Computational Physics (P452) Presentation

Sampled-Based Guided Quantum Walk

Non-variational Algorithm for Combinatorial Optimization

Based on Nzongani, U. *et al.* (2025); arXiv:2509.15138

By Gayatri P

School of Physical Sciences
NISER, Bhubaneswar

April 13th, 2026

Combinatorial Optimization Problem

- Combinatorial optimization problems involve finding the best, or “optimal” solution from a finite set of discrete possibilities, often in vast search spaces.
- For example, consider the **Maximum Independent Set problem**.

> In a graph, a maximum independent set is the largest subset of vertices in a graph where no two vertices share an edge.

This is an example of a “NP-hard” problem, which means that there are no efficient [i.e.solvable in $O(n^k)$] algorithm known for finding the optimal solution on any graph.

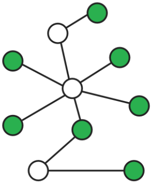

Figure 1: MIS of a graph

Historial Overview: MIS Problem

- MIS was among 21 NP-Complete problems in Richard Karp's monumental 1972 paper[1], which showed that finding an exact solution efficiently for all graphs is unlikely. The runtime for a brute-force approach to the problem runs in $\sim O\left(2^{n^3}\right)$. This problem has various real-life applications, like macromolecular docking, in genome mapping, comparative modeling of protein structures, graph coloring.
- Much work has been done to find *exact* algorithms to solve this, such as the one by Robson[2] [with runtime $O(1.2109^n)$] and Xiao & Nagamochi[3] [with runtime $O(1.1996^n)$].
- In the last decade, research has shifted to using quantum algorithms to solve the MIS problem, especially after it was found that atoms in a Rydberg state natively encodes the MIS constraint[4].

---

[1] Richard M. Karp. "Reducibility among Combinatorial Problems". en. In: *Complexity of Computer Computations*. Boston, MA: Springer US, 1972, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9.

[2] J.M Robson. "Algorithms for maximum independent sets". en. In: *Journal of Algorithms* 7.3 (1986), pp. 425–440. DOI: 10.1016/0196-6774(86)90032-5.

[3] Mingyu Xiao and Hiroshi Nagamochi. "Exact algorithms for maximum independent set". en. In: *Information and Computation* 255 (2017), pp. 126–146. ISSN: 08905401. DOI: 10.1016/j.ic.2017.06.001.

[4] Minhyuk Kim et al. "Rydberg quantum wires for maximum independent set problems". In: *Nature Physics* 18.7 (2022), pp. 755–759. DOI: 10.1038/s41567-022-01629-5.

Motivation

On a Classical Computer:

- In the case of MIS, if a graph has n vertices, there are 2^n possible subsets to go through, if we try to approach the problem through brute force.
- Similarly, for any such combinatorial optimization problem, the solution space is often exponential in size.
- In such cases, quantum algorithms are suited to explore massive solution spaces faster way than classical computers.
- Especially NP-hard problems (problems that have no known algorithm to solve it in $O(|x^k_{\text{inp}}|)$), the only way is to use brute force.

On a Quantum Computer:

- Quantum algorithms are suited to explore massive solution spaces faster way than classical computers.
- Just n qubits can be used to represent the candidate space as a 2^n-dimensional Hilbert space $\mathscr{H}^{\otimes n}$. Hence, the runtime will scale polynomially, instead of exponentially, w.r.t. the input size.

Formulation

- In particular, we will be trying to solve optimization problems of the form,

$$\vec{x}^* = \min_{\vec{x} \in \{0,1\}^n} \left\{ C(\vec{x}) \text{ such that } \vec{x} \in \mathscr{F} \right\} \tag{1}$$

  where $C : \{0,1\}^n \to \mathbb{R}$ is the cost function, and $\mathscr{F}$ the constraint set. $\vec{x}^*$ is the optimal solution. The **optimal solution** has the least value of cost function.

- We also call any $\vec{x} \in \mathscr{F}$ a **feasible decision** (i.e., any binary decision that respects the constraints). If x is a feasible decision, then an $\vec{\epsilon}$-approximate solution ($\vec{\epsilon} > 0$), such that,

$$C(\vec{x}) - C(\vec{x^*}) \leq \epsilon.$$

Formulation

- For an unconstrained problem, the space of solutions scales exponentially as 2^n.
- We can use n qubits to represent the candidate space as a N-dimensional Hilbert space $\mathscr{H}^{\otimes n}$ $(N \leq 2^n)$.
- The cost Hamiltonian will apply a bias on each computational basis state as,

$$H_C = \sum_{j=0}^{N-1} C(j) \left|j\right\rangle \left\langle j\right| . \tag{2}$$

The exact form of H_C depends on the problem we are trying to solve. For example, the cost hamiltonian for the MIS problem is derived later in Eq. (16).

Adiabatic Evolution

- Traditional adiabatic evolution states that if the system starts in the ground state of H_0 and evolves slowly enough, it will remain in the ground state of $H(t)$ during the evolution. Eventually, it will end up in the ground state of H_1 when $t = T$.
- Hamiltonians are chosen such that the ground state of H_0 is easy to prepare and that of H_1 encodes the solution of a combinatorial optimization problem – an adiabatic evolution would lead to the optimal solution[5].

$$H(t) = \left(1 - \frac{t}{T}\right) H_0 + \frac{t}{T} H_1. \tag{3}$$

- For unconstrained problems, the optimal starting point is the transverse magnetic field (the mixing Hamiltonian), whose ground state is a uniform superposition prepared via a Hadamard transform.

$$H_0 = H_M = -\sum_{j=0}^{n-1} \sigma_j^x \tag{4}$$

[5] Edward Farhi et al. "Quantum computation by adiabatic evolution". In: *arXiv quant-ph/0001106* (2000).

Adiabatic Evolution

In short,

$H_0 = H_M \rightarrow$ Ground state prepared from mixer hamiltonian

$H_1 = H_C \rightarrow$ Encodes the optimal solution, which is basically the ground state of the cost hamiltonian (i.e. minimising the cost) after we let it evolve from a superposition of all possible states

Adiabatic Evolution

Addressing questions asked in the presentation.

How how do you map the MIS problem to the adiabatic Hamiltonian eigenvalues, and what do the eigenvalues describe?

- Mapping the MIS problem to an adiabatic Hamiltonian involves encoding the graph structure into a “problem Hamiltonian”, which we call the cost Hamiltonian, H_C.
- The lowest eigenvalue corresponds to the configuration that satisfies all edge constraints (no adjacent vertices) and has the maximum number of selected vertices. It represents the ‘value’ of the Maximum Independent Set.
- By extension, the higher eigenvalues represent solutions that either violate the edge constraints or represent valid independent sets that are not the largest.

Guided Quantum Walk

- GQW generalizes the adiabatic evolution by using a time-dependent hopping rate, $\Gamma(t)$,

$$H(t) = \Gamma(t)\, H_M + H_C. \tag{5}$$

- To maximize amplitude transfer toward optimal solutions, the energy levels of $\Gamma(t) H_M$ and H_C must be balanced.
- On a hypercube, an edge exists between vertices j and k only if they differ by exactly one bit. When H_C acts locally on a single qubit to transition between $|0\rangle$ and $|1\rangle$, the local Hamiltonian for that specific transition is σ_i^x (i.e. with eigenvalues 1 and -1).

$$H_{jk}^M = -\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad H_{jk}^C = \begin{pmatrix} \langle j|H_C|j\rangle & 0 \\ 0 & \langle k|H_C|k\rangle \end{pmatrix} \tag{6}$$

$$\left|\Delta_{jk}^M\right| = 2 \qquad \Delta_{jk}^C = |\, \langle j|\, H_C \,|j\rangle - \langle k|\, H_C \,|k\rangle \,| \tag{7}$$

Guided Quantum Walk

Addressing questions asked in the presentation.

- The solution space can be represented as a hypercube. For a standard hypercube, the eigenvalues are $E_m = N - 2m, m \in [N]$.
- $\left|\Delta_{jk}^{M}\right|$ is the difference between 2 consecutive levels (since j and j adjacent in the initial state). Hence, the energy gaps comes out to be,

$$\left|\Delta_{jk}^{M}\right| = |E_{m+1} - E_m| = (N - 2(m+1) - N + 2m) = 2$$

  This is the minimal energy required to move amplitude from one "shell" of the hypercube to the next.
- Only for $|\Gamma\Delta_{jk}^{M}| \approx \Delta_{jk}^{C}$ the two Hamiltonians' relative strengths are balanced in Eq. (5). Thus, amplitude transfers can only occur efficiently among specific subsets of vertex pairs with approximately this energy gap, this is known as the balancing condition.
  Any imbalance, or one of the terms will dominate the other.

Guided Quantum Walk

The balancing condition $|\Gamma(E)\Delta^M_{jk}| \approx \Delta^C_{jk}$ yields the ideal hopping rate as a function of energy,

$$\Gamma(E) = \frac{\Delta^C(E)}{2} \tag{8}$$

where Δ^C is the average of the largest energy gaps at energy E. As a function of time,

$$\Gamma(t=0) = \Gamma(E_{\text{max}}), \qquad \Gamma(t=T) = \Gamma(E_{\text{min}})$$

Computing this exactly requires exponential resources, which original GQW authors attempted to bypass via variational optimization (by approximating the shape of $\Gamma(t)$ as cubic Bézier curves, which need six parameters to describe)[6].

[6] Sebastian Schulz, Dennis Willsch, and Kristel Michielsen. "Guided quantum walk". In: *Phys. Rev. Res.* 6 (1 2024).

Sample Based Guided Quantum Walk

SamBa-GQW is based on the idea is that one does not need to have the exact spectrum of H_C nor optimize a complex non-linear curve to obtain a good solution. A sufficiently good *estimation* of the spectrum of H_C is enough.

Algorithm 1 SAMBA-GQW

Input: Optimization problem of n variables
Output: An approximate solution

1: *Classical offline part*
   Perform the sampling protocol (**SAMPLER**, Alg. 2) for $q = \text{poly}(n)$ samples
   Build the annealing schedule $\Gamma(t)$ via **BUILDER**, Alg. 3
2: *Quantum online part*
   Run the continuous-time quantum walk evolution.
3: *Measurements*
   Perform poly(n) measurements in the computational basis and keep the best solution

Compute $\langle \Delta^C \rangle$ for q candidate points $x \in \{0,1\}^n$. We will show that with just $q = \text{poly}(n) \sim n^2$, the computation will be efficient.

Randomly select $x \in \mathscr{S} := \{\{0,1\}^n \cap \mathscr{F}\}$

$\downarrow$

Compute the energy gap with x and its neighbors states and we store the largest gap with its associated energy.

$\downarrow$

Remove x and (and any known symmetries of x, depending on the cost function) from $\mathscr{S}$.

Now compute the mean of the stored gaps for their corresponding energy value and obtain an approximation. Then, interpolate between discrete energy gaps the obtain a continuous and smooth hopping rate as a function of the energy.

Algorithm 2 SAMPLER

Input: $1 \leq q \leq N = 2^n$

Output: Set of energies E_S

$S_0 \leftarrow$ set of candidates with $|S_0| \leq 2^n$
$S_1 \leftarrow \emptyset$
$E \leftarrow$ empty dictionary
$\mathrm{B}(x) \leftarrow$ returns the neighbors of state x
for $i \in [1, q]$ **do** $\longleftarrow O(q)$
  Randomly generate $x \in S_0 \setminus S_1$
  $\Delta \leftarrow \{\Delta^C_{x,y} / |\Delta^M_{x,y}| \mid C(x) > C(y), \forall y \in \mathrm{B}(x)\}$ $\longleftarrow O(N_{\text{neigh}} \cdot T_{\text{cost}})$
  $E[C(x)] \leftarrow \text{mean}(E[C(x)] + \max \Delta)$
  Add x (and its symmetric) to S_1
end for
$E_S \leftarrow$ values of E
Add 0 to E_S
Remove all duplicates from E_S
Sort E_S in decreasing order

$$\boxed{T_{\text{sampler}} = O(q N_{\text{neigh}} T_{\text{cost}})}$$

Determining the Hopping Rate

If we starts in a local superposition of states $|j\rangle$ and $|k\rangle$, the probability of measuring the lower energy state $|k\rangle$ in this 2-state system is given by[7]:

$$\mathbb{P}_{jk}(t,\Gamma(E)) = \frac{1}{2} + \frac{\Gamma(E)\delta_{jk}}{\Gamma(E)^2+\delta_{jk}^2}\sin\left(t\sqrt{\Gamma(E)^2+\delta_{jk}^2}\right)^2, \tag{9}$$

where $\delta_{jk} = \Delta_{jk}^C/2$. Under the balancing condition, $\Gamma_{jk}^* = \delta_{jk}$. To find a good low energy solution, we maximise the probability, $\mathbb{P}_{jk}(t,\Gamma^*) = 1$.

$$\mathbb{P}_{jk}(t,\Gamma^*) = \frac{1}{2} + \frac{1}{2}\sin\left(t\Gamma_{jk}^*\sqrt{2}\right)^2 = 1 \implies t_{jk}^* = \frac{\pi}{2\sqrt{2}\Gamma_{jk}^*} \tag{10}$$

$$\implies T^* = \frac{\pi}{2\sqrt{2}}\sum_{jk\in E_M}\frac{1}{\Gamma_{jk}^*} \tag{11}$$

where $T*$ is the total evolution time to give the walker enough time to traverse *all* energy gaps, over the mixer edges E_M.

[7] Sebastian Schulz, Dennis Willsch, and Kristel Michielsen. "Guided quantum walk". In: *Phys. Rev. Res.* 6 (1 2024).

Determining the Hopping Rate

In practice, we only have the approximate value of Γ_{jk}, and we only have a poly(n) number of sample edges E_S instead of E_M. So,

$$T = \frac{\pi}{2\sqrt{2}} \sum_{jk \in E_S} \frac{1}{\Gamma_{jk}} \tag{12}$$

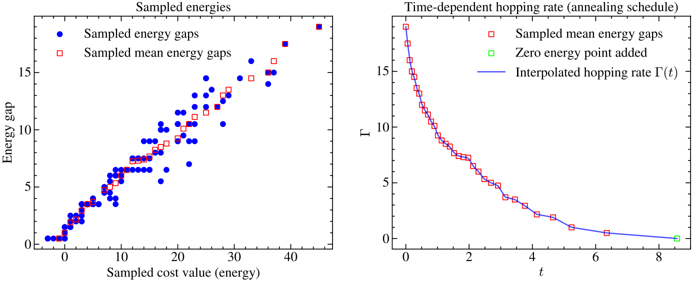

Figure 2: Hopping rate Γ for $n = 11$ qubits (for an MIS instance). (left) Approximated sampling with $q = n^2$ samples. (right) Linear interpolation of the hopping rate Γ. The approximated evolution time is given by Eq. (12).

Algorithm 3 BUILDER

Input: List of sampled energy gaps E_S sorted in decreasing order

Output: The hopping schedule $\Gamma(t)$

1: $dt\_list \leftarrow [0]$
2: $t \leftarrow 0$
3: **for** $\gamma \in E_S$ **do**
4: $\quad t_\gamma \leftarrow \frac{\pi}{2\sqrt{2}}\gamma^{-1}$
5: $\quad$ Add t_γ to dt_list
6: $\quad t \leftarrow t + t_\gamma$
7: $\quad$ Remove γ from E_S
8: **end for**
9: $\Gamma(t) \leftarrow$ interpolation of E_S where the distance between successive sampled points $E_S[i]$ and $E_S[j]$ $(i < j)$ is $dt\_list[i+1]$

Quantum Walk Evolution

To execute the continuous-time evolution on a gate-based quantum computer, the time-dependent Schrödinger equation must be discretized.

$$i\partial_t |\psi(t)\rangle = H(t)|\psi(t)\rangle ; \text{ where } |\psi(t)\rangle = H(t)|\psi(0)\rangle$$

Using the Lie product formula[8], the corresponding unitary operator $U(t)$ is,

$$U(t) = \lim_{p\to\infty}\left(\mathrm{e}^{-\frac{i}{p}\Lambda([0,t])H_M}\mathrm{e}^{-\frac{i}{p}tH_C}\right)^p ; \text{ where, } \Lambda([0,t]) = \int_0^t \Gamma(s)\mathrm{d}s$$

To maintain a manageable circuit depth while preserving accuracy, the evolution is split into nested time intervals:

1. Periods τ_l where the approximate hopping rate Γ is linear.
2. Subintervals p_l within those periods where Trotterization is applied.

[8] Brian C Hall. "Lie groups, Lie algebras, and representations". In: *Quantum Theory for Mathematicians*. Springer, 2013, pp. 333–366.

Quantum Walk Evolution

For a given subinterval indexed by r, the unitary $U(p,r)(T_l)$ is approximated as,

$$U^{(p,r)}(T_l) = \mathrm{e}^{-i\frac{\tau_l}{p_l}\Gamma_{l,r}H_M}\mathrm{e}^{-\frac{\tau_l}{p_l}H_C}$$

$$U(t) = \prod_{l=1}^{q}\left(\prod_{r=1}^{p_l} U^{(p,r)}(T_l)\right)$$

where $\Gamma_{l,r}$ is the average hopping rate value for that specific subinterval, and $U(t)$ is the total quantum circuit evolution, which is the product of all subinterval unitaries.

Each subinterval r_i corresponds to the application of a single layer of cost and mixer Hamiltonians, thus the number of layers in the circuit is $\bar{p} = \sum_l p_l$.

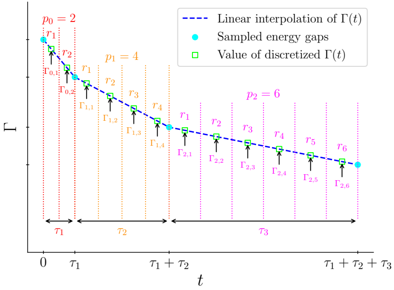

Figure 3: Discretization of the hopping rate Γ. Each interval T_l is cut into p_l subintervals indexed r_i with $i \in \{1, \ldots, p_l\}$.

Circuit Implementation

To decompose this circuit in terms of quantum gates, we split each $U^{(p,r)}(T_l) = U_M^{(p,r)}(T_l)\,U_C^{(p)}(T_l)$ as

$$\begin{cases} U_M^{(p,r)}(T_l) &= \mathrm{e}^{-\frac{i\tau_l}{p_l}\Gamma_{l,r}H_M}, \\ U_C^{(p)}(T_l) &= \mathrm{e}^{-\frac{i\tau_l}{p_l}H_C}. \end{cases}$$

Mixer: $U_M^{(p,r)}(T_l)$, The unitary implementation her for a hypercube (X-mixer) decomposes into rotation gates, because it is a sum of independent, single-qubit Pauli-X operators acting in parallel.

$$U_M^{(p,r)}(T_l) = \bigotimes_{b=0}^{n-1} R_X^b\left(-2\frac{\tau_l}{p_l}\Gamma_{l,r}\right). \tag{13}$$

Cost: The quantum circuit implementation of $U_C^{(p)}(T_l)$ consists of CNOT and R_Z for Quadratic/Higher order unconstrained binary optimization problems[9], and is dependent on the problem in question.

[9] Andrew Lucas. "Ising formulations of many NP problems". In: *Frontiers in Physics* 2 (2014). DOI: 10.3389/fphy.2014.00005.

Circuit Implementation

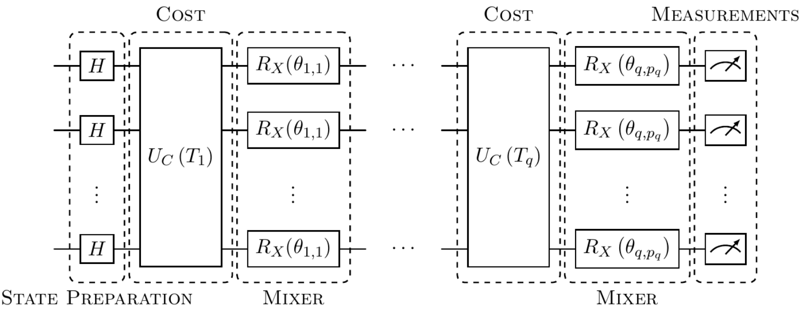

Figure 4: Typical structure of a quantum circuit implementation of the algorithm. The circuit contains layers of cost U_C and mixer U_M unitary operators. The angle of the mixer unitaries depends both on l and r, while the cost unitaries depend only on l.

Algorithm 4 Quantum circuit, (example Qiskit implementation shown in Figure 11)

Input: List of sampled energy gaps E_S sorted in decreasing order, list p of layer numbers

Output: A quantum circuit implementation of the continuous-time quantum walk

```
Add state preparation of initial state (depends on H_M)
for l ∈ {0, …, |E_S| − 1} do
    for r ∈ {1, …, p_l} do
        Γ_l ← E_S[l]
        τ_l ← Γ_l^{−1} · π/(2√2)
        Γ_{l,r} ← Γ_l − (r + 1/2) · (Γ_l − Γ_{l+1})/p_l
        Add exp(−iΔ_t^{(l)} H_C) for Δ_t^{(l)} = τ_l/p_l
        Add exp(−iΔ_t^{(l,r)} H_M) for Δ_t^{(l,r)} = Γ_{l,r} · τ_l/p_l
    end for
end for
Add measurements in the computational basis
```

Maximum Independent Set

- A subset $S \subseteq \mathcal{V}$ is said to be independent if no two vertex of S are adjacent. A graph G is represented by a 2D adjacency matrix.
- Each vertex i is associated to a binary variable $x_i \in \{0,1\}$, we set $x_i = 1$ if $i \in S$. Thus, $n = |\mathcal{V}|$.
- To ensure that the final set is independent, we define a penalty coefficient for including adjacent vertices in S,

$$\lambda = \max_{(i,j)\in\mathcal{E}} (w_i + w_j) + 1. \tag{14}$$

- The cost function will be,

$$C(x) = -\sum_{i\in\mathcal{V}} w_i x_i + \lambda \sum_{(i,j)\in\mathcal{E}} x_i x_j, \tag{15}$$

where w_i is the weight of vertex i. Initially we apply an edge appearance probability of $1/2$, and set $w_i = 1$ for all the vertices.

Maximum Independent Set

For example, consider the graph below.

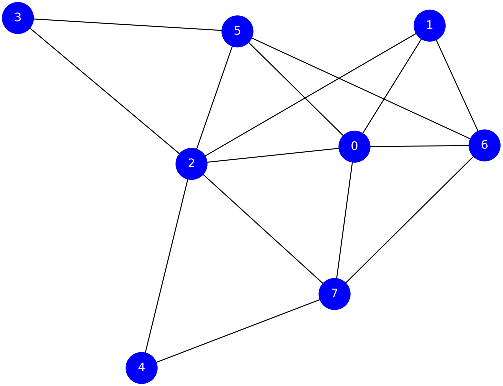

Figure 5: A graph with $n = 8$

Maximum Independent Set: Brute Force Method

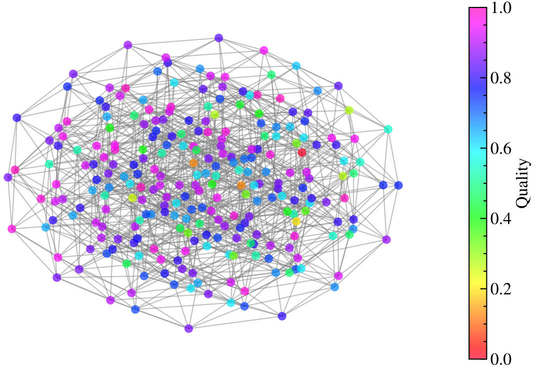

Figure 6: Brute force calculation of cost for all $2^8 = 256$ possible solutions. Every solution is connected to a different solution if they have a hamming weight of 1 (one bit difference). Quality is the measure of how optimal the solution is w.r.t. the cost function (i.e. minimimum cost function corresponds to maximum quality of 1).

Maximum Independent Set: Sampling

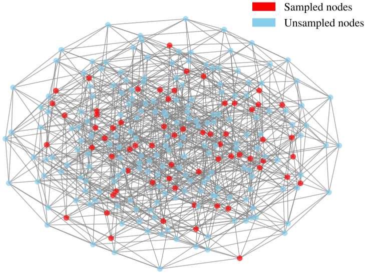

Figure 7: View of all the n^2 sampled nodes.

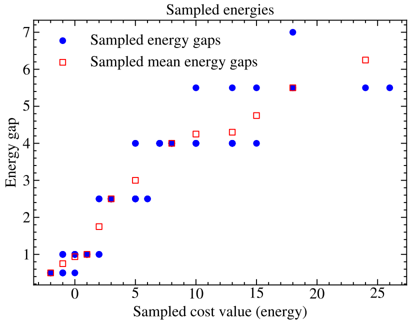

Figure 8: Sampled energy gaps.

Maximum Independent Set: Quantum Walk

Optimal solutions have a ranking of 0, second best have 1, and so on. The probability of measuring a decision of ranking r is

$$\mathbb{P}_r[\psi] = \sum_{x \in S_r} |\langle x|\psi\rangle|^2.$$

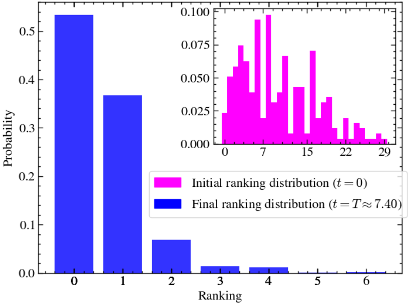

Figure 9: Ranking of optimal solutions.

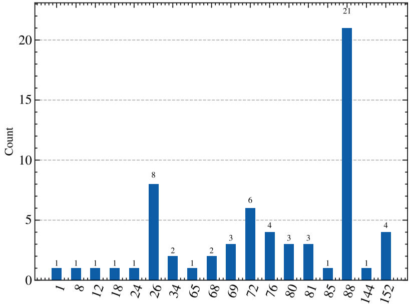

Figure 10: Measurement of the highest probability solution vectors.

Maximum Independent Set: Quantum Circuit

We define the cost and mixer Hamiltonians and the circuit implementing SamBa-GQW in Qiskit. The MIS cost function is:

$$C(x) = -\sum_{i\in\mathcal{V}} w_i x_i + \lambda \sum_{(i,j)\in\mathcal{E}} x_i x_j.$$

Using the relation $x_i = (1-s_i)/2$ with $s_i \in \{-1,1\}$, we rewrite the cost function as:

$$\begin{aligned} C(s) &= -\frac{1}{2}\sum_{i\in\mathcal{V}} w_i + \frac{1}{2}\sum_{i\in\mathcal{V}} w_i s_i + \frac{\lambda}{4}\sum_{(i,j)\in\mathcal{E}} (1 - s_i - s_j + s_i s_j) \\ &= \left(\frac{\lambda}{4}|\mathcal{E}| - \frac{1}{2}\sum_{i\in\mathcal{V}} w_i\right) + \frac{1}{2}\sum_{i\in\mathcal{V}} w_i s_i - \frac{\lambda}{4}\sum_{(i,j)\in\mathcal{E}} (s_i + s_j) + \frac{\lambda}{4}\sum_{(i,j)\in\mathcal{E}} s_i s_j. \end{aligned}$$

Constant term $\frac{\lambda}{4}|\mathcal{E}| - \frac{1}{2}\sum_{i\in\mathcal{V}} w_i$ is a global phase we can ignore, thus the cost Hamiltonian is:

$$H_C = \frac{1}{2}\sum_{i\in\mathcal{V}} w_i \sigma_i^Z - \frac{\lambda}{4}\sum_{(i,j)\in\mathcal{E}} \left(\sigma_i^Z + \sigma_j^Z\right) + \frac{\lambda}{4}\sum_{(i,j)\in\mathcal{E}} \sigma_i^Z \sigma_j^Z \tag{16}$$

where σ_i^Z is the Pauli-Z matrix applied on qubit i.

Maximum Independent Set: Quantum Circuit

Quantum circuit generated for this particular graph instance.

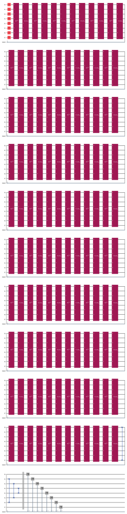

Figure 11: Quantum Circuit

Maximum Independent Set: Solution

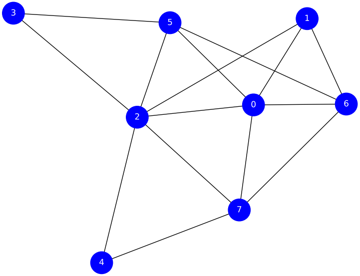

Figure 12: Our initial graph.

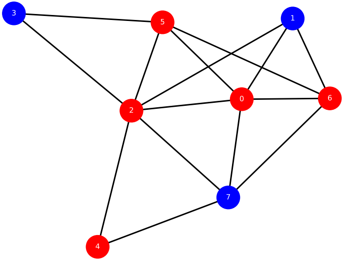

Figure 13: The final solution of the maximum independent set (blue). This matches with the brute force solution as well.

Linear, Quadratic and Cubic sampling

Between quadratic and linear sampling, there is a difference of approximately 0.838 in the probability of measuring the top 5% of rankings, compared to 0.014 between quadratic and cubic sampling. Thus, it is clear that the transition from linear to quadratic sampling has a much greater impact on the quality of the results than that from quadratic to cubic sampling

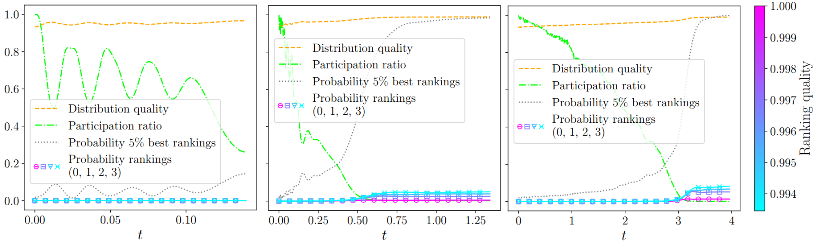

Figure 14: Results for a 20 qubit problem, for 3 different sampling cases.

Travelling Salesperson Problem

- Given a list of n cities with distances between them, is there a route with total distance k that visits all of cities exactly once and returns to the starting point. We need to find the shortest route that visits every city exactly once.
- The cost function is, where w_{ij} is the weight of edge (i,j) and $\lambda, \gamma > \mu \max_{(i,j)} w_{ij}$.
- The subcost C_1 puts a penalty if a sequence encodes an invalid city, i.e. if its input if greater than $n-1$. Let $n-1 = \tilde{x}_{\lceil \log_2(n) \rceil - 1} \cdots \tilde{x}_0 = \tilde{x}$ and S be the set of indices of $\tilde{x}$ equal to zero, i.e. $S = \{j \mid \tilde{x}_j = 0\}$. Hence, the first penalty reads:

$$C_1(x_i) = \sum_{j \in S} x_{i,j} \prod_{k=j+1}^{\lceil \log_2(n) \rceil - 1} \left(1 - (x_{i,k} - \tilde{x}_k)^2\right). \tag{17}$$

- The second subcost C_2 is used to verify if two sequences x and y encode the same city:

$$C_2(x,y) = \prod_{k=0}^{\lceil \log_2(n) \rceil - 1} \left(1 - (x_k - y_k)^2\right). \tag{18}$$

Travelling Salesperson Problem

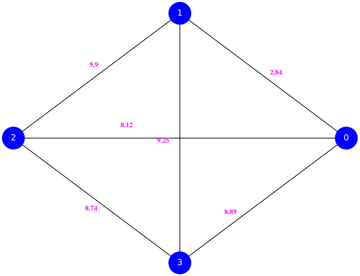

Figure 15: A TSP Problem Graph

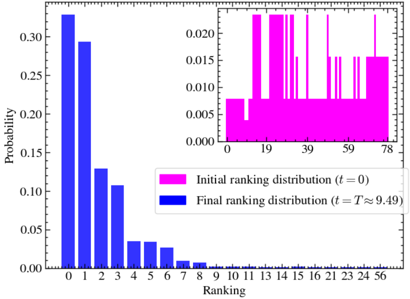

Figure 16: Optimal solution ranking

Travelling Salesperson Problem: Solution

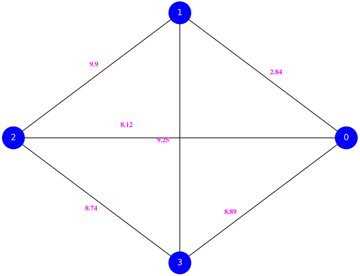

Figure 17: Our initial graph.

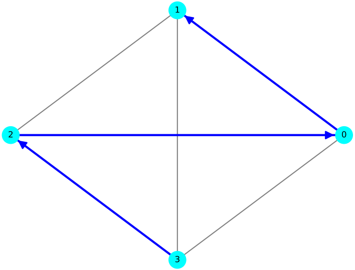

Figure 18: The final solution of the TSP.

Performance & Results

- The algorithm successfully shifts an initial uniform distribution into a very localized distribution around the optimal solution in a short evolution time.
- For problems up to size $n = 20$, sampling just n^2 states empirically yields high-quality approximate solutions.
- SamBa-GQW outperforms the original variational GQW by reducing classical execution time by at least one order of magnitude (e.g., from 24 hours to 9 minutes on $n = 20$, as seen in the original paper)
- By completely bypassing the need for a classical optimizer, it guarantees more stable scaling and avoids the common pitfalls of GQWs (like exponenial scaling and parameter dependency).

References I

[1] Edward Farhi et al. "Quantum computation by adiabatic evolution". In: *arXiv quant-ph/0001106* (2000).

[2] Brian C Hall. "Lie groups, Lie algebras, and representations". In: *Quantum Theory for Mathematicians*. Springer, 2013, pp. 333–366.

[3] Richard M. Karp. "Reducibility among Combinatorial Problems". en. In: *Complexity of Computer Computations*. Boston, MA: Springer US, 1972, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9.

[4] Minhyuk Kim et al. "Rydberg quantum wires for maximum independent set problems". In: *Nature Physics* 18.7 (2022), pp. 755–759. DOI: 10.1038/s41567-022-01629-5.

[5] Andrew Lucas. "Ising formulations of many NP problems". In: *Frontiers in Physics* 2 (2014). DOI: 10.3389/fphy.2014.00005.

References II

[6] Ugo Nzongani et al. *Sampled-Based Guided Quantum Walk*. 2025. DOI: 10.48550/ARXIV.2509.15138.

[7] J.M Robson. "Algorithms for maximum independent sets". en. In: *Journal of Algorithms* 7.3 (1986), pp. 425–440. DOI: 10.1016/0196-6774(86)90032-5.

[8] Sebastian Schulz, Dennis Willsch, and Kristel Michielsen. "Guided quantum walk". In: *Phys. Rev. Res.* 6 (1 2024).

[9] Mingyu Xiao and Hiroshi Nagamochi. "Exact algorithms for maximum independent set". en. In: *Information and Computation* 255 (2017), pp. 126–146. ISSN: 08905401. DOI: 10.1016/j.ic.2017.06.001.

Thank You