm corrects adversarial errors up to logarithmic weight [55]. As a consequence, for a local stochastic noise, the probability of correction failure decreases as the inverse of a polynomial in the block-length. Hence, when we deal with a circuit of size $|C| = \mathrm{poly}(n)$, the threshold depends on the polynomial rather than being a universal constant. Similarly, we could use 2-dimensional hyperbolic codes, but their minimal distance is logarithmic and thus we cannot start from any polynomial size circuit keeping a threshold which is a universal constant.

Theorem 6.5 (Fault-tolerant quantum computation with constant space overhead [46])**.** *For any asymptotic space overhead $\alpha > 1$, there exists a threshold $p_{\mathrm{th}} > 0$ such that the following holds. Let $p_{\mathrm{loc}} < p_{\mathrm{th}}$ and let* C *be a circuit with classical output, with n qubits, with T time steps and $|\mathrm{C}|$ locations where $|\mathrm{C}| = \mathrm{poly}(n)$.*
Then there exists another circuit C′ *such that* C′ *has the same output as* C *and fails with vanishing probability for large n when subjected to a local stochastic noise model with parameter p_{loc}. In addition,* C′ *has n' qubits where:*

$$n' = \alpha n + o(n).$$

6.3.1 Description of the protocol

The protocol of ref. [46] requires to start from a sequential circuit (a circuit is said to be sequential when at each time step, all but one location are wait locations). In fact, there exists a sequential circuit equivalent to C with n qubits and at most $n|\mathrm{C}| = \mathrm{poly}(n)$ locations. Hence, without loss of generality, C is assumed to be sequential.

To construct the circuit C′, we consider a family of quantum expander codes with rate $R := 2/(1+\alpha)$ and choose the smaller code in this family with parameters $[[N, K]]$ where $N \geq \sqrt{n}$. For simplicity, we assume that K divides n and we split the qubits of C into n/K blocks of size K. In C′, each block of C is replaced by a block of N qubits (called *data qubits*) and each location is replaced by a *simulation cycle* followed

by an *error correction cycle*. During a simulation cycle, the location of C is simulated in C′ using the protocols described in this section. During the error correction cycle, the n/K blocks are corrected in parallel: for each block, the syndrome is measured and the correction inferred by the small-set-flip algorithm is applied on the qubits.

As explained in this section, the number of data qubits in C′ is n/R, the number of ancilla qubits for the syndrome measurements is $(N-K)n/K = n(1/R-1)$ and the number of ancilla qubits for the simulation cycles is $o(n)$. Hence, the space overhead of this protocol is asymptotically equal to $1/R + 1/R - 1 = \alpha$.

Since the performance of quantum expander codes increases with their size, taking codes with bigger block length would be an advantage for error correction. However, we take $N = \Theta(\sqrt{n})$ for convenience in the proofs but $N = n/\mathrm{polylog}(n)$ would lead to constant space overhead as well.

a) Error correction cycle

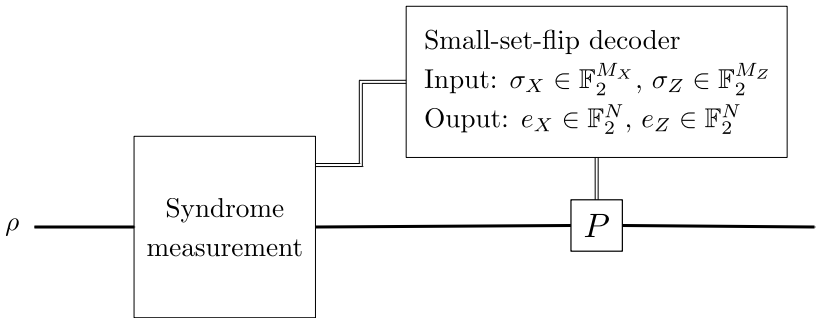

Figure 6.3: Circuit for performing error correction. The Pauli gate P depends on the bit-strings e_X and e_Z.

The error correction is performed with the usual circuit presented in Figure 6.3 where the thick line represents one block of the quantum expander code. There are three main steps in this circuit:

1. The stabilizer generators are measured, for example using the circuit presented in Figure 4.1 (these measurements can be done in a non fault-tolerant way because the generators have constant weight). The measurement result for the Z-type generators is denoted $\sigma_X \in \mathbb{F}_2^{M_X}$ and the measurement result for the X-type generators is denoted $\sigma_Z \in \mathbb{F}_2^{M_Z}$.

2. The small-set-flip decoder is used twice:

   - Once on the input σ_X. The output denoted by $e_X \in \mathbb{F}_2^N$ represents an X-type correction.
   - Once on the input σ_Z. The output denoted by $e_Z \in \mathbb{F}_2^N$ represents a Z-type correction.

Here, it is important to use a constant time decoder since the data qubits are waiting while the decoder is running. If the amount of time for the decoder to run was depending on N, there would be a non-constant number of wait locations between the syndrome measurement and the gate P, leading to a failure with high probability for large N. Hence, the small-set-flip decoder used here is the constant time parallel version analyzed in Section 5.2.4.

3. The Pauli correction P is applied on the data block where:

$$P = P(e_X, e_Z) := \left(\prod_{i \in \mathrm{supp}(e_X)} X_i \right) \left(\prod_{i \in \mathrm{supp}(e_Z)} Z_i \right).$$

All along this section, X_i (resp. Z_i) is the X Pauli matrix (Z Pauli matrix) applied on the i^{th} qubit of the register.

b) Simulation of state preparation

During the simulation of a state preparation location, the goal is to prepare the state $|\psi\rangle$ defined to be the logical $|0\rangle^{\otimes K}$ encoded in the quantum expander code. There exists a circuit D which creates $|\psi\rangle$ using $m = \Theta(N)$ qubits and $|\mathrm{D}| = \mathcal{O}(N^2)$ locations (D is not necessarily fault-tolerant) [43]. Let $\delta = 1/T^2 = 1/\mathrm{poly}(n)$, we apply Theorem 6.4 to D and the resulting circuit D$'$ is used to perform the state preparation in C$'$. The number of qubits in D$'$ is:

$$\begin{aligned} m' &= m \,\mathrm{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right) = m \,\mathrm{polylog}\Big(\mathrm{poly}(m)\,\mathrm{poly}(n)\Big) = m \,\mathrm{polylog}(n) \\ &= \Theta\Big(N \,\mathrm{polylog}(n)\Big). \end{aligned}$$

Hence, since $N = \Theta(\sqrt{n})$, the number of extra qubits needed here is $m' = o(n)$.

c) Simulation of measurement

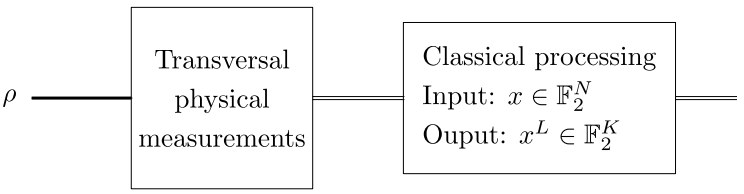

Figure 6.4: circuit for measurement.

To simulate a measurement, we use the circuit of Figure 6.4:

1. Measure the physical qubits on the Z-basis. The output is $z \in \mathbb{F}_2^N$ where for $i \in [\![1; N]\!]$, the bit z_i is the measurement result for the Pauli operator Z_i.

2. The classical processing step computes a bit-string $z^{\mathrm{L}} \in \mathbb{F}_2^K$ where for $k \in [\![1; K]\!]$:

   – The bit z_k^{L} represents the output of a noiseless measurement of the logical Pauli operator Z_k^{L}.

   We already have access to $z \in \mathbb{F}_2^N$, where for $i \in [\![1; N]\!]$:

   – The bit z_i represents the output of a noisy measurement of the physical Pauli operator Z_i.

   To compute z^{L} from z, we will define as an intermediate step the bit-string $z' \in \mathbb{F}_2^N$ such that for $i \in [\![1; N]\!]$:

   – The bit z'_i represents the output of a noiseless measurement of the physical Pauli operator Z_i.

   On the one hand, since Z_k^{L} is equal to a product of Z_i, the bit z_k^{L} is a sum of z'_i. On the other hand, z is a noisy version of z', thus the small-set-flip decoder corrects z to z' with high probability.

   Formally, there are three steps to compute z^{L}:

   2.1 Let $g_1, \ldots, g_{M_X}$ be the Z-type generators of the quantum expander code. We compute the syndrome $\sigma \in \mathbb{F}_2^{M_X}$ where for all $j \in [\![1; M_X]\!]$:

   $$\sigma_j := \bigoplus_{i \in \mathrm{supp}(g_j)} z_i.$$

   2.2 We run the small-set-flip algorithm on input σ and denote by $\hat{E} \in \mathbb{F}_2^N$ the decoder output. The bit-string z' is defined by:

   $$z' := z \oplus \hat{E}.$$

   In order to get the right value for z', we need to correct entirely the error on z. In addition, the syndrome σ has not been measured with noisy gates, thus the bits of σ are not subjected to noise. Hence, the small-set-flip algorithm used here is the logarithmic time decoder described in Section 5.2.5 and analyzed in Corollary 5.37.

   2.3 We compute $z^{\mathrm{L}} \in \mathbb{F}_2^K$, where for all $k \in [\![1; K]\!]$:

   $$z_k^{\mathrm{L}} := \bigoplus_{i \in \mathrm{supp}(Z_k^{\mathrm{L}})} z'_i.$$

As a conclusion, the bit-string z^{L} has been computed and the time complexity of the overall procedure is logarithmic in N. Pending the procedure completion, the other active wires in the circuit C' are corrected with the constant time small-set-flip algorithm.

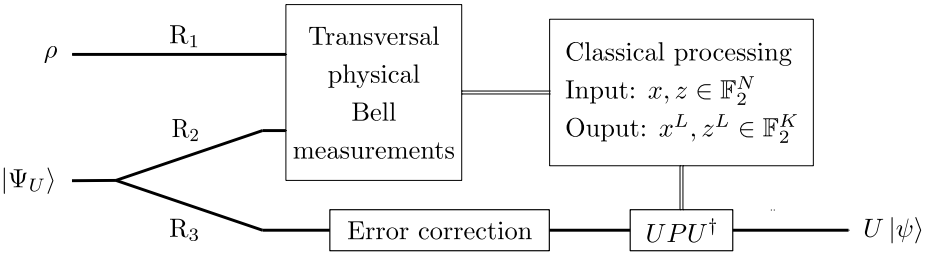

Figure 6.5: gate teleportation circuit. R_1, R_2 and R_3 are the name of three quantum registers. The Pauli gate P depends on the value of x^{L} and z^{L}.

d) Simulation of gate

When the location in C is a gate location, the goal of the simulation cycle is to apply the corresponding logical gate in C′.

To apply a logical gate U in C′, we use the teleportation circuit presented in Figure 6.5. At the beginning, the register R_1 contains the data qubits and registers R_2 and R_3 contain an ancilla state $|\Psi_U\rangle$ defined by:

$$|\Psi_U\rangle := \left(\mathbb{1}_{\mathrm{R}_2} \otimes U_{\mathrm{R}_3}\right) \mathcal{E}^{\otimes 2} \left[\left(|00\rangle_{\mathrm{R}_2,\mathrm{R}_3} + |11\rangle_{\mathrm{R}_2,\mathrm{R}_3} \right)^{\otimes K} \right],$$

where $\mathcal{E}$ is the encoding map associated to the quantum expander code. In other words, the state $|\Psi_U\rangle$ matches the following description:

- K EPR pairs are shared between R_2 and R_3.
- R_2 and R_3 are both encoded in the quantum expander code.
- The unitary U is applied on R_3.

There exists a circuit D which creates $|\Psi_U\rangle$ using $m = \Theta(N)$ qubits and $|\mathrm{D}| = \mathcal{O}(N^2)$ locations (D is not necessarily fault-tolerant). In C′, the state $|\Psi_U\rangle$ is prepared with the fault-tolerant circuit D′ we get when we apply Theorem 6.4 to D with $\delta = 1/T^2 = 1/\mathrm{poly}(n)$. The number of qubits in D′ is:

$$\begin{aligned} m' &= m \,\mathrm{polylog}\left(\frac{|\mathrm{D}|}{\delta}\right) = m \,\mathrm{polylog}\Big(\mathrm{poly}(m)\,\mathrm{poly}(n)\Big) = m \,\mathrm{polylog}(n) \\ &= \Theta\Big(N \,\mathrm{polylog}(n)\Big). \end{aligned}$$

Hence, since $N = \Theta(\sqrt{n})$, the number of extra qubits needed here is $m' = o(n)$.

Once the ancilla state $|\Psi_U\rangle$ has been created, the circuit of Figure 6.5 is done in the following way:

1. Perform a physical Bell measurement on registers R_1 and R_2, the outputs are denoted $x, z \in \mathbb{F}_2^N$. In more details, for $i \in [\![1; N]\!]$, x_i is the bit we get when the Pauli operator $X_{i,\mathrm{R}_1} \otimes X_{i,\mathrm{R}_2}$ is measured and z_i is the bit we get when the Pauli operator $Z_{i,\mathrm{R}_1} \otimes Z_{i,\mathrm{R}_2}$ is measured.

2. The classical processing step computes a bit-string $x^{\mathrm{L}} \in \mathbb{F}_2^K$ using $x \in \mathbb{F}_2^N$ and a bit-string $z^{\mathrm{L}} \in \mathbb{F}_2^K$ using $z \in \mathbb{F}_2^N$. Similarly to the procedure used for simulating a measurement, we define $x' \in \mathbb{F}_2^N$ and $z' \in \mathbb{F}_2^N$ such that for all $i \in [\![1; N]\!]$ and $k \in [\![1; K]\!]$:
   - The bit x_i is the output of a noisy measurement of $X_{i,\mathrm{R}_1} \otimes X_{i,\mathrm{R}_2}$.
   - The bit x'_i is the output of a noiseless measurement of $X_{i,\mathrm{R}_1} \otimes X_{i,\mathrm{R}_2}$.
   - The bit x^{L}_k is the output of a noiseless measurement of $X^{\mathrm{L}}_{k,\mathrm{R}_1} \otimes X^{\mathrm{L}}_{k,\mathrm{R}_2}$.
   - The bit z_i is the output of a noisy measurement of $Z_{i,\mathrm{R}_1} \otimes Z_{i,\mathrm{R}_2}$.
   - The bit z'_i is the output of a noiseless measurement of $Z_{i,\mathrm{R}_1} \otimes Z_{i,\mathrm{R}_2}$.
   - The bit z^{L}_k is the output of a noiseless measurement of $Z^{\mathrm{L}}_{k,\mathrm{R}_1} \otimes Z^{\mathrm{L}}_{k,\mathrm{R}_2}$.

   As described in more details for the case of a measurement simulation, we correct x (resp. z) using the logarithmic time small-set-flip algorithm of Section 5.2.5 to get x' (resp. z'). The bit-strings x^{L} and z^{L} are defined for all $k \in [\![1; K]\!]$ by:

$$x^{\mathrm{L}}_k := \bigoplus_{i \in \mathrm{supp}(X^{\mathrm{L}}_k)} x'_i, \qquad z^{\mathrm{L}}_k := \bigoplus_{i \in \mathrm{supp}(Z^{\mathrm{L}}_k)} z'_i.$$

   Since we use the small-set-flip algorithm of Section 5.2.5, the time complexity of the classical processing is logarithmic in N.

3. While the classical processing is running, the state in register R_3 is continuously corrected with the constant time small-set-flip decoder of Section 5.2.4.

4. Apply $UPU^\dagger$ on register R_3 where:

$$P = P(x^{\mathrm{L}}, z^{\mathrm{L}}) := \left(\prod_{k \in \mathrm{supp}(z^{\mathrm{L}})} X^{\mathrm{L}}_k \right) \left(\prod_{k \in \mathrm{supp}(x^{\mathrm{L}})} Z^{\mathrm{L}}_k \right).$$

   When the gate U is a logical Clifford, the unitary $UPU^\dagger$ is a logical Pauli which can be applied transversally. More generally, when U is in the i^{th} level of the Clifford hierarchy, $UPU^\dagger$ belongs to the $(i-1)^{\text{th}}$ level of the Clifford hierarchy.

The procedure described above is called *gate teleportation* [47] and can be used to apply a universal logical gate set on C':

- A logical Pauli is applied in C' with a circuit made of transversal Pauli gates.
- A logical Hadamard, a logical controlled-not or a logical S gate is applied with the gate teleportation protocol. In that case, the gate U is a logical Clifford gate and thus $UPU^\dagger$ is a logical Pauli that we apply transversally. For a controlled-not gate between two different code blocks, more registers are involved but a similar procedure works.
- A logical T gate is also implemented with gate teleportation. In that case, the unitary $UPU^\dagger$ is a logical Clifford and is applied with a second gate teleportation protocol.

6.3.2 Behavior against local stochastic Pauli noise

In this section, we show that if the locations of the circuit C' are subjected to a local stochastic Pauli noise, then the classical output of C' is the expected output with high probability. When a circuit is run with Pauli errors applied on a set of faulty locations, we will say that we have run a *faulty realization* of the circuit.

As we explain below, a faulty realization of C' is equivalent to running C' without faulty locations but where Pauli errors are applied on the data qubits after the simulation cycles, and where some measurement results are flipped before being used. Similarly, let D_1 be one of the circuits presented in Section 6.3 (Figure 6.3 or Figure 6.4 or Figure 6.5), then applying Pauli errors at faulty locations is equivalent to flipping some classical bits after the measurements, and to applying Pauli errors on the data qubits at the end of D_1. For instance, in the circuit of Figure 4.1 used to measure a Pauli operator, the Hadamard gates and the controlled-P gate are invertible Clifford operations. Hence, a faulty realization of this circuit is equivalent to the circuit presented in Figure 6.6. In addition, as shown in Figure 6.7, an X or Y gate followed by a measurement is equivalent to a measurement followed by a bit-flip, and a Z gate followed by a measurement is equivalent to a measurement. By the same arguments:

- A faulty realization of the error correction circuit of Figure 6.3 is equivalent to the circuit of Figure 6.8.
- A faulty realization of the measurement circuit of Figure 6.4 is equivalent to the circuit of Figure 6.9.
- A faulty realization of the gate teleportation circuit of Figure 6.5 is equivalent to the circuit of Figure 6.10.

Let D_1 be one of the circuits presented in Section 6.3 (Figure 6.3 or Figure 6.4 or Figure 6.5) and let D_2 be the equivalent circuit defined above (Figure 6.8 or Figure 6.9 or Figure 6.10). Using the previous points, running D_1 with a local stochastic Pauli noise on the locations is equivalent to run D_2 where the boxes "Some bits are flipped" represent a local stochastic noise on the classical bits and the boxes "Pauli gates" represent a local stochastic Pauli noise on the data qubits. Let p_{loc} be the parameter for the local stochastic noise on the locations of D_1, let p_{wire} be the parameter for the local stochastic noise on the data qubits and bits of D_2, and let $p_4 := \min(p_1, p_2, p_3)$ where p_1, p_2 and p_3 are the thresholds for the quantum expander codes as defined in Theorem 5.30 and Theorem 5.36. For this analysis to work, the inequality $p_{\mathrm{wire}} < p_4/3$ is required, thus we show below that p_{wire} can be made as small as desired by lowering p_{loc}.

An error on a data qubit q at the end of D_2 is necessarily the consequence of an error on a location which is *above* q in the circuit (*i.e.* a location which is linked to q by wires). Let L be the set of locations in D_1 and let V be the set of data qubits at the end of D_2. For a given set $S \subset V$, we denote by $A(S) \subseteq L$ the set of locations which are above at least one qubit of S. We have $|A(S)| \leq c_1|S|$ where c_1 is a constant and each location is above at most c_2 data qubits. Given a set of faulty locations $F \subseteq L$, the corresponding set of faulty data qubits $E \subseteq V$ satisfies:

$$E \subseteq \Big\{ q \in V : F \cap A(q) \neq \varnothing \Big\}$$

Thus for all $S \subseteq V$:

$$\begin{aligned}
\mathbb{P}\Big[S \subseteq E\Big] &\leq \mathbb{P}\Big[\exists R \subseteq A(S) : |R| \geq |S|/c_2, R \subseteq F\Big] \\
&\leq \sum_{r \geq |S|/c_2} \left[\sum_{R \subseteq A(S):|R|=r} \mathbb{P}\Big[R \subseteq F\Big] \right] \\
&\leq \sum_{r \geq |S|/c_2} \binom{|A(S)|}{r} p_{\text{loc}}^r \\
&\leq 2^{c_1|S|} \sum_{r \geq |S|/c_2} p_{\text{loc}}^r \\
&\leq K 2^{c_1|S|} p_{\text{loc}}^{|S|/c_2}.
\end{aligned}$$

For p_{loc} sufficiently small, we have $\mathbb{P}[S \subseteq E] \leq (p_4/4)^{|S|}$ and thus F is local stochastic with parameter $p_{\text{wire}} < p_4/4$.

Let $t \in [\![1; T]\!]$ be a time step of the initial circuit C and let $|\psi_t\rangle$ be the state of the circuit C$'$ after the t^{th} error correction cycle under the hypothesis that the locations are not faulty. We are going to build P_t a local stochastic noise on the qubits of $|\psi_t\rangle$ with parameter $p_4/2$ such that when the locations of C$'$ are subjected to a local stochastic noise with parameter p_{loc}, the state after the t^{th} error correction cycle is $P_t |\psi_t\rangle$ with high probability.

To analyze the error correction cycles described in Figure 6.3 and Figure 6.8, we use Theorem 5.30 about the constant time small-set-flip algorithm. If p_{loc} is sufficiently small, then the error on the data qubits and the syndrome bits are local stochastic with parameter sufficiently small to apply Theorem 5.30. In this theorem, c is chosen such that $p_{\text{ls}} < p_4/4$, then there is a small probability for this circuit to fail and otherwise the data qubits at the end are subjected to a local stochastic noise with parameter $p_4/2$. Formally:

(1) Let $|\psi\rangle$ be a code state of the quantum expander code. We run the error correction circuit D_1 of Figure 6.3 with the following hypothesis:

- The input of the circuit is $P_1 |\psi\rangle$ where P_1 is a local stochastic Pauli noise with parameter $< p_4$.
- The locations of D_1 are subjected to a local stochastic Pauli noise with parameter p_{loc}.

Then there exists P a local stochastic Pauli noise with parameter $< p_4/2$ such that:

$$\mathbb{P}\Big[\text{ The output of } \text{D}_1 \text{ is } P|\psi\rangle\Big] \geq 1 - \frac{1}{e^{\Theta(\sqrt{N})}}.$$

When circuit C$'$ simulates a measurement with Figure 6.4, the small-set-flip algorithm with logarithmic time is used. If p_{loc} is sufficiently small, the error on the bits

after the transversal measurement are local stochastic with parameter sufficiently small to apply Corollary 5.37. Hence the classical output of the circuit is the right one with high probability:

(2) Let $|\psi\rangle$ be a code state of the quantum expander code. We run the measurement circuit D_1 of Figure 6.4 with the following hypothesis:

- The input of the circuit is $P_1 |\psi\rangle$ where P_1 is a local stochastic Pauli noise with parameter $< p_4/2$.
- The locations of D_1 are subjected to a local stochastic Pauli noise with parameter p_{loc}.

Then:

$$\mathbb{P}\Big[\ \mathrm{D}_1 \text{ has the right output}\Big] \geq 1 - \frac{1}{e^{\Theta(\sqrt{N})}}.$$

Finally, when the gate teleportation protocol of Figure 6.5 is performed in C', we use Theorem 5.30 and Corollary 5.37 for the small-set-flip decoder:

(3) Let $|\psi\rangle$ be a code state of the quantum expander code and let U be a logical gate. We apply U with the gate teleportation circuit D_1 of Figure 6.5 with the following hypothesis:

- The data qubits at the entrance of D_1 are in the state $P_1 |\psi\rangle$ where P_1 is a local stochastic Pauli noise with parameter $< p_4/2$.
- The ancilla state is prepared in the state $P_2 |\Psi_U\rangle$ where P_2 is a local stochastic Pauli noise with parameter $< p_4/4$.
- The locations of D_1 are subjected to a local stochastic Pauli noise with parameter p_{loc}.

Then there exists P a local stochastic Pauli noise with parameter $< p_4$ such that:

$$\mathbb{P}\Big[\ \text{The output of } \mathrm{D}_1 \text{ is } PU |\psi\rangle\Big] \geq 1 - \frac{\mathrm{polylog}(N)}{e^{\Theta(\sqrt{N})}}.$$

We are now ready to conclude. We run the entire circuit C' with the following hypothesis:

- The locations of D_1 are subjected to a local stochastic Pauli noise with parameter p_{loc}.
- The ancilla qubits for gate teleportations are prepared in the state $P_1 |\Psi_U\rangle$ where P_1 is a local stochastic Pauli noise with parameter $< p_4/4$.
- The state preparation cycles prepare the state $P_2 |\psi\rangle$ where $|\psi\rangle$ is the logical zero state and P_2 is a local stochastic Pauli noise with parameter $< p_4$.

Let $|\psi_t\rangle$ be the state of the circuit C′ after the t^{th} error correction cycle when there is no faulty location. Using the properties (1), (2) and (3) above, we can perform an induction on t to show the existence of a Pauli error P_t with local stochastic parameter $< p_4/2$ such that:

$$\mathbb{P}\Big[\text{After the } t^{\text{th}} \text{ error correction cycle, the state is } P_t\,|\psi_t\rangle\Big] \geq 1 - \frac{t\,\text{poly}(N)}{e^{\Theta(\sqrt{N})}}.$$

After the final measurement:

$$\mathbb{P}\Big[\text{The output of C}' \text{ is right}\Big] \geq 1 - \frac{T\,\text{poly}(N)}{e^{\Theta(\sqrt{N})}} = 1 - o(1).$$

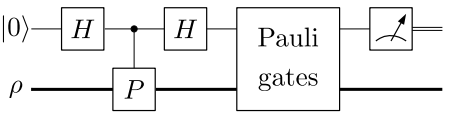

Figure 6.6: this circuit is equivalent to a faulty realization of Figure 4.1.

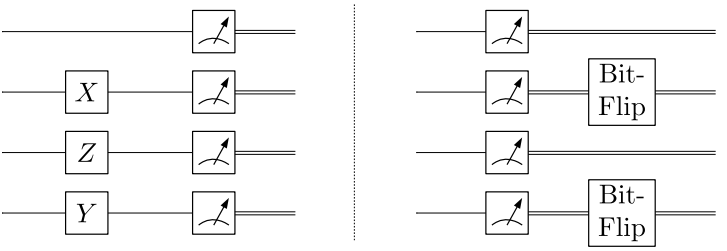

Figure 6.7: two equivalent circuits.

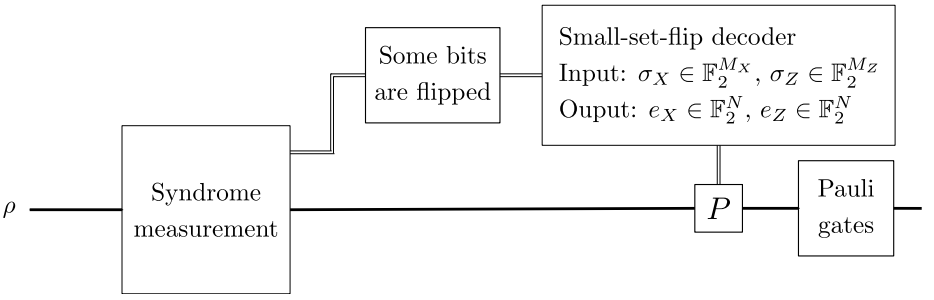

Figure 6.8: this circuit is equivalent to a faulty realization of Figure 6.3.

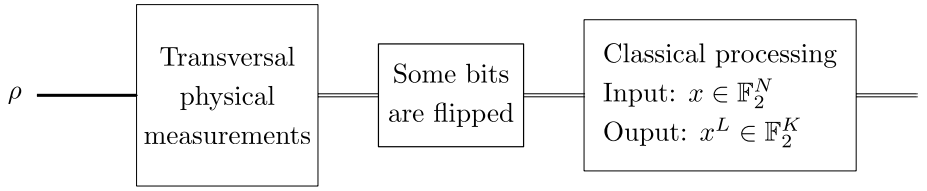

Figure 6.9: this circuit is equivalent to a faulty realization of Figure 6.4.

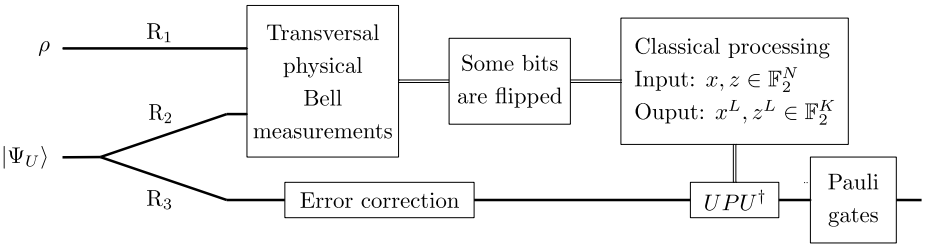

Figure 6.10: this circuit is equivalent to a faulty realization of Figure 6.5.

6.3.3 Behavior against local stochastic noise

We are now ready to give a proof sketch for Theorem 6.5. As described in Section 6.3.1, we use the fault-tolerant protocol with concatenated codes as a black-box for state preparation (the goal is to prepare either a logical zero or an ancilla state for gate teleportation). This protocol fails with probability at most $1/T^2$ and is used at most T times, thus the states are successfully prepared with probability $1-o(1)$. By Theorem 6.4, when a state preparation does not fail, the noise on the prepared state is local stochastic with parameter $c\, p_{\text{loc}}$. In what follows, we assume that p_{loc} is sufficient small to ensure $c\, p_{\text{loc}} < p_4/4$. Hence, similarly to the hypotheses in Section 6.3.2, the qubits of the prepared states are subjected to a local stochastic noise with parameter $< p_4/4$.

We assume that the locations of the circuit C' are subjected to a noise $\mathcal{N}$ with local stochastic parameter p_{loc}. Let L be the set of locations and let $\mathcal{L}$ be the power set of L, then the support of the location error is a set $F \in \mathcal{L}$. We say that $F \in \mathcal{L}$ is *problematic* when we can make C' fail by applying Pauli errors on the locations of F. Let $\mathcal{L}^* \subseteq \mathcal{L}$ be the set of error supports which are not problematic. We define a Pauli noise $\mathcal{P}$ with local stochastic parameter p_{loc} in the following way:

- The set of faulty locations F is chosen with the same probability distribution as $\mathcal{N}$.
- If $F \notin \mathcal{L}^*$, then $\mathcal{P}$ applies the problematic Pauli error associated to F.
- If $F \in \mathcal{L}^*$, then $\mathcal{P}$ applies an arbitrary Pauli on the faulty locations (for example the identity).

The local stochastic Pauli noise $\mathcal{P}$ has parameter p_{loc}. By Section 6.3.2, when the

locations are subjected to $\mathcal{P}$, the circuit C' has the right output with probability at least $1 - o(1)$ thus:

$$\mathbb{P}\Big[F \in \mathcal{L}^*\Big] \geq 1 - o(1).$$

We purify the circuit C' and use the noise $\mathcal{N}$ on the locations. If the error support satisfies $F \in \mathcal{L}^*$, then we can write the error applied by $\mathcal{N}$ as a sum of Pauli errors whose support are included in F. By definition of $\mathcal{L}^*$, each term of the sum is not problematic for C', thus the circuit C' has the right output by linearity. As a conclusion, when the circuit C' is subjected to a local stochastic noise with parameter p_{loc}, the output is the right one with probability $1 - o(1)$.

Chapter 7

Deferred proofs

The goal of this chapter is to compute the dimension of the classical product code that we used in Section 4.2 to compute the dimension of hypergraph product codes. In Section 7.1 we introduce the systematic form for classical codes and use it in Section 7.2 to compute the dimension of classical product codes.

7.1 Systematic form

In this section, we define the concept of *systematic form* which is a standard notion in classical error correction and will be used for the analysis of the classical product code. The terminology "systematic form" can be applied either to a code, to a generator matrix or to a parity check matrix .

Let $\mathcal{M}_{a,b}(\mathbb{F}_2)$ be the set of $a \times b$ binary matrices. A linear code is in systematic form if there exists $A \in \mathcal{M}_{n-k,k}(\mathbb{F}_2)$ such that the generator matrix $G \in \mathcal{M}_{n,k}(\mathbb{F}_2)$ is equal to the block matrix:

$$G = \begin{pmatrix} \mathbb{1}_k \\ A \end{pmatrix} \tag{7.1}$$

where $\mathbb{1}_k \in \mathcal{M}_{k,k}(\mathbb{F}_2)$ is the identity matrix. As illustrated in Figure 7.1, the first bits of a codeword $x = Gs$ are equal to the bits of s.

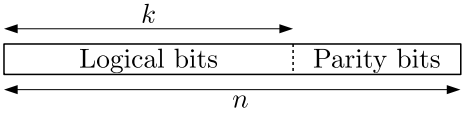

Figure 7.1: codeword of an $[n, k]$ systematic code.

Thereby, the codeword associated to s is the concatenation of s (the logical bits) and As (the parity bits).

Furthermore, let $H \in \mathcal{M}_{n-k,n}(\mathbb{F}_2)$ be the following parity check matrix defined by block:

$$H = \begin{pmatrix} A & \mathbb{1}_{n-k} \end{pmatrix}. \tag{7.2}$$

We have

$$HG = A \oplus A = 0$$

thus $\mathrm{span}(G) \subseteq \ker(H)$. Moreover, by the rank-nullity theorem $\dim(\ker H) = k = \dim(\mathrm{span}(G))$ thus $\mathrm{span}(G) = \ker(H)$ and the codes defined by H and G are equal. A generator matrix which has the form of eq. (7.1) and a parity check matrix which has the form of eq. (7.2) are said to be in systematic form.

The main property presented in this section is Proposition 7.1: up to permutation on the bits, any error correcting code admits a parity check matrix in systematic form.

Proposition 7.1. *Let $H \in \mathcal{M}_{m,n}(\mathbb{F}_2)$ be a matrix with rank $n-k$ then there exists an invertible matrix $Q \in \mathcal{M}_{m,m}(\mathbb{F}_2)$, a permutation matrix $P \in \mathcal{M}_{n,n}(\mathbb{F}_2)$ and a matrix $A \in \mathcal{M}_{n-k,k}(\mathbb{F}_2)$ such that:*

$$QHP = \begin{pmatrix} A & \mathbb{1}_{n-k} \\ 0_{m-n+k,n} \end{pmatrix}.$$

Before giving the proof of Proposition 7.1, we remark that:

- If $\mathcal{C}$ is the code with parity check matrix H then the block matrix $\begin{pmatrix} A & \mathbb{1}_{n-k} \end{pmatrix}$ is a parity check matrix for the code $\mathcal{C}' = \{P^{-1}x : x \in \mathcal{C}\}$ where the bits have been permuted.

- Similarly, for any generator matrix $G \in \mathcal{M}_{n,k}(\mathbb{F}_2)$, there exists an invertible matrix $Q \in \mathcal{M}_{k,k}(\mathbb{F}_2)$, a permutation matrix $P \in \mathcal{M}_{n,n}(\mathbb{F}_2)$ and a matrix $A \in \mathcal{M}_{n-k,k}(\mathbb{F}_2)$ such that:

  $$PGQ = \begin{pmatrix} \mathbb{1}_k \\ A \end{pmatrix}$$

  and PGQ is a generator matrix for $\mathcal{C}' = \{P^{-1}x : x \in \mathcal{C}\}$.

Proof. First, we remark that $\ker H = \ker QH$ holds for any invertible matrix $Q \in \mathcal{M}_{m,m}(\mathbb{F}_2)$, *i.e.* we can apply invertible operations on the rows of H without changing the corresponding error correcting code. Moreover, in the statement of Proposition 7.1, H is multiplied on the right by the permutation matrix P, meaning that we allow the operation consisting in permuting the columns of H.
Indeed, we show that using three invertible operations on H, we can turn it into the form promised in Proposition 7.1. The three operations are the following:

– Permuting the rows of the matrix.

- Permuting the columns of the matrix.
- Replacing row number i by the sum of row number i and row number $j \neq i$.

Using these three operations, we perform a Gaussian elimination on the matrix H in order to turn it into the desired form. First, up to a permutation on the columns of the matrix, it is equivalent to show that H can be turned into the matrix:

$$\begin{pmatrix} \mathbb{1}_{n-k} & A \\ 0_{m-n+k,n} \end{pmatrix}.$$

The idea is to transform the columns of H one by one starting from the first one. Formally, we show by induction on $l \in [\![0; n-k]\!]$ that using the three operations we can turn H into the following form:

$$H_l = \begin{pmatrix} \mathbb{1}_l & A_l \\ 0_{m-l,l} & B_l \end{pmatrix}$$

where $A_l \in \mathcal{M}_{l,n-l}(\mathbb{F}_2)$ and $B_l \in \mathcal{M}_{m-l,n-l}(\mathbb{F}_2)$.
The initialization of the induction for $l = 0$ is done with $B_0 = H$ and the other matrices being empty.
For the induction step, the rank of H and the rank of H_l are equal to $n-k$ thus, while $l < n-k$, there is a 1 somewhere in the matrix B_l. We put this 1 in the first entry of B_l using a permutation on the rows and the columns. In order to get the matrix H_{l+1}, we keep the 1 at position $(l+1, l+1)$ and delete all the other 1 in column $l+1$ by replacing for each $i \neq l+1$ the row i by the sum of row i and row $l+1$.
Finally, when $l = n-k$, because the matrices H and H_l have rank $n-k$, we have $B_l = 0_{m-l,n-l}$ and we get Proposition 7.1. □

7.2 Definition of classical product code

In this section we define a class of codes obtained by taking the product of two codes. Note that given two classical error correcting codes, two kinds of product appear in this PhD thesis: "the product code" which is a classical code and the "hypergraph product code" which is a quantum code. The product code construction is used to determine the parameters of the hypergraph product codes. For more details about the classical product code, see [35, 99].

Let $\mathcal{C}_1$ be an $[n_1, k_1, d_1]$ code and let $\mathcal{C}_2$ be an $[n_2, k_2, d_2]$ code then the product code $\mathcal{C}_1 \otimes \mathcal{C}_2$ is the code whose parity check matrix H is defined by:

$$H = \begin{pmatrix} \mathbb{1}_{n_1} \otimes H_2 \\ H_1 \otimes \mathbb{1}_{n_2} \end{pmatrix}$$

Note that the block-length of $\mathcal{C}_1 \otimes \mathcal{C}_2$ is the number of columns of H and is equal to $n_1 n_2$. In the case of a product code, it is relevant to represent a set of $n_1 n_2$ bits as a binary matrix with n_1 rows and n_2 columns. With this representation in mind, a matrix

$c \in \mathcal{M}_{n_1,n_2}(\mathbb{F}_2)$ is a codeword of $\mathcal{C}_1 \otimes \mathcal{C}_2$ if and only if each row of c is a codeword of $\mathcal{C}_2$ and each column of c is a codeword of $\mathcal{C}_1$ (see Figure 7.2).

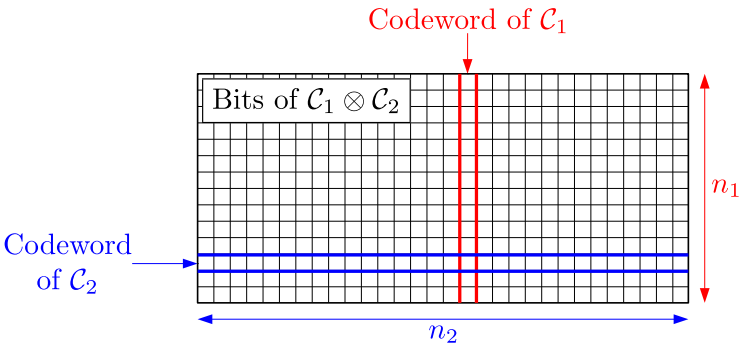

Figure 7.2: schematic representation of a product code.

Proposition 7.2. *The dimension of $\mathcal{C}_1 \otimes \mathcal{C}_2$ is equal to $k_1 k_2$.*

Proof. We reduce the general problem to the particular case where H_1 and H_2 are in systematic form (see Section 7.1).

By Proposition 7.1, there exist two invertible matrices $Q_1 \in \mathcal{M}_{m_1,m_1}(\mathbb{F}_2)$, $Q_2 \in \mathcal{M}_{m_2,m_2}(\mathbb{F}_2)$, two permutation matrices $P_1 \in \mathcal{M}_{n_1,n_1}(\mathbb{F}_2)$, $P_2 \in \mathcal{M}_{n_2,n_2}(\mathbb{F}_2)$ and two matrices $A_1 \in \mathcal{M}_{n_1-k_1,k_1}(\mathbb{F}_2)$, $A_2 \in \mathcal{M}_{n_2-k_2,k_2}(\mathbb{F}_2)$ such that:

$$Q_1 H_1 P_1 = H_1' := \begin{pmatrix} A_1 & \mathbb{1}_{n_1-k_1} \\ 0_{m_1-n_1+k_1,n_1} \end{pmatrix} \text{ and } Q_2 H_2 P_2 = H_2' := \begin{pmatrix} A_2 & \mathbb{1}_{n_2-k_2} \\ 0_{m_2-n_2+k_2,n_2} \end{pmatrix} \tag{7.3}$$

let H be the parity check matrix of $\mathcal{C}_1 \otimes \mathcal{C}_2$, let $\mathcal{C}_1'$, $\mathcal{C}_2'$ be the codes with party checks matrices H_1', H_2' and let H' be the parity check matrix of $\mathcal{C}_1' \otimes \mathcal{C}_2'$:

$$H = \begin{pmatrix} \mathbb{1}_{n_1} \otimes H_2 \\ H_1 \otimes \mathbb{1}_{n_2} \end{pmatrix} \qquad H' = \begin{pmatrix} \mathbb{1}_{n_1} \otimes H_2' \\ H_1' \otimes \mathbb{1}_{n_2} \end{pmatrix}$$

then using the product of block matrices, we can check that $H' = QHP$ where Q is an invertible matrix and P is a permutation matrix defined as the block matrices:

$$Q = \begin{pmatrix} P_1^{-1} \otimes Q_2 & 0_{n_1 m_2, m_1 n_2} \\ 0_{m_1 n_2, n_1 m_2} & Q_1 \otimes P_2^{-1} \end{pmatrix} \quad \text{and} \quad P = P_1 \otimes P_2.$$

In particular, we have:

$$\begin{aligned} &\dim(\mathcal{C}_1') = \dim(\mathcal{C}_1) = k_1 && \text{because} && \ker(H_1') = P_1^{-1} \ker H_1, \\ &\dim(\mathcal{C}_2') = \dim(\mathcal{C}_2) = k_2 && \text{because} && \ker(H_2') = P_2^{-1} \ker H_2, \\ &\dim(\mathcal{C}_1 \otimes \mathcal{C}_2) = \dim(\mathcal{C}_1' \otimes \mathcal{C}_2') && \text{because} && \ker(H') = P^{-1} \ker H. \end{aligned}$$

Hence, we have reduced the problem to the case where H'_1 and H'_2 are in systematic form and it remains to show $\dim(\mathcal{C}'_1 \otimes \mathcal{C}'_2) = k_1 k_2$.
The matrix H' is not necessarily in systematic form but we can anyway show that if we fix a set of $k_1 k_2$ logical bits as shown in Figure 7.3 then there is a unique possibility for the parity bits. Formally, for any $s \in \mathcal{M}_{k_1,k_2}(\mathbb{F}_2)$, there exists a unique $x \in \mathcal{C}'_1 \otimes \mathcal{C}'_2$ such that s is the upper left part of x.

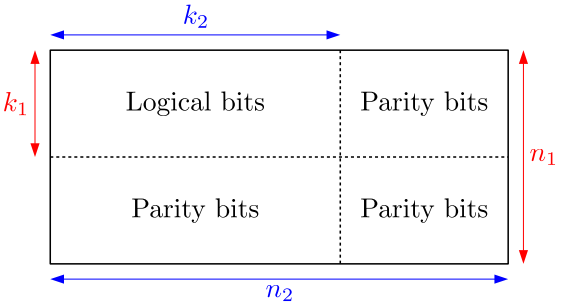

Figure 7.3: product code when the parity check matrices are in systematic form.

Using eq. (7.3), for any $x_1 \in \mathbb{F}_2^{n_1}$ and $x_2 \in \mathbb{F}_2^{n_2}$, we have:

$$x_1 \in \mathcal{C}'_1 \Leftrightarrow \forall i \in [\![1; n_1 - k_1]\!] : x_1(i + k_1) = \sum_{l=1}^{k_1} H'_1(i,l) x_1(l),$$

$$x_2 \in \mathcal{C}'_2 \Leftrightarrow \forall j \in [\![1; n_2 - k_2]\!] : x_2(j + k_2) = \sum_{l=1}^{k_2} H'_2(j,l) x_2(l).$$

If we fix some $s \in \mathcal{M}_{k_1,k_2}(\mathbb{F}_2)$ and try to find $x \in \mathcal{C}'_1 \otimes \mathcal{C}'_2$ whose upper left part is equal to s then the only possible choice is to use the following formulas:

$$\begin{cases} \forall i \in [\![1; k_1]\!] \\ \forall j \in [\![1; k_2]\!] \end{cases} \quad x(i,j) = s(i,j),$$

$$\begin{cases} \forall i \in [\![1; k_1 - n_1]\!] \\ \forall j \in [\![1; k_2]\!] \end{cases} \quad x(i + k_1, j) = \sum_{l=1}^{k_1} H'_1(i,l) s(l,j),$$

$$\begin{cases} \forall i \in [\![1; k_1]\!] \\ \forall j \in [\![1; n_2 - k_2]\!] \end{cases} \quad x(i, j + k_2) = \sum_{l=1}^{k_2} H'_2(j,l) s(i,l),$$

$$\begin{cases} \forall i \in [\![1; n_1 - k_1]\!] \\ \forall j \in [\![1; n_2 - k_2]\!] \end{cases} \quad x(i + k_1, j + k_2) = \sum_{l_1=1}^{k_1} \sum_{l_2=1}^{k_2} H'_1(i,l_1) H'_2(j,l_2) s(l_1,l_2).$$

Informally, the parity bits of the lower left part are uniquely determined by the column conditions and the parity bits of the upper right part are uniquely determined by the row conditions. Moreover, the parity bits of the lower right part can be determined either

using the row condition on the parity bits of the lower left part or by using the column condition on the parity bits of the upper right part. Note that both methods give the same result.
The conclusion is that x is uniquely determined by s and thus $\mathcal{C}'_1 \otimes \mathcal{C}'_2$ has dimension $k_1 k_2$. □

Remark 7.3. Let G_1 and G_2 be the generator matrices of $\mathcal{C}_1$ and $\mathcal{C}_2$ then $H(G_1 \otimes G_2) = 0$ and the equality between dimensions show that $G_1 \otimes G_2$ is a generator matrix for $\mathcal{C}_1 \otimes \mathcal{C}_2$. Moreover, as shown in [99], the minimal distance of $\mathcal{C}_1 \otimes \mathcal{C}_2$ is equal to $d_1 d_2$ but we do not use this fact in this manuscript.

Chapter 8

Conclusion

In this work, we showed that quantum expander codes and the small-set-flip decoder can be used in the construction of ref. [46] to achieve fault-tolerant quantum computation with constant space overhead. The small-set-flip decoder can be parallelized to run in constant time and, as a consequence, each location of the resulting fault-tolerant circuit needs to run a constant depth classical circuit. By comparison, the time complexity for decoding other family of codes is not known to be constant leading to a fault-tolerant circuit where each time step requires to run a classical circuit whose depth grows with the number of qubits.

As a future work, it would be interesting to try to reduce the time overhead of the construction presented in ref. [46]. In particular, the time overhead is large if the initial circuit is parallel since it has to be turned into a sequential circuit for the construction to work.

For quantum expander codes and other hypergraph product codes, it is crucial to find decoders with better performance and lower time consumption; for example with decoders based on the small-set-flip algorithm or the belief propagation decoder (see [85]). Furthermore, the simulations of ref. [49] require codes with large block-length and it would be interesting to study how the modified hypergraph product codes of ref. [65, 78] perform in practice. Finally, numerical studies for the parallel small-set-flip decoder and for the case of noisy syndrome measurements would be insightful as well.

Bibliography

[1] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM Journal on Computing*, 38(4):1207–1282, 2008.

[2] Panos Aliferis, Daniel Gottesman, and John Preskill. Quantum accuracy threshold for concatenated distance-3 codes. *arXiv preprint quant-ph/0504218*, 2005.

[3] Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes. In *2008 IEEE International Symposium on Information Theory*, pages 1173–1177. IEEE, 2008.

[4] CJ Ballance, TP Harty, NM Linke, MA Sepiol, and DM Lucas. High-fidelity quantum logic gates using trapped-ion hyperfine qubits. *Physical review letters*, 117(6):060504, 2016.

[5] Alexander Barg and Gilles Zémor. Error exponents of expander codes. *IEEE Transactions on Information Theory*, 48(6):1725–1729, 2002.

[6] Alexander Barg and Gilles Zemor. Distance properties of expander codes. *IEEE Transactions on Information Theory*, 52(1):78–90, 2005.

[7] Alexander Barg and Gilles Zémor. Multilevel expander codes. In *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.*, pages 1315–1319. IEEE, 2005.

[8] Charles H Bennett, David P DiVincenzo, John A Smolin, and William K Wootters. Mixed-state entanglement and quantum error correction. *Physical Review A*, 54(5):3824, 1996.

[9] Claude Berrou, Alain Glavieux, and Punya Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Proceedings of ICC'93-IEEE International Conference on Communications*, volume 2, pages 1064–1070. IEEE, 1993.

[10] Manuel Blum, Richard M. Karp, Oliver Vornberger, Christos H Papadimitriu, and Mihalis Yannakakis. The complexity of testing whether a graph is a superconcentrator. *Information Processing Letters*, 13(4-5):164–167, 1981.

[11] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J Bremner, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14(6):595, 2018.

[12] Héctor Bombín. Single-shot fault-tolerant quantum error correction. *Physical Review X*, 5(3):031043, 2015.

[13] Sergey Bravyi, David Gosset, and Robert Koenig. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018.

[14] Sergey Bravyi, David Poulin, and Barbara Terhal. Tradeoffs for reliable quantum information storage in 2d systems. *Physical review letters*, 104(5):050503, 2010.

[15] Sergey B Bravyi and A Yu Kitaev. Quantum codes on a lattice with boundary. *arXiv preprint quant-ph/9811052*, 1998.

[16] Nikolas P Breuckmann and Barbara M Terhal. Constructions and noise threshold of hyperbolic surface codes. *IEEE transactions on Information Theory*, 62(6):3731–3744, 2016.

[17] Nikolas P Breuckmann, Christophe Vuillot, Earl Campbell, Anirudh Krishna, and Barbara M Terhal. Hyperbolic and semi-hyperbolic surface codes for quantum storage. *Quantum Science and Technology*, 2(3):035007, 2017.

[18] David Burshtein. On the error correction of regular ldpc codes using the flipping algorithm. *IEEE Transactions on Information Theory*, 54(2):517–530, 2008.

[19] David Burshtein and Gadi Miller. Expander graph arguments for message-passing algorithms. *IEEE Transactions on Information Theory*, 47(2):782–790, 2001.

[20] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[21] Earl T Campbell. A theory of single-shot error correction for adversarial noise. *arXiv preprint arXiv:1805.09271*, 2018.

[22] Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings of the thiryfourth annual ACM symposium on Theory of computing*, pages 659–668. ACM, 2002.

[23] L Childress, MV Gurudev Dutt, JM Taylor, AS Zibrov, F Jelezko, J Wrachtrup, PR Hemmer, and MD Lukin. Coherent dynamics of coupled electron and nuclear spin qubits in diamond. *Science*, 314(5797):281–285, 2006.

[24] Sae-Young Chung. *On the construction of some capacity-approaching coding schemes*. PhD thesis, Massachusetts Institute of Technology, 2000.

[25] Patrick J Coles, Stephan Eidenbenz, Scott Pakin, Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, et al. Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*, 2018.

[26] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[27] Nicolas Delfosse. Tradeoffs for reliable quantum information storage in surface codes and color codes. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 917–921. IEEE, 2013.

[28] Nicolas Delfosse and Naomi H Nickerson. Almost-linear time decoding algorithm for topological codes. *arXiv preprint arXiv:1709.06218*, 2017.

[29] Nicolas Delfosse and Jean-Pierre Tillich. A decoding algorithm for css codes using the x/z correlations. In *2014 IEEE International Symposium on Information Theory*, pages 1071–1075. IEEE, 2014.

[30] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002.

[31] Michel H Devoret and John M Martinis. Implementing qubits with superconducting integrated circuits. In *Experimental aspects of quantum computing*, pages 163–203. Springer, 2005.

[32] Kasper Duivenvoorden, Nikolas P Breuckmann, and Barbara M Terhal. Renormalization group decoder for a four-dimensional toric code. *IEEE Transactions on Information Theory*, 65(4):2545–2562, 2018.

[33] MV Gurudev Dutt, L Childress, L Jiang, E Togan, J Maze, F Jelezko, AS Zibrov, PR Hemmer, and MD Lukin. Quantum register based on individual electronic and nuclear spin qubits in diamond. *Science*, 316(5829):1312–1316, 2007.

[34] Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B*, 69(125-130):55–56, 1965.

[35] Peter Elias. Error-free coding. 1954.

[36] Peter Elias. Coding for noisy channels. *IRE Conv. Rec.*, 3:37–46, 1955.

[37] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Constant overhead quantum fault-tolerance with quantum expander codes. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 743–754. IEEE, 2018.

[38] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Efficient decoding of random errors for quantum expander codes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 521–534. ACM, 2018.

[39] A Jimenez Felstrom and Kamil Sh Zigangirov. Time-varying periodic convolutional codes with low-density parity-check matrix. *IEEE Transactions on Information Theory*, 45(6):2181–2191, 1999.

[40] G David Forney. Concatenated codes. 1965.

[41] Michael H Freedman, David A Meyer, and Feng Luo. Z2-systolic freedom and quantum codes. *Mathematics of quantum computation, Chapman & Hall/CRC*, pages 287–320, 2002.

[42] Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[43] Daniel Gottesman. *Stabilizer codes and quantum error correction*. PhD thesis, California Institute of Technology, 1997.

[44] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. *arXiv preprint arXiv:0904.2557*, 2009.

[45] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. In *Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics*, volume 68, pages 13–58, 2010.

[46] Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *Quantum Information & Computation*, 14(15-16):1338–1372, 2014.

[47] Daniel Gottesman and Isaac L Chuang. Quantum teleportation is a universal computational primitive. *arXiv preprint quant-ph/9908010*, 1999.

[48] Geoffrey Grimmett. What is percolation? In *Percolation*, pages 1–31. Springer, 1999.

[49] Antoine Grospellier and Anirudh Krishna. Numerical study of hypergraph product codes. *arXiv preprint arXiv:1810.03681*, 2018.

[50] Lov K Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical review letters*, 79(2):325, 1997.

[51] Larry Guth and Alexander Lubotzky. Quantum error correcting codes and 4-dimensional arithmetic hyperbolic manifolds. *Journal of Mathematical Physics*, 55(8):082202, 2014.

[52] Richard W Hamming. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160, 1950.

[53] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[54] Aram W Harrow and Ashley Montanaro. Quantum computational supremacy. *Nature*, 549(7671):203, 2017.

[55] Matthew B Hastings. Decoding in hyperbolic spaces: quantum LDPC codes with linear rate and efficient error correction. *Quantum Information & Computation*, 14(13-14):1187–1202, 2014.

[56] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

[57] Xiao-Yu Hu, Evangelos Eleftheriou, and Dieter-Michael Arnold. Regular and irregular progressive edge-growth tanner graphs. *IEEE Transactions on Information Theory*, 51(1):386–398, 2005.

[58] Svante Janson. On percolation in random graphs with given vertex degrees. *Electronic Journal of Probability*, 14:86–118, 2009.

[59] Rolf Johannesson and Kamil Sh Zigangirov. *Fundamentals of convolutional coding*, volume 15. John Wiley & Sons, 2015.

[60] David Kielpinski, Chris Monroe, and David J Wineland. Architecture for a large-scale ion-trap quantum computer. *Nature*, 417(6890):709, 2002.

[61] A Yu Kitaev. Quantum error correction with imperfect gates. In *Quantum Communication, Computing, and Measurement*, pages 181–188. Springer, 1997.

[62] A Yu Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.

[63] Emanuel Knill and Raymond Laflamme. Theory of quantum error-correcting codes. *Physical Review A*, 55(2):900, 1997.

[64] Ralf Koetter and Alexander Vardy. Algebraic soft-decision decoding of reed-solomon codes. *IEEE Transactions on Information Theory*, 49(11):2809–2825, 2003.

[65] Alexey A Kovalev and Leonid P Pryadko. Improved quantum hypergraph-product ldpc codes. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 348–352. IEEE, 2012.

[66] Alexey A Kovalev and Leonid P Pryadko. Fault tolerance of quantum low-density parity check codes with sublinear distance scaling. *Physical Review A*, 87(2):020304, 2013.

[67] Anthony Leverrier, Jean-Pierre Tillich, and Gilles Zémor. Quantum expander codes. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 810–824. IEEE, 2015.

[68] Ye-Hua Liu and David Poulin. Neural belief-propagation decoders for quantum error-correcting codes. *arXiv preprint arXiv:1811.07835*, 2018.

[69] Vivien Londe and Anthony Leverrier. Golden codes: quantum LDPC codes built from regular tessellations of hyperbolic 4-manifolds. *arXiv preprint arXiv:1712.08578*, 2017.

[70] Michael Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, Daniel A Spielman, and V Stemann. Analysis of low density codes and improved designs using irregular graphs. In *STOC*, volume 98, pages 249–258, 1998.

[71] Michael Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, Daniel A Spielman, and Volker Stemann. Practical loss-resilient codes. In *STOC*, volume 97, pages 150–159, 1997.

[72] Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, 2001.

[73] Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on information Theory*, 47(2):585–598, 2001.

[74] Russell Lyons. Random walks, capacity and percolation on trees. *The Annals of Probability*, pages 2043–2088, 1992.

[75] J Majer, JM Chow, JM Gambetta, Jens Koch, BR Johnson, JA Schreier, L Frunzio, DI Schuster, Andrew Addison Houck, Andreas Wallraff, et al. Coupling superconducting qubits via a cavity bus. *Nature*, 449(7161):443, 2007.

[76] Yongyi Mao and Amir H Banihashemi. A heuristic search for good low-density parity-check codes at short block lengths. In *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No. 01CH37240)*, volume 1, pages 41–44. IEEE, 2001.

[77] Grigorii A Margulis. Explicit constructions of graphs without short cycles and low density codes. *Combinatorica*, 2(1):71–78, 1982.

[78] Denise Maurice. *Codes correcteurs quantiques pouvant se décoder itérativement*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2014.

[79] Brendan D McKay. Asymptotics for symmetric 0-1 matrices with prescribed row sums. *Ars Combin*, 19:15–25, 1985.

[80] Gadi Miller and Gerard Cohen. The rate of regular ldpc codes. *IEEE Transactions on Information Theory*, 49(11):2989–2992, 2003.

[81] Hendrik Poulsen Nautrup, Nicolas Delfosse, Vedran Dunjko, Hans J Briegel, and Nicolai Friis. Optimizing quantum error correction codes with reinforcement learning. *arXiv preprint arXiv:1812.08451*, 2018.

[82] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[83] Peter Oswald and Amin Shokrollahi. Capacity-achieving sequences for the erasure channel. *IEEE Transactions on Information Theory*, 48(12):3017–3028, 2002.

[84] Chin Hee Pah and Mohamed Ridza Wahiddin. Combinatorial interpretation of raney numbers and tree enumerations. *Open Journal of Discrete Mathematics*, 5(01):1, 2015.

[85] Pavel Panteleev and Gleb Kalachev. Degenerate quantum ldpc codes with good finite length performance. *arXiv preprint arXiv:1904.02703*, 2019.

[86] David Poulin and Yeojin Chung. On the iterative decoding of sparse quantum codes. *arXiv preprint arXiv:0801.1241*, 2008.

[87] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[88] Shruti Puri, Lucas St-Jean, Jonathan A Gross, Alexander Grimm, NE Frattini, Pavithran S Iyer, Anirudh Krishna, Steven Touzard, Liang Jiang, Alexandre Blais, et al. Bias-preserving gates with stabilized cat qubits. *arXiv preprint arXiv:1905.00450*, 2019.

[89] Irving S Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics*, 8(2):300–304, 1960.

[90] Thomas J Richardson and Rüdiger L Urbanke. Efficient encoding of low-density parity-check codes. *IEEE Transactions on information theory*, 47(2):638–656, 2001.

[91] Tom Richardson and Shrinivas Kudekar. Design of low-density parity check codes for 5g new radio. *IEEE Communications Magazine*, 56(3):28–34, 2018.

[92] Tom Richardson and Ruediger Urbanke. *Modern coding theory*. Cambridge University Press, 2008.

[93] Aline Roumy, Souad Guemghar, Giuseppe Caire, and Sergio Verdú. Design methods for irregular repeat accumulate codes. In *IEEE International Symposium on Information Theory, 2003. Proceedings.*, page 2. IEEE, 2003.

[94] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[95] Mahyar Shirvanimoghaddam, Mohammad Sadegh Mohammadi, Rana Abbas, Aleksandar Minja, Chentao Yue, Balazs Matuz, Guojun Han, Zihuai Lin, Wanchun Liu, Yonghui Li, et al. Short block-length codes for ultra-reliable low latency communications. *IEEE Communications Magazine*, 57(2):130–137, 2019.

[96] Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

[97] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[98] Michael Sipser and Daniel A Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996.

[99] N Sloane. The theory of error-correcting codes, 1981.

[100] Daniel A Spielman. Finding good ldpc codes. In *PROCEEDINGS OF THE ANNUAL ALLERTON CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING*, volume 36, pages 211–219. UNIVERSITY OF ILLINOIS, 1998.

[101] Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1954):2551–2577, 1996.

[102] Andrew M Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77(5):793, 1996.

[103] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of complexity*, 13(1):180–193, 1997.

[104] R Tanner. A recursive approach to low complexity codes. *IEEE Transactions on information theory*, 27(5):533–547, 1981.

[105] Barbara M Terhal and Guido Burkard. Fault-tolerant quantum computation for local non-markovian noise. *Physical Review A*, 71(1):012336, 2005.

[106] Jeremy Thorpe. Low-density parity-check (ldpc) codes constructed from protographs. *IPN progress report*, 42(154):42–154, 2003.

[107] Tao Tian, Christopher R Jones, John D Villasenor, and Richard D Wesel. Selective avoidance of cycles in irregular ldpc code construction. *IEEE Transactions on Communications*, 52(8):1242–1247, 2004.

[108] Jean-Pierre Tillich and Gilles Zémor. Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, 2014.

[109] Ryuhei Uehara et al. The number of connected components in graphs and its applications. *Manuscript. URL: http://citeseerx. ist. psu. edu/viewdoc/summary*, 1999.

[110] Chih-Chun Wang, Sanjeev R Kulkarni, and H Vincent Poor. Upper bounding the performance of arbitrary finite ldpc codes on binary erasure channels. In *2006 IEEE International Symposium on Information Theory*, pages 411–415. IEEE, 2006.

[111] Dominic Welsh. *Codes and cryptography*. Oxford University Press, 1988.

[112] Nicholas C Wormald et al. Models of random regular graphs. *London Mathematical Society Lecture Note Series*, pages 239–298, 1999.

[113] Jonathan S Yedidia, Erik B Sudderth, and Jean-Philippe Bouchaud. Projection algebra analysis of error-correcting codes. In *PROCEEDINGS OF THE ANNUAL ALLERTON CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING*, volume 39, pages 662–671. The University; 1998, 2001.

[114] JQ You and Franco Nori. Superconducting circuits and quantum information. *arXiv preprint quant-ph/0601121*, 2006.

[115] Junan Zhang and Alon Orlitsky. Finite-length analysis of ldpc codes with large left degrees. In *Proceedings IEEE International Symposium on Information Theory,*, page 3. IEEE, 2002.