DorEmy

Début d'ORdinateur pour EMY-22

Groupe EMY

23 avril 2022

Dans le cadre du développment du protocole EMY-22 (protocole de transmission quantique de clés cryptographiques), ainsi qu'en vue de l'implémentation de l'algorithme de Grover sur un ordinateur classique, ce projet a pour objectif de constituer une base logicielle aisée d'emploi et permettant de simuler, avec un réalisme (c'est-à-dire un abus) et un temps de calcul raisonnable les bases du calcul quantique.

1 Unités de mémoire – Qubits

Un bit est une unité de base de la mémoire qui peut prendre uniquement deux états : 0 et 1. Un qubit, ou bit quantique diffère d'un bit car il son état est une combinaison linéaire des deux états de base (ou *vecteurs propres*) du qubit, $|0\rangle$ et $|1\rangle$. Cette combinaison linéaire doit néanmoins respecter la condition de normalisation, si bien que l'ensemble des états possibles d'un qubit est :

$$E_Q = \{\alpha |0\rangle + \beta |1\rangle \mid \alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1\}$$

Un cas plus général est celui des qudits, qui sont une généralisation des qubits avec d vecteurs propres. Par exemple avec $d = 3$ on aurait les vecteurs propres $|0\rangle$, $|1\rangle$ et $|2\rangle$. L'ensemble des états d'un qudit est :

$$E_d = \left\{ \sum_{i=0}^{d-1} \alpha_i |i\rangle \,\middle|\, \forall i \in [\![0, d-1]\!], \alpha_i \in \mathbb{C}, \sum_{i=0}^{d-1} |\alpha_i|^2 = 1 \right\}$$

L'état est représenté sous la forme d'une matrice colonne, ce qui facilite l'applications de portes quantiques. Par exemple l'état $|\psi\rangle = a|0\rangle + b|1\rangle$, avec $a, b \in \mathbb{C}$ est assimilé à la matrice $\begin{pmatrix} a \\ b \end{pmatrix}$.

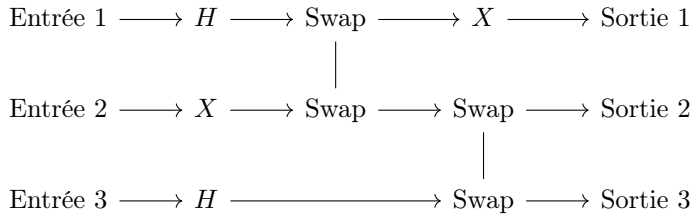
2 Calcul informatique quantique

Le calcul quantique à proprement parlé sera implémenté dans le module `qubit` ainsi que le module `portes` et éventuellement le module `circuit`.

De manière analogue aux ordinateurs classiques, les ordinateurs quantiques sont basés sur des circuits électroniques. Ces circuits sont entre autres composés de portes logiques, qui agissent sur un petit nombre d'unités de mémoire (comme des qubits) en modifiant leur état. Un circuit est alors définissable comme : une entrée (avec un état initial de la mémoire connu), et un graphe représentant les différentes portes logiques opérant sur la mémoire, dont fait partie la sortie qui présente l'état de la mémoire qui est le résultat des calculs.

On implémentera donc la structure de circuit sous forme de graphe orienté dont les sommets sont des portes logiques. La structure étant de manière générale bien plus linéaire qu'un graphe quelconque, on choisira plutôt de stocker les graphes sous forme de matrice, mais pas sous forme de matrice d'adjacence, sous une autre forme.

Exemple : pour le circuit ci-dessous (où H est la porte de Hadamard et X la porte `NOT` ou encore porte de Pauli-X).



On représentera ce circuit par la matrice suivante (où I est la porte identité) :

$$\begin{pmatrix} H & \text{Swap}_1 & X \\ X & \text{Swap}_1 & \text{Swap}_2 \\ H & I & \text{Swap}_2 \end{pmatrix}$$

Les lignes représentent les *étapes* (on commence à l'étape 0) et les colonnes représentent les *qubits* (on commence au qubit 0). Dans l'exemple précédent, la porte identité est à l'étape 1 du qubit 2.

Cette représentation est moins standard que celle des matrices d'adjacence, mais elle a de nombreux avantages :

- Facilité de la mise en série de plusieurs circuits (il suffit de concaténer les matrices)
- Lecture et écriture par un être humain relativement aisée
- Parcours d'un circuit par un qubit facile à implémenter

3 Calcul mathématique

Dans de nombreuses situations on aura besoin de réaliser des calculs de manière plus poussée que ce que permet nativement Python. Pour cela on implémentera un module, nommé `calcul`, dont le rôle sera de réaliser des calculs de manière formelle.

3.1 Ensembles de travail

Dans la construction des mathématiques par laquelle ce module passera, on admettra la construction des entiers relatifs $\mathbb{Z}$ (néanmoins on détaillera $\mathbb{N}$), et on implémentera quelques ensembles de valeurs :

— Les rationnels $\mathbb{Q}$, qu'on assimilera l'ensemble des couples $(p, q) \in \mathbb{Z} \times \mathbb{N}^*$ avec p et q premiers entre eux. On prendra garde à toujours simplifier les rationnels lors des calculs de manière à garder un couple d'entiers avec les bons signes et premiers entre eux.

— Les puissances rationnelles de rationnels positifs $\mathbb{P}$. En particulier, $\mathbb{P}$ contient l'ensemble des racines carrées des rationnels, ainsi que $\mathbb{Q}$ lui-même. La multiplication est interne dans $\mathbb{P}$ mais pas l'addition

On aura donc des réels que $\mathbb{P} = \{\sigma x^p \mid \sigma \in \{-1, 1\}, x \in \mathbb{Q}^+, p \in \mathbb{Q}\}$. Les complexes seront construits comme des couples : $x+iy$ sera assimilé à $(x, y) \in \mathbb{P}^2$. Le plus grand ensemble de travail est donc $\mathbb{P}[i] = \{x + iy \mid x, y \in \mathbb{P}\}$.

3.2 Sous-ensemblage

3.2.1 Principe

Une dynamique qu'il sera crucial d'implémenter est le **sous-ensemblage** : toute variable doit avoir à chaque fin de calcul (même partiel) un type (correspodant à un ensemble) le plus fin possible. Par exemple le calcul $\frac{2}{3} \times \frac{3}{2} = \frac{1}{1}$ doit automatiquement être converti en entier (un sous-ensemble des rationnels, ce qu'il est par ailleurs). On note en particulier les inclusions : $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{P}$.

Le sous-ensemblage est implémenté dans la fonction `sous`, qui appelle, sauf si aucune simplification n'est possible, la fonction `sous` du sous-type trouvé. Exemple pour le rationnel $\frac{2}{1} \in \mathbb{Q}$:

1. La fonction `sous` des rationnels détecte qu'on peut passer aux entiers relatifs. Elle construit donc un entier relatif correspondant : c'est $2 \in \mathbb{Z}$. On apelle ensuite `sous` sur le relatif 2.
2. Dans `sous` de 2, on trouve qu'on peut simplifier au naturel $2 \in \mathbb{N}$
3. On ne peut pas simplifier plus, donc le `sous` de $2 \in \mathbb{N}$ renvoie le même $2 \in \mathbb{N}$.

3.2.2 Simplification des puissances

L'ensemble $\mathbb{P}$ étant le plus grand sous-ensemble de $\mathbb{R}$ implémenté, il est important de mettre au point des techniques de simplification des puissances.

Soit $a = \sigma x^p \in \mathbb{P}$ une puissance. On a $\sigma \in \{-1, 1\}$ le *signe*, $x = \frac{n_x}{d_x} \in \mathbb{Q}^+$ le *signifiant* et $p = \frac{n_p}{d_p} \in \mathbb{Q}^{+*}$ l'*exposant*.

1. Si $p \in \mathbb{Z}$, on calcule $\sigma x^p \in \mathbb{Q}$. Si $p \geq 0$ son numérateur non simplifié est $\sigma {n_x}^p$ et son dénominateur non simplifié est ${d_x}^p$. Si $p < 0$ son numérateur non simplifié est $\sigma {d_x}^p$ et son dénominateur non simplifié est ${n_x}^p$.
2. Sinon, on cherche à savoir si il existe $r = \frac{n_r}{d_r} \in \mathbb{Q}+$ tel que $r^{d_p} = x$. On a alors $x = r^{n_p} \in \mathbb{Q}$.

Algorithme : racine d_p-ième exacte d'un rationnel.
On cherche à déterminer s'il existe un $r \in \mathbb{Q}^+$ tel que $r^{d_p} = x$, et si il existe à déterminer un couple numérateur-dénominateur.

```
Pour d_r allant de 1 à d_x
   Si η = x · d_r^{d_p} ∈ ℕ
      Si ∃n_r ∈ ⟦1, ⌊√η⌋⟧ tel que n_r^{d_p} = η
         On renvoie r = n_r / d_r
      Fin si
   Fin si
Fin pour
Il n'existe pas de r qui convient
```

3.3 Sur-ensemblage

La dynamique inverse existe aussi : on la nomme **sur-ensemblage**, mis en place dans la fonction `sur`. Elle est néanmoins temporaire puisqu'elle n'est utilisée que pour permettre les calculs entre deux types différents. Exemple pour le calcul $-\frac{1}{2} \times (-2) = 1$:

1. $-2 \notin \mathbb{Q}\backslash\mathbb{Z}$, on le projette sur l'ensemble $\mathbb{Q}$. On obtient $-\frac{2}{1} \in \mathbb{Q}$.
2. $-\frac{1}{2}$ et $-\frac{2}{1}$ sont de même type, on peut les muliplier. On obtient $\frac{1}{1} \in \mathbb{Q}$
3. Fin du calcul : on simplifie en appelant `sous` et on obtient finalement $1 \in \mathbb{N}$.

3.4 Matrices

Le nombre de lignes d'une matrice `m` est `m.p` et son nombre de colonnes est `m.q`, avec $p, q \in \mathbb{N}$. On peut accéder à p et q en même temps en regardant `m.forme`, qui vaut le tuple `(p, q)`. Les élements sont accessibles par un couple $(i, j) \in [\![0, p-1]\!] \times [\![0, q-1]\!]$, les indices commençant donc à 0. On aura par exemple `m[0, 0]` Pour les matrices lignes et colonnes, il n'est nécessaire que de spécifier l'une des deux coordonnées (l'autre étant forcément 0). On aura dans ce cas `m[2]` par exemple.

Les éléments d'une matrice ne sont jamais des `int`, des `float` et encore moins des `str` : tous les coefficients sont des nombres formellements stockés (voir 3.1), donc des éléments de $\mathbb{P}[i]$.