À la découverte des algorithmes quantiques

Table des matières

1 Introduction : Principe de fonctionnement des algorithmes classiques **2**

2 Le qubit **2**

3 Les portes quantiques : principe et exemples **3**
- 3.1 Opérateur dague † . . . 3
- 3.2 Les portes quantiques . . . 5
- 3.3 Les circuits quantiques . . . 5
- 3.4 Les principales portes quantiques . . . 6
- 3.5 Appliquer des portes quantiques sur plusieurs systèmes de qubits en parallèle . . . 10

4 Mesure **11**
- 4.1 Définition . . . 11
- 4.2 Exemples de circuit quantiques et de mesures . . . 11

5 L'algorithme de Deutsch-Jozsa **14**
- 5.1 Principe . . . 14
- 5.2 Circuit quantique . . . 14
- 5.3 État initial du système : en ψ_0 . . . 14
- 5.4 Après la première porte : en ψ_1 . . . 15
- 5.5 Après la boîte noire : en ψ_2 . . . 15
- 5.6 État final : en ψ_3 . . . 15
- 5.7 Mesure des n premiers qubits . . . 16
- 5.8 Programmation de l'algorithme de Deutsh-Jozsa pour $n = 1$ (algorithme de Deutsch) avec Qiskit 17

6 Bibliographie **19**

1 Introduction : Principe de fonctionnement des algorithmes classiques

Dans un algorithme classique, toutes les informations sont codées sous la forme d'une succession de 0 et de 1 : les bits. Ce système de codage en binaire (système de calcul en base 2) est ce que l'on nommera dans ce document « système classique ». Ce que nous voyons sur une interface de programmation n'est qu'un moyen permettant de faciliter la « communication » entre le programmeur et la machine, en utilisant des mots plutôt que des chiffres, qui sont ensuite interprétés par l'ordinateur.
Un algorithme consiste en un ensemble de bits sur lesquels sont appliquées différentes opérations logiques : ce sont les sept portes logiques, à partir desquelles tous les algorithmes que nous utilisons quotidiennement sont conçus. Ces portes sont : NOT, AND, OR, XOR, NAND, NOR, XNOR. Elles reçoivent deux bits en entrée (exceptée la porte NOT, qui n'en reçoit qu'un), et donnent un bit en sortie, dont la valeur varie en fonction des valeurs d'entrée. En ce sens, elles agissent de la même manière qu'une fonction.

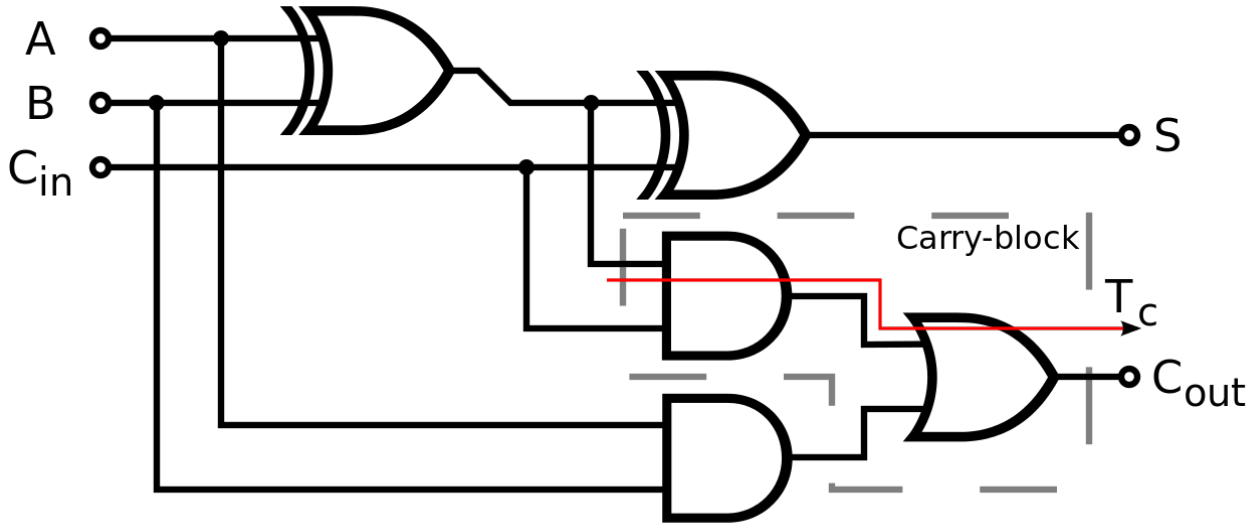


FIGURE 1 – Exemple de circuit classique : Additionneur (image reprise de : *https* : *//en.wikipedia.org/wiki/Adder_(electronics)#/media/File* : *Full − adder_logic_diagram.svg*

FIGURE 2 – De gauche à droite : Portes XOR, AND et OR (images reprises de : *https* : *//whatis.techtarget.com/definition/logic−gate−AND−OR−XOR−NOT−NAND−NOR−and−XNOR*)

Le but de ce système est de sommer deux bits A et B, ainsi qu'un bit de retenue C_{in}, et de sortir un bit de résultat S et un de retenue C_{out}.

2 Le qubit

De la même manière qu'un algorithme classique utilise des bits, un algorithme quantique utilise des bits quantiques (ou qubits). Cependant, si dans le système classique, un bit peut être dans deux états différents (0 ou 1), un qubit pourra avoir une infinité d'états, représentés par une matrice colonne de dimension 1×2 (vecteur d'un espace de dimension 2), et noté $|\psi\rangle$ (se lit *ket de* ψ).

Propriété. *On pose* $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ *et* $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. *Ainsi,* $\{|0\rangle, |1\rangle\}$ *est la base canonique de* $\mathbb{C}^2$. *Les coordonnées des vecteurs dans cette base sont appelées ici amplitudes.*

Une chaîne de n qubits a une norme de 1, autrement dit la somme des carrés des modules des amplitudes vaut 1.

Ex : Soit $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ un système d'un qubit. On a alors : $|\alpha|^2 + |\beta|^2 = 1$

Si on a un système de n qubits, les amplitudes sont les coordonnées dans la base canonique de $\mathbb{C}^{2^n}$. Les vecteurs de la base sont classés par ordre croissant, en binaire. Autrement dit, l'état d'un système de n qubits est égal au produit tensoriel des états individuels de chaque qubit.

Ex : Soit un système de trois qubits, dans des états individuels respectifs $|\psi_1\rangle = |0\rangle$, $|\psi_2\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ et $|\psi_3\rangle = i\,|1\rangle$. On a donc :

$$\begin{aligned}
|\psi_1,\psi_2,\psi_3\rangle &= |\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle = |0\rangle \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes i\,|1\rangle \\
&= \begin{pmatrix} 1 \times \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ 0 \times \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \end{pmatrix} \otimes |\psi_3\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \otimes |\psi_3\rangle \\
&= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \times \begin{pmatrix} 0 \\ i \end{pmatrix} \\ 1 \times \begin{pmatrix} 0 \\ i \end{pmatrix} \\ 0 \times \begin{pmatrix} 0 \\ i \end{pmatrix} \\ 0 \times \begin{pmatrix} 0 \\ i \end{pmatrix} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ i \\ 0 \\ i \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \frac{i}{\sqrt{2}} \left(|001\rangle + |011\rangle\right)
\end{aligned}$$

Ou, plus simplement :

$$\begin{aligned}
|\psi_1,\psi_2,\psi_3\rangle &= |\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle \\
&= |0\rangle \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes i\,|1\rangle \\
&= \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle) \otimes i\,|1\rangle \\
&= \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) \otimes i\,|1\rangle \\
&= \frac{i}{\sqrt{2}}(|00\rangle \otimes |1\rangle + |01\rangle \otimes |1\rangle) \\
&= \frac{i}{\sqrt{2}}(|001\rangle + |011\rangle)
\end{aligned}$$

3 Les portes quantiques : principe et exemples

3.1 Opérateur dague †

On définit l'opérateur † comme une fonction qui à une matrice $U \in \mathcal{M}(m,n,\mathbb{C})$; où $(m,n) \in \mathbb{N}^*$ sont les dimensions de la matrice ; associe la transposée de la matrice dont les éléments sont les conjugués des éléments de U.
Soit :

$$U = \begin{pmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mn} \end{pmatrix} \in \mathcal{M}(m,n,\mathbb{C})$$

$$U^\dagger := \begin{pmatrix} \overline{u_{11}} & \cdots & \overline{u_{m1}} \\ \vdots & \ddots & \vdots \\ \overline{u_{1n}} & \cdots & \overline{u_{mn}} \end{pmatrix} \in \mathcal{M}(n,m,\mathbb{C})$$

Propriété. *Soit U une matrice. On a :* $\left(U^\dagger\right)^\dagger = U$

Démonstration. Soit $U = \begin{pmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mn} \end{pmatrix} \in \mathcal{M}(m,n,\mathbb{C})$ une matrice.

$$\begin{aligned}
\left(U^\dagger\right)^\dagger &= \begin{pmatrix} \overline{u_{11}} & \cdots & \overline{u_{m1}} \\ \vdots & \ddots & \vdots \\ \overline{u_{1n}} & \cdots & \overline{u_{mn}} \end{pmatrix}^\dagger \\
&= \begin{pmatrix} \overline{\overline{u_{11}}} & \cdots & \overline{\overline{u_{1n}}} \\ \vdots & \ddots & \vdots \\ \overline{\overline{u_{m1}}} & \cdots & \overline{\overline{u_{mn}}} \end{pmatrix} \\
&= \begin{pmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mn} \end{pmatrix} \\
&= U
\end{aligned}$$

□

Propriété. *Soient U et V deux matrices telles que UV existe. On a :* $(UV)^\dagger = V^\dagger U^\dagger$

Démonstration. Soit $U = \begin{pmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mn} \end{pmatrix} \in M(m,n,\mathbb{C})$ et $V = \begin{pmatrix} v_{11} & \cdots & v_{1p} \\ \vdots & \ddots & \vdots \\ u_{n1} & \cdots & u_{np} \end{pmatrix} \in \mathcal{M}(n,p,\mathbb{C})$. On a :

$$\begin{aligned}
(UV)^\dagger &= \left(\begin{pmatrix} \overline{u_{11}} & \cdots & \overline{u_{1n}} \\ \vdots & \ddots & \vdots \\ \overline{u_{m1}} & \cdots & \overline{u_{mn}} \end{pmatrix} \begin{pmatrix} \overline{v_{11}} & \cdots & \overline{v_{1p}} \\ \vdots & \ddots & \vdots \\ \overline{v_{n1}} & \cdots & \overline{v_{np}} \end{pmatrix} \right)^t \\
&= \begin{pmatrix} \overline{v_{11}} & \cdots & \overline{v_{1p}} \\ \vdots & \ddots & \vdots \\ \overline{v_{n1}} & \cdots & \overline{v_{np}} \end{pmatrix}^t \begin{pmatrix} \overline{u_{11}} & \cdots & \overline{u_{1n}} \\ \vdots & \ddots & \vdots \\ \overline{u_{m1}} & \cdots & \overline{u_{mn}} \end{pmatrix}^t \\
&= V^\dagger U^\dagger
\end{aligned}$$

□

Rappel. *Soit V un vecteur de $\mathbb{C}^n$, $n \in \mathbb{N}^*$, autrement dit une matrice de $\mathcal{M}(n,1,\mathbb{C})$ (matrice colonne). La longueur, ou norme, de V est définie par :* $\|V\| = \sqrt{\sum_{i=1}^{n} (v_{i1})^2} = \sqrt{V.V}$

Propriété. *Soit V un vecteur de $\mathbb{C}^n$, $n \in \mathbb{N}^*$, autrement dit une matrice de $\mathcal{M}(n,1,\mathbb{C})$ (matrice colonne). On a :* $V^\dagger V = \|V\|^2 = V.V$

Démonstration. Soit $V \in \mathcal{M}(n,1,\mathbb{C})$. On a $V.V = \|V\|^2$. Or, comme V est à coefficients dans $\mathbb{C}$, alors : $V.V = \sum_{i=1}^{n} v_{i1}\overline{v_{i1}}$ car $\forall \omega \in \mathbb{C},\ \omega\overline{\omega} = |\omega|^2$.
Or $V^\dagger V = \overline{v_{11}}v_{11} + \overline{v_{21}}v_{21} + \ldots + \overline{v_{n1}}v_{n1}$. Par suite : $V^\dagger V = V.V = \|V\|^2$ □

Définition : Soit U une matrice. U est dite unitaire si et seulement si $U^\dagger U = I$, où $I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}$ est la matrice unité.

Corollaire. *Une matrice unitaire conserve les distances. Autrement dit, si on a U une matrice unitaire de n colonnes, et V un vecteur de $\mathbb{C}^n$, alors : $\|UV\| = \|V\|$*

Démonstration. Soient U une matrice unitaire de n colonnes et V un vecteur de $\mathbb{C}^n$. On a :

$$\begin{aligned}\|UV\| &= \sqrt{(UV)^\dagger (UV)} && \text{Car une norme est toujours positive}\\ &= \sqrt{V^\dagger U^\dagger UV}\\ &= \sqrt{V^\dagger I_n V}\\ &= \sqrt{V^\dagger V}\\ &= \|V\|\end{aligned}$$

□

Propriété. *Soient U et V deux matrices unitaires telles que UV existe. Alors UV est unitaire.*

Démonstration. Soit $U \in \mathcal{M}(m, n, \mathbb{C})$ et $V \in \mathcal{M}(n, p, \mathbb{C})$. On a :

$$\begin{aligned}(UV)^\dagger (UV) &= V^\dagger U^\dagger UV\\ &= V^\dagger I_n V\\ &= V^\dagger V\\ &= I_p\end{aligned}$$

Par définition, (UV) est unitaire. □

Définition : On définit $\langle\psi|$ (se lit *bra de* ψ) tel que : $\langle\psi| = |\psi\rangle^\dagger$

Définition : On définit $\langle\psi\,|\phi\rangle$ (se lit *bracket de* ψ *et* ϕ) : $\langle\psi\,|\phi\rangle = \langle\psi| \times |\phi\rangle$

Corollaire. *Avec les notations précédentes, on a : $\||\psi\rangle\|^2 = \langle\psi\,|\psi\rangle$.*

3.2 Les portes quantiques

Soit u un automorphisme de $\mathbb{C}^{2^n}$, avec $n \in \mathbb{N}$ le nombre de qubits sur lesquels on agit. On note $\mathcal{Q}$ la base canonique de $\mathbb{C}^{2^n}$, avec ses vecteurs notés sous la forme $|0\ldots0\rangle$, $|0\ldots01\rangle$,..., $|1\ldots1\rangle$. On note aussi $U = \mathcal{M}_{\mathcal{Q}}^{\mathcal{Q}}(u)$. On dit que u, ou U, est une porte quantique si et seulement si U est unitaire. La plupart du temps, les portes quantiques ont le même nom de fonction et de matrice, donc on utilisera de manière similaire $U\,|\psi\rangle$ et $U(|\psi\rangle)$, où $|\psi\rangle$ est un qubit.
Ainsi, à une entrée de n qubits doit correspondre une sortie de n qubits ; et les portes quantiques ont pour propriété de devoir être réversibles, contrairement à une porte classique.

Tout comme un algorithme classique est une succession de portes classiques, un algorithme quantique consiste en une succession de portes quantiques. La représentation schématique de cette succession se nomme circuit quantique.

Les portes quantiques étant des fonctions, l'état d'un système en sortie d'un algorithme quantique est l'image de l'état d'entrée par la composée de ces fonctions. Autrement dit, une succession de produits matriciels est nécessaire et suffisante pour connaître l'état final d'un système quantique.

3.3 Les circuits quantiques

Un circuit quantique est défini comme la représentation schématique de l'ensemble des portes appliquées aux différents qubits d'entrée. Il s'agit également d'un support visuel permettant de vérifier simplement si l'algorithme écrit correspond bien à celui que l'on souhaite exécuter.
Nous utiliserons Qiskit (téléchargeable sur *https : //qiskit.org/documentation*), une bibliothèque Python comportant différentes fonctions relatives aux algorithmes quantiques. Néanmoins Qiskit a également d'autres fonctions que la programmation quantique.

Résumé des différentes fonctions :

• La fonction `QuantumRegister(n, 'nom')` permet d'initialiser un vecteur qui sera par défaut : $|00\dots0\rangle$, avec `n` fois le nombre 0. On peut également nommer le vecteur mais cela n'est pas obligatoire.

• La fonction `ClassicalRegister(p, 'nom')` permet d'initialiser `p` bits classiques. `p` doit être inférieur ou égal à `n`. Ces bits seront utilisés lors de la mesure, lui servant de « support », et nous permettant ainsi de donner 0 ou 1 avec une probabilité dépendant des amplitudes.

• La fonction `QuantumCircuit(a, b,...)` permet d'initialiser le circuit quantique avec pour arguments les qubits, puis les bits classiques.

• La fonction `.measure(q, c)` permet d'effectuer la mesure d'un qubit `q`, mais aussi d'assigner le résultat à un bit classique `c`. Les arguments de cette fonction se comprennent comme : « le résultat de la mesure sur le qubit `q` est transféré sur le bit `c` ».

• La fonction `.draw(output='nom')` permet d'afficher le circuit quantique considéré sous un certain format (il en existe deux principaux, `'mpl'` ou `'latex'`).

• Pour implémenter une porte sur un qubit `q`, il faut entrer le code : `.'nom de la porte'(q)`. Par exemple, `.h(q)` permet d'entrer une porte de Hadamard, `.x(q)` une porte X, etc. (*cf.* ci-dessous pour la signification de ces portes)
Si une porte fonctionne sur plusieurs qubits, il faut les entrer dans l'ordre qubits contrôleurs, qubits cibles. La plupart des portes les plus utilisées sont ainsi implémentées.

De plus, les qubits sont stockés sous forme de listes. Nous pouvons donc demander d'effectuer l'opération sur `q[m]`, ce qui se traduit dans le circuit par l'application d'une porte sur le `m`-ième qubit.

Il est également possible de faire afficher un histogramme comportant les différentes probabilités d'obtenir chaque combinaison d'états.

De plus, lors de la création du circuit, il faut faire attention à l'ordre dans lequel les portes sont implémentées. En effet, elles apparaîtront, et seront appliquées, dans l'ordre dans lequel elles sont écrites dans le code. Par exemple, écrire `circuit.h(q)` puis `circuit.x(q)` appliquera en premier une porte de Hadamard sur le qubit `q` puis une porte X sur ce même qubit.

Il faut savoir que la mesure permet de déterminer l'état d'un qubit à un instant donné. Cependant, après mesure nous n'avons plus accès à l'état du qubit car celui-ci a changé, devenant $|0\rangle$ ou $|1\rangle$ suivant la mesure.

3.4 Les principales portes quantiques

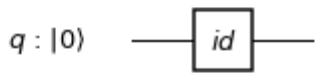
La condition d'unitarité d'une porte quantique étant relativement simple à remplir, il en existe une infinité. Cependant, certaines sont très employées, telles que celles qui suivent.

Porte Identité ou porte fait-rien (Id)

Cette porte correspond à la fonction Identité, et conserve donc l'état du qubit entré.

$$Id = I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad\qquad Id(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + \beta|1\rangle$$

Représentation :

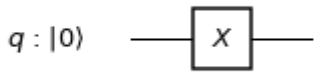


Porte Pauli-X (X)

Cette porte agit à la façon de la porte classique NOT ; c'est une rotation de π selon l'axe $(Ox) = (O, |0\rangle)$.

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad X(\alpha |0\rangle + \beta |1\rangle) = \beta |0\rangle + \alpha |1\rangle$$

Représentation :

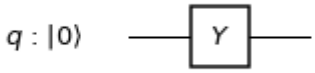


Porte Pauli-Y (Y)

Cette porte consiste en une rotation de π selon l'axe $(Oy) = (O, |1\rangle)$.

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Y(\alpha |0\rangle + \beta |1\rangle) = -i\beta |0\rangle + i\alpha |1\rangle$$

Représentation :

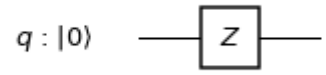


Porte Pauli-Z (Z)

Cette porte consiste en une rotation de π selon l'axe $(Oz) = (O, |0\rangle \wedge |1\rangle)$

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad Z(\alpha |0\rangle + \beta |1\rangle) = \alpha |0\rangle - \beta |1\rangle$$

Représentation :

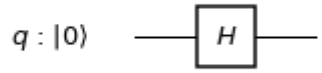


Porte de Hadamard (H)

Cette porte est surtout appliquée sur $|0\rangle$, permettant ainsi d'obtenir $|0\rangle$ ou $|1\rangle$ avec équiprobabilité, et donc de « tester tous les cas en une fois ».

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad H(\alpha |0\rangle + \beta |1\rangle) = \frac{\alpha + \beta}{\sqrt{2}} |0\rangle + \frac{\alpha - \beta}{\sqrt{2}} |1\rangle$$

Représentation :

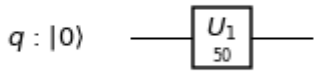


Porte de changement de phase de facteur ϕ (R_ϕ)

Cette porte modifie la phase (amplitude de $|1\rangle$) par une rotation d'angle ϕ. $e^{i\phi}$ est appelé facteur de phase.

$$R_\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix} \qquad R_\phi(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + \beta e^{i\phi}|1\rangle$$

Représentation :

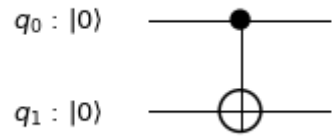


Porte Pauli-X contrôlée (cX ou CNOT)

Cette porte agit comme une porte X sur le deuxième qubit (qubit cible) si le premier qubit (qubit contrôleur) est « dans un état $|1\rangle$ ».

$$cX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad cX(\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle) = \alpha|00\rangle + \beta|01\rangle + \delta|10\rangle + \gamma|11\rangle$$

Représentation :

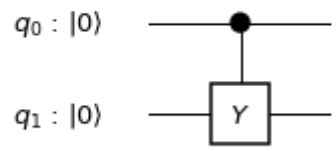


Porte Pauli-Y contrôlée (cY)

Cette porte agit comme une porte Y sur le deuxième qubit (qubit cible) si le premier qubit (qubit contrôleur) est « dans un état $|1\rangle$ ».

$$cY = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix} \qquad cY(\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle) = \alpha|00\rangle + \beta|01\rangle - i\delta|10\rangle + i\gamma|11\rangle$$

Représentation :

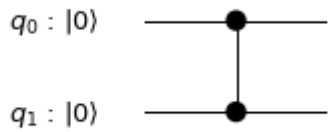


Porte Pauli-Z contrôlée (cZ)

Cette porte agit comme une porte Z sur le deuxième qubit (qubit cible) si le premier qubit (qubit contrôleur) est « dans un état $|1\rangle$ ».

$$cZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \qquad cZ(\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle) = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle - \delta|11\rangle$$

Représentation :

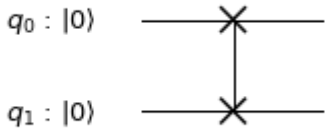


Porte Swap (S)

Cette porte intervertit les deux qubits.

$$S = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad S(\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle) = \alpha|00\rangle + \gamma|01\rangle + \beta|10\rangle + \delta|11\rangle$$

Représentation :



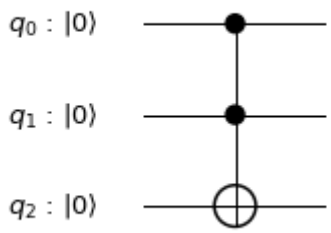
Porte de Toffoli ou porte Pauli-X contrôlée contrôlée (ccX ou CCNOT)

Cette porte correspond au AND et au NAND quantique : elle agit comme une porte X sur le qubit cible lorsque les deux qubits contrôleurs sont « dans un état $|1\rangle$ ». Si le qubit cible était « dans un état $|0\rangle$ », il deviendrait alors « dans un état $|1\rangle$ » (porte AND quantique) et inversement (porte NAND quantique)

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

$$ccX(\alpha|000\rangle + \beta|001\rangle + \gamma|010\rangle + \delta|011\rangle + \epsilon|100\rangle + \zeta|101\rangle + \eta|110\rangle + \theta|111\rangle)$$
$$= \alpha|000\rangle + \beta|001\rangle + \gamma|010\rangle + \delta|011\rangle + \epsilon|100\rangle + \zeta|101\rangle + \theta|110\rangle + \eta|111\rangle$$

Représentation :



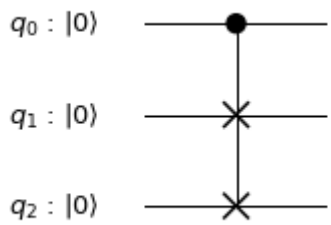
Porte de Fredkin ou porte Swap contrôlée (cS ou CSWAP)

Cette porte agit comme une porte Swap sur les deux qubits cibles si le qubit contrôleur est « dans un état $|1\rangle$ ».

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

$$\begin{aligned} &cS(\alpha\,|000\rangle + \beta\,|001\rangle + \gamma\,|010\rangle + \delta\,|011\rangle + \epsilon\,|100\rangle + \zeta\,|101\rangle + \eta\,|110\rangle + \theta\,|111\rangle) \\ &= \alpha\,|000\rangle + \beta\,|001\rangle + \gamma\,|010\rangle + \delta\,|011\rangle + \epsilon\,|100\rangle + \eta\,|101\rangle + \zeta\,|110\rangle + \theta\,|111\rangle \end{aligned}$$

Représentation :



3.5 Appliquer des portes quantiques sur plusieurs systèmes de qubits en parallèle

Nous savons appliquer une porte quelconque à un unique ensemble de qubits. Cependant, est-il possible d'appliquer des portes quantiques à plusieurs ensembles de qubits simultanément ?
En utilisant Qiskit c'est très simple, il suffit d'écrire `circuit.'nom de la porte'(qubit)` et la porte voulue sera appliquée sur chaque qubit ; ou plusieurs portes peuvent êtres programmées et seront executées en parallèle si elles n'agissent pas sur un qubit commun.
Mathématiquement, soit U une porte quantique agissant sur n qubits, V une porte agissant sur p qubits (avec $p \geq n$), et soit un système d'au moins $n+p$ qubits. Appliquer U sur n qubits et V sur p autres qubits revient aussi à appliquer une ou plusieurs portes fait-rien sur les autres qubits. Pour calculer la matrice Q équivalente à ce système, nous effectuons le produit tensoriel des portes appliquées, dans l'ordre des qubits (car le produit tensoriel n'est pas commutatif).
De la même façon qu'une répétition de n produits est abrégée par la notation puissance ^{n}, une répétition de n produits tensoriels est abrégée par la notation puissance tensoriel $^{\otimes n}$.

Ex : Soit un système de 2 qubits. Appliquons à chacun une porte de Hadamard H. La porte équivalente est :

$$\begin{aligned} H^{\otimes 2} &= H \otimes H \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} H & H \\ H & -H \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \end{aligned}$$

Ex 2 : Soit un système de 2 qubits. Appliquons au deuxième une porte de Hadamard H. La porte H_2 est :

$$\begin{aligned} H_2 &= Id \otimes H \\ &= \begin{pmatrix} H & 0H \\ 0H & H \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \end{aligned}$$

4 Mesure

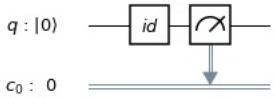
4.1 Définition

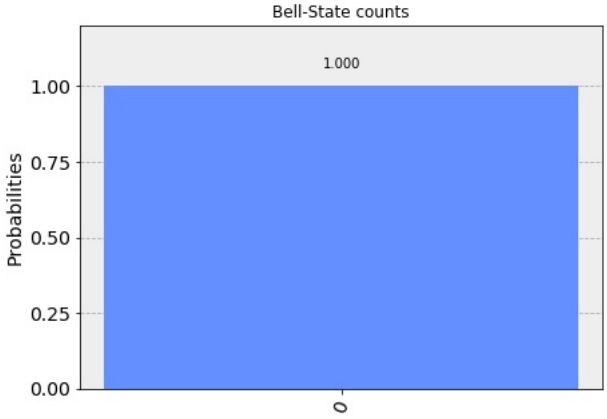
On peut se demander quelles informations il est possible d'obtenir d'un qubit.
Pour pouvoir extraire des données, on utilise la mesure. Tout d'abord expliquons comment cela fonctionne pour un seul qubit dans l'état $\alpha |0\rangle + \beta |1\rangle$. L'aspect fondamental de la mesure est que les informations obtenues seront des probabilités concernant des bits classique. Ici, il s'agira donc soit du bit 0 soit du bit 1. Or, comme le qubit se trouve dans une superposition d'états lors de la mesure, les bits classiques donnés en sortie auront une certaine probabilité d'être obtenus. Dans notre cas, la probabilité d'obtenir le bit 0 sera $|\alpha|^2$ et la probabilité d'obtenir le bit 1 sera $|\beta|^2$. Notons que ces probabilités sont bien des nombres positifs de somme 1, par définition des amplitudes.
La généralisation à plusieurs qubits n'est pas compliquée. Il s'agit du même principe, chaque chaîne de bits possible ayant une probabilité d'être obtenue égale au carré du module de son amplitude.
Lors de l'étude d'un système quantique représenté par des qubits, nous n'aurons jamais accès directement aux amplitudes des différents qubits. En effet, lorsqu'on effectue une mesure, on va interagir avec les qubits et donc tout en obtenant les informations en bits classiques, on va également modifier les différentes amplitudes de la superposition d'états, et « l'état postérieur » du qubit est le vecteur correspondant à la chaîne mesurée, avec une amplitude de 1. Individuellement, chaque qubit est dans l'état $|0\rangle$ s'il a été mesuré 0, et $|1\rangle$ sinon. Ce changement d'état pour les états de base est ce que l'on appelle la *réduction du paquet d'onde.*

4.2 Exemples de circuit quantiques et de mesures

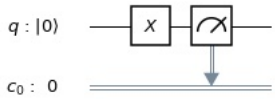
Dans les exemples qui suivent, les histogrammes montreront les chaînes de bits obtenus après 1024 lancements du circuit.

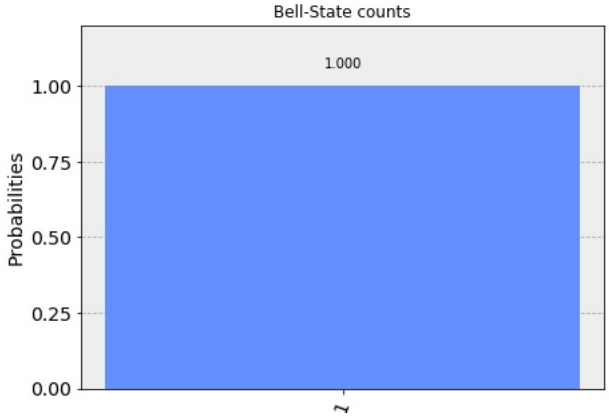
Circuit « vide », composé d'une porte fait-rien :
Après la porte



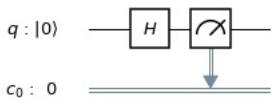


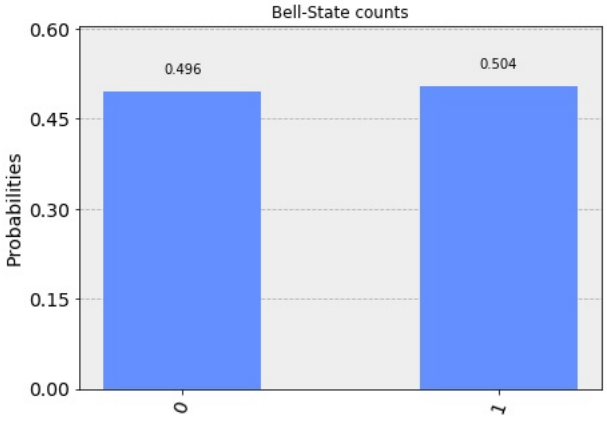
Circuit composé d'une porte Pauli-X :



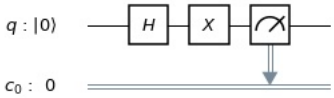


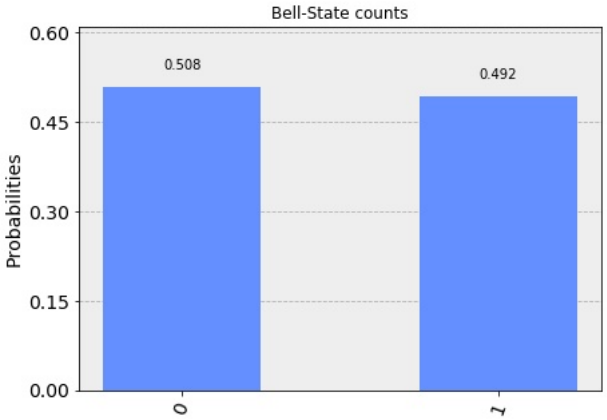
Circuit composé d'une porte d'Hadamard :
Après la porte H, le qubit est dans un état $H|0\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$. La probabilité d'avoir une mesure donnant 0 est donc : $\left|\frac{1}{\sqrt{2}}\right|^2 = 0,5$; et de même pour celle d'obtenir 1.



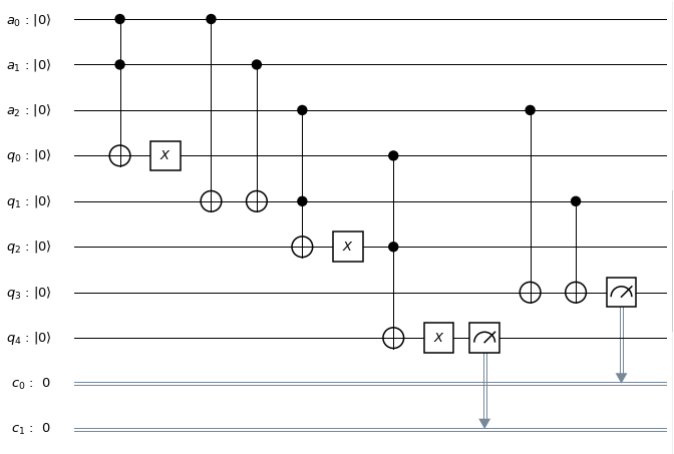


Circuit composé d'une porte d'Hadamard, suivie d'une porte Pauli-X :





Il est aussi à noter que tout circuit classique peut être adapté en circuit quantique. Par exemple, en reprenant l'Additionneur présenté plus tôt :



5 L'algorithme de Deutsch-Jozsa

5.1 Principe

L'algorithme de Deutsch-Jozsa a été introduit pour la première fois par David Deutsch dans *The Church-Turing principle and the universal quantum computer* en 1985 ; puis généralisé par David Deutsch et Richard Jozsa dans *Rapid solutions of problems by quantum computation* en 1992. Il s'agit de l'un des premiers algorithmes quantiques dont l'exécution est visiblement plus rapide que son équivalent classique.

Le but de cet algorithme est de déterminer la classe d'une fonction parmi deux possibles.

Plus précisément, nous avons une fonction inconnue $f : \{0,1\}^n \longrightarrow \{0,1\}$, qui est soit *équilibrée* soit *constante*. Cette fonction est dite constante si elle a toujours pour image 0 ou 1 ; et équilibrée si elle a pour image autant de 0 que de 1.
L'ensemble de départ contient 2^n éléments. Ainsi, pour qu'un algorithme classique détermine la classe de f, il lui faudra au maximum $2^{n-1}+1$ étapes : s'il teste les 2^{n-1} premiers cas, soit la moitié des cas possibles, et que l'image est constante, il ne pourra pas conclure si la fonction est constante ou s'il a testé tous les cas donnant le même résultat. Il faut donc tester un cas supplémentaire pour pouvoir trancher. En classique, cet algorithme a donc une complexité exponentielle $O(2^n)$.
Cependant, l'algorithme de Deutsh, son équivalent quantique, est capable de trancher en 1 étape, et a donc une complexité constante $O(1)$.
Ceci est principalement possible grâce à l'utilisation en début d'algorithme d'une porte de Hadamard sur chaque qubit à l'état $|0\rangle$, permettant ainsi de créer un système $|\psi\rangle$ étant une combinaison linéaire de chaque possibilité, et donc de « tester toutes les possibilités en une fois ».

5.2 Circuit quantique

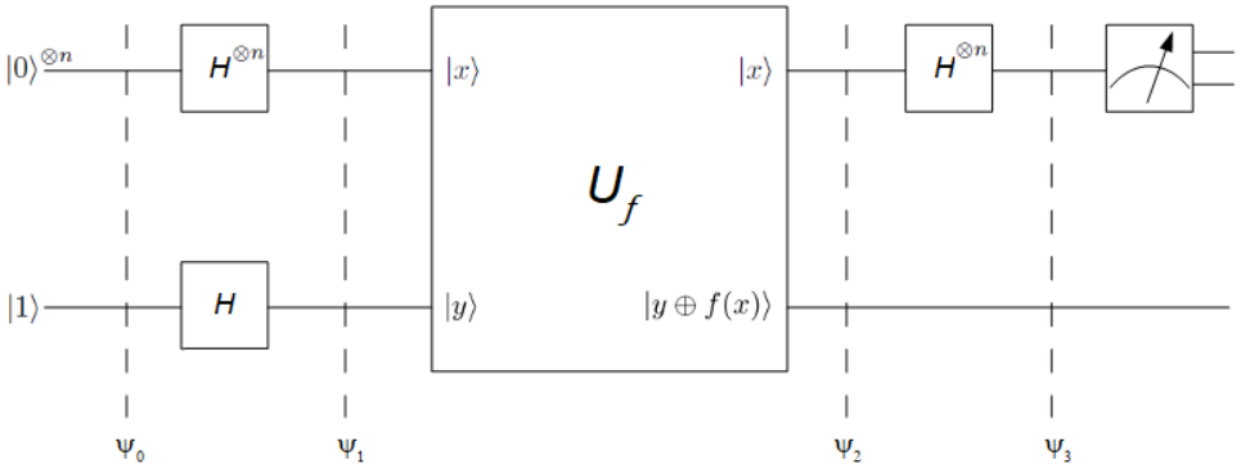


FIGURE 3 – Circuit de l'algorithme de Deutsch-Jozsa

On présentera dans la suite l'évolution du système quantique au cours du temps, lors de l'exécution de cet algorithme.

5.3 État initial du système : en ψ_0

$$|\psi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle$$

5.4 Après la première porte : en ψ_1

$$\begin{aligned}
|\psi_1\rangle &= (H|0\rangle)^{\otimes n} \otimes (H|1\rangle) \\
&= \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)^{\otimes n} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} \\
&= \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} |x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} \\
&= \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} \left(|x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) && \text{en développant}
\end{aligned}$$

5.5 Après la boîte noire : en ψ_2

$$|\psi_2\rangle = \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} \left(|x\rangle \otimes \frac{|0\oplus f(x)\rangle - |1\oplus f(x)\rangle}{\sqrt{2}}\right)$$

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

D'où :

$$\begin{aligned}
|\psi_2\rangle &= \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} \left(|x\rangle \otimes (-1)^{f(x)} \frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\
&= \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} \left((-1)^{f(x)}|x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\
&= \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} && \text{en factorisant}
\end{aligned}$$

5.6 État final : en ψ_3

$$|\psi_3\rangle = \frac{1}{\sqrt{2}^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left(H^{\otimes n}|x\rangle\right) \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Or, soit un qubit $|a\rangle$:

$$H|a\rangle = \frac{1}{\sqrt{2}} \sum_{b\in\{0,1\}} (-1)^{ab}|b\rangle \qquad \text{et } |x\rangle = \bigotimes_{i=1}^{n} |x_i\rangle \quad \text{avec les } x_i \text{ des qubits.}$$

D'où :

$$\begin{aligned}
H^{\otimes n}|x\rangle &= \bigotimes_{i=1}^{n} H|x_i\rangle \\
&= \bigotimes_{i=1}^{n} \frac{1}{\sqrt{2}} \sum_{z_i\in\{0,1\}} (-1)^{x_i z_i}|z_i\rangle \\
&= \frac{1}{\sqrt{2}^n} \sum_{\substack{z_1\in\{0,1\}\\ \vdots \\ z_n\in\{0,1\}}} (-1)^{x_1 z_1}|z_1\rangle \otimes (-1)^{x_2 z_2}|z_2\rangle \otimes \ldots \otimes (-1)^{x_n z_n}|z_n\rangle \\
&= \frac{1}{\sqrt{2}^n} \sum_{\substack{z_1\in\{0,1\}\\ \vdots \\ z_n\in\{0,1\}}} (-1)^{x_1 z_1 + x_2 z_2 + \ldots + x_n z_n}|z_1\rangle \otimes |z_2\rangle \otimes \ldots \otimes |z_n\rangle \\
&= \frac{1}{\sqrt{2}^n} \sum_{z\in\{0,1\}^n} (-1)^{x.z}|z\rangle
\end{aligned}$$

Soit :

$$\begin{aligned}
|\psi_3\rangle &= \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \sum_{z\in\{0,1\}^n} (-1)^{x.z}|z\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} \\
&= \frac{1}{2^n} \sum_{(x,z)\in(\{0,1\}^n)^2} (-1)^{f(x)+x.z}|z\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}
\end{aligned}$$

5.7 Mesure des n premiers qubits

Supposons f constante. Alors la probabilité que la mesure donne n 0 est :

$$\begin{aligned}
P\left(|z\rangle = |0\rangle^{\otimes n}\right) &= \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)+x.|0\rangle^{\otimes n}} \right|^2 \qquad \text{or } x.|0\rangle^{\otimes n} = 0 \\
&= \begin{cases} \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} 1 \right|^2 & \text{si f=0} \\ \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} -1 \right|^2 & \text{si f=1} \end{cases} \\
&= \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} 1 \right|^2
\end{aligned}$$

Or $\{0,1\}^n$ contient 2^n éléments, donc :

$$P\left(|z\rangle = |0\rangle^{\otimes n}\right) = \left|\frac{2^n}{2^n}\right|^2 = 1$$

Supposons maintenant f équilibrée.

$$P\left(|z\rangle = |0\rangle^{\otimes n}\right) = \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)+x.|0\rangle^{\otimes n}} \right|^2 \qquad \text{or } x.|0\rangle^{\otimes n} = 0$$

Or, comme f équilibrée, il y a autant de x tels que $f(x) = 0$ que de x tels que $f(x) = 1$. D'où :

$$P\left(|z\rangle = |0\rangle^{\otimes n}\right) = \left|\frac{0}{2^n}\right|^2 = 0$$

Ainsi, la mesure donne une chaîne de 0 si et seulement si f est constante, et on peut donc déduire la classe de f.

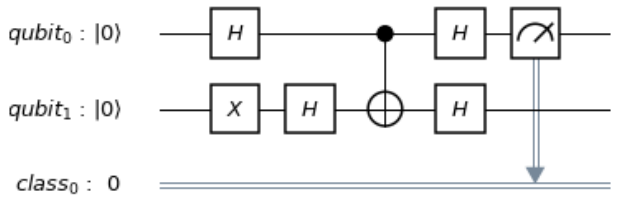
5.8 Programmation de l'algorithme de Deutsh-Jozsa pour $n = 1$ (algorithme de Deutsch) avec Qiskit

```
import numpy as np
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister
from qiskit import Aer, execute
from qiskit.tools.visualization import plot_histogram
from qiskit.providers.aer import QasmSimulator
import math as m
import scipy as sci


#les fonctions début

def boîte_noire(circuit, qubit_cible) :
    r = int(m.floor(4*sci.rand()))
    type_fonction = ['f{0,1} = {0,1}', 'f{0,1} = {1,0}', 'f{0,1} = {0,0}', 'f{0,1} = {1,1}']
    if r == 3 :
        circuit.x(qubit_cible[1])
    elif r == 2 :
        circuit.iden(qubit_cible)
    elif r == 0 :
        circuit.cx(qubit_cible[0], qubit_cible[1])
    elif r == 1 :
        circuit.x(qubit_cible[0])
        circuit.cx(qubit_cible[0], qubit_cible[1])
        circuit.x(qubit_cible[0])
    return type_fonction[r]

def algorithme_Deutsch(circuit, qubit_cible, classique) :
    circuit.x(qubit_cible[1])
    circuit.h(qubit_cible)
    fonction = boîte_noire(circuit, qubit_cible)
    circuit.h(qubit_cible)
    circuit.measure(qubit_cible[0], classique[0])
    print(fonction)
    return circuit

#les fonctions fin

#traitement début

qubit = QuantumRegister(2, 'qubit')
classique = ClassicalRegister(1, 'class')
circuit = QuantumCircuit(qubit, classique)
circuit = algorithme_Deutsch(circuit, qubit, classique)

#traitement fin

#mesure début

simulator = Aer.get_backend('qasm_simulator')
result = execute(circuit, simulator, shots=1).result()
counts = result.get_counts(circuit)
plot_histogram(counts, figsize=(7,7), color='red', title='mesure')

#mesure fin
```

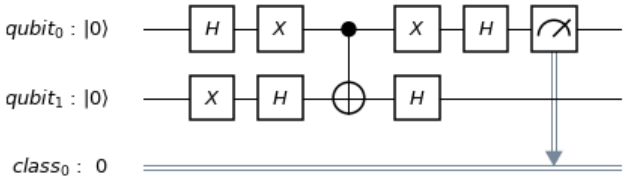
```
#conclusion début

if '0' in counts:
    print('f constante')
else:
    print('f équilibrée')

#conclusion fin
```

La fonction `boîte_noire` permet de créer aléatoirement une fonction $f : \{0,1\} \longrightarrow \{0,1\}$, qui est soit équilibrée, soit constante.

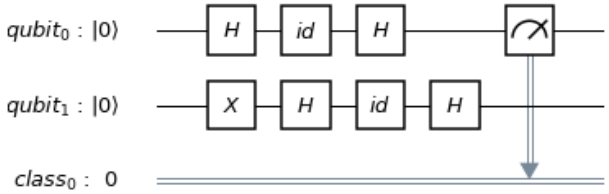
Circuit suivant la fonction, et donc la boîte noire implémentée :
$f(0) = 0$ et $f(1) = 1$:



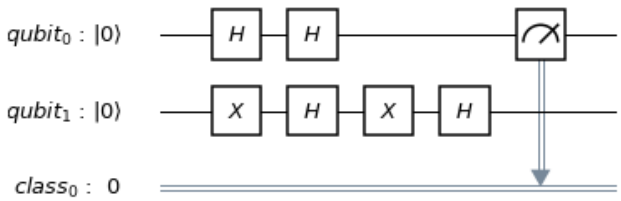
$f(0) = 1$ et $f(1) = 0$:



$f(0) = f(1) = 0$:



$f(0) = f(1) = 1$:



6 Bibliographie

• Wikipedia. *Additionneur* [en ligne] (page consultée le 14/05/2019). https://en.wikipedia.org/wiki/Adder_(electronics)#/media/File:Full-adder_logic_diagram.svg

• Andy Matuschak et Michael Nielsen. *Quantum computing for the very curious* [en ligne] (page consultée le 14/05/2019). https://quantum.country/qcvc/

• Daniel Koch. *Introduction to Coding Quantum Algorithms : A Tutorial Series Using Qiskit* [en ligne] (page consultée le 14/05/2019). https://arxiv.org/abs/1903.04359v1

• David Deutsch, *The Church-Turing principle and the universal quantum computer*, Proceedings of the Royal Society of London A, vol. 400, 1985, p. 97 [en ligne](page consultée le 15/05/2019). http://coblitz.codeen.org:3125/citeseer.ist.psu.edu/cache/papers/cs/13701/http:zSzzSzwww.qubit.orgzSzresourcezSzdeutsch85.pdf/deutsch85quantum.pdf

• David Deutsch et Richard Jozsa, *Rapid solutions of problems by quantum computation*, Proceedings of the Royal Society of London A, vol. 439, 1992, p. 553 [en ligne](page consulée le 15/05/2019).https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=2ahUKEwiroeL2Op3iAhWL2BQKHbE1ANIQFjABegQIBRAC&url=https%3A%2F%2Fqudev.phys.ethz.ch%2Fphys4%2Fstudentspresentations%2Fdjalgo%2FDeutschJozsa.pdf&usg=AOvVaw0a-fG1WV7b1wx0qW1Ck6aJ

•*Les différentes portes logiques* [en ligne] (page consultée le 15/05/2019). https://whatis.techtarget.com/definition/logic-gate-AND-OR-XOR-NOT-NAND-NOR-and-XNOR