An Implementation of the Quantum Verification of Matrix Products Algorithm

ISCA 2022 Submission **#1** – Confidential Draft – Do NOT Distribute!!

***Abstract*—We present a space-efficient implementation of the quantum verification of matrix products (QVMP) algorithm and demonstrate its functionality by running it on the Aer simulator with two simulation methods: statevector and matrix product state (MPS). We report circuit metrics (gate count, qubit count, circuit depth), transpilation time, simulation time, and a proof of Grover oracle correctness. Our study concludes that while QVMP can be simulated on moderately sized inputs, it cannot scale to a degree where we can observe any quantum advantage on current quantum hardware due to circuit depth and qubit count constraints. Further, the choice of simulation method has a noticeable impact on the size of the transpiled circuit which slows down development.**

I. Introduction

Grover's algorithm depends on a user-provided oracle that acts as a decision function over the input space. The oracle typically does not depend on quantum effects and can be programmed using classical operators. However, to use such an oracle in Grover's algorithm, we need to encode it as a quantum circuit. Programming reversible oracles can be tricky. It is difficult to reason about entanglement, non-trivial to debug, and cumbersome to formally verify correctness using machine-checked proofs.

REVS [3] and Quipper [6] have presented compilers that can convert oracles written in a high-level classical language to reversible circuits using variants of Bennett's method. To our knowledge, however, their work does not adequately address the space of compiling higher-level data structures like records, lists, and multi-dimensional arrays. There are interesting design choices to be made which can have an impact on the size and efficiency of the resulting circuit.

In this study we demonstrate one such design choice by presenting an implementation of the Grover search based version of QVMP [2]. This algorithm runs in $O(n^{\frac{7}{4}})$ time and improves upon the optimal classical bound provided by Freivalds [5]. Our implementation makes use of Quantum Read-Only Memory (QROM) [4] to efficiently encode the inputs, resulting in a quadratic improvement in space-efficiency ($O(n+m)$ qubits).

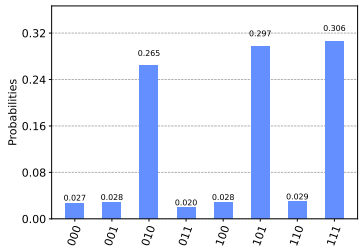
II. Implementation

The algorithm we implemented is summarized in Alg. 1. Fig. 3 shows an example circuit that performs one iteration of the Grover operator used in step 2.c. The sub-circuits used are QROM (denoted by *db*), out-of-place inner product (denoted by *dot*), and diffuser.

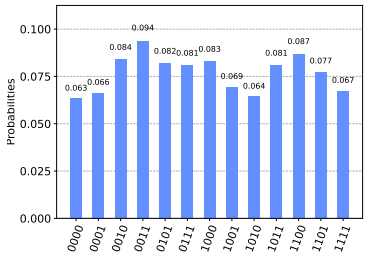
Algorithm 1 Quantum VMP using Grover Search [1]

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:
   a) Pick a random vector x of size $\sqrt{n}$
   b) Classically compute $y = B_i x$ and $z = C_i x$
   c) Using Grover search with $\sqrt{n}$ iterations, find a row of index j such that $(Ay \neq z)_j$
3) XOR the sub-results



(a) **Success**: 8×8 matrix with $(Ay \neq z)_j$ for $j \in \{2, 5, 7\}$



(b) **Failure**: 16×16 matrix with $(Ay \neq z)_j$ for $j \in \{1, 4, 5\}$

Fig. 1: Histogram showing the probability distribution of measuring the address qubits of the QVMP circuit

III. Results

A. Functionality

Fig. 1a demonstrates that our implementation is able to correctly find the row indices where there is a mismatch. Fig. 1b demonstrates a failure case that happens when the number of Grover iterations prescribed by the algorithm is not a multiple of the period, which we define to be the number of oscillations between the Grover-optimal number of iterations.

Dimension	Mismatches	ccx	cx	x	h	z	u1	u2	u3	Grover iterations	Depth	Qubits	Total gates
(4, 4)	1	30	1	11	2	1	2	2	0	1	44	11	49
(16, 8)	2	32	10108	69	284	2	8460	2536	3	2	16993	21	21494
(32, 8)	1	64	91408	272	997	4	91377	1056	8	4	150375	22	185186
(32, 32)	3	128	185912	151	2025	2	185897	2052	4	2	303901	70	376171
(64, 16)	2	128	786960	538	4190	4	786948	4232	5	4	1300351	39	1583005
(64, 64)	3	384	2329596	432	12396	3	2329587	12426	4	3	3843672	135	4684828

(a) MPS

Dimension	Mismatches	ccx	cx	x	h	z	u1	u2	u3	Grover iterations	Depth	Qubits	Total gates
(4, 4)	1	30	1	11	2	1	2	2	0	1	44	11	49
(16, 4)	2	16	0	75	4	2	3	12	1	2	261	13	113
(16, 8)	2	32	0	76	4	2	3	12	1	2	385	21	130
(32, 4)	2	24	0	208	5	3	4	24	2	3	684	14	270
(64, 8)	3	48	0	405	6	3	4	30	2	3	2040	23	498
(64, 8)	1	96	0	801	6	6	7	60	5	6	4077	23	981

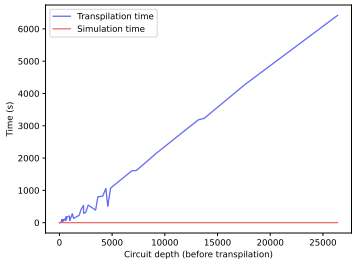
(b) Statevector

TABLE I: Circuit metrics for MPS and statevector simulation methods on select dimensions. Depth and total gates were measured after transpilation.

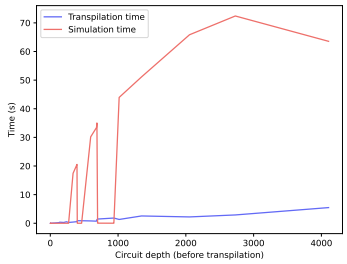
B. Circuit Metrics

We report circuit metrics (gate count, circuit depth, qubit count) in Table I measured with the MPS and statevector methods for the Grover search portion of QVMP (step 2.c in Alg. 1). The metrics are reported for circuits that used the optimal number of Grover iterations to achieve a successful output. When using the statevector method we see a small decrease in the circuit depth. Using MPS, on the other hand, results in two to three orders of magnitude more gate count and circuit depth.

C. Transpilation and Simulation



(a) MPS



(b) Statevector

Fig. 2: Circuit depth vs Transpilation/Simulation time

Fig. 2 showcases how transpilation time and simulation time change as the circuit depth increases. MPS seems to be spending more time on transpilation while its simulation time remains mostly constant. Statevector spends more time during simulation with transpilation time growing at a slow rate.

IV. Conclusions

In this study we present an implementation of QVMP in Qiskit that uses $O(n+m)$ qubits achieved by using QROM to encode inputs, a quadratic improvement over using bit-to-qubit encoding. We reported circuit metrics (gate count, depth, qubit count) as well as transpilation and simulation times. While QVMP can be simulated (and even run) on moderately-sized inputs, it cannot, with present hardware, scale to a degree where we can observe any quantum advantage. We demonstrate that the choice of simulation method has a noticeable impact on depth of the circuit, the effects of which trickle into transpilation and simulation time.

Acknowledgements

This research was supported in part through research infrastructure and services provided by the Rogues Gallery testbed [7] hosted by the Center for Research into Novel Computing Hierarchies (CRNCH) at Georgia Tech.

References

[1] A. Adedoyin, J. Ambrosiano, P. Anisimov, A. Bärtschi, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev, D. Gunter, S. Karra *et al.*, "Quantum algorithm implementations for beginners," *arXiv preprint arXiv:1804.03719*, 2018.

[2] A. Ambainis, H. Buhrman, P. Høyer, M. Karpinski, and P. Kurur, "Quantum matrix verification."

[3] M. Amy, M. Roetteler, and K. M. Svore, "Verified compilation of space-efficient reversible circuits," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 3–21.

[4] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, "Encoding electronic spectra in quantum circuits with linear t complexity," *Physical Review X*, vol. 8, no. 4, p. 041015, 2018.

[5] R. Freivalds, "Fast probabilistic algorithms," in *International Symposium on Mathematical Foundations of Computer Science*. Springer, pp. 57–69.

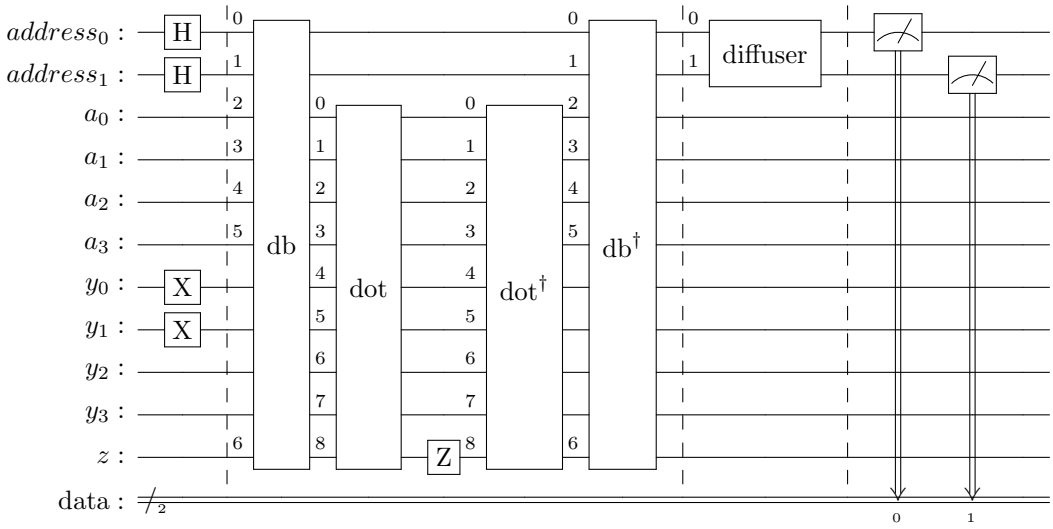


Fig. 3: QVMP circuit for a 4×4 matrix A performing one iteration

[6] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, "Quipper: a scalable quantum programming language," in *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation*, 2013, pp. 333–342.

[7] J. S. Young, J. Riedy, T. M. Conte, V. Sarkar, P. Chatarasi, and S. Srikanth, "Experimental insights from the rogues gallery," in *2019 IEEE International Conference on Rebooting Computing (ICRC)*, Nov 2019, pp. 1–8.