A BENCHMARKING STUDY OF GROVER'S ALGORITHM FOR SOLVING BOOLEAN SAT WITH QUANTUM CIRCUITS

Mr. Jirapas Unison Jipipob

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2024

การศึกษาการเปรียบเทียบประสิทธิภาพของอัลกอริธึมของกรูเวอร์ในการแก้ปัญหาบูลีนด้วยวงจรควอนตัม

นายจิรภาส ยูนิซัน จิปิภพ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2567

Thesis Title	A BENCHMARKING STUDY OF GROVER'S ALGORITHM FOR SOLVING BOOLEAN SAT WITH QUANTUM CIRCUITS
By	Mr. Jirapas Unison Jipipob
Field of Study	Computer Science
Thesis Advisor	Professor Prabhas Chongstitvatana, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Science

..... Dean of the Faculty of Engineering

(Associate Professor Witaya Wannasuphoprasit, Ph.D.)

THESIS COMMITTEE

..... Chairman

(Punnarai Siricharoen, Ph.D.)

..... Thesis Advisor

(Professor Prabhas Chongstitvatana, Ph.D.)

..... Examiner

(Punnarai Siricharoen, Ph.D.)

..... External Examiner

(Worasait Suwannik, Ph.D.)

จิรภาส ยูนิซัน จิปิภพ : การศึกษาการเปรียบเทียบประสิทธิภาพของอัลกอริธึมของกรูเวอร์ในการแก้ปัญหาบูลีนด้วยวงจรควอนตัม. (A BENCHMARKING STUDY OF GROVER'S ALGORITHM FOR SOLVING BOOLEAN SAT WITH QUANTUM CIRCUITS) อ.ที่ปรึกษาหลัก : ศ. ดร.ประภาส จงสถิตย์วัฒนา

การศึกษานี้สำรวจว่าประสิทธิภาพของอัลกอริธึมของกรูเวอร์ในการแก้ปัญหาบูลีนเป็นอย่างไร โดยใช้วงจรควอนตัม อัลกอริธึมนี้ถูกนำมาใช้งานโดยใช้เฟรมเวิร์กคิสคิทของไอบีเอ็ม และมีการเปรียบเทียบกับวิธีลองผิดลองถูกแบบดั้งเดิม การทดลองมุ่งเน้นไปที่ปัญหาสามแซท, สี่แซท, และห้าแซท โดยใช้ทั้งตัวจำลองควอนตัมและฮาร์ดแวร์ควอนตัมของไอบีเอ็ม ผลการทดลองแสดงให้เห็นว่าอัลกอริธึมของกรูเวอร์มีประสิทธิภาพมากกว่า โดยมีความได้เปรียบทางทฤษฎีแบบกำลังสองเมื่อเทียบกับวิธีคลาสสิก อย่างไรก็ตาม ในการใช้งานจริงยังคงเผชิญกับข้อจำกัด เช่น จำนวนคิวบิทที่มีอยู่จำกัด, สัญญาณรบกวนจากฮาร์ดแวร์, และความท้าทายในการปรับแต่งและเพิ่มประสิทธิภาพของวงจรควอนตัม งานวิจัยนี้แสดงให้เห็นว่าคอมพิวเตอร์ควอนตัมสามารถยกระดับความสามารถในการแก้ปัญหาทางคณิตศาสตร์ที่ซับซ้อนได้ และวางรากฐานสำหรับการศึกษาต่อไปในอนาคตเกี่ยวกับปัญหาบูลีนที่ซับซ้อนยิ่งขึ้น และการพัฒนาฮาร์ดแวร์ควอนตัมที่ล้ำหน้ามากยิ่งขึ้น

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อนิสิต .....

ปีการศึกษา 2567 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6670037521 : MAJOR COMPUTER SCIENCE

KEYWORD: quantum circuits, quantum noise, quantum simulators, computational efficiency, classical brute-force methods

Jirapas Unison Jipipob : A BENCHMARKING STUDY OF GROVER'S ALGORITHM FOR SOLVING BOOLEAN SAT WITH QUANTUM CIRCUITS. Advisor: Prof. Prabhas Chongstitvatana, Ph.D.

This study explores how well Grover's Algorithm performs in solving the Boolean Satisfiability Problem (SAT) using quantum circuits. The algorithm is implemented with IBM's Qiskit framework and compared to classical brute-force methods. Experiments focus on 3-SAT, 4-SAT, and 5-SAT problems, using quantum simulators and IBM quantum hardware. The results show that Grover's Algorithm is more efficient, offering a theoretical quadratic speedup over classical methods. However, practical issues like limited qubit availability, hardware noise, and optimization challenges impact its current performance. The data highlights the potential for quantum computing to scale and solve NP-complete problems. This research shows how quantum computing can improve problem-solving and lays the groundwork for future studies on more complex SAT problems and advanced quantum hardware.

Field of Study: Computer Science Student's Signature .....

Academic Year: 2024 Advisor's Signature .....

ACKNOWLEDGEMENTS

This research is supported by Chulalongkorn University

Jirapas Unison Jipipob

TABLE OF CONTENTS

Page

CHAPTER 1 INTRODUCTION

1.1 Motivation

Quantum computing represents a paradigm shift in how we approach computation, and I am deeply inspired by the potential it holds to revolutionize numerous fields. My interest in quantum computing is rooted in its ability to solve complex problems that are beyond the reach of classical systems. As traditional computing approaches the limits of Moore's Law, quantum computing offers a new frontier of possibilities that could drastically improve the efficiency of problem-solving, particularly in areas such as cryptography, optimization, and machine learning. By exploring Grover's Algorithm, I aim to delve into this emerging technology and contribute to its advancement, particularly in solving the Boolean Satisfiability Problem (SAT), a foundational challenge in both computer science and mathematical logic.

The SAT problem, being central to many computational tasks, offers an ideal testing ground for quantum algorithms. By comparing the performance of Grover's Algorithm with classical approaches to the SAT problem, I hope to showcase how quantum computing can significantly reduce computational complexity and time. My research seeks to optimize and implement these quantum circuits in Qiskit, a powerful tool that bridges the theoretical and practical aspects of quantum computing. I believe that such research is essential in understanding how we can make quantum computing scalable and accessible for solving real-world problems. Through this work, I hope to push the boundaries of current quantum research and inspire further experimentation in this field.

Ultimately, my motivation lies in the belief that quantum computing will play a crucial role in shaping the future of technology and human progress. As quantum computing continues to evolve, I am confident that it will become an integral part of everyday life, solving problems in medicine, energy, finance, and more. Contributing to this field at such an early stage excites me, as it allows me to be part of a larger movement that is not only reshaping computing but also positioning humanity to tackle some of its greatest challenges. Through this thesis, I aim to play my part in demonstrating how quantum computing can be harnessed to benefit society and advance the collective understanding of this groundbreaking technology.

1.2 Objective

The objective of this thesis is to explore and implement Grover's Algorithm within quantum circuits to address and solve the Boolean Satisfiability Problem (SAT). The research aims to evaluate the effectiveness of Grover's Algorithm in improving the efficiency of SAT problem-solving compared to classical methods, utilizing Qiskit for practical implementation. The study will involve designing and optimizing quantum circuits in Qiskit to enhance the performance and scalability of SAT problem solutions

1.3 Scope of work

1. Define the specific types of SAT problems (e.g., 3-SAT or higher) to be used in the study.
2. Implement quantum circuits in Qiskit for solving the chosen SAT problems using Grover's Algorithm.
3. Implement classical brute-force algorithms for solving the same SAT problems.
4. Conduct multiple experiments to compare the performance of both methods.
5. Analyze the results of the experiments to compare the performance of the quantum (Qiskit) and classical (brute-force) methods.
6. Summarize the findings and discuss the implications for the use of quantum computing in solving SAT problems.

1.4 Expected result

The hypothesis of this thesis is that applying Grover's Algorithm to solve the Boolean Satisfiability Problem (SAT) using quantum circuits in Qiskit will result in a significant improvement in computational efficiency compared to classical algorithms. Specifically, it is anticipated that Grover's Algorithm will provide a quadratic speedup in finding satisfying assignments by leveraging quantum parallelism, which should be evident through reduced computational time and resources. This hypothesis is grounded in the theoretical advantage of Grover's Algorithm for unstructured search problems and the potential for optimized quantum circuit design in Qiskit to enhance this advantage.

1.5 Research plan

1. Conduct a literature review on the SAT problem and refine the research question focusing on 3-SAT and higher variations.
2. Study both classical brute-force algorithms and Grover's Algorithm for solving SAT problems and familiarize with Qiskit for quantum simulations.
3. Implement classical brute-force algorithms and Grover's Algorithm in Qiskit, ensuring both methods are optimized for fair comparison.
4. Run experiments on both approaches using different SAT problem complexities and collect performance data.
5. Summarize findings, discussing implications for quantum computing and proposing directions for future research.
6. Complete the thesis with a clear presentation of results and submit for review and feedback.

1.6 Contribution

The main contribution of this thesis is the practical evaluation of Grover's Algorithm for solving Boolean satisfiability problems (3-SAT, 4-SAT, and 5-SAT) using both quantum simulations and real quantum hardware. While Grover's Algorithm is theoretically known to offer a quadratic speedup over classical search algorithms, this study bridges the gap between theory and practical application by implementing and testing the algorithm using Qiskit on IBM's quantum simulator and the IBM Sherbrooke quantum processor. The thesis presents a comparative analysis between quantum and classical brute-force approaches, highlighting how Grover's Algorithm performs under real-world hardware constraints. In addition to implementing the SAT-to-quantum oracle translation using Qiskit's PhaseOracle, the research explores how execution time and success rate vary with increasing problem complexity. The collected experimental data is used to visualize performance trends and to extrapolate the potential scalability of Grover's Algorithm as quantum hardware evolves. This work contributes to the growing field of quantum computing by providing empirical evidence of both the potential and current limitations of quantum algorithms for NP-complete problems like SAT and offers a foundation for future research into more advanced or scalable quantum SAT solvers.

1.7 Justification for Excluding Heuristic Methods

Although heuristic SAT solvers can be highly efficient in practice, they are not guaranteed to find a solution for every satisfiable formula and cannot prove unsatisfiability. This thesis focuses exclusively on complete methods, namely, brute-force search and Grover's quantum algorithm—which are capable of solving all SAT instances, regardless of satisfiability. This choice aligns with the research objective of comparing exhaustive classical and quantum approaches under a consistent framework of completeness and worst-case guarantees. Heuristic methods are acknowledged for their practical relevance but are outside the scope of this study.

CHAPTER 2 BACKGROUND KNOWLEDGE

This section is divided into four parts. First, it will cover the Boolean Satisfiability Problem (SAT), providing a fundamental understanding of its complexity and relevance in computational theory. Second, it will discuss Classical Approaches to SAT, focusing on traditional methods such as brute-force and its limitations in solving larger problems. The third section will introduce Grover's Algorithm, explaining how quantum computing offers a potential speedup for solving SAT problems compared to classical methods. Finally, the fourth part will explore Quantum Computing and Qiskit, outlining the principles of quantum computation and the use of the Qiskit framework to simulate and test quantum algorithms in this study.

2.1 Boolean Satisfiability Problem (SAT)

The Satisfiability (SAT) problem is one of the most critical issues in computational theory, particularly in logic, constraint satisfaction, and various areas such as VLSI design and machine learning. SAT is the task of determining whether there exists an assignment of truth values to variables that makes a given Boolean formula satisfiable. This formula is typically expressed in Conjunctive Normal Form (CNF), where clauses are composed of literals (variables or their negations) connected by logical OR, and the overall formula is a conjunction (AND) of these clauses.

SAT is central to a family of NP-complete problems, meaning that while a solution can be verified quickly, finding the solution itself may require non-deterministic polynomial time in the

worst case. It has broad applications, including optimization, artificial intelligence, and circuit design. Traditional methods for solving SAT focus on treating it as a decision problem, employing algorithms such as resolution-based techniques. However, more recent approaches have transformed the SAT problem into an optimization problem, where the objective is to minimize the number of unsatisfied clauses. This allows the use of iterative optimization techniques, including local search methods, which have shown increased efficiency for specific classes of SAT formulas compared to classical approaches [1]

2.2 Classical Approaches to SAT

Classically, SAT problems are often solved using algorithms like the brute-force method, which explores every possible combination of truth values, making it highly inefficient for large problems. More sophisticated algorithms, like the Davis-Putnam-Logemann-Loveland (DPLL) algorithm [2] and its variations, attempt to reduce the search space by recursively simplifying the problem. However, even these optimized algorithms struggle with the computational demand as the problem size increases. SAT solvers are widely used, but they have limitations when dealing with large, complex instances due to the exponential time complexity associated with NP-complete problems.

2.3 Grover's Algorithm

Grover's Algorithm, introduced by Lov Grover in 1996 [3], is a quantum search algorithm that offers a quadratic speedup over classical search algorithms. It is designed to search through an unsorted database or solution space of size N in $O(\sqrt{N})$ time, making it a powerful tool for solving problems like SAT. While classical brute-force methods for SAT operate in exponential time (O(2^n) for n variables), Grover's Algorithm can reduce the time complexity to $O(\sqrt{2^n})$, offering a significant performance advantage, particularly for large problem instances. Grover's Algorithm is particularly suited for structured search problems and demonstrates the potential of quantum computing to solve NP-complete problems more efficiently than classical methods.

2.4 Quantum Computing and Qiskit

Quantum computing leverages the principles of superposition and entanglement to perform operations on quantum bits (qubits), which can represent both 0 and 1 simultaneously. This allows quantum computers to process vast amounts of data in parallel, which classical computers cannot achieve [4]. Qiskit is an open-source quantum computing framework developed by IBM [5], which provides tools for designing, simulating, and running quantum circuits. Qiskit's quantum simulators enable the testing of quantum algorithms, such as Grover's, in a controlled environment before deploying them on actual quantum hardware. The framework is essential for experimenting with quantum algorithms and comparing their performance to classical methods.

2.5 Quantum Entanglement

Quantum entanglement, a foundational concept in quantum mechanics, refers to the non-classical correlations between subsystems of a compound quantum system. Initially recognized by Einstein, Podolsky, and Rosen, it took over 70 years to be fully appreciated as a tangible resource with profound implications for quantum processes. Entanglement plays a central role in quantum cryptography, teleportation, and dense coding, offering potential for advances in quantum communication and computation. Despite its complexity and environmental fragility, entanglement is robust in theoretical frameworks, with tools like Bell inequalities and entanglement witnesses used for its detection and characterization. The irreversibility of entanglement manipulations, particularly in the context of bound entanglement, highlights its unique role in quantum communication and computation [6].

2.6 NP-Complete

NP-Complete problems are a pivotal class within computational complexity theory [7]. A problem is in NP if, given a solution, it can be verified in polynomial time. Any problem in NP can be transformed into an NP-Complete problem in polynomial time. A problem P is NP-Complete if it is in NP and every problem Q in NP can be reduced to P using a polynomial-time transformation. This indicates that if any NP-Complete problem can be solved in polynomial time, all NP problems can also be solved in polynomial time, effectively establishing that P = NP [8].

CHAPTER 3 LITERATURE REVIEW

3.1 Satisfiability Problem (SAT)

In the field of computational complexity, the Satisfiability Problem (SAT) plays a central role due to its wide applicability in logic, optimization, artificial intelligence, and computational theory. The work of Thomas J. Schaefer in "The Complexity of Satisfiability Problems" [9] made a significant contribution to this area by classifying a broad range of SAT problems and demonstrating that each problem in this infinite class is either polynomial-time solvable or NP-complete, with no intermediate complexity classes.

Schaefer's paper specifically explores the distinction between SAT instances with clauses restricted to two literals, which are efficiently solvable in polynomial time, and SAT instances with three literals per clause, which are proven to be NP-complete. This establishes a foundational result: SAT problems with larger clause sizes are generally computationally harder and belong to the NP-complete class, for which no efficient (polynomial-time) solution is known unless P=NP. Schaefer's work broadens this understanding by presenting a classification theorem that applies to an infinite family of SAT problems. The classification determines whether a given SAT problem is in the polynomial-time decidable class or NP-complete, depending on the specific structure and constraints of the propositional formulas.

Additionally, Schaefer extends this analysis to quantified SAT problems, a more complex version involving quantifiers, and demonstrates that these problems are either solvable in polynomial time or require exponential space. The results from this paper serve as a framework for identifying new NP-complete problems, as well as polynomial-time problems, offering a deeper understanding of computational complexity in logic-based problem solving. Schaefer's classification theorem has been a critical tool in the study of satisfiability problems and has inspired extensive research in both theoretical and practical aspects of computational complexity.

3.2 The Significance of Quantum Computing

The field of quantum computing has emerged as a revolutionary area of research, blending concepts from classical information theory, computer science, and quantum physics. In a paper, Quantum Computing [10], Andrew Steane highlights the pivotal role quantum computing plays in

reshaping our understanding of computation and the natural world, particularly by integrating the concept of quantum information into the computational domain.

Steane begins by positioning quantum computing within the broader framework of information theory, tracing its roots back to mid-20th century developments in classical information theory and computer science. Classical theories of computation, such as Turing machines and Shannon's information theory, provide the foundation for understanding quantum computing's departure from classical systems. The difference is most notably captured in the Einstein, Podolsky, and Rosen (EPR) experiment and the EPR-Bell correlations [11], which distinguish quantum from classical physics. Quantum entanglement, as Steane notes, is a core component in these divergences, enabling new forms of computation and information transfer.

A central theme in Steane's review is the quantum bit (qubit), which serves as the fundamental unit of quantum information. Unlike classical bits, which are binary and exist as 0 or 1, qubits can exist in superpositions of states. This principle underlies the significant

3.3 Exploring Grover's Algorithm

In recent years, quantum computing has emerged as a transformative field with the potential to outperform classical algorithms in various applications, particularly in data processing and optimization. One notable quantum algorithm is Grover's algorithm, which offers a quadratic speedup for unstructured search problems compared to classical counterparts. Grover's algorithm operates on a quantum superposition of states, allowing it to search through N items in approximately $\sqrt{N}$ queries, a significant advantage over the O(N) time complexity required by classical search algorithms.

The theoretical underpinnings of Grover's algorithm demonstrate its applicability across various domains, including cryptography, database searching, and optimization problems. As the data landscape expands, with datasets reaching and surpassing petabytes, the demand for efficient algorithms like Grover's is increasingly pressing. This need is particularly pronounced in areas such as machine learning and artificial intelligence, where rapid data processing is crucial.

Recent studies have focused on implementing Grover's algorithm on real quantum hardware, such as IBM's quantum computers. Mandviwalla et al. explore the practical application of Grover's algorithm through multiple implementations on IBM Q devices, providing empirical results that

reflect the capabilities and limitations of current quantum technology. Their research highlights the challenges of achieving theoretical accuracy in practical applications, where factors such as qubit coherence and error rates play critical roles [12].

3.4 Grover's Algorithm's Impact on SAT Problems

The satisfiability problem, particularly in its NP-complete form, poses significant challenges in classical computing. Specifically, determining whether a given Boolean formula in conjunctive normal form (CNF) is satisfiable involves searching through an exponential number of potential variable assignments. Grover's algorithm can be employed to enhance the efficiency of this search process, offering a promising approach to tackle the SAT problem.

Cheng and Tao (2024) investigate the application of Grover's algorithm specifically for 3-SAT, a variant of SAT where each clause contains exactly three literals. They highlight the limitations imposed by current quantum technology, particularly the number of stable qubits available for practical implementations. The authors point out that the performance of Grover's algorithm is directly linked to the size of the oracle used and the number of repeated calls to it. This creates a constraint on the number of qubits that can be effectively employed without sacrificing performance [13].

3.5 Applications of Grover's Algorithm Beyond SAT Problems

One notable application of Grover's algorithm is in database search and pattern matching. Tezuka et al. (2024) proposes a novel approach that implements Grover's algorithm for image pattern matching, demonstrating its potential to enhance data retrieval processes significantly. The authors utilize an approximate amplitude encoding method in a shallow quantum circuit, enabling efficient data loading and amplitude amplification. This adaptation addresses the challenges previously faced in realizing the original motivations behind Grover's algorithm in practical settings [14].

The algorithm operates by encoding the data in a quantum state that resembles the query, followed by an amplitude amplification process independent of the target data index. This approach not only highlights the algorithm's capability in traditional database search scenarios but also

showcases its application in more complex contexts such as image processing, where pattern recognition and matching are crucial.

3.6 Heuristic Method Limitations

The study by Costa et al. (2024), *"Assessing Quantum and Classical Approaches to Combinatorial Optimization: Testing Quadratic Speed-ups for Heuristic Algorithms,"* explores the limitations of heuristic approaches such as WalkSAT and DOLL when applied to combinatorial problems like SAT. These heuristic algorithms are commonly used to find near-optimal or locally optimal solutions but lack guarantees of finding the global optimum or all possible solutions. For instance, WalkSAT uses a randomized search strategy that makes local variable flips to navigate the solution space. However, it can easily become trapped in local optima, making it ineffective in many cases. Because these methods generally explore only a limited portion of the search space, they risk missing valid solutions and may perform poorly on certain types of instances [15].

In contrast, Grover's algorithm offers a more robust alternative by enabling a quantum-enhanced, structured search through the entire solution space. With a high probability of success and the ability to guarantee the correct solution if repeated sufficiently, Grover's approach differs fundamentally from heuristics. While heuristic methods provide approximations, Grover's algorithm is designed for exact, exhaustive search, making direct comparisons between them methodologically unsound (Costa et al., 2024).

Furthermore, heuristic algorithms are not ideal benchmarks when evaluating the performance of quantum algorithms like Grover's against classical brute-force methods. Effective benchmarking requires consistency, reproducibility, and completeness—criteria heuristics do not reliably meet due to their instance-specific behavior, probabilistic nature, and lack of completeness guarantees. In contrast, brute-force search, though computationally expensive, is exhaustive and deterministic, providing a clear and unbiased baseline. Grover's algorithm enhances this exhaustive process through quantum amplitude amplification, yielding a quadratic speedup over classical brute-force. Therefore, comparing quantum algorithms to heuristics can lead to misleading conclusions, as it blends algorithmic performance with heuristics-specific tuning and randomness, rather than isolating true quantum advantages.

3.7 Heuristic Approaches for SAT Solving

While this thesis focuses on complete methods that guarantee solutions or proofs of unsatisfiability, heuristic algorithms have been widely studied and successfully applied to solve many SAT instances efficiently in practice. Heuristic solvers do not guarantee solving every instance but often provide significant speed advantages on large or structured problems.

A recent notable example is the work by Cen et al. [16], which proposes *FastFourierSAT*, a novel heuristic SAT solver that leverages GPU-accelerated continuous local search techniques. By representing Boolean variables as continuous values and applying fast Fourier transform (FFT) operations to efficiently explore the solution space, FastFourierSAT achieves remarkable runtime improvements compared to traditional local search and complete solvers.

The authors demonstrate that FastFourierSAT can solve large-scale SAT benchmarks, including industrial and randomly generated instances, with speedups up to two orders of magnitude on modern GPUs. Their experiments highlight that continuous optimization combined with hardware acceleration enables heuristic solvers to effectively handle SAT problems previously considered challenging.

This evidence supports the position that heuristic methods constitute a valuable class of SAT solvers in practical scenarios, complementing the complete approaches studied in this thesis.

3.8 Quantum Supremacy

Arute et al. (2019) demonstrated quantum supremacy by utilizing a noisy quantum processor to solve a specific random circuit sampling task, outperforming classical methods. This achievement marked a significant milestone in quantum computing, illustrating that quantum systems, despite being imperfect and subject to noise, can still provide a computational advantage for certain problems. While the quantum processor used in this experiment was not error-corrected, the results emphasize the potential of quantum computing to surpass classical methods in specific contexts, even with current noisy hardware. The experiment highlights the importance of

continuing to explore the capabilities of quantum systems in their present noisy state, as it offers valuable insights into the future development of quantum computing [17].

3.9 Differences Between Boolean SAT-3, SAT-4, and SAT-5 and Critical Transition

Boolean satisfiability problems, or SAT problems, are fundamental in computational complexity and involve determining if there exists an assignment of truth values that satisfies a given Boolean formula. The k-SAT problems, such as 3-SAT, 4-SAT, and 5-SAT, differ based on the number of literals in each clause. In 3-SAT, each clause contains exactly three literals, while in 4-SAT and 5-SAT, clauses contain four and five literals respectively. All these problems are NP-complete for $k \geq 3$, but increasing k tends to make individual clauses more complex and can influence the density and structure of the formula. A critical concept related to these problems is the phase transition, or critical transition, which refers to a sharp change in the satisfiability property of randomly generated SAT instances as the ratio of clauses to variables crosses a specific threshold. For 3-SAT, this critical ratio is approximately 4.26; below this value, most formulas are satisfiable, while above it, they become almost unsatisfiable. The location of this phase transition shifts for 4-SAT and 5-SAT, generally moving to higher clause-to-variable ratios due to the increased clause length. This transition is important because problem hardness peaks near the critical point, making SAT instances hardest to solve around this region. Understanding these transitions has significant implications for designing algorithms and studying computational hardness. The pioneering work by Monasson et al. provides a detailed analysis of these phase transitions in SAT problems, linking computational complexity to statistical physics phenomena [18].

CHAPTER 4 EXPERIMENTAL METHOLOGY

In this chapter, we outline the experimental methodology employed to evaluate the performance of Grover's algorithm on the satisfiability problems (3-SAT, 4-SAT, and 5-SAT). Utilizing Qiskit, we conduct experiments on both a quantum simulator and actual quantum hardware, when feasible. This allows us to compare the efficiency of Grover's algorithm against classical brute-force methods in solving these SAT problems.

We will detail the selection of SAT instances, the implementation of Grover's algorithm, and the execution of classical algorithms. Performance metrics will be established to quantify the execution time and success rates of each approach. The results will be visualized through graphical representations, illustrating how the size of the SAT problem impacts computational speed.

Additionally, we will discuss predictions regarding the scalability of quantum computing, particularly focusing on how an increase in the number of qubits might enhance performance relative to classical methods. This investigation aims to provide insights into the potential advantages of quantum computing in solving NP-complete problems, highlighting the future implications of advancements in quantum hardware.

4.1 Overview of the Experimental Setup

For the quantum simulations, we employed Qiskit, an open-source quantum computing framework, running on Google Colab with the Qiskit Aer Simulator as the backend. This environment allowed us to efficiently prototype and test Grover's algorithm across different SAT problem instances. The flexibility of Qiskit enables the creation of quantum circuits tailored to the specific requirements of each problem, facilitating quick iterations and optimizations.

To evaluate the performance of Grover's algorithm on actual quantum hardware, we used IBM's quantum processor, specifically the IBM Quantum System One, codenamed "Sherbrooke." This platform provides access to real quantum devices, allowing us to compare results obtained from simulations with those executed on a physical quantum computer. The Sherbrooke device offers a limited number of qubits, which is crucial for the implementation of Grover's algorithm, particularly for larger SAT problems.

For the classical brute-force approach, we implemented the SAT solving algorithm using Python in Google Colab. This setup enabled us to efficiently explore all possible combinations of

variable assignments to determine the satisfiability of each SAT instance. By running the classical algorithm alongside the quantum implementations, we aimed to draw direct comparisons between their performance metrics.

The experiments were designed to collect data on execution time and success rates for each algorithm across varying problem sizes. This comprehensive approach enables us to analyze the impact of problem size on computational speed and to predict the advantages of quantum computing as the number of qubits increases.

4.2 Description of the SAT Instances

In this section, we describe the structure and generation of the Boolean Satisfiability Problem (SAT) instances used in our experiments. Specifically, we focus on the 3-SAT, 4-SAT, and 5-SAT problems, which vary in complexity based on the number of literals per clause. Each SAT instance consists of a Boolean formula expressed in Conjunctive Normal Form (CNF), where the formula is a conjunction (AND) of clauses, and each clause is a disjunction (OR) of literals. A literal is either a variable or its negation.

4.2.1 3-SAT Problem

The 3-SAT problem is a special case of the SAT problem where each clause contains exactly three literals. It is one of the most well-known NP-complete problems, widely used in computational complexity research.

Example of a 3-SAT problem:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (x_1 \vee x_4 \vee \neg x_5)$$

Figure 1: 3-SAT problem

In this example, the formula is satisfiable if there exists a truth assignment for the variables x1, x2, x3, x4, x5 that makes the entire formula true. The challenge is to find a satisfying assignment or to prove that no such assignment exists.

4.2.2 4-SAT Problem

The 4-SAT problem extends 3-SAT by increasing the number of literals per clause to four. As the number of literals per clause increases, the complexity of finding a solution generally increases.

Example of a 4-SAT problem:

$$(x_1 \vee x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4 \vee x_5) \wedge (x_1 \vee \neg x_2 \vee x_4 \vee x_5)$$

Figure 2: 4-SAT problem

Like the 3-SAT problem, the task is to determine whether there is a truth assignment for x1, x2, x3, x4, x5 that satisfies all the clauses.

4.2.3 5-SAT Problem

The 5-SAT problem is a further extension where each clause contains five literals. This increases complexity even more, making it a more challenging problem to solve.

Example of a 5-SAT problem:

$$(x_1 \vee \neg x_2 \vee x_3 \vee x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_2 \vee \neg x_3 \vee x_4 \vee x_5) \wedge (x_1 \vee \neg x_3 \vee x_4 \vee x_5 \vee \neg x_6)$$

Figure 3: 5-SAT problem

In this case, the problem is to determine whether a truth assignment for the variables x1, x2, x3, x4, x5, x6 can make the entire formula true.

4.3 Implementation of Grover's Algorithm

In this section, we outline the process used to implement Grover's algorithm to solve the 3-SAT, 4-SAT, and 5-SAT problems. Grover's algorithm is a quantum search algorithm that can be used to find solutions to satisfiability problems efficiently, offering quadratic speedup over classical search algorithms. The implementation is done using Qiskit, IBM's quantum computing framework, and is structured as follows.

4.3.1 Expressing SAT Problems

To solve SAT problems using Grover's algorithm, we first represent each SAT instance in a human-readable format known as the Boolean string format. This format expresses the SAT problem as a conjunction (AND) of disjunctions (OR), where each clause consists of literals represented as variables or their negations.

4.3.2 Using Qiskit's PhaseOracle

The next step is to translate the Boolean SAT formula into a quantum circuit using the PhaseOracle function provided by Qiskit. The PhaseOracle takes the SAT problem expressed in string format and generates a corresponding quantum oracle, which marks the solutions (satisfying assignments) of the SAT problem.

4.3.3 Applying Grover's Operator

Once the oracle is created, we apply Grover's algorithm to search for a satisfying solution. After that, we prepare a register of qubits. Each qubit corresponds to one of the variables in the SAT problem. Finally, A Hadamard gate is applied to each qubit to create an equal superposition of all possible states. This allows the algorithm to explore all possible variable assignments in parallel.

4.3.4 Transpiling the Circuit

Before executing the quantum circuit, it needs to be optimized and compiled for the target quantum hardware. The transpiler in Qiskit is responsible for mapping the high-level circuit onto the hardware while minimizing gate errors and decoherence [19]. In our implementation, we set the optimization level to 3 to achieve the highest degree of optimization for both the simulator and the quantum hardware.

The transpiler reduces the gate count and optimizes the circuit layout, which is crucial for achieving accurate results, especially on real quantum hardware with limited qubit coherence time.

4.4 Experimental Procedure

This section outlines the procedure for conducting experiments to solve 3-SAT, 4-SAT, and 5-SAT problems using both quantum and classical methods. The goal is to compare the performance of Grover's algorithm implemented in Qiskit on a quantum simulator and actual quantum hardware against classical brute-force algorithms implemented in C.

4.4.1 Quantum Experimentation with Qiskit

The first set of experiments is conducted using the Qiskit framework to implement Grover's algorithm on a quantum simulator as well as on IBM's Sherbrooke quantum hardware.

4.4.2 Classical Brute-Force Experimentation

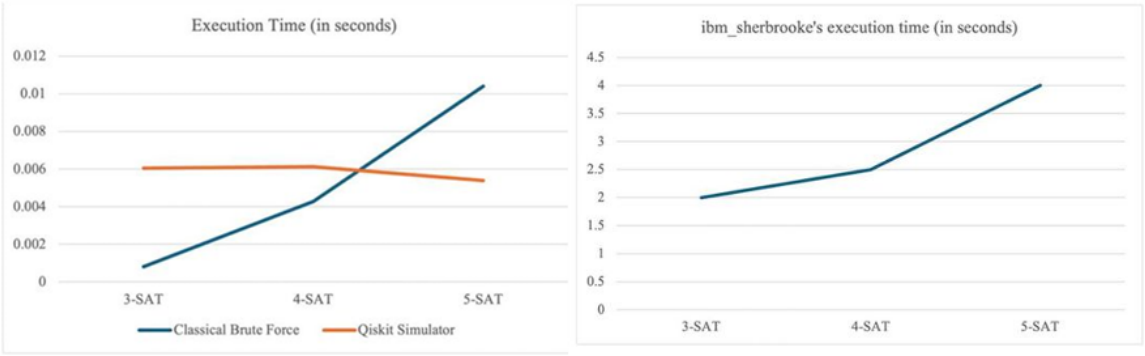
To compare the quantum results with classical methods, a brute-force algorithm is implemented in Python. The brute-force algorithm systematically checks every possible combination of variable assignments to find the solution that satisfies the SAT problem.

4.4.3 Comparison

After gathering the experimental results from both quantum and classical approaches, the data will be compared in terms of execution time and accuracy.

- Quantum Execution Time - Grover's algorithm was executed on both the Qiskit Aer simulator and IBM's quantum hardware (IBM Sherbrooke). The execution time was plotted against the size of the SAT problem (measured by the number of variables and clauses).
    - Qiskit Aer Simulator: Execution times increased modestly with problem size. The simulator consistently returned correct results with high accuracy across 3-SAT, 4-SAT, and 5-SAT problems. However, execution times were still significantly longer than classical brute force for small instances, largely due to the overhead of quantum circuit simulation.
    - IBM Sherbrooke Hardware: Real hardware execution times were orders of magnitude slower than both the simulator and classical methods, primarily due to queue delays, circuit compilation time, and limited coherence times. Accuracy

was also significantly lower, with success rates dropping as problem size increased. This is attributed to noise, gate errors, and decoherence inherent to current noisy intermediate-scale quantum (NISQ) devices.

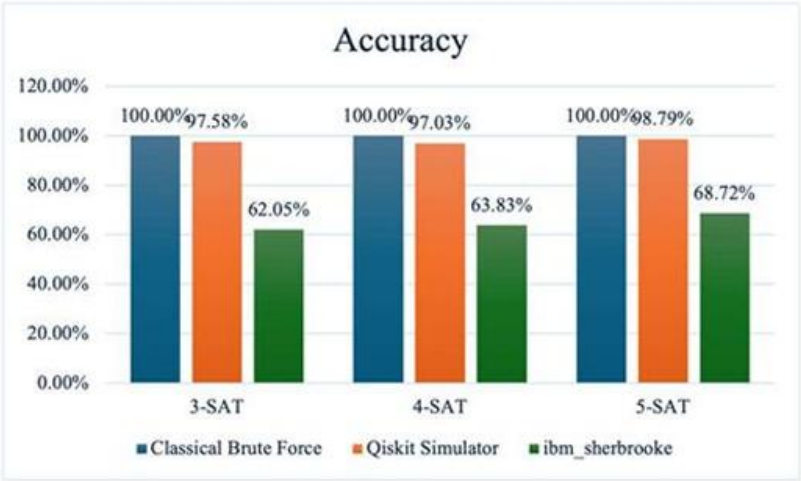


The chart compares the execution time of solving SAT problems using three approaches: classical brute-force (C implementation), Qiskit simulator, and IBM quantum hardware. The x-axis represents the size of the SAT problem (based on the number of variables and clauses), while the y-axis shows execution time in seconds.

- Classical Execution Time - The brute-force SAT solver implemented in C consistently achieved the fastest execution times across all tested problem sizes. It also returned correct results 100% of the time, confirming its reliability for small SAT instances. Execution time scaled exponentially, as expected, with the size of the problem, but remained manageable for 3-SAT to 5-SAT cases.

  Accuracy and Success Rate

  - Classical Solver: Maintained 100% accuracy across all problem sizes.
  - Qiskit Simulator: Demonstrated near-perfect accuracy, with success rates exceeding 95% even as problem complexity increased. This indicates the simulator's reliability for modeling Grover's algorithm in a noiseless environment.
  - Quantum Hardware: Showed success rates ranging from 60% to 80% for 3-SAT problems, which declined sharply for 4-SAT and 5-SAT instances. Error accumulation due to gate noise and limited coherence was a major limiting factor.



This chart illustrates the accuracy (or success rate) of solving SAT problems using the classical brute-force method, Qiskit simulator, and IBM quantum hardware. The x-axis shows the SAT problem size, while the y-axis represents the percentage of correct solutions found.

- Plotting the Results - The execution times for both quantum and classical approaches will be plotted on a graph to visualize the difference in scaling as the SAT problem size increases. The chart will help illustrate how quantum methods scale with increasing problem size in comparison to classical brute-force methods.
- Extrapolation for Larger SAT Problems - Based on the collected data, we will predict how the performance would change for larger SAT problems (such as 6-SAT or beyond) if quantum computers with a higher number of qubits were available. This prediction will be made by extrapolating the trends observed from both the quantum and classical experiments.

4.5 Data Collection and Visualization

In this section, we outline the methods employed for data collection and visualization of the experimental results obtained from both the quantum and classical computations.

4.5.1 Data Collection

Data collection was performed systematically during the experiments, ensuring that key metrics were recorded for each test case. For the quantum implementations, we gathered data on

execution times from both the Qiskit simulator and the IBM Sherbrooke quantum processor. Key metrics included:

- Execution Time - The total time taken for the algorithm to complete, measured in milliseconds
- Success Rate - The percentage of successful runs that resulted in the correct output for the given SAT instance

4.5.2 Data Visualization

To facilitate the analysis of the data collected, we utilized graphical representations. Various types of visualizations will be employed, including:

- Bar Charts - To compare the execution times of Grover's algorithm against the classical method for each problem size
- Line Graphs - To illustrate trends in execution time as the size of the SAT problem increases, highlighting the potential benefits of quantum speedup

These visualizations will aid in understanding how the performance of Grover's algorithm scales with problem size and the implication of increased qubit counts on computational efficiency. By analyzing these results, we aim to derive conclusions regarding the potential of quantum computing to outperform classical methods in solving NP-complete problems.

4.6 Limitations and Challenges

While the experiments conducted in this thesis offer valuable insights into the performance of Grover's algorithm for solving SAT problems, there are several key limitations and challenges that affect the scope and accuracy of the results:

4.6.1 Limited Qubit Availability

One of the most significant limitations of this study is the restricted number of qubits available on current quantum hardware. As Grover’s algorithm requires an increasing number of qubits to represent larger SAT problems (such as 6-SAT or higher), the available quantum computers, such as IBM Sherbrooke, do not have enough qubits to experiment with high-

complexity SAT instances. This limitation prevents us from exploring higher-order k-SAT problems and fully testing the scalability of Grover's algorithm on such problems.

4.6.2 Quantum Hardware Constraints

Quantum computers available to the public, such as those provided by IBM, often have limited access times. The restricted usage time for the quantum hardware means that only a limited number of SAT problem instances could be run and tested, making it difficult to conduct large-scale experiments or repeat experiments extensively to improve accuracy. Additionally, the current error rates in quantum hardware can lead to less reliable results, further contributing to the challenges of performing high-precision experiments.

4.6.3 Inaccuracy and Noise in Quantum Results

Due to noise and decoherence in quantum systems, the results of the quantum experiments may not always be accurate or consistent with theoretical expectations [20]. While Grover's algorithm is designed to provide quadratic speedup, this advantage can be diminished on real quantum hardware by issues like gate errors and readout errors. As a result, the outcomes presented in this thesis are more indicative of the potential future performance of quantum computers rather than an exact measure of their current capabilities.

4.6.4 Limited Problem Scope

The study focuses on 3-SAT, 4-SAT, and 5-SAT problems, which are relatively small instances due to the limitations of both classical and quantum computing resources. As a result, the experimental data may not fully represent how Grover's algorithm would perform on larger, real-world SAT problems. While the results provide insight into how increasing problem size affects execution time, the conclusions drawn here are more predictive and hypothetical for future experiments with more powerful quantum hardware.

4.6.5 Focus on Predictive Data

Given the limitations mentioned above, this thesis emphasizes the collection of data to make predictions about the future of quantum computing rather than delivering concrete, definitive

results. The experiments conducted serve as a preliminary foundation for future research. The predictions made in this thesis are based on extrapolations of the data collected from small-scale experiments, and more comprehensive studies are required once quantum computers with more qubits and greater stability become available.

4.6.6 Limitation of Heuristic Methods in Grover vs. Brute Force Benchmark

Heuristic methods, while often effective for classical SAT solving by exploiting problem-specific structures or patterns, are not directly comparable to Grover's algorithm in this benchmark due to fundamental differences in approach and guarantees. Grover's algorithm provides a guaranteed quadratic speedup for unstructured search problems, operating under the assumption that no exploitable structure is known beforehand. In contrast, heuristic classical methods rely heavily on domain-specific optimizations that can lead to variable performance and may not scale predictably across all problem instances. This makes it challenging to establish a fair and consistent baseline for comparison. Moreover, heuristics can sometimes fail to guarantee correctness or completeness, whereas Grover's algorithm systematically amplifies the probability of the correct solution. As a result, benchmarking Grover's algorithm against heuristic methods risks conflating fundamentally different problem-solving paradigms, limiting the clarity of performance insights [21].

4.6.7 Factors Affecting Accuracy and Speed of Grover's Algorithm in SAT Solving

The decrease in accuracy observed on IBM quantum hardware is primarily due to quantum noise, which includes factors like decoherence, gate errors, and measurement inaccuracies. As SAT problem sizes increase, the quantum circuits used for Grover's algorithm become deeper and more complex, making them more susceptible to these noise sources. This leads to a reduced success rate in finding the correct solution, especially when compared to the idealized performance seen in simulations.

While there are techniques available to address these issues such as zero-noise extrapolation or measurement error correction and quantum error correction—these methods introduce significant overhead. They often require additional circuit executions, increased qubit

resources, or complex calibration steps. Given the limited qubit count and short coherence times of current quantum devices, especially those used in this experiment, these techniques are not practical or scalable for the SAT problem sizes tested.

Moreover, incorporating such methods would complicate the experimental setup and make it difficult to fairly compare quantum and classical approaches under similar conditions. Therefore, to maintain a clear and unbiased evaluation, the experiment used standard, uncorrected implementations of Grover's algorithm, highlighting the real-world limitations of current quantum hardware. [3].

CHAPTER 5 EXPERIMENTAL RESULTS

In this chapter, we present the results of our experiments comparing classical and quantum algorithms for solving the k-SAT problem. The focus will be on the data collected from these experiments, which provide insights into the performance of each approach. For detailed data and analysis, please refer to the Appendices.

5.1 Disclaimer

It is important to note the following disclaimers regarding the accuracy of the data collected.

5.1.1 Limited Number of Problems

The experiments were conducted over a short period, resulting in a limited number of problem instances for testing.

5.1.2 Fixed Shots in Quantum Algorithm

For the quantum algorithm, we used a fixed number of shots (1024) with a custom parameter setup that may not be optimized.

5.1.3 Execution Time Accuracy

The execution times reported for the real quantum hardware may not be entirely accurate, as we utilized time data from the built-in metrics function, which displays times in seconds without floating-point precision

5.1.4 Fixed Optimization Levels

A fixed optimization level (3) was used for both the simulator and actual hardware, which may affect the performance results.

5.2 Conclusion

This study sets out to compare the performance of Grover's algorithm, implemented via Qiskit on both a quantum simulator and real IBM quantum hardware, against a classical brute-force

SAT solver written in C. The primary objective was to evaluate execution time, accuracy, and scalability across different computational platforms for solving SAT problems ranging from 3-SAT to 5-SAT. The experimental results provide valuable insight into the current state of quantum computing and its practical implications for combinatorial problem-solving.

The most immediate finding from this experiment is the clear performance advantage of classical methods. The brute-force solver consistently delivered the fastest execution times and perfect accuracy, even as the SAT problem size increased. This is unsurprising, given that classical processors are highly optimized for such deterministic tasks, and modern CPUs can efficiently explore small search spaces. For 3-SAT to 5-SAT problems, the classical method remains the most viable approach in terms of both speed and correctness.

In contrast, Grover's algorithm, though theoretically offering a quadratic speedup, revealed several limitations in practical implementation. On the Qiskit simulator, the algorithm achieved near-perfect accuracy, closely matching theoretical expectations. However, the execution time on the simulator was significantly slower than the classical method, primarily due to the overhead of simulating quantum gates and circuits on classical hardware. Nonetheless, the simulator provides a reliable environment for testing quantum algorithms in ideal conditions and serves as an essential tool for prototyping and theoretical exploration.

While noise and decoherence remain significant obstacles in practical implementations of Grover's algorithm, recent research suggests that error mitigation techniques such as optimized iteration counts, and noise-aware control methods can improve its resilience on noisy intermediate-scale quantum (NISQ) devices. These approaches help maintain the algorithm's advantages even when hardware noise is present, offering promising pathways to enhance performance as quantum hardware continues to evolve [22]. Nevertheless, more robust quantum error correction and improved hardware stability are crucial for realizing Grover's full potential in solving large-scale SAT problems.

If the coherence time of qubits is significantly increased, quantum circuits can execute deeper algorithms with reduced error accumulation, directly enhancing the performance of Grover's algorithm. Longer coherence times allow more Grover iterations to be performed without decoherence disrupting the computation, thereby improving both the success probability and accuracy of solutions. This advancement would enable quantum processors to handle larger SAT instances more reliably, bringing theoretical speedups closer to practical application. As Preskill (2018) emphasizes, extending coherence times is a crucial step toward transitioning from noisy intermediate-scale quantum (NISQ) devices to fault-tolerant quantum computing capable of solving real-world problems [23].

Given the limitations mentioned above, this paper emphasizes the collection of data to make predictions about the future of quantum computing rather than delivering concrete, definitive results. The experiments conducted serve as a preliminary foundation for future research. The predictions made in this paper are based on extrapolations of the data collected from small-scale experiments, and more comprehensive studies are required once quantum computers with more qubits and greater stability become available.

REFERENCES

[1] J. Gu, "Local search for satisfiability (SAT) problem," *IEEE Transactions on systems, man, and cybernetics,* vol. 23, no. 4, pp. 1108-1129, 1993.

[2] R. Nieuwenhuis, A. Oliveras, and C. Tinelli, "Solving SAT and SAT modulo theories: From an abstract Davis--Putnam--Logemann--Loveland procedure to DPLL (T)," *Journal of the ACM (JACM),* vol. 53, no. 6, pp. 937-977, 2006.

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212-219.

[4] J. Gruska, *Quantum computing*. McGraw-Hill London, 1999.

[5] R. Wille, R. Van Meter, and Y. Naveh, "IBM's Qiskit tool chain: Working with and developing for real quantum computers," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019: IEEE, pp. 1234-1240.

[6] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, "Quantum entanglement," *Reviews of modern physics,* vol. 81, no. 2, pp. 865-942, 2009.

[7] D.-Z. Du and K.-I. Ko, *Theory of computational complexity*. John Wiley & Sons, 2011.

[8] C. H. Papadimitriou, "Computational complexity," in *Encyclopedia of computer science*, 2003, pp. 260-265.

[9] T. J. Schaefer, "The complexity of satisfiability problems," in *Proceedings of the tenth annual ACM symposium on Theory of computing*, 1978, pp. 216-226.

[10] A. Steane, "Quantum computing," *Reports on Progress in Physics,* vol. 61, no. 2, p. 117, 1998.

[11] R. B. Griffiths, "EPR, Bell, and quantum locality," *American Journal of Physics,* vol. 79, no. 9, pp. 954-965, 2011.

[12] A. Mandviwalla, K. Ohshiro, and B. Ji, "Implementing Grover's algorithm on the IBM quantum computers," in *2018 IEEE international conference on big data (big data)*, 2018: IEEE, pp. 2531-2537.

[13] S.-T. Cheng and M.-H. Tao, "Quantum cooperative search algorithm for 3-SAT," *Journal of Computer and System Sciences,* vol. 73, no. 1, pp. 123-136, 2007.

[14] H. Tezuka, K. Nakaji, T. Satoh, and N. Yamamoto, "Grover search revisited: Application to image pattern matching," *Physical Review A,* vol. 105, no. 3, p. 032440, 2022.

[15] P. Costa, A. Sharma, and S. Jordan, *Assessing quantum and classical approaches to combinatorial optimization: Testing quadratic speed-ups for heuristic algorithms*, arXiv preprint arXiv:2412.13035, 2024.

[16] S. Cen, J. Zhang, Y. Song, and H. Zhou, "FastFourierSAT: GPU-Accelerated Continuous Local Search for SAT," in Proceedings of the 40th International Conference on Machine Learning (ICML), 2023, pp. 18751–18767.

[17] F. Arute *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature,* vol. 574, no. 7779, pp. 505-510, 2019.

[18] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, "Determining computational complexity from characteristic 'phase transitions'," Nature, vol. 400, no. 6740, pp. 133–137, 1999.

[19] N. Dilillo, E. Giusto, E. Dri, B. Baheri, Q. Guan, B. Montrucchio, and P. Rech, "Understanding the Effect of Transpilation in the Reliability of Quantum Circuits," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2023, vol. 2: IEEE, pp. 232-235.

[20] S. Resch and U. R. Karpuzcu, "Benchmarking quantum computers and the impact of quantum noise," *ACM Computing Surveys (CSUR),* vol. 54, no. 7, pp. 1-35, 2021.

[21] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.

[22] K. Temme, S. Bravyi, and J. M. Gambetta, "Error mitigation for short-depth quantum circuits," *Phys. Rev. Lett.*, vol. 119, no. 18, p. 180509, 2017.

[23] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018.

APPENDICES

APPENDIX A: Experiment Data on k-SAT Problem Solving

The following tables present the data recorded during experiments on k-SAT problem-solving using quantum and classical methods. The methods compared include the Qiskit simulator, real quantum hardware, and a classical brute-force approach. Please note that the accuracy of the classical brute-force method is consistently 100% due to its exhaustive search process, while the success rate for quantum methods may vary based on hardware and computational parameters.

Problem ID	Type	Expression
01	3-SAT	(a | ~b | c) & (b | ~c | ~d) & (~a | c | d)
02	3-SAT	(a | ~b | c) & (b | c | ~d) & (~a | ~c | d)
03	3-SAT	(a | ~b | c) & (b | ~c | d) & (~a | ~c | d)
04	3-SAT	(a | ~b | c) & (b | d | ~c) & (~a | ~d | c)
05	3-SAT	(a | b | ~c) & (d | ~a | c) & (~b | ~c | d)
06	4-SAT	(a | ~b | c | ~d) & (e | ~a | b | ~c) & (a | ~b | d | e) & (~a | c | ~ d | b) & (a | ~b | c | ~e) & (a | ~b | ~d | ~e) & (b | c | ~e | d)
07	4-SAT	(a | ~b | c | ~d) & (~e | ~a | b | ~c) & (a | ~b | d | e) & (~a | c | ~ d | b) & (a | ~b | c | ~e) & (~a | ~b | ~d | ~e) & (b | c | ~e | d)
08	4-SAT	(a | ~b | c | d) & (~e | ~a | b | ~c) & (a | ~b | d | e) & (a | ~c |

		~ d | b) & (a | ~b | c | ~e) & (~a | ~b | ~d | ~e) & (b | c | ~e | d)
09	4-SAT	(a | ~b | c | d) & (e | ~a | b | c) & (a | ~b | d | e) & (a | ~c | ~ d | b) & (a | ~b | c | ~e) & (a | ~b | ~d | ~e) & (b | c | ~e | d)
10	4-SAT	(a | ~b | c | d) & (~e | ~a | b | c) & (a | ~b | d | e) & (a | ~c | ~ d | b) & (a | ~b | c | ~e) & (a | ~b | d | ~e) & (b | c | ~e | d)
11	5-SAT	(a | ~b | c | d | ~e) & (f | ~a | b | ~c | e) & (a | ~c | d | ~f | b) & (a | ~b | ~c | e | f) & (~b | c | d | ~e | ~f) & (b | c | ~e | ~f | ~a) & (~b | ~c | d | e | f) & (~a | b | ~d | e | ~f) & (~a | ~b | ~c | f) & (a | b | ~e | f)
12	5-SAT	(~a | ~b | c | d | e) & (~f | ~a | b | ~c | e) & (a | ~c | d | ~f | b) & (a | ~b | ~c | e | f) & (~b | c | d | ~e | ~f) & (b | c | ~e | ~f | ~a) & (~b | ~c | d | e | f) & (~a | b | ~d | e | ~f) & (~a | ~b | ~c | f) & (a | ~b | ~e | f)
13	5-SAT	(a | b | c | d | e) & (~f | ~a | b | ~c | e) & (a | c | d | ~f | b) & (a | ~b | ~c | e | f) & (~b | c | d | ~e | ~f) & (b | c | ~e | ~f | ~a) & (~b | ~c | d | e | f) & (~a | b |

		~d | e | ~f) & (~a | ~b | ~c | f) & (a | ~b | ~e | f)
14	5-SAT	(~a | b | c | d | e) & (f | ~a | b | ~c | e) & (a | c | d | f | ~b) & (a | b | c | ~e | ~f) & (~b | c | d | ~e | ~f) & (b | c | ~e | ~f | ~a) & (~b | ~c | d | e | f) & (~a | b | ~d | e | ~f) & (~a | ~b | ~c | f) & (a | b | ~e | ~f)
15	5-SAT	(a | ~b | c | ~d | e) & (f | a | ~b | c | ~e) & (a | ~c | d | f | b) & (~a | b | c | e | ~f) & (~b | c | d | ~e | ~f) & (b | c | ~e | ~f | ~a) & (~b | ~c | d | e | f) & (~a | b | ~d | e | ~f) & (a | b | ~c | f) & (~a | b | ~e | f)

Problem ID	Method	Run Time(s)	Quantum Depth	Qubit Usage	Accuracy	Result	Date(Y-m-d)
01	Classical Brute Force	0.000523	-	-	100.00 %	See Appendix B, Figure B1	2024-11-03
01	Qiskit Simulator	0.002064154	28	4	98.34 %	See Appendix B, Figure B2	2024-11-03

01	ibm_sherbrooke	2	513	4	60.94 %	See Appendix B, Figure B3	2024-11-25
02	Classical Brute Force	0.000694	-	-	100.00 %	See Appendix B, Figure B4	2024-11-03
02	Qiskit Simulator	0.003580891	34	4	97.36 %	See Appendix B, Figure B5	2024-11-03
02	ibm_sherbrooke	2	533	4	60.35 %	See Appendix B, Figure B6	2024-11-25
03	Classical Brute Force	0.001352	-	-	100.00 %	See Appendix B, Figure B7	2024-11-03
03	Qiskit Simulator	0.002265116	58	4	97.66 %	See Appendix B, Figure B8	2024-11-03

03	ibm_sherbrooke	2	668	4	66.60 %	See Appendix B, Figure B9	2024-11-25
04	Classical Brute Force	0.000539	-	-	100.00 %	See Appendix B, Figure B10	
04	Qiskit Simulator	0.002349582	28	4	96.88 %	See Appendix B, Figure B11	2024-11-11
04	ibm_sherbrooke	2	490	4	61.04 %	See Appendix B, Figure B12	2024-11-25
05	Classical Brute Force	0.000894	-	-	100.00 %	See Appendix B, Figure B13	2024-11-19
05	Qiskit Simulator	0.019982796	30	4	97.66 %	See Appendix B, Figure B14	2024-11-19

05	ibm_sherbrooke	2	538	4	61.33 %	Appendix B, Figure B15	2024-11-25
06	Classical Brute Force	0.004512	-	-	100.00 %	Appendix B, Figure B16	2024-11-25
06	Qiskit Simulator	0.019914351	86	5	98.05 %	Appendix B, Figure B17	2024-11-25
06	ibm_sherbrooke	2	1556	5	64.84 %	Appendix B, Figure B18	2024-11-25
07	Classical Brute Force	0.004222	-	-	100.00 %	Appendix B, Figure B19	2024-11-25
07	Qiskit Simulator	0.002695367	224	5	91.70 %	Appendix B, Figure B20	2024-11-25
07	ibm_sherbrooke	2	2848	5	60.74 %	Appendix B, Figure B21	2024-11-25
08	Classical Brute Force	0.006382	-	-	100.00 %	Appendix B,	2024-11-26

						Figure B22	
08	Qiskit Simulator	0.002772032	116	5	98.34 %	Appendix B, Figure B23	2024-11-26
08	ibm_sherbrooke	2	1424	5	61.33 %	Appendix B, Figure B24	2024-11-26
09	Classical Brute Force	0.002811	-	-	100.00 %	Appendix B, Figure B25	2024-11-26
09	Qiskit Simulator	0.0027531	270	5	99.90 %	Appendix B, Figure B26	2024-11-26
09	ibm_sherbrooke	2	3373	5	65.72 %	Appendix B, Figure B27	2024-11-26
10	Classical Brute Force	0.003404	-	-	100.00 %	Appendix B, Figure B28	2024-11-26
10	Qiskit Simulator	0.002476287	116	5	97.17 %	Appendix B, Figure B29	2024-11-26

10	ibm_sherbrooke	3	1555	5	66.50 %	Appendix B, Figure B30	2024-11-26
11	Classical Brute Force	0.011309	-	-	100.00 %	Appendix B, Figure B31	2024-11-27
11	Qiskit Simulator	0.003710317	580	6	98.93 %	Appendix B, Figure B32	2024-11-27
11	ibm_sherbrooke	3	7309	6	68.75 %	Appendix B, Figure B33	2024-11-27
12	Classical Brute Force	0.011647	-	-	100.00 %	Appendix B, Figure B34	2024-11-27
12	Qiskit Simulator	0.004283379	650	6	99.32 %	Appendix B, Figure B35	2024-11-27
12	ibm_sherbrooke	3	7524	6	64.84 %	Appendix B, Figure B36	2024-11-27
13	Classical Brute Force	0.011336	-	-	100.00 %	Appendix B,	2024-11-27

						Figure B37	
13	Qiskit Simulator	0.010710869	574	6	97.07 %	Appendix B, Figure B38	2024-11-27
13	ibm_sherbrooke	3	7128	6	72.27 %	Appendix B, Figure B39	2024-11-27
14	Classical Brute Force	0.011001	-	-	100.00 %	Appendix B, Figure B40	2025-1-13
14	Qiskit Simulator	0.005438446	88	6	99.51 %	Appendix B, Figure B41	2025-1-13
14	ibm_sherbrooke	5	9372	6	67.87 %	Appendix B, Figure B42	2025-1-15
15	Classical Brute Force	0.006669	-	-	100.00 %	Appendix B, Figure B43	2025-1-13
15	Qiskit Simulator	0.002756161	84	6	99.12 %	Appendix B, Figure B44	2025-1-13

15	ibm_sherbrooke	3	7288	6	67.87 %	Appendix B, Figure B45	2025-1-15

APPENDIX B: Experiment Results

This appendix presents a detailed screenshot of the results obtained from the experiments conducted on the k-SAT problem.

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0}
Runtime: 0.000523 seconds
Date: 2024-11-03
```

Figure B1

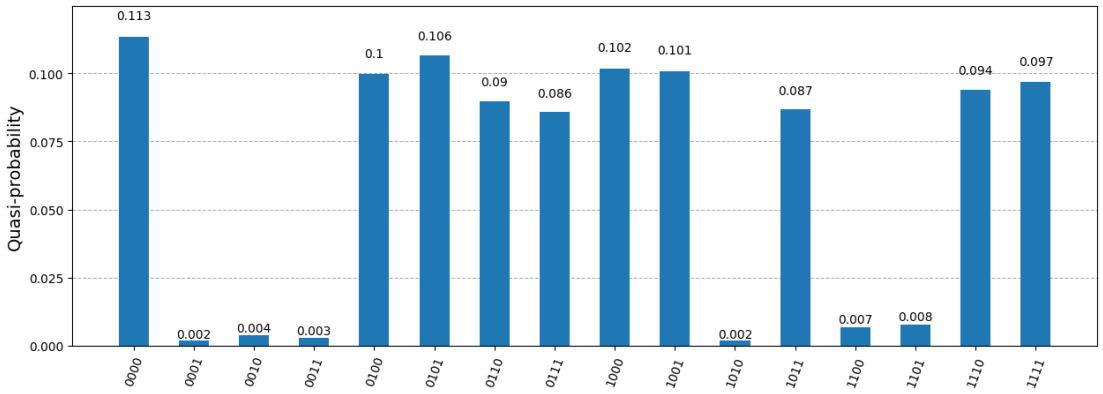


Figure B2

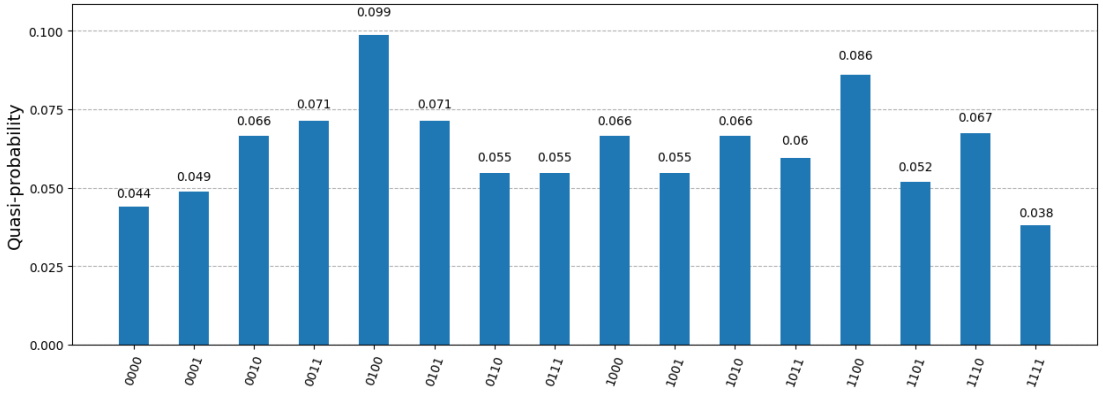


Figure B3

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0}
Runtime: 0.000694 seconds
Date: 2024-11-03
```

Figure B4

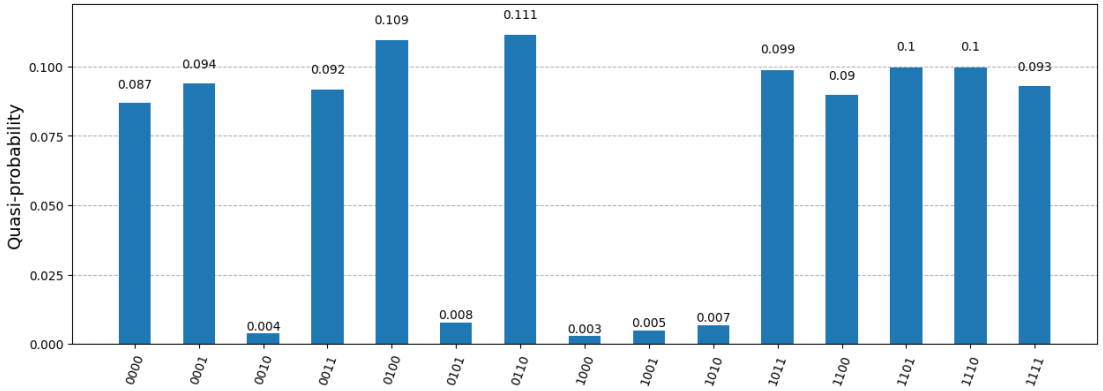


Figure B5

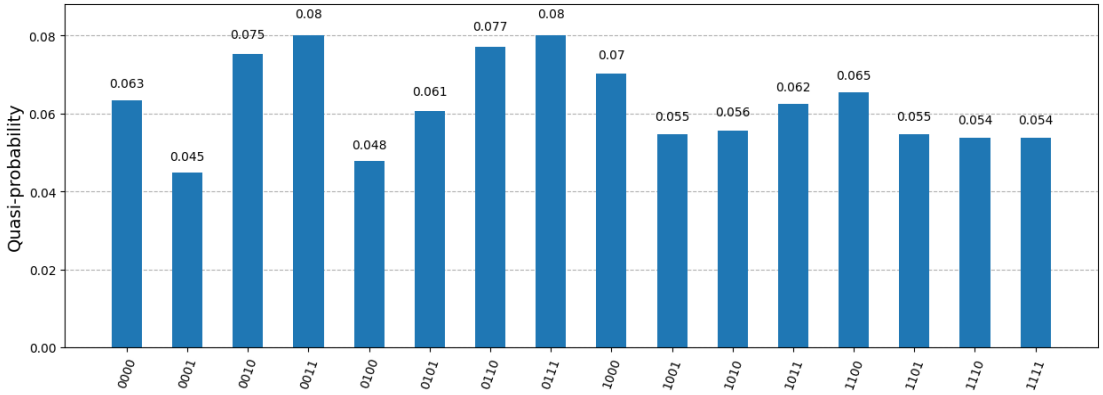


Figure B6

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0}
Runtime: 0.001352 seconds
Date: 2024-11-03
```

Figure B7

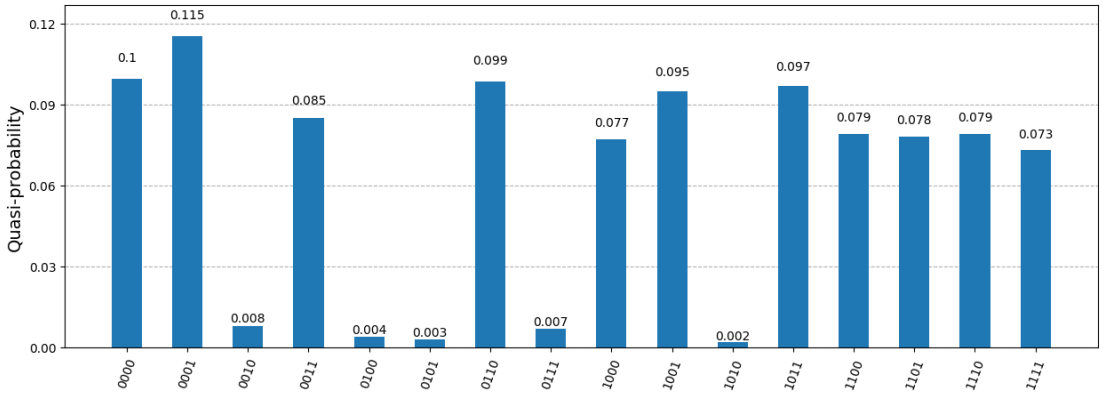


Figure B8

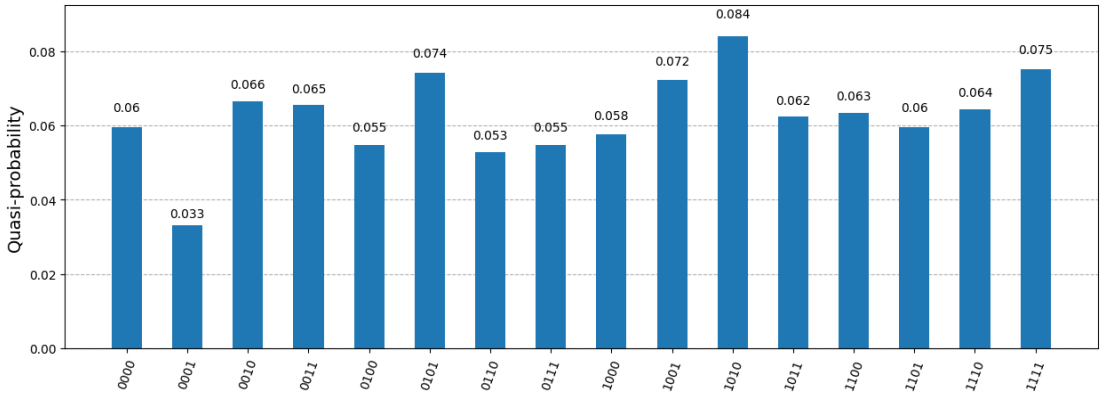


Figure B9

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0}
Runtime: 0.000539 seconds
Date: 2024-11-11
```

Figure B10

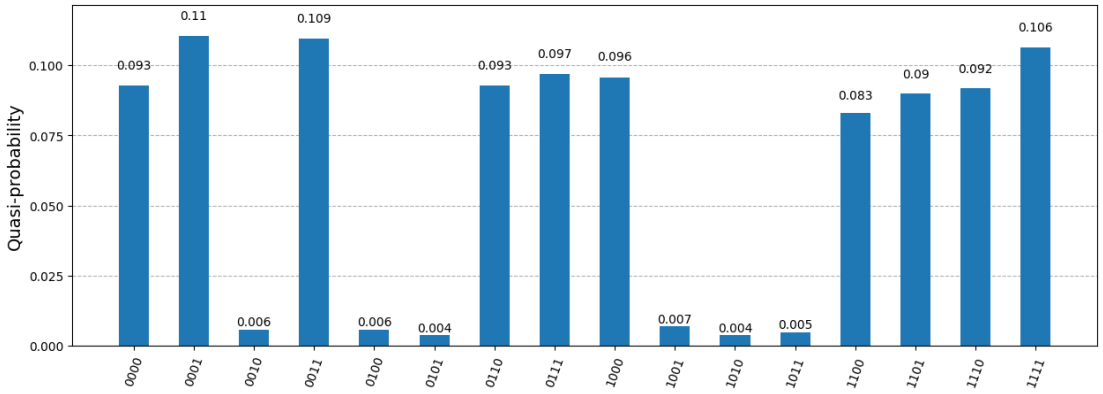


Figure B11

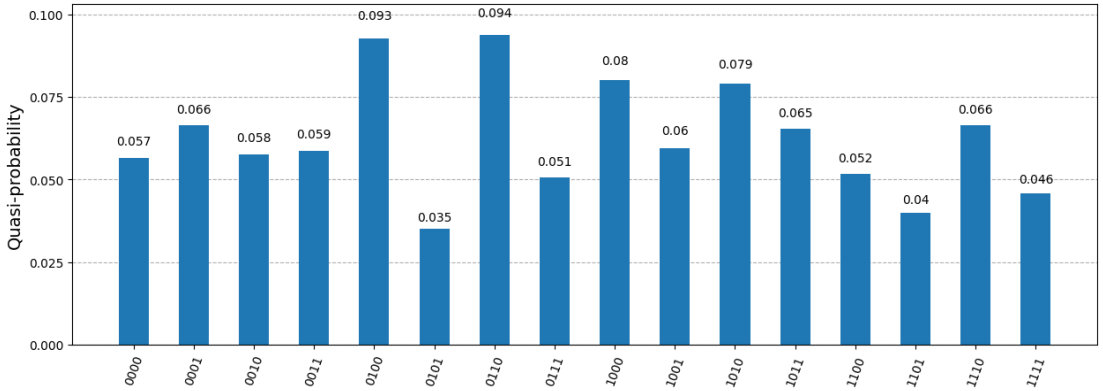


Figure B12

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0}
Runtime: 0.000894 seconds
Date: 2024-11-19
```

Figure B13

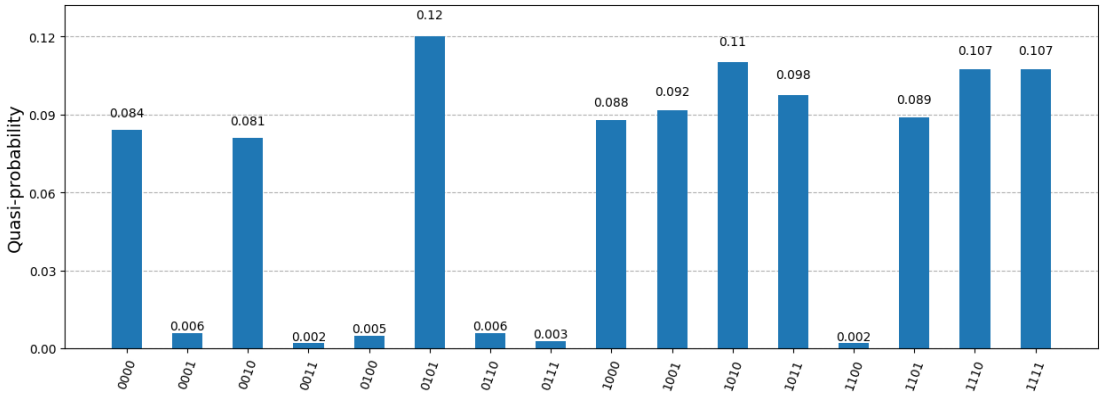


Figure B14

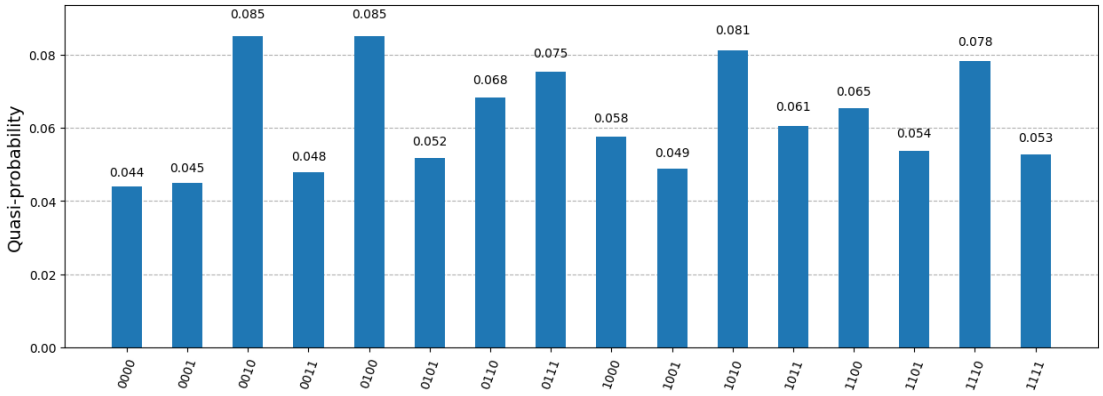


Figure B15

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
Runtime: 0.004516 seconds
Date: 2024-11-25
```

Figure B16

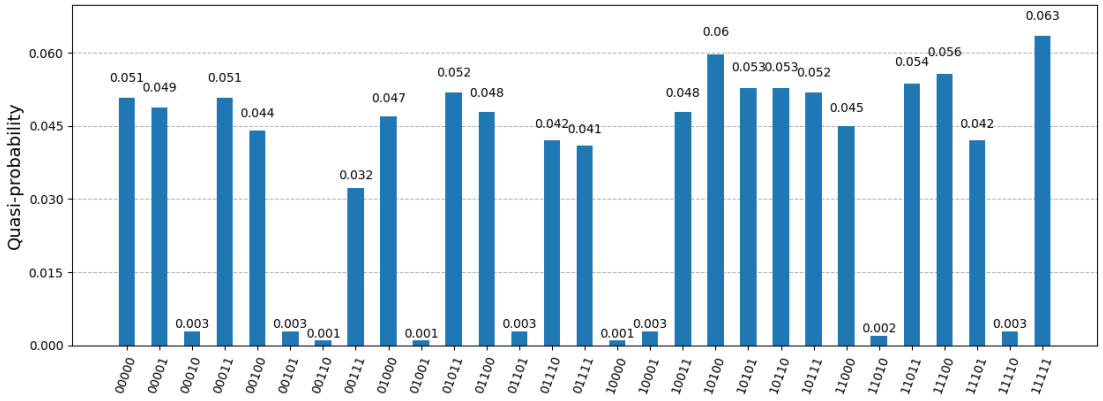


Figure B17

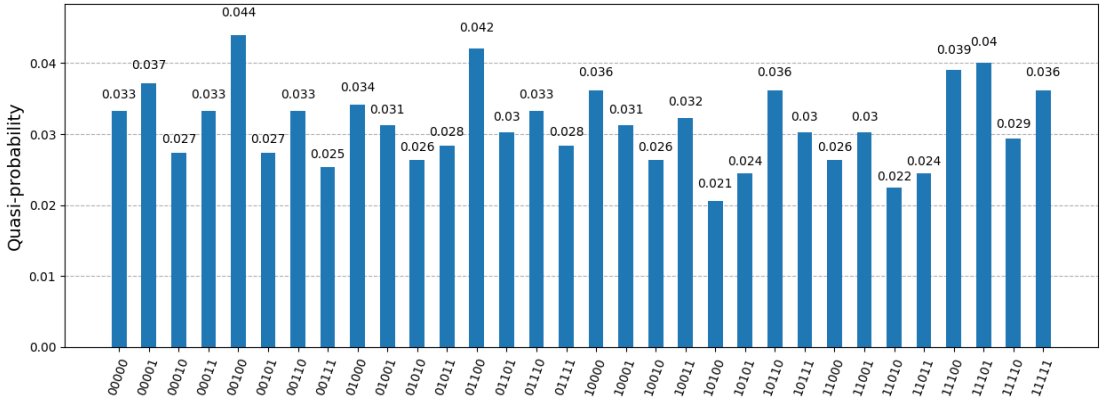


Figure B18

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
Runtime: 0.004222 seconds
Date: 2024-11-25
```

Figure B19

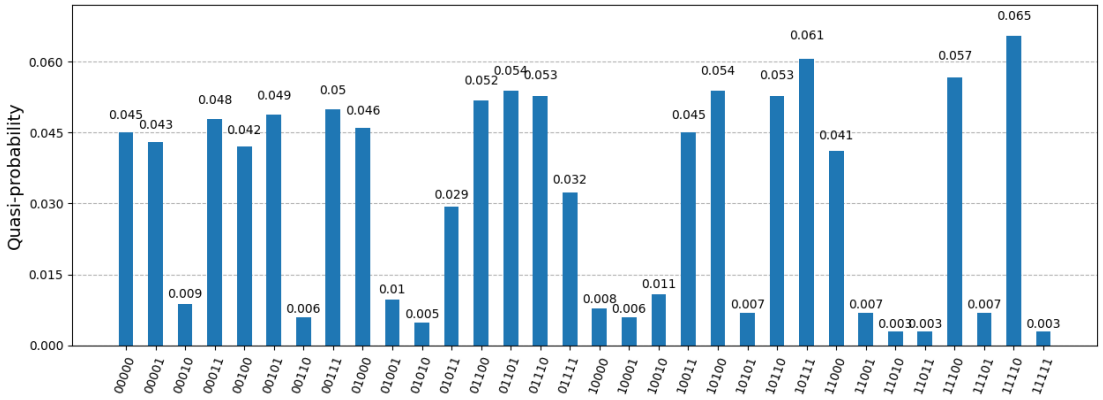


Figure B20

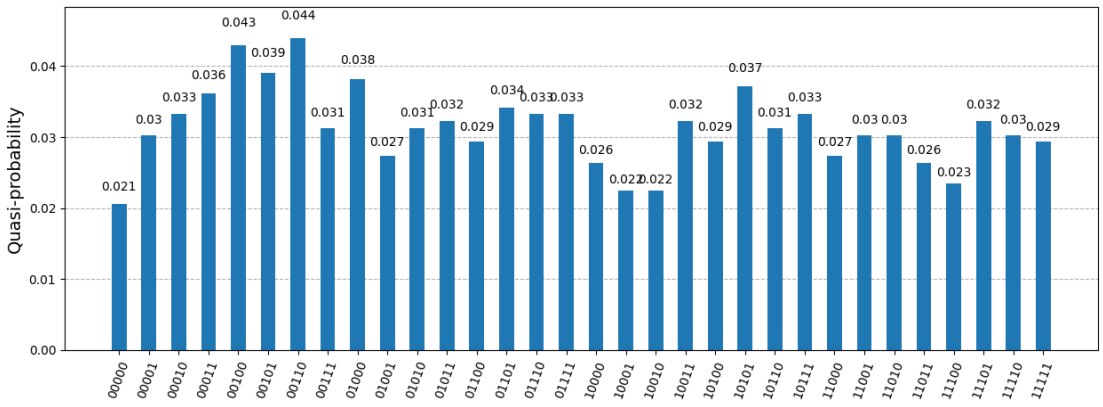


Figure B21

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
Runtime: 0.006382 seconds
Date: 2024-11-26
```

Figure B22

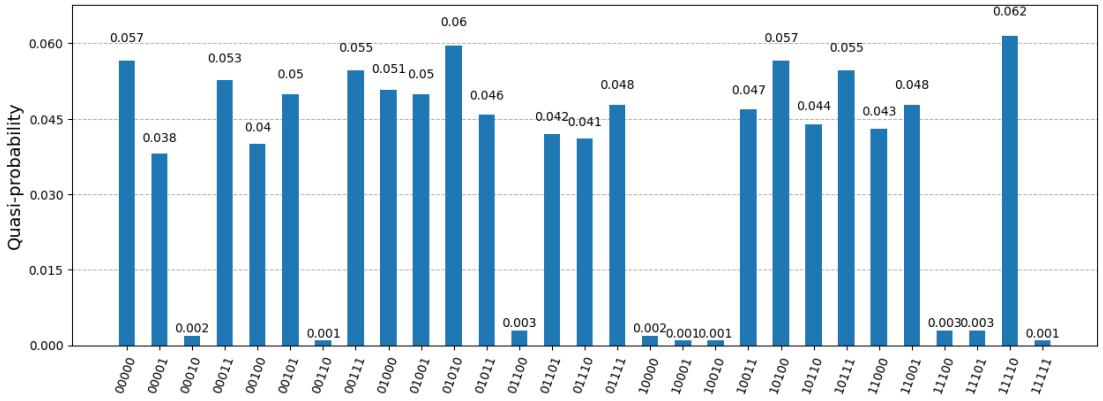


Figure B23

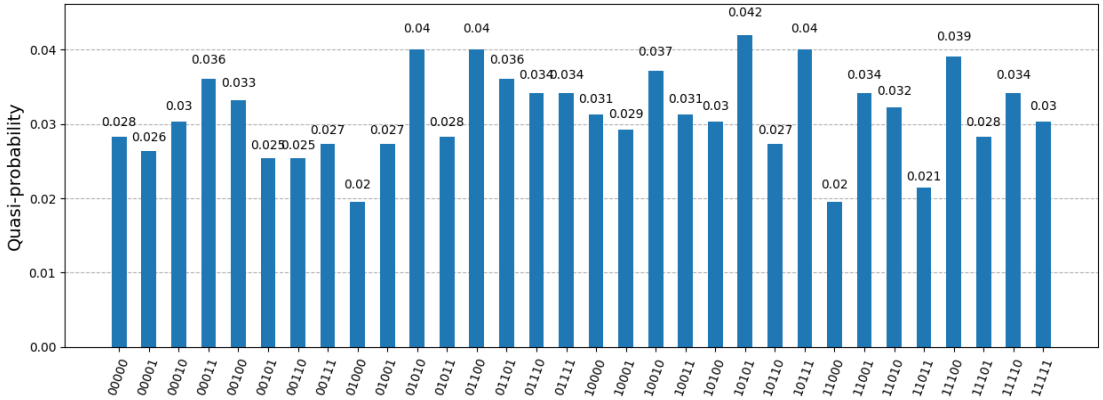


Figure B24

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
Runtime: 0.002811 seconds
Date: 2024-11-26
```

Figure B25

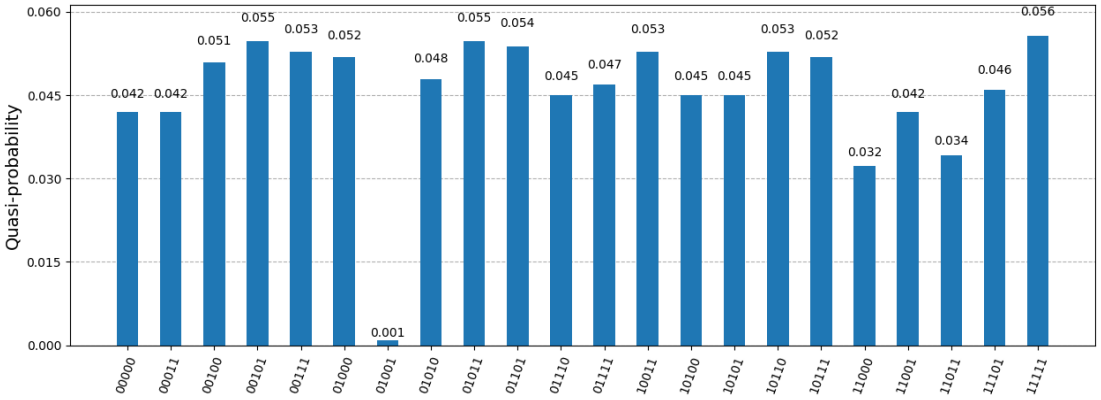


Figure B26

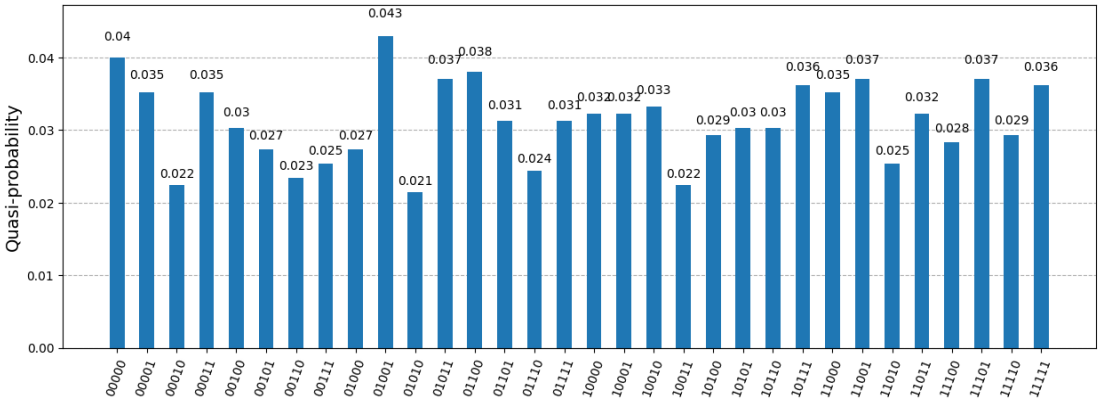


Figure B27

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0}
Runtime: 0.003404 seconds
Date: 2024-11-26
```

Figure B28

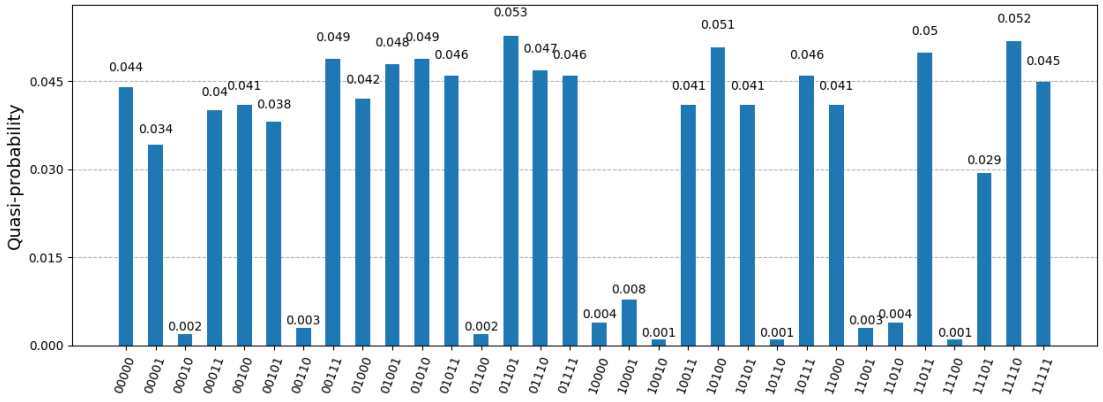


Figure B29

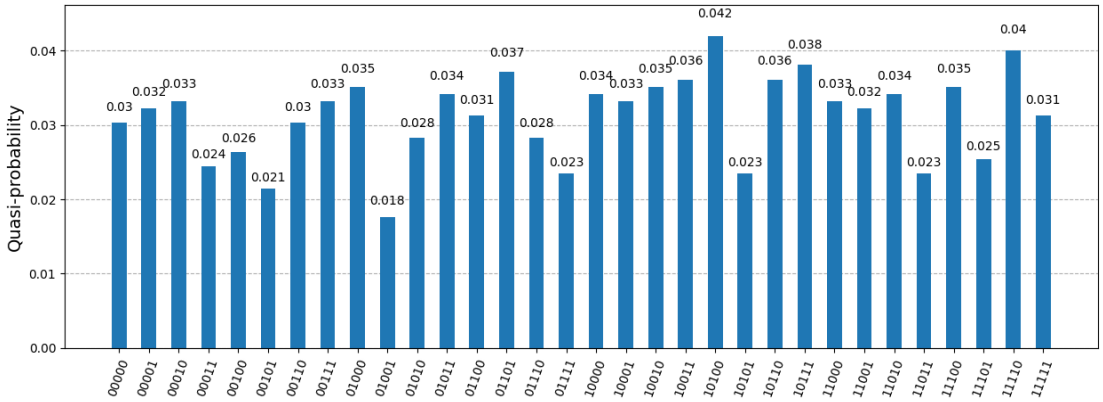


Figure B30

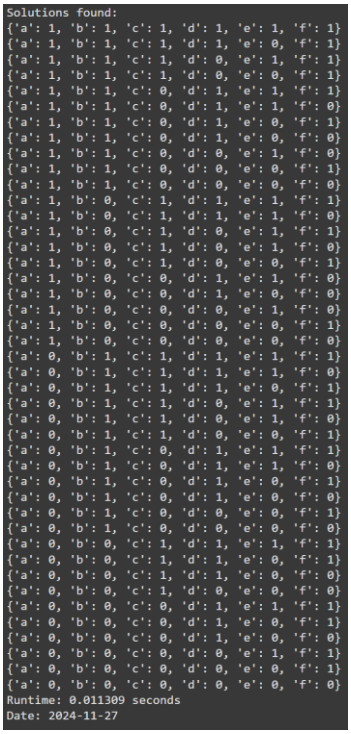


Figure B31

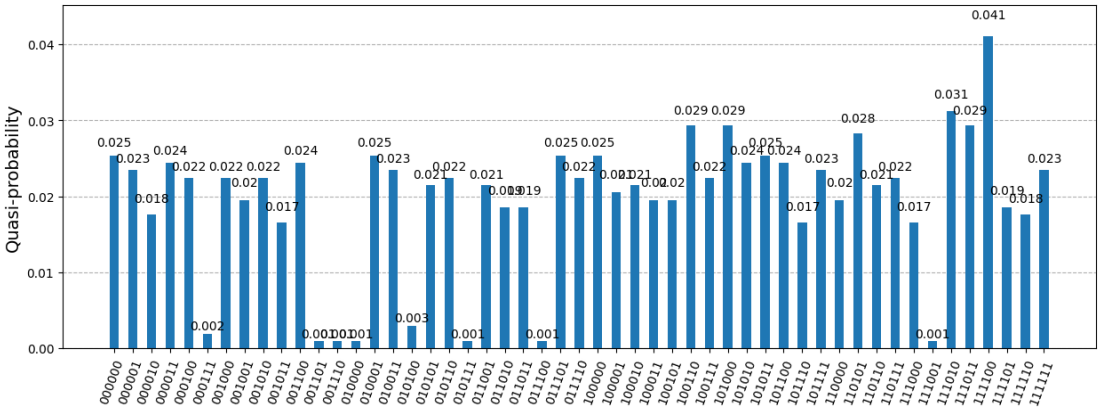


Figure B32

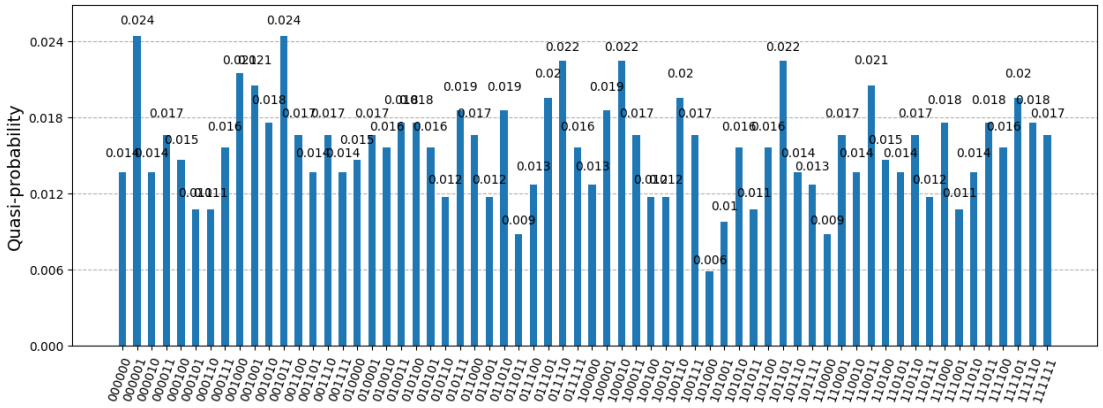


Figure B33

```
Solutions found:
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1, 'f': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 0, 'f': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 1, 'f': 1}
{'a': 1, 'b': 1, 'c': 1, 'd': 0, 'e': 0, 'f': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 1, 'f': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 1, 'f': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0, 'f': 1}
{'a': 1, 'b': 1, 'c': 0, 'd': 1, 'e': 0, 'f': 0}
{'a': 1, 'b': 1, 'c': 0, 'd': 0, 'e': 1, 'f': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 1, 'f': 1}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 1, 'f': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 1, 'e': 0, 'f': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 1, 'f': 0}
{'a': 1, 'b': 0, 'c': 1, 'd': 0, 'e': 0, 'f': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 1, 'f': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 1, 'e': 0, 'f': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 1, 'f': 0}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 1}
{'a': 1, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 0}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 1, 'f': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 1, 'e': 0, 'f': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 1, 'f': 1}
{'a': 0, 'b': 1, 'c': 1, 'd': 0, 'e': 0, 'f': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 1, 'f': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 0, 'f': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 1, 'e': 0, 'f': 0}
{'a': 0, 'b': 1, 'c': 0, 'd': 0, 'e': 0, 'f': 1}
{'a': 0, 'b': 1, 'c': 0, 'd': 0, 'e': 0, 'f': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 1, 'f': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 1, 'f': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 0, 'f': 1}
{'a': 0, 'b': 0, 'c': 1, 'd': 1, 'e': 0, 'f': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 1, 'f': 0}
{'a': 0, 'b': 0, 'c': 1, 'd': 0, 'e': 0, 'f': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1, 'f': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 1, 'f': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0, 'f': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 1, 'e': 0, 'f': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 1, 'f': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 1, 'f': 0}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 1}
{'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 0}
Runtime: 0.011647 seconds
Date: 2024-11-27
```

Figure B34

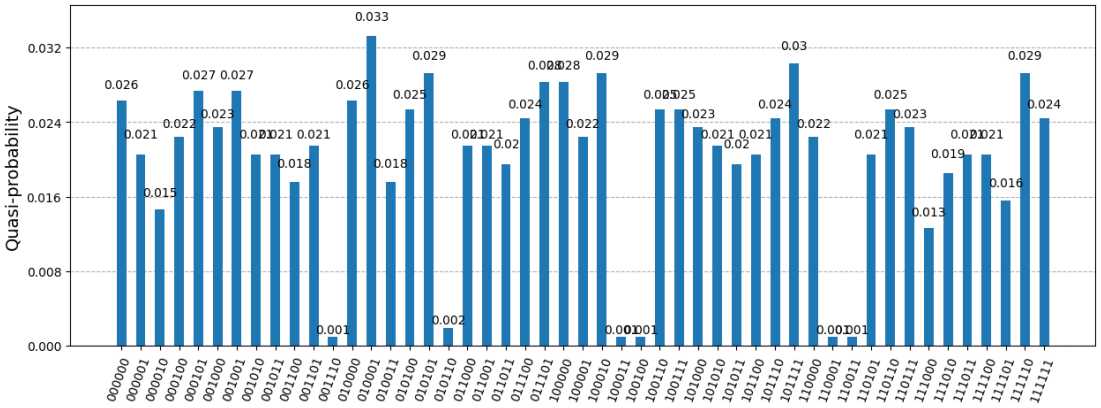


Figure B35

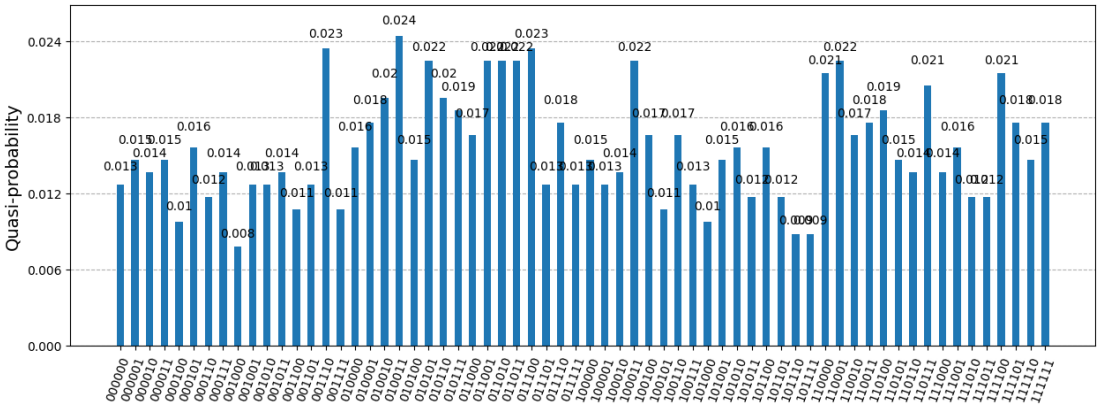


Figure B36

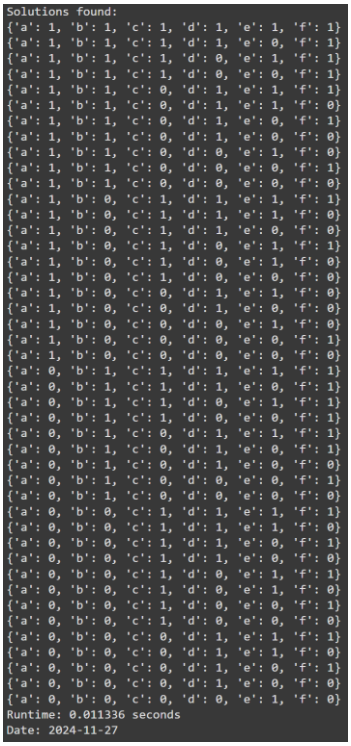


Figure B37

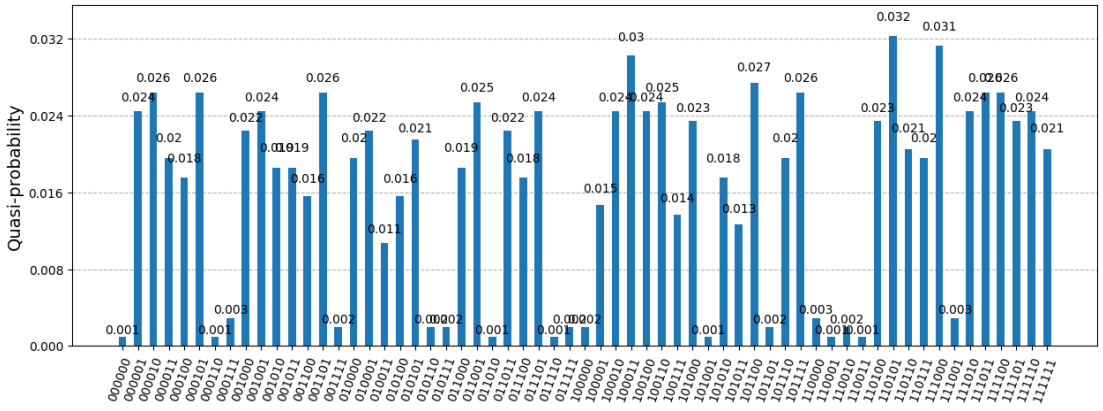


Figure B38

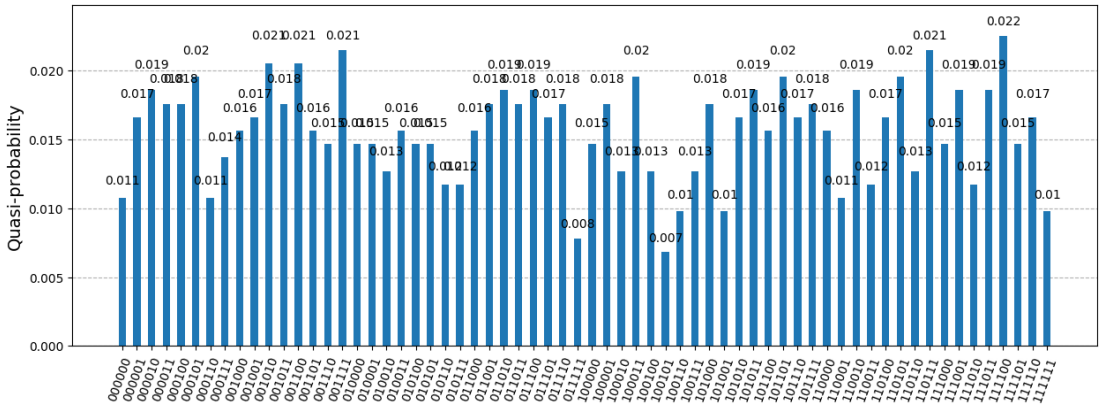


Figure B39

Figure B40

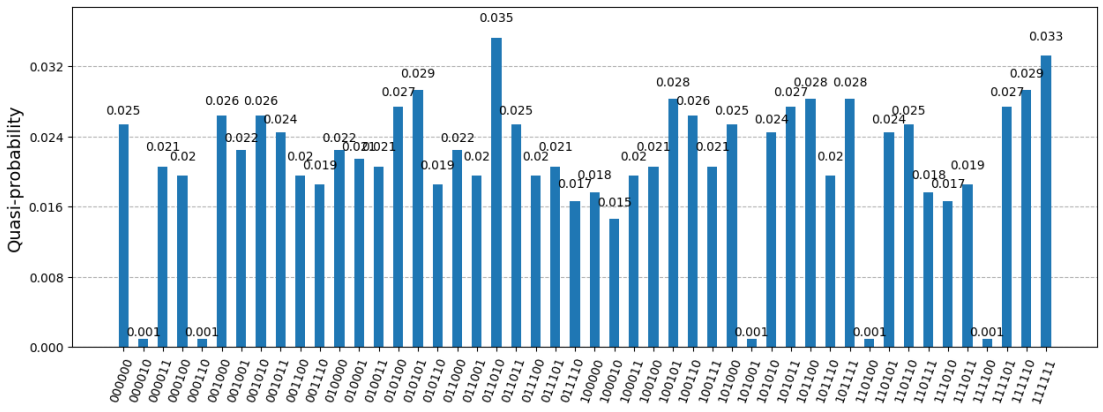


Figure B41

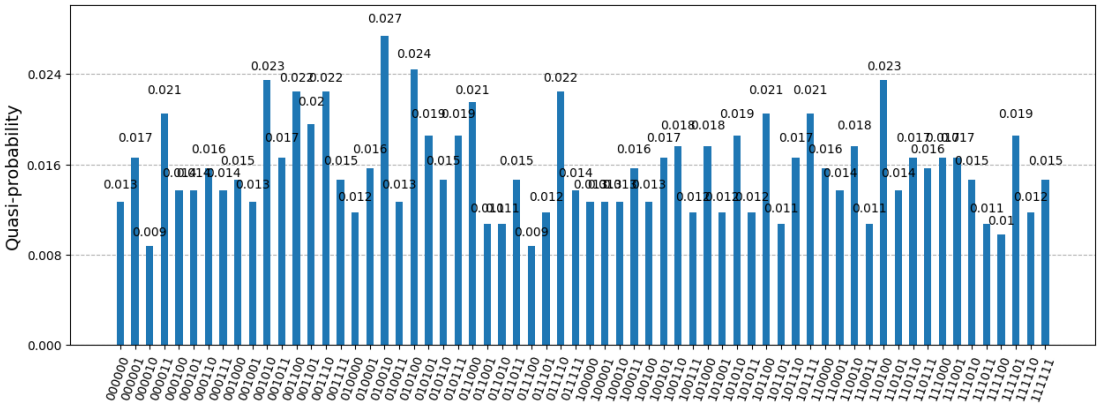


Figure B42

Figure B43

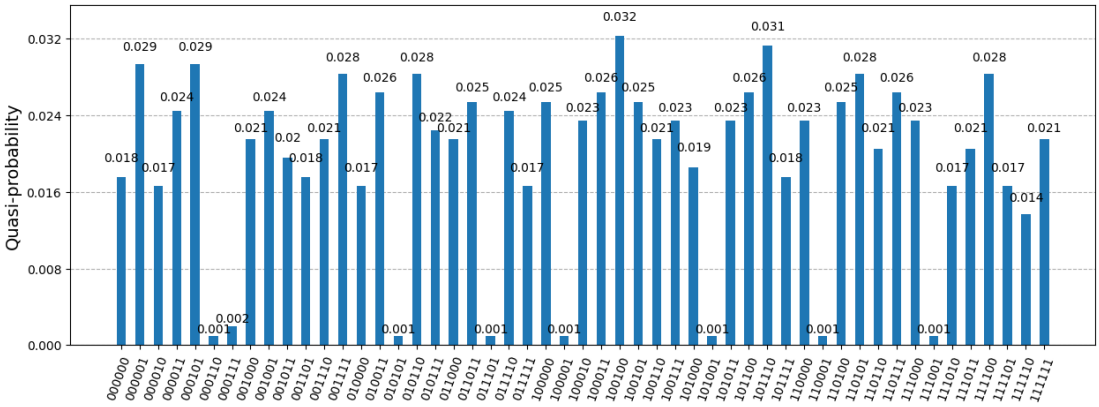


Figure B44

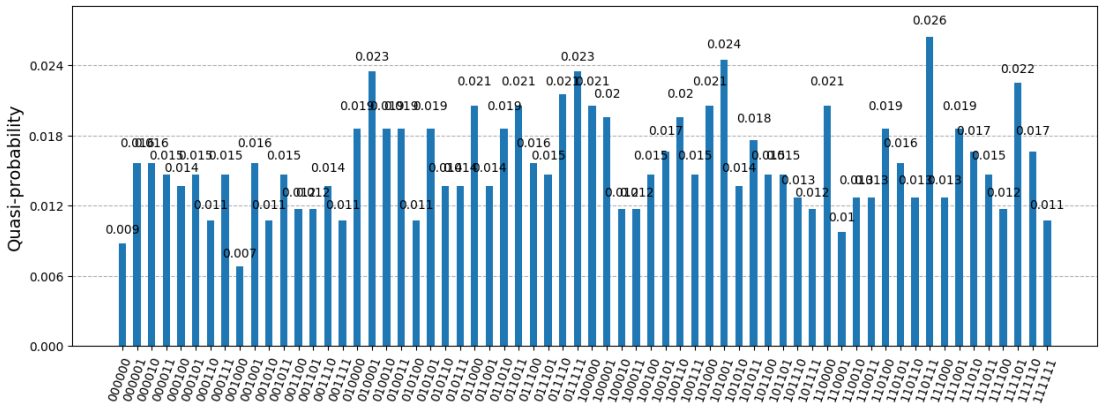


Figure B45

VITA

NAME	Jirapas Unison Jipipob
DATE OF BIRTH	5 October 2001
PLACE OF BIRTH	Nonthaburi
INSTITUTIONS ATTENDED	SIIT, Thammasat University
HOME ADDRESS	10/45 Soi Chomrom 3, Chaengwattana 14, Lak Si, Tung song hong, Bangkok 10210