Université de
Technologie de
Compiègne

TX n°7708 Simulateur pour l'algorithme de Shor

Rapport de Projet

Membre du binôme 1 :
Christler Jefferson MBOUOPDA FOTIE

Membre du binôme 2 :
Ruben MOUGOUÉ NJIETCHEU SIESHIEYO

Responsables :
- De la TX : Ahmed LOUNIS
- Du suivi : Vincent ROBIN

Printemps 2025

Sommaire

Introduction

À l'ère de la révolution numérique, la sécurité des données représente un enjeu majeur, conditionnant la confiance que nous accordons aux communications électroniques. Parmi les piliers de cette sécurité, la factorisation des grands nombres entiers occupe une place centrale. De nombreux systèmes de cryptographie, tels que le protocole RSA, reposent sur la difficulté intrinsèque de cette opération pour les ordinateurs classiques.

Cependant, l'apparition du calcul quantique est venue ébranler ces fondations. Dès 1994, Peter Shor proposa un algorithme quantique capable de factoriser des nombres entiers en un temps polynomial, défiant ainsi les limites des algorithmes classiques et annonçant de profondes transformations dans le domaine de la cybersécurité. L'algorithme de Shor représente ainsi un point de bascule potentiel, justifiant un intérêt croissant pour sa compréhension, sa simulation et son application.

Dans le cadre de ce travail de recherche, notre objectif est de concevoir un **simulateur informatique** de l'algorithme de Shor. Ce travail vise à offrir une exploration rigoureuse, à la fois théorique et pratique, des mécanismes fondamentaux qui sous-tendent cet algorithme, et à en analyser les performances en contexte simulé.

Notre rapport s'articule de la manière suivante :

— **Fondements théoriques** : nous reviendrons sur les concepts mathématiques et informatiques essentiels à la compréhension de l'algorithme, notamment l'algèbre linéaire quantique, la notion de calcul d'ordre modulo un entier, et les principes de base de l'informatique quantique.

— **Conception du simulateur** : nous détaillerons les choix de modélisation effectués, ainsi que l'architecture du simulateur que nous avons développé pour reproduire les principales étapes de l'algorithme de Shor.

À travers ce travail, nous cherchons à mieux appréhender non seulement les défis théoriques et pratiques liés au calcul quantique, mais aussi les implications profondes que de telles avancées pourraient engendrer sur les infrastructures numériques de demain.

Partie 1
Fondements Théoriques

Dans cette partie, nous présentons les bases théoriques nécessaires à la compréhension de l'algorithme de Shor, en mettant en lumière quatre concepts clés : le calcul de l'ordre, l'oracle quantique, la transformée de Fourier quantique inverse, et l'utilisation des fractions continues.

1.1 Calcul de l'ordre

Le problème de la factorisation d'un entier N repose, dans l'algorithme de Shor, sur la détermination de l'**ordre** d'un élément a dans le groupe multiplicatif modulo N. Plus précisément, l'ordre r de a est le plus petit entier strictement positif tel que :

$$a^r \equiv 1 \pmod{N}$$

Ce calcul est fondamental, car, sous certaines conditions, la connaissance de r permet d'obtenir des facteurs non triviaux de N grâce à des identités algébriques simples. En pratique, deux hypothèses doivent être vérifiées :

— L'ordre r doit être pair.

— $a^{r/2} \not\equiv -1 \pmod{N}$.

Sous ces hypothèses, la relation $(a^{r/2}-1)(a^{r/2}+1) \equiv 0 \pmod{N}$ permet, par un simple calcul de $\gcd$, de retrouver des diviseurs de N. Cependant, déterminer r est une tâche ardue sur un ordinateur classique, car elle nécessite en général une recherche exhaustive ou la connaissance préalable des facteurs de N (qui ne sont pas donné mais que l'on doit trouver).

1.2 Oracle quantique

L'oracle joue un rôle central dans l'algorithme de Shor : il encode la fonction exponentielle $f(k) = a^k \mod N$ dans un circuit quantique.
On rappelle qu'à une fonction $f : x \mapsto f(x)$ on associe l'oracle $F : (x, y) \mapsto (x, y \oplus f(x))$. Donc, l'oracle associé à la fonction $f : k \mapsto a^k \mod N$ est :

$$F : (k, y) \mapsto (k, y \oplus a^k \mod N),$$

Addition binaire. L'addition « $\oplus$ » est l'addition binaire sur les bits 0 et 1. Elle est équivalente au « ou exclusif » :

$$0 \oplus 0 = 0 \qquad 0 \oplus 1 = 1 \qquad 1 \oplus 0 = 1 \qquad 1 \oplus 1 = 0$$

Mais notre circuit sera toujours initialisé avec $y = 0$, donc dans notre situation nous considérerons :

$$F : (k, 0) \mapsto (k, a^k \mod N).$$

Le circuit de l'oracle est composé de deux registres :

— En entrée, le premier registre reçoit l'entier k, codé sur n bits ($2^n \geq N$), donc à l'aide de n lignes quantiques, et de même pour le second registre qui correspond à 0.
Codage de k sur n bits
Soit $k \in [0, N-1]$.
On peut écrire k en binaire sur n bits :

$$k = \sum_{i=0}^{n-1} k_i \cdot 2^i, \quad \text{avec } k_i \in \{0, 1\}$$

Le nombre k sera codé par le qubit $|k\rangle$, défini comme :

$$|k\rangle \equiv |k_{n-1} \dots k_1 k_0\rangle$$

Ce qubit est également équivalent au produit tensoriel des bits individuels :

$$|k\rangle = |k_0\rangle \otimes |k_1\rangle \otimes \cdots \otimes |k_{n-1}\rangle$$

— Nous avons également deux registres en sortie :
  — le premier renvoie k,
  — le second renvoie $a^k \mod N$.

L'oracle a bien pour action :

$$(k, 0) \mapsto (k, a^k \mod N).$$

En termes de qubits, si l'entrée de l'oracle est $|k\rangle \otimes |0\rangle$, alors la sortie est :

$$|k\rangle \otimes |a^k \mod N\rangle.$$

En exploitant la capacité du calcul quantique à superposer simultanément tous les états k, nous obtenons une superposition cohérente de toutes les valeurs a^k correspondantes. Cette propriété de *calcul parallèle quantique* est ce qui rend l'algorithme de Shor si puissant.

1.3 Transformée de Fourier quantique inverse

Après application de l'oracle, une mesure du second registre est effectuée. Cela projette le premier registre dans une superposition d'états séparés par l'ordre r recherché. Cependant, la structure périodique de cette superposition n'est pas directement exploitable sans traitement supplémentaire. La **transformée de Fourier quantique inverse** (QFT^{-1}) est alors appliquée au premier registre pour amplifier les fréquences correspondant aux multiples de $1/r$. La transformée de Fourier quantique est définie par :

$$\mathrm{QFT}(|k\rangle) = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{2i\pi \frac{kj}{2^n}} |j\rangle$$

Son inverse QFT^{-1} inverse ce processus de manière unitaire.

Illustration

Soit $\underline{a^{\beta_0}}$ la mesure du second registre (on peut considérer que l'état quantique du second registre s'est éffondré à $\left|\underline{a^{\beta_0}}\right\rangle$).

Alors le qubit du premier registre est :

$$\bar{\psi}_3 = \frac{r}{2^n} \sum_{\alpha=0}^{2^n/r-1} |\alpha r + \beta_0\rangle \,.$$

Calculons le qubit $\bar{\psi}_4$ obtenu après l'action de la transformée de Fourier inverse :

$$\bar{\psi}_4 = \hat{F}^{-1}\bar{\psi}_3 = \hat{F}^{-1}\left(\frac{r}{2^n} \sum_{\alpha=0}^{2^n/r-1} |\alpha r + \beta_0\rangle\right) = \frac{r}{2^n} \sum_{\alpha=0}^{2^n/r-1} \hat{F}^{-1} |\alpha r + \beta_0\rangle \,.$$

Appliquons maintenant la transformée de Fourier inverse sur chaque terme :

$$\bar{\psi}_4 = \frac{r}{2^n} \sum_{j=0}^{2^n-1} \sum_{\alpha=0}^{2^n/r-1} e^{-2i\pi(\alpha r+\beta_0)j/2^n} |j\rangle .$$

Pour la dernière égalité, nous avons interverti les deux sommes et utilisé que 2^n divise. Calculons maintenant le coefficient qui intervient dans cette somme à l'aide du lemme crucial :

$$\frac{1}{2^{n/r}} \sum_{\alpha=0}^{2^n/r-1} e^{-2i\pi \frac{\alpha j}{2^n/r}} = \begin{cases} 1 & \text{si } \frac{j}{2^n/r} \text{ est un entier,} \\ 0 & \text{sinon.} \end{cases}$$

Ainsi, nous obtenons :

$$\bar{\psi}_4 = \frac{1}{r} \sum_{j=0,\ldots,2^n-1,\ \frac{j}{2^n/r}\in\mathbb{Z}} e^{-2i\pi \frac{\beta_0 j}{2^n}} |j\rangle .$$

Pour la dernière égalité, nous avons simplement changé la notation des indices en posant $j = \frac{2^n \ell}{r}$.

Quelle est la valeur ajouté avec l'action de la transformée de Fourier ?

Tout d'abord, la constante β_0 n'apparaît que dans les coefficients des qubits et n'intervient plus après mesure.

Ensuite, la période que l'on veut déterminer est au dénominateur dans l'expression de l'entier $\frac{2^n \ell}{r}$, où n est le nombre de qubits utilisés (Les entiers n et ℓ ne sont pas connus (mais on sait que $0 \leq \ell \leq r-1$).).

Après cette étape, une mesure du premier registre fournit un résultat m proche d'un multiple de $\frac{2^n l}{r}$

Motivation pour les fractions continues

Nous avons fait une hypothèse simplificatrice : l'ordre r divise 2^n.

Ce n'est pas vrai en général, mais l'algorithme de Shor reste valide moyennant quelques adaptations.

Reprenons la fin du circuit de l'algorithme de Shor qui permet de calculer l'ordre r d'un élément.

— Si r divise 2^n, alors la mesure du premier registre conduit à un nombre rationnel $x = \frac{m}{2^n}$ qui est aussi égal à $\frac{\ell}{r}$. Ainsi, x est un multiple de $\frac{1}{r}$ et permet de retrouver r (ou au moins un facteur de r).

— Si r ne divise pas 2^n, alors la mesure du premier registre conduit à un nombre rationnel $x = \frac{m}{2^n}$ qui est proche d'un multiple de $\frac{1}{r}$ (mais n'est pas exactement un multiple).

1.4 Fractions continues

À partir de la valeur m mesurée, on calcule le rationnel :

$$x = \frac{m}{2^n}$$

qui est supposé proche d'une fraction de la forme $\frac{\ell}{r}$, $\ell \in \mathbb{Z}$. Cependant, en raison des approximations dues à la nature probabiliste de la mesure quantique, x n'est pas exactement égal à $\frac{\ell}{r}$. Pour récupérer r, on utilise le développement en **fractions continues** de x :

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots}}}$$

où chaque a_i est obtenu par divisions euclidiennes successives. En prenant les convergents successifs de cette expansion, on identifie la fraction dont le dénominateur r' satisfait les conditions du problème. Il faut ensuite vérifier que $a^{r'} \equiv 1 \pmod{N}$ pour confirmer que $r' = r$. Ce raffinement par fractions continues est indispensable surtout lorsque r ne divise pas 2^n exactement, situation générale dans la pratique.

1.5 Algorithme de Shor : description détaillée et fondements mathématiques

L'algorithme de Shor repose sur une combinaison d'étapes classiques et quantiques visant à factoriser un entier composite N.

Sa puissance repose sur la capacité des ordinateurs quantiques à extraire efficacement la **périodicité** d'une fonction arithmétique à l'aide de la **transformée de Fourier quantique (QFT)**.

Nous détaillons ici chaque étape de l'algorithme en introduisant les démonstrations associées aux traitements quantiques.

1.5.1 Partie Classique Préliminaire

1. **Test de primalité :**
   Avant toute chose, on vérifie si N est un nombre premier.
   Des algorithmes classiques comme le test de primalité AKS (Agrawal-Kayal-Saxena) peuvent accomplir cette tâche en temps polynomial.
   Si N est premier, il n'y a pas de facteurs non triviaux à trouver, et l'algorithme s'arrête.
   On peut aussi tester si N est une puissance d'un nombre premier (ex : $N = p^k$), ce qui peut se faire efficacement.
2. **Choix initial :**
   On sélectionne un entier a de manière aléatoire tel que $1 < a < N$.
3. **Recherche d'un facteur commun :**
   On calcule le plus grand commun diviseur (PGCD) de a et N à l'aide de l'algorithme d'Euclide.

   $$d = \gcd(a, N)$$

   Si $d > 1$, alors d est un facteur non trivial de N. Nous avons réussi à factoriser N et l'algorithme s'arrête. Si $d = 1$, cela signifie que a et N sont premiers entre eux. Nous devons alors passer à la partie quantique de l'algorithme pour trouver l'ordre de a modulo N.

1.5.2 Partie Quantique : Recherche de la Période

L'objectif de cette partie est de déterminer l'ordre r de a modulo N, qui est le plus petit entier positif tel que $a^r \equiv 1 \pmod{N}$. Cette valeur r est la période de la fonction $f(k) = a^k \pmod{N}$.

Cette étape est au cœur de l'algorithme et requiert une implémentation sur ordinateur quantique.

4. **Initialisation des registres quantiques :**
   Nous utilisons deux registres quantiques :
   - Le **premier registre** (ou registre d'entrée) est composé de n qubits, où n est choisi tel que $2^n \geq N^2$. Cette taille garantit une résolution suffisante pour distinguer les multiples de $1/r$.
   - Le **second registre** (ou registre de sortie) est composé de n qubits.

   Les deux registres sont initialisés à l'état $|0\rangle$. L'état initial du système global est donc :

$$|\psi_0\rangle = |0\rangle^{\otimes n} \otimes |0\rangle^{\otimes n} \tag{1.1}$$

   Pour simplifier la notation, nous l'écrirons $|0\rangle\,|0\rangle$.

5. **Création de la superposition uniforme :**
   On applique une porte d'Hadamard (H) à chacun des n qubits du premier registre. L'opérateur global est $H^{\otimes n} \otimes Id(|0\rangle\,|0\rangle)$.

   Démonstration mathématique :
   L'action d'une porte d'Hadamard sur un seul qubit est $H\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.
   Appliquée à n qubits, elle génère une superposition uniforme de tous les 2^n états de la base de calcul :

$$\begin{aligned} |\psi_1\rangle &= (H^{\otimes n} \otimes Id(|0\rangle\,|0\rangle) = \left(H^{\otimes n}\,|0\rangle^{\otimes n}\right) \otimes |0\rangle^{\otimes n} \\ &= \left(\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle\right) \otimes |0\rangle \end{aligned} \tag{1.2}$$

   Le premier registre contient maintenant toutes les valeurs possibles de k de 0 à $2^n - 1$ en superposition.

6. **Application de l'oracle d'exponentiation modulaire :**
   On applique un oracle quantique, implémenté par un opérateur unitaire U_f, qui calcule la fonction $f(k) = a^k \pmod N$ en parallèle.
   Son action sur un état de la base est $|k\rangle\,|y\rangle \mapsto |k\rangle\,|y \oplus a^k \pmod N\rangle$, où $\oplus$ est le OU exclusif bit à bit.
   En partant avec le second registre à $|0\rangle$, l'opération devient une simple affectation.

   Démonstration mathématique :
   En appliquant U_f à l'état $|\psi_1\rangle$:

$$\begin{aligned} |\psi_2\rangle = U_f\,|\psi_1\rangle &= U_f\left[\left(\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle\right) \otimes |0\rangle\right] \\ &= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} U_f(|k\rangle \otimes |0\rangle) \\ &= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle \otimes |a^k \pmod N\rangle \end{aligned} \tag{1.3}$$

   La valeur k dans le premier registre peut être décomposée par division euclidienne par la période r :

$$k = \alpha r + \beta \tag{1.4}$$

   où :

— $r = \mathrm{ord}_N(a)$ est l'ordre de a dans le groupe des unités $(\mathbb{Z}/N\mathbb{Z})^*$.
— β est le reste, avec $0 \leq \beta < r$.
— α est le quotient, avec $0 \leq \alpha < 2^n/r$ (puisque $k < 2^n$).

L'état du système après l'oracle peut donc être exprimé comme suit :

$$|k\rangle \otimes |a^k\rangle = |\alpha r + \beta\rangle \otimes |a^\beta\rangle \tag{1.5}$$

La simplification dans le second registre est justifiée par le fait que $a^r \equiv 1 \pmod N$:

$$a^k = a^{\alpha r+\beta} = (a^r)^\alpha \cdot a^\beta \equiv 1^\alpha \cdot a^\beta \equiv a^\beta \pmod N \tag{1.6}$$

À ce stade, les valeurs de $a^k \pmod N$ ont été calculées et sont intriquées avec les valeurs de k correspondantes.

7. **Mesure du second registre :**
On effectue une mesure sur le second registre.
Cela projette l'état du système.
Supposons que la mesure donne une valeur y_0. Comme la fonction $f(k) = a^k \pmod N$ est périodique avec une période r, plusieurs valeurs de k peuvent donner le même résultat y_0.

Démonstration mathématique :
Soit $y_0 = a^{\beta_0} \pmod N$ le résultat de la mesure, pour un certain $\beta_0 \in \{0, 1, \ldots, r-1\}$.
Le postulat de la mesure stipule que l'état du premier registre s'effondre en une superposition des seules valeurs de $|k\rangle$ compatibles avec ce résultat.
Ces valeurs de k sont celles qui satisfont $a^k \equiv y_0 \pmod N$, c'est-à-dire $k = \alpha \cdot r + \beta_0$ pour différents entiers α.
L'état du premier registre, après mesure et renormalisation, devient :

$$|\psi_3\rangle = \frac{1}{\sqrt{2}} \times \sqrt{r} \sum_{\alpha=0}^{(2^n/r)-1} |\alpha \cdot r + \beta_0\rangle \otimes |a_0^\beta\rangle$$

L'état $|\psi_3\rangle$ est une superposition périodique, où la période des indices est r.

8. **Application de la Transformée de Fourier Quantique Inverse (QFT^{-1}) :**
Pour extraire l'information sur la période r de l'état $|\psi_3\rangle$, on applique la QFT^{-1} au premier registre. La QFT^{-1} est conçue pour transformer un état périodique en un état où les pics de probabilité correspondent à la fréquence de cette période.

Démonstration mathématique :
L'action de la QFT^{-1} sur un état de base $|x\rangle$ est : $QFT^\dagger |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{-2\pi i x j/2^n} |j\rangle$.
Appliquons-la à $|\psi_3\rangle$:

$$\begin{aligned}
|\psi_4\rangle = QFT^\dagger |\psi_3\rangle &= \frac{1}{\sqrt{2}} \times \sqrt{r} \sum_{\alpha=0}^{(2^n/r)-1} QFT^\dagger |\alpha \cdot r + \beta_0\rangle \\
&= \frac{1}{\sqrt{2}} \sum_{\alpha=0}^{(2^n/r)-1} \left(\frac{1}{\sqrt{2^n}} \times \sqrt{r} \sum_{j=0}^{2^n-1} e^{-2\pi i(\alpha \cdot r+\beta_0)j/2^n} |j\rangle \right) \\
&= \frac{1}{\sqrt{2^n}} \times \sqrt{r} \sum_{j=0}^{2^n-1} \left(\sum_{\alpha=0}^{(2^n/r)-1} e^{-2\pi i(\alpha \cdot r+\beta_0)j/2^n} \right) |j\rangle
\end{aligned} \tag{1.8}$$

En factorisant le terme indépendant de q :

$$|\psi_4\rangle = \frac{1}{\sqrt{2^n}} \times \sqrt{r} \sum_{j=0}^{2^n-1} e^{-2\pi i \beta_0 j/2^n} \left(\sum_{\alpha=0}^{(2^n/r)-1} e^{-2\pi i r j/2^n} \right) |j\rangle \tag{1.9}$$

La somme sur α est une série géométrique.
Son amplitude est maximale lorsque tous les termes s'additionnent de manière constructive, c'est-à-dire lorsque la phase $e^{-2\pi i r j/2^n}$ est proche de 1.

9. **Mesure finale du premier registre :**
On mesure le premier registre.
La probabilité de mesurer un état $|j\rangle$ est proportionnelle au carré de son amplitude dans $|\psi_4\rangle$.
Cette probabilité est élevée uniquement pour les valeurs de j qui satisfont la condition d'interférence constructive :

$$\frac{rj}{2^n} \approx c \quad \text{pour un certain entier } c \in \{0, 1, \ldots, r-1\}$$

En réarrangeant, on obtient :

$$\frac{j}{2^n} \approx \frac{c}{r} \tag{1.10}$$

Ainsi, la mesure nous donnera une valeur j telle que la fraction $\frac{j}{2^n}$ est une excellente approximation d'une fraction ayant la période r comme dénominateur.

1.5.3 Partie Classique Post-Traitement

10. **Extraction de la période r :** À partir de la valeur mesurée j et de la taille du registre n, nous avons une approximation de c/r.
    (a) On utilise l'**algorithme des fractions continues** sur la valeur décimale $x = j/2^n$. Cet algorithme est extrêmement efficace pour trouver la meilleure approximation rationnelle c'/r' d'un nombre réel, avec un dénominateur $r' < 2^n$.
    (b) Avec une forte probabilité, le dénominateur r' trouvé sera la période r recherchée (ou un de ses diviseurs, si c et r n'étaient pas premiers entre eux).
    (c) On vérifie si notre candidat r' est bien la période en calculant $a^{r'} \pmod N$. Si le résultat est 1, nous avons trouvé un candidat valide pour la période. Sinon, il faut soit essayer les multiples de r', soit relancer l'algorithme quantique pour obtenir une autre valeur de j.
11. **Calcul des facteurs de N :** Une fois la période r trouvée, on l'utilise pour factoriser N.
    (a) Si r est un nombre impair, l'algorithme échoue pour ce choix de a. Il faut alors retourner à l'étape 2 et choisir un nouveau a.
    (b) Si r est pair, nous pouvons écrire l'équation $a^r \equiv 1 \pmod N$ comme $(a^{r/2}-1)(a^{r/2}+1) \equiv 0 \pmod N$. Cela signifie que N divise le produit $(a^{r/2}-1)(a^{r/2}+1)$.
    (c) Si de plus, $a^{r/2} \not\equiv -1 \pmod N$, alors N ne divise pas individuellement $(a^{r/2}+1)$. Dans ce cas, les facteurs de N doivent être partagés entre $(a^{r/2}-1)$ et $(a^{r/2}+1)$.
    (d) On peut donc trouver un facteur non trivial de N en calculant :

$$d_1 = \gcd(a^{r/2}-1, N) \quad \text{et} \quad d_2 = \gcd(a^{r/2}+1, N) \tag{1.11}$$

    Ces PGCD peuvent être calculés efficacement et fourniront des facteurs de N. Si $a^{r/2} \equiv -1 \pmod N$, l'algorithme échoue pour ce a et doit être redémarré. Il est prouvé qu'au moins la moitié des choix possibles pour a conduisent à un succès.

Ce processus, combinant des étapes classiques avec un cœur quantique puissant, permet de factoriser N en un temps polynomial en $\log N$, ce qui représente une améliioration considérable par rapport aux meilleurs algorithmes classiques connus.

1.6 Complexité et avantages par rapport aux algorithmes classiques

L'algorithme de Shor a profondément bouleversé notre perception des limites de l'informatique classique en matière de factorisation d'entiers. Dans cette section, nous analysons d'abord la complexité théorique de l'algorithme, puis nous discutons de ses avantages par rapport aux méthodes classiques.

1.6.1 Complexité de l'algorithme de Shor

L'algorithme de Shor est remarquable car il parvient à factoriser un entier N en un temps **polynomial** par rapport à la taille de l'instance (l'entrée), c'est-à-dire en $\mathcal{O}((\log N)^3)$ opérations quantiques. Plus précisément :

— La préparation de la superposition initiale est réalisée en $\mathcal{O}(n)$ opérations, où $n = \lceil \log_2 N \rceil$.

— L'évaluation de l'oracle, c'est-à-dire le calcul de $a^k \mod N$ pour tous les k, peut être effectuée efficacement grâce aux circuits d'exponentiation modulaire, en temps $\mathcal{O}(n^3)$.

— L'application de la transformée de Fourier quantique (QFT) inverse nécessite seulement $\mathcal{O}(n^2)$ opérations élémentaires.

— L'étape de traitement classique (fractions continues et vérification de r) est également polynomial en $\log N$.

Ainsi, dans son ensemble, l'algorithme de Shor permet de factoriser un nombre en un temps polynomial, une prouesse impossible avec les méthodes classiques connues.

1.6.2 Comparaison avec les algorithmes classiques

Les meilleurs algorithmes classiques de factorisation, tels que :

— l'algorithme de **crible quadratique** (Quadratic Sieve),

— l'algorithme du **crible général du corps de nombres** (GNFS),

ont une complexité sous-exponentielle, typiquement de la forme :

$$\exp\left((c + o(1))(\log N)^{1/3}(\log \log N)^{2/3}\right)$$

où c est une constante strictement positive. Cela signifie que, bien qu'ils soient plus rapides que des algorithmes exponentiels purs (comme la méthode de Fermat), leur temps d'exécution croît encore très rapidement avec la taille de l'entrée. En revanche, l'algorithme de Shor, en exploitant les propriétés de la mécanique quantique, effectue la recherche de la périodicité cachée de manière beaucoup plus efficace, rendant possible la factorisation d'entiers de plusieurs milliers de bits en un temps raisonnable — à condition de disposer d'un ordinateur quantique suffisamment puissant.

1.6.3 Impact en cryptographie

La cryptographie moderne repose en grande partie sur la difficulté de factoriser de grands entiers, notamment pour la sécurité des protocoles RSA. L'existence théorique de l'algorithme de Shor signifie que :

— Tous les systèmes fondés sur la difficulté de factorisation (RSA, clés Diffie-Hellman, etc.) seraient vulnérables à un ordinateur quantique opérationnel.

— De nouvelles normes cryptographiques, dites **post-quantiques**, doivent être développées pour garantir la sécurité à long terme.

Ainsi, au-delà de ses performances impressionnantes, l'algorithme de Shor a eu un impact majeur sur la recherche en sécurité informatique et continue d'influencer les stratégies de cryptographie moderne.

Conclusion

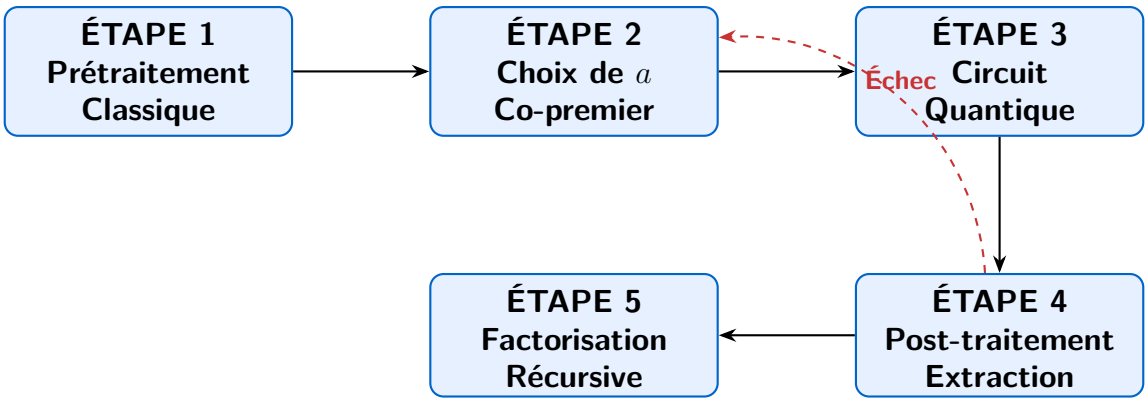
Ces outils théoriques constituent l'ossature de l'algorithme de Shor. Le calcul de l'ordre, l'oracle, la transformée de Fourier quantique inverse et l'analyse via les fractions continues interagissent harmonieusement pour contourner les limites de l'informatique classique et offrir une approche révolutionnaire de la factorisation.

Dans les sections suivantes, nous verrons comment mettre en pratique ces concepts pour développer un simulateur de l'algorithme.

Partie 2
Déroulement de l'algorithme pas à pas

Info : Objectif Principal de l'Algorithme

Factoriser efficacement un entier composé impair non trivial N en utilisant une sous-routine quantique pour déterminer la période r d'une fonction périodique liée à N.
Principe clé : Exploiter la périodicité de la fonction $f(x) = a^x \bmod N$ pour extraire des facteurs non triviaux de N via l'algorithme quantique d'estimation de phase.



Info : Complexité Quantique

$O(n^3)$ où n est le nombre de chiffres de N (donc en base 10), ce qui est polynomia

Info : Complexité Classique

Aucun algorithme connu n'a une complexité polynomiale ; les meilleurs sont sous-exponentiels

Info : Probabilité de Succès

$\geq 50\%$ Par tentative

ÉTAPE 1 : Prétraitement Classique Complet

Classical : Tests Séquentiels Obligatoires

Entrée : Un entier N strictement supérieur à 1, candidat à la factorisation.
Principe : Appliquer une série de tests classiques dans l'ordre suivant, afin de filtrer les cas triviaux avant d'envisager l'utilisation de l'algorithme de Shor.

1. **Test de Trivialité**
   - **Condition :** $N \leq 1$ *ou* N a déjà été entièrement factorisé
   - **Action :** ARRÊT IMMÉDIAT — aucun traitement supplémentaire requis
2. **Test de Parité**
   - **Condition :** N est pair ($N \bmod 2 = 0$)
   - **Action :** RETOUR du facteur 2 ; poursuivre la factorisation récursivement sur $N/2$
3. **Test de Primalité (Miller-Rabin)**
   - **Condition :** N est probablement premier (selon le test de Miller-Rabin appliqué à plusieurs bases)
   - **Action :** RETOUR immédiat de N comme facteur premier
4. **Test de Puissance Parfaite**
   - **Condition :** $N = a^b$ pour des entiers $a \geq 1$ et $b \geq 2$
   - **Action :** RETOUR de la base a ; factorisation récursive de a
5. **Candidat Validé pour Shor**
   - **Conditions :** N est composé, impair, et n'est pas une puissance parfaite
   - **Action :** Passage à l'**Étape 2** : sous-routine quantique de recherche de période

Error/Failure : Cas Particuliers et Optimisations

- **Division d'essai :** tester la divisibilité de N par une liste de petits nombres premiers $(2, 3, 5, 7, \dots)$ pour éliminer rapidement des facteurs simples
- **Nombres de Carmichael :** utiliser plusieurs bases dans le test de Miller-Rabin afin d'éviter les faux positifs (pseudo-premiers)

ÉTAPE 2 : Choix d'une Base a Co-première avec N

Quantum : Sélection Aléatoire de la Base

Objectif : Identifier un entier a tel que $\gcd(a, N) = 1$. Cette condition est indispensable pour assurer que la fonction $f(x) = a^x \bmod N$ soit périodique sans être constante.
Remarque : Si $\gcd(a, N) > 1$, alors a partage un facteur non trivial avec N — on a donc directement trouvé un facteur.

Procédure de Sélection

1. **Génération aléatoire** : Tirer uniformément un entier a dans l'intervalle $[2, N-2]$.
2. **Test de co-primalité** : Calculer $d = \gcd(a, N)$.
   — Si $d > 1$: Retour immédiat du facteur d.
   — Si $d = 1$: a est co-premier avec $N \rightarrow$ passer à l'étape quantique.

ÉTAPE 3 : Sous-routine Quantique — Estimation de la Période r

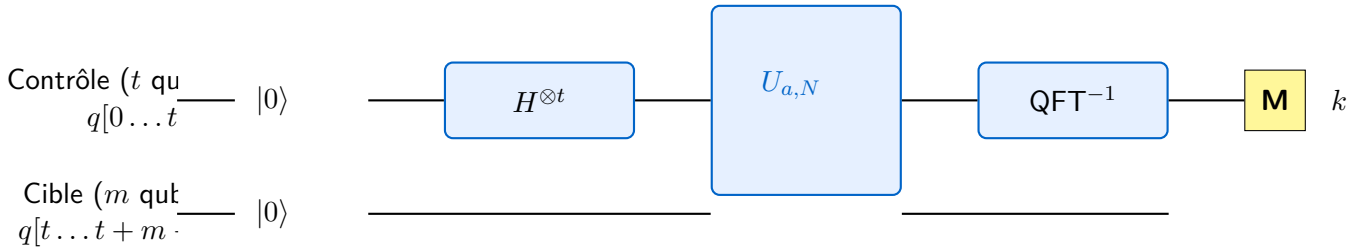
Quantum : Architecture du Circuit Quantique

Objectif : Estimer la période r de la fonction $f(x) = a^x \bmod N$, supposée périodique pour un entier a tel que $\gcd(a, N) = 1$.

Paramètres du Circuit

— **Registre de contrôle** : $t = \lceil 2 \log_2 N \rceil$ qubits
— **Registre cible** : $m = \lceil \log_2 N \rceil$ qubits
— **Précision souhaitée** : $Q = 2^t \geq N^2$
— **But** : Permettre une estimation fine de la phase $k/Q \approx s/r$

Séquence d'Opérations Détaillée



Légende : H = porte Hadamard, $U_{a,N}$ = oracle modulaire, QFT^{-1} = transformée de Fourier inverse, M = mesure (sur le registre de contrôle uniquement).

Description Formelle des Étapes

1. **Initialisation** :
$$|\psi_0\rangle = |0\rangle^{\otimes t} \otimes |0\rangle^{\otimes m}$$

2. **Création de la superposition** :
$$|\psi_1\rangle = \frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x\rangle \otimes |0\rangle^{\otimes m}$$
via l'application de $H^{\otimes t}$ sur le registre de contrôle.

3. **Évaluation de la fonction modulaire (oracle $U_{a,N}$)** :
$$|\psi_2\rangle = \frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x\rangle \otimes |a^x \bmod N\rangle$$

4. **Transformée de Fourier Quantique Inverse (QFT^{-1})** : appliquée sur le registre de contrôle uniquement.

5. **Mesure du registre de contrôle** : on obtient une valeur entière k telle que :
$$\frac{k}{Q} \approx \frac{s}{r}, \quad \text{avec } r \text{ la période cherchée.}$$

ÉTAPE 4 : Estimation de la Période r — Post-Traitement Classique

Quantum : Approximation par Fractions Continues

Objectif : À partir de la phase mesurée $\frac{k}{Q}$, déterminer une fraction irréductible $\frac{s}{r}$ telle que $a^r \equiv 1 \mod N$. Cette étape est réalisée par un algorithme classique.

Méthode par Fractions Continues

1. **Phase estimée** : $\phi = \frac{k}{Q}$, avec $k \in \mathbb{Z}$ et $Q = 2^t$.
2. **Développement en fraction continue** : calculer l'expansion $[c_0; c_1, c_2, \dots]$ de ϕ.
3. **Calcul des convergentes** : les fractions p_i/q_i donnent des approximations successives de ϕ.
4. **Filtrage** : pour chaque $q_i < N$, tester si $a^{q_i} \equiv 1 \mod N$.
5. **Sélection de la période** : le plus petit q_i vérifiant cette condition est un candidat pour la période r.

Classical : Extraction des Facteurs à partir de la Période

Entrée : un entier a tel que $\gcd(a, N) = 1$, et une période r vérifiée telle que $a^r \equiv 1 \pmod N$.

Tests de Validité de la Période r

Afin de garantir que la période r mène à une factorisation utile de N, les conditions suivantes doivent être vérifiées :

1. **Parité** : r doit être pair. Sinon, la tentative est abandonnée.
2. **Condition sur** $a^{r/2}$: calculer $x = a^{r/2} \mod N$. Si $x \equiv \pm 1 \pmod N$, la tentative échoue.

Info : Motif de l'Échec

Lorsque $x \equiv \pm 1 \pmod N$, les identités $(x - 1)$ et $(x + 1)$ ne révèlent aucun facteur non trivial de N.

Extraction Effective des Facteurs de N

Résumé

Si r est pair et $x = a^{r/2} \not\equiv \pm 1 \pmod N$, alors les facteurs non triviaux de N sont donnés par :

$$f_1 = \gcd(x - 1, N), \quad f_2 = \gcd(x + 1, N).$$

Info : Fondement Théorique

En effet, $a^r - 1 \equiv 0 \pmod{N}$ s'écrit :

$$N \mid (a^{r/2} - 1)(a^{r/2} + 1)$$

et comme N ne divise pas chacun des facteurs individuellement (sinon échec), on extrait un facteur strict via le PGCD.

Gestion des Échecs

— En cas d'échec (période non paire, ou $x \equiv \pm 1$), choisir un nouvel entier a aléatoire avec $\gcd(a, N) = 1$.

— Relancer la procédure quantique (étape 3) avec ce nouvel a.

[Probabilité de Succès] La probabilité de succès est élevée, et l'algorithme converge rapidement en moyenne.

ÉTAPE 5 : Factorisation Récursive Complète

Classical : Algorithme Récursif

Décomposer un entier N en facteurs premiers en appliquant récursivement l'algorithme de Shor sur chaque facteur composé.

Structure de la Récursion

— Initialement, on applique les tests classiques (Étape 1) sur N pour détecter rapidement des facteurs triviaux.

— Pour chaque facteur f_i obtenu :

  — Si f_i est premier, on l'ajoute à la liste finale.

  — Sinon, on relance l'algorithme de Shor sur f_i pour le décomposer davantage.

— Ce processus récursif s'arrête lorsque tous les facteurs sont premiers.

Optimisations Possibles

— **Parallélisation** : lancer plusieurs instances de Shor en parallèle sur des facteurs différents.

— **Limites de taille** : pour des facteurs trop petits, utiliser des méthodes classiques plus efficaces.

— **Cache des facteurs premiers** : mémoriser les facteurs déjà trouvés pour éviter les recalculs.

Analyse Complète des Cas d'Échec de la Routine Quantique

Error/Failure : Classification des Échecs et Stratégies

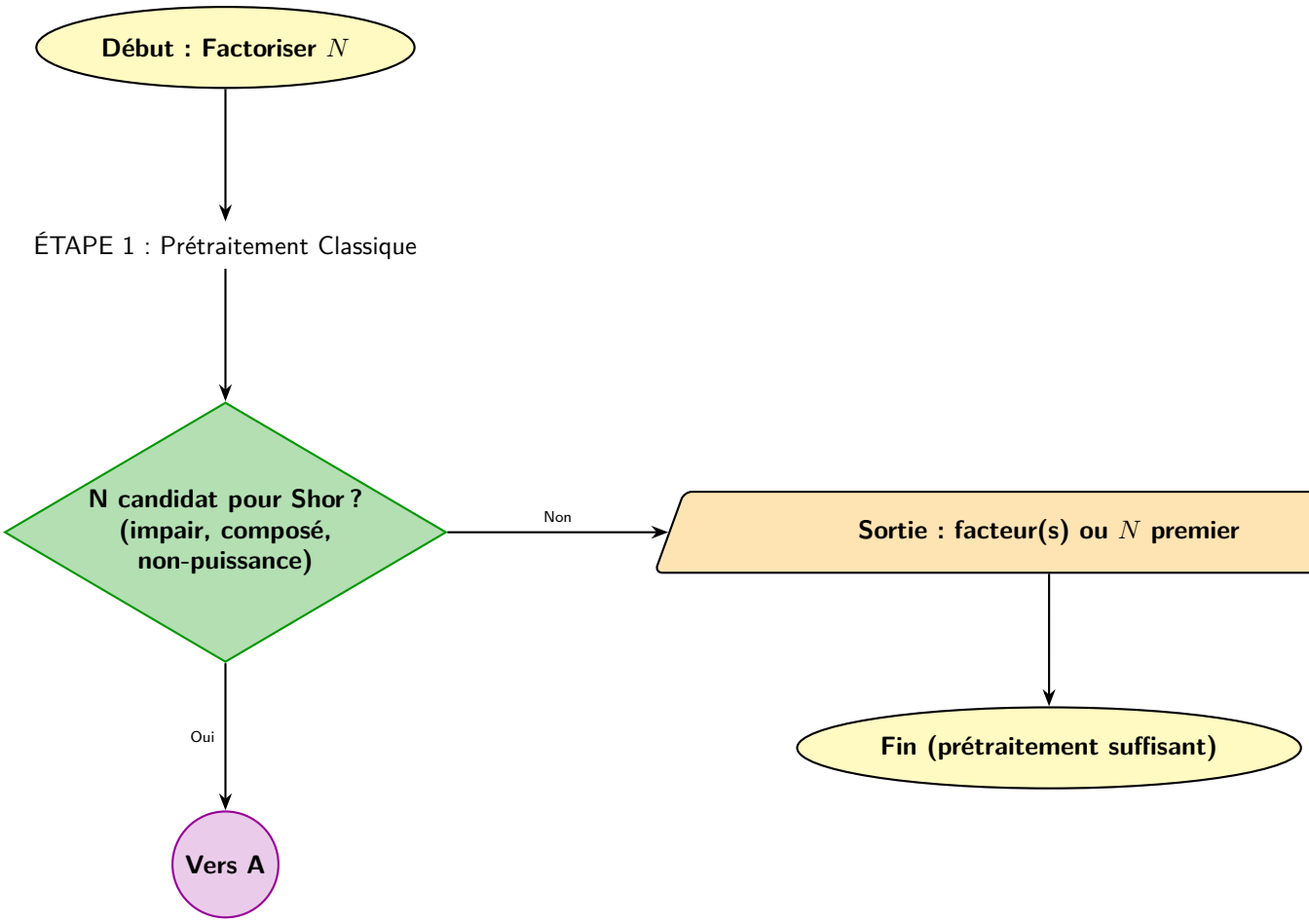
Les différentes causes d'échec et les méthodes de récupération associées sont synthétisées dans le tableau ci-dessous.

Type d'Échec	Cause / Observation	Action de Récupération	Probabilité Estimée	Impact
$\gcd(a, N) > 1$	a partage un facteur non trivial de N.	Succès immédiat : facteur trouvé.	$\approx 1 - \frac{\varphi(N)}{N}$	Positif
Période r non trouvée ou incorrecte	Phase k/Q imprécise, $\gcd(s, r) > 1$, taille Q insuffisante, bruit quantique.	Refaire la mesure, augmenter t, choisir nouveau a.	Faible si t choisi correctement	Échec temporaire
Période r impaire	Ordre de a modulo N est impair.	Choisir un nouvel a.	$\leq \frac{1}{2^{L-1}}$ (pour N L bits)	Échec
$a^{r/2} \equiv -1 \pmod N$	Valeur $x = N - 1$.	Choisir un nouvel a.	$\leq \frac{1}{2^{L-1}}$	Échec
$a^{r/2} \equiv 1 \pmod N$	$x = 1$, la période réelle est un diviseur strict de r.	Essayer $r/2$ ou nouveau a.	Cas particulier	Échec
Facteurs triviaux extraits	Cas où $x \equiv \pm 1$.	Relancer avec nouveau a.	Globalement $\leq \frac{1}{2}$	Échec
Phase $k = 0$ (information nulle)	$s = 0$, fraction nulle.	Répéter la mesure ou changer a.	$1/r$, faible	Répétition
Échec global après M tentatives	Limite du nombre d'essais, N premier ou erreur matérielle.	Vérifier primalité de N, augmenter t, arrêter si nécessaire.	Très faible	Fin de l'algorithme

Stratégies de Gestion des Échecs Globaux

- **Compteur de tentatives** : limiter le nombre de relances pour éviter boucle infinie.
- **Adaptation dynamique** : ajuster la précision t du registre de contrôle en fonction des échecs.
- **Changement d'entier** a : systématique pour diversifier les essais.
- **Test de primalité préalable** : éviter d'appliquer Shor sur un nombre déjà premier.

Diagramme de Flux Détaillé



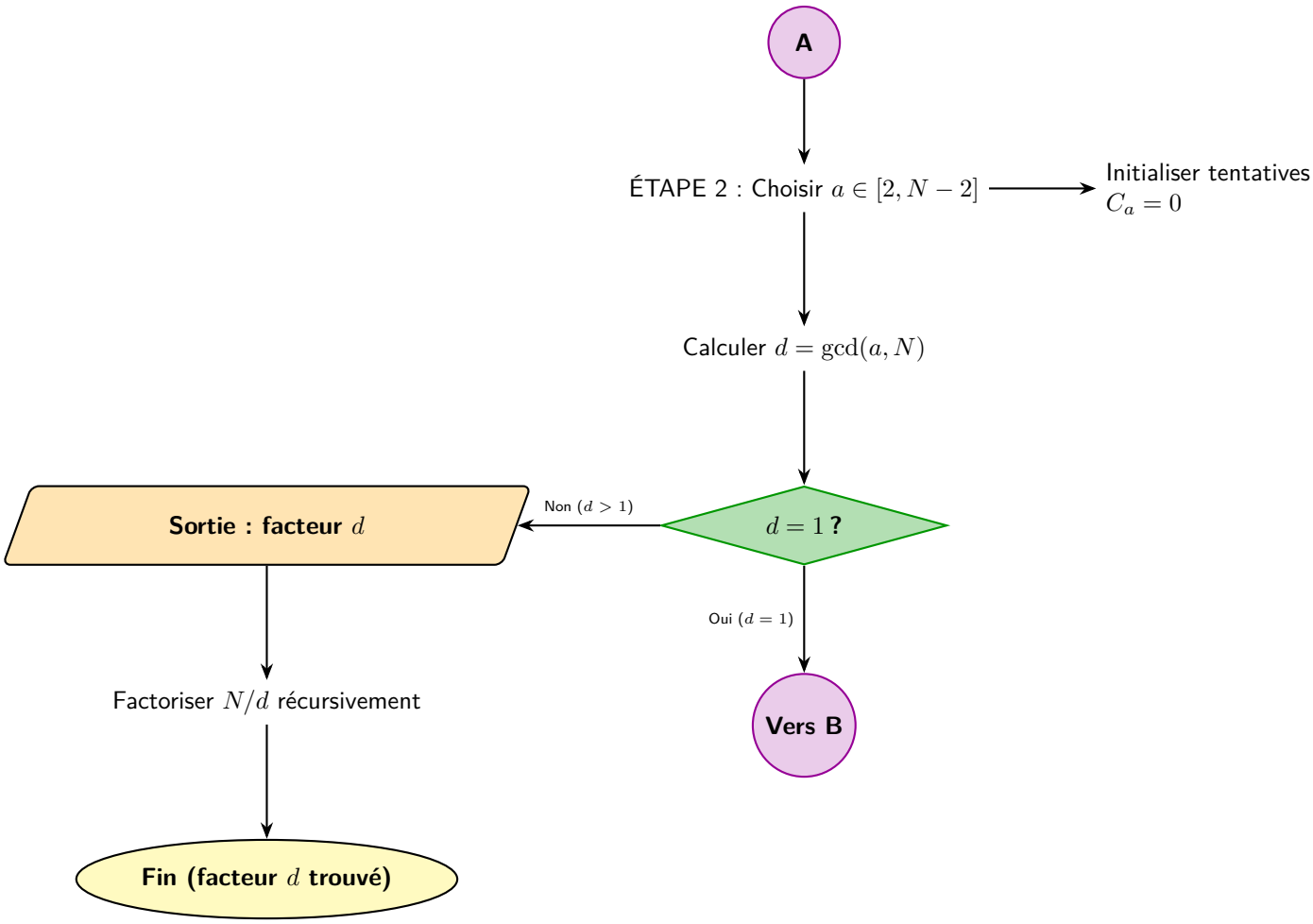
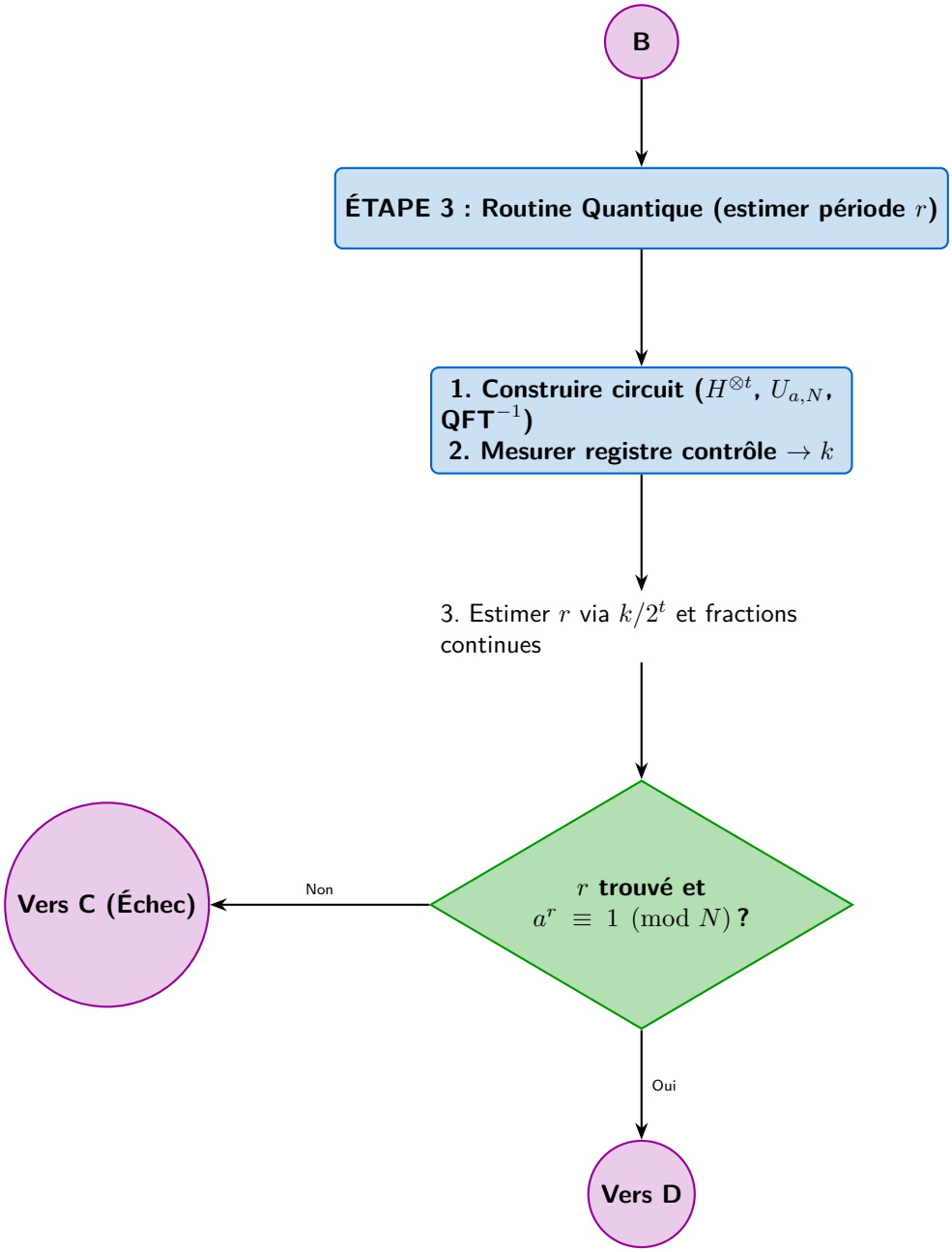
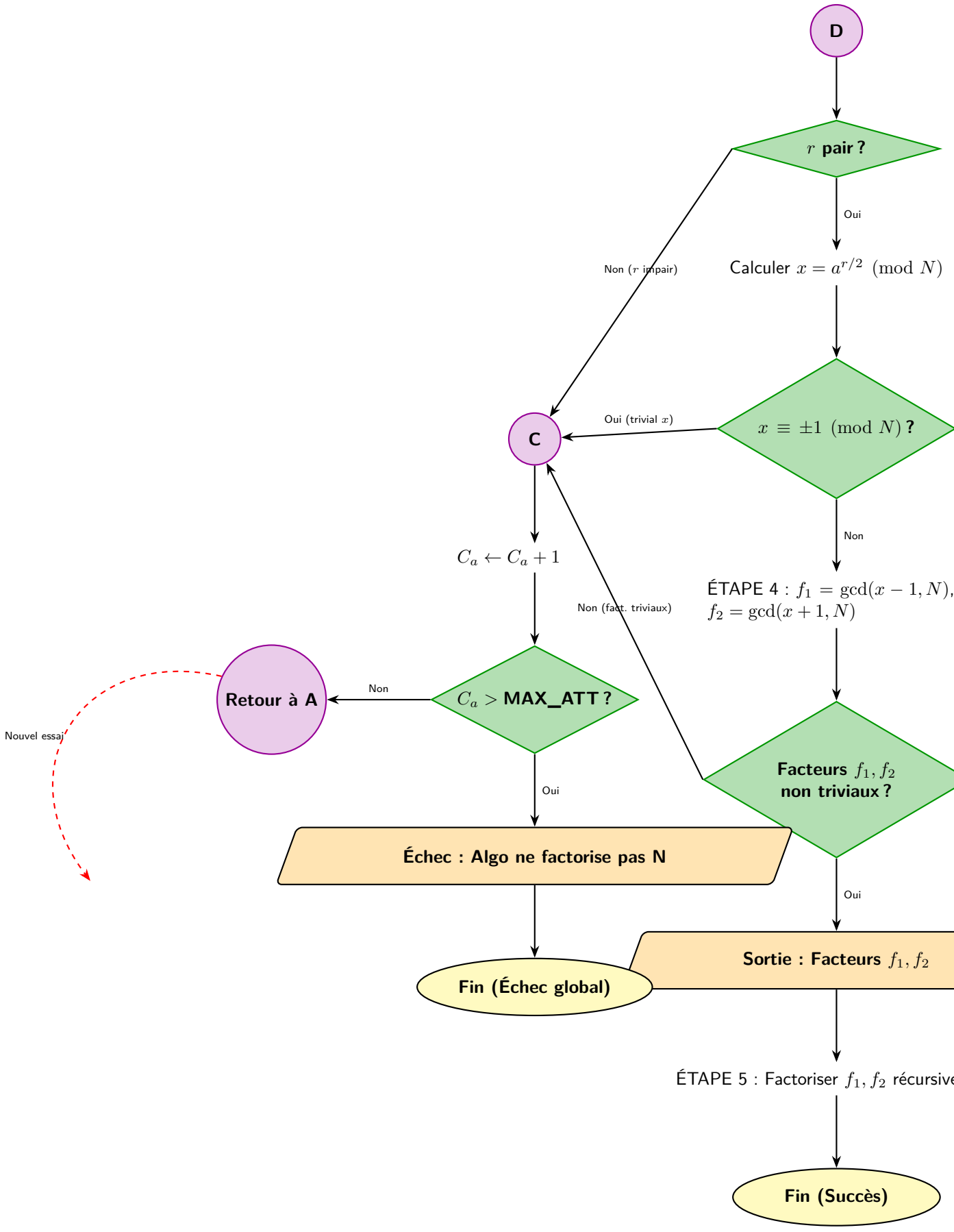

A
ÉTAPE 2 : Choisir $a \in [2, N-2]$
Initialiser tentatives
$C_a = 0$
Calculer $d = \gcd(a, N)$
$d = 1$?
Non ($d > 1$)
Sortie : facteur d
Oui ($d = 1$)
Vers B
Factoriser N/d récursivement
Fin (facteur d trouvé)

Diagramme de Flux Détaillé : Routine Quantique et Période




D
r pair ?
Oui
Non (r impair)
Calculer $x = a^{r/2} \pmod N$
$x \equiv \pm 1 \pmod N$?
Oui (trivial x)
C
Non
$C_a \leftarrow C_a + 1$
ÉTAPE 4 : $f_1 = \gcd(x-1, N)$, $f_2 = \gcd(x+1, N)$
Non (fact. triviaux)
Retour à A
Non
$C_a >$ MAX_ATT ?
Nouvel essai
Facteurs f_1, f_2 non triviaux ?
Oui
Échec : Algo ne factorise pas N
Oui
Sortie : Facteurs f_1, f_2
Fin (Échec global)
ÉTAPE 5 : Factoriser f_1, f_2 récursive
Fin (Succès)

Modules du Simulateur de l'Algorithme de Shor

Module (Ex : Classe)	Méthode Principale	Fonctionnalité Détaillée
`QuantumRegister/ QuantumCircuit`	`apply_hadamard(indices)`	Applique Hadamard aux qubits spécifiés. Crée superposition.
	`apply_mod_exp_oracle (a, N, ctrl, trgt)`	Implémente oracle $U_{a,N} : \|x\rangle\|0\rangle \to \|x\rangle\|a^x \bmod N\rangle$.
	`measure_qubits(indices)`	Simule mesure des qubits. Retourne valeur classique.
	`_qft_matrix(num_q, inv=F)`	Génère matrice QFT ou QFT^{-1}.
	`apply_qft_inverse(indices)`	Applique QFT^{-1} aux qubits de contrôle.
`ShorRunner`	`run_period_finding (a, N)`	Exécute circuit quantique complet (Init, H, Oracle, QFT^{-1}, Mesure). Retourne k.
	`_log_operation(op, dets)`	Journalise étapes et états intermédiaires pour débogage/analyse.
	`display_circuit_state (state, name)`	Affiche représentation (ASCII/graphique) de l'état du circuit.
`ClassicalPostProcessor`	`cont_frac_coeffs (num, den)`	Calcule coefficients de fraction continue de k/Q.
	`best_rational_approx (k, Q, N_val)`	Convergentes de $k/Q \to$ dénominateur $r_{\text{cand}} < N$.
`ShorOrchestrator`	`factorize(N, attempts)`	Orchestre algorithme complet. Gère prétraitement, choix a, routine quantique, post-traitement, récursion.
	`post_process_period (a, N, r_cand)`	Vérifie r_{cand} (pair, $a^{r/2} \not\equiv \pm 1$). Calcule facteurs.
	`find_smallest_r (a, N, r_cand)`	Si $a^{r_{\text{cand}}} \equiv 1$, trouve plus petit r (diviseurs de r_{cand}).
`ClassicalAlgos`	`trial_division(n, limit)`	Division d'essai par petits premiers.
	`is_prime_miller_rabin (n, k)`	Test de primalité Miller-Rabin.
	`extended_gcd(a,b)`	PGCD étendu de a, b.
	`is_perfect_power(n)`	Vérifie si $n = x^y, y \geq 2$.
	`modular_exponentiation (b,e,m)`	Calcule $(b^e) \bmod m$ efficacement.
`UserInterfaceCLI`	`run_cli()`	Lance interface utilisateur en ligne de commande.
	`configure_simulation()`	Permet config. interactive des paramètres.
	`factorize_interactive()`	Factorise nombre fourni par utilisateur.

Applications Pratiques de l'Algorithme de Shor

L'algorithme de Shor, bien que nécessitant un ordinateur quantique à grande échelle pour être pratiquement utile pour de grands nombres, a des implications profondes :

— **Cryptanalyse de RSA :** C'est l'application la plus célèbre. La sécurité du cryptosystème RSA repose sur la difficulté de factoriser de grands entiers semi-premiers. L'algorithme de Shor peut factoriser ces nombres en temps polynomial, rendant RSA vulnérable si des ordinateurs quantiques suffisamment puissants sont construits. Cela motive la recherche en cryptographie post-quantique.

— **Résolution du problème du logarithme discret :** Une variante de l'algorithme de Shor peut résoudre le problème du logarithme discret, qui est la base de la sécurité d'autres cryptosystèmes comme Diffie-Hellman et la cryptographie sur les courbes elliptiques (ECC).

— **Recherche de périodes dans les systèmes physiques et mathématiques :** Au-delà de la factorisation, la capacité de trouver des périodes est fondamentale. L'algorithme d'estimation de phase quantique, qui est le cœur de Shor, a des applications dans la simulation de systèmes quantiques, la chimie quantique, et la science des matériaux.

— **Démonstration du pouvoir du calcul quantique :** L'algorithme de Shor est l'un des exemples les plus frappants d'un problème pour lequel les ordinateurs quantiques offrent un avantage exponentiel par rapport aux meilleurs algorithmes classiques connus. Il stimule la recherche et le développement dans le domaine du matériel et des logiciels quantiques.

— **Développement de nouveaux algorithmes quantiques :** Les techniques utilisées dans l'algorithme de Shor, telles que la Transformée de Fourier Quantique et l'estimation de phase, sont des outils importants qui sont utilisés ou inspirent d'autres algorithmes quantiques.

Info : Conclusion

L'algorithme de Shor est une avancée théorique majeure qui illustre le potentiel transformationnel du calcul quantique. Sa "carte complète" révèle une interaction fascinante entre la théorie des nombres classique, l'algèbre, et les principes de la mécanique quantique. Bien que les défis technologiques pour sa mise en œuvre à grande échelle soient considérables, son étude continue d'enrichir notre compréhension des capacités quantiques et de façonner l'avenir de la cryptographie et du calcul.

Annexe – Ressources supplémentaires

Lien de l'application

Vous pouvez accéder au simulateur interactif développé dans le cadre de ce projet à l'adresse suivante :

https://shor-simulator-tx7704-p25.streamlit.app/

Code source

Le code source du simulateur de l'algorithme de Shor est disponible sur GitHub à l'adresse suivante :

`https://github.com/jefferson1203/Shor-simulator/tree/main`

Partie 3
Conclusion

Ce projet de recherche s'est attaché à la conception et au développement d'un simulateur pour l'algorithme de Shor, un algorithme quantique révolutionnaire capable de factoriser de grands nombres entiers à une vitesse inaccessible aux ordinateurs classiques. Notre objectif était de démystifier son fonctionnement en passant de la théorie à une application pratique et observable.

Notre démarche a d'abord consisté à décomposer les fondements théoriques complexes de l'algorithme. Nous avons étudié en détail les concepts mathématiques et quantiques essentiels : la recherche de l'ordre d'un nombre, le rôle de l'oracle quantique, la puissance de la Transformée de Fourier Quantique pour extraire une périodicité cachée, et enfin l'utilisation des fractions continues pour affiner le résultat.

Forts de cette compréhension, nous avons architecturé et programmé un simulateur modulaire. Celui-ci reproduit fidèlement chaque étape du processus : depuis la préparation classique des données, en passant par l'exécution de la routine quantique au cœur de l'algorithme, jusqu'au traitement final qui révèle les facteurs du nombre initial. Les diagrammes de flux et l'analyse des cas d'échec ont permis de construire une logique robuste, capable de gérer les différents scénarios possibles.

Le principal apport de ce travail est de rendre l'algorithme de Shor plus tangible et accessible. En transformant des concepts abstraits en un programme fonctionnel, notre simulateur offre un outil pédagogique et pratique. Il permet à quiconque s'intéresse au sujet d'explorer les mécanismes de la factorisation quantique et de visualiser son déroulement, sans disposer d'un véritable ordinateur quantique.

Finalement, ce projet illustre concrètement le changement de paradigme que représente le calcul quantique. Il met en lumière la menace que cette technologie fait peser sur les systèmes de cryptographie actuels, comme le RSA, et souligne l'urgence de développer des solutions de sécurité "post-quantiques". Ce simulateur n'est donc pas seulement l'aboutissement de notre recherche ; il constitue une base solide pour de futures explorations dans le domaine fascinant et prometteur de l'informatique quantique.

Bibliographie

Sources :

— Michael A. Nielsen et Isaac L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2010.

— Peter W. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, dans les actes de la *35th Annual Symposium on Foundations of Computer Science*, IEEE, 1994, pp. 124–134.

— Arnaud Bodin, *Un peu de mathématiques pour l'informatique quantique : Algorithmes et Mathématiques*, cours accessible en ligne, 2023.

— Richard J. Lipton, *Quantum Algorithms and Cryptographic Hardness*, blog et notes de cours, 2022.

— Walter Schon, *Informatique Quantique - Transformée de Fourier Qunatique et Algorithme de Shor, semestre d'automne 2024*, cours - Université de Technologie de Compiègne.

— *Shor's Algorithm*, Wikipédia, 2024. Disponible à l'adresse : `https://en.wikipedia.org/wiki/Shor's_algorithm`.