Team: Quantum Explorers

Team Member Name: Keerthika Devi S

Team Member Name: Devadharshika S S

SUMMARY

Universal Statistical Simulator

The Quantum Galton Board (QGB) is designed to produce statistical distributions — particularly binomial and normal — using quantum parallelism to achieve exponential speedup over classical simulation.

In a classical Galton board, the probability of a ball ending in bin kafter n layers is:

where pp is the probability of a rightward deflection. For the unbiased case, p=$\frac{1}{2}$, this converges to a normal distribution for large n via the Central Limit Theorem.

The quantum version prepares a superposition over all possible 2^n paths:

$$P(k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Here each bitstring xx encodes a sequence of left/right decisions, which are then measured to obtain the same binomial distribution - but computed in $O(n^2)$ rather than $O(2^n)$ time.

From Wooden Pegs to Quantum Gates

Each quantum peg operates on a set of qubits q_0,q_1,q_2 (working) and cc (control). The Hadamard gate HH on the control qubit creates:

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle$$

Controlled-SWAP gates (Fredkin) exchange positions of balls conditioned on c, and CNOT gates propagate decisions through the circuit.

Scaling to nn pegs yields a total gate complexity:

$$G(n) \approx an^2 + bn$$

where aa and bb are hardware-dependent constants (here, roughly half of earlier designs).

Beyond the Bell Curve - Bias Control

Bias is introduced by replacing HH with a rotation gate $Rx(\theta)$:

$$R_x(\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

The probability of a rightward deflection becomes:

$$p(\theta) = \cos^2\left(\frac{\theta}{2}\right)$$

This allows precise shaping of the output distribution — even enabling arbitrary probability mass functions through peg-by-peg parameter tuning.

Testing and Performance

The researchers tested their design in two ways:

1. Simulation — using Qiskit locally to verify correctness in an ideal, noiseless environment.
2. Real quantum hardware — running on IBM-QX machines, where noise became a noticeable factor, particularly from the multi-controlled SWAP operations.

In both cases, the QGB produced output in the form of n-bit strings with a single '1'. Post-processing then converted these into more familiar numerical data formats.

Efficiency: The QGB handles 2^n possible paths with only $O(n^2)$ resources — a huge step up from classical methods that must simulate each path one at a time. For larger boards (many layers of pegs), the required number of gates still scales quadratically, which remains feasible for simulations and small-to-medium quantum devices.

Applications Beyond Demonstration

While the QGB is a compelling educational tool for illustrating quantum superposition and parallelism, its potential use cases extend into applied fields:

- Complex systems & networks — modeling random walks on graphs, which are fundamental in epidemiology, traffic modeling, and internet routing.
- Finance — simulating stock price movements, option pricing, and risk models.
- Machine learning — providing quantum-enhanced randomness and efficient sampling for training algorithms.
- Cryptography — enabling new quantum-secure protocols or generating high-quality random keys.
- Sampling & search — rapidly exploring enormous probability spaces in optimization problems.

Limitations and Future Outlook

The algorithm is sound, but present-day NISQ (Noisy Intermediate-Scale Quantum) devices limit what's practical. Major constraints include:

- Gate noise — especially from Fredkin (controlled-SWAP) gates, which are not natively supported on most hardware.
- Qubit count — large boards require more qubits than many current systems provide.

The authors suggest that as hardware improves — either through better native gate support or error-corrected qubits — the QGB could become a staple tool for quantum-enhanced statistical modeling. For now, it complements rather than replaces classical algorithms like the Ziggurat method.

They also released their OpenQASM code, encouraging replication and experimentation by other researchers.

References

[1] https://arxiv.org/abs/2202.01735

[2] https://arxiv.org/pdf/2202.01735.pdf

[3] https://en.wikipedia.org/wiki/Quantum_Fourier_transform

[4] https://inspirehep.net/literature/2026775

[5] https://en.wikipedia.org/wiki/Quantum_simulator

[6] https://fab.cba.mit.edu/classes/862.22/notes/computation/Lloyd-1996.pdf

[7]https://www.fz-juelich.de/en/research/research-fields/information/quantum-technology/quantum-emulators

[8] https://www.bluequbit.io/quantum-computing-simulators

[9] https://www.sciencedirect.com/science/article/abs/pii/S0304885324001501

[10] https://quantum-journal.org/papers/q-2025-06-05-1765/

[11] https://www.ibm.com/quantum/blog/hadron-dynamics-simulations

[12] https://research.ibm.com/haifa/ponderthis/challenges/December2021.html

[13] https://arxiv.org/html/2412.14703v1

[14] https://ieeexplore.ieee.org/document/9951249/

[15] https://link.aps.org/doi/10.1103/PhysRevA.106.032408

[16] https://en.wikipedia.org/wiki/Galton_board