UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Paulo André Viana de Araujo

A Quantum Genetic Algorithm Framework for the MaxCut Problem

Recife

2025

Paulo André Viana de Araujo

A Quantum Genetic Algorithm Framework for the MaxCut Problem

Trabalho apresentado ao Programa de Pósgraduação em Ciências da Computação do Centro de Informática da Universidade Federal de Pernambuco, como um dos requisitos para a obtenção do grau de Mestre em Ciências da Computação.

Área de Concentração: Inteligência Computacional

Orientador (a): Fernando Maciano de Paula Neto

Recife

2025

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Viana de Araújo, Paulo André.
A Quantum Genetic Algorithm Framework For The MaxCut Problem / Paulo André Viana de Araújo. - Recife, 2025.
58f.: il.

Dissertação (Mestrado) - Universidade Federal de Pernambuco, Centro de Informática, Programa de Pós-Graduação em Ciências da Computação, 2025.
Orientação: Fernando Maciano de Paula Neto.
Inclui referências.

1. Computação quântica; 2. Otimização combinatória; 3. Graph Theory. I. Paula Neto, Fernando Maciano de. II. Título.

UFPE-Biblioteca Central

Paulo André Viana de Araujo

“A Quantum Genetic Algorithm Framework For The MaxCut Problem”

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional

Aprovado em: 30/01/2025.

BANCA EXAMINADORA

AGRADECIMENTOS

Agradeço primeiramente à Deus por ter me guiado durante toda minha trajetória, nos piores aos melhores momentos, sempre estado a meu lado. Em seguida, agradeço a meu Pai, a quem sempre me deu suporte, mesmo nos momentos mais difíceis. Agradeço também ao Professor e meu orientador, Fernando Maciano de Paula Neto, por ter me orientado ao longo do mestrado, que teve paciência durante diversos problemas que encontrei durante o programa e que me possibilitou crescer como pesquisador. Agradeço aos meus amigos e colegas que trabalharam diretamente comigo e a todos que conheci e estiveram junto a mim durante esta jornada. Por fim, agradeço à CAPES e o CNPq pelo apoio financeiro à pesquisa desenvolvida.

ABSTRATO

O problema do MaxCut é um problema fundamental da Otimização Combinatória, com implicações significativas em diversas áreas, como logística, projeto de redes e física estatística. O algoritmo proposto representa uma abordagem inovadora que equilibra rigor teórico com escalabilidade prática. O método introduz um Algoritmo Genético Quântico (QGA) baseado em um arcabouço evolucionário com Grover e princípios de divisão e conquista. Ao particionar grafos em subgrafos manejáveis, otimizá-los de forma independente e aplicar contração de grafos para combinar as soluções, o método explora a simetria binária inerente ao MaxCut para garantir um desempenho mais eficiente e robusto em termos de aproximação. A análise teórica estabelece a base para um desempenho superior do algoritmo, enquanto as avaliações empíricas fornecem evidências quantitativas de sua eficácia. Em grafos completos, o método proposto alcança consistentemente os valores ótimos verdadeiros do MaxCut, superando a abordagem por Programação Semidefinida (SDP), que fornece até 99,7% da solução ótima em grafos maiores. Em grafos aleatórios de Erdős–Rényi, o QGA apresenta desempenho competitivo, atingindo soluções medianas dentro de 92–96% dos resultados da SDP. Esses resultados destacam o potencial do arcabouço QGA para fornecer soluções competitivas, mesmo sob restrições heurísticas, ao mesmo tempo em que demonstram sua promessa de escalabilidade conforme o hardware quântico evolui.

Palavras-chaves: MaxCut. Computação Quântica. Algorítmo Genético Quânticdo. COtimização Combinatória. Grover. Grafos.

ABSTRACT

The MaxCut problem is a fundamental problem in Combinatorial Optimization, with significant implications across diverse domains such as logistics, network design, and statistical physics. The algorithm represents innovative approaches that balance theoretical rigor with practical scalability. The proposed method introduces a Quantum Genetic Algorithm (QGA) using a Grover-based evolutionary framework and divide-and-conquer principles. By partitioning graphs into manageable subgraphs, optimizing each independently, and applying graph contraction to merge the solutions, the method exploits the inherent binary symmetry of MaxCut to ensure a more efficient and robust approximation performance. Theoretical analysis establishes a foundation for a better performance of the algorithm, while empirical evaluations provide quantitative evidence of its effectiveness. On complete graphs, the proposed method consistently achieves the true optimal MaxCut values, outperforming the Semidefinite Programming (SDP) approach, which provides up to 99.7% of the optimal solution for larger graphs. On Erdős-Rényi random graphs, the QGA demonstrates competitive performance, achieving median solutions within 92-96% of the SDP results. These results showcase the potential of the QGA framework to deliver competitive solutions, even under heuristic constraints, while demonstrating its promise for scalability as quantum hardware evolves.

Keywords: MaxCut. Quantum Computing. Quantum Genetic Algorithm. Combinatorial Optimization. Grover. Graphs.

LISTA DE FIGURAS

Figura 1 – Diagram of the proposed framework. The fitness operator, oracle and the graph partitioning/contraction are original contributions specific to the MaxCut problem, where the remaning blocks follow the general framework. 37

LISTA DE TABELAS

Tabela 1 – Key differences between GA, QGA, and the proposed RQGA model 38
Tabela 2 – Comparison of QGA and SDP on Complete Graphs. 49
Tabela 3 – Comparison of QGA Run 1 and SDP on Erdős-Rényi Random Graphs. . . . 50
Tabela 4 – Comparison of QGA Best runs and SDP on Erdős-Rényi Random Graphs. . 51

LISTA DE ABREVIATURAS E SIGLAS

GA	Genetic Algorithm
GW	Goemans-Williamson
MaxCut	Maximum Cut
NISQ	Noisy Intermediate-Scale Quantum
QAOA	Quantum Approximate Optimization Algorithm
QEA	Quantum Evolutionary Algorithm
QGA	Quantum Genetic Algorithm
QUBO	Quadratic Unconstrained Binary Optimization
RQGA	Reduced Quantum Genetic Algorithm
SDP	SemiDefinite Programming

SUMÁRIO

Lista de Figuras . . . 7

1 INTRODUCTION . . . 12

1.1 RELATED WORKS . . . 14

1.2 OBJECTIVES . . . 15

1.2.1 Other Works . . . 16

1.2.2 Work Overview . . . 16

2 LITERATURE REVIEW . . . 17

2.1 QUANTUM COMPUTING . . . 17

2.2 GENETIC ALGORITHM . . . 27

2.3 THE MAXIMUM CUT . . . 29

2.4 THE QUANTUM GENETIC ALGORITHM . . . 31

3 QGA FRAMEWORK FOR THE MAXCUT PROBLEM . . . 34

3.1 OVERVIEW OF THE FRAMEWORK . . . 34

3.1.1 Standard QGA Framework . . . 34

3.1.2 Enhanced Approach . . . 35

3.1.2.1 Superposition and Fitness Register . . . 36

3.1.2.2 Optimization with Grover's Algorithm . . . 37

3.1.3 Key Differences . . . 37

3.1.4 Circuit Initialization . . . 39

3.1.5 Fitness Subcircuit . . . 40

3.1.6 Oracle Subcircuit . . . 41

3.1.6.1 The Quantum Adder . . . 42

3.1.7 Grover Diffuser . . . 42

3.2 DIVIDE-AND-CONQUER HEURISTIC FOR QGA . . . 43

3.2.1 Complexity Analysis . . . 46

4 RESULTS AND CONCLUSION . . . 48

4.1 EXPERIMENTAL SETUP . . . 48

4.2 RESULTS FOR SMALL GRAPHS . . . 48

4.3 RESULTS FOR COMPLETE GRAPHS . . . 49

4.4 RESULTS FOR ERDŐS-RÉNYI RANDOM GRAPHS . . . 49

4.5 ANALYSIS AND DISCUSSION 52

4.5.1 Complete Graphs 52

4.5.2 Erdős-Rényi Random Graphs 52

4.6 CONCLUSION 53

BIBLIOGRAPHY 56

1 INTRODUCTION

When David Hilbert proposed the *Entscheidungsproblem* (decision problem) in 1928, he sought an algorithm to determine whether a given statement in first-order logic could be proven. This foundational question aimed to formalize mathematics under a complete and decidable system.

However, in 1936, Alan Turing published a paper introducing the concept of the famous Turing Machine. He proved that no such general algorithm exists, thereby showing that the Entscheidungsproblem has no solution. Since then, all algorithms, from Euclidean to modern ones, share the same underlying mathematical abstraction: they are computable by a Turing Machine.(TURING, 1936)

Alan Turing was well-versed in the emerging fields of his time, particularly quantum physics. Nevertheless, as we now know, Turing did not attempt to formalize a "quantum version"of his machine, despite being aware of the quantum revolution occurring in physics.

Since the publication of Turing's paper, Computability Theory has evolved through the independent work of researchers such as Turing, Church, Kleene, and Post. This led to the Church-Turing Thesis: any function computable by a discrete, finite system is computable by a Turing Machine. (CHURCH, 1936) To this day, the Church-Turing Thesis remains universally accepted. (SIPSER, 2012) However, its polynomial extension—that every efficiently computable function is equivalently efficiently computable by a Turing Machine—is being tested by the development of quantum computers. Notably, Shor's quantum algorithm for integer factorization was a groundbreaking step.

Definability is an important property in classical computation, as computer processes can often be formalized as Boolean functions. However, definability is distinct from computability, as there exist many non-computable numbers. (SIPSER, 2012) Quantum computers promise to impact our notion of definability rather than computability. Quantum computers do not expand computability in the classical sense,anything a quantum computer can compute is still Turing-computable. However, they do expand what is efficiently definable or tractably distinguishable. For instance, problems in the class BQP (Bounded-Error Quantum Polynomial Time) might be intractable classically, but efficiently solvable on a quantum computer. Thus, quantum computation influences what structures, patterns, or relationships can be practically expressed or accessed, even if the underlying functions remain classically computable. (NIELSEN; CHUANG, 2012)

As quantum computers become more viable, we may witness significant advancements in how we process classical data—not by solving previously uncomputable problems, but by accessing structures and regularities that are hard to define or detect classically. Moreover, with quantum data encoded as quantum states, quantum computation may allow us to define and manipulate complex information that lies beyond classical descriptive tools. In this way, the power of quantum computation is not in redefining what is computable, but in expanding the boundaries of what is efficiently definable, expressible, and observable in both classical and quantum domains.

This brings us to the domain of combinatorial optimization, a branch of mathematics and computer science focused on finding the best solution from a finite set of possibilities, often subject to constraints. One central problem in this field is the MaxCut problem, which seeks to partition the vertices of a graph into two subsets such that the number of edges between the subsets is maximized. Formally, for a graph $G = (V, E)$, the goal is to maximize:

$$\text{Cut}(S) = \sum_{(u,v)\in E} \delta(u \in S, v \notin S),$$

where S is a subset of V and δ is an indicator function. MaxCut is computationally challenging, classified as NP-hard, which motivates the exploration of novel computational techniques.

Quantum computing offers new avenues for addressing combinatorial optimization problems like MaxCut. By leveraging the principles of superposition and entanglement, quantum algorithms can evaluate multiple solutions simultaneously. For MaxCut, quantum approaches often encode the problem as a Quadratic Unconstrained Binary Optimization (QUBO) problem. This encoding allows quantum systems, such as those employing Grover's search or adiabatic quantum optimization, to explore the solution space efficiently. These techniques exploit the symmetry of the MaxCut problem, represented mathematically by the $\mathbb{Z}_2$ group, to ensure efficient exploration and solution identification.

Genetic algorithms (GAs) are bio-inspired optimization methods that mimic the process of natural evolution. They maintain a population of candidate solutions, iteratively improving them through selection, crossover, and mutation. Fitness functions guide the evolution by quantifying how close a candidate is to the optimal solution. While effective for many problems, classical GAs often struggle with scalability and convergence, especially in high-dimensional solution spaces.

Quantum Genetic Algorithms (QGAs) combine the principles of quantum computing with the evolutionary mechanisms of GAs. Instead of explicitly maintaining a population, QGAs use quantum superposition to encode all possible solutions within a single quantum state. Quantum gates replace classical genetic operators, and Grover's algorithm is often employed to enhance the selection process by amplifying high-fitness solutions. This quantum parallelism allows QGAs to explore solution spaces more efficiently than their classical counterparts. (UDRESCU; PRODAN; VLăDUţIU, 2006)

The Reduced Quantum Genetic Algorithm (RQGA) framework refines the QGA framework by eliminating unnecessary genetic operators such as cross-over and mutation. Instead, RQGA focuses on Grover's search to identify high-fitness solutions directly. The algorithm begins by encoding the entire solution space into a quantum superposition. A quantum oracle marks the states corresponding to high fitness, and Grover iterations amplify these states. This process significantly reduces computational overhead while maintaining robust optimization performance. (ARDELEAN; UDRESCU, 2022)

The RQGA variation presented here is a particularly tailored version for the MaxCut problem. Although there are QGAs that aim to solve the MaxCut problem, they do not offer competitive results; on the other hand, the original RQGA framework was not designed for problems with the nature of the MaxCut problem. Also, with the addition of a divide-and-conquer approach, the ability to exploit the problem's binary symmetry is expanded for higher graphs. By partitioning the graph into smaller subgraphs, the algorithm optimizes each independently and merges the solutions through graph contraction techniques. This ensures that the global solution respects the structural properties of the original graph. Empirical studies suggest that the variation here presented consistently achieves the true value for a cut for small graphs and graphs that preserve symmetry when partitioned, highlighting its potential to redefine the computational boundaries of combinatorial optimization.

1.1 RELATED WORKS

The study of MaxCut algorithms begins with classical approaches such as the Goemans-Williamson (GW) algorithm, which uses SemiDefinite Programming (SDP) techniques to achieve a guaranteed approximation ratio of 0.878. Although computationally intensive, the GW algorithm laid the foundation for advancements in approximation methods. Spectral algorithms later emerged as a faster alternative to SDP with slightly lower approximation guarantees, offering practical trade-offs between speed and accuracy.

Building on these classical methods, quantum optimization techniques have introduced significant innovations. The Quantum Approximate Optimization Algorithm (QAOA) applies quantum variational principles to map the MaxCut problem onto a quantum Hamiltonian. QAOA represents an important step toward harnessing Noisy Intermediate-Scale Quantum (NISQ) devices for combinatorial optimization. Despite its promise, QAOA is limited by scalability issues, as the number of required qubits grows linearly with the graph size. The QAOA-in-QAOA (QAOA^2) method addressed this limitation by employing a divide-and-conquer strategy, partitioning large graphs into subgraphs solvable on smaller quantum devices. By leveraging $\mathbb{Z}_2$ symmetry, QAOA^2 efficiently merges local solutions into a global one.

Quantum Genetic Algorithms (QGAs) build on the principles of quantum optimization by combining evolutionary strategies with quantum computing. Unlike QAOA, QGAs integrate selection mechanisms inspired by natural selection with quantum superposition and Grover's search. This hybrid approach enables efficient exploration of high-dimensional solution spaces.

The Reduced Quantum Genetic Algorithm (RQGA) refines the QGA framework by removing traditional genetic operators such as crossover and mutation. Instead, RQGA relies on Grover's search to amplify high-fitness solutions within a quantum superposition. This simplification enhances computational efficiency while preserving the algorithm's robustness.

1.2 OBJECTIVES

The objectives of this research is to develop and explore the advantages and limitations of a Quantum Genetic Algorithm adapted for the specific Combinatorial Optimization problem of the MaxCut. More specifically:

- To establish and explore theoretical foundations for the use of quantum evolutionary techniques in combinatorial optimization problems, showing approximation guarantees and the potential of its applicability for lange-scale graphs.

- To develop a Grover-based framework of the Quantum Genetic Algorithm for the MaxCut problem and analyze its advantages and limitations.

- To extend its analysis and applicability using graph contraction has a divide-and-conquer heuristic.

1.2.1 Other Works

During period of his studies, the student worked full-time for Accenture Brasil, producing a patented work called *"Dual-Unit Quantum Genetic Algorithm,"* under the registration *23-387/04810-PR-US*, which is a Hybrid Quantum Genetic Algorithm. He also participated in the Liga Acadêmica de Computação e Informação Quântica (LACIQ) as the director of the *Quantum Cryptography* sector. Meanwhile, the student was invited to participate at the *World Youth Festival 2024* (WYF 2024) in Sochi, Russia. With all costs paid. Subsequently, he was invited to vist the *Russian Quantum Center* (RQC) in Moscow, Russia.

1.2.2 Work Overview

The body of this document is organized into chapters. Chapter 2 gives an overview of Quantum Computing in general, introducing Quantum Gates, Grover's Algorithm and their related concepts. Also it develops the general framework for the Genetic Algorithm and its limitations, following as a presentation of the MaxCut problem and its historical development, as well as some necessary theoretical results and classical algorithms. The general framework of a Quantum Genetic Algorithm is also introduced. Chapter 3 provides the Grover-based framework and the MaxCut QGA developed using it, also it explains the divide-and-conquer heuristic utilizing graph contraction. Finally it gives the overview of the body of the work. Chapter 4 explains the results, challenges and conclusion of this work.

2 LITERATURE REVIEW

2.1 QUANTUM COMPUTING

Basic Concepts of Quantum Computing

Differently from Classical Computers, Quantum computers process information using quantum bits, or qubits, which can represent by quantum physical states $|0\rangle$, $|1\rangle$ using the Dirac notation. More generally, any single-qubit quantum computational state can be represented as a unit vector in a two-dimensional complex Hilbert space: (NIELSEN; CHUANG, 2012)

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

where $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ are the computational basis states, and $\alpha, \beta \in \mathbb{C}$ are complex amplitudes satisfying the normalization condition $|\alpha|^2 + |\beta|^2 = 1$.

2. Multi-Qubit Systems and Entanglement

For a system of n qubits, the state space is the tensor product of individual qubit spaces, resulting in a 2^n-dimensional complex vector space. A general state of an n-qubit system is:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} c_i|i\rangle$$

where $|i\rangle$ denotes the computational basis states ranging from $|00\ldots0\rangle$ to $|11\ldots1\rangle$, and $c_i \in \mathbb{C}$ are complex coefficients satisfying $\sum_{i=0}^{2^n-1} |c_i|^2 = 1$.

A notable feature of multi-qubit systems is *entanglement*, a phenomenon where qubits become correlated such that one qubit state cannot be independent of the system. For example, the Bell state:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

is an entangled state of two qubits.

Quantum gates are applied to qubits via unitary operators. A quantum circuit comprises a sequence of such gates, analogous to logical gates in classical computation. Some commonly used quantum gates include:

- **Hadamard Gate (H):** Creates superposition. $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.
- **Pauli Gates (X, Y, Z):** Perform bit flips and phase shifts.
- **Controlled-NOT (CNOT):** Flips the target qubit conditioned on the control qubit if they are not in a ground state.

Quantum Projective Measurement

The process of measurement collapses the general quantum state of a qubit into one of its basis states, $|0\rangle$ or $|1\rangle$, with probabilities determined by the state's amplitudes. The measurement operation is the only one which is not continuos, after measurement, the qubit instantaneously collapses to the observed state, irreversibly losing superposition.

Classical Gates

In classical circuits, gates like AND, OR, and NOT are used. A gate set is universal if one can implement any Boolean function $f : \{0,1\}^n \to \{0,1\}$ using the gate set. An example of universal sets, like $\{OR, NOT\}$, $\{NAND\}$, $\{AND, NOT\}$, etc...

Definitions of Classical Gates

- NOT **Gate**: Takes one input $x_1 \in \{0,1\}$ and returns the negation of x_1.
- AND **Gate**: Takes two inputs $x_1, x_2 \in \{0,1\}$ and returns 1 if and only if both x_1 and x_2 are equal to 1.
- OR **Gate**: Takes two inputs $x_1, x_2 \in \{0,1\}$ and returns 1 if $x_1 = 1$ and/or $x_2 = 1$.

Reversible Computing

By examining the outputs of AND and OR gates, it is evident that the input cannot be uniquely deduced from the output. This loss of information is called **irreversibility**. Irreversible computation dissipates heat into the environment.

In contrast, the NOT gate is **reversible** since the input can always be uniquely reconstructed from the output. A computation consisting entirely of reversible operations is called **reversible computation**.

Universal Reversible Gates

A set of gates is **universal** if it can be used to implement any other gate. Universal reversible gate sets include:

- $\{AND, NOT\}$: Can implement any classical operation reversibly.
- Toffoli Gate ($CCNOT$): A three-input gate that flips the third bit if and only if the first two bits are 1. This gate alone is universal for classical reversible computation.

Quantum vs. Classical Logic Gates

Classical computation is built on irreversible and reversible gates, such as AND, OR, NOT, and NAND. While classical gates manipulate binary values (0 or 1), quantum logic gates manipulate qubits, which can exist in a superposition state described by $|\psi\rangle = c_0 |0\rangle + c_1 |1\rangle$, where $|c_0|^2 + |c_1|^2 = 1$. Measurement collapses the qubit to $|0\rangle$ or $|1\rangle$, a feature absent in classical systems.

Quantum gates are reversible and represented by unitary matrices. For instance, the quantum NOT gate, defined by the matrix $U_{\text{NOT}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, flips the state of a qubit. Importantly, quantum gates can create entanglement, a phenomenon where the state of one qubit depends on another, enabling the execution of quantum algorithms.

Quantum Gates

NOT Gate (X Gate)

The Pauli-X gate flips the state of a qubit. Its matrix representation is:

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

- Transforms the basis states:

$$X|0\rangle = |1\rangle, \quad X|1\rangle = |0\rangle.$$

- For a general qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the X gate swaps the amplitudes:

$$X|\psi\rangle = \alpha|1\rangle + \beta|0\rangle.$$

The X gate performs a bit-flip operation, analogous to the classical NOT gate, interchanging the probabilities of the qubit being in the $|0\rangle$ and $|1\rangle$ states.

Hadamard Gate (H Gate)

The Hadamard gate creates superposition states. Its matrix representation is:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- Transforms the basis states:

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

- For a general qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$:

$$H|\psi\rangle = \frac{1}{\sqrt{2}}[(\alpha + \beta)|0\rangle + (\alpha - \beta)|1\rangle].$$

The H gate maps the computational basis states $|0\rangle$ and $|1\rangle$ to equal superpositions, effectively placing the qubit into a state where measurement outcomes are probabilistically equal.

Z Gate

The Pauli-Z gate flips the phase of the $|1\rangle$ state. Its matrix representation is:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

- Transforms the basis states:

$$Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle.$$

- For a general qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$:

$$Z|\psi\rangle = \alpha|0\rangle - \beta|1\rangle.$$

The Z gate leaves the $|0\rangle$ state unchanged while inverting the phase of the $|1\rangle$ state. It is crucial in quantum algorithms that rely on phase kickbacks.

Controlled-NOT Gate (CNOT or CX Gate)

The CNOT gate flips the state of the target qubit if the control qubit is in the $|1\rangle$ state. Its matrix representation is:

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- On the computational basis states:

$$\begin{aligned} \text{CNOT}|00\rangle &= |00\rangle, \quad \text{CNOT}|01\rangle = |01\rangle, \\ \text{CNOT}|10\rangle &= |11\rangle, \quad \text{CNOT}|11\rangle = |10\rangle. \end{aligned}$$

- For a general two-qubit state $|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$:

$$\text{CNOT}|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|11\rangle + \delta|10\rangle.$$

The CNOT gate flips the computation state of qubits, associating information from one qubit to the other, and making it possible to encode classical boolean functions into quantum circuits.

Toffoli Gate (CCX Gate)

The Toffoli gate, or CCX gate, is a three-qubit gate that flips the state of the target qubit if both control qubits are in the $|1\rangle$ state.

The CCX gate acts as follows:

$$CCX\,|abc\rangle = \begin{cases} |ab\,\bar{c}\rangle & \text{if } a = 1 \text{ and } b = 1, \\ |abc\rangle & \text{otherwise.} \end{cases}$$

- It extends the logic of the CNOT gate to three qubits, where two qubits serve as controls.
- When both control qubits are $|1\rangle$, the target qubit is flipped.

The Toffoli gate is universal for reversible classical computation. It is used in building quantum circuits that simulate classical logic gates, such as AND and OR. While for quantum computation, the Toffoli, together with the single-qubit Hadamard gate, form a Universal set of gates.

Universal Quantum Gates

A set of quantum gates is universal if it can construct any unitary operation on a quantum system. Two major milestones define universality in quantum gates:

- **Three-Qubit Gates:** Deutsch introduced a universal three-qubit gate capable of simulating arbitrary unitary transformations. An example is the Toffoli gate (Controlled-Controlled-NOT), which applies a NOT operation to the target qubit if both control qubits are in state $|1\rangle$.
- **Two-Qubit Gates:** Later developments showed that two-qubit gates, such as the Controlled-NOT (CNOT) gate, combined with single-qubit operations, suffice for universality. The CNOT gate is represented by:

$$U_{\text{CNOT}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

It flips the target qubit conditioned on the control qubit being $|1\rangle$.

Quantum logic gates form the bread-and-butter of quantum computation, enabling tasks that are apparently infeasible on classical systems.

Implementing Boolean Functions with Quantum Circuits

Using reversible gates, any Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be implemented on a quantum computer. The quantum operator U_f for such a function is defined as:

$$U_f : |x\rangle |y\rangle \mapsto |x\rangle |y \oplus f(x)\rangle ,$$

where $\oplus$ denotes bitwise addition modulo 2 (XOR). This mapping is reversible, regardless if f is itself invertible or not.

Phase Kickback

A vector v is an **eigenvector** of a matrix A with **eigenvalue** λ if $Av = \lambda v$. Consider the quantum state $|-\rangle = \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle$. Applying the NOT operator to $|-\rangle$:

$$|-\rangle \xrightarrow{NOT} \frac{1}{\sqrt{2}} |1\rangle - \frac{1}{\sqrt{2}} |0\rangle = -\left(\frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle \right) = -|-\rangle .$$

Thus, $|-\rangle$ is an eigenstate of the NOT operator with eigenvalue -1.

When the first qubit is $|0\rangle$, the $CNOT$ operator has no effect:

$$|0\rangle |-\rangle \xrightarrow{CNOT} |0\rangle |-\rangle .$$

When the first qubit is $|1\rangle$:

$$|1\rangle |-\rangle \xrightarrow{CNOT} -|1\rangle |-\rangle .$$

For a two qubit system, with the first qubit $\alpha |0\rangle + \beta |1\rangle$ and the second qubit in $|-\rangle$:

$$\alpha |0\rangle |-\rangle + \beta |1\rangle |-\rangle \xrightarrow{CNOT} \alpha |0\rangle |-\rangle - \beta |1\rangle |-\rangle = (\alpha |0\rangle - \beta |1\rangle) |-\rangle .$$

The sign of the $|1\rangle$ amplitude of the first qubit flips after $CNOT$. This phenomenon is called **phase kickback**, where the eigenvalue is "kicked back"to the control register.

Analysis of $CNOT$ Behavior

Initial State

The system starts in state $|01\rangle$. The states of the qubits before applying $CNOT$ are:

$$\text{First qubit:} \quad |0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle,$$
$$\text{Second qubit:} \quad |1\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle.$$

The composite state is:

$$\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) \otimes \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right).$$

Effect of $CNOT$

The $CNOT$ operator flips the state of the second qubit when the first qubit is $|1\rangle$. After applying $CNOT$:

$$\frac{1}{\sqrt{2}}|0\rangle\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) + \frac{1}{\sqrt{2}}|1\rangle\left(\frac{1}{\sqrt{2}}|1\rangle - \frac{1}{\sqrt{2}}|0\rangle\right).$$

The resulting state can be rewritten as:

$$\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) \otimes \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right).$$

Grover's Algorithm

Grover's algorithm is a quantum algorithm designed to solve the problem of searching for a marked element in an unstructured database or solving a black-box function inversion problem. It provides a quadratic speedup over classical counterparts, reducing the search complexity from $O(N)$ to $O(\sqrt{N})$, where N is the size of the search space. (GROVER, 1996)

Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function such that:

$$f(x) = \begin{cases} 1, & \text{if } x = x^*, \\ 0, & \text{otherwise.} \end{cases}$$

Here, x^* represents the single "marked"element. The goal is to find x^* with as few evaluations of f as possible. Grover's algorithm uses quantum parallelism and amplitude amplification to locate x^*. It involves the following steps: **State Initialization**, **Oracle Query**, **Amplitude Amplification** and **Iterative Search**. Below are the detailed each one step of Grover's Algorithm.

State Initialization

The quantum system is initialized to an equal superposition of all the basis states.

1. Prepare the n-qubit system in the initial state:

$$|\psi_0\rangle = |0\rangle^{\otimes n}.$$

2. Apply the Hadamard transform $\mathbf{H}^{\otimes n}$ to generate the uniform superposition:

$$|\psi_1\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle, \quad \text{where } N = 2^n.$$

Oracle Query

The oracle $\mathbf{O}$ is a quantum operator that flips the sign of the amplitude of the marked state $|x^*\rangle$. Mathematically, it is defined as:

$$\mathbf{O}|x\rangle = \begin{cases} -|x\rangle, & \text{if } x = x^*, \\ |x\rangle, & \text{otherwise.} \end{cases}$$

After applying the oracle, the quantum state becomes:

$$|\psi_2\rangle = \frac{1}{\sqrt{N}} \left(\sum_{x \neq x^*} |x\rangle - |x^*\rangle \right).$$

Amplitude Amplification (Grover Diffusion Operator)

Grover's diffuser operator, often called the inversion about the mean, is a key component of Grover's search algorithm. It amplifies the amplitudes of marked states (those satisfying the Oracle's condition) while reducing the amplitudes of unmarked states, effectively focusing the search on desired solutions. This section details the definition, implementation, and significance of Grover's diffuser operator.

The diffuser operator, denoted as D, performs the transformation:

$$D = 2|s\rangle\langle s| - I,$$

where:

- $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$: The equal superposition state.

- I: The identity operator.
- N: The total number of states in the search space.

This operation reflects the quantum state about the average amplitude of all states, enhancing the probability of measuring marked states.

The diffuser operator is implemented using the following steps:

1. **Initialization to Superposition:** Apply a Hadamard gate to each qubit to create the equal superposition state $|s\rangle$ if not already prepared.
2. **Phase Inversion:**

   The diffuser reflects the amplitudes of all states about their average, indirectly amplifying the marked states (those identified by the Oracle) through iterative applications. This is represented by the operator $-I$.
3. **Reflection About Mean:** Reflect all states about the mean amplitude. This is achieved using:

   $$D = H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n},$$

   where $H^{\otimes n}$ is the Hadamard gate applied to n qubits.

Generally, the diffuser can be broken down into the following steps:

1. Compute the mean amplitude of all states.
2. Invert the amplitude of each state about this mean.

The next step increases the amplitude of the marked state using the Grover diffusion operator $\mathbf{D}$, which reflects the amplitudes about their average value. The operator is defined as:

$$\mathbf{D} = 2\,|\psi\rangle\,\langle\psi| - \mathbf{I}.$$

After applying the diffusion operator, the new state is given by:

$$|\psi_3\rangle = \mathbf{D}(\mathbf{O}\,|\psi_1\rangle).$$

Iterative Search

The combination of the oracle $\mathbf{O}$ and the Grover diffusion operator $\mathbf{D}$ is applied repeatedly. This combined operation is often called the **Grover operator** $\mathbf{G}$, defined as:

$$\mathbf{G} = \mathbf{D}\mathbf{O}.$$

The state evolves iteratively as:

$$|\psi_{k+1}\rangle = \mathbf{G}\,|\psi_k\rangle\,, \quad k = 0, 1, \ldots, r-1.$$

The number of iterations r required to maximize the probability of measuring the marked state is approximately:

$$r = \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor.$$

Measurement

After r iterations, the quantum state is dominated by the marked state $|x^*\rangle$. A measurement in the computational basis yields x^* with high probability.

2.2 GENETIC ALGORITHM

Genetic Algorithm (GA) (EIBEN; SMITH, 2015) are a class of evolutionary algorithms inspired by the principles of natural selection and genetics. They are widely used to solve optimization problems across various domains, including engineering, economics, and artificial intelligence. GAs operate on a population of potential solutions, evolving them over generations to approximate optimal solutions.

Chromosome Representation

In GAs, each potential solution is encoded as a chromosome, which can be represented in various forms such as binary strings, real-valued vectors, or permutations depending on the problem domain. The choice of representation significantly impacts the algorithm's performance and its ability to effectively explore the solution space.

Population Initialization

The algorithm begins with an initial population of chromosomes, typically generated randomly to ensure diversity. This diversity is crucial,, as it provides a broad search space and reduces the likelihood of the algorithm converging prematurely to suboptimal solutions.

Fitness Evaluation

Each chromosome is evaluated using a fitness function, which quantifies how well it solves the problem at hand. The fitness function is problem-specific and guides the selection process by assigning higher fitness values to better solutions.

Genetic Operators

Selection

Selection mechanisms determine which chromosomes are chosen to reproduce and form the next generation. Common selection methods include:

- **Roulette Selection**: Chromosomes are selected probabilistically based on their fitness values, with higher fitness individuals having a greater chance of selection.
- **Tournament Selection**: A subset of chromosomes is chosen randomly, and the one with the highest fitness in this subset is selected.

Crossover

Crossover combines genetic information from two parent chromosomes to produce offspring. It promotes the exchange of beneficial traits and enhances exploration of the solution space. Common crossover techniques include:

- **One-Point Crossover**: A single crossover point is selected, and the segments after this point are swapped between two parents.
- **Two-Point Crossover**: Two crossover points are selected, and the segment between them is exchanged between parents.
- **Uniform Crossover**: Each gene is independently chosen from one of the parents with equal probability.

Mutation

Mutation introduces random alterations to individual genes in a chromosome, maintaining genetic diversity within the population and preventing premature convergence. Mutation rates are typically low to preserve the integrity of high-fitness solutions. For binary representations, mutation may involve flipping bits, while for real-valued representations, it could involve adding a small random value.

Advantages and Limitations

GAs are robust and flexible, capable of handling complex, multimodal, and non-differentiable functions. They do not require gradient information and can escape local optima due to their stochastic nature. However, GAs may require significant computational resources and careful parameter tuning to achieve optimal performance. Limitations such as slow convergence, difficulty in parameter tuning, poor scalability with complex problems, reliance on random search and challenges in properly representing solutions within the algorithm.

2.3 THE MAXIMUM CUT

Introduction

The Maximum Cut (MaxCut) problem is a foundational optimization problem in graph theory with wide-ranging theoretical and practical implications. Formally, given an undirected graph $G = (V, E)$ with edge weights $w_{ij} \in \mathbb{R}^+$ for $(i, j) \in E$, the objective is to partition the vertex set V into two subsets (S, T) such that the sum of the weights of edges between S and T is maximized. This is mathematically expressed as:

$$\text{Maximize} \sum_{(i,j)\in E} w_{ij}(1 - z_i z_j)/2,$$

where $z_i \in \{-1, 1\}$ represents the partition assignment of vertex i.

The Max Cut problem is NP-hard and was one of the original NP-complete problems identified by Karp. It finds applications in various domains, including statistical physics, network clustering, and circuit design. (KARP, 1972)

Lower Bound

A natural lower bound for the Max Cut problem can be derived by considering a random partition of the vertices into two subsets. For any graph G, this approach ensures that each edge has an equal probability of being cut or not cut. Mathematically, the expected weight of a random cut is:

$$\mathbb{E}[\text{Weight of Random Cut}] = \frac{1}{2} \sum_{(i,j)\in E} w_{ij}.$$

This provides a baseline for evaluating approximation algorithms. Since a valid cut must have a weight at least equal to this expectation, the Max Cut problem satisfies the inequality:

$$\text{Max Cut}(G) \geq \frac{1}{2} \sum_{(i,j)\in E} w_{ij}.$$

This lower bound highlights that even the simplest random partition provides a meaningful starting point, achieving an approximation ratio of $1/2$ relative to the total edge weight. (**??**)

Analytical Solution for Complete Graphs

For a complete graph K_n, every vertex is connected to every other vertex. If all edges have uniform weight w, the Max Cut value can be determined analytically as (EDWARDS, 1973):

$$\text{Max Cut}(K_n) = \lfloor \frac{n^2 w}{4} \rfloor,$$

where the partition divides the vertices as evenly as possible into two subsets. This solution arises because each edge contributes to the cut if and only if it spans the two subsets.

Semidefinite Programming Approach

The seminal algorithm proposed by Goemans and Williamson utilizes semidefinite programming (SDP) to approximate the Max Cut problem with a performance guarantee of approximately 0.878. The approach relaxes the combinatorial problem by representing the binary variables $z_i \in \{-1, 1\}$ as vectors v_i on the unit sphere in $\mathbb{R}^n$. The relaxed problem is:

$$\text{Maximize} \sum_{(i,j)\in E} \frac{w_{ij}}{2}(1 - v_i \cdot v_j),$$

subject to $\|v_i\| = 1$ for all $i \in V$.

The SDP relaxation is solved efficiently using standard optimization techniques. A randomized rounding procedure is then applied: vectors v_i are projected onto a random hyperplane, and the resulting binary partition is derived from the sign of the projections. This method ensures the approximation ratio $\alpha = 0.878$, which is optimal assuming the Unique Games Conjecture. (**??**)

Rounding Procedure Details

The randomized rounding begins by selecting a random hyperplane passing through the origin in $\mathbb{R}^n$. Each vector v_i is assigned a partition based on the sign of its dot product with the hyperplane's normal vector. This guarantees that the expected weight of the cut matches the optimal value of the relaxed problem, up to the approximation factor α.

2.4 THE QUANTUM GENETIC ALGORITHM

Introduction

The idea of a Quantum Evolutionary Algorithm (QEA) represents an integration of quantum computing with Evolutionary Algorithms. Unlike classical evolutionary algorithms, like the GA, a QEA incorporates the evolution of a quantum system to enhance population diversity, global search capabilities, and convergence speed. The particular instance of the Quantum Genetic Algorithm (QGA) is a effective candidate framework for optimization problems due to its ability to use the unique properties of quantum mechanics, such as superposition and parallelism.

General Description

The QGA employs quantum bits to encode individuals in the population. A qubit can represent a superposition of states, enabling richer population diversity compared to classical approaches. The population size can remain small because each quantum chromosome can maintain significant diversity. This feature makes QGA more suitable for parallel processing and large-scale optimization problems.

Each individual in the population is encoded as follows:

$$q_j^{(t)} = \begin{bmatrix} \alpha_{j1}^{(t)} & \alpha_{j2}^{(t)} & \cdots & \alpha_{jm}^{(t)} \\ \beta_{j1}^{(t)} & \beta_{j2}^{(t)} & \cdots & \beta_{jm}^{(t)} \end{bmatrix},$$

where m is the chromosome length, α and β are probability amplitudes satisfying $\alpha^2 + \beta^2 = 1$. This quantum representation enables a chromosome to encode 2^m probability amplitudes, significantly increasing the algorithm's information capacity.

Steps of QGA

The process of QGA is outlined below:

1. **Initialization:** Initialize the population $Q(t)$ with $\alpha = \beta = \frac{1}{\sqrt{2}}$ for each qubit.
2. **Measurement:** Measure each quantum chromosome to obtain a classical solution $x_j^{(t)}$. This involves generating a random number r for each qubit and setting the bit to 1 if $r > \alpha^2$, otherwise 0.
3. **Evaluation:** Evaluate the fitness of each solution using a predefined fitness function f. Update the best solution if a better one is found.
4. **Operators:** Crossover and mutation are employed in order to prepare a new population
5. **Quantum Gate Update:** Update the population using a quantum rotation gate:

$$U(\Delta\theta) = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix},$$

   where $\Delta\theta$ is determined based on the problem *a priori* to improve the population's fitness.
6. **Termination:** Repeat the process until a stopping criterion (e.g., maximum iterations or satisfactory fitness) is met.

Challenges

- **Crossover Operator:** The implementation of a crossover operator on quantum framework is not clear, not well-defined

- **Fitness Function:** Classical genetic operators are replaced by Grover's search mechanism, simplifying the algorithm.
- **Invalid States:** Handling invalid individuals (e.g., infeasible solutions) requires fitness functions that map them to distinct regions of the fitness register.

While the potential of quantum computation in this context is significant, some aspects of the approach have not yet fully harnessed this power.

Integration with Other Algorithms

The QGA has been successfully combined with other optimization methods, such as particle swarm optimization and differential evolution, to balance global and local search capabilities effectively, while a hybrid approach is always welcome, a pure quantum circuit approach is to be well-defined.

3 QGA FRAMEWORK FOR THE MAXCUT PROBLEM

3.1 OVERVIEW OF THE FRAMEWORK

The QGA usual framework has inherent flaws that are hard to deal with direcly, however the work done by (UDRESCU; PRODAN; VLăDUţIU, 2006) showcases a novel approach into the implementation of a QGA, this framework is dubbed as the RQGA.This approach deals with the fitness values in parallel with its corresponding individual of the population, meaning that both the individual and its fitness value are in superposition simutaneously, making the QGA's need of measurements to compute fitness values unnecessary. Moreover the frameworks gives a base to develop any new potential QGA based on a specific problem, one which was implemented by one of the authors (ARDELEAN; UDRESCU, 2022). Here the algorithm is distinctvely implemented for the MaxCut algorithm, with the addition of a divide-and-conquer heuristic in form of graph contraction, inspired by (ZHOU et al., 2023)). Firstly, the QGA for the MaxCut will be explained and afterwards the graph contraction heuristic.

3.1.1 Standard QGA Framework

The standard QGA operates as forcing the classical genetic algorithm components into a quantum framework. Its steps typically include:

1. **Population Initialization:** Representing a population of individuals as quantum states. The population is encoded as a superposition of all possible solutions:

$$|\psi_{\text{pop}}\rangle = \frac{1}{\sqrt{2^N}} \sum_{i=0}^{2^N-1} |u_i\rangle,$$

   where N is the number of qubits used to encode an individual.

2. **Quantum Operators:** Applying quantum equivalents of classical genetic operators:

   - *Quantum Mutation:* In standard QGA implementations, mutation is conceptualized as a quantum operation that perturbs the qubit amplitudes similar to classical bit-flipping.

     This is usually done by applying quantum rotation gates, such as Pauli-X, Pauli-Y, or Pauli-Z, to individual qubits in a chromosome register.

These gates rotate the qubit state vector on the Bloch sphere, slightly changing the probability amplitudes of the basis states and thus exploring nearby states in the search space.

- *Quantum Crossover:* Quantum crossover is not directly implementable due to quantum constraints such as no-cloning and the unitarity requirement.

  So, crossovers are approximated by relabeling qubits across entangled registers or applying controlled-swap operations between qubits of "parent"quantum chromosomes. Basically, features from multiple quantum states combine to create new solutions.

3. **Fitness Evaluation:** Measuring the fitness of each individual in the population by collapsing the superposition. Which is in-itself problematic due to the collapse of the wavefunction.

4. **Selection:** Using quantum principles, such as amplitude amplification, to prioritize fitter individuals.

5. **Iteration:** Repeating the process for a fixed number of generations or until convergence.

3.1.2 Enhanced Approach

The algorithm presented for the MaxCut begins by recursively partitioning (ZHOU et al., 2023) the graph using the METIS library until each subgraph reaches a number of vertices less than or equal to a set limit determined by the number of qubits available. For each subgraph, the individual register for each vertex and the fitness register, which is limited by the number of edges in the graph, are created, and then, all of them are set in superposition, using quantum parallelism to represent the solution space of all possible results simultaneously. In this approach, all possible candidate solutions are represented simultaneously in a quantum superposition, with each individual entangled, therefore, directly associated to its corresponding fitness value. Instead of using classical genetic operations like crossover and mutation to explore the search space iteratively, a custom oracle can be used to mark the individuals with the highest fitness. To do so, the fitness function written as an unitary operator for the MaxCut is applied to compute the fitness of each individual, encoding these values into the fitness register. Subsequently, an oracle is applied to mark valid individuals whose values were set by the fitness operator, and Grover's diffusion operator is used to amplify the amplitudes of these marked solutions. The system is then measured, collapsing the quantum state to one of the high-fitness individuals. This cycle of fitness computation, marking, amplitude amplification, and measurement is repeated until the threshold, theoretically limited by Grover's search and the nature of the MaxCut problem, ceases to improve, signaling convergence. (NETO, 2025) Finally, the algorithm outputs the chromosome corresponding to the highest observed fitness value as the solution. The diagram of the process can be view on figure 1.

3.1.2.1 Superposition and Fitness Register

Instead of relying on mutation and crossover, the enhanced approach encodes the entire population into a single quantum superposition, combining individual and fitness registers:

$$|\psi\rangle = \frac{1}{\sqrt{2^N}} \sum_{i=0}^{2^N-1} |u_i\rangle \otimes |0\rangle.$$

An unitary operator U_f based on a boolean function f is applied to compute the fitness values:

$$U_f : |u\rangle \otimes |0\rangle \rightarrow |u\rangle \otimes |f(u)\rangle.$$

This ensures the fitness values are calculated without requiring explicit genetic operators, reducing computational complexity. (UDRESCU; PRODAN; VLăDUţIU, 2006)
This makes mutation and crossover no longer needed because the entire solution space is encoded into a single quantum superposition. This allows the algorithm to evaluate all possible individuals simultaneously by entangling each with its corresponding fitness value. Instead of evolving a population through iterative genetic variation, the algorithm uses Grover's search to amplify the amplitude of optimal individuals, making them more likely to be observed upon measurement. As a result, the traditional mechanisms of mutation and crossover become redundant. While the algorithm retains the idea of fitness-based selection, it no longer follows the classical model of evolutionary computation, making it more accurately described as a quantum optimization strategy inspired by genetic algorithms rather than a true evolutionary algorithm.

3.1.2.2 Optimization with Grover's Algorithm

The enhanced framework reduces the fitness evaluation to a quantum maximum-finding problem using Grover's algorithm. (GROVER, 1996) That means that the traditional process of computing and comparing fitness values for each individual into a single quantum search procedure. Instead of evaluating fitness sequentially and storing results for a population, the enhanced framework encodes all individuals and their corresponding fitness values into a quantum superposition using a unitary fitness operator. Grover's algorithm is then used to search directly for the individual with the highest fitness, effectively identifying the best solution without explicitly comparing each one.
By defining a specific oracle and employing amplitude amplification, it identifies the individual with the highest fitness value in $O(\sqrt{N})$ iterations, significantly outperforming classical exhaustive search methods.

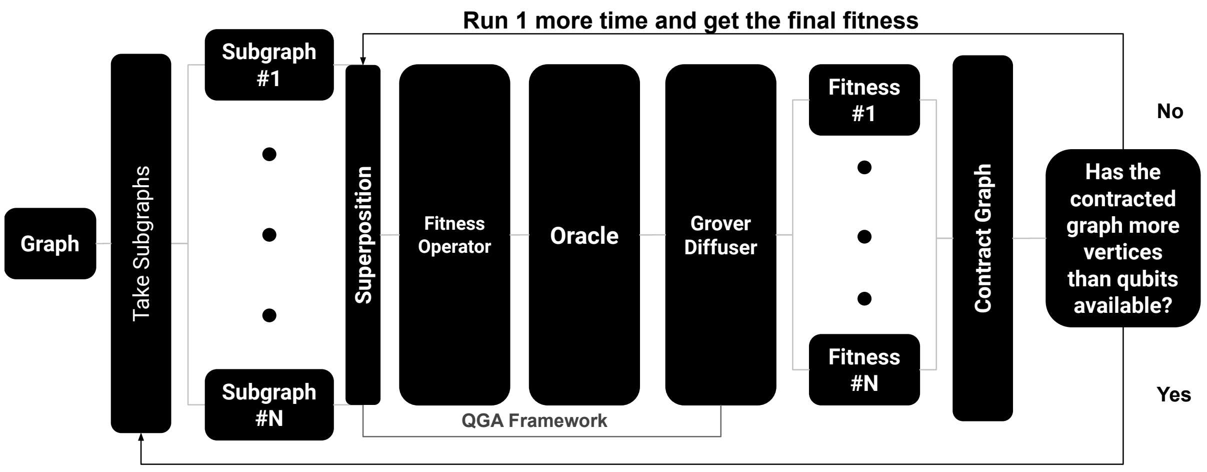

Figura 1 – Diagram of the proposed framework. The fitness operator, oracle and the graph partitioning/contraction are original contributions specific to the MaxCut problem, where the remaning blocks follow the general framework.

3.1.3 Key Differences

While traditional QGAs rely heavily on adaptations of classical genetic operators to quantum computers, the proposed model keeps the quantum state in superposition throughout the computation, avoiding intermediate measurements and classical steps. Making the algorithm to remain entirely quantum from initialization to solution, differently from earlier approaches

Algorithm	**GA**	**QGA**	**This Work**
Population	Explicit	Partial superposition	Full superposition
Variation Operators	Yes	Partial (e.g. Ry gates)	*None*
Fitness Evaluation	Individually	Measured	Unitary Gate U_f
Selection	Fitness-based	Measured + updated	Grover Diffusion
Invalid Solutions	Penalized	Filtered	Encoded exclusion
Method	Evolution	Hybrid loop	Grover search

Tabela 1 – Key differences between GA, QGA, and the proposed RQGA model

that alternated between quantum and classical operations. With the integration of the fitness values directly besides their respective individual using quantum parallelism and using Grover's algorithm to amplify the best solutions, the proposed model overcomes several limitations of conventional QGAs for the MaxCut, like the dependency of computing angles and repetitive measurements.

- **Population Representation:** In standard QGAs, the population is represented solely as a superposition of individual states. Each state encodes a potential solution, but there is no direct association with fitness values within the quantum state. Conversely, the enhanced approach integrates the fitness information directly into the quantum state by combining individual and fitness registers. This integration allows simultaneous processing of both solution candidates and their fitness evaluations.

- **Genetic Operators:** The standard algorithms heavily rely on quantum analogues of classical genetic operators, such as mutation and crossover, to explore the solution space. These operations require iterative application to generate diverse solutions. The enhanced approach eliminates the need for these operators, since the solution is already established by the superposition, it utilizes Grover's algorithm to efficiently amplify the best solution's amplitude without requiring constantly reading the population to adjust circuit parameters.

- **Fitness Evaluation:** Fitness is typically evaluated by collapsing the quantum state and measuring each candidate solution, a process that requires repeated measurements to identify optimal solutions. The enhanced framework replaces this approach with a unitary operator that computes fitness values directly within the quantum superposition, avoiding the need for repeated state collapses.

- **Excluding Undesired Solutions:** By default, there's no explicit mechanism for distinguishing individuals within the population. This can lead to inefficiencies if undesired solutions are prioritized. The enhanced approach bi-partitions the fitness space into subspaces according to theoretical Lower Bound of the MaxCut value, guaranteeing that a bad solution do not interfere with the optimization process. (NETO, 2025)

- **Optimization Methodology:** Standard QGAs rely on iterative selection and modification processes, which can be computationally intensive. The enhanced approach employs a single-step optimization methodology using Diffusion operator of the Grover's algorithm to identify the best solution with quadratic speed-up compared to classical exhaustive search.

- **Computational Complexity:** The computational cost of standard QGAs depends on the number of iterations and the size of the population, often scaling poorly for large problems. In contrast, the enhanced framework achieves a complexity of $O(\sqrt{N})$, inheriting the quantum speed-up of the Grover's algorithm.

3.1.4 Circuit Initialization

The MaxCut problem requires the encoding of a $|V|$-vertices Graph cut solutions into a quantum register, where each binary combination represents a partition. The chromosome is represented as an $(n \times |V|)$-qubit quantum register, where $|V|$ is the number of vertices and n is the number of qubits used to represent the solution value. This number is determined *a priori* by knowing that the maximum value of the cut is the number of Edges of the graph. Giving us that, for any graph with $|E|$ edges, $n = \lceil \log_2 |E| \rceil$.
Similarly, due to the theoretical knowledge of the MaxCut problem we can ignore all those solutions which have a cut less than $\frac{1}{2}$ of the total sum of the edges. So all those solutions are explicitly encoded in a subset which will be ignored, ensuring clear separation between the

Algorithm 1 Enhanced Quantum Genetic Algorithm

```
1: Initialize quantum registers: $|u\rangle$ (individuals) and $|f\rangle$ (fitness values)
2: Prepare a uniform superposition of all possible individuals:
       $$|\psi\rangle = \frac{1}{\sqrt{2^N}} \sum_{i=0}^{2^N-1} |u_i\rangle \otimes |0\rangle$$
3: Apply the fitness function $U_f$ to compute fitness values:
       $$U_f : |u\rangle \otimes |0\rangle \rightarrow |u\rangle \otimes |f(u)\rangle$$
4: Define the oracle subcircuit $O$ to mark states with the highest fitness values:
       $$O : |u\rangle \otimes |f\rangle \rightarrow (-1)^{g(f)}|u\rangle \otimes |f\rangle$$
5: Apply Grover's iterations:
    1. Apply the oracle $O$ to mark the highest fitness states.
    2. Perform the diffusion operator to amplify the marked states.
6: Since Grover's algorithms requires $O(\sqrt{N}))$ queries to the oracle, where $N$ is the size of
   the search space. For a $M$-qubit fitness register, we have a search space of size $2^M$, and
   by knowing that for every graph we have 2 exact solutions for the problem, we would know
   that $N = \frac{2^M}{2} = 2^{M-1}$. Giving us $\approx O(\sqrt{2^{M-1}}))$ necessary queries.
7: Measure the quantum state to obtain the individual with the highest fitness value.
```

desired potential cut values and unneeded solutions.

3.1.5 Fitness Subcircuit

Let G be the set of chromossome-registers and K the set of their respective fitness valule. The fitness function $f : \{G, K\} \rightarrow \mathbb{N}$ is defined as follows:

- Returns 0, if the individual is undesired, i.e., has its $MaxCut \leq \frac{1}{2}|E|$.
- Returns $x \in \mathbb{N}$, where x is the number of edges with adjacent vertices belonging to different partitions (cut).

The operator U_{fit} is defined by the function f, performing the transformation:

$$U_{\text{fit}} : |u\rangle \otimes |0\rangle \rightarrow |u\rangle \otimes |f(u)\rangle,$$

where:

- $|u\rangle$: Quantum state representing an individual.

- $|0\rangle$: Initial state of the fitness register.
- $|f(u)\rangle$: State encoding the fitness value of u, determined by the fitness function f.

The U_{fit} operator is implemented using Controlled-NOT (CNOT) and Toffoli gates. It uses n-qubit register to represent the individual solution $|u\rangle$ and m-qubit fitness register initialized to $|0\rangle$. Auxiliary qubits are used to encode the fitness function in the m-qubit. To achieve this, counting the number of cut edges in a MaxCut is first translated into a network of logic gates. This circuit is then decomposed into quantum primitives. Since classical Boolean functions are generally not reversible, ancilla (auxiliary) qubits are introduced to preserve reversibility and avoid information loss. The Toffoli gate (a universal reversible gate) is used to implement AND-like logic, while CNOT and NOT gates handle XOR and bit-flip operations. (BARENCO et al., 1995)

Furthermore, to maintain unitarity, all intermediate computation must be uncomputed (i.e., reversed) after the result has been written into the fitness register.

3.1.6 Oracle Subcircuit

The Oracle circuit is implemented to perform the transformation:

$$O : |u\rangle \otimes |f(u)\rangle \to (-1)^{g(f(u),T)}|u\rangle \otimes |f(u)\rangle,$$

where:

- $|u\rangle$: Represents the quantum state encoding an individual.
- $|f(u)\rangle$: Represents the quantum state encoding the fitness value of the individual.
- T: A predefined threshold value, which for the MaxCut is the $|E|$
- $g(f(u),T)$: A Boolean function that evaluates to 1 if $f(u) > T$ and 0 otherwise.

This transformation marks the states with fitness values greater than the threshold T by applying a phase flip. The maximum possible fitness corresponds to the scenario where all edges are cut (a bipartite graph), making the fitness value equal to the total number of edges in the graph. By setting the threshold to the number of edges, the algorithm ensures that the search is confined to valid configurations, where each edge contributes to the fitness. (ARDELEAN; UDRESCU, 2022) (NETO, 2025)

3.1.6.1 The Quantum Adder

To evaluate whether $f(u) > T$, a quantum ripple-carry adder is utilized. The quantum adder operates reversibly and proceeds through a sequence of controlled operations as follows:

$$|a\rangle \otimes |b\rangle \otimes |c_0\rangle \rightarrow |a\rangle \otimes |S\rangle \otimes |c_n\rangle,$$

where $|a\rangle$ represents the fitness value $f(u)$, $|b\rangle$ represents the threshold value T, and $|c_0\rangle$ is an ancillary qubit. (DRAPER, 2000) The adder computes the sum $S = f(u) + T$ modulo 2^n, storing the result in $|b\rangle$, while the carry bit c_n is stored in the ancillary qubit $|c_n\rangle$. The operation is performed using a sequence of *MAJ* (majority) gates to compute carry bits and *UMA* (UnMajority and Add) gates to compute the sum and reverse intermediate changes. (CUCCARO et al., 2004)

The adder evaluates whether $f(u) > T$ by checking the carry qubit or the most significant bit (MSB) of the result, which indicates whether an overflow occurred during addition. This method ensures a reversible computation, adhering to the principles of quantum mechanics.

The Oracle circuit, incorporating this quantum ripple-carry adder guides the QGA optimization process. By comparing $f(u)$ and T and marking individuals with high fitness, the Oracle ensures that genetic operations focus on promising solutions, accelerating convergence to the optimal result. The combination of the ripple-carry adder and reversible logic is particularly effective for solving the MaxCut, as it efficiently evaluates and identifies optimal solutions.

3.1.7 Grover Diffuser

Grover's diffuser operator, often called the inversion about the mean, is a key component of Grover's search algorithm. It amplifies the amplitudes of marked states (those satisfying the Oracle's condition) while reducing the amplitudes of unmarked states, effectively focusing the search on desired solutions. This document details the definition, implementation, and significance of Grover's diffuser operator.

The diffuser operator, denoted as D, performs the transformation:

$$D = 2|s\rangle\langle s| - I,$$

where:

- $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$: The equal superposition state.

- I: The identity operator.
- N: The total number of states in the search space.

This operation reflects the quantum state about the average amplitude of all states, enhancing the probability of measuring marked states.

The diffuser operator is implemented using the following steps:

1. **Initialization to Superposition:** Apply a Hadamard gate to each qubit to create the equal superposition state $|s\rangle$ if not already prepared.
2. **Phase Inversion:**

   Apply a conditional phase flip to invert the amplitudes of the marked states. This is represented by the operator $-I$.
3. **Reflection About Mean:** Reflect all states about the mean amplitude. This is achieved using:
$$D = H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n},$$
   where $H^{\otimes n}$ is the Hadamard gate applied to n qubits.

Generally, the diffuser can be broken down into the following steps:

1. Compute the mean amplitude of all states.
2. Invert the amplitude of each state about this mean.

This is equivalent to the operation:

$$D|x\rangle = 2\langle s|x\rangle|s\rangle - |x\rangle.$$

3.2 DIVIDE-AND-CONQUER HEURISTIC FOR QGA

Here it is outlined the implementation of a divide-and-conquer heuristic for the Quantum Genetic Algorithm (QGA), inspired by its application in Quantum Approximate Optimization Algorithms (QAOA) (ZHOU et al., 2023). The heuristic uses graph partitioning, local optimization, and solution merging to solve large-scale problems, specifically the MaxCut problem, efficiently within the constraints of NISQ quantum hardware.

Graph Retraction

1. **Partition the Graph:** The goal is to divide the input graph $G(V, E)$ into smaller subgraphs $\{G_i(V_i, E_i)\}$, where each subgraph G_i has the number of vertices $|V_i| \leq n$. The parameter n represents the maximum number of vertices that can fit in the quantum register used in the Quantum Genetic Algorithm (QGA).

   - *Graph Representation:* The input graph G is represented by its set of vertices V and edges E. For large graphs, directly encoding all vertices in the quantum register is infeasible for NISQ devices, necessitating a partitioning process.
   - *Initial Division:* Divide G into subgraphs $\{G_i\}$ such that each G_i has fewer than or equal to n vertices. These subgraphs are created to balance size and structure, ensuring they are manageable within the constraints of the quantum hardware.
   - *Recursive Partitioning:* After the initial partitioning, some subgraphs may still exceed the quantum register's size limit n. Recursive partitioning ensures that all subgraphs eventually satisfy the size constraint.
   - *Condition Check:* For each subgraph G_i, check if $|V_i| > n$. If so, apply the partitioning process again to G_i.
   - *Divide Further:* Break the oversized subgraph into smaller subgraphs using iterative techniques to ensure size constraints are met.
   - *Repeat Until Completion:* The process continues recursively until all subgraphs G_i satisfy $|V_i| \leq n$.

Practical Limitations

- **Boundary Edges:** During partitioning, edges connecting vertices in different subgraphs are known as boundary edges. These edges require careful handling, especially when applying algorithms like MaxCut, as their inclusion/exclusion affects the overall optimization.
- **Subgraph Characteristics:** The partitioning process often seeks to create subgraphs that are not only small enough to fit within the quantum register but also retain meaningful structures.

- **Recursive Depth:** The depth of recursion is determined by the initial graph size and the register limit n. For very large graphs, this could result in multiple levels of nested subgraphs.

The graph G is represented as a set of disjoint or overlapping subgraphs $\{G_i(V_i, E_i)\}$, each of which satisfies the condition $|V_i| \leq n$. These smaller subgraphs can then be processed independently within the constraints of the quantum register, enabling for the optimization of the MaxCut problem.

Local Optimization with QGA

1. **Encode Subgraphs:**
   - Represent individuals as quantum states in a superposition.
   - Use fitness evaluation circuits tailored to the MaxCut problem to calculate cuts for each subgraph.
2. **Apply QGA:**
   - Perform selection, crossover, and mutation operations on the quantum register.
   - Compute fitness values without intermediate measurements by leveraging superposition.
   - Terminate when the stopping criterion (e.g., iteration count or convergence) is met.
3. **Store Subgraph Solutions:** Save the optimized solutions $\{x_i\}$ for each subgraph.

Solution Merging

1. **Reformulate the Problem:**
   - Treat each subgraph as a node in a new meta-graph $G'(V', E')$.
   - Assign weights to edges in G' based on connectivity and edge weights between subgraphs in the original graph G.

- Handle symmetry (e.g., $\mathbb{Z}_2$) by considering both x_i and its complement $\bar{x}_i$ as valid solutions for each subgraph.

2. **Solve the Meta-Graph:**

   - Use QGA to find the optimal cut for G'.
   - Interpret the meta-graph solution to determine the global cut for G.

Recursive Refinement

If the meta-graph G' exceeds the size limit n:

1. **Recursive Application:**

   - Partition G' into smaller subgraphs.
   - Optimize the subgraph solutions using QGA.
   - Repeat until the size of the final meta-graph is manageable within the quantum register's capacity.

3.2.1 Complexity Analysis

The complexity analysis of the Algorithm for the MaxCut problem accounts for the inherent challenges of simulating quantum algorithms on classical computers.) the simulation of quantum circuits requires exponential runtime with respect to the circuit size. The total number of qubits required for QGA in the MaxCut problem is determined by the function (UDRESCU; PRODAN; VLăDUţIU, 2006):

$$f(|V|, n, M, m) = |V| \cdot n + 2 \cdot (M + m) + 3,$$

where:

- $|V|$: The number of vertices in the graph.
- n: The number of qubits required to encode vertex states or partitions.
- M: The number of qubits needed to represent the fitness value in two's complement, typically $\lceil \log_2(E) \rceil$, where E is the number of edges in the graph.

- m: The number of Grover iterations, calculated as $m = O(\sqrt{2^M})$.
- Additional qubits: 3 qubits are used for carry-in, oracle workspace, and a validity flag, while Grover's adders require 2 qubits for carry-out during each iteration.

Remembering that a complete graph is a upper-bound of the number of edges for any graph witn n vertices, we can derivate a function,

$$g(n) = n^2 + 2^{\frac{n(n-1)}{4}+1} + 3,$$

Which bounds the f and give us the minimum amount of qubits necessary to run a graph with n vertices. Due to the exponential growth of required resources with $|V|$, M, and m, the scalability of the algorithm in simulation is limited. These constraints underscore the necessity of access to actual quantum hardware to fully exploit the potential of the application for large graph instance without the downside of a divide-and-conquer heuristic. Despite these simulation challenges, the theoretical framework of QGA retains its quantum advantage, offering an efficient approach to solve combinatorial optimization problems like MaxCut.

4 RESULTS AND CONCLUSION

The QGA for the MaxCut has been evaluated experimentally on various classes of graphs, including complete graphs and Erdős-Rényi random graphs, to demonstrate its effectiveness in solving the MaxCut problem. It is presented a comparison of the QGA with the Semidefinite Programming (SDP) approach and highlights its performance on specific graph types. All experiments were conducted under similar conditions to ensure the validity of comparisons. The measurements were conducted by implementing the algorithm using Qiskit, with simulations performed on the IBM quantum platform. The $ibmq\_qasm\_simulator$ backend (Qiskit Development Team, 2023), provided by the $ibm-q$ provider, was utilized. This simulator is a versatile, context-aware tool capable of simulating quantum circuits under ideal conditions or with noise modeling, supporting circuits with up to 29 qubits. The code can be found here: https://github.com/pauloaviana/maxcut-qga

4.1 EXPERIMENTAL SETUP

The experiments were conducted on two types of graphs on a Qiskit Simulator:

- **Complete Graphs:** Graphs where every pair of vertices is connected by an edge.
- **Erdős-Rényi Random Graphs:** Graphs generated with a fixed probability for edge inclusion, denoted as $G(n, p)$.

For each graph type, the performance of QGA and SDP was compared based on the cut values achieved. Results were averaged over multiple runs to account for randomness inherent in both algorithms.

4.2 RESULTS FOR SMALL GRAPHS

For graphs small enough to run the QGA directly, those without a divide-and-conquer heuristic (which by itself loses boundary edges), are limited up to $|V| = 8$ vertices. Both the the SDP and the QGA got the optimal result for graphs (complete or Randomly generated) up to such size.

4.3 RESULTS FOR COMPLETE GRAPHS

The experiments on complete graphs showed that the QGA consistently found the true optimal MaxCut values, which are determined by $\left\lfloor \frac{n^2}{4} \right\rfloor$, while the SDP approach achieved approximate results withing the theoretical limit 0.878, Table 2 summarizes these results.

Tabela 2 – Comparison of QGA and SDP on Complete Graphs.

Number of Vertices	QGA (Optimal Value)	SDP Value	QGA Ratio	SDP Ratio
3	2	2	1.0	1.0
5	6	6	1.0	1.0
8	16	15	1.0	0.9375
12	36	35	1.0	0.9722
23	132	130	1.0	0.9848
31	240	237	1.0	0.9875
56	784	780	1.0	0.9949
80	1600	1593	1.0	0.9969
128	4096	4085	1.0	0.9973

The table shows that QGA propostod got the true value for the MaxCut for all the tested instances, while the SDP slowly gets worst results as the number of vertices grows, as expected.

4.4 RESULTS FOR ERDŐS-RÉNYI RANDOM GRAPHS

For Erdős-Rényi random graphs, the performance of QGA varied depending on the specific instance, but it consistently demonstrated competitive or superior results compared to SDP. Two separate tables, Tables 3 and 4 respectivelly show the results for the median value found during the runs and for the best result found for the QGA. A comparison of the performance of QGA runs and SDP values is made, including the ratio of RQGA results to SDP values.

Tabela 3 – Comparison of QGA Run 1 and SDP on Erdős-Rényi Random Graphs.

Graph Instance	QGA Med.	SDP Value	QGA/SDP Ratio
$G(50, 0.1)$	91	92	0.9891
$G(50, 0.25)$	194	210	0.9238
$G(50, 0.5)$	346	360	0.9611
$G(50, 0.75)$	478	524	0.9122
$G(100, 0.1)$	232	329	0.7052
$G(100, 0.25)$	674	786	0.8576
$G(100, 0.5)$	1297	1361	0.9529
$G(100, 0.75)$	1894	2016	0.9394
$G(200, 0.1)$	1017	1211	0.8401
$G(200, 0.25)$	2550	2778	0.9180
$G(200, 0.5)$	5095	5326	0.9566
$G(200, 0.75)$	7494	7815	0.9589
$G(350, 0.1)$	3120	3611	0.8640
$G(350, 0.25)$	7771	8236	0.9436
$G(350, 0.5)$	15443	16030	0.9634
$G(350, 0.75)$	22941	23530	0.9749
$G(500, 0.1)$	6335	7097	0.8926
$G(500, 0.25)$	15684	16520	0.9493
$G(500, 0.5)$	31316	33110	0.9456
$G(500, 0.75)$	46875	48130	0.9740

This table takes one average run of the QGA and compares the result with the SDP, The results are competitive to the SDP and since there are lost boundary edges using the graph contraction approach, the results obtained are inferior to an expected application with more qubits.

Tabela 4 – Comparison of QGA Best runs and SDP on Erdős-Rényi Random Graphs.

Graph Instance	**QGA Best**	**SDP Value**	**QGA/SDP Ratio**
$G(50, 0.1)$	96	92	1.0435
$G(50, 0.25)$	240	210	1.1429
$G(50, 0.5)$	320	360	0.8889
$G(50, 0.75)$	512	524	0.9771
$G(100, 0.1)$	343	329	1.0426
$G(100, 0.25)$	783	786	0.9962
$G(100, 0.5)$	1375	1361	1.0103
$G(100, 0.75)$	2024	2016	1.0040
$G(200, 0.1)$	1250	1211	1.0322
$G(200, 0.25)$	2861	2778	1.0299
$G(200, 0.5)$	5423	5326	1.0182
$G(200, 0.75)$	7875	7815	1.0077
$G(350, 0.1)$	3639	3611	1.0078
$G(350, 0.25)$	8583	8236	1.0421
$G(350, 0.5)$	16030	16030	1.0000
$G(350, 0.75)$	23740	23530	1.0089
$G(500, 0.1)$	7034	7097	0.9911
$G(500, 0.25)$	17140	16520	1.0375
$G(500, 0.5)$	33140	33110	1.0009
$G(500, 0.75)$	48200	48130	1.0015

The results shown are the best picks after multiple runs, it's possible to see slightly better results even with the disadvantage of the graph contraction heuristic. Such results are not conclusive but they show a potential advantage of the QGA over the SDP even with a low number of qubits available.

4.5 ANALYSIS AND DISCUSSION

4.5.1 Complete Graphs

The QGA consistently achieved the true MaxCut values for all tested complete graphs. This comes from the fact that for complete graphs you do not lose edges when partitioning it with the implemented heuristic due to its symmetry. This shows that the QGA can consistently optimize for the subgraphs and their weighted version when recursively partitioned. In contrast, the SDP approach occasionally underperformed as the number of vertices go up, as it's expected from it.

4.5.2 Erdős-Rényi Random Graphs

The Erdős-Rényi random graph is a fundamental model in graph theory that generates random graphs through a probabilistic process. $G(n, p)$: In this model, a graph with n vertices is constructed by adding each possible edge between any two vertices independently with probability p. This means the presence of each edge is determined randomly, leading to graphs with varying numbers of edges, though the expected number of edges is $\binom{n}{2}p$.

The Erdős-Rényi model exhibits specific statistical properties. In the $G(n, p)$ model, the degree of each vertex (the number of edges connected to it) follows a binomial distribution, which approximates a Poisson distribution for large n and small p. The model also displays sharp transitions or threshold phenomena, where certain properties of the graph, such as connectivity or the emergence of a giant connected component, appear suddenly as p changes.

The clustering coefficient, which measures how likely neighbours of a vertex are to be connected, is equal to p in this model. For sufficiently large n and moderate p, the diameter of the graph (the longest shortest path between two vertices) tends to grow logarithmically with n, indicating that the graph remains relatively small in terms of overall distance between vertices.

The Erdős-Rényi model is a well-known method in the literature for understanding random graph behavior, and generating random graphs.

The QGA displayed variability in results across multiple runs, likely due to probabilistic measurement. However, it frequently outperformed SDP when selected for the best result run. Moreover, even when taken on average, the QGA yields a good performance in comparison with the SDP.

4.6 CONCLUSION

This thesis proposed a fully quantum algorithm for solving the MaxCut problem using a Reduced Quantum Genetic Algorithm (RQGA) framework built on Grover's maximum-finding algorithm. While it it departs from the original concept of a Evolutionary Algorithm, its roots and idealization came from it. By eliminating classical genetic operations such as crossover and mutation, and encoding the entire solution space into a single quantum superposition, the proposed model streamlines the search for optimal solutions through amplitude amplification alone. The architecture integrates the fitness evaluation and selection processes into a purely unitary sequence, preserving quantum coherence and minimizing classical control overhead. This model addresses major limitations of standard Quantum Genetic Algorithms (QGAs), which often rely on classical post-processing, intermediate measurements, and hybrid genetic operators, leading to resource inefficiency and decoherence.

The uniqueness of the proposed algorithm lies in its adaptation and expansion of the RQGA framework for the MaxCut problem. While the original RQGA article introduced the conceptual structure of a quantum genetic algorithm reduced to Grover's search, it remained largely abstract and focused on foundational components such as the oracle and fitness entanglement scheme. In contrast, this thesis delivers a complete, application-oriented implementation tailored for the MaxCut problem: it designs a quantum fitness function for MaxCut, encodes valid bipartitions with a binary validity mechanism, constructs a Grover-compatible oracle specific to the problem, and introduces a novel divide-and-conquer heuristic to overcome hardware limitations. Moreover, the model is quantitatively analized using Qiskit simulations.

Experimental validation demonstrates the algorithm's correctness and practical viability in solving instances of MaxCut across various types of graphs. For small and complete graphs, the RQGA consistently found the true maximum cut, often in fewer iterations than expected, demonstrating both efficiency and reliability. In the case of Erdős–Rényi random graphs, where problem structure is less regular, the proposed model still achieved competitive or superior results. This outcome is particularly notable considering that the divide-and-conquer heuristic, used to scale the algorithm to larger graphs, discards boundary edges by design—yet the algorithm maintained strong performance despite this inherent loss.

The resource estimates reported in Table 3; covering qubit count, circuit depth, and simulation fidelity—suggest that the algorithm is implementable on current or near-future quantum hardware, particularly as Noisy Intermediate-Scale Quantum (NISQ) technology progresses. Due to current simulator limitations, the thesis focused on graphs ranging from 5 to 10 vertices, with the divide-and-conquer heuristic proposed as a scalability solution. This heuristic partitions the input graph into smaller, manageable subgraphs, solves each independently with the RQGA, and then applies a final MaxCut procedure to a meta-graph representing the partitioned solution. Though boundary edges are lost in the partitioning process, the method remains efficient and generalizable, and can be adjusted as more qubits become available in hardware.

A direct comparison with standard QGAs was not performed due to the absence of publicly available, problem-specific QGA implementations for MaxCut, and because the fully quantum and coherence-preserving nature of the proposed model diverges methodologically from hybrid quantum-classical approaches. However, this remains an important direction for future work, and the thesis acknowledges the value of even conceptual or small-scale empirical comparisons.

For future work, several promising avenues emerge. A hardware implementation using available NISQ systems would allow for empirical evaluation under real noise models and connectivity constraints. Further optimization of subcircuits, particularly the fitness function, oracle, and Grover diffuser could reduce the algorithm's qubit footprint and circuit depth. In parallel, hybrid strategies that combine Grover-based RQGA with variational quantum circuits (VQCs) may offer more flexible optimization by replacing discrete search with gradient-based parameter updates. (PERUZZO et al., 2014)

Such a variational approach could leverage cost functions encoding problem constraints and compare favorably in terms of convergence speed, scalability, and quantum resource consumption.

Beyond MaxCut, the RQGA framework could be adapted to solve other NP-hard graph problems such as graph coloring, vertex cover, or the Traveling Salesman Problem (TSP), particularly as support for weighted constraints and larger solution spaces becomes feasible. Integration with real-world applications in areas such as VLSI circuit design, statistical physics, logistics, and scheduling would further demonstrate the framework's practical potential. Moreover, improvements in the divide-and-conquer strategy, including smarter partitioning methods or quantum-aware boundary edge handling, could significantly increase solution quality while preserving scalability.

In conclusion, this work lays the foundation for a fully quantum approach to heuristic optimization that is both coherent and modular. It demonstrates the viability of solving combinatorial problems like MaxCut entirely within a quantum circuit model using structured superposition, quantum fitness evaluation, and amplitude amplification while also offering a roadmap for extending and deploying such models in real-world and large-scale contexts as quantum hardware matures.

BIBLIOGRAPHY

ARDELEAN, S. M.; UDRESCU, M. Graph coloring using the reduced quantum genetic algorithm. *PeerJ Computer Science*, v. 7, 2022.

BARENCO, A.; BENNETT, C. H.; CLEVE, R.; DIVINCENZO, D. P.; MARGOLUS, N.; SHOR, P.; SLEATOR, T.; SMOLIN, J. A.; WEINFURTER, H. Elementary gates for quantum computation. In: *Physical Review A*. [S.l.: s.n.], 1995. v. 52, n. 5, p. 3457–3467.

CHURCH, A. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, v. 58, n. 2, p. 345–363, 1936.

CUCCARO, S. A.; DRAPER, T. G.; KUTIN, S. A.; MOULTON, D. P. A new quantum ripple-carry addition circuit. *arXiv preprint*, 2004.

DRAPER, T. G. Addition on a quantum computer. In: *arXiv preprint*. [S.l.: s.n.], 2000.

EDWARDS, C. S. Some extremal properties of bipartite subgraphs. *Canadian Journal of Mathematics*, v. 25, n. 3, p. 475–485, 1973.

EIBEN, A. E.; SMITH, J. E. *Introduction to evolutionary computing*. [S.l.: s.n.], 2015. (Natural Computing Series).

GOSSETT, P. Quantum carry-save arithmetic. 1998.

GROVER, L. K. A fast quantum mechanical algorithm for database search. In: *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*. [S.l.: s.n.], 1996.

GUERRESCHI, G. G.; MATSUURA, A. Y. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific Reports*, v. 9, n. 1, 2019.

KARP, R. M. Reducibility among combinatorial problems. In: *Complexity of Computer Computations*. [S.l.: s.n.], 1972. p. 85–103.

KAYE, P.; LAFLAMME, R.; MOSCA, M. *An Introduction to Quantum Computing*. [S.l.: s.n.], 2006.

KITAEV, A.; SHEN, A.; VYALYI, M. *Classical and quantum computation*. [S.l.: s.n.], 2002. (Graduate Studies in Mathematics).

KUMAR, S.; KUMAR, T. V. A novel quantum-inspired evolutionary view selection algorithm. *Sādhanā*, v. 43, n. 10, p. 1–20, 2018.

KWOK, J.; PUDENZ, K. *Graph coloring with quantum annealing*. 2020. Available at <https://arxiv.org/abs/2012.04470>.

LAHOZ-BELTRA, R. Quantum genetic algorithms for computer scientists. *Computers*, v. 5, n. 4, p. 24, 2016.

MAHMOUDI, S.; LOTFI, S. Modified cuckoo optimization algorithm (mcoa) to solve graph coloring problem. *Applied Soft Computing*, v. 33, p. 48–64, November 2015.

MALOSSINI, A.; BLANZIERI, E.; CALARCO, T. Quantum genetic optimization. *IEEE Transactions on Evolutionary Computation*, v. 12, n. 2, p. 231–241, 2008.

MIRKA, R.; WILLIAMSON, D. P. An experimental evaluation of semidefinite programming and spectral algorithms for max cut. *ACM Journal of Experimental Algorithmics*, v. 28, p. 1–18, 2023.

MOUSSA, C.; CALANDRA, H.; HUMBLE, T. S. Function maximization with dynamic quantum search. In: *International Workshop on Quantum Technology and Optimization Problems*. Berlin: Springer, 2019. p. 86–95.

MUTHUKRISHNAN, A.; STROUD, C. R. Multivalued logic gates for quantum computation. *Physical Review A*, v. 62, n. 5, 2000.

NANNICINI, G. An introduction to quantum computing, without the physics. *SIAM Review*, v. 62, n. 4, p. 936–981, 2020.

NETO, P. A. V. . F. M. A quantum genetic algorithm framework for the maxcut problem. In: *arXiv preprint*. [S.l.: s.n.], 2025.

NIELSEN, M. A.; CHUANG, I. L. *Quantum Computation and Quantum Information*. [S.l.: s.n.], 2012.

PERUZZO, A.; MCCLEAN, J.; SHADBOLT, P.; YUNG, M.-H.; ZHOU, X.-Q.; LOVE, P. J.; ASPURU-GUZIK, A.; O'BRIEN, J. L. A variational eigenvalue solver on a quantum processor. *Nature Communications*, v. 5, p. 4213, 2014.

Qiskit Development Team. *Qiskit: An Open-source Framework for Quantum Computing*. 2023. <https://qiskit.org/>.

SIPSER, M. *Introduction to the Theory of Computation*. 3rd. ed. [S.l.]: Cengage Learning, 2012.

TURING, A. M. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42, n. 1, p. 230–265, 1936.

UDRESCU, M.; PRODAN, L.; VLăDUţIU, M. Implementing quantum genetic algorithms. In: *Proceedings of the 3rd Conference on Computing Frontiers*. [S.l.: s.n.], 2006. p. 71–82.

VEDRAL, V.; BARENCO, A.; EKERT, A. Quantum networks for elementary arithmetic operations. *Physical Review A*, v. 54, n. 1, p. 147–153, 1996.

WANG, F.; LUO, M.; LI, H.; QU, Z.; WANG, X. Improved quantum ripple-carry addition circuit. *Science China Information Sciences*, v. 59, n. 4, 2016.

ZHANG, G. Quantum-inspired evolutionary algorithms: A survey and empirical study. *Journal of Heuristics*, v. 17, n. 3, p. 303–351, 2010.

ZHOU, Z.; DU, Y.; TIAN, X.; TAO, D. Qaoa-in-qaoa: Solving large-scale maxcut problems on small quantum machines. *Physical Review Applied*, v. 19, n. 2, 2023.