SOCIETY FOR NATURAL SCIENCES STUDENT JOURNAL

Magic Square Enumeration: a Quantum Approach

Lucas Van Mol[1]

[1]School of Liberal Arts and Natural Sciences, University of Birmingham

Supervisors Dr Miriam Backens and Dr Abigail Bellamy-Carter

Abstract

In this paper, a simplified version of the magic square is used to explore the use of quantum algorithms in relation to their construction and enumeration. Grover's algorithm is used for their construction, and is found to be a reasonably good alternative to similar classical alternatives. For enumeration, the quantum counting algorithm is successfully implemented for the simple square, but their use in counting solutions for higher order squares is questioned. The full paper is available on GitHub[1].

1 Introduction

Magic Squares

History Traditionally, magic squares are constructed from a set of positive integers $1, 2, ..., n^2$, such that every row, column and the two major diagonals all have the same sum, and no number is used twice.

2	9	4
7	5	3
6	1	8

Figure 1: *A magic square of order 3, with sum 15.*

Apart from their status as a mathematical curiosity, they have historically seen use in astrology, divination and the occult.[2] While there are a handful of direct, real world applications of magic squares in areas such as computer science and cryptography,[3,4] perhaps their most useful property is their links to groups, combinatorics and matrices.[5] For this paper, they will serve as a basis for exploring the applications and advantages of quantum computation in relation to search problems.

Construction Magic squares can be constructed efficiently using techniques such as the Siamese method[6] for odd n. Other deterministic methods, for singly and doubly even squares, also exist.[3,6] However, these are not able to generate arbitrary solutions, and will not be able to effectively enumerate all possible solutions, for an n-magic square. Stochastic methods for magic square generation have been proposed[7,8] using hill-climbing and evolutionary algorithms to explore the search space. In this paper, the intrinsic randomness that arises from quantum superposition and measurement will be used to generate arbitrary solutions to magic squares in a similar way to these stochastic methods.

Enumeration The number of unique solutions of a magic square of order n is an unsolved problem in mathematics for $n > 5$.

It is common to refer to the number of solutions "up to isomorphism" due to the fact that a square's "magic" property is conserved by rotation and reflection. These isomorphisms are an important feature as they allow the search space to be reduced significantly for higher order enumerations. The full sequence up to $n = 5$ is archived by the *On-Line Encyclopedia of Integer Sequences*[9] and is stated as follows:

$$1,\ 0,\ 1,\ 880,\ 275305224. \tag{1}$$

The $n = 5$ results was first computed by Schroeppel in 1973,[5] and it marked the transition of the magic square enumeration problem from a mathematical one into a computational one.

The use of exhaustive search with computers becomes extremely difficult for this problem as n grows. Since an n by n magic square is filled with n^2 digits, the total number of ways to fill in the magic square is the number of permutations of n^2 elements ($n^2!$). Enumeration techniques used by Schroeppel and Lin *et al.*[10] use backtracking, and try to take advantage of isomorphisms to reduce the search space as much as possible. However, the $O(n^2!)$ complexity of the problem suggests it would be extremely difficult to use these same techniques for higher n.

Despite this, estimates can be made. One estimate for the number of order 6 magic squares is $(1.7745 \pm 0.0016) \times 10^{19}$, which was found using Monte-Carlo methods.[11] This work hopes to investigate whether it's possible to improve upon this estimate using a quantum algorithm known as quantum counting.

Quantum Algorithms

In this section we describe the interdisciplinary approach taken for magic square enumeration, by making use of two quantum algorithms.

Grover's algorithm Also known as the quantum search algorithm, Grover's algorithm[12] is a quantum algorithm for unstructured search that will be used to find arbitrary solutions to magic squares. These magic squares will conform to an *oracle*, a black box function that differentiates valid solutions. The number of oracle queries needed is $O(\sqrt{N})$, where N is the size of the functions' domain.

Quantum counting Quantum counting[13] is an algorithm designed to determine the number of solutions M to an N item search problem. Such an algorithm on a classical computer would take $\Theta(N)$ oracle calls. As explained in section 1, the factorial growth of the magic square enumeration problem make this a very difficult task for classic computers.

The number of counting qubits t required for a desired accuracy m and a probability of finding a correct solution of $1 - \epsilon$ is

$$t \equiv m + \lceil \log_2(2 + (2\epsilon)^{-1}) \rceil \tag{2}$$

The upper bound for the error is $|\Delta M|$ is

$$|\Delta M| < \left(\sqrt{2MN} + \frac{N}{2^{m+1}} \right) 2^{-m}, \tag{3}$$

Hence, if t is large enough, then $|\Delta M| \approx 0$ so a precise value for M can theoretically be produced with high probability, given enough qubits. However, quantum counting needs an exponential (in t) amount of grover iterations, meaning that finding precise values of M to large search problems is still plagued by exponential complexity problems.

2 Methodology

Simplified Squares

Current physical realizations of quantum computers are quite limited in their ability, especially those that are publicly available. Whilst physical quantum computers are a

very active area of research, current examples of quantum advantage over classical algorithms are generally limited to specialized experiments.[14] This will drive us to investigate the magic square enumeration problem using a simplified version of the problem, in order to run simulator experiments.

The simplified magic square used in the experimentation section of this paper consists of a 3 by 3 grid, where each cell consists of a binary digit. The rules for a valid magic square is that all rows and columns must sum to the same number - we refer to this simplified square as a 'binary, semi-magic square'. An example is shown in fig. 2.

0	1	1
1	0	1
1	1	0

Figure 2: *A valid binary semi-magic square. In this example, rows and columns sum to 2.*

A function that checks valid solutions to binary semi-magic squares needs only two operations: summing 3 binary digits, and comparing the results of two of these sums. In the next section, these operations are implemented in a quantum circuit.

Subroutines

Full Adder The first subroutine introduced is a full adder, shown in fig. 3. The circuit only needs three CNOT gates and three Toffoli gates, and the output is stored on a two qubit register *anc*, since the maximum sum of 3 bits is $1 + 1 + 1 = 11_2$.

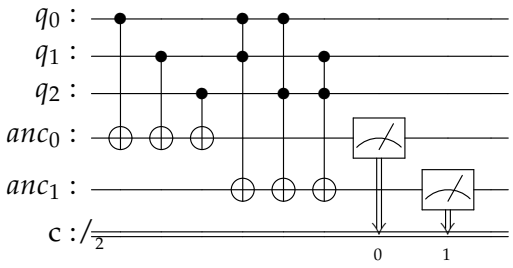


Figure 3: *Quantum full adder. The inputs are labelled $q_{0,1,2}$ and the two-bit output is $anc_{0/1}$.*

Equality Check The second subroutine needed is an equality check. In order to verify the equality of two sums, two adder circuits can be used on the 6 qubits involved, as illustrated in fig. 4. The entire circuit is explored in more detail in the full paper.[1]

Grover's Oracle

There are a total of 5 equalities, or clauses, that must be checked in order to verify the validity of a binary semi-magic square solution. The phrase 'all rows and columns must be equal' can be written mathematically as

$$r_1 = r_2 = r_3 = c_1 = c_2 = c_3, \tag{4}$$

where r_i and c_i represent the sum of row and column i, respectively. The above equation can be split into 5 distinct equality operations in a number of ways. One way of doing so is with the following 5 clauses:

$$\begin{aligned} C_1 &: \quad r_1 = r_2 \\ C_2 &: \quad r_2 = r_3 \\ C_3 &: \quad r_3 = c_1 \\ C_4 &: \quad c_1 = c_2 \\ C_5 &: \quad c_2 = c_3 \end{aligned} \tag{5}$$

The validity of a clause can be determined by the subroutine shown in fig. 4.

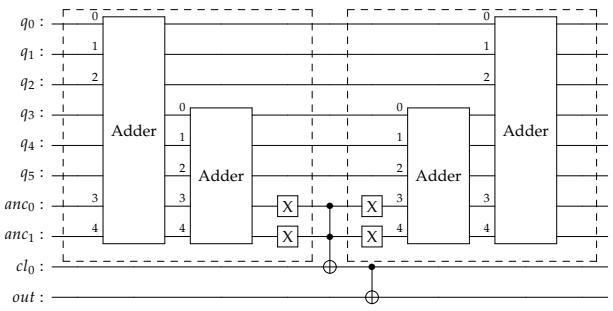
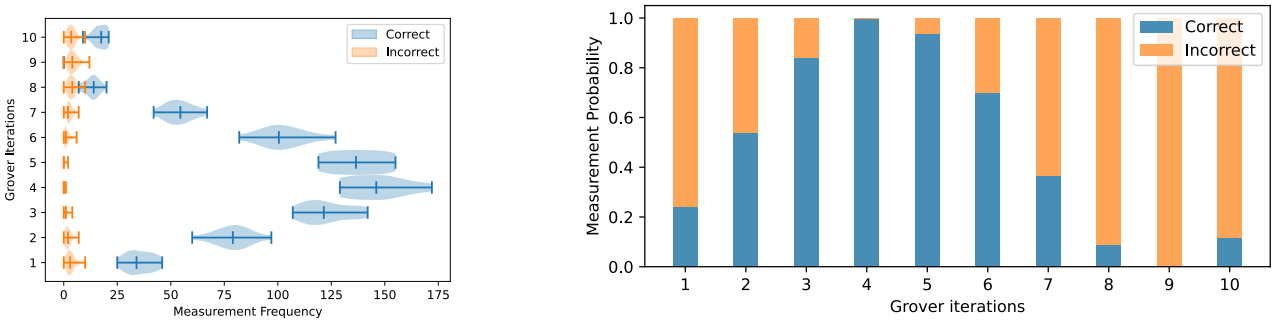


Figure 4: *Computing the truthfulness of a clause cl_0 with uncomputation of* anc. *The two grouped elements correspond to the computation and uncomputation steps respectively.*

By repeating the pattern shown for each of the 5 clauses, we can construct an oracle that



(a) *Measurement distribution*

(b) *Algorithm accuracy*

Figure 5: *Results of running Grover's algorithm, varying iterations from 1 to 10, with 2048 shots each. The maximum accuracy is found to be 99.76%, at 4 iterations. In fig. 5a, we count how many times each possible state (from $|0\rangle^9$ to $|1\rangle^9$) is measured, and group them into correct and incorrect solutions. Correct solution states are clearly seen to be measured with higher frequency when R is close to 4.*

uses a total of $9_{input} + 2_{anc} + 5_{clause} + 1_{out} = 17$ qubits.

However, we can further optimize this oracle by reducing the amount of *Adder* gates used. Clauses can essentially be 'chained' efficiently if they are put in the right order. The order shown in eq. (5) is a good example: we don't actually need to uncompute the r_2 sum after the first clause, since we can reuse it for the second clause. This results in significant reduction in the amounts of gates used.

3 Results and Analysis

For the purposes of this paper, we have used IBM's Qiskit framework, and use their freely available simulator to run the quantum algorithms described. IBM's *ibmq_QASM_simulator* allows for simulating the noise profiles of real quantum hardware, giving a good theoretical approximation of each algorithm's performance. The noise profile chosen is that of *ibm_washington*.

Grover's Algorithm

Grover's algorithm is run with the oracle described in section 2. The maximum accuracy achieved occurs at 4 iterations, with the algorithm measuring correct solution states 99.76% of the time as shown in fig. 5.

The large separation between the measurement frequency of correct vs incorrect solutions for $1 \leq R \leq 7$ in fig. 5a suggests it can be easy to determine the correctness of a solution if the algorithm is run for multiple shots, even for low R. Verification of a given solution can be done efficiently on a classical computer in a polynomial $O(n^2)$ steps for an n-magic square, since we simply need to iterate over each cell O(1) times in order to verify all the sums required. There is therefore a tradeoff between running more Grover iterations R to increase separability or choosing a lower R but with more shots. Finding an optimal balance will depend on the speed and accuracy of the physical quantum device.

Quantum Counting

The results of the quantum counting algorithm are shown in fig. 6. We take one of these states, say $|\psi\rangle$, and compute their phase using their binary decimal representation:

$$\theta = \frac{0.0111001_2}{2^t} 2\pi, \tag{6}$$

in a way analogous to quantum phase estimation.

Nielsen and Chuang[15] derive the equation for the estimated number of solutions M_{est} giving the equation

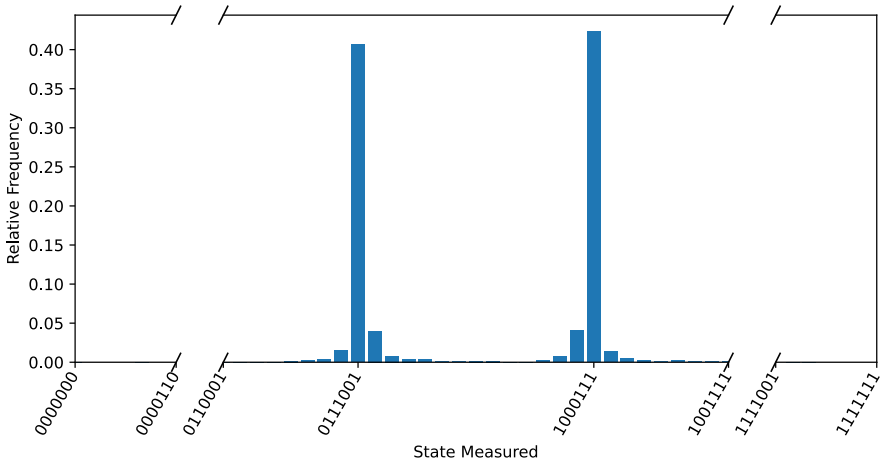


Figure 6: *Results for quantum counting with $t = 7$ counting qubits (4000 shots, simulator). Two states are measured with much higher frequency than the others: $|\psi\rangle = |0111001_2\rangle$ and $|\psi'\rangle\,|1000111_2\rangle$.*

$$M_{\text{est}} = N \sin^2(\theta/2). \tag{7}$$

Plugging in $|\psi\rangle$ or $|\psi'\rangle$ gives $M_{\text{est}} = 15$ This is within one to the actual count of 14. The upper bound for the error, as stated in eq. (3), is $|\Delta M| = 11.21$ (choosing $m = t - 1$) which is still quite significant. This can be improved with more counting qubits, although we start to reach the limits of what can be simulated due to the high amounts of entangled states in the algorithm.

Generalization to n-Magic Squares

n-magic square encoding The analysis on binary semi-magic squares means we instantly have a space-efficient, n^2-qubit encoding of an n by n binary semi-magic square. For a general magic square, we will need to find a way to encode their state in a quantum computer, which will also have consequence on the oracle implementation.

An n by n magic square can be represented as a permutation of the set $\{1, 2, ..., n^2\}$. Since Grover's algorithm starts with a superposition over the state space, the ideal encoding would be an equal superposition of all possible permutations of $\left|1, 2, ..., n^2\right\rangle$. A reversible operation that generates such a state is:

$$\begin{aligned} &\left|1, 2, ..., n^2\right\rangle |0\rangle \\ &\xrightarrow{O} \quad \frac{1}{\sqrt{n^2!}} \sum_{k=1}^{n^2!} \hat{\pi}_k \left|1, 2, ..., n^2\right\rangle |k\rangle\,, \end{aligned} \tag{8}$$

where $\hat{\pi}_k$ generates the kth permutation of the sequence. Note that this requires additional ancillary qubits that can store this number k in order for O to be reversible.

Another solution could be to use less restrictive superpositions that don't require additionally ancillary qubits. One candidate is the Dicke state[16] $\left|D_k^t\right\rangle$, corresponding to all binary strings of length t with a *Hamming weight* of k (i.e. containing k ones). For example, for a magic square of order 3, any permutation of

$$\{0000,\ 0001,\ 0010,\ 0011,\ 0100,\ 0101,\ 0110,\ 0111,\ 1000\}$$

will contain exactly $k = 13$ ones. Since the magic square contains 1 through 9, the state can be encoded with $t = 9\lceil \log_2(9) \rceil = 36$ qubits. The Dicke state $\left|D_{13}^{36}\right\rangle$ can therefore be used, reducing N from $36! = O(10^{41})$ to ${}_{36}C_{13} = O(10^9)$. Note that some clearly invalid states would be allowed, such as

$$\{0000,\ 0000,\ 0000,\ 0000,\ 0000,\ 0001,\ 1111,\ 1111,\ 1111\},$$

requiring some post-processing. Finally, a more complex superposition encoding, perhaps inspired by the generating vectors of Lin *et al.*[10] may also be able to further reduce N by eliminating certain isomorphisms.

***n*-qubit addition** For n-magic squares, row and column summation could be achieved using Draper adders,[17] which perform addition in the Fourier basis by making use of the quantum Fourier transform.

4 Conclusion

In this paper, it is shown that it is possible to find arbitrary solutions to magic squares using Grover's algorithm in $O(2^{\frac{n}{2}^2})$. The current state of quantum computing means it is too early to determine the practical advantage, or lack thereof, that a quantum alternative could have over machine learning-based methods that generate solutions stochastically.[7]

As for magic square enumeration, while section 3 demonstrates positive results for low qubit numbers, quantum counting is still an exponentially difficult problem, only improving on the classical search algorithm from $O(2^{n^2})$ to $O(2^{\frac{n}{2}^2})$.

This paper has explored a toy example in order to demonstrate a small subset of the potential benefits of quantum computing. The field of experimental quantum computing is currently in its infancy, patiently awaiting the contingent arrival of full scale quantum computers that break out of the NISQ era. The potential for these systems to be able to accurately simulate complex physical models would have enormous consequences in a wide array of fields across the sciences.

References

[1] Lucas Van Mol. *Magic Square Enumeration: a Quantum Approach*. https://github.com/lucasvanmol/quantum-squares/. Accessed 1 Dec 2023. 2023.

[2] Ho Peng Yoke. "Magic Squares in China". In: *Encyclopaedia of the History of Science, Technology, and Medicine in Non-Western Cultures* (Mar. 2008), pp. 1251–1252. DOI: 10.1007/978-1-4020-4425-0_9350.

[3] Zhenhua Duan et al. "Improved even order magic square construction algorithms and their applications in multi-user shared electronic accounts". In: *Theoretical Computer Science* 607 (Nov. 2015), pp. 391–410. DOI: 10.1016/J.TCS.2015.07.053.

[4] Chin-Chen Chang et al. "An image authentication scheme using magic square". In: IEEE. 2009, pp. 1–4. DOI: 10.1109/ICCSIT.2009.5234866.

[5] Martin Gardner. "Mathematical Games". In: *Scientific American* 234.4 (1976), pp. 126–130.

[6] Maurice Kraitchik. *Mathematical recreations*. Courier Corporation, 2006.

[7] Tao Xie and Lishan Kang. "An evolutionary algorithm for magic squares". In: *The 2003 Congress on Evolutionary Computation*. Vol. 2. 2003, pp. 906–913. DOI: 10.1109/CEC.2003.1299763.

[8] Ahmed Kheiri and Ender Özcan. "Constructing Constrained-Version of Magic Squares Using Selection Hyper-heuristics". In: *The Computer Journal* 57.3 (Mar. 2014), pp. 469–479. DOI: 10.1093/COMJNL/BXT130.

[9] *A006052 - OEIS*.

[10] Ziqi Lin et al. "Generation of all magic squares of order 5 and interesting patterns finding". In: *Special Matrices* 4.1 (Jan. 2016), pp. 110–120. DOI: 10.1515/SPMA-2016-0011/PDF.

[11] Klaus Pinn and Christian Wieczerkowski. "Number of Magic Squares from Parallel Tempering Monte Carlo". In: *International Journal of Modern Physics C* 9.4 (1998), pp. 541–546. DOI: 10.1142/S0129183198000443.

[12] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219. DOI: 10.1145/237814.237866.

[13] Gilles Brassard, Peter HØyer, and Alain Tapp. "Quantum counting". In: *Automata, Languages and Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 820–831. DOI: 10.1007/BFb0055105.

[14] Andrew J Daley et al. "Practical quantum advantage in quantum simulation". In: *Nature* 607 (2022). DOI: 10.1038/s41586-022-04940-6.

[15] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000. DOI: 10.1017/CBO9780511976667.

[16] R H Dicke. "Coherence in Spontaneous Radiation Processes". In: *Physical Review* 93.1 (Jan. 1954), pp. 99–110. DOI: 10.1103/PhysRev.93.99.

[17] Thomas G. Draper. *Addition on a Quantum Computer*. 2000. DOI: 10.48550/arXiv.quant-ph/0008033.