

Implementing QVMP using QROM

Elton Pinto

April 11, 2022

Main contributions

- Working implementation of QVMP in Qiskit
- Circuit metrics (gate count, circuit depth)
- Transpilation and simulation times

Motivation

- Grover search: popular quantum search algorithm
- Depends on a black-box oracle to perform the search
- Offers quadratic speedup over classical linear search with a runtime of $O(\sqrt{N})$
- Related: Amplitude amplification, a generalization of Grover search

Motivation (contd)

- Core of Grover search straightforward to implement
- Main challenge: encoding the oracle as a quantum circuit
 - Debugging oracles is tricky due to non-determinism
 - How to verify correctness?

Goal

Implement QVMP to better understand these challenges and investigate enhancements

QVMP

- Quantum Verification of Matrix Products
- Given $n \times n$ matrices A , B and C , check if $AB = C$
- Two quantum algorithms:
 - Grover search based: $O(n^{\frac{7}{4}})$
 - Quantum random walk based: $O(n^{\frac{5}{3}})$

QVMP Algorithm

Algorithm 1 Quantum VMP using Grover Search

Input: $n \times n$ matrices A, B, C

Output: 1 if $AB = C$ and 0 otherwise

Procedure:

1. Partition B and C into sub-matrices of size $n \times \sqrt{n}$
 2. Perform amplitude amplification for $n^{\frac{1}{4}}$ iterations using this subroutine:
 - 2.1 Pick a random vector x of size \sqrt{n}
 - 2.2 Classically compute $y = B_i x$ and $z = C_i x$
 - 2.3 Using Grover search with \sqrt{n} iterations, find a row of index j such that $(Ay \neq z)_j$
 3. XOR the sub-results
-

QVMP Implementation

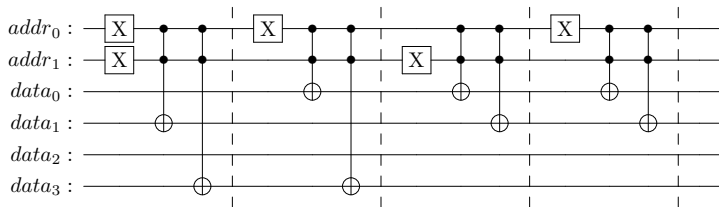
```
1  # QVMP oracle described using a classical function
2
3  def find_row_mismatch(A, y, z):
4      z_prime = A * y
5      for j, value in enumerate(z_prime):
6          if value != z[j]:
7              return j
8      return -1
```

- The above snippet is encoded as a quantum circuit and constitutes the oracle
- QROM is used for efficiently encoding the matrix
- Out-of-place inner product performs the row-vector multiplication

QROM - Quantum Read-only Memory

- Encodes an $n \times m$ binary matrix using only $n + \log_2(n)$ qubits
- Outputs the value of the j th row indexed using address qubits
- Can use superposition to extract multiple rows

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$



(a) QROM encoding for a 4×4 matrix

Inner product

- Computes the inner product between two binary vectors using $2n + 1$ qubits
- Outputs the result in a separate qubit

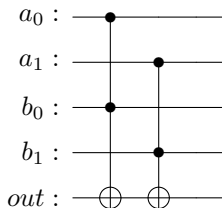


Figure 2: Inner product circuit for a $2D$ vectors

QVMP oracle

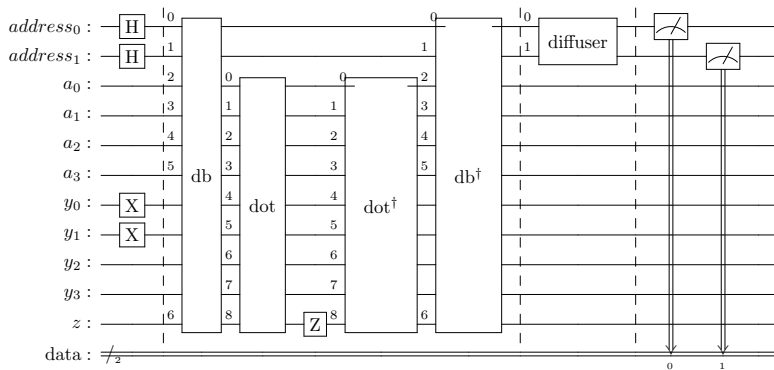


Figure 3: QVMP oracle for a 4×4 matrix A performing one iteration

Sample execution

- **Input:** 16×16 matrix A and two vectors y and z with $(Ay \neq z)_j$ for $j \in \{0, 5, 4\}$

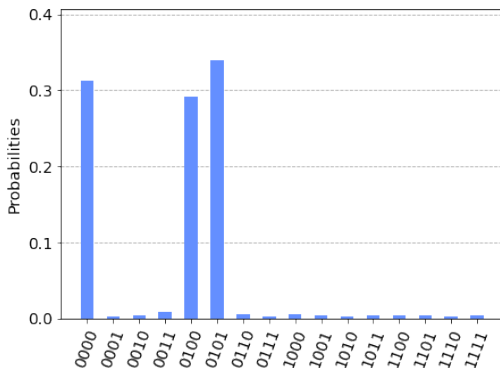


Figure 4: Probability of measuring the row-index j after running the QVMP oracle

Future work

Automated synthesis of oracles

Extend existing work on reversible compilers to support higher-level programming constructs like lists, records, multi-dimensional arrays

- Encoding classical decision functions into quantum circuits is error-prone and cumbersome
- Previous work (REVS, Quipper) have shown that we can automate classical to reversible compilation

```
1  (* Example program describing the QVMP oracle *)
2
3  [@@oracle]
4  let find_row_mismatch a y z =
5    find_idx (fun idx value -> value <> z[idx]) (a * y)
```

Future work (contd)

Better encoding of matrices

Investigate more efficient encodings of matrices and related operations

- n -qubit quantum system can encode a total of 2^n states
- QROM is still linear in space complexity
- Alternative approach encodes the entries of the matrix as amplitudes of the quantum system but is harder to work with

Future work (contd)

Transpilation time bottlenecks

Investigate bottle-necks in transpilation

- Transpilation becomes exponentially slow as the number of qubits increases
- Makes it harder to scale and test circuit implementations
- cursory investigation into Qiskit's transpile method revealed that SWAPs may be the bottleneck

End of talk

Questions?