Quantum Multi-string Matching

Allen Liu Kevin Tong

University of Waterloo

ECE 405C Winter 2025
April 2, 2025

Presentation Overview

String Matching

- Text $S[0...n-1]$
- Pattern (or key) $P[0...m-1]$
- Strings from alphabet Σ
- Find set of indices i such that $S[i..i+m] = P$

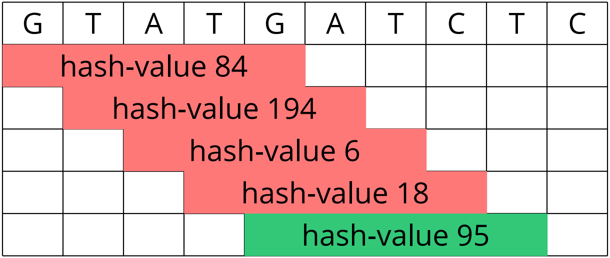
Example:

- T = `"GTAT GATC TC"` (ignore spaces)
- P_1 = `"ATCT"`
- P_2 = `"TGAT"`
- P_3 = `"ACCC"`
- P_1 matches at 5, P_2 matches at 3, P_3 has no matches

- Multi-string matching: search $P_1, P_2, \ldots$ in S simultaneously

Classical Algorithms

- Brute Force $\longrightarrow O(mn)$
- Boyer-Moore $\longrightarrow O(n+m)$, $O(mn)$ worst case
- Knuth-Morris-Pratt $\longrightarrow O(n+m)$ worst case
- Suffix Tree/Array $\longrightarrow O(m)/O(n \log n)$ + preproc
- Karp-Rabin - rolling hash $\longrightarrow O(n+m)$ expected



- Polynomial Matching $\longrightarrow O(n \log n)$

Polynomial Matching

Calculate a fingerprint for P and every m character sequence in S; matching fingerprints suggest pattern match!

$$A \mapsto -3$$
$$C \mapsto 5$$
$$G \mapsto -7$$
$$T \mapsto 11$$

$$\begin{aligned}
\texttt{"ATCT"} &\longrightarrow \mathrm{T}x^3 + \mathrm{C}x^2 + \mathrm{T}x + \mathrm{A} \\
&\longrightarrow 11x^3 + 5x^2 + 11x - 3 \\
\texttt{"GTAT GATC TC"} &\longrightarrow -7x^{15} + 11x^{14} - 3x^{13} + \ldots + 11x^7 + 5x^6
\end{aligned}$$

Note that P encoding is reversed (similar to convolution)

Polynomial Matching

Given polynomials

$$P(x) = \sum_{i=0}^{m} a_i x^i, S(x) = \sum_{j=0}^{n} b_j x^j$$

then

$$R(x) = \sum_{k=0}^{m+n} c_k x^k$$

where

$$c_k = \sum_{i=0}^{k} a_i b_{k-i}, \quad \text{for } 0 \leq k \leq m+n,$$

with the convention that $a_i = 0$ for $i > m$ and $b_j = 0$ for $j > n$.

Polynomial Matching

Coefficients of $R(x)$ correspond to "dot products" between substrings.

$$P(x) = 11x^3 + 5x^2 + 11x - 3$$

$$S(x) = -7x^{15} + 11x^{14} - 3x^{13} + \ldots + 11x^7 + 5x^6$$

$$R(x) = \ldots + 71x^8 + 0x^9 + 276x^{10} + 98x^{11} - 4x^{12} + \ldots$$

Degrees map to index:

15 yields index 0

14 yields index 1

...

11 yields index 4

10 yields index 5

9 yields index 6

Polynomial Matching

Coefficients of $R(x)$ correspond to "dot products" between substrings.

$$P(x) = 11x^3 + 5x^2 + 11x - 3$$
$$S(x) = -7x^{15} + 11x^{14} - 3x^{13} + \ldots + 11x^7 + 5x^6$$
$$R(x) = \ldots + 71x^8 + 0x^9 + 276x^{10} + 98x^{11} - 4x^{12} + \ldots$$

Notice that $\|P\|^2 = 11^2 + 5^2 + 11^2 + (-3)^2 = 276$

We get exact fingerprint match for single patterns. Let's extend to multiple patterns...

Polynomial Matching

Add patterns together:

$$P(x) = P_1(x) + P_2(x)$$
$$S(x) = -7x^{15} + 11x^{14} - 3x^{13} + \ldots + 11x^7 + 5x^6$$
$$R(x) = \ldots + 94x^8 + 240x^9 + 272x^{10} + 64x^{11} + 296x^{12} - 60x^{13} \ldots$$

No more exact matches, but higher values = likelier index.

Polynomial Multiplication

Multiple algorithms:

- Naïve polynomial multiplication $\longrightarrow O(n^2)$
- Karatsuba Algorithm $\longrightarrow O(n^{\log_3 2}) = O(n^{1.59})$
- FFT $\longrightarrow O(n \log n)$

Point Value Form

Theorem (Lagrange Interpolation)

For any n points $(x_i, y_i) \in \mathbb{R}^2$ with no two x_i the same, there exists a unique polynomial $A(x)$ of degree at most $n-1$ that interpolates these points.

Given polynomials $A(x)$ and $B(x)$ of degree n, we can represent them as

$$A : \{(x_0, A(x_0)), (x_1, A(x_1)), \ldots, (x_{n-1}, A(x_{n-1}))\}$$
$$B : \{(x_0, B(x_0)), (x_1, B(x_1)), \ldots, (x_{n-1}, B(x_{n-1}))\}$$

Then their product $C(x) = A(x)B(x)$ is

$$C : \{(x_0, A(x_0)B(x_0)), (x_1, A(x_1)B(x_1)), \ldots, (x_n, A(x_{n-1})B(x_{n-1}))\}$$

Polynomial Multiplication

Algorithm idea:

1. Evaluate $A(x)$ and $B(x)$ at n points
   - $O(n)$ multiplications per point
   - $O(n)$ points
   - $O(n^2)$ overall, no speedup
2. Multiply element-wise
3. Interpolate to find $C(x)$ coefficients

But this works for any n inputs. What if we try the n^{th} roots of unity?

$$\hat{a}_k = A(\omega_n^k) = \sum_{j=0}^{n} a_j e^{\frac{2\pi ijk}{n}}$$

for $0 \leq k \leq n-1$. This is DFT!

Polynomial Multiplication with FFT

1. Apply FFT to the coefficients of $A(x)$ and $B(x)$
   With $\vec{a} = [a_0\ a_1 \dots a_{n-1}]^T$ and $\vec{b} = [b_0\ b_1 \dots b_{n-1}]^T$, then

$$\vec{\alpha} = \text{FFT}(\vec{a}) \qquad O(n \log n)$$

$$\vec{\beta} = \text{FFT}(\vec{b}) \qquad O(n \log n)$$

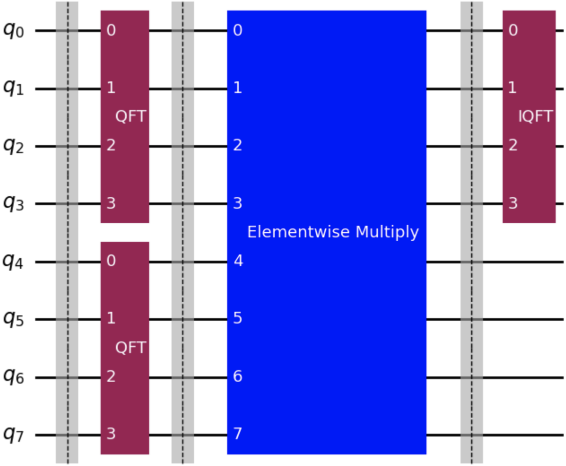
2. Perform the Hadamard (element-wise) product of coefficients

$$\begin{aligned} \vec{\gamma} &= \vec{\alpha} \odot \vec{\beta} \\ &= [\alpha_0\beta_0\ \ \alpha_1\beta_1 \dots \alpha_{n-1}\beta_{n-1}]^T \qquad O(n) \end{aligned}$$

3. Apply Inverse FFT

$$\vec{c} = \text{IFFT}(\vec{\gamma}) \qquad O(n \log n)$$

Polynomial Multiplication with QFT



Polynomial multiplication for degree up to $n = 2^4 - 1 = 15$

Elementwise Multiply

Is there such a unitary operator? Define

$$\begin{aligned} |\alpha\rangle &= \alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle \\ |\beta\rangle &= \beta_0|00\rangle + \beta_1|01\rangle + \beta_2|10\rangle + \beta_3|11\rangle \\ |\alpha \odot \beta\rangle &= C\left(\alpha_0\beta_0|00\rangle + \alpha_1\beta_1|01\rangle + \alpha_2\beta_2|10\rangle + \alpha_3\beta_3|11\rangle\right) \end{aligned}$$

for some normalization constant C

$$U|x\rangle|y\rangle = U|x\rangle|x \odot y\rangle$$

If $|y\rangle = \frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right) \longrightarrow$ violates no-cloning theorem

Elementwise Multiply

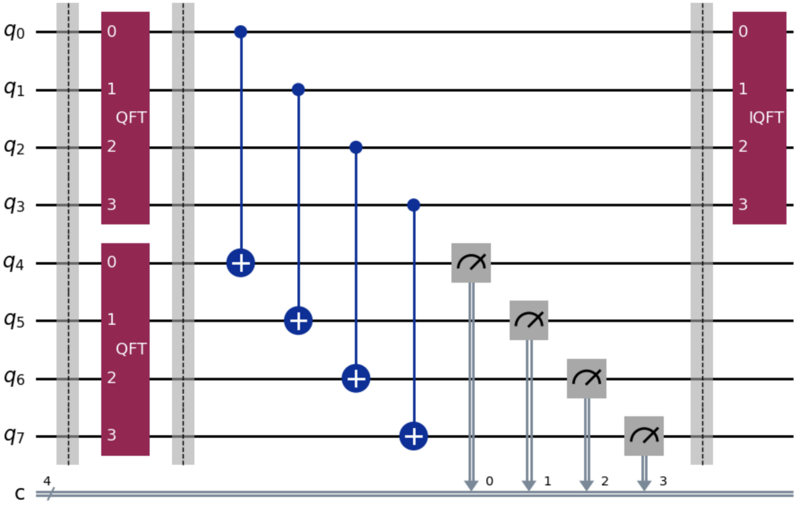
Allow probabilistic operator:

$$
\begin{aligned}
|\alpha\rangle|\beta\rangle = \alpha_0\beta_0|0000\rangle + \alpha_0\beta_1|0001\rangle + \dots \\
\alpha_1\beta_0|0100\rangle + \alpha_1\beta 1|0101\rangle + \dots \\
\dots + \alpha_2\beta_2|1010\rangle + \dots \\
\dots + \alpha_3\beta_3|1111\rangle + \dots
\end{aligned}
$$

Want when first two bits = last two bits.

Use CNOT to test for equality.

Elementwise Multiply



Elementwise Multiply

$$\begin{aligned}|\alpha\rangle|\beta\rangle = \alpha_0\beta_0|0000\rangle + \alpha_0\beta_1|0001\rangle + \dots \\ \alpha_1\beta_0|0100\rangle + \alpha_1\beta 1|0101\rangle + \dots \\ \dots + \alpha_2\beta_2|1010\rangle + \dots \\ \dots + \alpha_3\beta_3|1111\rangle + \dots\end{aligned}$$

$$\begin{aligned}\text{CNOT}_{0,2}\text{CNOT}_{1,3}\ |\alpha\rangle|\beta\rangle = \alpha_0\beta_0|0000\rangle + \alpha_0\beta_1|0001\rangle + \dots \\ \alpha_1\beta_0|0101\rangle + \alpha_1\beta 1|0100\rangle + \dots \\ \dots + \alpha_2\beta_2|1000\rangle + \dots \\ \dots + \alpha_3\beta_3|1100\rangle + \dots\end{aligned}$$

Measure the last two qubits, success when $|00\rangle \longrightarrow$ elementwise multiply in first two qubits!

Algorithmic Analysis

Let n be the number of qubits needed to represent the polynomial produced by multiplying $A(x)$ and $B(x)$.

Initial QFT: $O(n^2)$ gates.

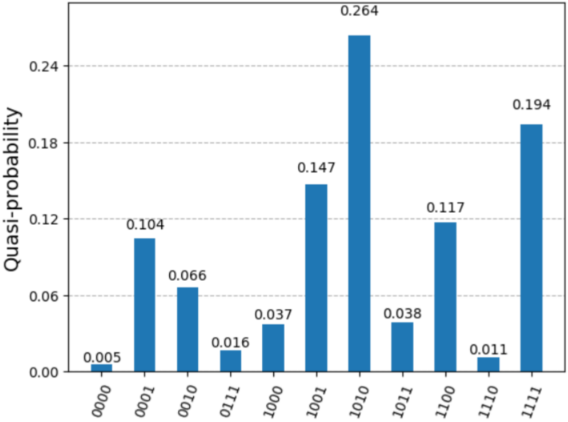
Element-wise multiplication: $O(n)$ CNOT gates, but requries an expected $O(2^n)$ measurements for a result where the last n qubits are all zero.

Final IQFT: $O(n^2)$ gates.

So the total cost is $O(n^2)$ gates, but expected $O(2^n)$ shots.

Qiskit Results

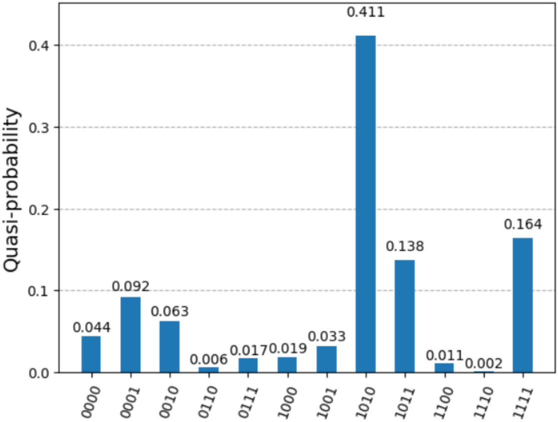
Text: `"GTAT GATC TC"` Key: `"ATCT"`



Notice that the highest quasi-probability is for state 1010_2!

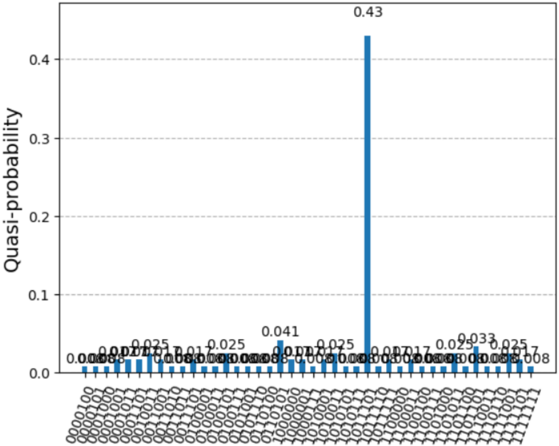
Multi-String Matching

- Text: `"GTAT GATC TC"`
- Key 1: `"ATCT"`
- Key 2: `"TGAT"`
- Key 3: `"ACCC"`



More Qiskit Results

- Text: 90 randomly generated characters $\in \{A, T, C, G\}$
- Key: 52 characters



Next Steps

1. Sometimes patterns destructively interfere $\longrightarrow$ better combination method than addition?
2. Encoding uses primes $\longrightarrow$ better interference patterns?
3. Wasteful elementwise multiply $\longrightarrow$ reconstruct data using shifted multiply?

   E.g. measure $|01\rangle$:

$$|\psi\rangle = \alpha_0\beta_1|0001\rangle + \alpha_1\beta_0|0101\rangle + \alpha_2\beta_3|1001\rangle + \alpha_3\beta_2|1101\rangle$$

References

James W. Cooley, John W. Tukey (1965)
An Algorithm for the Machine Calculation of Complex Fourier Series
https://www.ams.org/journals/mcom/1965-19-090/S0025-5718-1965-0178586-1/S0025-5718-1965-0178586-1.pdf

Richard M. Karp & Michael O. Rabin (1987)
Efficient Randomized Pattern-matching Algorithms
https://doi.org/10.1147/rd.312.0249

Mehdi Ramezani, Morteza Nikaeen, Farnaz Farman, Seyed Mahmoud Ashrafi and Alireza Bahrampour (2023)
Quantum Multiplication Algorithm Based on the Convolution Theorem
https://arxiv.org/pdf/2306.08473