Borne supérieure sur la capacité de correction des codes stabilisateurs

Denise Maurice
sous la direction de Jean-Pierre Tillich

19 août 2009

Le contexte général

Mon stage a porté sur la recherche d'une borne supérieure sur la capacité de certains codes quantiques. Il a été réalisé à l'INRIA Rocquencourt, dans l'équipe SECRET.

L'informatique quantique est un domaine relativement nouveau, et il reste beaucoup de choses qui, si elles sont simples et déjà largement connues dans le monde classique, le sont beaucoup moins dans le quantique. C'est le cas des codes correcteurs. Il est naturel d'imaginer qu'un qubit est, dans la "nature" tout aussi (voire plus) soumis à des erreurs qu'un bit classique. On connaît différents codes quantiques, et on sait en dégager certaines propriétés, par contre la *capacité* des canaux reste encore très difficile à estimer.

Le problème étudié

Le but de ce stage était de chercher une borne supérieure sur la capacité de correction des codes stabilisateurs, qui sont un type de codes correcteurs quantiques. Actuellement, même dans le cas d'un canal très simple, le canal de Pauli (l'équivalent quantique du canal binaire symétrique), on n'a essentiellement pas de bornes non triviales.

La difficulté vient du fait que, dans un code quantique et contrairement au classique, il est parfois possible qu'un syndrome corresponde à plusieurs erreurs, et que la correction qu'on applique marche pour toutes ces erreurs.

Les codes stabilisateurs sont définis par un groupes d'erreurs qui le laisse stable. Travailler avec des groupes stabilisateurs a donc l'avantage de permettre d'étudier un espace discret (l'espace des erreurs est isomorphe à $\mathbb{F}_4{}^n$) au lieu de l'espace des mots de code, qui est continu.

La contribution proposée

L'idée de la démarche est la suivante.

On fixe une probabilité d'erreur p, et on essaie de voir à partir de quel p il n'existe pas de famille de code (de rendement non nul) dont la probabilité d'erreur tend vers 0.

L'idée est de partitionner la boule de rayon pn en sous-ensembles dont les éléments, deux à deux, diffèrent par un élément du stabilisateur. Ils représentent les ensembles d'erreurs qui sont corrigées "ensemble". De là, on considère qu'on ne peut pas corriger correctement les erreurs s'il y a, en probabilité, "trop" de tels ensembles qui ne pourront pas avoir de syndrome attribué. Il faut donc estimer cette somme, par des arguments combinatoires. Bien sûr ce n'est pas sa valeur exacte qui compte, mais son terme dominant (son exposant), car c'est ça qui nous dira si la probabilité d'erreur est négligeable ou pas (et donc si elle peut tendre vers 0).

On cherche dans deux voies : d'une part une borne supérieure stricte, d'autre part une borne inférieure estimée la borne supérieure optimale, qui peut-être une bonne conjecture de la borne réelle.

Les arguments en faveur de sa validité

On obtient, avec des calculs numériques vers la fin, une borne supérieure assez mauvaise sur p, supérieure à la seule connue, le tout sous certaines hypothèses. Comme la borne inférieure supposée est également au dessus, il est probable qu'affiner le résultat n'aide pas à obtenir une borne de cette façon, en supposant que les hypothèses faites sont correctes (ce qu'il est raisonnable de penser). Par contre il y a une hypothèse non encore utilisée, qui peut elle donner de bons résultats.

Le bilan et les perspectives

Cette approche n'était qu'un début à la recherche de bornes supérieures sur la capacité des codes. Il y a encore beaucoup de travail à faire dans le domaine. Déjà, même avec cette méthode, voir si on peut se passer des hypothèses faites. Enfin, il reste l'hypothèse qui n'a pas été utilisée, concernant le stabilisateur, qui n'est pas un simple sous-groupe de ${\mathbb{F}_4}^n$. Notamment, il est évident que la borne inférieure sur la borne optimale est calculée à partir d'exemples qui ne vérifient clairement pas cette hypothèse. Le temps nous a manqué pour chercher comment l'utiliser, mais il y a probablement des résultats intéressants à obtenir.

Table des matières

1 Les codes correcteurs quantiques

Comment se corrige une erreur quantique ? Comme dans un cas classique, on va effectuer une tranformation qui enverra k qubits sur n qubits, n étant évidemment plus grand que k pour compenser l'erreur. La correction se fera sur une mesure du système, c'est à dire le projeter dans un espace de dimension k. Ensuite, avec la mesure, qui nous donne une information sur le système, on peut éventuellement effectuer une opération sur le système ainsi projeté.

Avant de comprendre comment corriger une erreur quantique, il convient de savoir quel genre d'erreur on peut rencontrer. On commence par regarder un analogue au cas classique.

1.1 Exemples

Que peut-il arriver à un qubit $\alpha|0\rangle + \beta|1\rangle$? Il peut subir un "flip" et être changé en $\alpha|1\rangle + \beta|0\rangle$. Pour se protéger de ce type d'erreur, on utilise une adaptation du code à répétition :

$$\begin{array}{rcl} |0\rangle & \rightarrow & |000\rangle \\ |1\rangle & \rightarrow & |111\rangle \end{array} \tag{1}$$

On va montrer (comme pour le code à répétition) que si il n'y a pas plus d'une erreur, on peut la corriger. L'espace complet peut se décomposer en quatre sous-espaces orthogonaux :

$$\begin{array}{rcl} C_{00} & = & Vect(|000\rangle, |111\rangle) \\ C_{01} & = & Vect(|100\rangle, |011\rangle) \\ C_{10} & = & Vect(|010\rangle, |101\rangle) \\ C_{11} & = & Vect(|001\rangle, |110\rangle) \end{array}$$

Supposons que le premier qubit de $|\phi\rangle = \alpha|000\rangle + \beta|111\rangle$ soit changé. La mesure dans la base indiquée ci-dessus nous indique qu'on est dans C_{01} et qu'on doit changer le premier qubit. On lui applique la tranformation nécessaire, et on retrouve bien le qubit de départ. De même pour les autres qubits : la mesure nous apprend dans quel sous-espace (entre $C_{00}, C_{01}, C_{10}, C_{11}$) où $|\phi\rangle$ se trouve, et donc quelle transformation on doit effectuer (respectivement rien du tout, changer le premier, le deuxième, le troisième).

Si, au lieu de changer complètement un des qubits, on le change "partiellement", c'est à dire :

$$\begin{array}{rcl} |000\rangle & \rightarrow & a|000\rangle + b|100\rangle + c|010\rangle + d|001\rangle \\ |111\rangle & \rightarrow & a|111\rangle + b|011\rangle + c|101\rangle + d|110\rangle \end{array}$$

avec $a^2 + b^2 + c^2 + d^2 = 1$.

Alors, lors de la mesure, avec probabilité a^2 l'état $\alpha(a|000\rangle + \ldots) + \beta(a|111\rangle + \ldots)$ est projeté dans C_{00}, et on retrouve bien l'état de départ : α et β ne sont

pas changés. Avec probabilité b^2, on se retrouve dans l'état $\alpha|100\rangle + \beta|011\rangle$, et comme la mesure nous apprend qu'on doit changer le premier qubit, on se retrouve également dans la configuration de départ.

On peut donc corriger une erreur de "flip" de façon sûre s'il n'y a pas plus d'une erreur.

Remarque 1. *Un clonage des qubits revient à obtenir* $(\alpha|000\rangle + \beta|111\rangle) \otimes (\alpha|000\rangle + \beta|111\rangle) \otimes (\alpha|000\rangle + \beta|111\rangle)$*, ce qui n'est pas notre cas : on respecte bien la condition de non-clonage.*

D'autres erreurs peuvent survenir, notamment des erreurs dites de phase :

$$\begin{aligned} |0\rangle &\rightarrow |0\rangle \\ |1\rangle &\rightarrow -|1\rangle \end{aligned}$$

Une méthode analogue va nous permettre d'y pallier. On s'occupe de corriger les erreurs de phase seulement pour le moment, on verra plus bas comment faire les deux à la fois.

On effectue donc le codage suivant :

$$\begin{aligned} |0\rangle &\rightarrow \frac{1}{2^{3/2}}(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \\ |1\rangle &\rightarrow \frac{1}{2^{3/2}}(|0\rangle - |1\rangle) \otimes (|0\rangle - |1\rangle) \otimes (|0\rangle - |1\rangle) \end{aligned}$$

Autrement dit, si on note $|+\rangle = \frac{1}{2}(|0\rangle + |1\rangle)$ et $|-\rangle = \frac{1}{2}(|0\rangle - |1\rangle)$, cela revient à :

$$\begin{aligned} |0\rangle &\rightarrow |+++\rangle \\ |1\rangle &\rightarrow |---\rangle \end{aligned}$$

Une erreur de phase correspond ici à transformer un $|+\rangle$ en $|-\rangle$ et inversement : c'est une erreur de flip sur cette nouvelle base. On peut donc appliquer la même méthode de mesure, en prenant comme base :

$$\begin{aligned} C_{++} &= Vect(|+++\rangle, |---\rangle) \\ C_{+-} &= Vect(|-++\rangle, |+--\rangle) \\ C_{-+} &= Vect(|+-+\rangle, |-+-\rangle) \\ C_{--} &= Vect(|++-\rangle, |--+\rangle) \end{aligned}$$

Avec le même raisonnement que plus haut, en remplaçant les 0 par des + et les 1 par des −, on peut corriger de telles erreurs.

À présent, comment combine-t-on les deux types de correction ? Avec le code suivant, à 9 qubits :

$$\begin{aligned} |0\rangle &\rightarrow |\tilde{0}\rangle = \tfrac{1}{2^{3/2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ |1\rangle &\rightarrow |\tilde{1}\rangle = \tfrac{1}{2^{3/2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \end{aligned} \quad (2)$$

Les "triplets" corrigent les erreurs de flip, les triplets "+" et "−" corrigent les erreurs de phase. De façon plus détaillée, on projette ces qubits (dimension 2^9) dans un espace de dimension 2. Il y a donc une base composée de $2^8 = 256$ sous-espaces, chacun contenant deux vecteurs de base. Quelques exemple de ces tels sous-espaces (le premier étant le code lui-même) :

$$\begin{array}{ll} C_{00,00,00,++} & = Vect\left(\frac{1}{2^{3/2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle),\right. \\ & \left.\frac{1}{2^{3/2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)\right) \\ C_{01,00,11,+-} & = Vect\left(\frac{1}{2^{3/2}}(|100\rangle - |011\rangle)(|000\rangle + |111\rangle)(|001\rangle + |110\rangle),\right. \\ & \left.\frac{1}{2^{3/2}}(|100\rangle + |011\rangle)(|000\rangle - |111\rangle)(|001\rangle - |110\rangle)\right) \end{array}$$

On peut corriger là une erreur de flip *et* une erreur de phase. On peut noter qu'on peut parfois corriger plusieurs erreurs de flip (si elles frappent des "blocs" de 3 différents), mais dans le pire des cas, on en corrige au moins une (et une erreur de phase).

Cela corrige donc également une erreur de type :

$$\begin{array}{lll} |0\rangle & \rightarrow & -|1\rangle \\ |1\rangle & \rightarrow & |0\rangle \end{array}$$

puisqu'il s'agit d'une combinaison d'une erreur de flip et d'une de phase (on reviendra plus tard sur ce type d'erreur).

Par contre, on ne corrige pas d'erreur de type :

$$\begin{array}{lll} |0\rangle & \rightarrow & \omega|0\rangle \\ |1\rangle & \rightarrow & \omega|1\rangle \end{array}$$

avec $|\omega|^2 = 1$. Mais ce cas correspond à des états indistinguables, donc n'est pas gênant.

En fait, ce qui compte réellement n'est pas d'obtenir $|\phi\rangle$ à la fin, mais plutôt que sa matrice de densité $\rho = |\phi\rangle\langle\phi|$ soit retrouvée, puisque c'est ce qui donne les probabilités de mesurer les 0 ou les 1.

1.2 Définitions

De façon générale, une erreur sur un système quantique apparaît du fait de l'isolement imparfait du système avec l'environnement. Cet environnement étant mesuré en permanence, le système complet (environnement et système) est projeté, modifiant ainsi le système censé être isolé du monde. Un peu comme si la célèbre boîte du le chat de Schrödinger (contenant le système |chat vivant⟩+ |chat mort⟩) n'est pas parfaitement isolée phoniquement et laisse échapper quelques miaulements.

En pratique, une erreur est un opérateur unitaire qui est appliqué au système. Elle est de la forme, pour un système à n qubits : $E_1 \otimes E_2 \otimes \ldots \otimes E_n$, avec $E_n \in \{I, X, Y, Z\}$ où X, Y, Z sont les opérateurs de Pauli :

Définition 1 (Opérateurs de Pauli de base)**.**

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

On remarque que $Y = iXZ$. La raison d'être du i et qu'il permet d'avoir $Y^2 = I$, comme les autres matrices de Pauli, et on a vu de toutes façons qu'une multiplication par un facteur ne gênait pas la correction d'erreur.

On appelle cet ensemble d'erreurs le groupe de Pauli :

Définition 2 (Groupe de Pauli)**.**

$$\mathcal{G}_n = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}^{\otimes n}$$

Ce groupe fini, de taille 16^n, possède les propriétés suivantes :
– $\forall \mathcal{M} \in \mathcal{G}_n, \mathcal{M}^\dagger = \mathcal{M}^{-1}$, autrement dit les matrices sont toutes unitaires,
– $\forall \mathcal{M} \in \mathcal{G}_n, \mathcal{M}^2 = \pm I$,
– $\forall \mathcal{M}, \mathcal{N} \in \mathcal{G}_n, \mathcal{M}N = \pm \mathcal{N}\mathcal{M}$.

On définit alors le *poids de Pauli* d'une erreur comme le nombre de matrices du produit tensoriel différentes de l'identité à un facteur multiplicatif près.

1.3 Propriétés de base

Si on se donne un ensemble $E \subset G_n$ d'erreurs pouvant être corrigées par un code C, voyons quelles sont les conditions nécessaires et suffisantes sur E et sur C. Si on note $|\bar{i}\rangle, i \in 0, 2^n - 1$ les vecteurs de base du code, une condition nécessaire est :

$$\langle \bar{j} | E_a^\dagger E_b | \bar{i} \rangle = 0 \qquad \forall i \neq j, \quad \forall E_a \neq E_b \in E \tag{3}$$

En effet, si l'un de ces produits scalaires donne un $\varepsilon \neq 0$, alors ils ne sont plus orthogonaux : avec probabilité $|\varepsilon|^2$, les systèmes $E_a|\bar{j}\rangle$ et $E_b|\bar{i}\rangle$ sont projetés sur le même vecteur. Ainsi, on va effectuer la même opération pour "corriger" l'erreur. Donc on obtiendra deux fois le même système après correction (alors qu'on avait deux systèmes différents au départ).

Une condition suffisante assez simple à établir est :

$$\langle \bar{j} | E_a^\dagger E_b | \bar{i} \rangle = \delta_{i,j} \delta_{E_a, E_b} \tag{4}$$

En effet, elle peut être réécrite comme $\langle \bar{j} | E_a^\dagger E_b | \bar{i} \rangle = \delta_{(E_a,i),(E_b,j)}$. Autrement dit, les $E_a\,|i\rangle$ forment une base orthonormée de l'espace (ou du moins, du sous-espace engendré par les erreurs possibles). On peut donc fabriquer la décomposition en sous-espaces suivante :

$$\oplus_{E_a} Vect\left((E_a\,|i\rangle)_i\right)$$

et on associe à chaque sous espace la "réparation" $E_a^\dagger$ correspondante.

La propriété suivante (donnée dans [2]) donne la condition nécessaire et suffisante :

Proposition 1. *Un code C corrige l'ensemble d'erreurs E si et seulement si la condition suivante est réalisée :*

$$\forall E_a, E_b \in E,\ \forall |\bar{i}\rangle, |\bar{j}\rangle \in C \tag{5}$$

$$\langle \bar{j}|E_a^\dagger E_b|\bar{i}\rangle = \delta_{i,j} C_{ab} \tag{6}$$

où C_{ab} est une constante ne dépendant pas de $|\bar{i}\rangle$, mais dépendant à priori de E_a et E_b.

Démonstration. Voir en annexe, page 21 □

1.4 Les codes stabilisateurs

On va définir une certaine catégorie de codes, appelés *codes stabilisateurs*. C'est sur ces codes qu'on cherchera la borne supérieure sur la capacité, car ils sont plus faciles à étudier que les codes généraux.

Comme on a vu plus haut que la phase importait peu (une erreur du type $iI^{\otimes n}$ ne gène pas), il est intéressant de considérer le groupe quotienté, celui où on "oublie" la phase :

Définition 3 (Groupe de Pauli quotienté)**.**

$$G_n = \mathcal{G}_{n/\{\pm I, \pm iI\}}$$

Par convention, on notera avec des lettres droites M les matrices du groupe quotienté, et avec des lettres rondes $\mathcal{M}$ les matrices du groupe de Pauli d'origine. On écrira donc $[\mathcal{M}] = M$.

Ce nouveau groupe est commutatif (vues les propriétés de $\mathcal{G}_n$), et est isomorphe à $\mathbb{F}_4^n$, où I est associé à 0, Z à 1, X à ω et Y à $1+\omega$.

Si la phase ne nous intéresse pas, en revanche il est intéressant de garder une trace de la commutation ou de l'anti-commutation des opérateurs. On définit donc une opération appelée ★ :

Définition 4 (Opérateur ★)**.**

$$\forall \mathcal{M}, \mathcal{N} \in \mathcal{G}_n, M \bigstar N = \begin{cases} 0 \text{ si } \mathcal{M}\mathcal{N} = \mathcal{N}\mathcal{M} \\ 1 \text{ sinon} \end{cases}$$

Cet opérateur ★ peut s'exprimer en fonction des vecteurs de $\mathbb{F}_4^n$: M représente un $(m_i) \in \mathbb{F}_4^n$, et N est un (n_i). On a alors :

Définition 5.

$$M \bigstar N =< m, n >= \mathrm{tr}\left(\sum_{i=1}^{n} m_i \bar{n_i}\right)$$

où $\bar{x} = x^2$ dans $\mathbb{F}_4^n$, et $\mathrm{tr}(x) = x + \bar{x}$.

Remarque 2. *Le résultat de cette opération est dans $\mathbb{F}_2$.*

Équivalence des deux définitions. Voir en annexe page 22

□

Définition 6 (Code stabilisateur)**.** *Si S est un sous-groupe abélien de $\mathcal{G}_n$ ne contenant pas $-I$, alors S est le* stabilisateur *du code C si :*

$$|\psi\rangle \in C \Longleftrightarrow M|\psi\rangle = |\psi\rangle \quad \forall \mathcal{M} \in S$$

Remarque 3. *Si S n'était pas abélien, on aurait un $\mathcal{M}$ et un $\mathcal{N}$ tels que $\mathcal{MN}|\psi\rangle = |\psi\rangle = -\mathcal{NM}|\psi\rangle = -|\psi\rangle$, ce qui serait contradictoire si le code est non trivial.*

De même, si $-I$ était dans le stabilisateur, on aurait $-|\psi\rangle = |\psi\rangle$, ce qui impliquerait que le code est trivial.

Par exemple, pour le code (1), le stabilisateur est :

$$S = \{\pm i, \pm 1\}\{IZZ, ZIZ, ZZI\}.$$

Comme la phase $(i, -i, 1, -1)$ n'a pas d'importance, et que l'un de ces stabilisateurs est le produit des deux autres, on note donc :

$$S = < IZZ, ZZI > .$$

Pour le code (2), le stabilisateur est :

$$S = < IZZI^6, ZZII^6, I^3IZZI^3, I^3ZZII^3, I^6IZZ, I^6ZZI, I^3X^6, X^6I^3 >$$

Remarque 4. *On peut définir un espace d'erreurs dites "atteignables", ie toutes les erreurs de la forme $E|\psi\rangle, E \in G_n, |\psi\rangle \in C$. On a alors $\forall |\psi\rangle$, où $|\psi\rangle$ est "atteignable", $\forall \mathcal{M} \in S$, $\mathcal{M}|\psi\rangle = \pm|\psi\rangle$.*
En effet, soit $\mathcal{A}$ tel que $\mathcal{A}|\psi\rangle \in C$. Alors,

$$\begin{aligned} \forall \mathcal{M} \in S, \quad & \mathcal{MA}|\psi\rangle = \mathcal{A}|\psi\rangle \\ \forall \mathcal{M} \in S, \quad & \pm\mathcal{AM}|\psi\rangle = \mathcal{A}|\psi\rangle \\ \forall \mathcal{M} \in S, \quad & \pm\mathcal{M}|\psi\rangle = |\psi\rangle \end{aligned}$$

1.4.1 Propriétés

On peut remarquer, qu'il y avait, dans le premier code, 2 générateurs du stabilisateur pour 1 qubit d'information, et 3 qubits au total. De même, dans le second, 8 générateurs pour 1 qubit d'information, et 9 qubits au total. De façon plus générale,

Théorème 1. *Si S a k générateurs indépendants, alors le code stabilisé par S est de dimension 2^{n-k}.*

Démonstration. Voir en annexe page 22 □

1.4.2 Syndromes

La définition par stabilisateur donne une façon naturelle de décrire un équivalent quantique d'un *syndrome*. Déjà, une erreur "détectable" est une erreur qui ne commute pas avec le stabilisateur entier. En effet, si c'était le cas, pour un certain $|\psi\rangle \in C, \forall \mathcal{M} \in S$,

$$\mathcal{M}(E|\psi\rangle) = E\mathcal{M}|\psi\rangle = E|\psi\rangle$$

Donc le vecteur $E|\psi\rangle$ est dans le code, ce qui fait échouter toute tentative de correction.

Si on a une erreur E, on définit le *syndrome* de la façon suivante :

Définition 7 (Syndrome)**.** *Si* $S =< M_1, \ldots, M_{n-k} >$, $S(E) = (E \bigstar M_i)_{1 \leq i \leq n-k}$

Ce syndrome ne dépend pas de l'état $|\psi\rangle$.

Comment ce syndrome est-il relié à une mesure et à une projection ? On définit, pour chaque syndrome s, l'ensemble $C_s = \{E|\psi\rangle, S(E) = s, |\psi\rangle \in C\}$. On a donc la décomposition de l'espace suivante : $\oplus_s C_s$. Reste à vérifier que cette décomposition est bien orthogonale.

Soient s, s' deux syndromes et E, E' deux erreurs qui ont ce syndrome. Soient $|\psi\rangle \, et \, |\phi\rangle \in C$. Soit i l'indice où $s_i \neq s'_i$. Cela signifie qu'on a $\mathcal{M}_i$ qui commute avec E mais pas E', ou l'inverse.

$$\begin{aligned} \langle\psi| E^\dagger E' |\phi\rangle &= \langle\psi| E^\dagger \mathcal{M}_i^\dagger \mathcal{M}_i E' |\phi\rangle \\ &= -\langle\psi| \mathcal{M}_i^\dagger E^\dagger E' \mathcal{M}_i |\phi\rangle \\ &= -\langle\psi| E^\dagger E' |\phi\rangle \end{aligned}$$

D'où $\langle\psi| E^\dagger E' |\phi\rangle = 0$.

Ainsi, on a une décomposition orthogonale en sous-espaces, de dimension 2^k (puisqu'il y a 2^{n-k} syndromes possibles). Et le syndrome donne exactement dans quel espace on se situe.

Exemple d'encodage et de décodage Voir en annexe page 25

Codes dégénérés ou pas

Avec les codes stabilisateurs, on peut réécrire la notion de distance minimale. La plus petite (au sens d'erreur de Pauli) erreur de syndrome nul (c'est à dire qui soit dans $S^\perp$) qui ne soit pas dans le stabilisateur (une erreur du stabilisateur ne pose aucun problème de correction). Autrement dit :

Définition 8.

$$d := \min\{\mathrm{w}(E), E \in S^\perp \backslash S\}$$

La définition équivalente au cas classique serait "la plus petite erreur non-triviale de syndrome nul" :

$$d' := \min\{\mathrm{w}(E), E \in S^\perp, E \neq I\}$$

On a de façon évidente $d' \leq d$ (le premier ensemble est inclus dans le second). L'égalité est parfois vérifiée : on définit de cette façon les codes *dégénérés* et *non-dégénérés*.

Définition 9 (Code non-dégénéré/dégénéré)**.** *Un code stabilisateur est dit non-dégénéré si $d' = d$. Il est dit dégénéré si $d' < d$.*

Cette définition est cantonnée aux codes stabilisateurs, nous allons donc en donner une plus générale, à partir de la condition 5.

Définition 10 (Code non-dégénéré)**.** *Soit t le poids maximum d'une erreur correctible et E l'ensemble des erreurs de poids inférieur ou égal à t. Un code est non-dégénéré si :*

$$\langle \bar{j} | E_a^\dagger E_b | \bar{i} \rangle = \delta_{i,j}\delta_{a,b}, \quad \forall E_a, E_b \in E$$

(Dans le cas dégénéré on a une constante C_{ab} au lieu du $\delta_{a,b}$).

Équivalence des définitions. Voir en annexe page 23 □

2 Recherche d'une borne supérieure

2.1 Le canal de Pauli

Pour définir la borne cherchée, il faut choisir un modèle de canal. On choisira ici l'analogue quantique du canal binaire symétrique, le canal de Pauli.

Définition 11 (Canal de Pauli)**.** *On fixe une probabilité $p \leq 3/4$. Sur chaque qubit, on effectue l'opération suivante :*
- *Avec probabilité $1-p$, on ne fait rien,*
- *Avec probabilité $p/3$, on applique X,*
- *Avec probabilité $p/3$, on applique Y,*
- *Avec probabilité $p/3$, on applique Z.*

Ici va chercher à partir de quel p (probabilité d'erreur dans le canal de Pauli) la capacité s'annule. Autrement dit, à partir de quel p il n'existe aucune famille de code (de rendement non nul) dont la probabilité d'erreur tend vers 0. On notera qu'en $3/4$, le canal a une capacité nulle car I, X, Y, Z ont chacun une probabilité $1/4$ d'apparaître.

Dans cette section, on travaillera beaucoup sur S le stabilisateur, et on notera $k = \log |S|$. Ce k correspond au $n - k_{\text{quantique}}$.

2.2 Idée de base

On suppose qu'on a p en deçà de cette limite, et donc $P(\text{erreur}) = o(1)$ quand n devient grand. On a la répartition des erreurs suivante :

$$(1-\varepsilon)p \leq \mathrm{w}(E) \leq (1+\varepsilon)p \quad \text{avec proba } 1 - o(1) \text{ quand } \varepsilon \to 0$$

En effet, $\mathbb{E}[\mathrm{w}(E)] = \sum_i P[E_i \text{ soit non nul}] = pn$. De plus, toutes ces erreurs sont équiprobables, plus exactement

$$\frac{1}{n}\log P(E) = \frac{\log\left(\frac{1}{|B_t|}\right)}{n} + \varepsilon'$$

où $\varepsilon' \to 0$ quand $\varepsilon \to 0$.

Comment se passe le décodage ? Pour chaque syndrome détecté (il y en a $2^{n-k_{\text{quantique}}} = 2^k$ ici), l'algorithme va associer un ensemble d'erreurs, différentes deux à deux par un élément du stabilisateur.

On va donc partitionner la boule B_t (avec $t = pn$) en $B_t = \mathcal{B}_1, \mathcal{B}_2, \ldots$ de telle sorte $\forall x, y \in \mathcal{B}_i, x - y \in S$. On les numérote de façon à ce que ces sous-ensembles soient classés par taille : $b_i = |\mathcal{B}_i|$ et $b_1 \geq b_2 \geq \ldots$. On définit ensuite, pour un syndrome s :

$$j(s) = \text{indice du } \mathcal{B}_j \text{ tel que } P[\mathcal{B}_j \text{ donne s}] \text{ soit la plus élevée}$$

On prendra le plus petit indice en cas de probabilités égales. Comme les erreurs sont supposées à peu près équiprobables, les j associés vont être les plus gros. On aura donc au mieux 2^k $\mathcal{B}_j$ associés On a donc :

$$P[\text{erreur}] = P[E \text{ n'ait pas de bon syndrome associé}] \geq \sum_{i=2^k+1}^{\infty} b_i 2^{\varepsilon' n}$$

Si les erreurs sont corrigées on a :

$$\sum_{i=2^k+1}^{\infty} \frac{b_i}{B_t} 2^{\varepsilon' n} = o(1) \text{ quand } \varepsilon \to 0 \text{ et } n \to \infty$$

On cherche donc à partir de quel t cette quantité ne devient plus négligeable, c'est à dire quand $\sum_{i=1}^{2^k} b_i$ est inférieure à $(1 - o(1))|B_t|$. Pour cela,

On écrit l'inégalité de Cauchy-Schwartz :

$$\begin{aligned}
\sum_{i=1}^{2^k} b_i &\leq \sqrt{2^k \sum {b_i}^2} \\
&\leq \sqrt{2^k \left(\sum b_i(b_i - 1) + \sum b_i\right)} \\
&\leq \sqrt{2^k \left(\sum b_i(b_i - 1) + |B_t|\right)}
\end{aligned}$$

On cherche donc à calculer :

$$\begin{aligned}\sum_{i=1}^{2^k} b_i(b_i-1) &= \sum_{c\in S} \#\{(x,y)\in B_t, x-y=c\} \\ &\leq \sum_{s=0}^{2t} \#\{(x,y)\in B_t, x-y=c \text{ fixé de poids } s\}\ \#\{c\in S, \mathrm{w}(c)=S\} \\ &= \sum_{s=0}^{2t} M(s,t)a_s\end{aligned}$$

et ensuite, on cherche à voir s'il y a des valeurs de t pour lequel le terme de droite devient inférieur à $|B_t| = \binom{n}{t}3^t$. On cherchera à évaluer la valeur de l'exposant divisé par n, car c'est ce qui donne l'ordre de grandeur. On utilisera par la suite la majoration suivante, qui est aussi une bonne approximation :

$$\binom{n}{t} \leq 2^{n\,\mathrm{h}(t/n)} \tag{7}$$

avec $h(x) = -x\log x - (1-x)\log(1-x)$.

2.3 Majoration de $M(s,t)$

2.3.1 Calcul explicite de $M(s,t)$

Il faut compter, pour un c fixé de poids s, $\#\{x | x\in B_t, x-c\in B_t\}$. On va supposer x de poids t_x. On cherche donc $\#\{x|\,\mathrm{w}(x)=t_x, x-c\in B_t\}$, et plus précisément, on va fixer un t_I et on va chercher $\#\{x|\,\mathrm{w}(x)=t_x, |I|=t_I, x-c\in B_t\}$, où I est défini comme sur la figure 1.

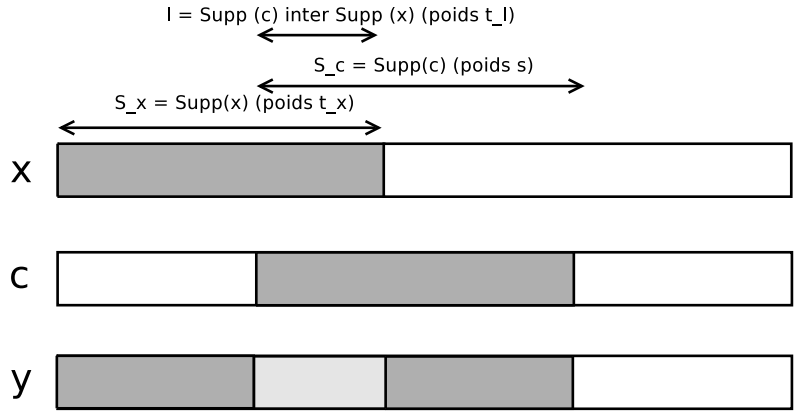

FIG. 1 – Notations pour les ensembles d'indices

Sur cette figure, la zone grise représente le support (ie les emplacements où il y a un élément non nul). Si x est fixé, on peut voir que le support de y comprend au moins $S_x \backslash I, S_c \backslash I$. Il peut aussi y avoir des éléments dans la zone gris clair.

Supposons que t_x et t_I sont fixés (c est déjà fixé et donc s aussi). Combien de choix possibles avons-nous pour x ? D'abord, on fixe les éléments "tous seuls", ie ceux de $\mathrm{Supp}(x) \backslash I$. On a donc $t_x - t_I$ emplacements à choisir, et à remplir avec des éléments de $\mathbb{F}_4$ non nuls :

$$\binom{n-s}{t_x - t_I} 3^{t_x - t_I}$$

Ensuite, pour chacun de ces choix, il faut choisir les éléments de l'intersection. Comme dit plus haut, le poids de y comprend déjà les deux parties grisées sur la figure 1 : $S_x \backslash I, S_c \backslash I$, c'est à dire un poids $t_x + s - 2t_I$. y ne doit pas avoir un poids dépassant t, donc on va choisir sa taille, de la forme $t_x + s - 2t_I \quad + \quad t_J$, où $J \subset I$ sera l'ensemble des emplacements où y est non nul (à l'intérieur de I). Ce t_J va donc varier entre 0 et $t - (t_x + s - 2t_I)$.

On choisit donc, pour chacun de ces t_J, un ensemble J, et on a pour les valeurs de x deux choix : il faut que ça soit un élément de $\mathbb{F}_4$ qui ne soit ni 0 ni la valeur en c. Cela donne :

$$\binom{t_I}{t_J} 2^{t_J}$$

Conclusion : à t_x et t_I fixés, on a :

$$\binom{n-s}{t_x - t_I} 3^{t_x - t_I} \sum_{t_J=0}^{t - t_x - s + 2t_I} \binom{t_I}{t_J} 2^{t_J}$$

Ensuite, quelle doit être la borne sur t_I pour que cela soit possible? Il faut que y soit de poids inférieur à t, or ce poids est au minimum de $t_x - t - I + s - t_I$. Autrement dit, il faut $t_x + s - 2t_I \leq t$. Cela donne $t_I \geq \left\lceil \frac{t_x + s - t}{2} \right\rceil$. Bien entendu, on a également $t_I \leq \min(t_x, s)$. Cela nous donne donc les bornes sur t_I.

Et que dire des bornes sur t_x ? De façon évidente, $0 \leq t_x \leq t$. Et pour que t_I soit positif ou nul, il faut (et il suffit) d'avoir $\frac{t_x + s - t}{2} \leq \min(t_x, s)$. Cela donne deux inégalités : $t_x \geq s - t$ et $t - x \leq s + t$. La deuxième condition est inutile puisqu'on a déjà $t_x \leq t$. On a donc : $\max(0, s - t) \leq t_x \leq t$.

Ce qui donne au final, pour $\#\{\{x, y\} \subset B_t, x - y = c\}$:

$$M(s,t) = \sum_{t_x = \max(0, s-t)}^{t} \sum_{t_I = \left\lceil \frac{t_x + s - t}{2} \right\rceil}^{\min(t_x, s)} \binom{n-s}{t_x - t_I} 3^{t_x - t_I} \sum_{t_J=0}^{t - t_x - s + 2t_I} \binom{t_I}{t_J} 2^{t_J}$$

2.3.2 Majoration de la somme

On note $m(\frac{s}{n}, \frac{t}{n}) = \frac{1}{n} \log M(s,t)$. C'est l'exposant m qui nous intéresse, aussi on va majorer cette somme par son terme dominant, multiplié par le nombre

de termes. Comme ce nombre est polynômial (n^3), il devient négligeable dans $m(s/n, t/n)$.

On cherche donc à majorer

$$f(n,s,t) = \max\left(\binom{n-s}{t_x-t_I}3^{t_x-t_I}\binom{t_I}{t_J}2^{t_J}\right) \quad \text{avec} \quad \begin{array}{l} 0 \le t_x \le t \\ \frac{t_x+s-t}{2} \le t_I \le \min(t_x, s) \\ 0 \le t_J \le t - t_x - s + 2t_I \end{array}$$

Commençons par majorer la partie finale du terme, qu'on notera

$$A(n,s,t,t_x,t_I) = \binom{t_I}{t_J}2^{t_J}$$

pour $0 \le t_J \le t-t_x-s+2t_I$. Avec la majoration 7, on a : $\binom{t_I}{t_J}2^{t_J} \le 2^{t_I\,\mathrm{h}(t_J/t_I)+t_J}$. D'où :

$$\begin{array}{rcl} \binom{t_I}{t_J}2^{t_J} & \le & 2^{t_I\,\mathrm{h}(t_J/t_I)+t_J} \\ & \le & 2^{t_I(h(t_J/t_I)+t_J/t_I)} \\ & \le & 2^{\frac{t_I}{\ln 2}a(x)} \quad \text{avec } x = t_J/t_I \end{array}$$

Cette valeur est maximale quand la fonction a l'est, c'est à dire quand $x = 2/3$ (voir en annexe page 26 les détails).

Si par contre $\frac{t_{Jmax}}{t_I} = \frac{t-t_x-s+2t_I}{t_I} \le 2/3$, alors le maximum est en t_Jmax.

Le terme cherché est donc majoré par

$$A(n,s,t,t_x,t_I) = \begin{cases} 2^{t_I\left(\mathrm{h}(\frac{t_{Jmax}}{t_I})+\frac{t_{Jmax}}{t_I}\right)} & \text{si } \frac{t_{Jmax}}{t_I} \le \frac{2}{3} \\ 3^{t_I} & \text{sinon} \end{cases}$$

Cas numéro 1 Étudions le cas où $\frac{t_{Jmax}}{t_I} \ge 2/3$. On cherche donc à majorer

$$M_1(n,s,t) \le \binom{n-s}{t_x-t_I}3^{t_x} \quad \text{avec} \quad \begin{array}{l} 0 \le t_x \le t \\ \frac{t_x+s-t}{2} \le t_I \le \min(t_x, s) \end{array}$$

On utilise la majoration (7) :

$$2^{(n-s)\,\mathrm{h}\left(\frac{t_x-t_I}{n-s}\right)}3^{t_x}$$

La valeur maximum du terme quand on fait varier t_I, à t_x fixé, est atteinte pour $\frac{t_x-t_I}{n-s} = 1/2$ si c'est possible, sinon pour la valeur maximale de $\frac{t_x-t_I}{n-s}$. Autrement dit la valeur minimale de t_I possible. Pour avoir $t_{Jmax} = t-t_x-s+2t_I \ge 2/3t_I$, on obtient $t_I \ge 3/4(t_x+s-t)$.

$$\begin{cases} 3^{t_x}\ 2^{(n-s)\,\mathrm{h}\left(\frac{3t+t_x-3s}{4(n-s)}\right)} & \text{si } \frac{3t+t_x-3s}{4(n-s)} \le 1/2 \\ 3^{t_x}\ 2^{(n-s)} & \text{sinon} \end{cases}$$

Dans les deux cas, le terme maximum est atteint pour t_x maximal : $t_x = t$.

Cela signifie donc que le maximum est atteint pour un mot x de taille t, une intersection avec c de taille $t_I = \frac{3}{4}(t_x + s - t) = 3s/4$, donc une intersection avec y de taille $t_J = 2/3t_I = s/2$. Donc y est de poids $t_x + s - 2t_I + t_J = t$.

Cela donne un terme majoré par :

$$M_1(n,s,t) \leq \begin{array}{ll} 3^t\ 2^{(n-s)\,\mathrm{h}\left(\frac{4t-3s}{4(n-s)}\right)} & \text{si } \frac{4t-3s}{4(n-s)} \leq 1/2 \\ 3^t\ 2^{n-s} & \text{sinon} \end{array}$$

C'est à dire, en posant $S = \frac{s}{n}, x = \frac{t_I}{n}$:

$$m_1(S,T) \leq \begin{array}{ll} T\log 3 + (1-S)\,\mathrm{h}\left(\frac{4T-3S}{4(1-S)}\right) & \text{si } \frac{4T-3S}{4(1-S)} \leq 1/2 \\ T\log 3 + (1-S) & \text{sinon} \end{array}$$

Cas 2 : simplification de la somme Dans le cas où on a $\frac{t_{Jmax}}{t_I} \leq 2/3$, on va supposer que le terme maximum de la somme est atteint sur la sphère S_t, ce qui simplifie la majoration du terme principal ($t_x = t$). On obtient, une fois qu'on a majoré les termes binômiaux :

$$M_2(n,s,t) \leq 2^{(n-s)\,\mathrm{h}\left(\frac{t-t_I}{n-s}\right)+\log_2 3(t-t_I)+t_I\left(\mathrm{h}\left(\frac{2t_I-s}{t_I}\right)+2t_I-s\right)}$$

En prenant l'exposant de 2 divisé par n, on obtient :

$$T(t_I) = (1-\frac{s}{n})\,\mathrm{h}\left(\frac{t-t_I}{n-s}\right) + \log_2 3(\frac{t}{n} - \frac{t_I}{n}) + \frac{t_I}{n}\,\mathrm{h}\left(\frac{2t_I-s}{t_I}\right) + 2\frac{t_I}{n} - \frac{s}{n}$$

On pose $x = \frac{t_I}{n}$:

$$T(x) = (1-S)\,\mathrm{h}\left(\frac{T-x}{1-S}\right) + \log_2 3(T-x) + x\,\mathrm{h}\left(\frac{2x-S}{x}\right) + 2x - S$$

T a plusieurs valeurs pour lesquelles sa dérivée s'annule (voir les détails page 27) qui selon les valeurs de S ou de T peuvent correspondre au maximum. On les notera $x_{\max 1}, x_{\max 2}$.

Le terme maximum cherché est donc le max de celui en $x_{\max 1}$, celui en $x_{\max 2}$, voire de clui en 0, puisque la fonction peut être décroissante en 0.

$$m_2(S,T) \leq \max\left(T(x_{\max 1}), T(x_{\max 2}), (1-S)\,\mathrm{h}\left(\frac{T}{1-S}\right) + T\log_2 3 - S\right)$$

Enfin il faut prendre le max des deux cas étudiés précédemment :

$$m(S,T) \leq \max(m_1(S,T), m_2(S,T))$$

2.4 Majoration des a_s

On cherche à présent à calculer, pour un s donné, a_s : le nombre d'éléments de taille s dans S. On va utiliser un résultat sur les codes classiques, dans [1], en adaptant le cas à $\mathbb{F}_4^n$. On notera $\mathcal{A}(S,R) = \frac{1}{n}\log a_s$.

2.4.1 Majoration stricte

On traite donc S comme un code classique sur $\mathbb{F}_4$, de longueur n, de rendement $R = \dim_2(S)/n$ et de distance minimale $d = \delta n$. Soit $R^*(\delta)$ une borne supérieure sur le rendement des codes en fonction de δ. Soit également $\alpha \in [0, 1]$. On fera l'hypothèse suivante sur S :

Hypothèse 1. *On supposera que si $c \in S$, alors $\alpha c \in, \forall \alpha \in \mathbb{F}_4$.*

On commence par partitionner les coordonnées $\{1, \dots n\}$ de la sorte :

$$\{1, \dots n\} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \dots \mathcal{A}_t \cup \mathcal{B}$$

Avec comme conditions

$$\mathrm{rg}_{\mathbb{F}_4}(\text{colonnes d'indices dans } \mathcal{A}_i \text{de la matrice génératrice de } S) = \lfloor \alpha R n \rfloor$$

et

$$\mathrm{rg}_{\mathbb{F}_4}(\text{colonnes d'indices dans } \mathcal{B} \text{de la matrice génératrice de } S) < \alpha R n$$

Autrement dit, on forme des blocs de $\lfloor \alpha R n \rfloor$ coordonnées de telle sorte que les sous-matrices associées soient de rang plein sur $\mathbb{F}_4$, puis $\mathcal{B}$ représente les coordonnées restantes. t est le nombre maximum de tels sous-ensembles qu'on peut former. On notera $t\alpha R n = \beta n$

Si un mot de S est de poids s, alors par le principe des tiroirs, il existe au moins un ensemble $\mathcal{A}_i$ pour lequel ce mot a moins de $\lfloor s/t \rfloor$ entrées non nulles. On dira que ce mot est *mauvais* pour i. Pour un i donné, et un motif donné (sur l'ensemble $\mathcal{A}_i$), on compte

$$2^{nR - 2\alpha n R} = 2^{(1-2\alpha)Rn}$$

mots possibles de S ayant ce motif sur $\mathcal{A}_i$. En effet, le motif fixe $\alpha R n$ coordonnées, ce qui correspond à un nombre de possibilités "bloquées" $2^{2\alpha R n}$, car si la dimension sur 4 de l'espace engendré par les coordonnées $\mathcal{A}_i$ est $\alpha R n$, avec l'hypothèse 1, il est de dimension $2\alpha R n$ en tant que $\mathbb{F}_2$-espace vectoriel.

Cela nous fait donc, pour un $\mathcal{A}_i$ donné, le nombre maximum de mots qui sont mauvais pour i :

$$a_s \leq \sum_{i=0}^{\lfloor s/t \rfloor} \binom{\alpha R n}{i} 3^i 2^{(1-2\alpha)Rn}$$

On multiplie cette quantité par t pour obtenir une majoration sur le nombre total de mots de S de poids s (chaque mot est mauvais pour au moins un i). On majore ensuite cette somme :

$$\begin{aligned} a_s &\leq t \sum_{i=0}^{\lfloor s/t \rfloor} \binom{\alpha R n}{i} 3^i 2^{(1-2\alpha)Rn} \\ &\leq t \sum_{i=0}^{\lfloor s/t \rfloor} 2^{\alpha R n \, \mathrm{h}\left(\frac{i}{\alpha R n}\right) + i \log 3 + (1-2\alpha)Rn} \end{aligned}$$

Le terme maximum de la somme est :

– pour $\frac{i}{\alpha Rn} = 3/4$ si cette valeur est atteinte, autrement dit si $\frac{s}{\alpha tRn} \geq 3/4$,
– pour $i = s/t$ sinon

Il faut donc savoir quel est le m/t le plus grand possible, autrement dit le t le plus petit possible (ie le β le plus petit possible). Il est calculé dans le lemme 1.

Lemme 1. *La borne cherchée sur β, avec les notations ci-dessus est $\beta^*(\alpha)$ solution de l'équation :*

$$\frac{(1-\alpha)R}{\beta} = R^*\left(\frac{\delta}{\beta}\right)$$

Démonstration. En annexe, page 24. □

Si on revient au calcul de la borne supérieure, on a donc :

$$\begin{cases} R & \text{si } \frac{S}{\beta^*(\alpha)} \geq 3/4 \\ R\left[\alpha\left(\mathrm{h}\left(\frac{S}{\beta^*(\alpha)}\right) + \frac{S}{\beta^*(\alpha)}\right) + 1 - 2\alpha\right] & \text{sinon} \end{cases}$$

avec $S = s/n$.

Enfin, α a été fixé arbitrairement. On peut donc le faire varier entre 0 et 1/2, afin d'avoir le meilleur résultat possible. La borne cherchée est donc :

$$\mathcal{A}(S,R) \leq R \min_{0\leq\alpha\leq 1} \begin{cases} \alpha\left(\mathrm{h}\left(\frac{S}{\beta^*(\alpha)}\right) + \frac{S}{\beta^*(\alpha)}\right) + 1 - 2\alpha & \text{si } \frac{S}{\beta^*(\alpha)} \geq 3/4 \\ 1 & \text{sinon} \end{cases} \quad (8)$$

Dans notre cas, on prendra $R^*(\delta) = 1$, ce qui nous donne $\beta^*(\alpha) = R(1-2\alpha)$. On obtient donc :

$$\mathcal{A}(S,R) \leq$$
$$R \min_{0\leq\alpha\leq 1/2} \begin{cases} \alpha\left(\mathrm{h}\left(\frac{S}{R(1-2\alpha)}\right) + \frac{S}{R(1-2\alpha)}\right) + 1 - 2\alpha & \text{si } \frac{S}{R(1-2\alpha)} \geq 3/4 \\ 1 & \text{sinon} \end{cases} \quad (9)$$

2.4.2 Borne inf sur la borne sup (et estimation)

On peut donner une borne inférieure sur la borne supérieure sur $\frac{\log a_s}{n}$, en donnant quelques familles de codes simples. Dans [1], l'auteur conjecture cette borne comme la borne supérieure optimale.

Un premier exemple est : $\begin{pmatrix} M & 0 \end{pmatrix}$ où M a $k/2$ colonnes et k lignes, et est telle qu'elle soit de rang k sur $\mathbb{F}_2$ (et de rendement $R = k/n$). Le reste de la matrice génératrice est rempli de 0. De façon évidente, on peut générer tous les mots possibles sur la première partie, ce qui donne

$$a_s = \binom{k/2}{s} 3^s$$

pour $s \leq k/2$, ce qui donne avec la majoration des binômiaux :

$$2^{\frac{nR}{2}\left(\mathrm{h}\left(\frac{2S}{R}\right) + \frac{2S}{R}\log 3\right)}$$

Cette valeur est maximale pour $2S/R = 3/4$, c'est à dire $S = 3R/8$.

Pour $S \geq 3R/8$, on va fabriquer d'autres codes, et utiliser un argument probabiliste.

Lemme 2. *La répartition moyenne des poids des éléments des codes de longueur n et de rendement $R = k/n$ est donnée par :*

$$\bar{a_t} = \frac{\binom{n}{t} 3^t}{\frac{2^{2n}-1}{2^k-1}}$$

Démonstration.

$$\bar{a_t} = \sum_{\mathrm{w}(s)=t} P[s \in C]$$

Il faut donc étudier $P[s \in C]$, pour s donné et C aléatoire. Or, si $s \neq 0$, alors cette quantité ne dépend pas de s. On écrira donc :

$$\begin{aligned}
\sum_s P[s \in C] &= \left(2^{2n}-1\right) P[s \in C, s \neq 0] + 1 \\
&= \sum_s \frac{\sum_C 1_{\{s \in C\}}}{\#\text{ codes}} \\
&= \frac{1}{\#\text{ codes}} \sum_C \sum_s 1_{\{s \in C\}} \\
&= \frac{1}{\#\text{ codes}} \sum_C 2^k \\
&= 2^k
\end{aligned}$$

□

On va prendre maintenant un code dont le support est de taille n' (mais toujours de longueur totale n) et de dimension k. On a donc le résultat ci-dessus avec n' à la place de n. On a le résultat (en moyenne, mais cela signifie qu'il existe des codes pour lesquels on est à cette valeur ou au dessus) :

$$\frac{1}{n} \log a_t = \frac{n'}{n} \left(\mathrm{h}\left(\frac{t}{n'}\right) + \log 3 \frac{t}{n'} - 2 \right) + \frac{k}{n} + O\left(\frac{1}{n 2^k}\right)$$

Si on prend un support $n' = 4/3t$ où t est le poids pour lequel on veut la borne inférieure, alors on obtient $\frac{1}{n} \log a_t \simeq R$. On peut avoir cette borne inférieure pour $n' \geq k/2$, parce le code doit être dimension k. On a donc, pour un poids supérieur à $3/8k$, une borne inférieure qui est à 2^{nR}. Cette borne est valable jusqu'à ce que $n' = n$, c'est à dire un poids de $3n/4$. Au delà, la borne importe peu. En effet, si la probabilité d'erreur est égale à $3/4$, alors la capacité est clairement nulle.

On a donc au final la borne inférieure suivante (pour l'exposant de 2 divisé par n) :

$$\mathcal{A}(S,R) \geq \begin{cases} \frac{R}{2}\left(\mathrm{h}(\frac{2S}{R}) + \log 3 \frac{2S}{R}\right) & \text{tant que } S \leq 3R/8 \\ R & \text{ensuite} \end{cases}$$

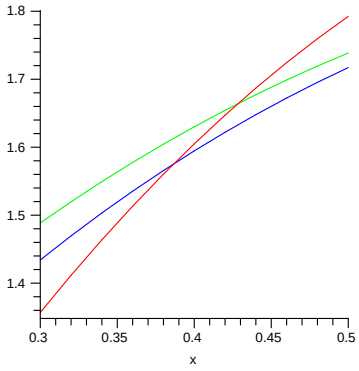

FIG. 2 – Bornes supérieures sur la somme (borne sup et borne inf sur la borne sup), et $|B_t|$ en comparaison (courbe du dessous)

Remarque 5. *Les codes utilisés ici pour atteindre la borne inf sur la borne sup ne sont pas à priori des cas qu'on rencontrera, entre autres ils ne vérifient pas la contrainte d'orthogonalité. Si on utilisait cette contrainte, on pourrait probablement avoir de meilleurs résultats.*

2.5 Majoration finale : résultats

On rappelle qu'on cherchait à majorer

$$\sqrt{2^k \left(\sum_{s=1}^{2t} M(s,t) a_s + |B_t| \right)}$$

On prend donc l'exposant de 2 divisé par n, et on majore les sommes par l'exposant max. Cela donne donc :

$$\frac{1}{2} \left(R + \sup_{S \leq 2T} (m(S,T) + \mathcal{A}(S,R)) \right)$$

et on cherche à savoir quand cette borne devient inférieure à $\frac{1}{n} \log |B_t|$.

On obtient, avec une résolution numérique, les courbes données sur la figure 2. Les deux courbes de borne sur la somme coupent la courbe de $|B_t|$ pour des valeurs non triviales de T (c'est à dire inférieures à $3/4$). On peut calculer ce point d'intersection, et on obtient environ 0.386717 pour la borne inférieure, et 0.428318 pour la borne supérieure.

3 Conclusion

Les bornes obtenues sont supérieures à celle (1/4) donnée par Preskill dans [2]. Cependant, le résultat n'est pas négatif pour autant. Déjà, il faut régler le problème des deux hypothèses faites au cours du raisonnement. La première, à propos du terme maximum de la somme : on a supposé que dans un des sous-cas, le maximum était atteint sur la sphère. La seconde est plus fine, c'est celle qui suppose que si x est dans S, alors $x*$tout élément de S est aussi dans S. Sans cette hypothèse, la preuve sur la majoration des a_s semble plus difficile à faire fonctionner – ou alors pas de cette façon. Il faudrait donc trouver une façon de raisonner sans cette hypothèse.

Ensuite, et surtout, on peut également améliorer ces bornes, notamment en utilisant l'hypothèse d'orthogonalité de S. Il est assez évident que les exemples utilisés pour la borne inf ne respectent pas cette hypothèse, on peut donc espérer de meilleures bornes sur a_s en exploitant cette piste.

En conclusion, il reste encore beaucoup de choses à chercher dans le domaine de la capacité des codes quantiques, avec cette méthode ou avec d'autres.

Remerciements

Je remercie Jean-Pierre Tillich, mon directeur de stage, qui a su me guider tout au long de ce stage. Je remercie également toute l'équipe SECRET, pour leur bonne humeur communicative.

Références

[1] Alexei E. Ashikhmin, Gérard D. Cohen, Michael Kirvelevich, and Simon N. Litsyn. Bounds on distance distributions in codes of known size. *IEEE*, 51(1), jan 2005.

[2] John Preskill. Lecture notes, quantum error correction. 1999.

A Annexes

A.1 Preuves

Preuve de la propriété 5, donnée page 8. On va montrer que la condition est suffisante.

Pour cela, il nous faut fabriquer une décomposition en sous-espaces orthogonaux dans lesquels effectuer la mesure, comme dans le cas de la condition (4). Les $E_a |i\rangle$ ne sont pas *à priori* orthogonaux entre eux, donc on ne peut pas prendre ces vecteurs tels quels. En fait, on cherche des opérateurs E'_a, tels qu'on puisse avoir une décomposition en $\oplus_{E_a} Vect\left((E'_a |i\rangle)_i\right)$, avec des $E'_a |i\rangle$ orthogonaux entre eux deux à deux.

Etant donnée un espace vectoriel $Vect\left((E_a\left|i\right\rangle)_a\right)$, on peut en déduire une base orthonormée $(\left|e\right\rangle)$. On sait qu'elle va se construire sous la forme $\left|e\right\rangle = \sum_a k_{ea} E_a \left|i\right\rangle$, donc ses vecteurs sont de la forme $E'_a \left|i\right\rangle$. Mieux, cette construction ne dépend que des produits scalaires $\left\langle i\right| E_a^\dagger E_b \left|i\right\rangle$, donc ne dépend pas de $\left|i\right\rangle$. Donc on a les mêmes E'_a pour tous les vecteurs du code.

Ainsi, on a des opérateurs E'_a qui vérifient $\left\langle i\right| {E'_a}^\dagger E'_b \left|i\right\rangle = \delta_{a,b}$. On peut donc effectuer la décomposition voulue, et comme on a :

$$E_a \left|i\right\rangle = \sum_c \lambda_{ac} E'_c \left|i\right\rangle ,$$

lors de la mesure, l'état va être projeté sur l'un de ces sous-espaces. Comme les $E_a \left|i\right\rangle$ étaient orthogonaux pour $i \neq j$, ça reste le cas pour les E'_a, et donc va se retrouver sur l'un des $E'_c \left|i\right\rangle$.

Le seul problème restant à régler est la correction de l'erreur : rien ne prouve que les $E'c$ soient unitaires. Cela dit, ils conservent le produit hermitien sur l'espace du code, par construction. On peut donc les étendre en opérateurs unitaires E''_a sur l'espace entier, et on associe à chaque sous-espace la correction correspondante : $E''a^\dagger$, puisque ${E''_a}^\dagger E'_a \left|i\right\rangle = \left|i\right\rangle \quad \forall \left|i\right\rangle \in C$.

□

Équivalence de définitions, cf remarque 2. On a $M \bigstar N =< m, n >= \sum_{i=1}^n \mathrm{tr}(m_i \bar{n_i})$ puisque tr est linéaire. Ce qui compte donc est la parité du nombre de 1 dans les $\mathrm{tr}(m_i \bar{n_i})$. D'autre part, puisque les matrices $\mathcal{M}$ et $\mathcal{N}$ se décomposent en produit tensoriel de "blocs" dont les produits commutent ou anti-commutent, on compte la parité du nombre de blocs qui ne commutent pas pour savoir si les deux matrices complètes commutent ou pas.

En résumé, il suffit de regarder ce qui se passe quand $n = 1$.

Quand l'une des deux matrices est I, alors ça commute. Pour la trace, on aura $m_i \bar{n_i} = 0$, et donc la trace est nulle aussi. Lorsqu'on a deux matrices qui ne sont pas l'identité, et donc qui anti-commutent, on regarde les différents cas :

$$\begin{array}{lcll} XY & \rightarrow & \omega(1 \bar{+} \omega) = \omega^2 = 1 + \omega & \rightarrow 1 \\ YX & \rightarrow & (1+\omega)(\bar{\omega}) = 1 + \omega^2 = \omega & \rightarrow 1 \\ XZ & \rightarrow & \omega \bar{1} = \omega & \rightarrow 1 \\ ZX & \rightarrow & 1\bar{\omega} = 1 + \omega & \rightarrow 1 \\ ZY & \rightarrow & 1(1 \bar{+} \omega) = 1 + \omega^2 = \omega & \rightarrow 1 \\ YZ & \rightarrow & (1+\omega)\bar{1} = 1 + \omega & \rightarrow 1 \end{array}$$

On a bien un 1 dans tous les cas d'anti-commutation et un 0 dans les cas de commutation. □

Preuve du théorème 1, page 9. Pour ce faire, on va utiliser le lemme suivant :

Lemme 3. *Pour S vu comme espace vectoriel sur $\mathbb{F}_2$, on a :*

$$\dim(S^\perp) = 2n - \dim(S)$$

où $S^\perp = \{M, M\bigstar N = 0 \quad \forall N \in S\}$. Contrairement à un orthogonal "classique", là rien n'empêche S lui-même d'être dans cet espace, ce qui arrive notamment quand S est commutatif.

Démonstration. Pour $N \in S$, on a :

$$M\bigstar N = 0$$
$$\sum_i \mathrm{tr}(m_i\bar{(}n_i)) = 0$$

L'opération $x \in \mathcal{F}_2^2 \Rightarrow x\bar{n_i}$ est linéaire, de même que l'opération tr. On a donc, pour un N donné, une forme linéaire qu'on appellera $\mathcal{A}_N$. Cela nous donne donc une application de G_n dans son dual, et donc, si on a k générateurs indépendants dans S, alors on aura k générateurs de formes linéaires dans le dual. Ainsi $M \in \S^\perp$ est caractérisé par k relations indépendantes de la forme : $\mathcal{A}_N(M) = 0$.

On a donc $2n$ inconnues, k relations linéaires indépendantes, donc la dimension sur $\mathcal{F}_2$ de $S^\perp$ est de $2n - k = 2n - \dim(S)$. □

Pour prouver le théorème, on va procéder par récurrence.

Si le stabilisateur contient juste l'identité, alors tout l'espace est le code, ce qui donne bien une dimension 2^n.

Supposons que le stabilisateur soit $< \mathcal{M}_1, \ldots \mathcal{M}_k >$ et que la propriété soit vraie pour tout $k' < k$. Soit $S =< \mathcal{M}_1, \ldots \mathcal{M}_k >$ et $S' =< \mathcal{M}_1, \ldots \mathcal{M}_{k-1} >$. On appelle C et C' les codes respectifs (on a $C \subset C'$). Par hypothèse de récurrence, C' est de dimension 2^{n-k+1}.

On sait donc que $\dim S = \dim S' + 1$, et donc que $\dim S'^\perp = \dim S^\perp - 1$, par le lemme 3. Il existe donc une matrice $\mathcal{N} \in S'^\perp \backslash S^\perp$, autrement dit il existe une matrice $\mathcal{N}$ qui commute avec tout S', mais pas avec tout S. Autrement dit, $\mathcal{N}$ commute avec $\mathcal{M}_1, \ldots \mathcal{M}_{k-1}$ et anti-commute avec $\mathcal{M}_k$.

Pour un $|\psi\rangle \in C$:

$$\mathcal{M}_k(\mathcal{N}|\psi\rangle) = -\mathcal{N}\mathcal{M}_k|\psi\rangle = -(\mathcal{N}|\psi\rangle)$$

Et si $|\psi\rangle \in C' \backslash C$:

$$\mathcal{M}_k(\mathcal{N}|\psi\rangle) = -\mathcal{N}\mathcal{M}_k|\psi\rangle = (\mathcal{N}|\psi\rangle)$$

Du fait que $\mathcal{N}$ commute avec tous les autres $\mathcal{M}_i, i < k$, l'opération "multiplication par $\mathcal{N}$" est une bijection de C' dans lui-même. Les observations ci-dessus montrent que cette bijection envoie les éléments de C sur les éléments de $C' \backslash C$ et inversement : elle coupe en deux l'espace du code C'. Il y a donc deux fois moins de vecteurs de base dans C que dans C', donc sa dimension est 2^{n-k}.

□

Équivalence des définitions 9 et 10 page 11. Montrons d'abord le sens $d = d' \Rightarrow C_{ab} = \delta_{a,b}$.

Soient $E_a \neq E_b$ dans l'ensemble E défini plus haut, et $|\psi\rangle$ un vecteur du code. On veut montrer que $\langle\psi| E_a^\dagger E_b |\psi\rangle = 0$. Pour cela, on va chercher un $\mathcal{M} \in S$ tel que $\mathcal{M}$ commute avec E_a et pas E_b, ou l'inverse. En effet, si on dispose d'un tel opérateur, on aura :

$$\begin{aligned}\langle\psi| E_a^\dagger E_b |\psi\rangle &= \langle\psi| E_a^\dagger \mathcal{M}^\dagger \mathcal{M} E_b |\psi\rangle \\ &= -\langle\psi| \mathcal{M}^\dagger E_a^\dagger E_b \mathcal{M} |\psi\rangle \\ &= -\langle\psi| E_a^\dagger E_b |\psi\rangle\end{aligned}$$

D'où ce produit est égal à 0.

Si on avait $s(E_a) = s(E_b)$, comme les erreurs E_a et E_b sont supposées pouvoir se corriger, cela signifierait qu'elles sont corrigées par la même opération $\mathcal{M}_1$. On a donc $E_a^\dagger E_b \in S$. Or le poids de $E_a^\dagger E_b$ est inférieur à $2t$ donc strictement inférieur à d. De plus cette erreur appartient à S et n'est pas l'identité. Donc elle appartient à $\{E \in S^\perp, E \neq I\}$. Donc $d' \leq \mathrm{w}(E_a^\dagger E_b) < d$, ce qui est contraire à l'hypothèse.

En revanche, si $s(E_a) \neq s(E_b)$, on prend $\mathcal{M}_i$ tel que $s(E_a)_i \neq s(E_b)_i$, ce qui signifie bien que $\mathcal{M}_i$ commute avec l'une des erreurs et pas avec l'autre, d'où le résultat.

Sens $C_{ab} = \delta_{a,b} \Rightarrow d = d'$.

Supposons, par l'absurde que $d > d'$. Alors il existe une erreur E, qui est dans S, de poids inférieur à d et qui n'est pas l'identité. Alors on peut "couper" E en deux erreurs de poids chacune $\leq t$, $E_a^\dagger$ et E_b. Ces erreurs étant de poids $\leq t$, elles sont dans l'ensemble d'erreurs dites "correctibles". Or, pour $|\psi\rangle \in C$, on a :

$$\begin{aligned}\langle\psi| E_a^\dagger E_b |\psi\rangle &= \langle\psi| E |\psi\rangle \\ &= \langle\psi| |\psi\rangle \\ &= 1\end{aligned}$$

ce qui contredit l'hypothèse $\langle\psi| E_a^\dagger E_b |\psi\rangle = 0$ si $E_a \neq E_b$ (et c'est le cas sinon E serait l'identité). D'où la conclusion.

□

Preuve du lemme 1, page 18. On définit le code D, à partir de S de la façon suivante :

- La longueur de D est $\alpha R t n = \beta n$. Autrement dit on supprime les coordonnées de $\mathcal{B}$.
- Les mots de code sont ceux de S dont le support appartient à $\mathcal{A}_1 \cup \ldots \cup \mathcal{A}_t$.

Cherchons maintenant quelle est le rendement de ce code et sa distance minimale. On fera l'hypothèse (notée) que le code S est stable par multiplication par tout élément de $\mathbb{F}_4$.

$$R' = \frac{\dim_{\mathbb{F}_2} D}{\beta n}$$

La dimension sur $\mathbb{F}_2$ du code est celle de S moins la dimension de l'espace engendré par tous les mots de code dont le support n'est pas seulement dans $\mathcal{A}_1 \cup \ldots \cup \mathcal{A}_t$. Or la dimension de cette espace, sur F_4, est majorée par αRn vue la construction des $\mathcal{A}_i$. Sur $\mathbb{F}_2$, cette dimension est donc au maximum de $2\alpha Rn$. D'où

$$R' \geq \frac{Rn - 2\alpha Rn}{\beta n} = \frac{R(1-2\alpha)}{\beta}$$

Sa distance minimale est donnée par le plus petit élément (au sens du poids sur $\mathbb{F}_4$) non nul de D. Clairement, si un tel élément atteint la distance minimale dans D, ce même élément complété par des 0 est un élément de S et a le même poids. D'où

$$d' \geq d \iff \frac{d'}{\beta n} = \delta' \geq \frac{\delta}{\beta}$$

On obtient donc les inégalités suivantes, puisque la borne R^* est décroissante :

$$\frac{R(1-2\alpha)}{\beta} \leq R' \leq R^*\left(\frac{\delta}{\beta}\right) \tag{10}$$

Pour que l'inégalité soit possible, il faut que le terme de gauche soit inférieur au terme de droite. Celui de gauche est clairement décroissant en β, tandis que celui de droite est croissant. Pour $\beta = 1$, le terme de gauche est clairement inférieur au terme de droite, et pour $\beta = 2\delta$, le terme de droite est nul alors que le terme de gauche est positif : β a donc une valeur limite minimale, qui correspond à la racine de $\frac{R(1-2\alpha)}{\beta} = R^*\left(\frac{\delta}{\beta}\right)$ qu'on notera $\beta^*(\alpha)$. □

A.2 Exemple

Exemple d'encodage et de décodage

Prenons l'exemple du code à trois qubits (1). Sur la figure 3, on peut voir la construction pratique du code à 3 qubits, avec deux portes CNOT, et les deux qubits supplémentaires. On peut voir que $|0\rangle$ est envoyé sur $|000\rangle$ et $|1\rangle$ sur $|111\rangle$.

De là, si on applique le circuit inverse à la réception, c'est à dire comme sur la figure 4, si aucune erreur n'a eu lieu, la mesure des deux qubits supplémentaires donne 00. S'il y a eu des erreurs, cette mesure peut changer, et on va voir qu'en fait, cette mesure donne exactement le syndrome comme défini plus haut.

En effet, si l'erreur qui est apparue est IIX, ie au lieu de $|000\rangle, |111\rangle$ on a $|001\rangle, |110\rangle$, alors le syndrome mesuré est 01. Si l'erreur est IXI, le syndrome est 11, si c'est XII, le syndrome est 10. Le syndrome correspond bien, ici, à la définition 7, avec $\mathcal{M}_1 = ZZI, \mathcal{M}_2 = IZZ$.

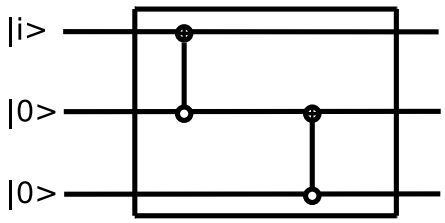

FIG. 3 – Circuit d'encodage pour le code à trois qubits

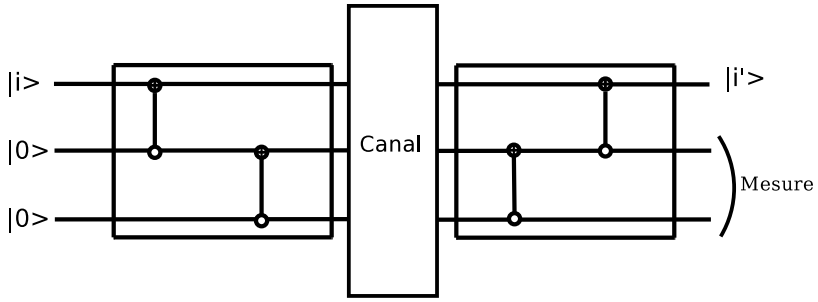

FIG. 4 – Encodage et décodage pour le code à 3 qubits

A.3 Détails de calculs

Maximum de la fonction $a(x)$

Soit la fonction $a(x) = \mathrm{h}(x) + x$. Étudions les variations de la fonction a entre 0 et 1.

$$\begin{aligned} a'(x) &= -1 - \ln x + 1 + \ln(1-x) + \ln 2 \\ &= \ln(\frac{1-x}{x}) + \ln 2 \end{aligned}$$

Cette fonction tend vers $+\infty$ quand x tend vers 0, et vers $-\infty$ quand x tend vers 1.

$$\begin{aligned} a'(x) &= 0 \\ -\ln 2 &= \ln(\frac{1-x}{x}) \\ \frac{1}{1-x} &= 3 \\ x &= 2/3 \end{aligned}$$

On a donc un unique maximum en $x = 2/3$: $\log 3$.

Maximum de la fonction T

$T(x) = T(x) = (1-S)\,\mathrm{h}\left(\frac{T-x}{1-S}\right) + \log_2 3(T-x) + x\,\mathrm{h}\left(\frac{2x-S}{x}\right) + 2x - S$

Avec $\mathrm{h}'(x) = \frac{1}{\ln 2}\ln(\frac{1}{x} - 1)$, on a :

$$\begin{aligned} T'(x) &= (1-S)\frac{1}{\ln 2}\frac{-1}{1-S}\ln\left(\frac{1-S}{T-x} - 1\right) - \log_2 3 \\ &\quad + x * \frac{1}{\ln 2} * \frac{S}{x^2}\ln\left(\frac{x}{2x-S} - 1\right) + \mathrm{h}\left(\frac{2x-S}{x}\right) + 2 \\ &= -\frac{1}{\ln 2}\ln\left(\frac{1-S-T+x}{T-x}\right) \\ &\quad + \frac{S}{x\ln 2}\ln\left(\frac{S-x}{2x-S}\right) \\ &\quad - \frac{1}{\ln 2}\left(\frac{2x-S}{x}\ln\left(\frac{2x-S}{x}\right)\right) \\ &\quad - \frac{1}{\ln 2}\left(\frac{S-x}{x}\ln\left(\frac{S-x}{x}\right)\right) \\ &\quad + 2 - \log_2 3 \\ &= 2 - \log_2 3 \\ &\quad + \frac{1}{\ln 2}\ln\left(\frac{T-x}{1+x-S-T}\ \frac{x^2}{(2x-S)^2}\ \frac{S-x}{x}\right) \\ &\quad + \frac{S}{x\ln 2}\ln\left(\frac{S-x}{2x-S}\ \frac{2x-S}{x}\ \frac{x}{S-x}\right) \\ &= 2 - \log_2 3\ +\ \frac{1}{\ln 2}\ln\left(\frac{x(x-T)(x-S)}{(2x-S)^2(1+x-S-T)}\right) \end{aligned}$$

Cette fonction s'annule pour x tel que :

$$\begin{aligned} & \frac{x(x-T)(x-S)}{(2x-S)^2(1+x-S-T)} = 2^{-2+\log_2 3} \\ \iff\quad & x(x-T)(x-S) - \tfrac{3}{4}(2x-S)^2(1+x-S-T) = 0 \end{aligned}$$

De plus, ce polynôme p a 3 racines réelles. En effet, si on regarde quelques valeurs :

$$p(0) = -\frac{3}{4}(-S)^2(1-S-T) \leq 0$$

$$\begin{aligned} p(\frac{S}{2}) &= \frac{S}{2}(\frac{S}{2}-T)(-\frac{S}{2}) - \frac{3}{4}(2\frac{S}{2}-S)^2(1+x-S-T) \\ &= \frac{S^2}{4}(T-\frac{S}{2}) \geq 0 \end{aligned}$$

$$\begin{aligned} p(S) &= -\frac{3}{4}(s)^2(1+S-S-T) \\ &= -\frac{3}{4}(S)^2(1-T) \leq 0 \end{aligned}$$

De plus, p tend vers $+\infty$ en $-\infty$. Donc p s'annule une fois pour une valeur négative, puis une autre fois entre 0 et $S/2$, et une troisième fois entre $S/2$ et S. Selon le signe de $(1+x-S-T)$, une des deux racines donnera le maximum.

Avec un logiciel de calcul formel, on obtient :

$$x_1 = \frac{P(S,T)^{1/3}}{12} - \frac{12Q(S,T)}{P(S,T)^{1/3}} + \frac{T}{3} + \frac{5S}{6} - \frac{1}{2}$$

$$\begin{aligned} x_2 = &-\frac{P(S,T)^{1/3}}{24} + \frac{6Q(S,T)}{P(S,T)^{1/3}} + \frac{T}{3} + \frac{5S}{6} - \frac{1}{2} \\ &+\frac{1}{2}i\sqrt{3}\left(\frac{P(S,T)^{1/3}}{12} + \frac{12Q(S,T)}{P(S,T)^{1/3}}\right) \end{aligned}$$

$$\begin{aligned} x_3 = &-\frac{P(S,T)^{1/3}}{24} + \frac{6Q(S,T)}{P(S,T)^{1/3}} + \frac{T}{3} + \frac{5S}{6} - \frac{1}{2} \\ &-\frac{1}{2}i\sqrt{3}\left(\frac{P(S,T)^{1/3}}{12} + \frac{12Q(S,T)}{P(S,T)^{1/3}}\right) \end{aligned}$$

Avec

$$\begin{aligned} P(T,S) &= 192ST^2 + 264S^2T - 576ST - 234S^2 + 432S - 26S^3 + 64T^3 \\ &- 288T^2 + 432T - 216 \\ &+ 6\sqrt{\begin{array}{l} -1296S^2T + 648S^3 + 2592S^2T^2 + 1296S^3T - 2592S^3T^2 \\ +360S^4T - 1728S^2T^3 + 960S^3T^3 + 672S^4T^2 + 384S^2T^4 \\ -648S^5T - 1323S^4 + 738S^5 - 9S^6 \end{array}} \\ Q(S,T) &= -\frac{2}{9}ST + \frac{1}{3}S - \frac{5}{72}S^2 - \frac{1}{9}T^2 + \frac{1}{3}T - \frac{1}{4} \end{aligned}$$

Les racines carrées sont des racines au sens complexe, car les quantités sous la racine ne sont pas forcément positives.

La racine qui nous importe est la plus grande, vu le raisonnement précédent. Il est difficile d'intuiter laquelle est la bonne, on définit donc $x_{\max 1}(S,T) = \min\left(3S/4, \max\left(x_1, x_2, x_3\right)\right)$. On a ajouté aussi la condition $t_I \leq 3/4s$ (sinon on est dans le cas "simple"), $x_{\max 2}(S,T) = 2\text{e terme}(x_1, x_2, x_3)$.