Presentation Script

Response to Ramesh & Vinay, (2003)

String Matching in $\tilde{O}(\sqrt{n}+\sqrt{m})$ Quantum Time

Matthew Evans, Ariz Siddiqui, Nathan Puskuri

April 27, 2025

Deterministic Sampling

Imagine you have a pattern of length m that you want to match against a block of text of length roughly $m/2$. Naively there are $m/2$ possible alignments to check, and even a quantum search over all of them costs $sqrt(m)$ time each. Deterministic sampling is a clever trick (originally due to Vishkin) that lets us rule out all but one candidate alignment by examining only an $O(\log m)$-sized set of "sample" of pattern positions.

Here's the key idea for aperiodic patterns: imagine laying down all $m/2$ shifted copies of your pattern above the text block. Because the pattern is aperiodic, there must be at least one column (pattern index) in which two of these copies differ. If you pick that column and look at the two distinct characters there, then at most one of the copies can match your text at that column. Eliminating the other half of the copies—and repeating this process—quickly isolates a single surviving copy.

Steps

1. Initialize
    - Label copies $1, 2, \ldots, m/2$
    - Let the sample set S be empty
2. Repeat $O(\log m)$ times
    - Find a column c where at least two surviving copies disagree.
    - Add c to S.
    - Pick the minority character $\mathcal{X}$ at c, and discard any copy whose character at $c \neq \mathcal{X}$.
    - This halfs the number of "survivors" each round.
3. Result

- You end up with exactly one copy $p^\star$ and a sample set S of size $O(\log m)$.
- Any alignment that matches the text at *any* positions in S must be that copy; you've reduced $m/2$ possibilities to one candidate.

Efficiency

Classical sampling takes $O(m)$ time to scan all columns each round; quantumly we can replace those scans with Grover-search and quantum minimum-finding to achieve an overall preprocessing time of $\widetilde{O}()\sqrt{m}\log^2 m$.

Integration

In the full quantum string-matching algorithm, you run this sampling once on the pattern, reuse the same S for every text block, and only need one expensive $\sqrt{m}$-time check per block—yielding a sublinear quantum speed-up.

Conclusion

Deterministic sampling thus gives a small, carefully chosen set of pattern positions that rules out almost all bad alignments. It's the linchpin that turns a naive *sqrtnm* quantum approach into an $\widetilde{O}(\sqrt{n}+\sqrt{m})$ algorithm.

DH96 - Minimum Finding Oracle

Concept & Intuition

1. **Random Starting Point** Pick an index k uniformly at random.
2. **Grover Search for Improvement** Use Grover's subroutine to search for any index i such that
$$\text{item}[i] < \text{item}[k].$$
   - If none is found, k is the minimum.
   - If one is found, set $k \leftarrow i$ (you've found a strictly smaller item).
3. **Iterate** Repeat the Grover search/update step a small constant number of times until no better index appears.
4. **Geometric Speed-Up** Each search costs $O(\sqrt{n/t})$ where t is the number of strictly smaller elements (initially up to $n-1$), and the sequence of improvements converges in $O(\sqrt{n})$ total time.

Why It Matters

- **Quantum vs. Classical:** Reduces the "find-min" cost from linear to square-root time.
- **General-Purpose Primitive:** You can wrap any Boolean test ("is this element in our candidate set?") and efficiently extract the smallest satisfying index.
- **Key Uses in Ramesh–Vinay:**
  - Identifying the leftmost text block that might contain a match.
  - Finding the leftmost or rightmost surviving pattern copy when building the deterministic sample.

Wrap-up

The DH96 minimum-finding oracle turns a linear scan into an $O(\sqrt{n})$ quantum subroutine. By interleaving random picks with Grover searches for improvements, it guarantees a high-probability success in square-root time. This powerful tool underlies every step in the paper where we need the "smallest" or "first" index—crucial for block selection, sample building, and periodic-pattern handling.