Genetic Quantum Algorithm for Combinatorial Optimization [1]: A Detailed Report

António Almeida FC58235

Informatics Department,
Faculty of Sciences of the University of Lisbon
30th of May 2025
Lisboa, Portugal

I. Introduction

Combinatorial optimization problems represent a significant challenge in computer science due to their computational complexity and real-world applications. The knapsack problem for example, which involves selecting items to maximize value without exceeding max capacity, serves as a NP-hard problem that tests algorithmic efficiency. This report examines the Genetic Quantum Algorithm (GQA) proposed by Han and Kim, which integrates principles from quantum computing with evolutionary algorithms to tackle combinatorial optimization challenges.

GQA fundamentally differs from classical genetic algorithms by using quantum-inspired representations. Where traditional approaches use binary, numeric, or symbolic encodings, GQA utilizes quantum bits (qubits) that can exist in superpositions of states. This enables simultaneous representation of various probabilistic solutions while being very compact. The primary objectives of GQA include: (1) Leveraging quantum superposition for more efficient exploration of the solution space; (2) Replacing conventional genetic operators with quantum gates; and (3) Achieving higher convergence speed and solution quality compared to classical methods.

II. State of the Art

Quantum computing has demonstrated theoretical advantages for specific problem classes since the early 1990s. Shor's polynomial-time factorization algorithm and Grover's $\sqrt{n}$-speedup for unstructured search established the potential of quantum computing for exponential speedups over classical approaches. However, practical quantum algorithms are still few, creating a gap between theoretical potential and real-world applications.

Evolutionary algorithms emerged as powerful heuristics for optimization problems, particularly when exact methods become computationally infeasible. Classical genetic algorithms (GAs) employ selection, crossover, and mutation operators to evolve solutions. For constraint handling in problems like knapsack, traditional GAs use penalty functions, repair mechanisms, or specialized decoders. Despite their versatility, GAs often suffer from premature convergence to local optima and inefficient exploration of large solution spaces.

Prior to GQA, research merging evolutionary and quantum computing followed two paths: developing quantum algorithms via automatic programming techniques like genetic programming, and creating quantum-inspired evolutionary algorithms for classical computers. The latter category incorporated quantum principles like interference and coherence but lacked GQA's direct use of quantum state representations and gate operations.

III. Main Achievements

The Genetic Quantum Algorithm shows three significant advances in evolutionary optimization. First, it introduces a novel qubit chromosome representation that encodes solution superpositions very compactly. For example, a single 3-qubit chromosome can simultaneously represent four binary solutions with distinct probabilities, whereas classical approaches require four separate chromosomes. This representation maintains diversity while reducing memory needed.

Second, GQA replaces traditional genetic operators with quantum gates. Rotation gates adaptively adjust the qubit probabilities based on the solution's fitness, guiding the population towards optimal regions without premature convergence. This quantum-inspired operation removes the need for crossover and mutation operators whose parameters often require extensive tuning and are very hard to find.

And third, extensive experiments on knapsack problems demonstrate GQA's practical superiority in combinatorial problems. With populations as small as one chromosome, GQA achieved 8-10% higher profits than the best classical genetic algorithms. Remarkably, it accomplished these results while being 10-100 times faster (in some cases) in computation time. For 500-item problems, GQA maintained linear convergence rates where classical methods plateaued very early, demonstrating superior global search capabilities without sacrificing solution quality.

IV. Technical Implementation

A. Quantum Representation

GQA's base lies on its qubit-based representation. A single qubit is defined as a unit vector in two-dimensional Hilbert space:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (1)$$

where α and β are complex probability amplitudes satisfying $|\alpha|^2 + |\beta|^2 = 1$. For computational purposes, these amplitudes are maintained as real numbers representing the probabilities of observing 0 or 1 states upon measurement.

An m-qubit chromosome is represented as:

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{bmatrix} \quad (2)$$

This representation provides probabilistic expression of 2^m possible states simultaneously. During observation, each qubit collapses to 0 or 1 based on $|\alpha_i|^2$ and $|\beta_i|^2$ probabilities, generating a candidate binary solution.

B. *Algorithmic Workflow*

The GQA procedure operates through five iterative steps:

1) **Initialization:** Initialize population $Q(t)$ of n qubits chromosomes with $\alpha_i = \beta_i = \frac{1}{\sqrt{2}}$ for all qubits, representing uniform superposition of all possible states.
2) **Observation:** Generate binary solutions $P(t)$ by probabilistically measuring each qubit. For each qubit i, generate random number $r \in [0, 1]$. If $r \leq |\alpha_i|^2$, set $x_i = 0$; otherwise $x_i = 1$ (r acts as a treshold).
3) **Repair:** Adjust infeasible solutions to satisfy constraints. For knapsack problems:
   - While $\sum w_i x_i > C$: Remove items (heaviest first)
   - While $\sum w_i x_i < C$: Add items (best $\frac{p_i}{w_i}$ ratio first)
4) **Evaluation:** Calculate fitness $f(x) = \sum_{i=1}^{m} p_i x_i$ for each solution.
5) **Quantum Update:** Modify qubit amplitudes using rotation gates.

$$U(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \quad (3)$$

$$\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = U(\theta_i) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \quad (4)$$

Rotation angles θ_i are determined via a lookup table that compares current and best solutions. For instance, if current bit is 1 and best is 0, rotate to decrease $|1\rangle$ probability using $\theta_i = -0.025\pi$. The magnitude 0.025π balances convergence speed and solution quality.

V. Experimental Validation

Experiments compared GQA against eight classical GA variants on knapsack problems with 100, 250, and 500 items. Strongly correlated datasets used weights $w_i \sim uniform[1, 10)$ and profits $p_i = w_i + 5$. Key findings include:

Solution Quality: GQA with population size 10 (GQA(10)) achieved 612.5 best-case profit for 100 items, better than the best classical GA (582.2) by 5.2%. This advantage increased with problem size, giving a 10.3% improvement for 500 items.

Computational Efficiency: GQA with population size 1 (GQA(1)) completed runs in 0.054 seconds for 100 items — 24 times faster than classical GAs (1.329 seconds). And GQA(10) remained 3.5 times faster and gave even better solutions.

Convergence Behavior: Figure 1 of [1] illustrates GQA's superior convergence characteristics. While classical methods plateaued after 200 generations due to premature convergence, GQA maintained steady improvement toward optimal solutions through 1000 generations. The average population profit consistently went up for GQA, indicating effective maintenance of solution diversity.

VI. Conclusion

The Genetic Quantum Algorithm (GQA) represents a significant advancement in combinatorial optimization by joining quantum computing principles with evolutionary algorithms. Its main innovation is in the qubit-based representation, which compactly encodes solution superpositions and provides efficient exploration of the solution space. By replacing traditional genetic operators with quantum gates, GQA achieves superior convergence speed and avoids premature convergence to local optima. Experimental tests on knapsack problems demonstrate GQA's consistently better performance than of classical genetic algorithms in both solution quality (8–10% higher profits) and computational efficiency (10–100× speedups). The algorithm's ability to maintain solution diversity while rapidly converging near optimal solutions shows its robustness for NP-hard problems. Future work could explore adaptive rotation mechanisms, hybrid mixes with classical methods, and applications to domains like logistics or finance.

VII. Prototype Implementation

An implementation of the analyzed GQA for a reduced number of items can be found here [2]

References

[1] Kuk-Hyun Han and Jong-Hwan Kim, "Genetic quantum algorithm and its application to combinatorial optimization problem," Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512), La Jolla, CA, USA, 2000, pp. 1354-1360 vol.2, doi: 10.1109/CEC.2000.870809.

[2] GQA-Implementation-for-Knapsack-Problem, https://github.com/Pichers/GQA-Implementation-for-Knapsack-Problem