ALGORITHMES QUANTIQUES POUR LA RECHERCHE DE MOTIFS

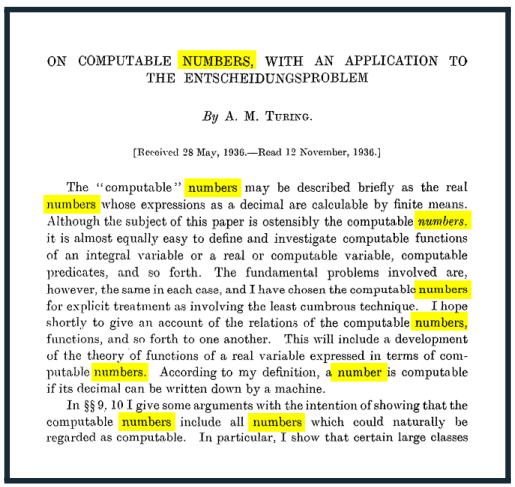
Mattéo Delabre

IFT6155 — Informatique quantique
Université de Montréal
3 mai 2022

PLAN

DÉFINITION DU PROBLÈME (1/2)

- **Recherche de motif *(pattern matching)* :** trouver la ou les occurrences d'un motif à l'intérieur d'un ensemble de données



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. Turing.

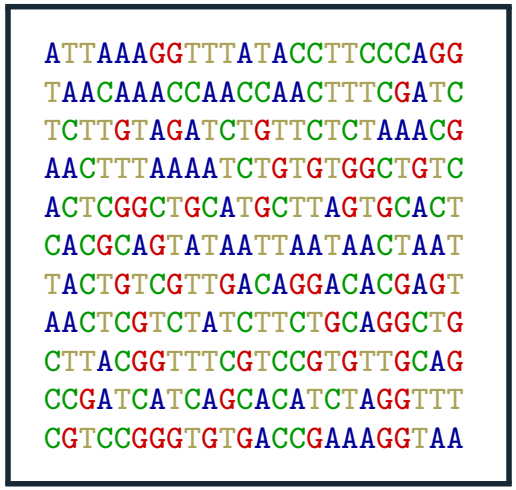
[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*. it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.

In §§ 9, 10 I give some arguments with the intention of showing that the computable numbers include all numbers which could naturally be regarded as computable. In particular, I show that certain large classes

Recherche dans un document

Dimension 1, exacte



ATTAAAGGTTTATACCTTCCCAGG
TAACAAACCAACCAACTTTCGATC
TCTTGTAGATCTGTTCTCTAAACG
AACTTTAAAATCTGTGTGGCTGTC
ACTCGGCTGCATGCTTAGTGCACT
CACGCAGTATAATTAATAACTAAT
TACTGTCGTTGACAGGACACGAGT
AACTCGTCTATCTTCTGCAGGCTG
CTTACGGTTTCGTCCGTGTTGCAG
CCGATCATCAGCACATCTAGGTTT
CGTCCGGGTGTGACCGAAAGGTAA

Identification de séquences

Dimension 1, avec « indels »

Reconnaissance d'objets

Dimension 2, approximative

DÉFINITION DU PROBLÈME (2/2)

Recherche de motif (dimension 1)

- Entrée :
  - **Alphabet** Σ (ex : $\Sigma = \{0, 1\}$, $\Sigma = \{A, C, G, T\}$, ...)
  - **Texte** t de N caractères
  - **Motif** p de M caractères

  $\Big\}\ N \gg M$
- Sortie :
  - Position des occurrences de p dans t

DÉFINITION DU PROBLÈME (2/2)

Recherche de motif (dimension 1)

- Entrée :
  - **Alphabet** Σ (ex : $\Sigma = \{0, 1\}$, $\Sigma = \{A, C, G, T\}$, ...)
  - **Texte** t de N caractères
  - **Motif** p de M caractères

  $\Big\}\ N \gg M$
- Sortie :
  - Position des occurrences de p dans t

- Variantes
  - Recherche simultanée de $P = \{p_1, p_2, ..., p_K\}$ motifs
  - Recherche d'occurrences approximatives (Hamming, Levenshtein)

ALGORITHME CLASSIQUE NAÏF

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
t	1	0	1	1	0	1	0	0	1	1	1	0	1	0	0	0
p	1	1	0	1	0	0										

Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

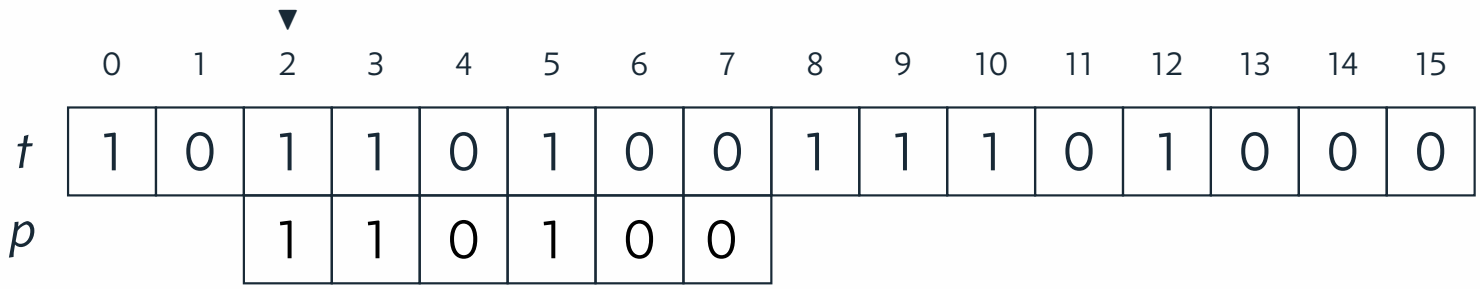
ALGORITHME CLASSIQUE NAÏF

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
t	1	0	1	1	0	1	0	0	1	1	1	0	1	0	0	0
p		1	1	0	1	0	0									

Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

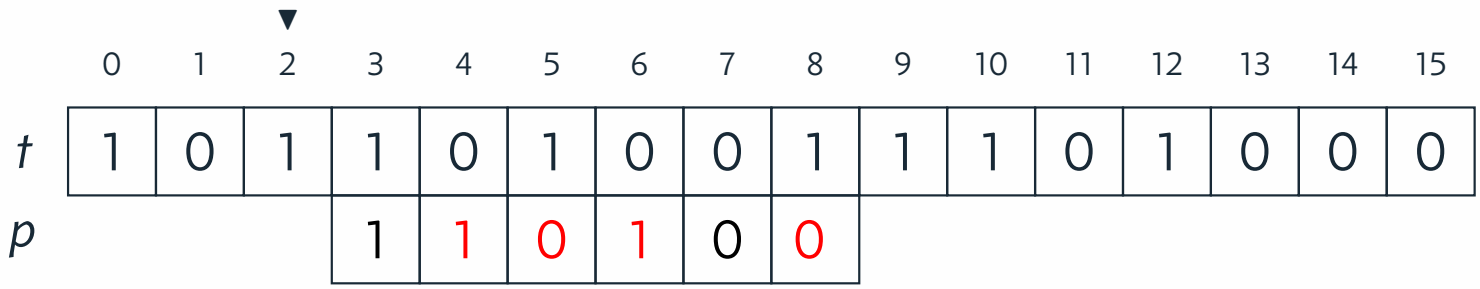
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

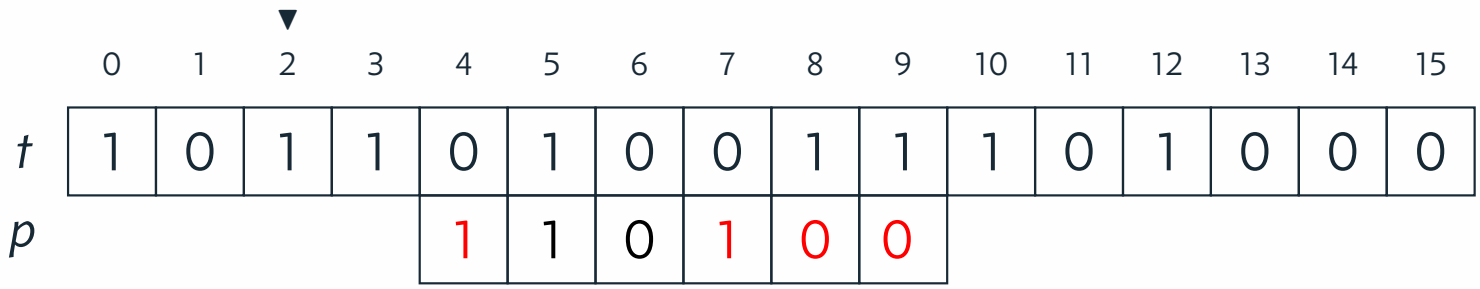
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

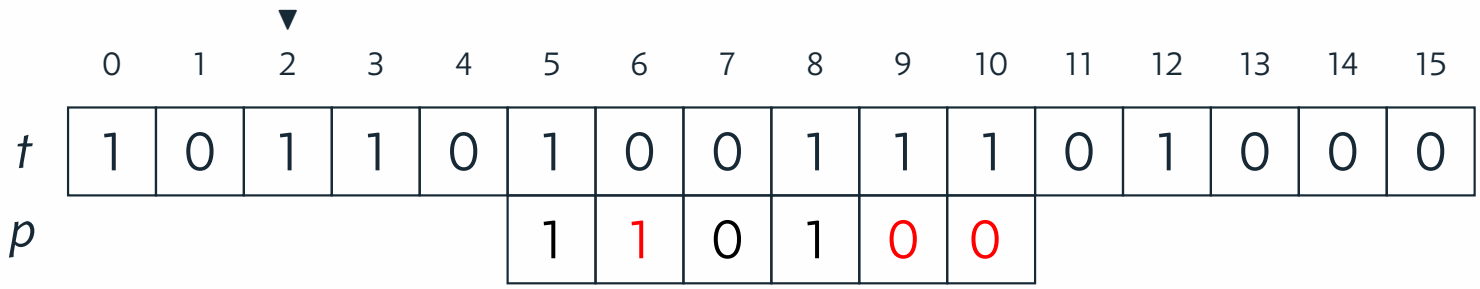
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

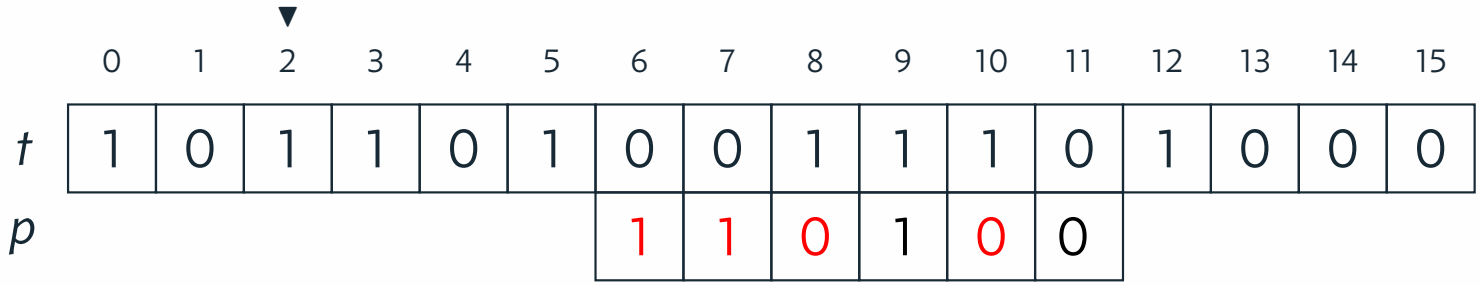
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

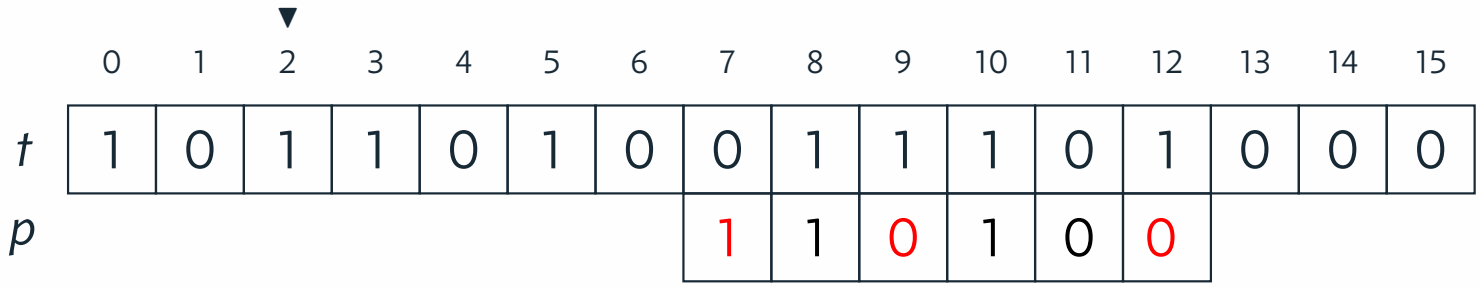
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

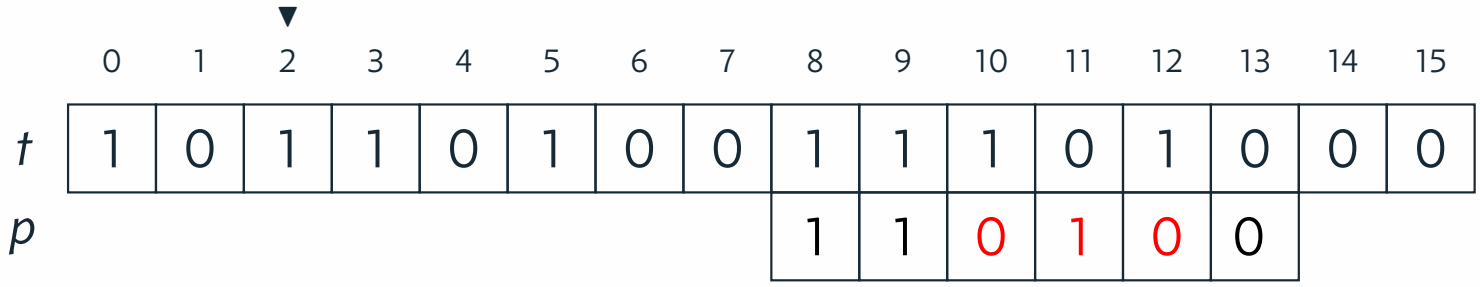
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

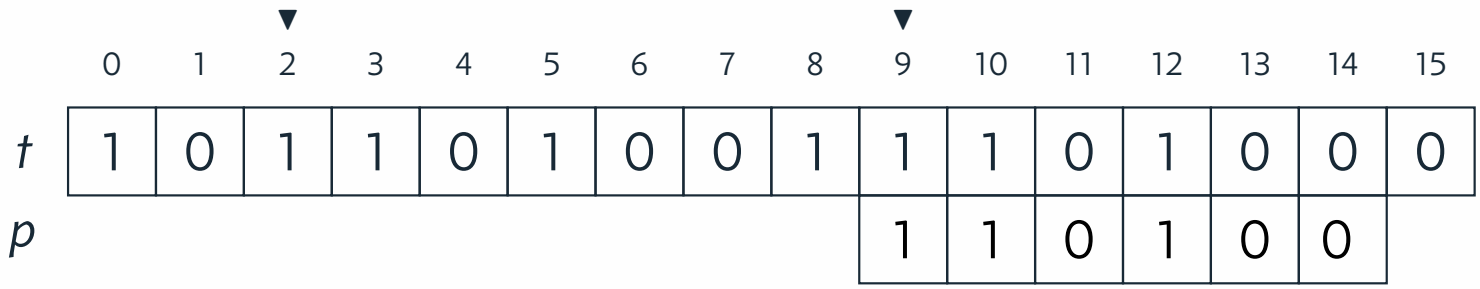
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

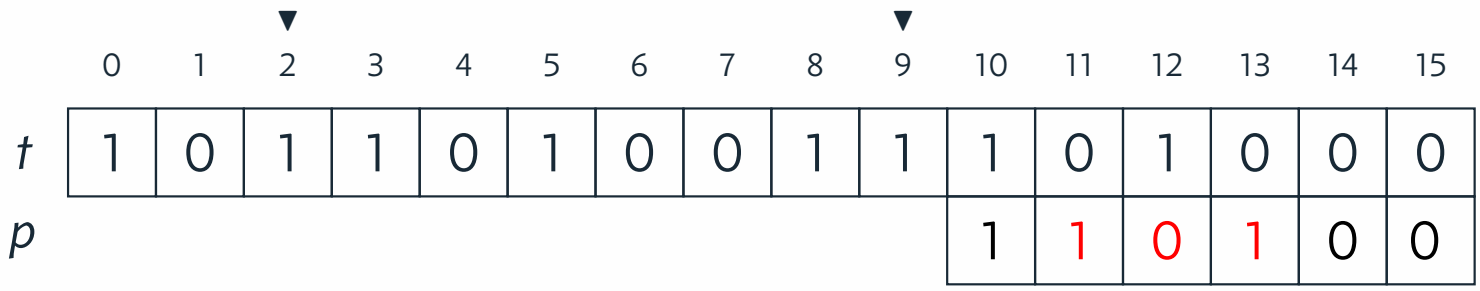
ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

ALGORITHME CLASSIQUE NAÏF



Pour i **de** 0 **à** $N - M$, **faire** :
    $j \leftarrow 0$
    Tant que $j < M$ et $p[j] = t[i + j]$, **faire** :
        $j \leftarrow j + 1$
    Si $j = M$ **alors**
        Signaler occurrence en i

$\Theta(NM)$

ALGORITHMES CLASSIQUES STANDARDS

- Recherche exacte d'**un motif**

(Naïf)	$\Theta(NM)$
Knuth–Morris–Pratt	$\Theta(N)$
Boyer–Moore	$\mathcal{O}(N)$ en moyenne

- Recherche exacte **simultanée de K motifs**

(Naïf)	$\Theta(NKM)$
Aho–Corasick	$\Theta(N + KM + c)$
Rabin–Karp	$\mathcal{O}(N + KM)$ en moyenne

AMÉLIORATION QUANTIQUE

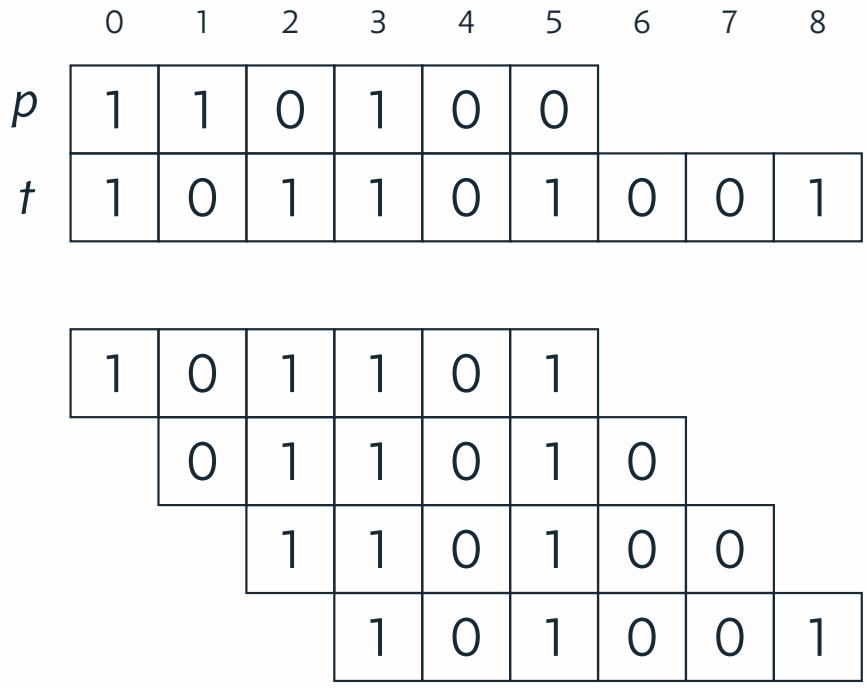
- Malgré l'efficacité des algorithmes classiques, trop **lents** pour certaines données **très volumineuses**
- Espoir d'une meilleure complexité avec un ordinateur quantique ?
- Algorithme de **Grover** : **accélération quadratique** pour recherche dans une base de données non-structurée
- Comment appliquer ce résultat à la recherche de motif ?

PLAN

SUPERPOSITION UNIFORME DES SOUS-MOTS

	0	1	2	3	4	5	6	7	8
p	1	1	0	1	0	0			
t	1	0	1	1	0	1	0	0	1

SUPERPOSITION UNIFORME DES SOUS-MOTS



SUPERPOSITION UNIFORME DES SOUS-MOTS

	0	1	2	3	4	5	6	7	8
p	1	1	0	1	0	0			
t	1	0	1	1	0	1	0	0	1

$$|\Psi_t\rangle = \tfrac{1}{2}\big(\,|0\rangle|\,1\,0\,1\,1\,0\,1\,\rangle + |1\rangle|\,0\,1\,1\,0\,1\,0\,\rangle + |2\rangle|\,1\,1\,0\,1\,0\,0\,\rangle + |3\rangle|\,1\,0\,1\,0\,0\,1\,\rangle\big)$$

registre de position
$\lg(N - M + 1)$ qubits

registre de recherche
M qubits

COMPARAISON DES SOUS-MOTS EN PARALLÈLE

$$p = 110100 \qquad t = 101101001$$

$$|\Psi_t\rangle = \frac{1}{2}(|0\rangle\,|101101\rangle + |1\rangle\,|011010\rangle + |2\rangle\,|110100\rangle + |3\rangle\,|101001\rangle)$$

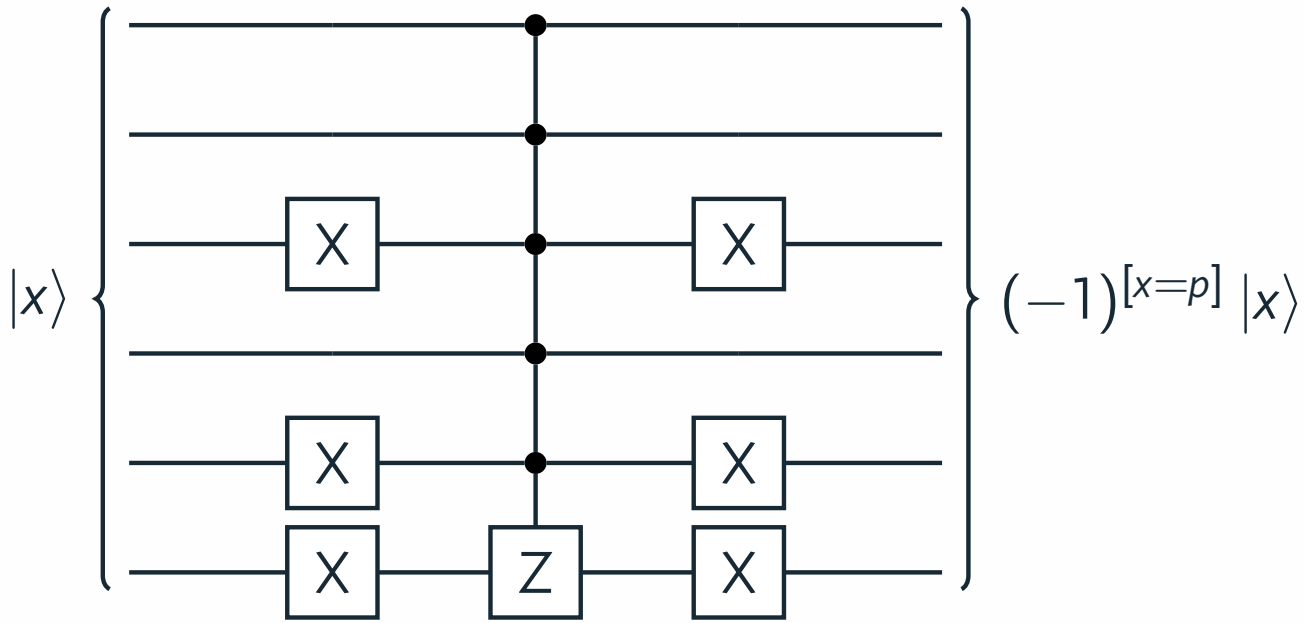
$$\downarrow \oplus p$$

$$|\Psi'_t\rangle = \frac{1}{2}(|0\rangle\,|011001\rangle + |1\rangle\,|101110\rangle + |2\rangle\,|000000\rangle + |3\rangle\,|011101\rangle)$$

- Occurrences de p : positions associées à un **registre de recherche égal à** $|0^M\rangle$ dans $|\Psi'_t\rangle$

CONSTRUCTION D'UN ORACLE

- Oracle O_p, pour $p = 110100$



- Profondeur : $\mathcal{O}(\lg M)$
- Portes : $\mathcal{O}(M)$

ÉBAUCHE DE L'ALGORITHME

1. Former la **superposition uniforme des sous-mots** pour t

$$|\Psi_t\rangle = \frac{1}{2}(|0\rangle|101101\rangle + |1\rangle|011010\rangle + |2\rangle|110100\rangle + |3\rangle|101001\rangle)$$

2. En utilisant O_p et l'algorithme de Grover, **amplifier l'amplitude** des états dont le registre de recherche correspond à p
3. **Mesurer** le registre de position pour obtenir une occurrence

---

L. C. L. Hollenberg. « Fast quantum search algorithms in protein sequence comparisons : Quantum bioinformatics ». In : *Physical Review E* 62.5 (nov. 2000), p. 7532-7535. doi : 10.1103/physreve.62.7532.

PLAN

CONSTRUCTION EFFICACE DE LA SUPERPOSITION

- À partir de $t = 101101001$ et $p = 110100$, comment **construire** $|\Psi_t\rangle$ **efficacement** ?

$$|\Psi_t\rangle = \frac{1}{2}(|0\rangle|101101\rangle + |1\rangle|011010\rangle + |2\rangle|110100\rangle + |3\rangle|101001\rangle)$$

- Besoin d'un temps $< \mathcal{O}(NM)$, sinon retour à l'algorithme naïf

P. Niroula et Y. Nam. « A quantum algorithm for string matching ». In : *npj Quantum Information* 7.1 (fév. 2021). doi : 10.1038/s41534-021-00369-3.

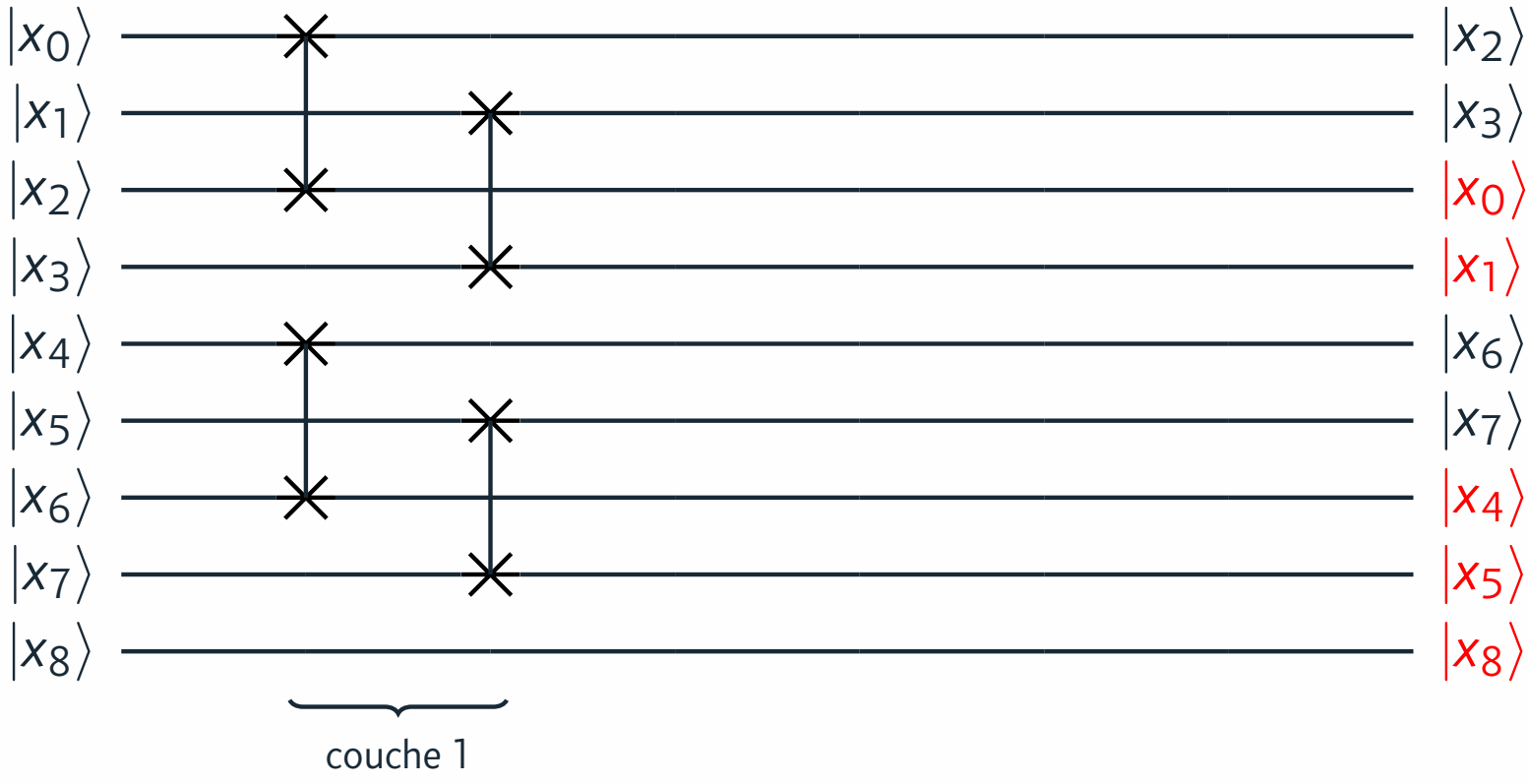
DÉCALAGE CYCLIQUE DU TEXTE

- $S_k^{[N]}$ **décale** ses N qubits d'entrée de k positions vers la gauche
  - Exemple : $S_2^{[9]} |101101001\rangle = |110100110\rangle$
  - En général : $S_k^{[N]} \bigotimes_{i=0}^{N-1} |x_i\rangle = \bigotimes_{i=0}^{N-1} |x_{(i+k) \bmod N}\rangle$

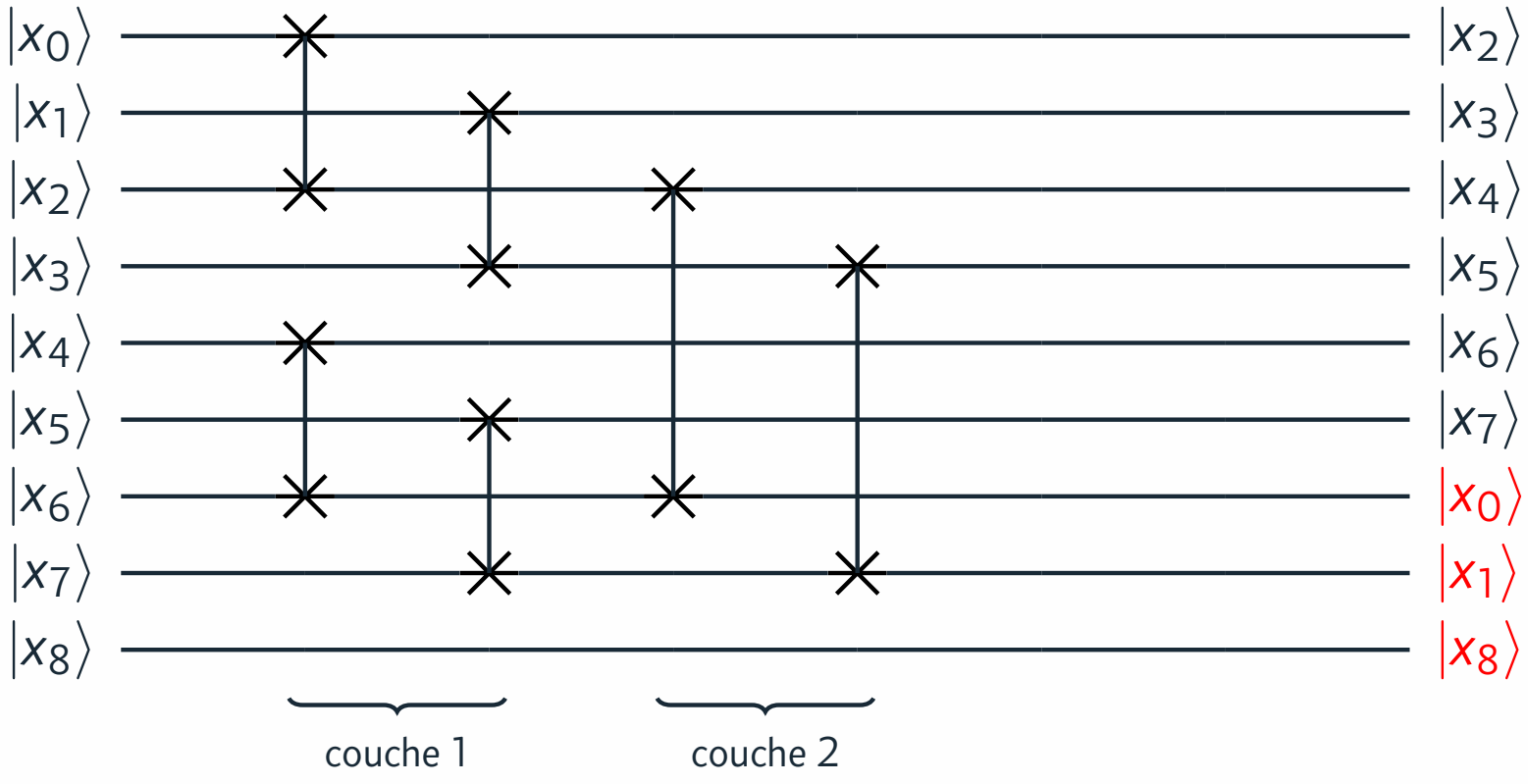
- **Composition :** $S_k^{[N]} S_{k'}^{[N]} = S_{k+k'}^{[N]}$
- $S_k^{[N]}$ peut être réalisée par un réseau de portes SWAP

CIRCUIT DE DÉCALAGE CYCLIQUE $s_2^{[9]}$

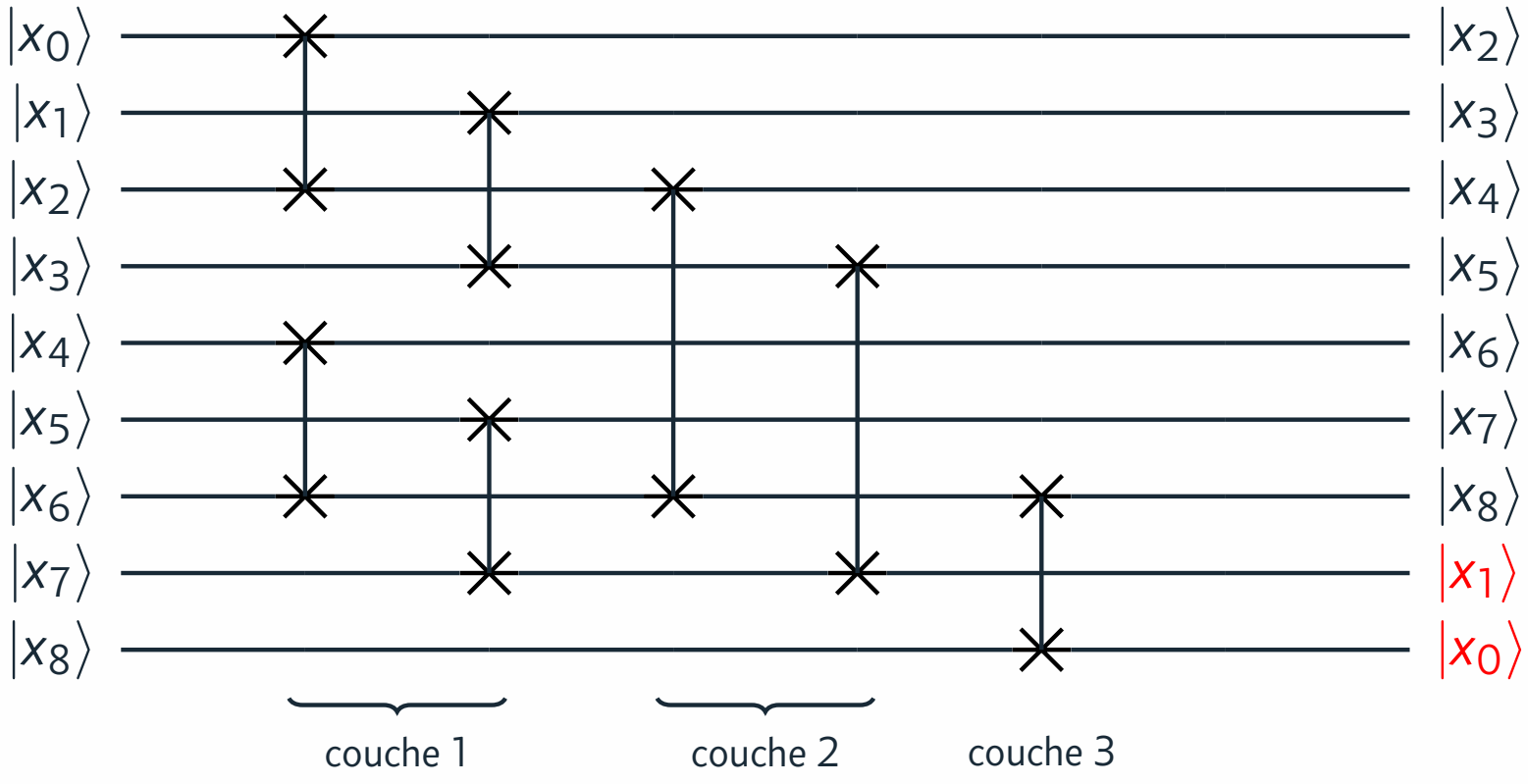
CIRCUIT DE DÉCALAGE CYCLIQUE $s_2^{[9]}$



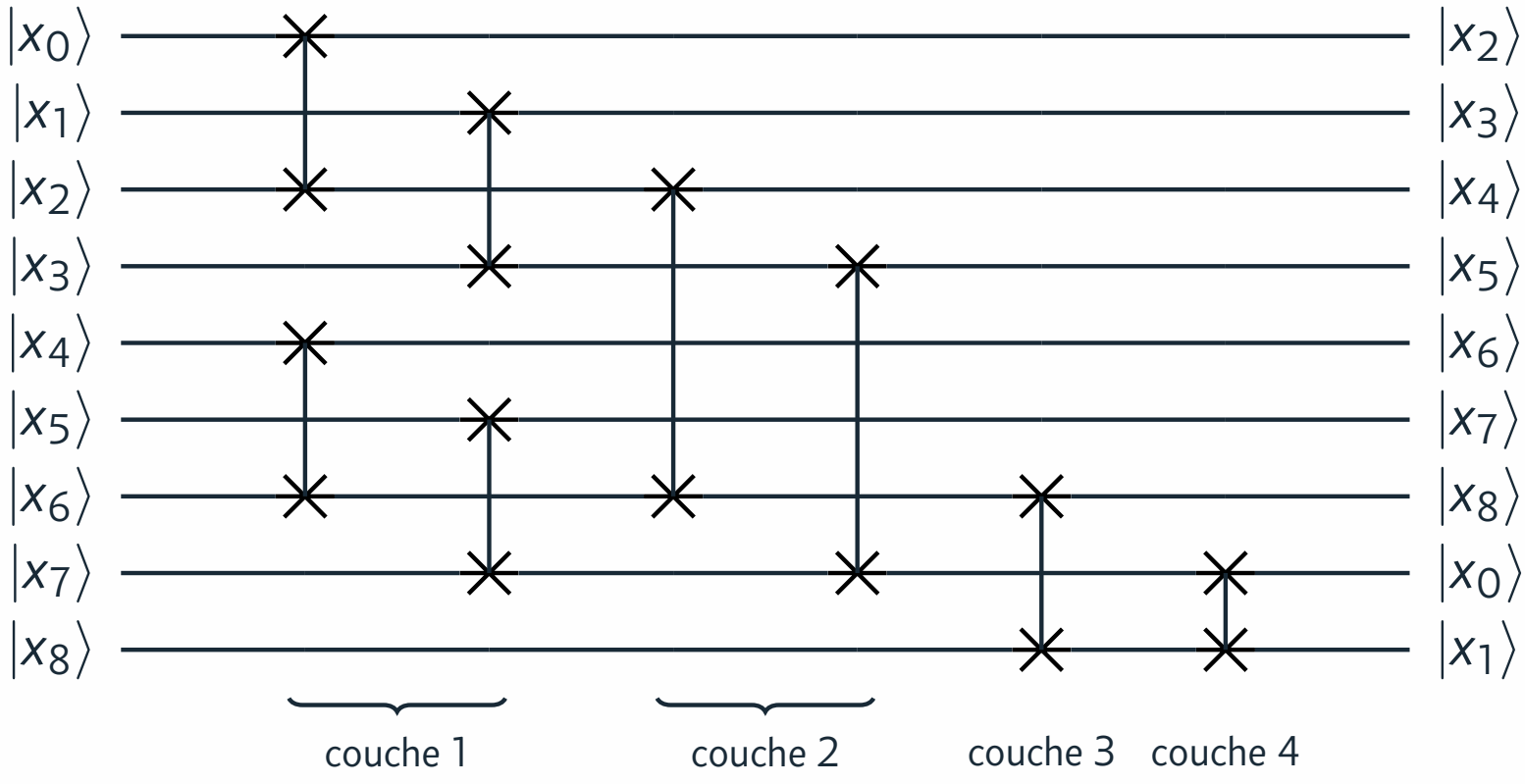
CIRCUIT DE DÉCALAGE CYCLIQUE $s_2^{[9]}$



CIRCUIT DE DÉCALAGE CYCLIQUE $s_2^{[9]}$



CIRCUIT DE DÉCALAGE CYCLIQUE $s_2^{[9]}$

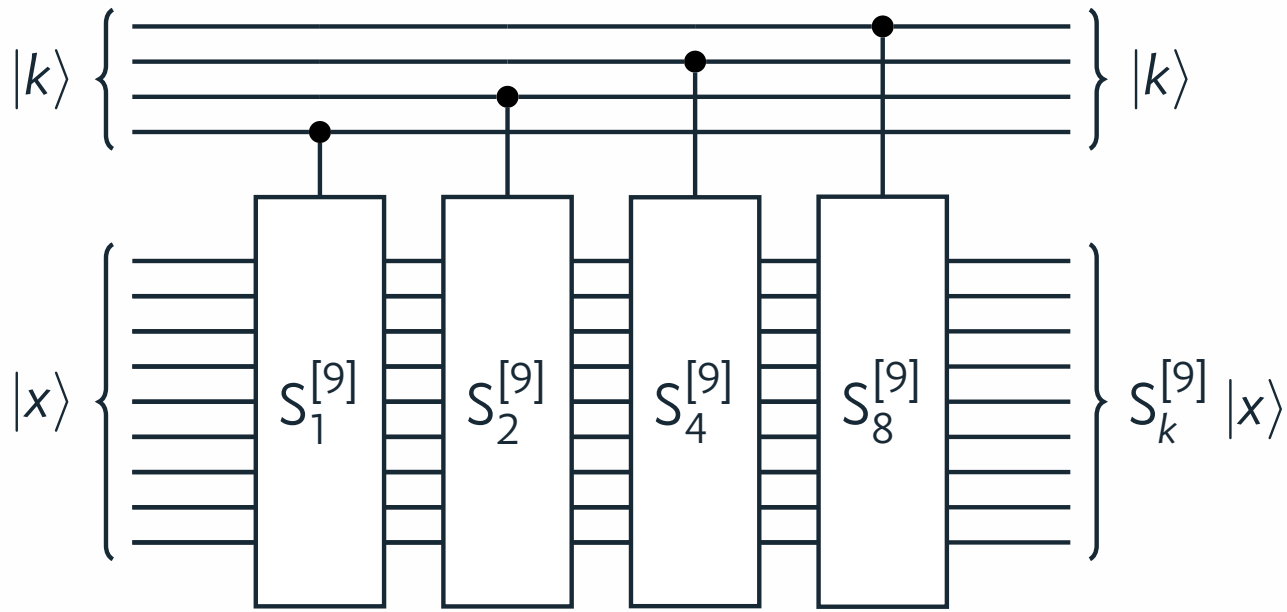


- Profondeur : $\mathcal{O}(\lg N)$
- Portes : $\mathcal{O}(N)$

DÉCALAGE CYCLIQUE CONTRÔLÉ

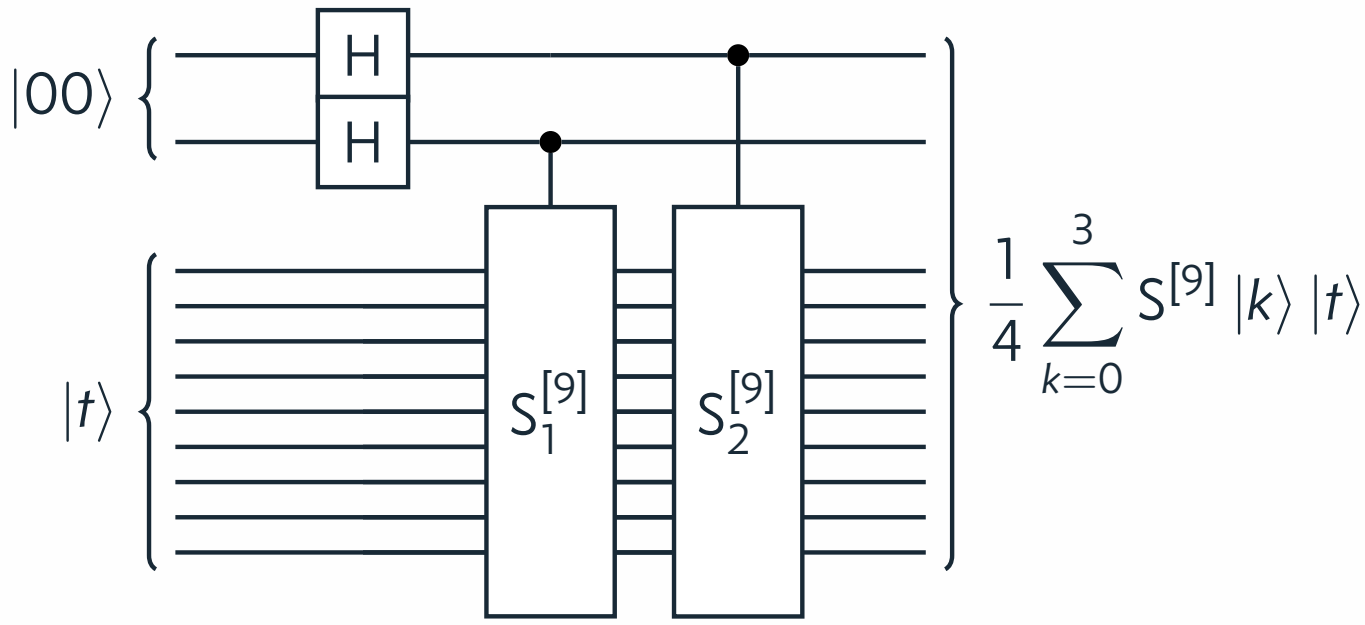
- $S^{[N]}$ décale ses N qubits d'entrée d'un nombre de positions égal à la **valeur d'un registre de position**
  - Exemple : $S^{[9]} |2\rangle |101101001\rangle = |2\rangle |110100110\rangle$
  - En général : $S^{[N]} |k\rangle |x\rangle = |k\rangle S_k^{[N]} |x\rangle$

- Encoder $|k\rangle$ en binaire et tirer parti de la composition des $S_k^{[N]}$



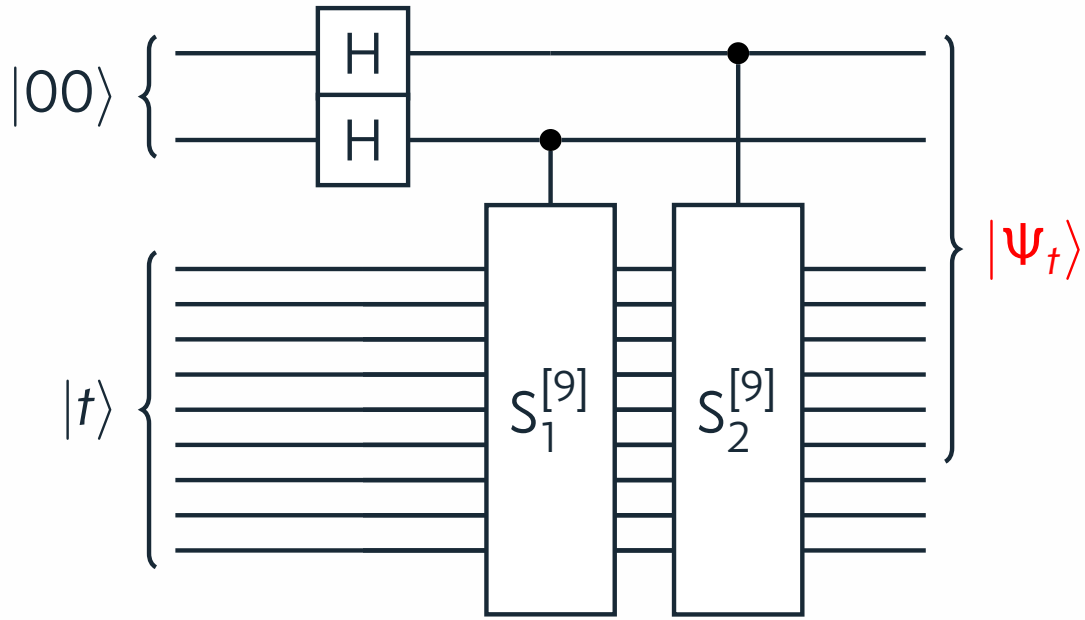
SUPERPOSITION UNIFORME DES DÉCALAGES

- Superposition uniforme sur le registre de position



SUPERPOSITION UNIFORME DES DÉCALAGES

- Superposition uniforme sur le registre de position



- **Construction de $|\Psi_t\rangle$!**
- Profondeur : $\mathcal{O}(\lg^2 N)$
- Portes : $\mathcal{O}(N \lg N)$

PLAN

RASSEMBLONS LES PIÈCES DU PUZZLE

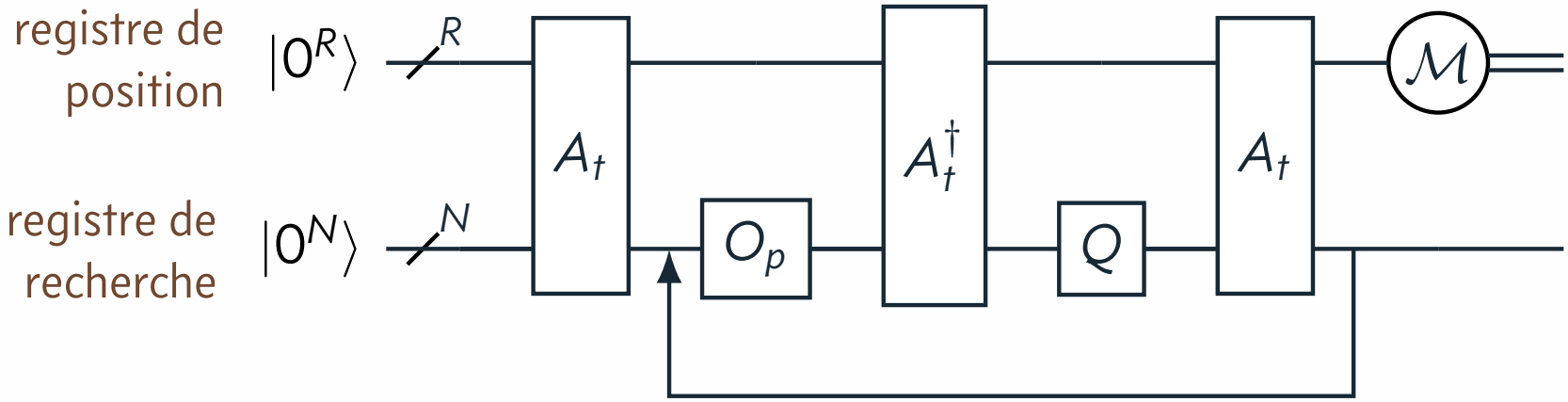
- **Algorithme** A_t : donne superposition uniforme des sous-mots de t
  - Qubits : $N + \lg(N - M + 1)$
  - Profondeur : $\mathcal{O}(\lg^2 N)$
  - Portes : $\mathcal{O}(N \lg N)$

- Exécution de A_t + mesure donne une occurrence avec probabilité $a = c/(N - M + 1)$ (où c est le nombre total d'occurrences)

RASSEMBLONS LES PIÈCES DU PUZZLE

- **Algorithme** A_t : donne superposition uniforme des sous-mots de t
  - Qubits : $N + \lg(N - M + 1)$
  - Profondeur : $\mathcal{O}(\lg^2 N)$
  - Portes : $\mathcal{O}(N \lg N)$
- Exécution de A_t + mesure donne une occurrence avec probabilité $a = c/(N - M + 1)$ (où c est le nombre total d'occurrences)
- **Oracle** O_p : inverse l'amplitude des états correspondant au motif p
  - Qubits : M
  - Profondeur : $\mathcal{O}(\lg M)$
  - Portes : $\mathcal{O}(M)$

ALGORITHME DE RECHERCHE DE MOTIF (1/2)



(Avec $R = \lg(N - M + 1)$ et $Q|x\rangle = (-1)^{[x \neq 0^n]}|x\rangle$.)

G. Brassard, P. Høyer, M. Mosca et A. Tapp. « Quantum amplitude amplification and estimation ». In : *Contemporary Mathematics* 305 (2002), p. 53-74. doi : 10.1090/conm/305/05215.

ALGORITHME DE RECHERCHE DE MOTIF (2/2)

- Si c est le nombre d'occurrences de p dans t
- Nombre de répétitions de l'amplification (espérance) :

$$1/\sqrt{a} = \mathcal{O}\left(\sqrt{N/c}\right)$$

G. Brassard, P. Høyer, M. Mosca et A. Tapp. « Quantum amplitude amplification and estimation ». In : *Contemporary Mathematics* 305 (2002), p. 53-74. doi : 10.1090/conm/305/05215.

ALGORITHME DE RECHERCHE DE MOTIF (2/2)

- Si c est le nombre d'occurrences de p dans t
- Nombre de répétitions de l'amplification (espérance) :

$$1/\sqrt{a} = \mathcal{O}\left(\sqrt{N/c}\right)$$

- Profondeur du circuit (espérance) :

$$\mathcal{O}\left(\sqrt{N/c}\,(\lg^2 N + \lg M)\right)$$

- Portes du circuit (espérance) :

$$\mathcal{O}\left(\sqrt{N/c}\,(N \lg N + M)\right)$$

G. Brassard, P. Høyer, M. Mosca et A. Tapp. « Quantum amplitude amplification and estimation ». In : *Contemporary Mathematics* 305 (2002), p. 53-74. doi : 10 . 1090 / conm/305/05215.

EXTENSIONS DE L'ORACLE

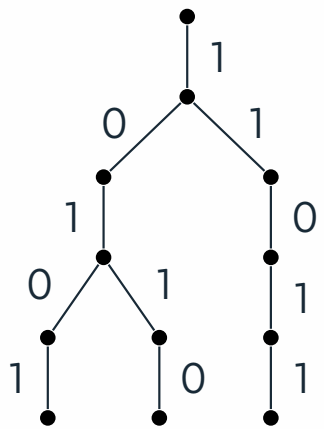
- N'est pas limité à des motifs fixes : **tout motif pouvant être reconnu par un circuit** peut être recherché

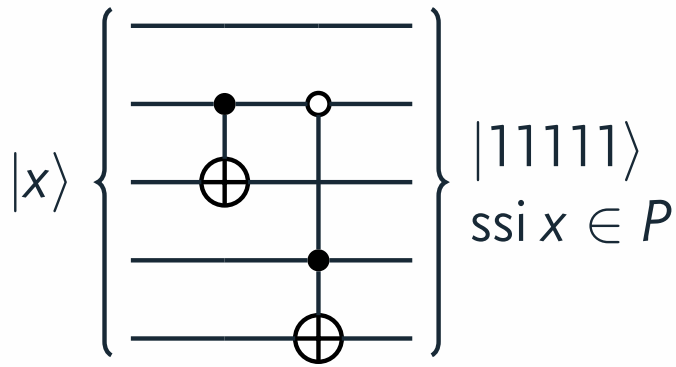
1. Recherche avec **caractères jokers** (ex. : $p = 01??001?$)
   - Ignorer les positions qu'on veut laisser libre

EXTENSIONS DE L'ORACLE

- N'est pas limité à des motifs fixes : **tout motif pouvant être reconnu par un circuit** peut être recherché

1. Recherche avec **caractères jokers** (ex. : $p = 01??001?$)
   - Ignorer les positions qu'on veut laisser libre
2. Recherche simultanée de **motifs multiples**
   - Exemple : $P = \{11011, 10110, 10101\}$





PLAN

COMPARAISON AUX ALGORITHMES CLASSIQUES

- Recherche exacte d'**un motif**

Knuth–Morris–Pratt	$\Theta(N)$
Boyer–Moore	$\mathcal{O}(N)$ en moyenne
Quantique	$\mathcal{O}(\sqrt{N/c}\,(\lg^2 N + \lg M))$ en moyenne

- Recherche exacte **simultanée de K motifs**

Aho–Corasick	$\Theta(N + KM + c)$
Rabin–Karp	$\mathcal{O}(N + KM)$ en moyenne
Quantique	?

ET EN PRATIQUE ?

- Sur des **simulateurs** (Qiskit Aer) : fonctionne comme attendu
- Sur un **processeur quantique** à 5 qubits (IBM Belem) :
  - *Ne fonctionne pas*
  - Circuit trop complexe : trop de bruit

- Peu de chance d'être utile à court terme !
  - Nécessite $\mathcal{O}(N)$ qubits
  - Avantage pratique (potentiel) seulement sur des **longs textes**

PERSPECTIVES

- D'autres approches connexes
  - H. Ramesh et V. Vinay. « String matching in $\tilde{\mathcal{O}}(\sqrt{n}+\sqrt{m})$ quantum time ». In : *Journal of Discrete Algorithms* 1.1 (fév. 2003), p. 103-110. doi : 10.1016/s1570-8667(03)00010-8
  - P. Mateus et Y. Omar. « Quantum pattern matching ». In : (août 2005). arXiv : quant-ph/0508237 [quant-ph]

- Pistes de recherche
  - **Réduire** le nombre de **qubits** utilisés pour l'initialisation
  - **Réduire** le nombre d'opérations d'**échange** de qubits
  - Recherche avec **indels** (application des algorithmes vectoriels ?)