Rapport AC20

Corentin HAUTEFAYE

Janvier 2025

Introduction

Dans le cadre de l'UE AC20 (*Semestre A24*), j'ai choisi de travailler sur de l'informatique quantique. En effet, il s'agit d'un problème important dans le domaine de la recherche. Ainsi, ce domaine a potentiellement la capacité de remettre en cause la manière dont on traite l'information, mais également comment traiter un problème complexe qui dépasse les limites des ordinateurs classiques.

J'ai trouvé de plus fortement intéressant de travailler sur cette manière, dont j'ignorais beaucoup auparavant. En sus de cela, j'ai apprécié ce sujet dans la mesure où les domaines des Mathématiques, de la Physique et de l'Informatique sont enchevêtrés.

Avant de continuer, je tiens également à remercier M. Holweck pour m'avoir donné l'opportunité de réaliser ce travail, mais également pour le temps (*notamment les mardis après-midi*) et les conseils qu'il m'a accordés. J'aurais eu beaucoup plus de difficultés sans sa guidance.

En somme, l'objectif de ce rapport est de fournir une vue globale des principes fondamentaux de l'informatique quantique, tout en présentant ses applications et ses intérêts. Dans un premier temps, les notions mathématiques les plus importantes sont présentées, afin que le lecteur puisse saisir les théorèmes de calcul qui suivent. Ensuite, la partie qui suit porte sur l'algorithmie classique et le formalisme utilisé pour caractériser un problème de difficile ou non. Quelques notions sur les portes logiques sont également présentées. Puis, une partie détaille les axiomes de calcul quantiques sur des systèmes quantiques composés. Les conventions de circuit quantique sont explorées. De plus, on aborde également des algorithmes quantiques célèbres qui mettent en évidence les points sur lesquels l'informatique quantique ont un intérêt. Enfin, on propose dans la dernière partie de mettre informatique classique et informatique quantique côte à côte sur un problème, afin de mener une analyse.

Table des matières

Chapitre 1

Notions Mathématiques

Définition 1. On définit le symbole de Kronecker, comme l'application $\begin{array}{rccl} \delta_{\cdot,\cdot} : & \mathbb{Z} & \to \mathbb{Z} & \\ & (i,j) & \mapsto \begin{cases} 1 & \text{si } i=j \\ 0 & \text{sinon} \end{cases} & . \end{array}$

1.1 Structures algébriques

1.1.1 Lois de composition

Définition 2. On appelle loi de composition interne sur un ensemble E toute application $\perp$ de $E \times E$ dans E. Si (x, y) est un couple d'éléments de E, on note $x \perp y$ l'image de (x, y) par $\perp$.

Définition 3. Soit E et F deux ensembles. On appelle loi de composition externe sur E à domaine d'opérateurs F toute application $\top$ de $F \times E$ dans E. Si (ω, x) est un couple d'éléments de $F \times E$, on note $\omega \top x$ l'image de (ω, x) par $\top$.

Définition 4. Soit E un ensemble muni d'une loi de composition interne $\perp$. Une partie A de E est dite stable par $\perp$ si et seulement si pour tout $(x, y) \in A^2, (x \perp y) \in A$.

Définition 5. Soit E un ensemble muni d'une loi de composition interne $\perp$ et A une partie de E stable par $\perp$. L'application $\begin{array}{ll} A \times A & \to A \\ (x,y) & \mapsto x \perp y \end{array}$ est correctement définie et est une loi de composition interne sur A, que l'on appelle loi de composition interne induite par $\perp$ sur A. Notons tout de même que le fait de noter cette application par $\perp$ ne porte pas à confusion, malgré des domaines de départ et d'arrivée différents.

Définition 6. Soit $\perp$ une loi de composition interne sur un ensemble E. On dit que la loi $\perp$ est associative si et seulement si pour tout $(x, y, z) \in E^3, (x \perp y) \perp z = x \perp (y \perp z)$. On dit que la loi $\perp$ est commutative si et seulement si pour tout $(x, y) \in E^2, x \perp y = y \perp x$.

Remarque 1. Soit E un ensemble muni d'une loi de composition interne $\perp$. Si $\perp$ est associative, alors pour tout $(x, y, z) \in E^3$, on note $x \perp y \perp z$ l'élément $x \perp (y \perp z)$.

Définition 7. Soit E un ensemble muni de deux lois de composition interne notées $\perp$ et $\star$. On dit que $\perp$ est distributive par rapport à $\star$ si et seulement si pour tout $(x, y, z) \in E^3, \begin{cases} x \perp (y \star z) & = (x \perp y) \star (x \perp z) \\ (y \star z) \perp x & = (y \perp x) \star (z \perp x) \end{cases}$.

Définition 8. Soit $\perp$ une loi de composition interne sur un ensemble E. On appelle élément neutre pour $\perp$ un élément e de E tel que pour tout $x \in E, x \perp e = e \perp x = x$.

Proposition 1. *Soit E un ensemble muni d'une loi de composition interne $\perp$. Si E possède un élément neutre pour $\perp$ alors celui-ci est unique.*

Démonstration. Soit e et e' deux éléments neutres de E pour $\perp$. Par définition, comme e est neutre, on a $e \perp e' = e'$. De même, on a $e \perp e' = e$. Donc $e = e'$, d'où le résultat. □

Définition 9. Soit E un ensemble muni d'une loi de composition interne $\perp$ possédant un neutre que l'on note e. On dit qu'un élément a de E est symétrisable si et seulement s'il existe $a' \in E$ tel que $a \perp a' = a' \perp a = e$.

Proposition 2. *Soit E un ensemble muni d'une loi de composition interne associative $\perp$ possédant un neutre e et a un élément de E supposé ici symétrisable. Alors il existe un unique élément a' de E tel que $a \perp a' = a' \perp a = e$. Ainsi a' est appelé le symétrique de a par $\perp$.*

Démonstration. Soit a' et a'' deux éléments de E qui soient des symétriques de a pour $\perp$. Par définition, $(a' \perp a) \perp a'' = e \perp a'' = a''$ et $a' \perp (a \perp a'') = a' \perp e = a'$. Or par associativité de $\perp$, on en tire $a' = a''$ d'où l'unicité du symétrique. □

Remarque 2. Les deux définitions précédentes se simplifient dans le cas où les lois considérées sont commutatives. De plus, il est à noter que l'élément neutre d'un ensemble E muni d'une loi de composition interne $\perp$, s'il existe, est un élément symétrisable de E qui est son propre symétrique.

Proposition 3. *Soit E un ensemble muni d'une loi de composition interne associative $\perp$ possédant un neutre e. Soit a et b deux éléments symétrisables de E dont on note les symétriques respectifs par a' et b'. Ainsi l'élément $a \perp b$ de E est symétrisable et son symétrique est $b' \perp a'$.*

Démonstration. Par associativité, $(a \perp b) \perp (b' \perp a') = a \perp (b \perp b') \perp a' = a \perp e \perp a' = a \perp a' = e$, donc $(a \perp b) \perp (b' \perp a') = e$. De même pour $(b' \perp a') \perp (a \perp b) = e$. L'unicité du symétrique d'un élément de E permet de conclure. □

Proposition 4. *Soit E un ensemble muni d'une loi de composition interne associative $\perp$ admettant un neutre et a un élément symétrisable de E. Deux éléments x et y de E tels que $a \perp x = a \perp y$ sont nécessairement égaux. De même pour deux éléments x et y de E tels que $x \perp a = y \perp a$.*

Démonstration. Soit $(x, y) \in E^2$ tel que $x \perp a = y \perp a$. Notons a' le symétrique de a. Donc $(x \perp a) \perp a' = (y \perp a) \perp a'$, d'où par associativité $x \perp (a \perp a') = y \perp (a \perp a')$. Comme $a \perp a' = e$, on en tire $x = y$. De manière analogue, on montre l'autre résultat. □

1.1.2 Structure de groupe

Définition 10. On appelle groupe tout ensemble muni d'une loi de composition interne associative, possédant un neutre et pour laquelle tout élément est symétrisable. Un groupe est dit abélien si et seulement si sa loi est commutative.

Définition 11. Soit $(G, .)$ un groupe. On appelle sous-groupe de G toute partie H de G non vide, stable par la loi de composition interne de G et qui contient le symétrique de tous ses éléments.

Remarque 3. Soit $(G, .)$ un groupe. Un sous-groupe de G muni de la loi de composition interne induite par celle de G est lui-même un groupe. Il contient nécessairement le neutre de G et ce dernier est son propre neutre.

Définition 12. Soit $(G, .)$ et $(G', \star)$ deux groupes. On appelle morphisme de groupes de G vers G' toute application $f : G \to G'$ telle que pour tout $(x, y) \in G^2, f(x.y) = f(x) \star f(y)$. On appelle isomorphisme de groupes tout morphisme de groupes bijectif.

Proposition 5. *Soit $(G, .)$ et $(G', \star)$ deux groupes et f un morphisme de groupes de G vers G'.*
- *Si e désigne le neutre de G alors $f(e)$ est le neutre de G'.*
- *Pour tout $x \in G$, l'image par f du symétrique de x est le symétrique de $f(x)$.*

Démonstration.
- Notons e et e' les neutres respectifs de G et G'. Comme f est un morphisme de groupes, $f(e.e) = f(e) \star f(e)$ d'où $f(e) \star e' = f(e) \star f(e)$. Ainsi, on a nécessairement $f(e) = e'$.
- Soit $x \in G$ dont on note x' le symétrique. Comme f est un morphisme de groupes, $f(x.x') = f(x) \star f(x')$, d'où $f(e) = f(x) \star f(x')$. Or, le point précédent assure que $f(x) \star f(x') = e'$. De manière analogue, on montre que $f(x') \star f(x) = e'$. Ainsi, $f(x')$ est le symétrique de $f(x)$.

□

Proposition 6. *Soit $(G,.)$ et $(G',\star)$ deux groupes et f un morphisme de groupes de G vers G'.*
- *L'image directe par f d'un sous-groupe H de G est un sous-groupe de G'.*
- *L'image réciproque par f d'un sous-groupe H' de G' est un sous-groupe de G.*

Démonstration. ■ Notons e et e' les neutres respectifs de G et G'. Comme $e \in H$, $f(e) \in f(H)$. La proposition précédente assure que $e' \in f(H)$. Soit $(x',y') \in (f(H))^2$. Ainsi par définition, il existe $(x,y) \in H^2$ tel que $x' = f(x)$ et $y' = f(y)$. Comme f est un morphisme de groupes, on a :

$$x' \star y'^{-1} = f(x) \star f(y)^{-1} = f(x) \star f(y^{-1}) = f(x.y^{-1})$$

Comme H est un sous-groupe de G, $x.y^{-1} \in H$. Ainsi, $x' \star y'^{-1} \in f(H)$. Finalement, $f(H)$ est un sous-groupe de G'.

■ Preuve analogue à la première.

□

Définition 13. Soit $(G,.)$ et $(G',\star)$ deux groupes et f un morphisme de groupes de G vers G'.
- On appelle image de f et on note $\mathrm{Im}(f)$ l'image directe par f de G.
- Si on note e le neutre de G', on appelle noyau de f et on note $\mathrm{Ker}(f)$ l'image réciproque par f de $\{e\}$.

Proposition 7. *Soit $(G,.)$ et $(G',\star)$ deux groupes et f un morphisme de groupes de G vers G'.*
- *L'image de f est un sous-groupe de G', égal à G' si et seulement si f est surjectif.*
- *Le noyau de f est un sous-groupe de G, réduit au neutre de G si et seulement si f est injectif*

1.1.3 Anneaux et corps

Définition 14. On appelle anneau tout triplet $(A,+,\times)$, où A est un ensemble muni de deux lois de composition interne usuellement notées $+$ et $\times$, et respectivement dénommées addition et multiplication, qui vérifie les axiomes suivants :

1. $(A,+)$ est un groupe abélien. Son neutre, noté 0, est appelé élément nul.
2. $\times$ est associative et possède un neutre, noté 1, et appelé élément unité.
3. La multiplication est distributive par rapport à l'addition.

De plus, on dit que $(A,+,\times)$ est commutatif si et seulement si sa multiplication est commutative. On appelle anneau nul tout anneau contenant un seul élément.

Définition 15. Soit $(A,+,\times)$ un anneau commutatif. On appelle diviseur de 0 tout élément non nul x de A tel qu'il existe un élément non nul y de A et qui vérifie $xy = 0$. Un anneau est dit intègre si et seulement s'il est commutatif et sans diviseur de 0.

Définition 16. Soit $(A,+,\times)$ un anneau. On appelle caractéristique de A le plus petit entier naturel non nul n tel que $n \times 1_A = 0_A$. Si un tel entier n'existe pas, on dit que A est de caractéristique nulle.

Définition 17. On appelle corps tout anneau commutatif non nul où tout élément distinct de 0 est symétrisable pour $\times$.

Définition 18. Soit $(A,+,\times)$ un anneau. On appelle sous-anneau de A toute partie B de A contenant 1, stable par $\times$ et telle que B soit un sous-groupe de $(A,+)$.

Définition 19. Soit $(A,+,\times)$ et $(A',+',\times')$ deux anneaux dont note 1_A et $1_{A'}$ les éléments unités respectifs. On appelle morphisme d'anneaux de A vers A' toute application $f : A \to A'$ telle que $f(1_A) = 1_{A'}$ et pour tout $(x,y) \in A^2, \begin{cases} f(x+y) &= f(x) +' f(y) \\ f(x \times y) &= f(x) \times' f(y) \end{cases}$. On appelle isomorphisme d'anneaux tout morphisme d'anneaux bijectif.

Proposition 8. *Soit $(A,+,\times)$ un anneau.*

1. *Pour tout $x \in A, x \times 0 = 0$ et $0 \times x = 0$.*
2. *Pour tout $(x,y) \in A^2, (-x)y = x(-y) = -(xy)$.*

Proposition 9. *Soit $(A, +, \times)$ un anneau. L'ensemble des éléments inversibles de A muni de la loi $\times$ est un groupe.*

Proposition 10. *Soit $(A, +, \times)$ un anneau. Soit $(n, a, b) \in \mathbb{N}^* \times A^2$ tel que $ab = ba$.*

1. $(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$
2. $a^n - b^n = (a-b) \sum_{k=0}^{n} a^k b^{n-1-k}$

Remarque 4. L'ensemble des matrices carrées muni des lois usuelles est un anneau.

1.1.4 Espaces vectoriels

Définition 20. Soit $(\mathbb{K}, +, \times)$ un corps. On appelle $\mathbb{K}$-espace vectoriel tout triplet $(E, \oplus, \perp)$, où E est un ensemble muni d'une loi de composition interne $\oplus$ et d'une loi de composition externe $\perp$ à domaine d'opérateurs $\mathbb{K}$, qui vérifie les axiomes suivants :

1. $(E, \oplus)$ est un groupe abélien.
2. Pour tout $(\lambda, x, y) \in \mathbb{K} \times E^2, \lambda \perp (x \oplus y) = (\lambda \perp x) \oplus (\lambda \perp y)$.
3. Pour tout $(\lambda, \mu, x) \in \mathbb{K}^2 \times E, (\lambda + \mu) \perp x = (\lambda \perp x) \oplus (\mu \perp x)$.
4. Pour tout $(\lambda, \mu, x) \in \mathbb{K}^2 \times E, \lambda \perp (\mu \perp x) = (\lambda \times \mu) \perp x$.
5. Pour tout $x \in E; 1_{\mathbb{K}} \perp x = x$.

Les éléments de E sont appelés vecteurs et ceux de $\mathbb{K}$ sont appelés scalaires.

Définition 21. Soit $(\mathbb{K}, +, \times)$ un corps et $(E, \oplus, \perp)$ un $\mathbb{K}$-espace vectoriel. On appelle sous-espace vectoriel de E toute partie F de E qui contient le neutre pour l'addition de E et telle que pour tout $(\lambda, x, y) \in \mathbb{K} \times F^2, (\lambda x + y) \in F$.

Remarque 5. Soit $\mathbb{K}$ un corps et E un $\mathbb{K}$-espace vectoriel. Par définition, pour tout $(\lambda, x) \in \mathbb{K} \times E, (\lambda.x = 0) \iff (x = 0 \text{ ou } \lambda = 0)$.

Proposition 11. *Soit $\mathbb{K}$ un corps et E un $\mathbb{K}$-espace vectoriel. L'intersection de sous-espaces vectoriels de E est un sous-espace vectoriel de E.*

Remarque 6. Ce résultat est faux pour l'union.

1.2 Rappels d'algèbre linéaire

Dans cette partie, $\mathbb{K}$ désigne un corps et E un $\mathbb{K}$-espace vectoriel.

Définition 22. Soit $n \in \mathbb{N}^*$. Soit $\mathcal{F} = (e_i)_{i \in [\![1;n]\!]} \in E^n$. On dit que $\mathcal{F}$ est une famille libre si pour tout $((\lambda_i)_{i \in [\![1;n]\!]}, (\mu_i)_{i \in [\![1;n]\!]}) \in (\mathbb{K}^n)^2, (\sum_{k=1}^{n} \lambda_k e_k = \sum_{k=1}^{n} \mu_k e_k) \implies (\forall k \in [\![1;n]\!], \lambda_k = \mu_k)$. Le cas contraire, $\mathcal{F}$ est dite liée.

Proposition 12. *Soit $n \in \mathbb{N}^*$. Soit $\mathcal{F} = (e_i)_{i \in [\![1;n]\!]} \in E^n$. On dit que $\mathcal{F}$ est une famille libre si pour tout $(\lambda_i)_{i \in [\![1;n]\!]} \in \mathbb{K}^n, (\sum_{k=1}^{n} \lambda_k e_k = 0_E) \implies (\forall k \in [\![1;n]\!], \lambda_k = 0)$.*

Définition 23. Soit $n \in \mathbb{N}^*$. Soit $\mathcal{F} = (e_i)_{i \in [\![1;n]\!]} \in E^n$. Pour tout $(\lambda_i)_{i \in [\![1;n]\!]} \in \mathbb{K}^n$, on appelle combinaison linéaire de $\mathcal{F}$ toute expression $\sum_{k=1}^{n} \lambda_k e_k$.

Définition 24. Soit $U \in \mathcal{P}(E)$ de cardinal $n \in \mathbb{N}^*$. On appelle sous-espace vectoriel engendré par U, et on note $\text{Vect}(U)$, l'ensemble $\{\sum_{k=1}^{n} \lambda_k u_k, (\lambda_i)_{i \in [\![1;n]\!]} \in \mathbb{K}^n, (u_i)_{i \in [\![1;n]\!]} \in U^k\}$.

Proposition 13. *Soit $U \in \mathcal{P}(E)$. $\text{Vect}(U)$ est le plus petit sous-espace vectoriel au sens de l'inclusion qui contient U.*

Définition 25. Soit $\mathcal{F}$ une famille de vecteurs de E. On dit que $\mathcal{F}$ est génératrice si $\text{Vect}(\mathcal{F}) = E$.

Définition 26. On dit qu'une famille $\mathcal{F}$ de vecteurs de E est une base de E si elle est libre et génératrice. On appelle alors dimension de E et on note $\dim_{\mathbb{K}}(E)$ le cardinal commun à l'ensemble de ses bases.

1.3 Espaces de Hilbert

Dans cette partie, $\mathbb{K}$ désigne un corps.

1.3.1 Normes & Distances

Définition 27. Soit E un $\mathbb{K}$-espace vectoriel. On appelle norme sur E, et on note $\|.\|$, une application $E \to \mathbb{R}^+$ telle que :

1. Pour tout $x \in E, \|x\| = 0 \iff x = 0_E$.
2. Pour tout $(\lambda, x) \in \mathbb{K} \times E, \|\lambda x\| = \lambda \|x\|$.
3. Pour tout $(x, y) \in E^2, \|x + y\| \leq \|x\| + \|y\|$.

Définition 28. On appelle $\mathbb{K}$-espace vectoriel normé tout $\mathbb{K}$-espace vectoriel doté d'une norme.

Définition 29. Soit E un $\mathbb{K}$-espace vectoriel. On dit qu'une application $\|.\|$ de E dans $\mathbb{R}^+$ est une semi-norme sur E si elle vérifie l'inégalité triangulaire et est linéaire par rapport aux scalaires.

Définition 30. Soit E un ensemble. On appelle distance sur E toute application $d : E^2 \to \mathbb{R}^+$ telle que :

1. Pour tout $(x, y) \in E^2, d(x, y) = 0 \iff x = y$.
2. Pour tout $(x, y) \in E^2, d(x, y) = d(y, x)$.
3. Pour tout $(x, y, z) \in E^3, d(x, z) \leq d(x, y) + d(y, z)$.

Définition 31. On appelle espace métrique tout ensemble muni d'une distance.

Définition 32. Soit $(u_n)_{n\in\mathbb{N}}$ une suite complexe. u est dite de Cauchy ssi $\forall \epsilon \in \mathbb{R}^{+*}, \exists N \in \mathbb{N}, \forall (n, p) \in \mathbb{N} \times \mathbb{R}^+, (n \geq N) \implies (|u_{n+p} - u_n| < \epsilon)$.

Définition 33. Soit E un espace métrique. On dit que E est complet (ou de Banach) si toute suite de Cauchy converge au sein de cet espace.

Remarque 7. Un espace de Banach est nécessairement connexe (i.e un espace sans "trou").

1.3.2 Formes quadratiques & formes hermitiennes

Définition 34. Soit E et F deux $\mathbb{K}$-espaces vectoriels et $\varphi : E \times F \to \mathbb{K}$. On dit que φ est une forme bilinéaire si $\begin{cases} \forall x \in E, & y \mapsto \varphi(x, y) \text{ est linéaire} \\ \forall y \in F, & x \mapsto \varphi(x, y) \text{ est linéaire} \end{cases}$.

Définition 35. Soit E et F deux $\mathbb{C}$-espaces vectoriels et $\varphi : E \times F \to \mathbb{C}$. On dit que φ est une forme sesquilinéaire si $\begin{cases} \forall x \in E, & y \mapsto \varphi(x, y) \text{ est linéaire} \\ \forall y \in F, & x \mapsto \varphi(x, y) \text{ est semi-linéaire} \end{cases}$.

Remarque 8. Toute forme sesquilinéaire est une forme bilinéaire si les espaces considérés sont des $\mathbb{R}$-espaces vectoriels.

Définition 36. Soit E un $\mathbb{K}$-espace vectoriel. Soit φ une forme bilinéaire sur E. On dit que φ est symétrique (resp. anti-symétrique) si pour tout $(x, y) \in E^2, \varphi(x, y) = \varphi(y, x)$ (resp. $\varphi(x, y) = -\varphi(y, x)$).

Remarque 9. Si $\mathbb{K}$ est de caractéristique 2, alors la symétrie est équivalente à l'anti-symétrie.

Définition 37. On suppose $\mathbb{K}$ de caractéristique différente de 2. Soit E un $\mathbb{K}$-espace vectoriel. Soit φ une forme bilinéaire symétrique sur E. On appelle forme quadratique sur E toute application $q : \begin{array}{ccc} E & \to & \mathbb{K} \\ x & \mapsto & \varphi(x, x) \end{array}$.

Proposition 14. *On suppose $\mathbb{K}$ de caractéristique différente de 2. Soit E un $\mathbb{K}$-espace vectoriel. Soit q une forme quadratique sur E. Alors il existe une unique forme bilinéaire symétrique φ telle que pour tout $x \in E, q(x) = \varphi(x, x)$. φ est appelée la forme polaire de q et on dispose de l'identité de polarisation : $\forall (x, y) \in E^2, \varphi(x, y) = \frac{1}{4}(q(x + y) - q(x - y))$.*

Définition 38. Soit E un $\mathbb{C}$-espace vectoriel. Soit φ une forme sesquilinéaire sur E. On dit que φ est à symétrie hermitienne si pour tout $(x,y) \in E^2, \varphi(x,y) = \overline{\varphi(y,x)}$.

Définition 39. Soit E un $\mathbb{C}$-espace vectoriel. Soit φ une forme sesquilinéaire sur E à symétrie hermitienne. On appelle forme hermitienne sur E toute application $\begin{array}{rccc} \phi : & E & \to \mathbb{R} \\ & x & \mapsto \varphi(x,x) \end{array}$.

Proposition 15. *Soit ϕ une forme hermitienne. Alors il existe une unique forme sesquilinéaire à symétrie hermitienne φ telle que pour tout $x \in E, \phi(x) = \varphi(x,x)$. φ est appelée la forme polaire de ϕ et on dispose de l'identité de polarisation : $\forall (x,y) \in E^2, \varphi(x,y) = \frac{1}{4}\left(\phi(x+y) - \phi(x-y) + i\phi(x-iy) - i\phi(x+iy)\right)$.*

1.3.3 Espaces préhilbertiens & espaces de Hilbert

Définition 40. Soit E un $\mathbb{K}$-espace vectoriel et f une forme bilinéaire sur E. On dit que f est :
- définie si pour tout $x \in E, (f(x,x) = 0) \implies (x = 0)$.
- positive si pour tout $x \in E, f(x,x) \geq 0$.
- définie-positive si elle est positive et définie.

Définition 41. Soit E un $\mathbb{R}$-espace vectoriel (resp. un $\mathbb{C}$-espace vectoriel). Soit ϕ une forme quadratique (resp. hermitienne) sur E, définie-positive. Sa forme polaire est appelée produit scalaire (resp. produit scalaire hermitien) sur E, et on note $\langle . \mid . \rangle$.

Définition 42. Soit E un $\mathbb{R}$-espace vectoriel. On dit que E est un espace préhilbertien réel s'il est muni d'un produit scalaire. Si E est de dimension finie, alors il est qualifié d'euclidien.

Définition 43. Soit E un $\mathbb{C}$-espace vectoriel. On dit que E est un espace préhilbertien complexe s'il est muni d'un produit scalaire hermitien. Si E est de dimension finie, alors il est qualifié d'hermitien.

Les résultats suivants sont analogues pour les espaces préhilbertiens réels.

Définition 44. Soit $(E, \langle . \mid . \rangle)$ un espace préhilbertien complexe. Ainsi, l'application $\begin{array}{rccc} \|.\| : & E & \to \mathbb{R}^+ \\ & x & \mapsto \sqrt{\langle x|x \rangle} \end{array}$ définit une norme sur E que l'on appelle norme euclidienne sur E.

Définition 45. Soit $(E, \langle . \mid . \rangle)$ un espace préhilbertien complexe. Un vecteur $x \in E$ est dit unitaire si $\|x\| = 1$.

Définition 46. Soit $n \in \mathbb{N}^*$. Soit une famille de vecteurs de E tous non nuls $\mathcal{F} = (e_i)_{i \in [\![1;n]\!]}$.
- $\mathcal{F}$ est dite orthogonale si pour tout $(i,j) \in [\![1;n]\!]^2, (i \neq j) \implies (\langle e_i | e_j \rangle = 0)$.
- $\mathcal{F}$ est dite orthonormale (ou orthonormée) si pour tout $(i,j) \in [\![1;n]\!]^2, \langle e_i | e_j \rangle = \delta_{i,j}$
- Si E est hermitien de dimension n et que $\mathcal{F}$ est une famille orthonormale, alors $\mathcal{F}$ est une base orthonormale de E.

Proposition 16. *Dans un espace hermitien, toute famille orthonormale est libre.*

Définition 47. Soit $(E, \langle . \mid . \rangle)$ un espace préhilbertien complexe et $U \in \mathcal{P}(E)$. On appelle orthogonal de U, et on note $U^\perp$, l'ensemble $\{x \in E, \forall u \in U, \langle x|u \rangle = 0\}$.

Proposition 17. *Soit $(E, \langle . \mid . \rangle)$ un espace préhilbertien complexe et $U \in \mathcal{P}(E)$.*
- $(\mathrm{Vect}(U))^\perp = U^\perp$.
- *$U^\perp$ est un sous-espace vectoriel de E.*

Proposition 18. *Soit $(E, \langle . \mid . \rangle)$ un espace hermitien. Soit F un sous-espace vectoriel de E. Alors $F \oplus F^\perp = E$.*

Définition 48. Soit E un espace préhilbertien complexe et F un sous-espace vectoriel de E. On définit la projection orthogonale sur F, la projection $p_F : E \to E$ sur F parallèlement à $F^\perp$.

Définition 49. On appelle espace de Hilbert tout espace préhilbertien complet.

Proposition 19. *Soit $n \in \mathbb{N}^*$. $\mathbb{C}^n$ est un espace de Hilbert.*

Démonstration. Soit $n \in \mathbb{N}^*$.
- $\mathbb{C}^n$ muni de son produit scalaire hermitien canonique est clairement un espace hermitien.
- La densité de $\mathbb{C}$ dans $\mathbb{C}$ permet de conclure quant à la complétude de $\mathbb{C}^n$.

□

1.4 Produit tensoriel

La théorie complète des tenseurs n'est pas présentée ici. On se restreint ainsi au produit de Kronecker entre matrices, ainsi qu'au produit tensoriel de deux espaces vectoriels.

Définition 50. Soit $(m,n,p,q) \in (\mathbb{N}^*)^4$. Soit $(A,B) \in \mathcal{M}_{m,n}(\mathbb{C}) \times \mathcal{M}_{p,q}(\mathbb{C})$ dont on note respectivement les coefficients $(a_{i,j})_{(i,j)\in[\![1;m]\!]\times[\![1;n]\!]}$ et $(b_{i,j})_{(i,j)\in[\![1;p]\!]\times[\![1;q]\!]}$. On définit le produit de Kronecker entre A et B, et on note $A \otimes B$ (lu "A tenseur B"), la matrice $(c_{i,j})_{(i,j)\in[\![1;p]\!]\times[\![1;q]\!]}$ de taille $mp \times nq$ telle que :

$$A \otimes B = \begin{pmatrix} a_{1,1}B & \dots & a_{1,n}B \\ \vdots & & \vdots \\ a_{m,1}B & \dots & a_{m,n}B \end{pmatrix}$$

Proposition 20. *Le produit de Kronecker est bilinéaire et associatif.*

Définition 51. Soit E et F deux $\mathbb{C}$-espaces vectoriels. Il existe un espace vectoriel, que l'on note $E \otimes F$, et une forme bilinéaire $\otimes : E \times F \to E \otimes F$ telle que pour tout $\mathbb{C}$-espace vectoriel G et pour toute forme bilinéaire $\phi : E \times F \to G$, il existe une unique application linéaire $f : E \otimes F \to G$ telle que pour tout $(x,y) \in E \times F, \phi(x,y) = f(x \otimes y)$. On dit alors que $E \otimes F$ est le produit tensoriel de E et de F.

Remarque 10. Dans la suite, on prendra le produit de Kronecker comme forme bilinéaire $\otimes$ respectant l'énoncé ci-dessus.

Remarque 11. Le couple $(E \otimes F, \otimes)$ est unique à isomorphisme près.

Théorème 1. *Soit E et F deux $\mathbb{C}$-espaces vectoriels de dimensions finies respectives $m \in \mathbb{N}^*$ et $n \in \mathbb{N}^*$. On note respectivement $(e_i)_{i\in[\![1;m]\!]}$ et $(f_i)_{i\in[\![1;n]\!]}$ des bases de E et de F. Ainsi, $(e_i \otimes f_j)_{(i,j)\in[\![1;m]\!]\times[\![1;n]\!]}$ est une base de $E \otimes F$ et $\dim(E \otimes F) = \dim(E) \times \dim(F)$.*

Proposition 21. *Soit E un $\mathbb{C}$-espace vectoriel. Soit $n \in \mathbb{N} \setminus \{0;1\}$. On note $E^{\otimes_n}$ le produit tensoriel de E n fois avec lui-même.*

1.5 Formalisme de Dirac

1.5.1 Dualité

Soit $\mathbb{K}$ un corps. Soit E un $\mathbb{K}$-espace vectoriel.

Définition 52. On appelle dual de E, et on note E^*, l'ensemble des formes linéaires de $E \to \mathbb{K}$.

Proposition 22. *E^* est un espace vectoriel. Si E est de dimension finie, alors E^* est aussi de dimension finie et égale à celle de E.*

On suppose E de dimension finie égale à $n \in \mathbb{N}^*$.

Définition 53. Soit $(e_i)_{i\in[\![1;n]\!]} \in E^n$ une base de E. Soit $(u_i)_{i\in[\![1;n]\!]}$ n formes linéaires de $E \to \mathbb{K}$ telle que pour tout $x \in E, x = \sum_{k=1}^{n} u_k(x)e_k$. Alors, $(u_i)_{i\in[\![1;n]\!]}$ est une base de E^* et est appelée base duale de $(e_i)_{i\in[\![1;n]\!]}$.

Proposition 23. *Soit $(u_i)_{i\in[\![1;n]\!]}$ une base de E^*. Alors, il existe une unique base $(e_i)_{i\in[\![1;n]\!]}$ de E dont $(u_i)_{i\in[\![1;n]\!]}$ est la base duale. Ainsi, $(e_i)_{i\in[\![1;n]\!]}$ est appelée base antéduale de $(u_i)_{i\in[\![1;n]\!]}$.*

1.5.2 Endomorphisme adjoint

Soit E un espace hermitien dont on note $\langle .|. \rangle$ le produit scalaire hermitien.

Théorème 2. *Soit u un endomorphisme de E. Il existe un unique endomorphisme u^* de E, que l'on appelle adjoint de u, tel que pour tout $(x,y) \in E^2, \langle u(x)|y\rangle = \langle x|u^*(y)\rangle$.*

Proposition 24. *Pour tout* $u \in \mathcal{L}(E), \mathrm{Ker}(u^*) = (\mathrm{Im}(u))^\perp$.

Proposition 25. *Pour tout* $(u,v) \in \mathcal{L}(E)^2, \begin{cases} (u^*)^* & = u \\ (uv)^* & = v^*u^* \end{cases}$.

Définition 54. Soit $(m,n) \in (\mathbb{N}^*)^2$. Soit $A \in \mathcal{M}_{m,n}(\mathbb{C})$. On appelle matrice adjointe (ou trans-conjuguée) de A, et on note A^* ou $A^\dagger$, la matrice $\overline{A}^T \in \mathcal{M}_{n,m}(\mathbb{C})$.

Proposition 26. *Soit* $u \in \mathcal{L}(E)$. *Notons* $\mathcal{B}$ *une base orthonormée de* $n \in \mathbb{N}^*$ *vecteurs de* E. *Soit* $A = \mathcal{M}at_{\mathcal{B}}(u) \in \mathcal{M}_n(\mathbb{C})$. *Alors,* A^* *est la matrice représentative de* u^* *dans* $\mathcal{B}$.

Définition 55. Soit $n \in \mathbb{N}^*$. Soit $U \in \mathcal{M}_n(\mathbb{C})$. On dit que U est unitaire si $UU^* = U^*U = I_n$.

1.5.3 Notations de Dirac

Soit E un espace hermitien dont on note $\langle .|. \rangle$ le produit scalaire hermitien. On note $\mathcal{B}$ une base orthonormée de E.

Définition 56. Soit $u \in E$. On définit $|u\rangle$ (lu "Ket u") comme la matrice colonne des coordonnées de u dans la base $\mathcal{B}$.

Définition 57. Soit $u \in E$. On définit $\langle u|$ (lu "Bra u") comme la forme linéaire $f \in E^*$ (dont on note U_f la matrice représentative dans la base $\mathcal{B}$) telle que pour tout $x \in E, f(x) = U_f\,|x\rangle = \langle u|x\rangle$. Matriciellement, $\langle u|$ est une matrice ligne.

Proposition 27. *En dimension finie, il existe une bijection entre* $\langle .|$ *et* $|.\rangle$. *En effet, un "bra" est l'adjoint d'un "ket", et inversement.*

1.6 Probabilités

1.6.1 Définitions

Définition 58. On appelle expérience aléatoire toute expérience non-déterministe. Les résultats possibles sont appelés issues de l'expérience. L'ensemble de toutes les issues, noté Ω, est appelé univers des possibilités.

Définition 59. On appelle événement aléatoire un événement ayant une chance de se réaliser ou non. Il s'agit d'une partie de Ω.

Définition 60. Soit $A \in \mathcal{P}(\Omega)$. Si à l'issue d'une expérience aléatoire, le résultat appartient à A, alors on dit que A s'est réalisé.

Remarque 12. Ω est l'événement certain tandis que $\emptyset$ est l'événement impossible.

Définition 61. Soit $(A,B) \in \mathcal{P}(\Omega)^2$.

1. On appelle disjonction de A et de B, et on note (A ou B), l'événement $A \cup B$.
2. On appelle conjonction de A et de B, et on note (A et B), l'événement $A \cap B$.
3. On appelle contraire de A, et on note $\overline{A}$, le complémentaire de A dans Ω.
4. L'inclusion $A \subset B$ indique que la réalisation de A entraîne celle de B, et on note ($A \implies B$).

Définition 62. Soit $(A,B) \in \mathcal{P}(\Omega)^2$. On dit que A et B sont incompatibles ssi $A \cap B = \emptyset$.

Définition 63. Soit I un ensemble de cardinal fini $n \in \mathbb{N}^*$. Soit $(A_i)_{i\in I}$ une famille d'événements. On dit que A forme un système complet d'événements ssi $\begin{cases} \bigcup\limits_{i\in I} A_i & = \Omega \\ \forall (i,j) \in I^2, (i \neq j) & \implies (A_i \cap A_j = \emptyset) \end{cases}$.

Définition 64. Soit E un ensemble. On appelle variable aléatoire sur Ω toute application $X : \Omega \to E$.

Remarque 13. Soit X une variable aléatoire sur Ω. On note l'ensemble d'arrivée de X par $X(\Omega)$. Pour un résultat $x \in X(\Omega)$, la notation $(X = x)$ désigne l'image réciproque de x par X, soit $X^{-1}(\{x\}) = \{\omega \in \Omega, X(\omega) = x\}$.

1.6.2 Espace probabilisé fini

Définition 65. Soit Ω un ensemble fini non vide. On appelle probabilité sur Ω, toute application $P : \mathcal{P}(\Omega) \to [0;1]$ telle que $\begin{cases} P(\Omega) & = 1 \\ \forall (A,B) \in \mathcal{P}(\Omega)^2, & (A \cap B = \emptyset) \implies (P(A \cup B) = P(A) + P(B)) \end{cases}$.

Définition 66. Un événement est dit négligeable lorsque sa probabilité est nulle.

Définition 67. Soit Ω un ensemble fini non vide et P une probabilité sur Ω. On dit que le couple (Ω, P) est un espace probabilisé fini. De plus, pour tout $A \in \mathcal{P}(\Omega)$, on appelle probabilité de l'événement A le réel $P(A)$.

Définition 68. Soit (Ω, P) un univers probabilisé fini et $A \in \mathcal{P}(\Omega)$. On définit la probabilité uniforme de A comme $P(A) = \frac{\text{Card}(A)}{\text{Card}(\Omega)}$.

Théorème 3. *Soit (Ω, P) un univers probabilisé fini. Soit $(A, B) \in \mathcal{P}(\Omega)^2$.*

1. $P(\overline{A}) = 1 - P(A)$.
2. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.
3. $P(A \setminus B) = P(A) - P(A \cap B)$.
4. $(A \subset B) \implies (P(A) \leq P(B))$.

1.6.3 Conditionnement

Définition 69. Soit (Ω, P) un univers probabilisé fini. Soit $B \in \mathcal{P}(\Omega)$ tel que $P(B) > 0$. Pour tout $A \in \mathcal{P}(\Omega)$, on appelle probabilité conditionnelle de A sachant B, et on note $P_B(A)$ ou encore $P(A|B)$, le nombre $\frac{P(A \cap B)}{P(B)}$.

Proposition 28. *Soit (Ω, P) un univers probabilisé fini. Soit $B \in \mathcal{P}(\Omega)$ tel que $P(B) > 0$. L'application* $P_B : \begin{array}{ll} \mathcal{P}(\Omega) & \to [0;1] \\ A & \mapsto P_B(A) \end{array}$ *est bien une probabilité sur Ω.*

Proposition 29. *Soit (Ω, P) un univers probabilisé fini. Soit $(A, B) \in \mathcal{P}(\Omega)^2$ tel que $P(B) > 0$. On a $P(A \cap B) = P(A) \times P(B)$.*

Proposition 30. *Soit (Ω, P) un univers probabilisé fini. Soit $n \in \mathbb{N}$ tel que $n \geq 2$. Soit $(A_i)_{i \in [\![1;n]\!]}$ une famille d'événements de Ω tel que $P\left(\bigcap_{i=1}^{n-1} A_i\right) > 0$. On a $P\left(\bigcap_{i=1}^{n} A_i\right) = P(A_1) \times \prod_{i=2}^{n} P\left(A_i | \bigcap_{k=1}^{i-1} A_k\right)$.*

Théorème 4. *Soit (Ω, P) un univers probabilisé fini. Soit $n \in \mathbb{N}$. Soit $(A_i)_{i \in [\![1;n]\!]}$ un système complet d'événements non négligeables de Ω. Alors, pour tout $B \in \mathcal{P}(\Omega)$, $P(B) = \sum_{k=1}^{n} P(B \cap A_k) = \sum_{k=1}^{n} P(A_k) \times P(B|A_k)$.*

Théorème 5. *Soit (Ω, P) un univers probabilisé fini. Soit $n \in \mathbb{N}$. Soit $(A_i)_{i \in [\![1;n]\!]}$ un système complet d'événements non négligeables de Ω. Alors, pour tout $B \in \mathcal{P}(\Omega)$ tel que $P(B) > 0$ et pour tout $j \in [\![1;n]\!]$, on a $P(A_j|B) = \frac{P(A_j) \times P(B|A_j)}{\sum_{i=1}^{n} P(A_i) \times P(B|A_i)}$.*

1.6.4 Indépendance

Définition 70. Soit (Ω, P) un univers probabilisé fini. On dit que deux événements A et B de Ω sont indépendants ssi $P(A \cap B) = P(A) \times P(B)$.

Proposition 31. *Soit (Ω, P) un univers probabilisé fini. Soit A et B deux événements indépendants de Ω. Alors, les événements A et $\overline{B}$ sont indépendants.*

Définition 71. Soit (Ω, P) un univers probabilisé fini. Soit $n \in \mathbb{N}^*$. Soit $(A_1, \ldots A_n) \in \mathcal{P}(\Omega)^n$. On dit que $k \in [\![1;n]\!]$ événements sont mutuellement indépendants si pour tout $(i_k)_{i \in [\![1;k]\!]} \in [\![1;n]\!]^k, P(A_{i_1} \cap \ldots \cap A_{i_k}) = P(A_{i_1}) \times \ldots \times P(A_{i_k})$.

Proposition 32. *Soit (Ω, P) un univers probabilisé fini. Toute sous-famille d'une famille d'événements mutuellement indépendants de Ω est formée d'événements mutuellement indépendants.*

Chapitre 2

Informatique Classique

Dans les prochaines parties, $\mathbb{B} = \{0; 1\}$ désigne l'ensemble des booléens.

2.1 Algorithmie

L'algorithmie est fortement liée à l'évolution des Mathématiques mais aussi de l'Informatique. La question initiale est de savoir pour une tâche donnée et répétitive, s'il est possible de trouver une méthode ou un ensemble d'instructions, permettant d'atteindre systématiquement le bon résultat. Ainsi, on retrouve des traces des premiers algorithmes à l'époque des Babyloniens (*environ -2000*) ; le but était de calculer des aires de lopins de terre pour l'agriculture (*i.e résolution d'équations quadratiques, encore non formellement définies*). Nous pouvons citer également Euclide (*-300*) qui a fourni une méthode pour calculer le PGCD de deux entiers, Al-Khwârizmî (*IXe siècle*) qui a donné son nom à ce domaine des Sciences, ou encore Ada Lovelace (*XIXe siècle*), considérée comme la première programmeuse, qui a écrit des algorithmes de calcul des nombres de Bernoulli pour la machine de Babbage.

Ainsi, nous allons définir dans cette partie ce qu'est un algorithme, ainsi que la théorie qui permet de déterminer la "vitesse" d'un algorithme et de le démontrer.

2.1.1 Définitions

Définition 72. On appelle algorithme toute suite finie d'instructions ou d'opérations non ambiguës qui prend une entrée afin de produire une sortie.

Remarque 14. Un algorithme respecte les principes de quantification d'objets (*i.e un objet n'existe pas s'il n'a pas été défini avant ou dans le bloc d'assertions en cours de lecture*). Il est alors nécessaire de définir des conventions d'écriture ainsi qu'un lexique.

Remarque 15. Un algorithme est indépendant du langage de programmation dans lequel il est ensuite implémenté. Notons de plus que le pseudo-code ne constitue pas un algorithme formel.

Définition 73. On appelle fonction tout algorithme qui retourne un objet ou une valeur. On appelle procédure tout algorithme qui ne retourne pas explicitement un objet.

Remarque 16. Une fonction en Algorithmie est parfaitement identifiable à une fonction en Mathématiques.

Définition 74. On définit la signature d'un algorithme comme étant une description formelle d'un algorithme sans rentrer dans les détails de ce dernier. Elle doit préciser à minima le nom (*aussi appelé identificateur*), les paramètres d'entrées ainsi que leur nature (*que l'on appelle type*) ainsi que la nature de la sortie. Quand il n'y a pas de sortie explicite, on utilise le caractère $\emptyset$.

Maintenant que l'on a défini ce qu'est un algorithme, un autre problème se pose. Comment lire ou exécuter automatiquement un algorithme ? Alan Turing a imaginé en 1936 un automate capable de répondre à cette question.

Définition 75. On appelle machine de Turing tout quintuplet $(Q, \Gamma, q_0, \delta, F)$, où Q est un ensemble fini d'états, Γ est l'alphabet de travail et qui contient un élément neutre que l'on appelle le blanc (*noté usuellement 0 ou B*), $q_0 \in Q$ est l'état initial de l'automate, $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ est la fonction de transition (*lecture, écriture*) et $F \subseteq Q$ l'ensemble des états terminaux.

Remarque 17. On définit plus communément la machine de Turing comme une machine composée de quatre éléments :

1. Un ruban non fini, divisé en cases pouvant contenir un élément de Γ. On suppose que le ruban est par défaut rempli d'un nombre non fini du caractère blanc. De plus, le premier symbole du ruban est l'entrée de l'automate.
2. Un registre d'état qui par défaut se trouve à q_0. Si ce registre contient un élément de F, alors l'automate est placé en état final et ne peut plus évoluer.
3. Une tête de lecture/écriture qui peut lire et écrire des éléments du ruban. Elle peut également se déplacer à gauche ou à droite selon l'opération dans la table d'actions.
4. Une table d'actions qui indique quel symbole à écrire sur le ruban, comment déplacer la tête, et l'état dans lequel il faut passer selon le symbole lu sur le ruban. On considère qu'un état ou une opération non valide (*ou non définie*) conduit à un état terminal.

Remarque 18. Une machine de Turing est un algorithme.

Théorème 6. *Il existe une machine de Turing, que l'on appelle machine universelle de Turing, capable de simuler toute machine de Turing.*

On dispose à présent d'un moyen pour traiter un algorithme. La question fondamentale qui se pose désormais est de savoir si un algorithme peut être vu comme une machine de Turing. Ces interrogations découlent du théorème d'incomplétude de Gödel mais également du problème de la décision (*ou Entscheidungsproblem en Allemand*). Ainsi, le mathématicien Alonzo Church a travaillé sur la question et a produit un énoncé formellement indémontrable :

Théorème 7 (Thèse de Church-Turing)**.** *On considère qu'une fonction est calculable ssi elle peut être traitée par une machine de Turing.*

En d'autres termes, tout algorithme peut être traité par une machine de Turing.

2.1.2 Induction

Avant de continuer, il est nécessaire de définir le principe d'induction. En effet, un grand nombre de propositions et d'algorithmes est prouvé en utilisant cette méthode. Ce principe est basé sur l'axiomatique de Peano, qui permet de construire l'ensemble des entiers naturels $\mathbb{N}$.

On rappelle à présent les axiomes de Peano:

1. $0 \in \mathbb{N}$.
2. Tout entier naturel a un unique successeur, qui est également un entier naturel.
3. Aucun entier naturel n'admet 0 pour successeur.
4. Si deux entiers naturels ont le même successeur, alors ils sont égaux.
5. L'ensemble qui contient 0 ainsi que tous ses successeurs est $\mathbb{N}$.

On en tire entre autres que toute partie non vide de N contient un plus petit élément (*borne inférieure atteinte : existence + unicité*). D'où le théorème suivant :

Théorème 8. *Soit $n_0 \in \mathbb{N}$ et $\mathcal{B}$ une partie de $\mathbb{N}$ telle que $\mathcal{B} = \{n \in \mathbb{N}, n \geq n_0\}$. Soit $(H_i)_{i \in \mathcal{B}}$ une famille d'assertions logiques. Si H_{n_0} est vraie et que pour tout $n \in \mathcal{B}, H_n \implies H_{n+1}$, alors le principe de récurrence assure que pour tout $n \in \mathcal{B}, H_n$ est vraie. On parle alors de récurrence sur $n \in \mathbb{N}$.*

Théorème 9. *Soit $n_0 \in \mathbb{N}$ et $\mathcal{B}$ une partie de $\mathbb{N}$ telle que $\mathcal{B} = \{n \in \mathbb{N}, n \geq n_0\}$. Soit $(H_i)_{i \in \mathcal{B}}$ une famille d'assertions logiques. Si H_{n_0} est vraie et que pour tout $n \in \mathcal{B}, (H_i)_{i \in [\![n_0; n]\!]} \implies H_{n+1}$, alors le principe de récurrence assure que pour tout $n \in \mathcal{B}, H_n$ est vraie. On parle alors de récurrence forte sur $n \in \mathbb{N}$.*

Remarque 19. En principe, pour les démonstrations, on travaille sur une partie connexe de $\mathbb{N}$ de cardinal fini. Ainsi, on parle d'itération finie, et non de récurrence.

2.1.3 Correction d'un algorithme

Il convient désormais d'être en mesure de déterminer si un algorithme est valide ou non. En effet, il est possible que celui-ci ne termine pas ou qu'il renvoie le mauvais résultat.

Définition 76. On dit qu'un algorithme converge (*ou termine*) ssi la machine de Turing associée arrive à un état terminal. On parle aussi de la terminaison d'un algorithme.

Définition 77. On parle de correction partielle d'un algorithme quand celui-ci renvoie le bon résultat. On parle de correction totale d'un algorithme si celui-ci est partiellement correct et termine ; on dit alors que l'algorithme est correct.

La seule utilisation de l'induction afin de montrer qu'un algorithme est correct n'est, la plupart du temps, pas suffisante. En effet, il devient compliqué de s'assurer notamment de la terminaison dans le cas de la récursivité, ou de boucle. Ainsi, il est nécessaire de définir des outils supplémentaires.

Définition 78. On appelle variant de boucle, la suite d'entiers naturels strictement décroissante à chaque entrée de boucle.

Définition 79. On appelle invariant (de boucle) toute propriété qui reste vraie au cours de l'exécution de l'algorithme.

2.1.4 Complexité d'un algorithme

Une autre question importante concernant les algorithmes est de savoir s'il existe une relation d'ordre total afin de les comparer. Les outils mathématiques de comparaison asymptotique nous permettent d'introduire la notion de complexité d'un algorithme. On peut en conséquence mesurer l'efficacité de ce dernier. À partir de cela, on peut déterminer pour un problème donné s'il est complexe selon l'existence ou non d'un algorithme efficace.

Définition 80. On définit la complexité temporelle $T : \mathbb{N} \to \mathbb{R}^+$ d'un algorithme, dont l'entrée est de taille $n \in \mathbb{N}^*$, comme étant le temps d'exécution de l'algorithme.

Définition 81. On définit la complexité spatiale $S : \mathbb{N} \to \mathbb{R}^+$ d'un algorithme, dont l'entrée est de taille $n \in \mathbb{N}^*$, comme étant la mémoire occupée par l'algorithme.

Remarque 20. Notons que dans le cas de la complexité, on ne s'intéresse pas aux performances de la machine, mais uniquement à celles de l'algorithme. Ainsi, on considère des unités de temps et de mémoire de complexité.

Remarque 21. On admet que les opérations élémentaires, telles que les opérations algébriques ($+$, -, $\times$, $\div$) et d'assignement (*ce qui peut être faux parfois*), se réalisent en temps constant.

On considère usuellement trois types de complexités, à savoir la complexité pire cas (*que l'on cherche absolument à éviter*), en moyenne, et meilleur cas. On utilise ainsi les notations de Landau pour décrire de tels comportements.

Définition 82. Soit $f : \mathbb{N} \to \mathbb{R}^+$. On définit l'ensemble des fonctions asymptotiquement minorées par f au voisinage de $+\infty$, que l'on note $\mathcal{O}(f)$ (*lu "grand omicron de f"*), l'ensemble :

$$\mathcal{O}(f) = \{g \in \mathcal{F}(\mathbb{N}, \mathbb{R}^+), \exists(\epsilon, N) \in \mathbb{R}^{+*} \times \mathbb{N}, \forall n \in \mathbb{N}, (n \geq N) \implies (g(n) \leq \epsilon f(n))\}$$

Définition 83. Soit $f : \mathbb{N} \to \mathbb{R}^+$. On définit l'ensemble des fonctions qui majorent asymptotiquement f (*au sens de Knuth, fonctions qui dominent f*) au voisinage de $+\infty$, que l'on note $\Omega(f)$ (*lu "grand oméga de f"*), l'ensemble :

$$\Omega(f) = \{g \in \mathcal{F}(\mathbb{N}, \mathbb{R}^+), \exists(\epsilon, N) \in \mathbb{R}^{+*} \times \mathbb{N}, \forall n \in \mathbb{N}, (n \geq N) \implies (g(n) \geq \epsilon f(n))\}$$

Définition 84. Soit $f : \mathbb{N} \to \mathbb{R}^+$. On définit l'ensemble des fonctions du même ordre de grandeur que f au voisinage de $+\infty$, que l'on note $\Theta(f)$ (*lu "grand thêta de f"*), l'ensemble :

$$\Theta(f) = \{g \in \mathcal{F}(\mathbb{N}, \mathbb{R}^+), \exists(\epsilon, \eta, N) \in (\mathbb{R}^{+*})^2 \times \mathbb{N}, \forall n \in \mathbb{N}, (n \geq N) \implies (\epsilon f(n) \leq g(n) \leq \eta f(n))\}$$

Remarque 22. Soit $f : \mathbb{N} \to \mathbb{R}^+$ et $g : \mathbb{N} \to \mathbb{R}^+$. On voit souvent l'abus de notation de type $f = \mathcal{O}(g)$ au lieu de $f \in \mathcal{O}(g)$.

Proposition 33. *Soit $f : \mathbb{N} \to \mathbb{R}^+$ et $g : \mathbb{N} \to \mathbb{R}^+$. $f = \Theta(g) \iff \begin{cases} f &= \mathcal{O}(g) \\ f &= \Omega(g) \end{cases}$.*

Proposition 34. *Soit $f : \mathbb{N} \to \mathbb{R}^+$ et $g : \mathbb{N} \to \mathbb{R}^+$. On suppose que g ne s'annule pas au voisinage de $+\infty$.*

1. *Si la limite en $+\infty$ du quotient $\frac{f}{g}$ existe, est finie et est strictement supérieure à 0, alors $\begin{cases} f &\in \Theta(g) \\ g &\in \Theta(f) \end{cases}$.*
2. *Si la limite en $+\infty$ du quotient $\frac{f}{g}$ existe, est finie et est supérieure à 0, alors $f \in \mathcal{O}(g)$.*
3. *Si la limite en $+\infty$ du quotient $\frac{f}{g}$ existe, et est $+\infty$, alors $f \in \Omega(g)$.*

En pratique, on s'intéresse surtout à la complexité pire cas afin de majorer le temps d'exécution de l'algorithme. On peut ainsi définir plusieurs classes de complexité :

Soit $\alpha \in \mathbb{R}$ tel que $\alpha > 1$.

Classe	Temps/Mémoire
Constant	$\mathcal{O}(1)$
Logarithmique	$\mathcal{O}(\log(n))$
Linéaire	$\mathcal{O}(n)$
Linéarithmique	$\mathcal{O}(n\log(n))$
Polynomiale	$\mathcal{O}(n^\alpha)$
Exponentielle	$\mathcal{O}(e^{\alpha n})$
Factorielle	$\mathcal{O}(n!)$

Une question essentielle se pose à présent. Est-il possible pour un problème dit complexe de trouver un algorithme de complexité polynomiale ? La question est loin d'être triviale puisqu'il s'agit d'un problème fondamental. Ainsi, on a défini des classes de complexité de problème :

Définition 85. On dit qu'un problème calculatoire appartient à la classe de complexité P (*polynomial*) s'il existe un algorithme qui résout le problème en temps polynomial.

Définition 86. On dit qu'un problème calculatoire appartient à la classe de complexité NP (*nondeterministic polynomial*) s'il existe un algorithme qui vérifie la validité d'une solution en temps polynomial.

On a clairement l'inclusion P $\subset$ NP. Le tout est de savoir si l'inclusion réciproque est vraie, ou si le problème est indécidable.

2.2 Circuits électriques

2.2.1 Stockage de l'information & Langage d'assemblage

Les premiers ordinateurs (années 1940) étaient des machines très coûteuses qui ne tenaient pas entièrement dans une pièce. On estimait ainsi un marché mondial pour quelques ordinateurs (*environ trois*). En 1945, le premier "vrai" ordinateur électronique, l'ENIAC (*Electronic Numerical Integrator And Computer*) est inventé. Il s'agit d'une machine complète au sens de Turing, capable ainsi de résoudre tout problème numérique. Ce dernier était composé de tubes à vide, un dispositif électronique composé d'un filament de chauffage qui émet un courant d'électrons dans le vide. Or, cette technologie posait deux problèmes. Premièrement, on a besoin de beaucoup de tubes afin de faire des calculs, d'où une place importante occupée par le tout. Enfin, le fil est chauffé à 1000°C, ce qui implique une consommation énergétique importante.

C'est dans ce contexte que l'on a étudié les semi-conducteurs. Ainsi, en 1947, le premier transistor est inventé. Il s'agit d'un composant amplificateur de tension et de courant électrique, à l'échelle micrométrique. Ainsi, dans le cas des circuits intégrés, on peut considérer qu'un transistor agit comme une mémoire pouvant avoir l'état 0 ou l'état 1. On peut donc définir un support de stockage de l'information.

Définition 87. On appelle bit, l'unité élémentaire de l'information de valeur booléenne 0 ou 1. Un groupe de 8 bits est appelé octet.

On remarque ainsi très rapidement qu'un ordinateur travaille en base 2 et non dans notre base décimale usuelle (*ce qui n'est pas le cas de l'ENIAC*).

Définition 88. Soit $n \in \mathbb{N}^*$. On appelle registre à n bits, un espace indexé par $[\![1; n]\!]$ dans lequel on stocke une chaîne de n bits encodant une information. Selon le boutisme (*endianness en Anglais*), les bits sont classés dans un registre du plus significatif au moins significatif (*gros-boutisme*), ou dans le sens inverse (*petit-boutisme*). Par défaut, on considère qu'un registre est en petit-boutisme.

Tout ordinateur dispose d'une unité centrale de calcul (*Central Process Unit*), que l'on appelle plus communément processeur. Celle-ci dispose d'un jeu d'instructions (*ou opcodes*) qui lui est propre que l'on appelle assembleur. Elle est également composée de registres qui ont chacun un but donné. Si on considère un adressage 16 bits, on obtient la liste suivante :

- Général :
  - Accumulateur `ax`
  - Base `bx`
  - Compteur `cx`
  - Données `dx`
- Segments :
  - Segment de code `cs` (*non mutable sauf lors de sauts ou d'appels*)
  - Segment de données `ds`
  - Extra segment `es`
  - Segment de pile (*stack*) `ss`
- Décalage (*offset*) :
  - Pointeur d'instruction `ip` (*non mutable*)
  - Pointeur de pile `sp` (*pointe le haut de la pile*)
  - Indice de source `si` (*utilisé pour manipuler chaînes de données*)
  - Indice de destination `di` (*idem*)
  - Pointeur de base `bp` (*idem*)
- Drapeaux (*mis à jour après chaque interruption ou calcul*) :
  - Retenue `CF`
  - Parité `PF`
  - Retenue auxiliaire `AF`
  - Zéro `ZF`
  - Signature `SF`
  - Trap `TF`
  - Interruption `IF`
  - Direction `DF`
  - Dépassement (*overflow*) `OF`

Sans perdre de généralités, pour un registre général `ax`, `ah` permet d'accéder à l'octet des bits de poids fort (*High*) tandis que `al` permet d'accéder à l'octet des bits de poids faible (*Low*). De plus, la pile en assembleur évolue vers le bas de la mémoire.

Enfin, il existe deux syntaxes générales en assembleur : syntaxe Intel (de la forme `INSTRUCTION %DEST, %SRC`) et la syntaxe AT&T (de la forme `INSTRUCTION %SRC, %DEST`).

2.2.2 Portes logiques

Il est possible de représenter tout algorithme par un circuit électrique composé de résistances, résistors, condensateurs et bobines.

Définition 89. On appelle porte logique tout ensemble de composants électroniques qui applique une opération booléenne sur un signal d'entrée et qui produit un signal de sortie.

On définit désormais les trois portes élémentaires.

Définition 90. 1. On définit la porte logique *NOT* (non logique), notée $\not$ou $\neg$, comme l'opérateur qui inverse le signal d'entrée. Son symbole schématique est un triangle suivi d'un cercle.

2. On définit la porte logique AND (et logique), notée $\wedge$ ou ., comme l'opérateur qui place 1 sur le signal de sortie, si les deux signaux d'entrée valent 1. Son symbole schématique est un demi-carré côté entrées, et demi-cercle côté sortie.
3. On définit la porte logique OR (ou logique, ou inclusif), notée $\vee$ ou $+$, comme l'opérateur qui place 1 sur le signal de sortie, si au moins une entrée vaut 1. Son symbole schématique est un demi-cercle convexe côté entrées, et triangle arrondi côté sortie.

Théorème 10. *Tout algorithme peut être représenté par un circuit faisant intervenir les portes logiques élémentaires.*

Il est également possible de définir d'autres portes, notamment pour simplifier les notations. Notons néanmoins que celles-ci peuvent également être exprimées en fonction des portes AND, OR et NOT. Ainsi :

Définition 91. On définit la porte logique XOR (ou exclusif), notée $\veebar$ ou $\oplus$, comme l'opérateur qui place 1 sur le signal de sortie, si un seul des signaux d'entrée vaut 1.

Remarque 23. On peut citer également les portes $NAND, NOR$ et $NXOR$ qui sont des négations logiques des portes précédentes.

Chapitre 3

Informatique Quantique

En 1965, le physicien George Moore a établi une loi empirique éponyme qui stipule que le nombre de transistors dans les microprocesseurs double tous les deux ans en moyenne, et pour un coût constant ou réduit. Néanmoins, avec des composants de plus en plus petits, les limites physiques de la microélectronique sont vite atteintes et cette loi ne peut dès lors plus être considérée comme valide. En effet, quand on approche l'ordre de quelques atomes pour les transistors, l'information que l'on cherche à garder en mémoire se retrouve affectée par des effets de décohérence quantique. Il devient ainsi intéressant d'étudier la physique quantique afin de trouver de nouveaux modèles et de nouveaux composants utilisables dans de telles conditions.

Dans cette partie, nous allons présenter les axiomes nécessaires afin de pouvoir effectuer des calculs quantiques. Nous allons également expliquer comment un circuit quantique se présente, mais également comment définir un algorithme quantique.

3.1 Calcul Quantique

Dans cette partie, on considère (Ω, P) un univers probabilisé fini.

3.1.1 Système quantique & Qudit

Définition 92. On appelle système quantique toute collection d'entités physiques dont le comportement est décrit par les principes de la mécanique quantique.

Définition 93. On définit un état quantique comme une entité mathématique permettant de modéliser un système quantique et qui est soumis aux axiomes décrits dans la section 3.1.3.

Définition 94. Soit $d \in \mathbb{N}$ tel que $d > 1$. On appelle système quantique à d niveaux tout système quantique pouvant exister dans un état de superposition de ces d états quantiques.

Définition 95. Soit $d \in \mathbb{N}$ tel que $d > 1$. On appelle qudit un système quantique à d niveaux. Autrement dit, il s'agit d'un vecteur unitaire d'un espace de Hilbert de dimension d.

3.1.2 Qubit

En utilisant le formalisme de Dirac, notons les vecteurs de la base canonique (orthonormale) de $\mathbb{C}^2$ par $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ et $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. On parle également de vecteurs d'état.

Définition 96. On appelle qubit tout vecteur unitaire de $|\psi\rangle \in \mathbb{C}^2$. Ainsi, il existe $(\alpha, \beta) \in \mathbb{C}^2$ tel que $|a|^2 + |b|^2 = 1$ et $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$. α et β sont appelés amplitudes de l'état $|\psi\rangle$.

Remarque 24. Un qubit est un qudit à 2 niveaux.

Contrairement à l'information classique, où un bit contient une valeur définie, l'information quantique est de nature probabiliste. En effet, un qubit est une combinaison linéaire de deux états ; il est ainsi nécessaire de le mesurer dans une base orthonormale donnée de $\mathbb{C}^2$ afin d'obtenir une valeur définie.

Définition 97. Soit $|\psi\rangle$ un qubit. Procéder à la mesure de $|\psi\rangle$ dans une base orthonormale de $\mathbb{C}^2$ revient à projeter l'état $|\psi\rangle$ sur cette base.

Soit $\mathcal{B} = (|\phi_1\rangle, |\phi_2\rangle)$ une base orthonormale de $\mathbb{C}^2$. Il existe alors $(\alpha, \beta) \in \mathbb{C}^2$ tel que $|a|^2 + |b|^2 = 1$ et $|\psi\rangle = \alpha |\phi_1\rangle + \beta |\phi_2\rangle$. On définit la probabilité d'observer $|\phi_1\rangle$ (resp. $|\phi_2\rangle$) après mesure, et on note $P(|\phi_1\rangle)$ (resp. $|\phi_2\rangle$), le nombre $|\alpha|^2$ (resp. $|\beta|^2$). Après la mesure, l'état $|\psi\rangle$ est rabattu sur l'état observé.

Proposition 35. *On pose* $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *et* $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. *La base de Hadamard* $H = (|+\rangle, |-\rangle)$ *est une base orthonormale de* $\mathbb{C}^2$.

Remarque 25. Les propriétés de la physique quantique nous autorisent à prendre le spin magnétique d'un électron (*que l'on réduit à "haut" et "bas"*) comme vecteurs d'état.

3.1.3 Axiomes

Soit $n \in \mathbb{N}^*$. Soit $\mathcal{H}$ un espace de Hilbert de dimension n. Soit $\mathcal{B} = (|e_i\rangle)_{i \in [\![1;n]\!]}$ une base orthonormale de $\mathcal{H}$. Il est impossible de continuer si on ne définit pas clairement quelles opérations sont valides à la fois mathématiquement, mais aussi physiquement. On a ainsi les axiomes suivants :

Axiome 1 (Principe de superposition)**.** *Soit* $|\psi\rangle \in \mathcal{H}$. *Il existe* $(\alpha_i)_{i \in [\![1;n]\!]} \in \mathcal{H}$ *tel que* $|\psi\rangle = \sum_{k=1}^{n} \alpha_k |e_k\rangle$.

Axiome 2 (Mesure)**.** *Soit* $|\psi\rangle \in \mathcal{H}$ *dont on note* $(\alpha_i)_{i \in [\![1;n]\!]} \in \mathcal{H}$ *les coordonnées dans la base* $\mathcal{B}$. *Les amplitudes complexes de* $|\psi\rangle$ *vérifient* $\sum_{k=1}^{n} |\alpha_k|^2 = 1$.

Soit $i \in [\![1;n]\!]$. *La probabilité d'observer l'état* $|e_i\rangle$ *en mesurant* $|\psi\rangle$ *dans la base* $\mathcal{B}$ *est* $P(|e_i\rangle) = |\alpha_i|^2$.

Axiome 3 (Post-mesure)**.** *Soit* $|\psi\rangle \in \mathcal{H}$. *Soit* $i \in [\![1;n]\!]$. *On suppose que l'événement observer l'état* $|e_i\rangle$ *dans la base* $\mathcal{B}$ *est réalisé. Ainsi, après la mesure, l'état* $|\psi\rangle$ *reste projeté sur* $|e_i\rangle$.

Axiome 4 (Évolution en fonction du temps)**.** *Un état quantique* $|\psi\rangle$ *évolue en fonction du temps par une transformation unitaire. Ainsi, il existe* U *une transformation unitaire et* $|\psi_0\rangle$ *l'état initial de* $|\psi\rangle$ *tels que pour tout* $t \in \mathbb{R}^+$, $|\psi\rangle(t) = U(t) |\psi_0\rangle$.

Remarque 26. Il est impossible de dupliquer/copier un état quantique. En effet, cela reviendrait à lire et à figer l'état de superposition des états de base, ce qui est impossible d'après l'axiome de post-mesure. Autre remarque, la simple mesure d'un état ne donne aucun renseignement sur ses probabilités.

3.1.4 Systèmes composés

Maintenant que nous avons défini ce qu'est un qubit ainsi que les axiomes associés, nous pouvons étudier le comportement d'un système de plusieurs qubits. On rappelle comme dans la section 1.4 que le produit tensoriel de deux espaces vectoriels est un espace vectoriel.

Définition 98. Soit $n \in \mathbb{N}^*$. Soit $(\mathcal{H}_i)_{i \in [\![1;n]\!]}$ une famille d'espaces de Hilbert. Pour tout $i \in [\![1;n]\!]$, on pose $|\psi_i\rangle \in \mathcal{H}_i$. On définit l'espace de Hilbert $\mathcal{H}$ du système de n états quantiques $(|\psi_i\rangle)_{i \in [\![1;n]\!]}$ par $\mathcal{H} = \bigotimes_{i=1}^{n} \mathcal{H}_i$.

Remarque 27. Soit $n \in \mathbb{N}^*$. Soit $(|\psi_i\rangle)_{i \in [\![1;n]\!]}$ n états quantiques quelconques. Pour éviter de trop longues notations, on note $|\psi_1 \dots \psi_n\rangle = \bigotimes_{i=1}^{n} |\psi_i\rangle$.

Remarque 28. Pour un système à $n \in \mathbb{N}^*$ états quantiques à $d \in \mathbb{N}^*$ niveaux, l'espace résultant est de dimension d^n.

Dans la suite de cette partie, on ne travaille plus qu'avec des qubits. On reprend ainsi les notations de la section 3.1.2. Soit $n \in \mathbb{N}$ tel que $n > 1$.

Proposition 36. *Soit $|\psi\rangle$ un système composé de n qubits. Notons $E = \{\underbrace{0\dots0}_{n}, \dots, 1\dots1\}$ l'ensemble des nombres binaires dans l'intervalle entier $[\![0; 2^n - 1]\!]$. Ainsi, il existe $(\alpha_i)_{i\in E} \in \mathbb{C}^{2^n}$ tel que $\sum_{k\in E}|\alpha_k|^2 = 1$ et $|\psi\rangle = \sum_{i\in E} \alpha_i |i\rangle$.*

Proposition 37. *On reprend les mêmes notations que dans la proposition précédente. Soit $i \in [\![1; n]\!]$. Soit $k \in \mathbb{B}$. La probabilité de mesurer $|k\rangle$ pour le i-ème qubit de $|\psi\rangle$ est :*

$$P(\psi_i = |k\rangle) = \sum_{j\sim(0|1)^{i-1}k(0|1)^{n-i}} |\alpha_j|^2$$

en notant que $(0|1)^{i-1}k(0|1)^{n-i}$ est l'expression régulière d'une chaîne de n booléens dont le i-ème bit vaut k.

On suppose que l'on a mesuré le i-ème qubit et qu'on a obtenu $|k\rangle$. D'après l'axiome de post-mesure, on projette $|k\rangle$ sur ψ, d'où :

$$\psi \rightsquigarrow \frac{1}{P(\psi_i = |k\rangle)} \times \sum_{j\sim(0|1)^{i-1}k(0|1)^{n-i}} \alpha_j |j\rangle$$

Notons tout d'abord que la dimension de l'espace résultant augmente de manière exponentielle en fonction du nombre de qubits. Ainsi, un système composé de 10 qubits peut être vu comme un espace de dimension 1024. Or, d'après la proposition précédente, le fait de mesurer un qubit donné dans un tel système modifie son état général, ce qui n'est pas convenable pour des applications pratiques. Ainsi, pourquoi ne pas travailler avec des systèmes quantiques à 1024 niveaux qui ne subissent pas de tels effets ? La réponse, non triviale, repose sur un problème d'existence. En effet, existe-t-il un tel système quantique dans la nature ? Ensuite, le cas échéant, est-il possible de mesurer facilement et en temps constant un état donné ?

3.1.5 Intrication quantique

Définition 99. Soit $n \in \mathbb{N}^*$. Soit $\mathcal{H} = \bigotimes_{i=1}^{n} \mathcal{H}_i$ un espace de Hilbert composé. Soit $|\psi\rangle \in \mathcal{H}$. On dit que $|\psi\rangle$ est séparable ssi pour tout $i \in [\![1; n]\!]$, il existe $|\psi_i\rangle \in \mathcal{H}_i$ tels que $|\psi\rangle = \bigotimes_{i=1}^{n} |\psi_i\rangle$. Le cas contraire, on dit que $|\psi\rangle$ est intriqué ou enchevêtré.

Ainsi, le fait de mesurer un qubit donné dans un système composé dit séparable, n'a aucun effet sur la superposition des autres états.

3.1.6 Sphère de Bloch

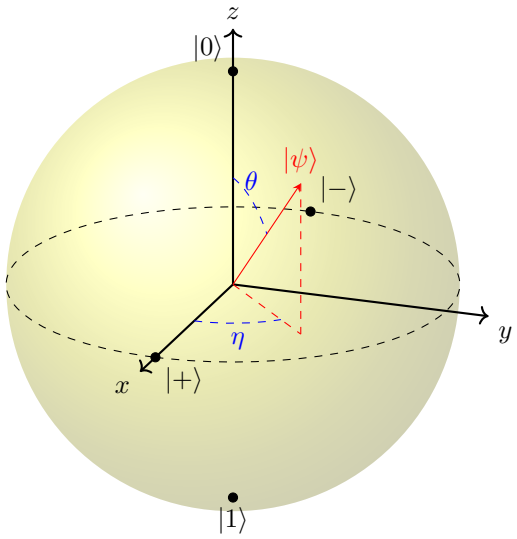
On rappelle que pour tout $z \in \mathbb{C}^*$, il existe un unique $(r, \theta) \in \mathbb{R}^+ \times [0; 2\pi[$ tel que $z = re^{i\theta}$. Notons que cette formule est valide également pour 0, mais on perd l'unicité car toute mesure d'angle avec un module nul peut donner 0. Ainsi, pour éviter des erreurs d'écriture, on considère que $\arg(0) = 0$. Soit $|\psi\rangle \in \mathbb{C}^2$ un qubit tel que $|\psi\rangle \neq \begin{pmatrix}0\\0\end{pmatrix}$. Donc, il existe $(\alpha, \beta) \in \mathbb{C}^2$ tel que $|\alpha|^2 + |\beta|^2 = 1$ et $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. On pose $\begin{cases} r_1 &= |\alpha| \\ \eta_1 &= \arg(\alpha) \end{cases}$, $\begin{cases} r_2 &= |\beta| \\ \eta_2 &= \arg(\beta) \end{cases}$ et $\eta = \eta_2 - \eta_1$. Ainsi, on a $\begin{cases} \alpha &= r_1 e^{i\eta_1} \\ \beta &= r_2 e^{i\eta_2} \end{cases}$. D'où :

$$\begin{aligned} |\psi\rangle &= r_1 e^{i\eta_1}|0\rangle + r_2 e^{i\eta_2}|1\rangle \\ &= r_1 e^{i\eta_1}|0\rangle + r_2 e^{i(\eta+\eta_1)}|1\rangle \\ &\equiv r_1|0\rangle + r_2 e^{i\eta}|1\rangle \qquad\qquad e^{i\eta} \text{ est une phase globale et peut être omise} \end{aligned}$$

Or, il existe $\theta \in [0; \pi]$ tel que $\begin{cases} r_1 &= \cos\left(\frac{\theta}{2}\right) \\ r_2 &= \sin\left(\frac{\theta}{2}\right) \end{cases}$. En effet, $r_1^2 + r_2^2 = \cos^2\left(\frac{\theta}{2}\right) + \sin^2\left(\frac{\theta}{2}\right) = 1$. De plus, comme r_1 et r_2 sont des modules ou nuls, on a $\begin{cases} r_1 &\geq 0 \\ r_2 &\geq 0 \end{cases}$ donc $\begin{cases} \cos\left(\frac{\theta}{2}\right) &\geq 0 \\ \sin\left(\frac{\theta}{2}\right) &\geq 0 \end{cases}$ d'où $\theta \in [0; \pi]$.

Finalement, pour tout $|\psi\rangle \in \mathbb{C}^2$ non nul, il existe un unique $(\eta, \theta) \in [0; 2\pi[\times[0; \pi]$ tel que $|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right) e^{i\eta}|1\rangle$.

Ainsi, il est possible d'exprimer un qubit comme un couple de mesures d'angle. On obtient alors une représentation géométrique que l'on appelle Sphère de Bloch (*ou Sphère de Poincaré*) :



Citons quelques exemples de points sur la sphère de Bloch :
— $|0\rangle_{(\theta,\eta)} = (0, 0)$
— $|1\rangle_{(\theta,\eta)} = (\pi, 0)$
— $|+\rangle_{(\theta,\eta)} = (\frac{\pi}{4}, 0)$
— $|-\rangle_{(\theta,\eta)} = (\frac{\pi}{4}, \pi)$

3.2 Circuits Quantiques

De manière analogue à la section 2.1, on définit un algorithme quantique comme un ensemble d'opérations appliquées sur une entrée pour produire une sortie.

3.2.1 Conventions de schéma

Dans cette partie, on explique les quelques conventions requises pour réaliser ou lire un diagramme de circuit quantique. Tout d'abord, on considère que le temps évolue de gauche à droite positivement. Ainsi, les portes sont placées dans l'ordre chronologique de lecture du diagramme.

On considère que l'entrée d'un circuit quantique est encodée par un état quantique, que l'on appelle registre quantique. On représente un registre par un fil (*trait fin continu*) et on place en regard à gauche le nom et/ou la valeur initiale du registre. On représente de plus un registre classique par un double trait.

On représente une porte quantique par un rectangle dans lequel se trouve le nom de la porte ou de la transformation unitaire associée. De plus, on peut également préciser certains paramètres avec une fonte plus petite tels que la phase d'une porte. Une porte agit sur un ou plusieurs registres quantiques, et produit une sortie.

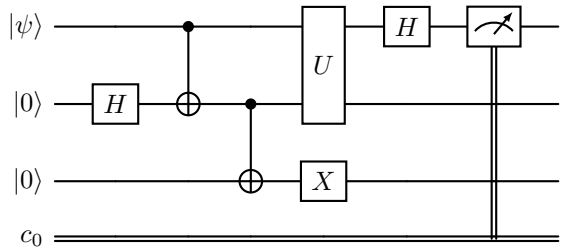
Une porte contrôlée est une porte quantique qui agit sur au moins deux qubits, un qui sert à contrôler la porte (*que l'on appelle qubit de contrôle*) et un autre sur lequel on applique une transformation. On la représente par le symbole qui lui est associé sur les qubits modifiés, ainsi que par un petit disque noir sur les qubits de contrôle. Par ailleurs, ces disques sont reliés à la porte par des traits fins, perpendiculaires aux registres quantiques.

On représente également l'opérateur de mesure (*ou compteur*) qui prend un registre quantique en entrée, et qui retourne une valeur définie dans un registre classique. Quand le registre correspondant est représenté

sur le schéma, on place une flèche ou un double trait, de l'opérateur vers le registre, perpendiculaire aux autres fils.

Enfin, un trait incliné à travers un registre, avec un entier naturel non nul indiqué en exposant, précise le nombre de registres utilisés.

Voici un exemple de circuit quantique :



3.2.2 Portes quantiques

Une porte quantique n'est rien d'autre qu'une transformation unitaire, que l'on peut exprimer sous forme matricielle. La question est de savoir si les opérateurs logiques usuels existent dans le monde quantique. En effet, on a vu dans la section 2.2.2 que tout algorithme peut être créé numériquement à l'aide des portes NOT, AND et OR. Ainsi, est-ce aussi le cas pour les algorithmes quantiques. Or, un premier problème se pose : ces opérateurs ne sont pas bijectifs (*e.g si on a deux propositions A et B telles que $A \vee B = 1$, il n'est pas possible de savoir $A = 1$ ou $B = 1$ sans plus d'informations*), donc irréversibles. Néanmoins, nous allons montrer qu'il est simple de les rendre réversibles. Avant cela, il est nécessaire de définir des portes élémentaires.

Définition 100. On appelle porte de Hadamard, et on note H, la matrice unitaire $\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$. Elle permet de passer de la base canonique de $\mathbb{C}^2$ à la base de Hadamard (*et inversement*).

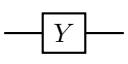
Son symbole est :

Définition 101. On appelle porte X de Pauli, et on note X, la matrice unitaire $\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$. Elle permet d'inverser $|0\rangle$ en $|1\rangle$ (*et inversement*) et agit donc comme un NON logique. Sur la sphère de Bloch, cela revient à appliquer une rotation de π radians selon l'axe (Ox).

Son symbole est :

Définition 102. On appelle porte Y de Pauli, et on note Y, la matrice unitaire $\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}$. Elle permet d'inverser $|0\rangle$ en $i\,|1\rangle$ et $|1\rangle$ en $-i\,|0\rangle$. Sur la sphère de Bloch, cela revient à appliquer une rotation de π radians selon l'axe (Oy).

Son symbole est :



Définition 103. On appelle porte Z de Pauli, et on note Z ou R_π, la matrice unitaire $\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$. Elle permet d'inverser uniquement $|1\rangle$ en $-|1\rangle$. Sur la sphère de Bloch, cela revient à appliquer une rotation de π radians selon l'axe (Oz).

Son symbole est :

Définition 104. Soit $\phi \in [0; 2\pi[$. On appelle porte changement de phase, et on note R_ϕ, la matrice unitaire $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$. Elle permet d'inverser uniquement $|1\rangle$ en $e^{i\phi}|1\rangle$.

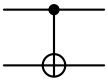
Son symbole est :

Définition 105. On appelle porte Swap, et on note S, la matrice unitaire $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$. Elle permet d'intervertir les deux qubits sur lesquels elle agit.

Son symbole est :

Définition 106. On appelle porte NON contrôlée, et on note $CNOT$ ou cX, la matrice unitaire $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$. Elle agit ainsi sur deux qubits et applique une porte X de Pauli sur le premier registre, si le deuxième se trouve à $|1\rangle$.

Son symbole est :



Définition 107. On appelle portes universelles tout ensemble de portes quantiques qui permet de représenter par un nombre fini de portes toute opération réalisable sur un ordinateur quantique.

Théorème 11. *Soit $n \in \mathbb{N}^*$. On note $\mathcal{U}_n$ l'ensemble des matrices unitaires d'ordre n. Ainsi, $\mathcal{U}_2$ (i.e l'ensemble des portes agissant sur un seul qubit) et la porte $CNOT$ sont universelles.*

Remarque 29. Il existe d'autres variantes de ce théorème où la porte $CNOT$ est remplacée par une porte équivalente (*e.g une porte* $\sqrt{SWAP}$).

Remarque 30. Ainsi, tout opérateur unitaire de $\mathcal{U}_{2^n}$ qui agit sur n qubits peut être décomposé en un ensemble de portes $CNOT$ et de portes agissant sur un seul qubit.

Remarque 31. Soit U une matrice unitaire d'ordre 2 dont on note $U^\dagger$ (*ou* U^* *en algèbre*) la matrice adjointe de U. Soit $|x\rangle$ un qubit dont on note $|y\rangle$ l'image par U. Ainsi, comme $U^\dagger U = I_2$, on obtient $|x\rangle = U^\dagger |y\rangle$ d'où la réversibilité de U.

Il est possible de rendre un opérateur réversible. Pour cela, il suffit de stocker une information supplémentaire afin de le rendre bijectif. On a alors la définition suivante :

Définition 108. On appelle registre auxiliaire (*ou ancilla*) un qubit utilisé pour stocker une information.

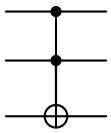
Comme les opérateurs OR et AND agissent sur deux entrées, il est nécessaire d'en ajouter une troisième afin de les rendre réversibles. Ainsi, il existe une unique porte qui applique ces opérations.

Définition 109. On appelle porte de Toffoli, notée $CCNOT$ ou $TOFFOLI$, la matrice unitaire :

$$\begin{pmatrix} 1 & 0 & \dots & & & & \dots & 0 \\ 0 & 1 & \ddots & & & & & \vdots \\ \vdots & \ddots & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & \ddots & & \vdots \\ & & & & \ddots & 1 & 0 & 0 \\ \vdots & & & & & 0 & 0 & 1 \\ 0 & \dots & & & \dots & 0 & 1 & 0 \end{pmatrix}$$

Cette porte agit sur le troisième registre, et prend les deux premiers en registres de contrôle.

Son symbole est :



Proposition 38. *Soit $|abc\rangle$ un système à trois qubits. On note $|a'b'c'\rangle$ l'image de $|abc\rangle$ par la matrice de Toffoli.Pour toutes les valeurs possibles de $|abc\rangle$, on obtient la "table de vérité" suivante :*

$\|abc\rangle$	$\|a'b'c'\rangle$
$\|000\rangle$	$\|000\rangle$
$\|001\rangle$	$\|001\rangle$
$\|010\rangle$	$\|010\rangle$
$\|011\rangle$	$\|011\rangle$
$\|100\rangle$	$\|100\rangle$
$\|101\rangle$	$\|101\rangle$
$\|110\rangle$	$\|111\rangle$
$\|111\rangle$	$\|110\rangle$

D'où :

1. $(a = b = 1) \implies (c' = \neg c)$.
2. $(c = 0) \implies (c' = a \wedge b)$.
3. $(c = 1) \implies (\neg(a \wedge b))$.
4. *Le fait d'avoir $c = 0$ et de place deux portes de Toffoli à la suite permet d'obtenir $a \vee b$ sur le registre auxiliaire.*

En somme, $TOFFOLI$ renvoie $|a\rangle \otimes |b\rangle \otimes |c \oplus (a \wedge b)\rangle$ où $\oplus$ désigne un "ou" exclusif.

3.3 Algorithmes Quantiques

Dans cette partie, nous présentons quelques algorithmes quantiques célèbres et leur apport face à l'algorithmie classique.

3.3.1 Complexités de requête et de circuit

De manière analogue à la section 2.1.4, on définit la complexité d'un algorithme quantique comme une mesure des ressources requises pour l'exécution de l'algorithme.

On distingue ainsi pour un circuit quantique, complexités de circuit et complexités de requête, d'où les définitions suivantes :

Définition 110. Soit $n \in \mathbb{N}^*$. On définit la complexité d'un circuit à n qubits comme le nombre de portes élémentaires requises pour effectuer des opérations sur les registres d'entrée.

Définition 111. Soit $n \in \mathbb{N}^*$. Soit $f : \mathbb{B}^n \to \mathbb{B}$ dont on note U_f la transformation unitaire. Dans un circuit quantique, on définit une porte U_f dite oracle ou boîte noire, dans laquelle on ne peut voir les portes élémentaires qui la composent. On sait néanmoins que pour tout registre $|x\rangle$ à n qubits, et un registre auxiliaire $|y\rangle$, on a $U_f |x, y\rangle = |x, y \oplus f(x)\rangle$.

Définition 112. Soit $n \in \mathbb{N}^*$. Pour un circuit à n qubits faisant intervenir au moins un oracle, on définit la complexité de requête comme le nombre d'appels à l'oracle lors de l'exécution de l'algorithme sur un registre de taille n.

Proposition 39. *Soit $|x\rangle$ un registre quantique. Soit f une transformation unitaire. On appelle oracle de phase U_f, la porte dont l'entrée est $|x\rangle$ et dont l'ancilla est placé à $|-\rangle$. Ainsi, $U_f |x, -\rangle = \left|(-1)^{f(x)} |x\rangle, |-\rangle\right\rangle$.*

Démonstration. Soit $|x\rangle$ un registre quantique. Soit f une transformation unitaire dont on note U_f la matrice unitaire.

$$\begin{aligned} |x\rangle |-\rangle &= |x\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\ &= \frac{1}{\sqrt{2}} (|x\rangle |0\rangle - |x\rangle |1\rangle) \end{aligned}$$

D'où :

$$\begin{aligned} U_f |x\rangle |-\rangle &= U_f \frac{1}{\sqrt{2}} (|x\rangle |0\rangle - |x\rangle |1\rangle) \\ &= \frac{1}{\sqrt{2}} (U_f |x\rangle |0\rangle - U_f |x\rangle |1\rangle) \\ &= \frac{1}{\sqrt{2}} (|x\rangle |0 \oplus f(x)\rangle - |x\rangle |1 \oplus f(x)\rangle) \\ &= \frac{1}{\sqrt{2}} (|x\rangle |f(x)\rangle - |x\rangle |\neg f(x)\rangle) \\ &= \begin{cases} |x\rangle |-\rangle & \text{si } f(x) = 0 \\ -|x\rangle |-\rangle & \text{si } f(x) = 1 \end{cases} \\ &= (-1)^{f(x)} |x\rangle |-\rangle \end{aligned}$$

D'où le résultat. □

3.3.2 Algorithme de Deutsch

En 1985, le physicien David Deutsch propose un des premiers algorithmes quantiques qui montre l'intérêt de l'application de la physique quantique en informatique. L'intitulé du problème est très simple : Soit $f : \mathbb{B} \to \mathbb{B}$. Est-ce que f est constante ? De manière intuitive, pour répondre à cette question, il suffit de vérifier que $f(0) = f(1)$.

Ainsi, on a l'algorithme suivant :

Algorithme 1: Vérifier si une fonction booléenne est constante

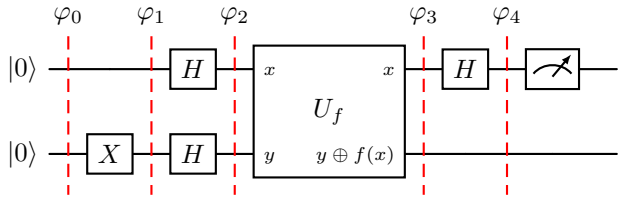
Entrée: $f : \mathbb{B} \to \mathbb{B}$
Sortie: $\mathbb{B}$
Fonction *estConstante(f : fonction<$\mathbb{B}$, $\mathbb{B}$>) : booléen*
Début
  tmp : booléen $\leftarrow f(0)$;
  estConstante $\leftarrow (tmp - f(1) = 0)$;
Fin

Or, on constate que l'on fait appel à deux reprises à la fonction f. Néanmoins, il est possible de ne faire qu'un appel à f en utilisant le principe de superposition quantique.
On a ainsi l'algorithme suivant :

Algorithme 2: Vérifier si une fonction booléenne est constante

Entrée: $U_f \in \mathcal{U}_4$
Registres: $|x\rangle$
Ancilla: $|y\rangle$
Sortie: $\mathbb{B}$
Fonction *deutschAlgorithm(U_f : matrice unitaire d'ordre 4) : booléen*
Début
  $|xy\rangle \leftarrow |00\rangle$;
  $|y\rangle \leftarrow X\,|y\rangle$;
  $|xy\rangle \leftarrow (H \otimes H)\,|xy\rangle$;
  $|xy\rangle \leftarrow U_f\,|xy\rangle$;
  $|x\rangle \leftarrow H\,|x\rangle$;
  res : booléen $\leftarrow$ mesure($|x\rangle$);
  deutschAlgorithm $\leftarrow (res = 0)$;
Fin

D'où le circuit :



Procédons désormais à la démonstration de l'algorithme, en reprenant les notations du diagramme.

— $|\varphi_0\rangle = |00\rangle$
— $|\varphi_1\rangle = |01\rangle$
— $|\varphi_2\rangle = (H \otimes H)\,|\varphi_1\rangle = |+-\rangle = \frac{1}{\sqrt{2}}(|0\rangle\,|-\rangle + |1\rangle\,|-\rangle)$
— $|\varphi_3\rangle = U_f \frac{1}{\sqrt{2}}(|0\rangle\,|-\rangle + |1\rangle\,|-\rangle) = \frac{1}{\sqrt{2}}(U_f\,|0\rangle\,|-\rangle + U_f\,|1\rangle\,|-\rangle) = \frac{1}{\sqrt{2}}((-1)^{f(0)}\,|0\rangle + (-1)^{f(1)}\,|1\rangle)\,|-\rangle$.
  Donc $|\varphi_3\rangle \equiv \frac{(-1)^{f(0)}}{\sqrt{2}}\left(|0\rangle + (-1)^{f(1)-f(0)}\,|1\rangle\right)$
  Si f est constante, $|\varphi_3\rangle \equiv \frac{(-1)^{f(0)}}{\sqrt{2}}(|0\rangle + |1\rangle)$.
  Sinon, $|\varphi_3\rangle \equiv \frac{(-1)^{f(0)}}{\sqrt{2}}(|0\rangle - |1\rangle)$.
— $|\varphi_4\rangle = \begin{cases} |0\rangle\,|-\rangle & \text{si } f(0) = f(1) \\ |1\rangle\,|-\rangle & \text{sinon} \end{cases}$.

3.3.3 Algorithme de Deutsch-Jozsa

Définition 113. Soit $n \in \mathbb{N}^*$. Soit $f \in \mathbb{B}^n \to \mathbb{B}$. On dit que f est équilibrée ssi $\text{Card}(f^{-1}(\{0\})) = \text{Card}(f^{-1}(\{1\}))$.

En 1992, David Deutsch et Richard Jozsa ont généralisé l'algorithme de Deutsch (3.3.2) en essayant de résoudre le problème suivant : Soit $n \in \mathbb{N}^*$. Soit $f : \mathbb{B}^n \to \mathbb{B}$. Est-ce que f est constante ou équilibrée ?

On obtient ainsi l'algorithme suivant :

Algorithme 3: Vérifier si une fonction booléenne est constante ou équilibrée

Entrée: $n \in \mathbb{N}^*, f : \mathbb{B}^n \to \mathbb{B}$
Sortie: Renvoie 0 si la fonction est constante et 1 si elle est équilibrée
Fonction *estEquilibree(n : entier non signé, f : fonction<$\mathbb{B}^n$, $\mathbb{B}$>) : booléen*
Début
  N : entier non signé $\leftarrow 2 * *n$;
  i : entier non signé;
  res : entier $\leftarrow 0$;
  Pour i **de** *0* **à** *N-1* **Faire**
    x : chaîne de n bits $\leftarrow$ toBinary(i);
    Si *f(x)* **Alors**
      $res \leftarrow res + 1$;
    Sinon
      $res \leftarrow res - 1$;
    Fin Si
  Fin Pour
  estEquilibree $\leftarrow (res = 0)$;
Fin

Remarque 32. Dans le cas où on suppose que la fonction f est soit constante, soit équilibrée, il suffit de vérifier $2^{n-1} + 1$ images pour en déterminer la nature.

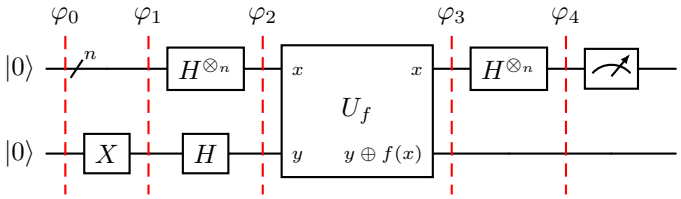
On remarque très rapidement que l'algorithme est de complexité exponentielle ($\mathcal{O}(2^n)$), ce qui est extrêmement mauvais. Or, il est possible de ne faire qu'un appel à la fonction f et donc de passer en $\Theta(1)$.

On a ainsi l'algorithme suivant :

Algorithme 4: Vérifier si une fonction booléenne est constante ou équilibrée

Entrée: $n \in \mathbb{N} \setminus \{0; 1\}, U_f \in \mathcal{U}_{2^n}$
Registres: $|x\rangle = |x_1 \dots x_n\rangle$
Ancilla: $|y\rangle$
Sortie: Renvoie 1 si la fonction est constante et 0 si elle est équilibrée
Fonction *deutschJAlgorithm(n : entier > 1, U_f : matrice unitaire d'ordre 2^n) : booléen*
Début
  $|xy\rangle \leftarrow |0\rangle^{\otimes_{n+1}}$;
  $|y\rangle \leftarrow X|y\rangle$;
  $|xy\rangle \leftarrow H^{\otimes_{n+1}}|xy\rangle$;
  $|xy\rangle \leftarrow U_f|xy\rangle$;
  $|x\rangle \leftarrow H^{\otimes_n}|x\rangle$;
  res : chaîne de n bits $\leftarrow$ mesure($|x\rangle$);
  deutschJAlgorithm $\leftarrow (res = |0\rangle^{\otimes_n})$;
Fin

D'où le circuit :



Procédons désormais à la démonstration de l'algorithme, en reprenant les notations du diagramme.

— $|\varphi_0\rangle = |0\rangle^{\otimes_n} \otimes |0\rangle$

— $|\varphi_1\rangle = |0\rangle^{\otimes_n} \otimes |1\rangle$

— $|\varphi_2\rangle = H^{\otimes_n} |0\rangle^{\otimes_n} |-\rangle = |+\rangle^{\otimes_n} |-\rangle = \frac{1}{\sqrt{2^n}} (|0\dots0\rangle + |0\dots01\rangle + \dots + |1\dots1\rangle) |-\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{B}^n} |x\rangle |-\rangle$

— $|\varphi_3\rangle = U_f \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{B}^n} |x\rangle |-\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{B}^n} U_f |x\rangle |-\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{B}^n} (-1)^{f(x)} |x\rangle |-\rangle$

— $|\varphi_4\rangle \equiv \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{B}^n} (-1)^{f(x)} H^{\otimes_n} |x\rangle$

Or pour tout $x \in \mathbb{B}^n, H^{\otimes_n} |x\rangle = H |x_1\rangle \otimes \dots \otimes H |x_n\rangle$ avec pour tout $i \in [\![1; n]\!], H |x_i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_i} |1\rangle)$.

En notant . le produit scalaire de deux booléens (bit par bit), on a :

$$|\varphi_4\rangle \equiv \frac{1}{2^n} \sum_{x\in\mathbb{B}^n} (-1)^{f(x)} \sum_{z\in\mathbb{B}^n} (-1)^{x.z} |z\rangle \equiv \frac{1}{2^n} \sum_{(x,z)\in(\mathbb{B}^n)^2} (-1)^{x.z+f(x)} |z\rangle$$

Pour tout $z \in \mathbb{B}^n$, notons A_z l'amplitude de l'état $|z\rangle$ $\frac{1}{2^n} \sum_{x\in\mathbb{B}^n} (-1)^{x.z+f(x)}$. Considérons $A_{0\dots0} = \frac{1}{2^n} \sum_{x\in\mathbb{B}^n} (-1)^{f(x)}$. Notons que si f est constante, alors $A_{0\dots0} = \pm1$ d'où une probabilité de mesurer $|0\rangle^{\otimes_n}$ de 1 ; on parle d'interférences constructives. Si f est équilibrée, alors $A_{0\dots0} = 0$ (car autant de -1 que de 1) d'où une probabilité de mesurer $|0\rangle^{\otimes_n}$ de 0 ; on parle d'interférences destructives.

Ainsi, dans le cas où l'amplitude $A_{0\dots0}$ vaut 1 ou -1, toutes les autres amplitudes sont nulles conformément à l'axiome de mesure.

Finalement, $\begin{cases} f \text{ est constante} & \text{si } \varphi_4 = |0\rangle^{\otimes_n} |-\rangle \\ f \text{ est équilibrée} & \text{sinon} \end{cases}$.

3.3.4 Analyse

On remarque ainsi que pour un tel problème, l'algorithmie quantique fournit de meilleurs résultats que son analogue classique. Néanmoins, il ne faut pas partir du principe que les calculs quantiques sont nécessairement mieux et plus efficaces que les calculs classiques. En effet, on ne peut pas directement comparer complexité de circuit et complexité temporelle, car les concepts sont fondamentalement différents. En effet, dans un cas, on fait des calculs déterministes (*on est surs d'obtenir toujours le bon résultat*), mais cela peut prendre du temps. Dans l'autre cas, on traite un volume de données important afin d'obtenir le résultat le plus probable pour résoudre un problème ; au final, le processus est important, et pas vraiment le résultat (*pour l'algorithme précédent, on sait qu'une fonction est constante ou non, néanmoins, on ignore si elle constante égale à 0 ou à 1*).

3.3.5 Autres algorithmes

Il existe également d'autres algorithmes quantiques qui ont une application plus intéressante que les algorithmes de Deutsch et de Deutsch-Jozsa. On peut ainsi citer l'algorithme de Grover qui permet de rechercher un ou plusieurs éléments donnés parmi un ensemble indexé de 2^n objets non classés avec une complexité en $\mathcal{O}(\sqrt{2^n})$, contre $\mathcal{O}(2^n)$ pour son analogue classique. Le circuit est composé d'un oracle qui vérifie si un état d'entrée correspond au critère de recherche, et d'un opérateur d'amplification d'amplitude. L'oracle et l'amplificateur sont répétés $\mathcal{O}(\sqrt{2^n})$ fois et fournissent après mesure l'élément répondant au critère de recherche avec une forte probabilité. L'amplificateur ou opérateur de diffusion de Grover, est une matrice unitaire d'ordre 2^n et d'expression $H^{\otimes_n}(2 |0\rangle^{\otimes_n} \langle 0|^{\otimes_n} - I_{2^n})H^{\otimes_n}$.

Enfin, on peut également citer l'algorithme de Shor qui à partir d'un nombre très grand n tel qu'il existe p et q deux nombres premiers tels que $n = pq$, renvoie le couple (p, q) en temps polynomial, contre exponentiel pour son analogue classique. Cet algorithme est assez important, car il s'agit du premier capable de "casser" l'encryption RSA, utilisée de nos jours afin de sécuriser tout échange de données entre deux machines.

Chapitre 4

Expérimentation

Afin de conclure notre étude, nous avons décidé de mener une petite expérience. Le but est de montrer sur un exemple simple quelles peuvent être les différences entre informatique classique et informatique quantique.

4.1 Intitulé du problème

On note $\mathbb{B} = \{0; 1\}$. Soit $n \in \mathbb{N}^*$. Soit $(c_i)_{i \in [\![1;n]\!]} \in \mathbb{B}^n$ et $d \in \mathbb{B}$.

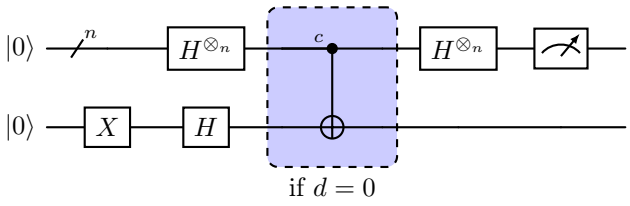
Let $$\begin{array}{rccl} f: & \mathbb{B}^n & \to & \mathbb{B} \\ & (x_1, \dots, x_n) & \mapsto & \begin{cases} \bigoplus_{i=1}^{n} \delta_{c_i,1} x_i & \text{si } d = 0 \\ 0 \text{ ou } 1 & \text{sinon} \end{cases} \end{array}$$

Ainsi, le but du problème est de générer aléatoirement cette fonction f sans avoir son expression et de vérifier si elle est équilibrée ou constante.

4.2 Algorithmes

On reprend l'algorithme de Deutsch-Jozsa pour la partie quantique, ainsi que l'algorithme classique de la partie 3.3.3 pour comparaisons pour répondre à la question.

Soit $|x\rangle = |x_1 \dots x_n\rangle$ et $|y\rangle$ deux registres quantiques. Ainsi, f peut être vue comme la transformation unitaire U_f qui ne fait rien si $d = 1$, et sinon qui pour tout $i \in [\![1; n]\!]$ applique une porte $CNOT$ sur $|y\rangle$ avec le qubit de contrôle mis sur $|x_i\rangle$. La présence de $|y\rangle$ rend la transformation réversible. On obtient ainsi le circuit suivant :



4.3 Scripts & Qiskit

Afin de pouvoir "comparer" les résolutions de ce problème avec informatique classique et calculs quantiques, nous utiliserons le langage interprété Python 3.12 ainsi que l'outil Qiskit d'IBM. Ce dernier permet de créer simplement et rapidement des circuits quantiques et de les envoyer vers une unité de traitement quantique (QPU). Pour cela, Qiskit procède en quatre étapes.

Tout d'abord, on définit un circuit quantique avec un certain nombre de registres quantiques mais également un certain nombre de registres classiques afin de stocker le résultat évalué. Ensuite, il se connecte au service IBM afin de récupérer l'identifiant d'une machine quantique, non trop occupée. À partir d'ici, il

est nécessaire de "transpiler" le circuit. En effet, tout QPU a une structure et des portes différentes ; il est donc obligatoire d'adapter son circuit en fonction des portes disponibles. Le théorème 11 assure que cette opération est licite. Ensuite, le circuit transpilé est envoyé sur la file de travaux du QPU, et est ensuite évalué. En raison de la nature probabiliste des calculs réalisés (*et donc un risque d'obtenir des erreurs*), l'expérience est répétée un certain nombre de fois. Enfin, les résultats sont renvoyés au client.

Dans notre cas, le client utilisé (*qui exécute la fonction classique*) dispose d'une unité de calcul 12 cœurs et une fréquence d'horloge de $4,15GH_Z$. En moyenne, ici un processus Python utilise 5% du processeur. Les QPU utilisés disposent de 127 qubits, d'un volume quantique de 128 (*i.e le plus grand circuit aléatoire donnant les résultats attendus*) et de 30 000 opérations de couche de circuit par seconde (*i.e la vitesse de traitement d'un circuit quantique*).

On choisit un niveau modéré d'optimisation de la transpilation de circuit. En sus de cela, on choisit de répéter l'expérience 1000 fois, pour un nombre de qubits variant de 4 à 64.

Le script Python ainsi que le carnet Jupyter sont disponibles ici.

4.4 Résultats

L'expérience a été interrompue à partir de 34 qubits pour des raisons expliquées ci-dessous. On dispose ainsi d'un échantillon de 16 résultats.

4.4.1 Constante ou équilibrée ?

La probabilité qu'une fonction soit constante est de 0,5625 (*soit 56,25%*). On obtient de plus le tableau suivant :

n	$P(\|0\rangle^{\otimes_n})$	Constante ?
4	0,7%	Non
6	0,1%	Non
8	0,5%	Non
10	97,4%	Oui
12	96,7%	Oui
14	95,6%	Oui
16	95,2%	Oui
18	0%	Non
20	0%	Non
22	92%	Oui
24	0%	Non
26	90,1%	Oui
28	87,7%	Oui
30	93,4%	Oui
32	84,2%	Oui
34	0%	Non

Il y a donc une forte probabilité d'obtenir le bon résultat.

4.4.2 Temps d'exécution

Sur l'échantillon, les calculs quantiques se sont effectués en temps constant (*2 secondes*). Pour la fonction classique, on obtient le tableau suivant :

n	$T(n)$ (en s)
4	0,000
6	0,000
8	0,000
10	0,000
12	0,000
14	0,010
16	0,048
18	0,257
20	1,124
22	3,965
24	20,920
26	73,429
28	316,857
30	1335,983
32	6324,900
34	29727≈ 20 jours

On peut ainsi effectuer une régression "puissance" sur les données ci-dessus, d'où le graphe suivant :

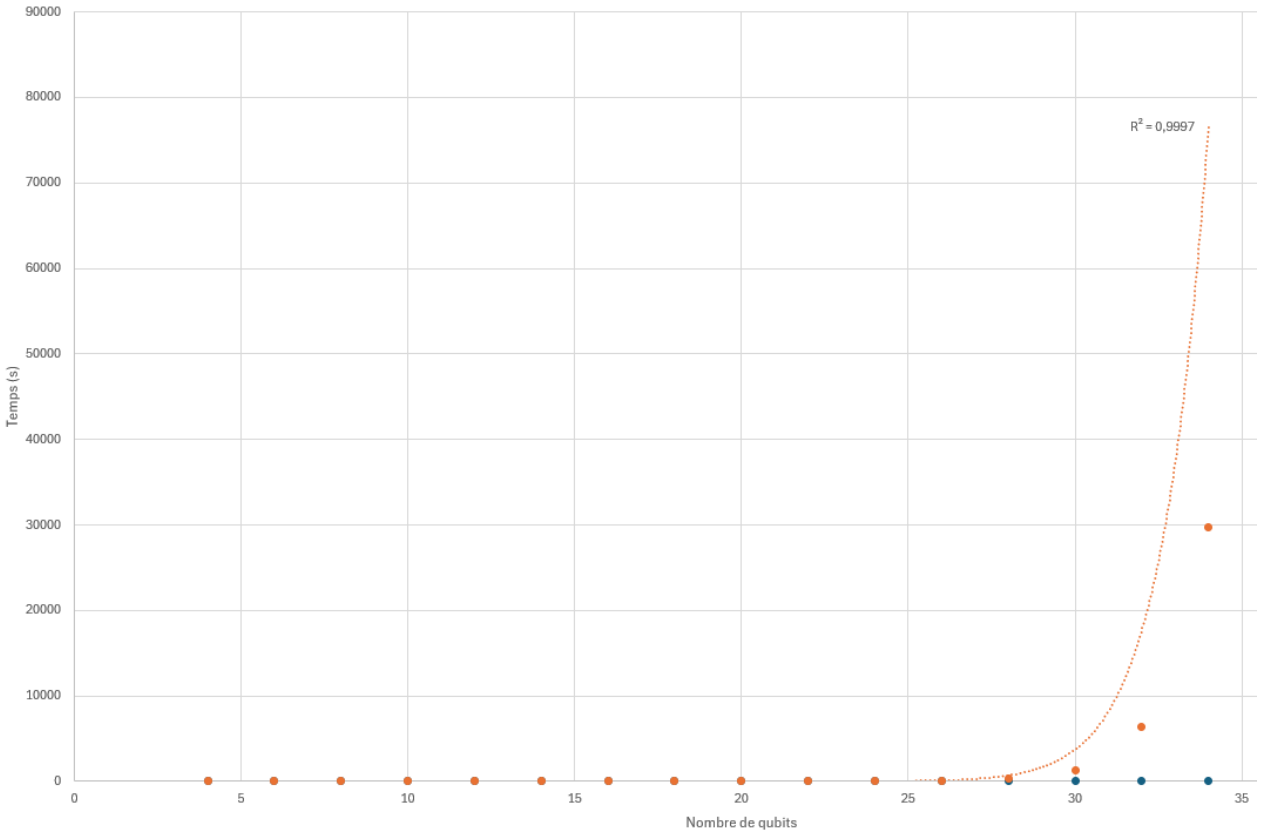


FIGURE 4.1 – Graphe du temps d'exécution (en s) en fonction du nombre de qubits

Un coefficient de détermination R^2 supérieur à 0,99 indique que les données sont très proches du modèle choisi (*ici, fonction puissance*).

4.5 Analyse des résultats

On constate que le problème classique devient extrêmement long à calculer à partir de $n = 32$. Pour $n = 34$, il faudrait environ 20 jours pour que la fonction termine, et ce dans le meilleur des cas. Notons de plus que la résolution du problème commence à devenir perceptible à partir de $n = 22$. Néanmoins, il est à noter que ces résultats dépendent de plusieurs facteurs. On peut citer la vitesse de traitement du processeur ainsi que le taux d'utilisation du CPU pour la tâche associée. Aussi, n'oublions pas que l'algorithme a été implémenté en Python, qui est un langage interprété. Ainsi, les performances sont biaisées par défaut dans la mesure où un langage compilé, comme le C, a en général de bien meilleures performances. Il faut toutefois garder en tête que dans ce cas, le gain est négligeable.

Cependant, on note que peu importe le nombre de qubits, l'exécution totale se fait toujours en deux secondes. Sachant que l'expérience est répétée 1000 fois, on peut estimer que le circuit est exécuté en 2 ms. Ainsi, cela vérifie bien la remarque dans la section 3.3.4.

Conclusion

In fine, l'informatique quantique constitue une avancée qui pourrait transformer les secteurs où l'on traite de grands jeux de données, à savoir, la médecine, l'intelligence artificielle, la chimie, la physique fondamentale, mais aussi la cryptographie. Malgré le fait que cette science se trouve encore au début de son développement, de nombreux progrès, aussi bien sur le plan technologiques que sur le plan théorique, ont été réalisés depuis 2020 (*notamment par IBM*) permettent de faire avancer le sujet.

Néanmoins, il demeure encore des problèmes importants qui ne peuvent être négligés, à savoir la stabilité des systèmes quantiques utilisés (*i.e qubits*) et la correction des erreurs.

En somme, même si l'informatique quantique n'est pas encore prête à remplacer l'informatique classique, notamment dans le domaine du civil, elle permet tout de même de résoudre des problèmes, jusque-là considérés comme impossibles.

Bibliographie

[1] David Deutsch (1985), *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer*, Proceedings of the Royal Society of London A

[2] David Deutsch, Richard Jozsa (1992), *Rapid solution of problems by quantum computation*, Proceedings of the Royal Society of London A

[3] Fréderic Holweck (2021), *IT41 Lecture*, Université de Technologie de Belfort-Montbéliard

[4] IBM Quantum Platform (2025), *Qiskit Documentation* [en ligne] (https://docs.quantum.ibm.com/guides), International Business Machines Corporation

[5] Michael A. Nielsen & Isaac L. Chuang (2010), *Quantum Computation and Quantum Information*, Cambridge University Press, 2e ed.

[6] Xavier Gourdon (2009), *les maths en tête ALGÈBRE*, Ellipses Éditions Marketing S.A., 2e ed.

[7] Xavier Gourdon (2008), *les maths en tête ANALYSE*, Ellipses Éditions Marketing S.A., 2e ed.