

Physique Quantique et Cryptographie
Partie 4

Attaques Quantiques en Cryptographie

Jacques Patarin

Cryptanalyse Quantique



Classes de Complexité, Définitions

Problème de décision : problème dont la réponse est « oui » ou « non ».

En classique (= non quantique)

P : ensemble des problèmes de décision pour lesquels il existe un algorithme (non quantique) déterministe polynomial en pire cas.

BPP : ensemble des problèmes de décision pour lesquels il existe un algorithme (non quantique) probabiliste polynomial (erreur max 1/3 pour toute itération).

NP : ensemble des problèmes de décision qui admettent un certificat lorsque la réponse est 'oui' (i.e. lorsque la réponse est 'oui' il existe une petite valeur qui permet de vérifier en temps polynomial déterministe en pire cas qu'effectivement la réponse est oui).

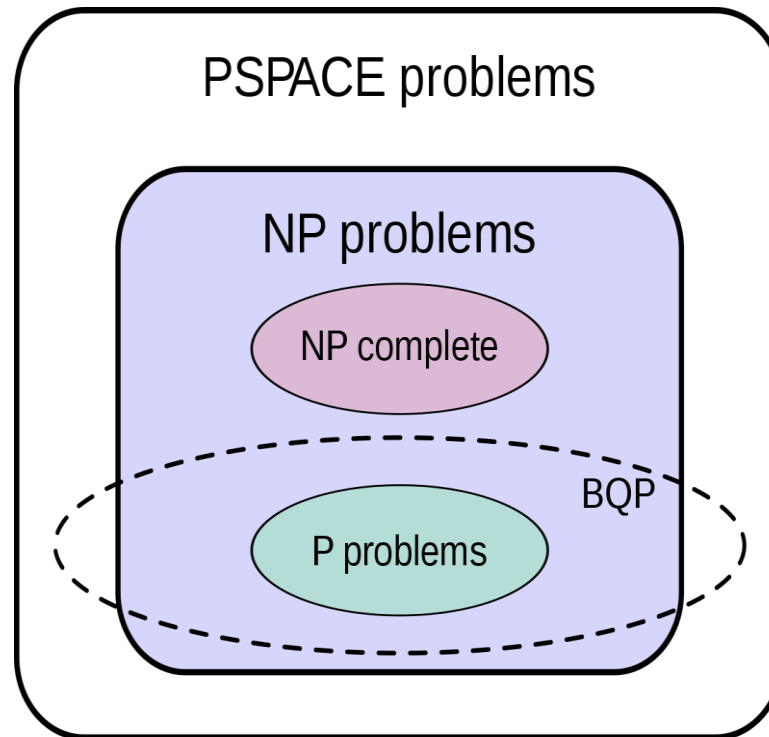
NP complet : ensemble des problèmes de la classe NP qui ont la propriété qu'à partir d'un nombre polynomial d'appel à eux on peut résoudre en temps polynomial déterministe en pire cas tout problème de la classe NP. (Ce sont donc les problèmes les plus « flexibles » ou « généraux » de NP, plutôt que les plus « difficiles »).

En Quantique

QP (parfois noté **EQP**) : ensemble des problèmes de décision pour lesquels il existe un algorithme quantique déterministe polynomial en pire cas.

BQP : ensemble des problèmes de décision pour lesquels il existe un algorithme quantique probabiliste polynomial (erreur max 1/3 pour toute itération).

Les problèmes NP complets sont supposés être
en dehors de PQP



1. L'algorithme de Deutsch-Jozsa (1985, 1992)

L'algorithme de Deutsch-Jozsa est moins important en pratique que les algorithmes quantiques qui vont suivre.

Cependant il est assez surprenant. Et il a été à l'origine de la découverte des autres.

Soit f une fonction de $\{0,1\}^n \rightarrow \{0,1\}$.

On suppose que f est soit la fonction constante à 0, soit la fonction constante à 1, soit une fonction « équilibrée » c'est-à-dire valant 0 la moitié du temps (et donc aussi 1 la moitié du temps).

Le problème est de savoir si f est constante ou équilibrée de façon déterministe (i.e. non probabiliste).

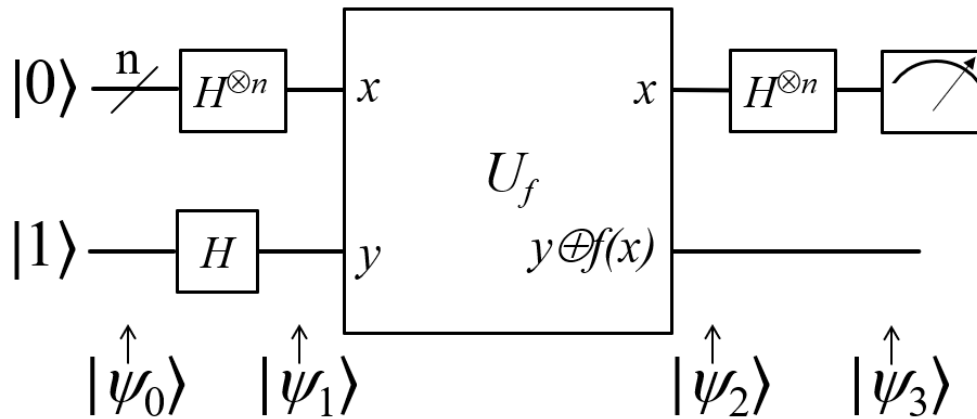
En classique : il faut faire $2^{n-1} + 1$ évaluations de f .

En quantique : une seule évaluation de f suffit.

L'algorithme de Deutsch-Jozsa semble donc montrer une différence entre P et QP.

1985 : étude du cas $n = 1$, 1992 : n quelconque, 1998 : améliorations

Deutsch-Jozsa, image



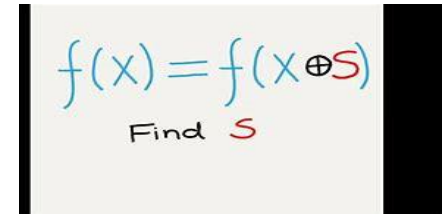
2. L'algorithme de Simon (1994)

Soit f une fonction de $\{0,1\}^n \rightarrow \{0,1\}^n$ admettant une période s .

Plus précisément, on suppose que :

$$\forall x, y \in \{0,1\}^n \quad f(x) = f(y) \Leftrightarrow (y = x) \text{ ou } (y = x \oplus s).$$

Le problème est de trouver s .



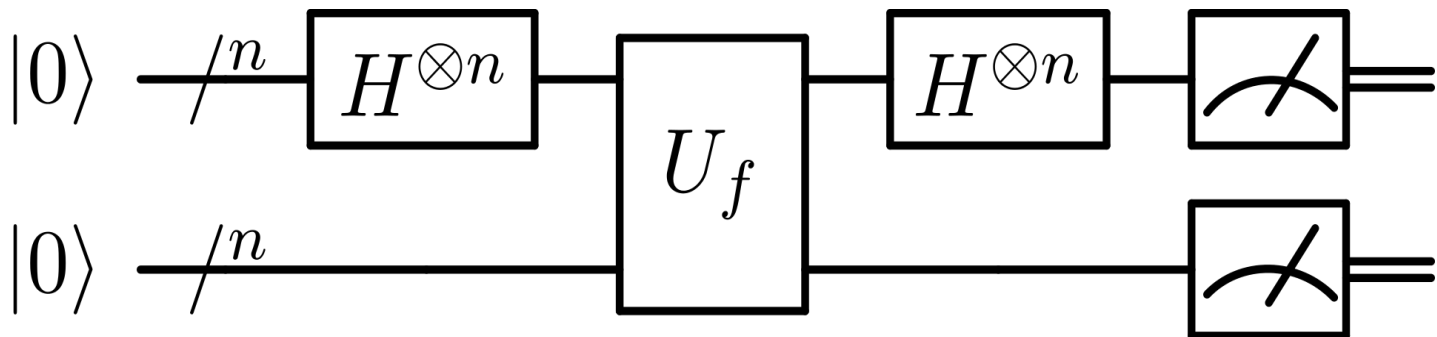
A handwritten equation on a light background: $f(x) = f(x \oplus s)$. Below the equation, the text "Find s" is written. The variable s is highlighted in red in both the equation and the text.

Sur un ordinateur classique il faut faire $O(2^{n/2})$ requêtes à la fonctions et autant de calculs pour trouver s (via le paradoxe des anniversaires, et on ne peut pas faire mieux).

Sur ordinateur quantique, avec l'algorithme de Daniel Simon, $O(n)$ requêtes suffisent, et ensuite il suffit de résoudre un système linéaire de $n-1$ équations à n inconnues (donc au plus $O(n^3)$ calculs).

L'algorithme de Simon semble donc montrer une différence entre BPP et BQP.

Simon, image



3. La factorisation Quantique : L'algorithme de Shor (1994)

- L'algorithme de Shor est un algorithme quantique pour factoriser un nombre N .

Temps de calcul : $O((\log N)^2 (\log \log N) (\log \log \log N))$

Espace mémoire : $O(\log N)$.

- 2001 : $15 = 3 * 5$ (avec 7 qubits)
- 2012 : $21 = 3 * 7$



Fonctionnement de l'algorithme de Shor (1/2)

- But : factoriser $n = pq$.
- Etape 1 (Classique) (C'est-à-dire non Quantique)
Si, pour un a aléatoire, on connaît r tel que
 $a^r \equiv 1 \pmod{n}$, alors on a une bonne probabilité de factoriser n .
 r est la période de $x \mapsto a^x \pmod{n}$.

r est pair donc $a^{r/2} \pmod{n}$ est une racine carrée de 1.

Si modulo n : $a^{r/2} \not\equiv 1$ et $a^{r/2} \not\equiv -1$ (on a 4 racines carrées de 1 mod pq) alors
 $\text{PGCD}(a^{r/2} - 1, n)$ donne p ou q .

Remarque : pour la plupart des a , r sera $\text{ppcm}(p-1, q-1)$

Fonctionnement de l'algorithme de Shor (2/2)

- Etape 2 (Quantique)

But : trouver r la période de $x \mapsto a^x \bmod n$.

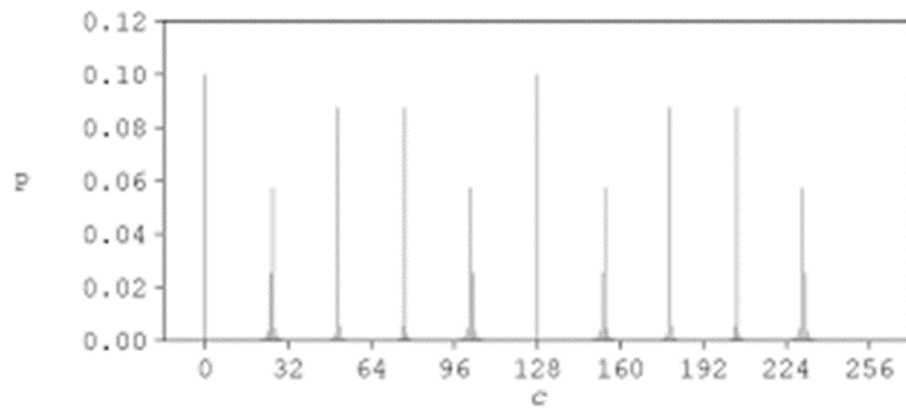
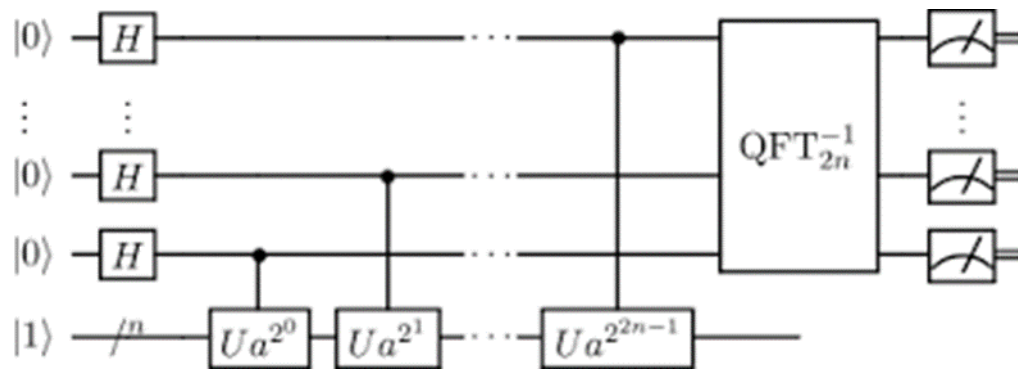
(Ceci peut être vu un cas particulier de l'algorithme de Simon.)

Cette valeur r s'obtient par en effectuant une Transformé de Fourier Quantique sur une superposition d'états des $a^x \bmod n$.

- 1) x est stocké avec des qubits, de telle sorte que toutes les valeurs possibles peuvent apparaître (on crée une superposition d'états).
- 2) On calcule $a^x \bmod n$ comme une transformation quantique (sans casser l'état intriqué de superposition quantique). (Ceci peut se faire par « square and multiply » quantique).
- 3) On effectue la Transformée de Fourier Quantique.
- 4) On fait une mesure.

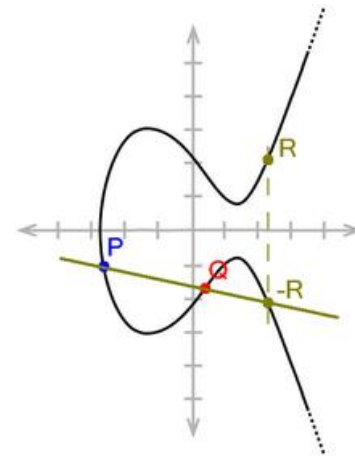
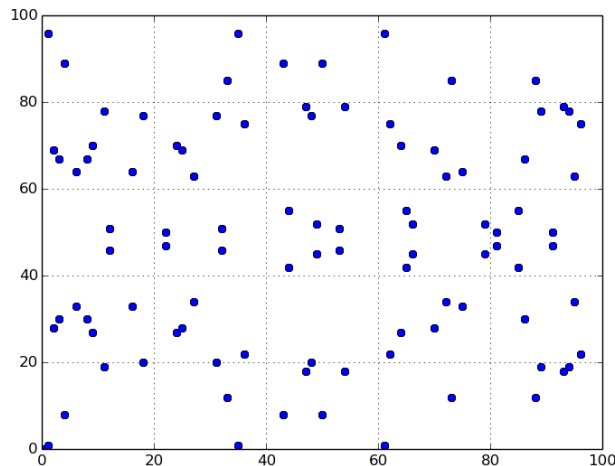
En pratique, l'étape 2 est la plus difficile (c'est celle qui utilise le plus de qubits).

Shor, images



4. Calcul du logarithme discret (toujours avec l'algorithme de Shor, 1994)

- Avec l'algorithme de Shor il est également possible de calculer le logarithme discret sur n'importe quel groupe avec une complexité polynomial (avec des ordinateurs quantiques).
- Par exemple sur ces groupes :
 - 1) mod p , avec p premier
 - 2) Sur le groupe multiplicatif d'un corps fini
 - 3) Sur les courbes elliptiques



Principe du calcul du logarithme discret avec Shor

But : dans un groupe fini G on cherche à résoudre en x l'équation :

$$g^x = y.$$

Notons p le nombre d'éléments de G .

Soit f l'application :

$$\begin{aligned} \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/p\mathbb{Z} &\rightarrow G \\ (a, b) &\mapsto g^a y^{-b} \end{aligned}$$

Si x est une valeur telle que $g^x = y$, alors $(x, 1)$ est une période de cette fonction f .

En effet $f(a + x, b + 1) = g^{a+x} y^{-b-1} = g^a y^{-b} g^x y^{-1} = g^a y^{-b} = f(a, b)$.

Ainsi en trouvant la période de f , on trouve $(x, 1)$ donc x .

Exemples en sécurité classique 2^{80}

- Pour factoriser un RSA de 1024 bits il faudrait environ **2000 qubits** (avec Shor).
- Pour calculer un log discret sur une courbe elliptique de 160 bits il faudrait environ **1000 qubits** (avec Shor), soit deux fois moins environ.

Algorithmes cryptographiques à clé publique concernés

- Factorisation : RSA, Fiat-Shamir, Guillou-Quisquater
- Log discret mod p : Diffie-Hellman, ElGamal, DSA
- Log discret sur courbes elliptiques : ECC (Elliptic curve cryptography)
- Log discret sur un corps fini : XTR (Lenstra, Verheul)

Soit **plus de 99 %** de la cryptographie à clé publique utilisée actuellement dans le monde.



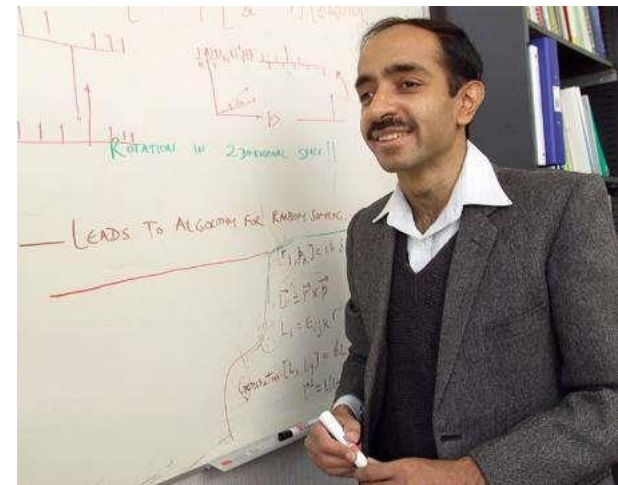
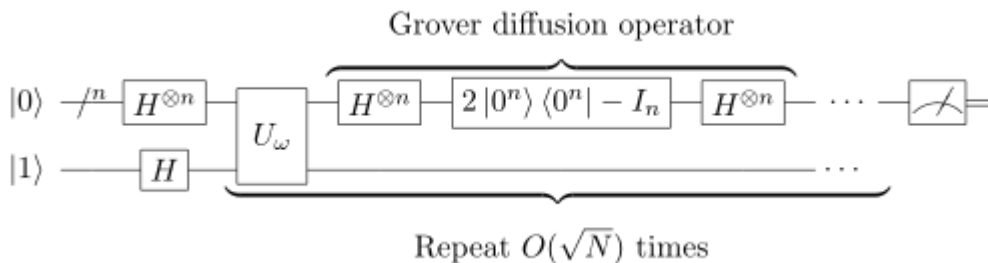
5. L'algorithme de Grover (1996)

Problème : trouver x tel que $f(x) = y$ lorsque x appartient à un ensemble de taille N .

L'algorithme de Grover est un algorithme quantique en $O(N^{1/2})$ en temps et en $O(\ln N)$ espace mémoire.

Ainsi on pourrait être amené à doubler la taille des clés cryptographiques des algorithmes à clé secrète.

L'algorithme de Grover n'obtient pas un gain aussi spectaculaire que celui de Shor, mais il s'applique à des problèmes bien plus divers.



Exemples d'applications récentes de l'algorithme de Simon

2010 : Kuwakado et Morii trouvent un **distingueur quantique polynomial en CPA sur les schémas de Feistel équilibrés de 3 tours** (au lieu de 2 tours en classique).

2019 : Ito, Hosoyamada, Matsumoto, Sasaki et Iwata trouvent un **distingueur quantique en CCA sur les schémas de Feistel équilibrés en 4 tours** (au lieu de 3 tours en classique).

2020 : Jacques Patarin, Aline Gouget, Ambre Toulemonde : distingueur sur certains MISTY et sur certains schémas de Feistel déséquilibrés (en préparation).

Remarques

1. Lorsque l'on cherche un distingueur plutôt que la période, on utilise une variante de l'algorithme de Simon qui fonctionne même s'il y a de multiples collisions en plus de la vraie période.

2. On peut aussi adapter ces attaques quantiques pour retrouver des clés secrètes lorsque les sous-clés sont Xorées à chaque tour.

Le problème du Sous-Groupe Caché (Hidden Subgroup problem)

Soit G un groupe, H un sous-groupe de G , et X un ensemble.

Soit f une fonction de G vers X .

Par définition, on dit que « f cache H » lorsque :

$$\forall g_1, g_2 \in G, f(g_1) = f(g_2) \Leftrightarrow g_1 H = g_2 H.$$

(Ceci signifie donc que f est constante sur les « cosets » de H , mais différente entre les cosets, où un « coset » (classe suivant un sous-groupe) est un ensemble de la forme $g H$).

Le problème du Sous-Groupe caché est, à partir d'un oracle donnant une telle fonction f , de trouver un ensemble générateur de H .

Lorsque G est abélien (i.e. commutatif) on connaît des algorithmes quantiques (Shor est un cas particulier de ce problème).

Lorsque G n'est pas abélien c'est un problème ouvert.

6. Set equality problem

On veut savoir si deux ensembles de N points sont égaux.

Complexité classique : N déterministe

Complexité quantique : $N^{1/3}$ (queries and computations)

(Cf Zhandry, Arxiv 2013, « A Note on the Quantum Collision and Set Equality Problems »).

7. Collisions pour une fonction aléatoire

On cherche une à résoudre $f(x) = f(y)$ où f est une fonction **aléatoire** de M point vers N points.

Pour qu'il au moins une solution avec une bonne probabilité il faut que $N \leq M^2$.

- Complexité classique : $N^{1/2}$ calculs (paradoxe des anniversaires).
- Complexité quantique : **$N^{1/3}$**

(Cf Zhandry, Arxiv 2013, « A Note on the Quantum Collision and Set Equality Problems »).

8. Recherche d'une seule collision

On cherche une à résoudre $f(x) = f(y)$ où f est une fonction de M point vers N points. On suppose qu'il existe une seule solution.

- Complexité classique : M calculs.
- Complexité quantique : $M^{2/3}$

(cf Ambainis).