Comparative study of quantum optimization methods for train crew scheduling

Jules Duhamel – EPFL and SNCF

SNCF GROUPE

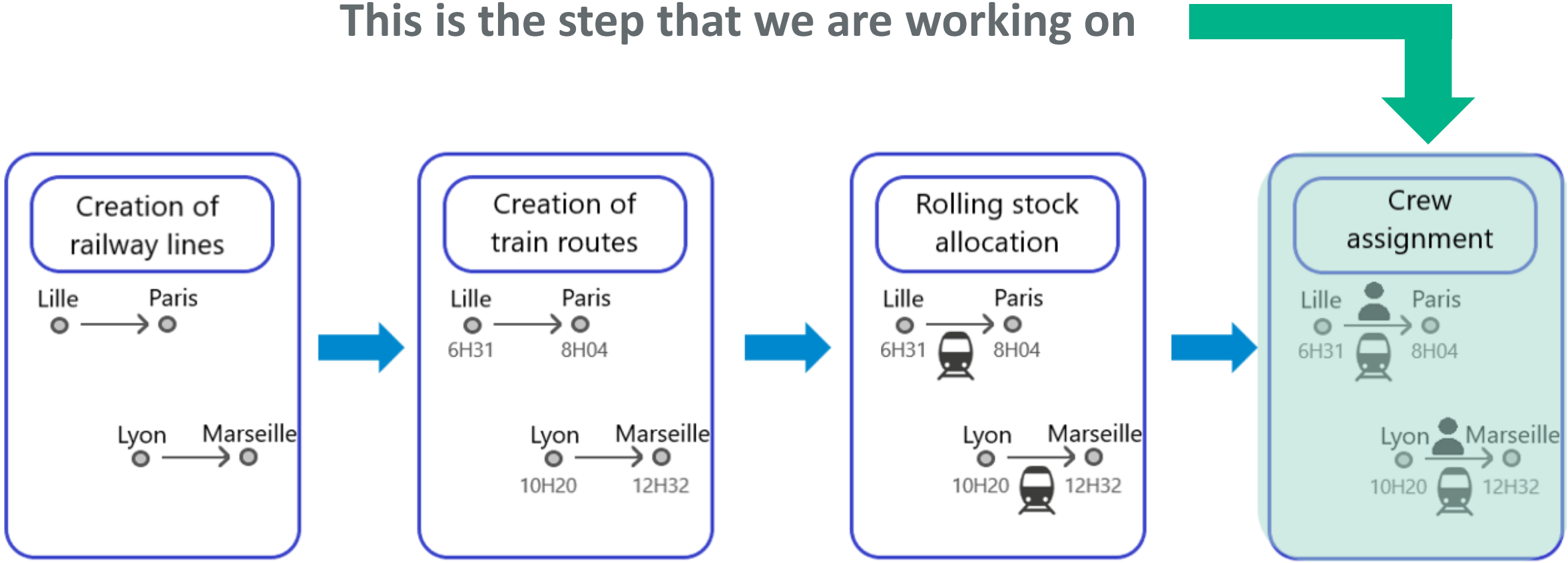
MAISON QUANTIQUE OCCITANIE

FRANCE HYBRID HPCQUANTUM INITIATIVE

Goal:

We want to associate train crews to train trips in a way that minimizes the operational costs.

This is the step that we are working on

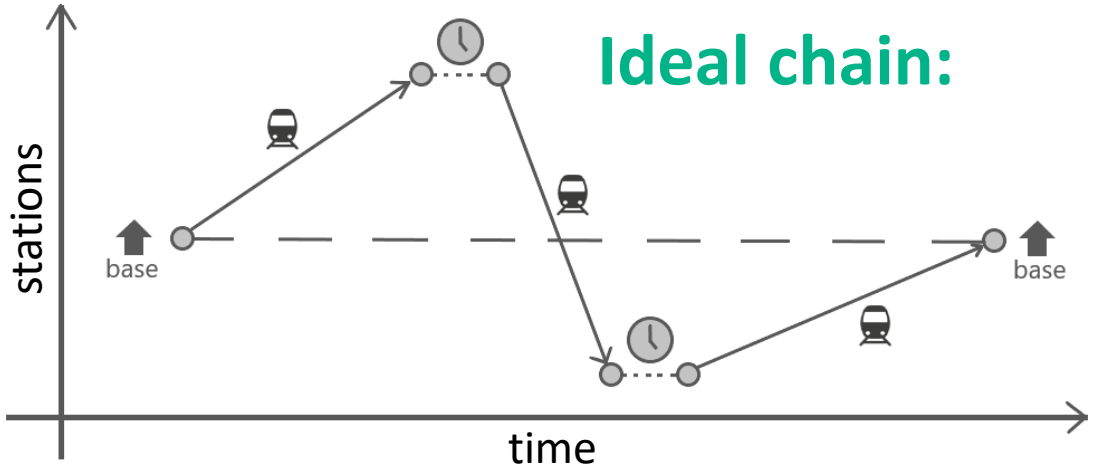


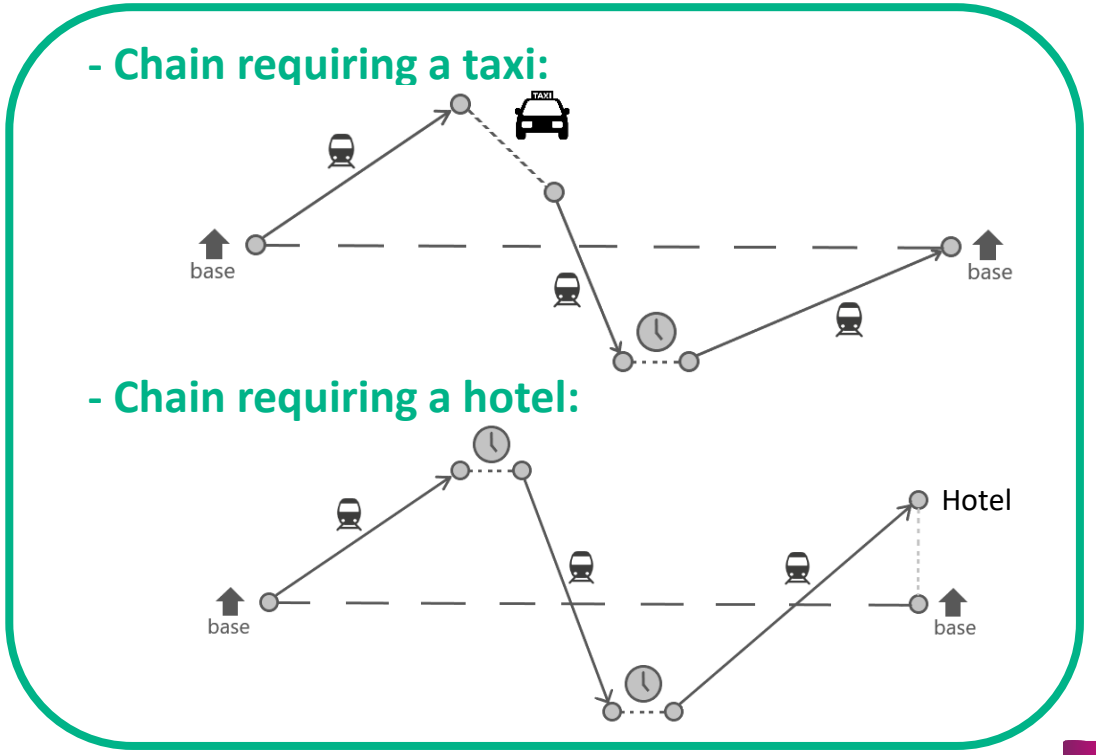
Planification chain

Operational traduction of the goal

We need to create « chains » of trips, that we will attribute to train crews. We must minimize:

- **Taxi** usage between two distant trips
- **Hotel** usage at the end of service of a crew
- **Long waiting times** that are avoidable between two successive trips.



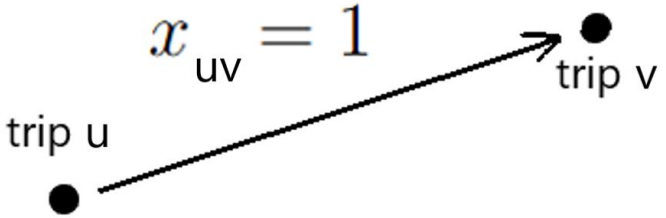


Penalty based methods

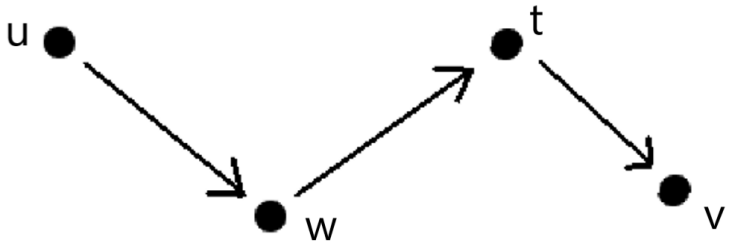
QUBO formulation of the problem:

Main binary variables:

$x_{uv} = 1$ iff a driver from u takes v directly



$y_{uv} = 1$ iff a chain starts with u and ends with v



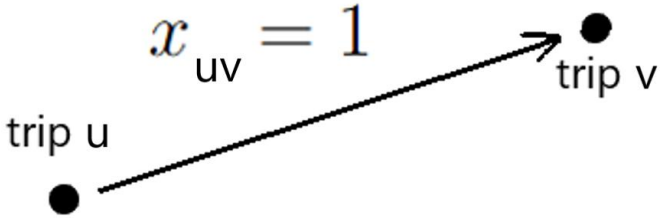
⚠ The nodes represent trips, not stations
The edges represent possible connexions, not trips

Penalty based methods

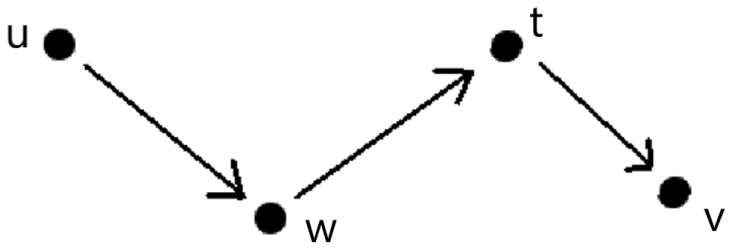
QUBO formulation of the problem:

Main binary variables:

$x_{uv} = 1$ iff a driver from u takes v directly



$y_{uv} = 1$ iff a chain starts with u and ends with v



Cost function:

$$C = \sum_{(u,v)\in U} \mathcal{W}_{uv} x_{uv} + c_h \sum_{(u,v)\in V} H_{uv} y_{uv}$$

Encode waiting times and taxi costs between u and v

Encode hotel costs for a chain from u and v

Penalty based methods

QUBO formulation of the problem:

$$C = \sum_{(u,v)\in U} \mathcal{W}_{uv} x_{uv} + c_h \sum_{(u,v)\in V} H_{uv} y_{uv}$$

⚠ y_{uv} variables have a **non trivial dependence** on x_{uv} variables.
This is the reason why we need **penalty terms.**

Example: Penalty term that imposes a set of chains structure:

$$\mathcal{P}_{\text{chains}} = \lambda_{\text{chains}} \left(\sum_{u\in T} \left(s_u + \sum_{v\in T|(v,u)\in U} x_{vu} - 1 \right)^2 + \sum_{u\in T} \left(e_u + \sum_{v\in T|(u,v)\in U} x_{uv} - 1 \right)^2 \right)$$

⚠ There are many more necessary penalty terms to conciliate x and y variables.

Method 1: Quantum annealing

1) The cost function is translated to the problem Hamiltonian:

$$C = \sum_{(u,v)\in U} \mathcal{W}_{uv} x_{uv} + c_h \sum_{(u,v)\in V} H_{uv} y_{uv} + \sum \mathcal{P} \longrightarrow H_1 = \sum_i h_i \hat{\sigma}_i^z + \sum_{i<j} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z$$

2) The system is prepared in the ground state of a simple Hamiltonian:

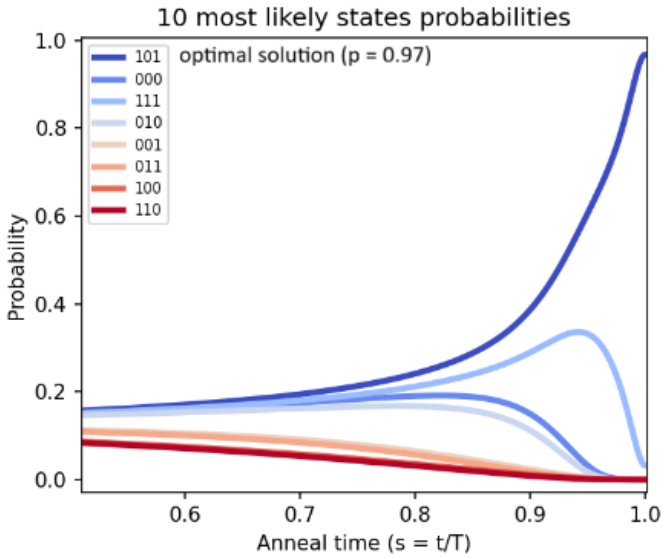
$$H_0 = -\sum_{i=1}^{n} \hat{\sigma}_i^x$$

3) Adiabatic transformation from H0 to H1:

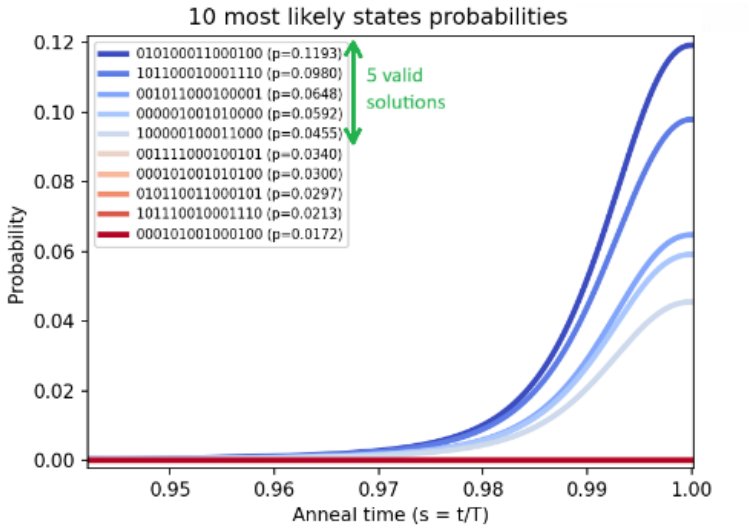
$$H(t) = A(t)H_0 + B(t)H_1$$

Under the adiabatic condition, the system remains in the ground state. At the end of the annealing, the system should be in the ground state of the problem Hamiltonian H1, which encodes the optimal solution.

Quantum Annealing Results



(a) Classical simulation of quantum annealing for a database of 2 trips, after pre treatment (from 10 to 3 qubits). The optimal solution dominates by far with a probability of 97%.

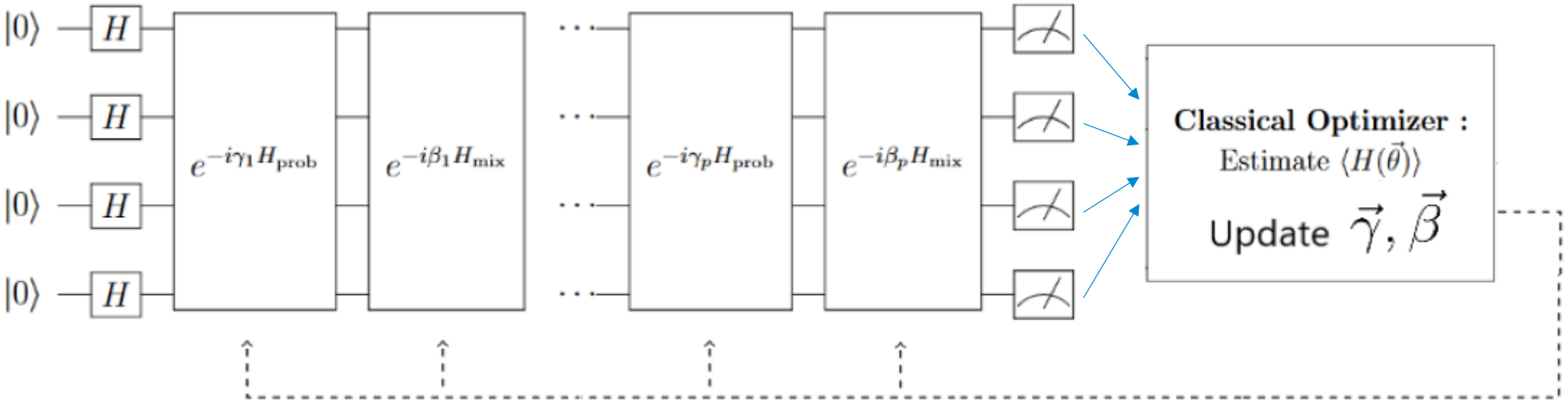


(b) Classical simulation of quantum annealing for a database of 3 trips, after pre treatment (from 22 to 15 qubits). The valid solutions dominate the invalid ones, and the least expensive solutions are effectively the most likely to be measured.

- Only valid solutions emerge. This proves that the QUBO used behaves as expected.
- However, even for 3 trips, the energy gaps between feasible solutions are small, which makes it hard to discriminate high cost feasible solutions.
- This suggests that QA might not be suitable for real size problems.

Method 2: QAOA

QAOA prepares a parameterized quantum state whose amplitudes are **biased** toward **low-cost solutions** of our problem.

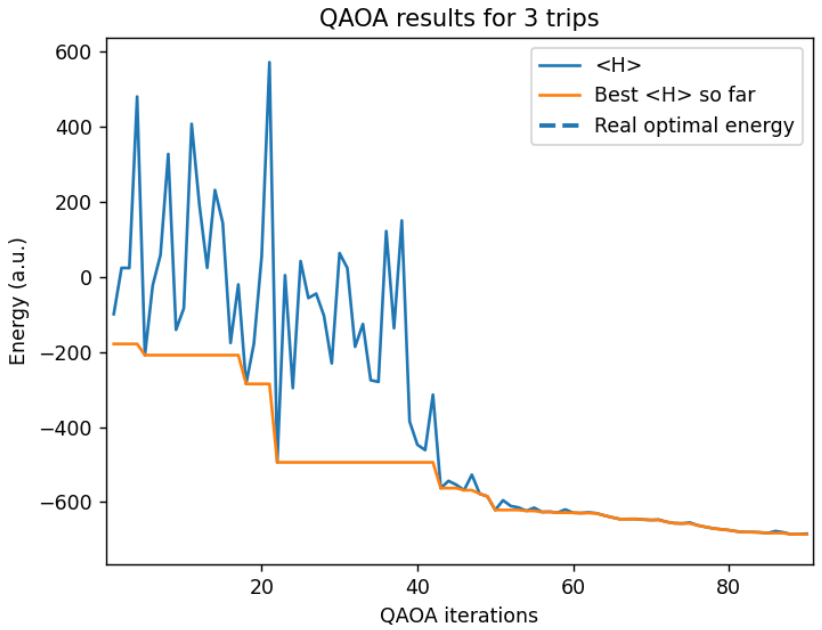


Where **H_prob** is the problem Hamiltonian (Ising).

The **problem** unitary $e^{-i\gamma_l H_{\text{prob}}}$ encodes the cost function.
The **mixer** unitary $e^{-i\beta_l H_{\text{mix}}}$ enables transitions between bitstrings.

QAOA Results

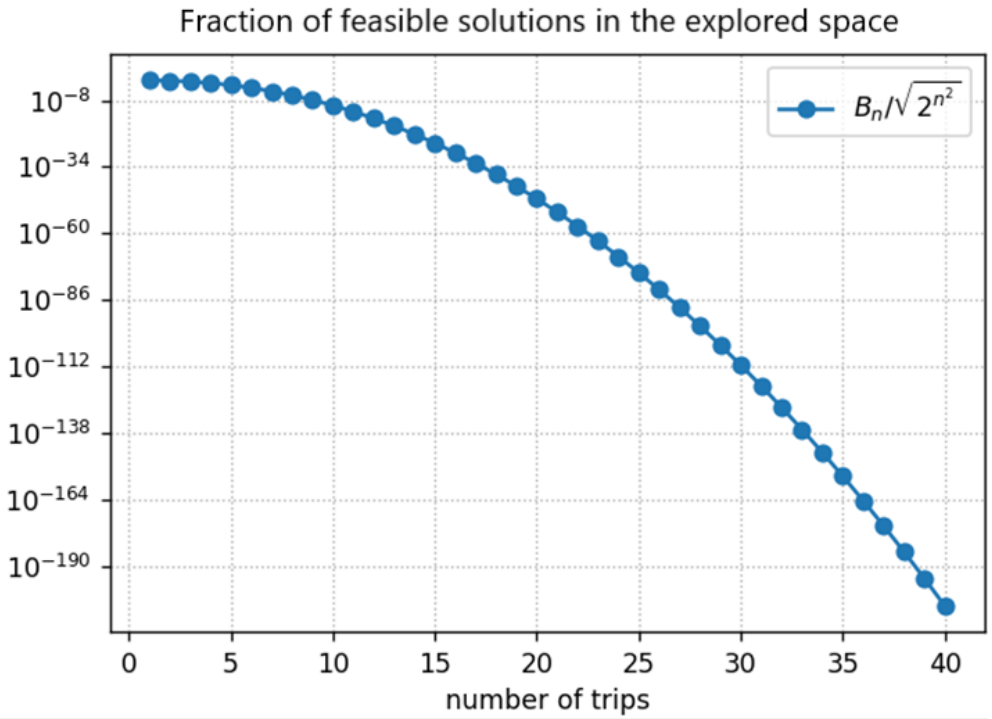
With a depth of p=3, the expected energy after optimization is far from the optimal energy.



- QAOA falls to a local minimum at around <H> = -700 for this 3 trips instance.
- However, all feasible solutions lie at around E = -2800.
- COBYLA fails to even find feasible solutions
- It suggests that feasible solutions are very isolated.

Conclusion: We have a working QUBO that penalizes unfeasible solutions, but the methods using it do not perform well.

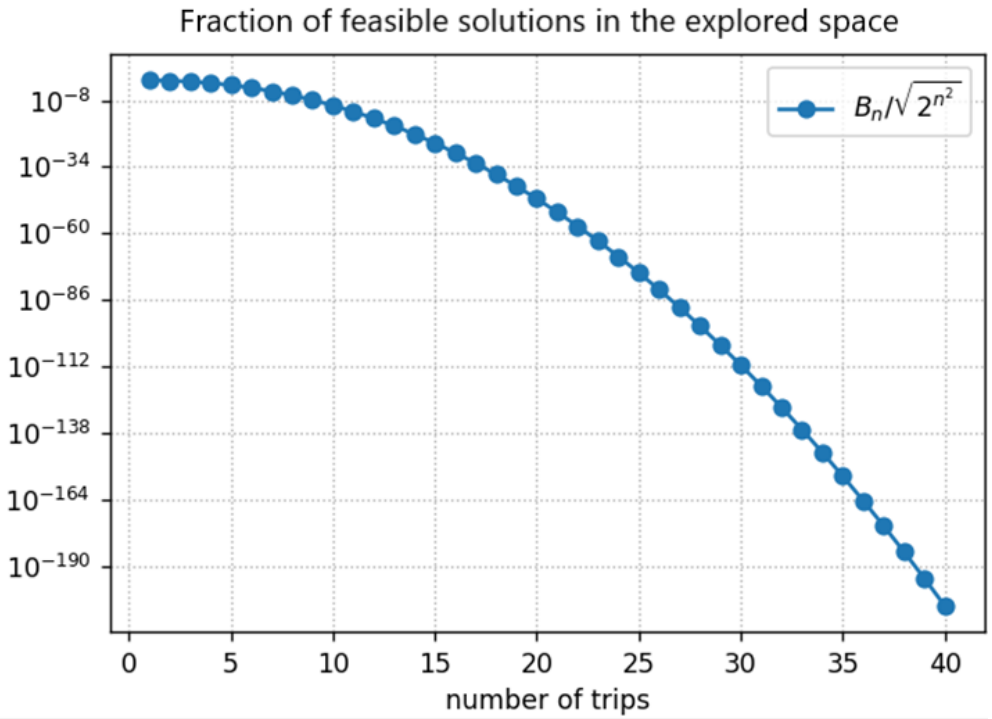
Why are penalty based methods not suited for this problem?



- The ratio of feasible solutions over the total number of solutions is more than exponentially low: $B_N/\sqrt{2^{N^2}}$.

B_N if the Nth Bell number, or the number of feasible solutions for N trips.

Why are penalty based methods not suited for this problem?

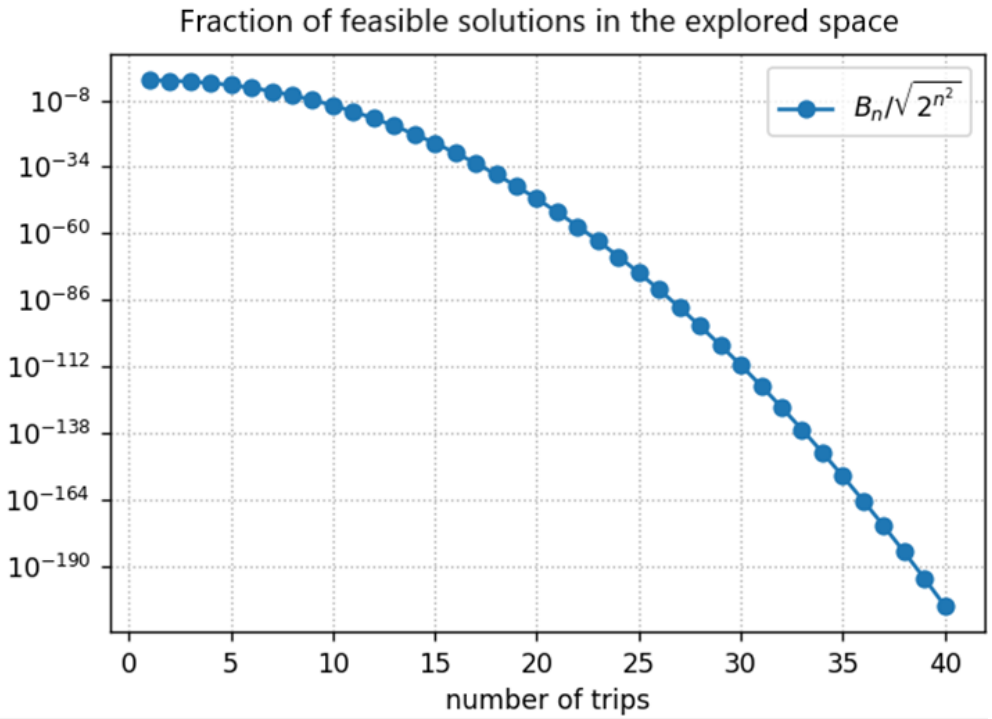


- The ratio of feasible solutions over the total number of solutions is more than exponentially low: $B_N/\sqrt{2^{N^2}}$.

- The algorithms will focus on finding a feasible solution (and fail) rather than finding a near optimal one.

B_N if the Nth Bell number, or the number of feasible solutions for N trips.

Why are penalty based methods not suited for this problem?



- The ratio of feasible solutions over the total number of solutions is more than exponentially low: $B_N/\sqrt{2^{N^2}}$.

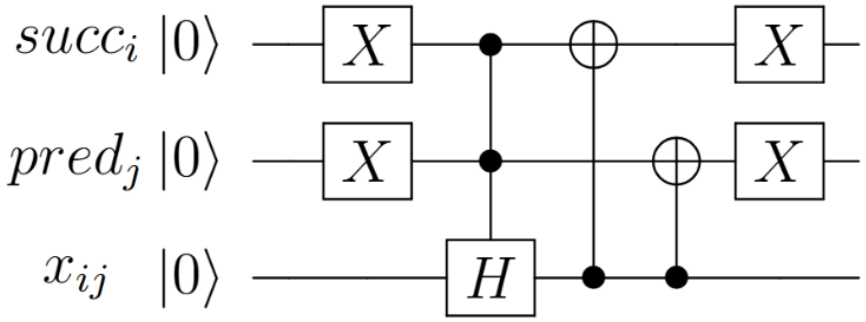
- The algorithms will focus on finding a feasible solution (and fail) rather than finding a near optimal one.

B_N if the Nth Bell number, or the number of feasible solutions for N trips.

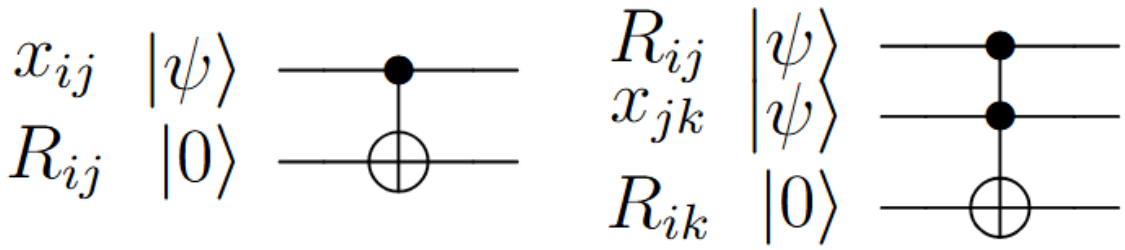
-> We need to find a way to explore only the **subspace of feasible solutions**.

State preparation on feasible solutions

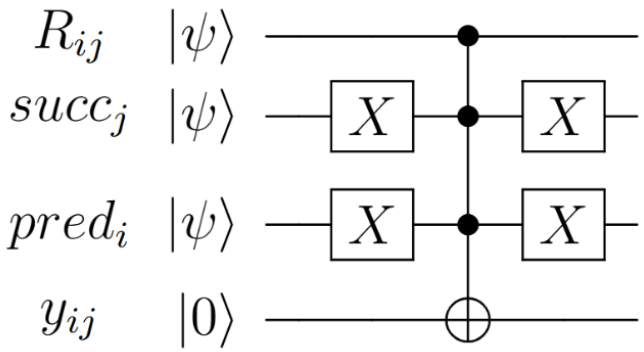
1) **Restriction** to feasible graphs:

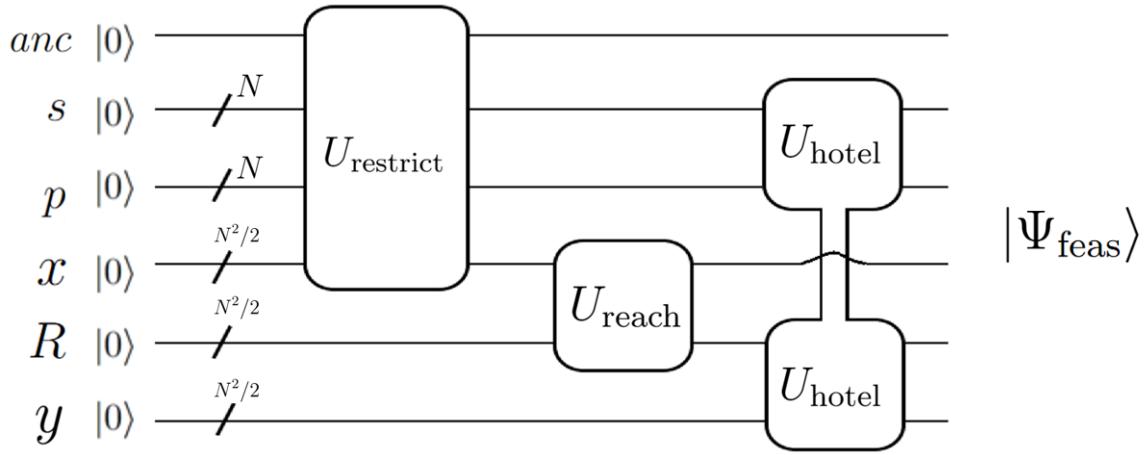


2) We compute the **reachability** variables Rij



3) We compute the need for **hotels** for each solution





State preparation on feasible solutions

We obtain the following state:

$$|\Psi_{\text{feas}}\rangle = \sum_{x \in \text{feas}} c_x |x\rangle \otimes |y(x)\rangle$$

Pros:

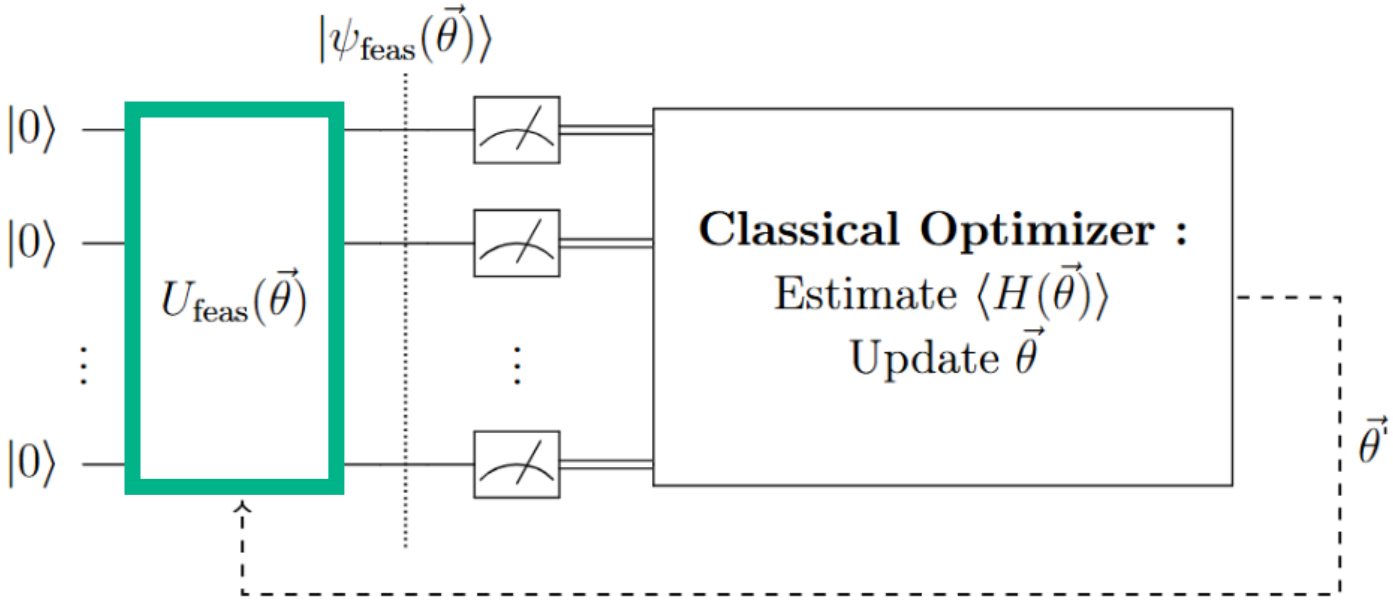
- Only **feasible** solutions can be measured.

Cons:

- Most methods, such as QAOA or QA, can hardly use this state preparation (we must find other algorithms).

Variational Quantum Eigensolver (VQE)

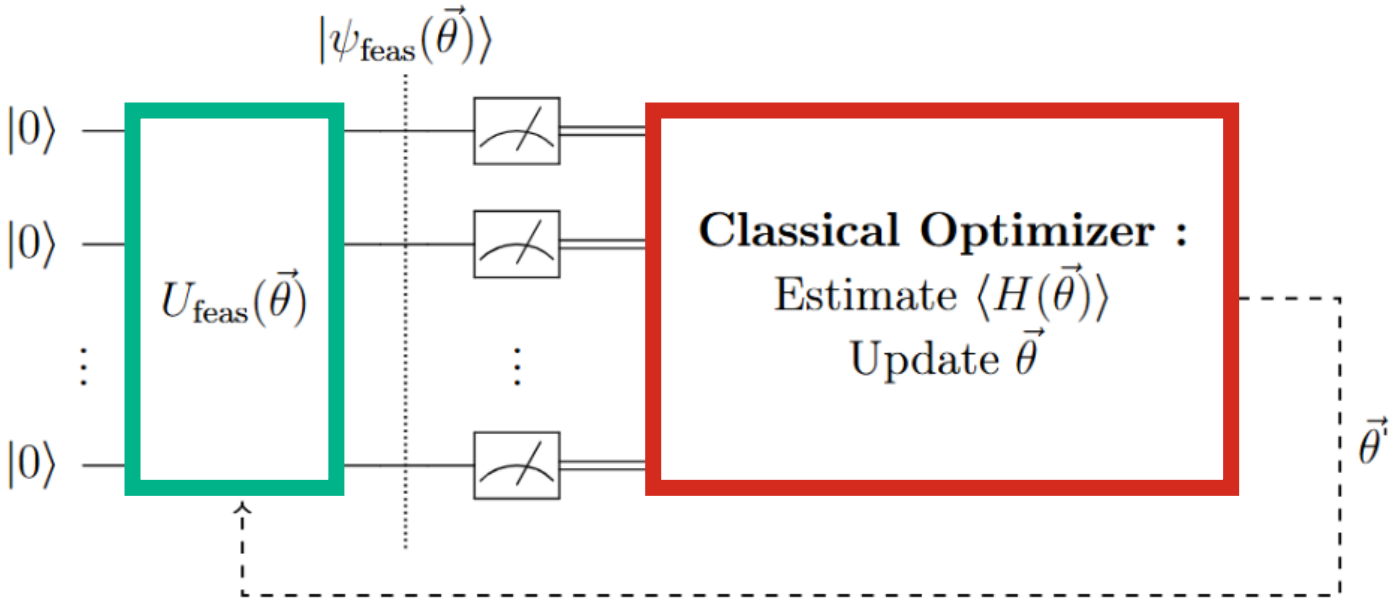
- VQE aims to minimize the expected cost of the measured solutions.
- By **replacing H gates** by $R_Y(\theta)$ gates, we can control how the amplitude of some components are changed, and therefore try to **amplify the near optimal** and optimal components.



Variational Quantum Eigensolver (VQE)

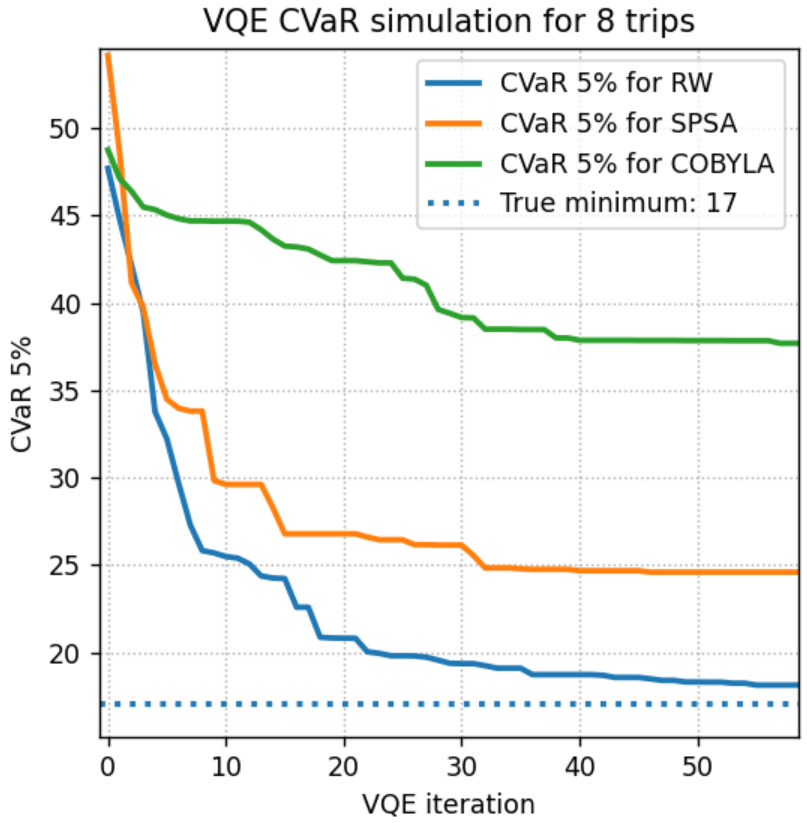
- $U_{\text{feas}}(\vec{\theta})$ becomes a parametrized sampler of feasible solutions.

- $\vec{\theta}$ is updated at each step by a classical optimizer.



VQE Results

- Example for 8 trips:



For 8 trips, there are **4140** feasible solutions.

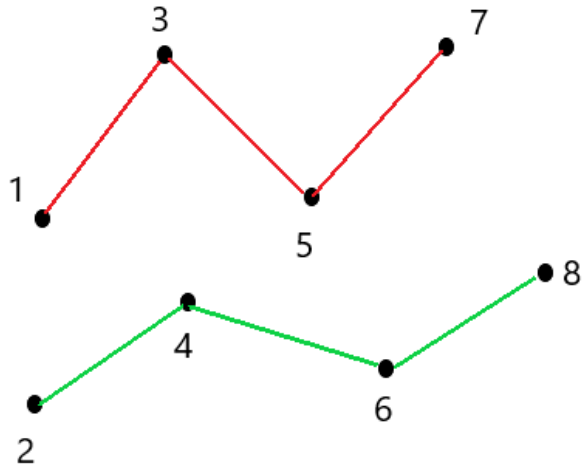
50 shots are used to estimate <H>.

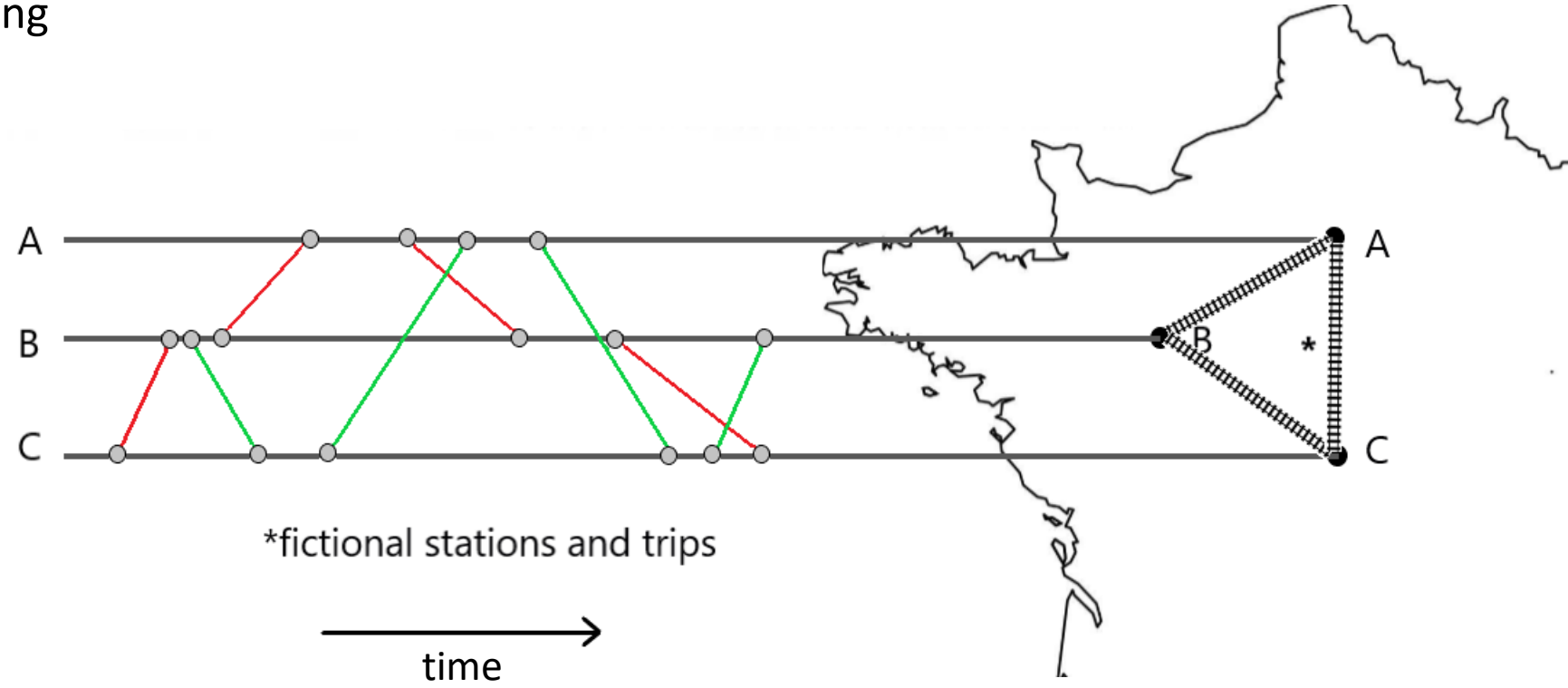
The random walk (RW) reaches the optimal solution the fastest (at 50 iterations, the optimal solution is measured about half of the time).

VQE Results

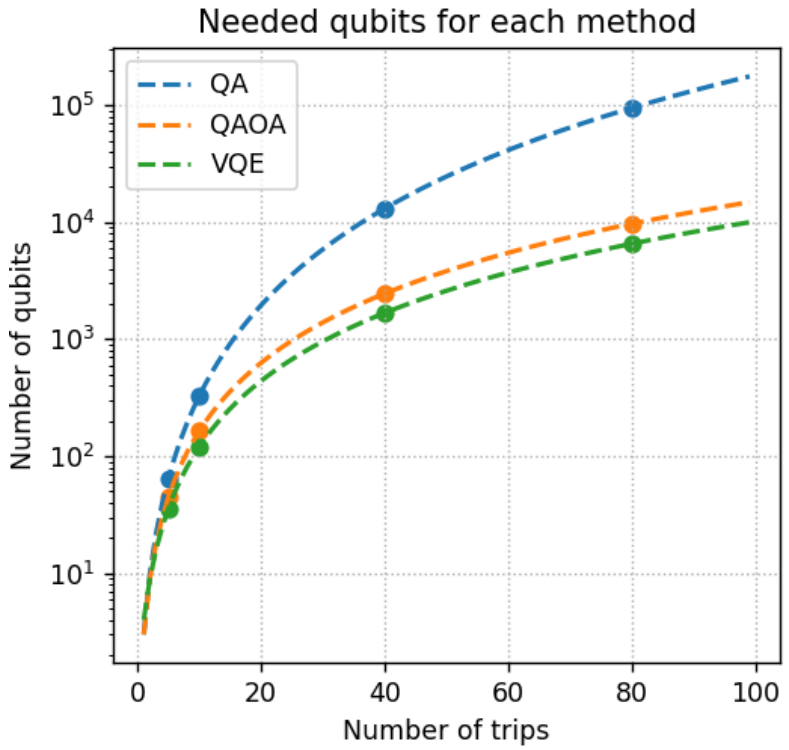
Optimal graph, with a cost of 17:

- no taxi or hotel needed
- minimal waiting





Conclusion



	Penalty based methods (QAOA, QA)	VQE CVaR
Feasible solutions ratio	low	100%
Near optimal solutions:	almost never	good for tested problem sizes

- Penalty based methods imply exploring a space where **non feasible solutions dominate** feasible solutions.

- Building a state preparation that prevents non feasible solutions allowed us to **explore feasible solutions** much more efficiently **with VQE** in the spirit of a **QAOA mixer**.

- However, it is not testable on current quantum computers.