Hybrid classical/quantum algorithms: QAOA

Ville Kotovirta

(Part of Day 2 lecture Hybrid classical/quantum algorithms with Franz Fuchs, Veiko Palge)

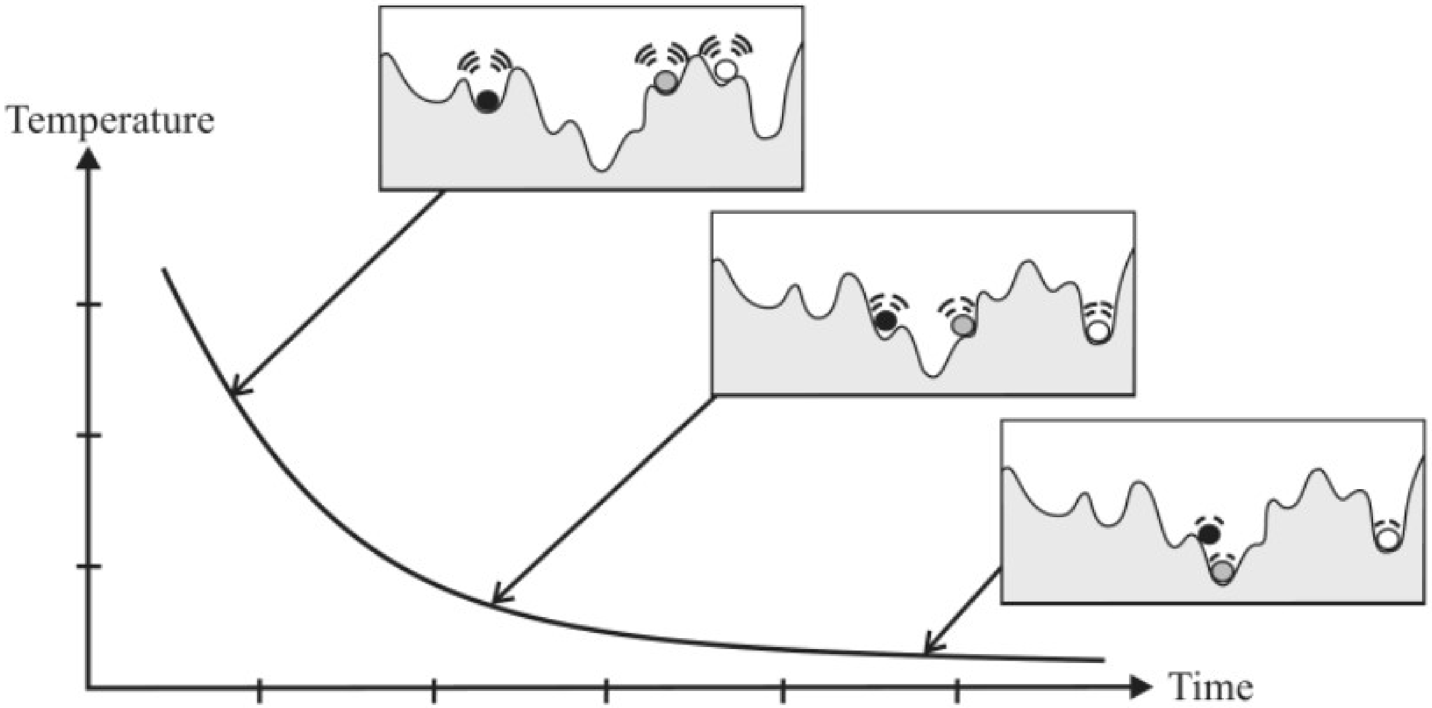
Introduction to Quantum Computing and hybrid HPC-QC systems, 8-9 June 2022

Quantum Approximate Optimization Algorithm (QAOA)

- Variational algorithm for combinatorial optimisation problems
- Good for NISQ devices
  - Low-depth circuit
  - has some inherent robustness against noise (https://arxiv.org/abs/1909.02196)
- Idea is to encode the optimization problem as an energy landscape and find the optimal solution as the global minimum
- Motivated by classical Simulated Annealing and Quantum Annealing

Farhi, Goldstone, Gutmann. 2014. A Quantum Approximate Optimization Algorithm. https://arxiv.org/abs/1411.4028

Annealing principle



Sergio Ledesma, Jose Ruiz and Guadalupe Garcia. Simulated Annealing Evolution, 2012, DOI: 10.5772/50176

QAOA steps (verbal)

1. Describe the objective function as an energy operator, i.e. Hamiltonian
2. Create the QAOA ansatz, a circuit consisting of:
   a) Initialization to a uniform superposition over computational basis states
   b) Two parametrized operators repeated p >=1 times
      1) One operator that approximates Hamiltonian evolution based on the objective function (towards local minimum)
      2) one that shuffles the quantum state in order to explore the energy landscape (in order to avoid local minima)
3. Use quantum computer to sample the Hamiltonian state
4. Compute the expectation value of the Hamiltonian energy, and use optimisation methods (e.g. gradient descent) to update parameters towards maximising the expectation value
5. Repeat 3 and 4 until convergence
6. Analyse the sample distribution, a sufficient number of iterations will produce a state which represents a close enough solution to the ground state of the Hamiltonian

QAOA steps (math)

- 1. Problem

$$C(z) = \sum_{\alpha=1}^{m} C_\alpha(z)$$

- where $z = z_1 z_2 \ldots z_n$ is the bit string and $C_\alpha(z) = 1$ if z satisfies clause and 0 otherwise

- 2. Circuit
  - 2.a Initialization

$$|s\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle$$

  - 2.b.1 Hamiltonian evolution

$$U(C,\gamma) = e^{-i\gamma C} = \prod_{\alpha=1}^{m} e^{-i\gamma C_\alpha}$$

  - 2.b.2 Shuffling

$$B = \sum_{j=1}^{n} \sigma_j^x \qquad U(B,\beta) = e^{-i\beta B} = \prod_{j=1}^{n} e^{-i\beta\sigma_j^x}$$

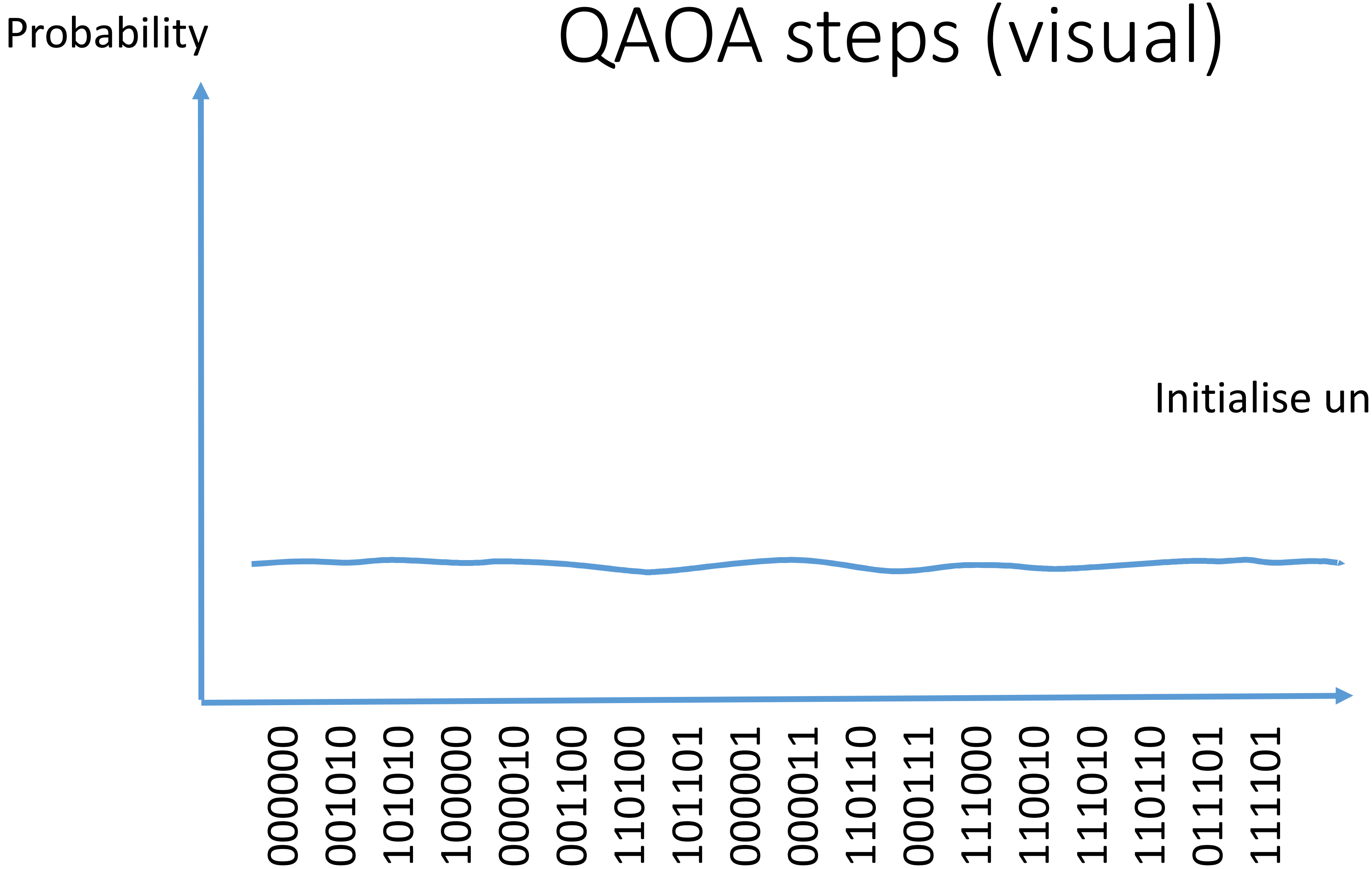
- 3. Parameter-dependent state

$$|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle = U(B,\beta_p)\,U(C,\gamma_p)\cdots U(B,\beta_1)\,U(C,\gamma_1)\;|s\rangle$$

- 4. Optimisation of expectation value

$$F_p(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle\boldsymbol{\gamma},\boldsymbol{\beta}|\,C\,|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle \qquad M_p = \max_{\boldsymbol{\gamma},\boldsymbol{\beta}} F_p(\boldsymbol{\gamma},\boldsymbol{\beta})$$

QAOA steps (visual)



Initialise uniform superposition

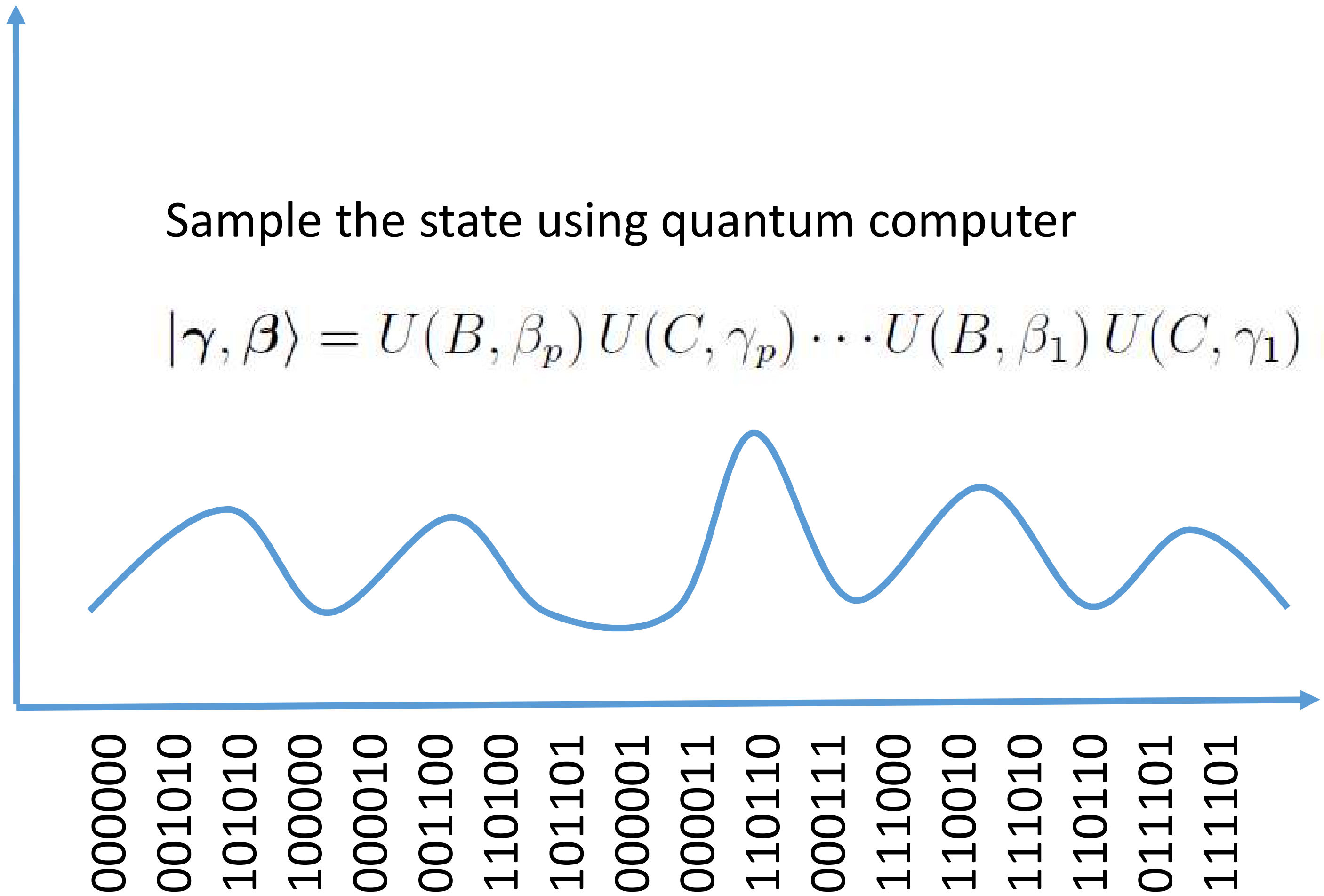
$$|s\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle$$

QAOA steps (visual)

Probability

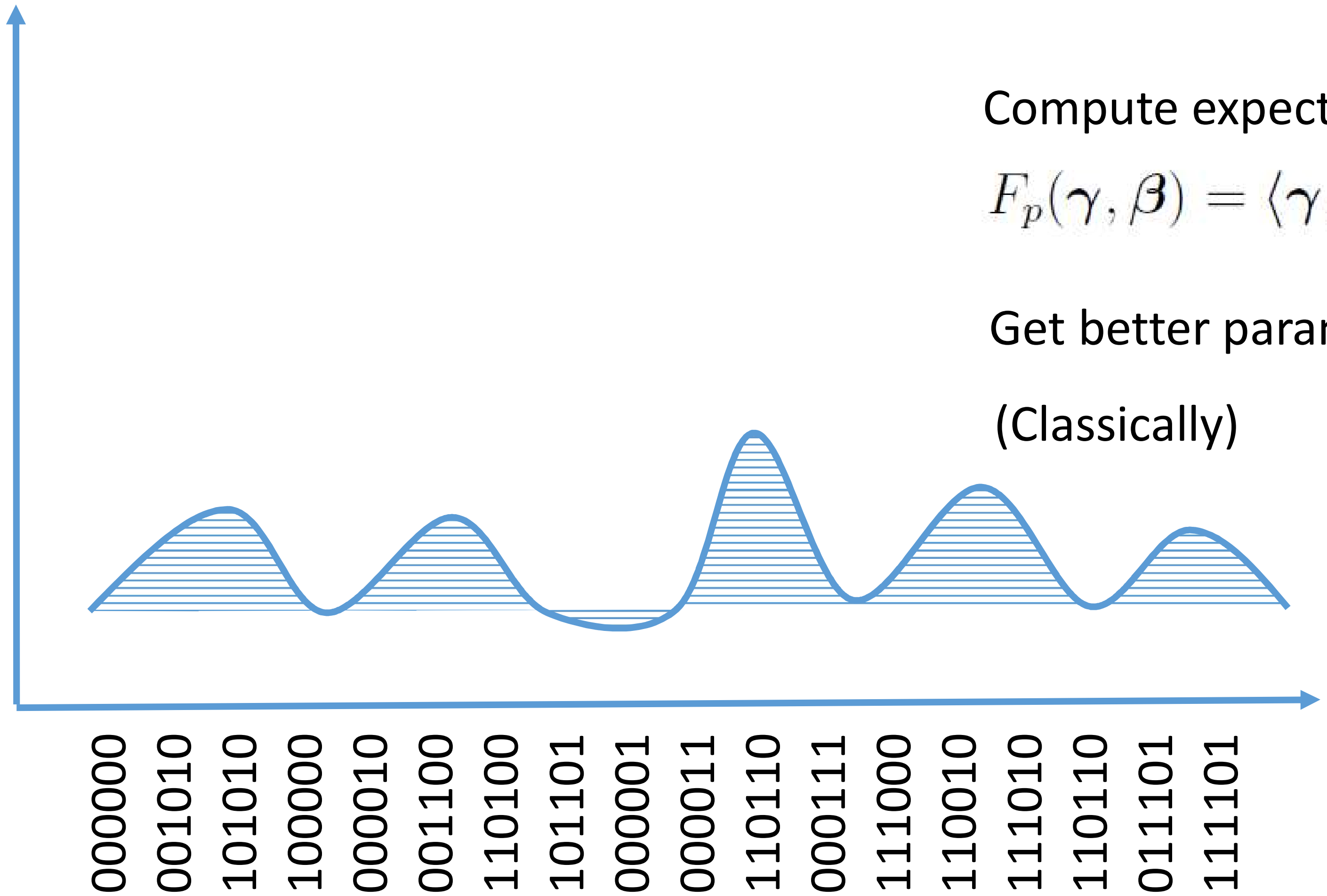
Sample the state using quantum computer

$$|\gamma, \beta\rangle = U(B, \beta_p)\, U(C, \gamma_p) \cdots U(B, \beta_1)\, U(C, \gamma_1)\ |s\rangle$$



QAOA steps (visual)

Probability



Compute expectation value

$$F_p(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle$$

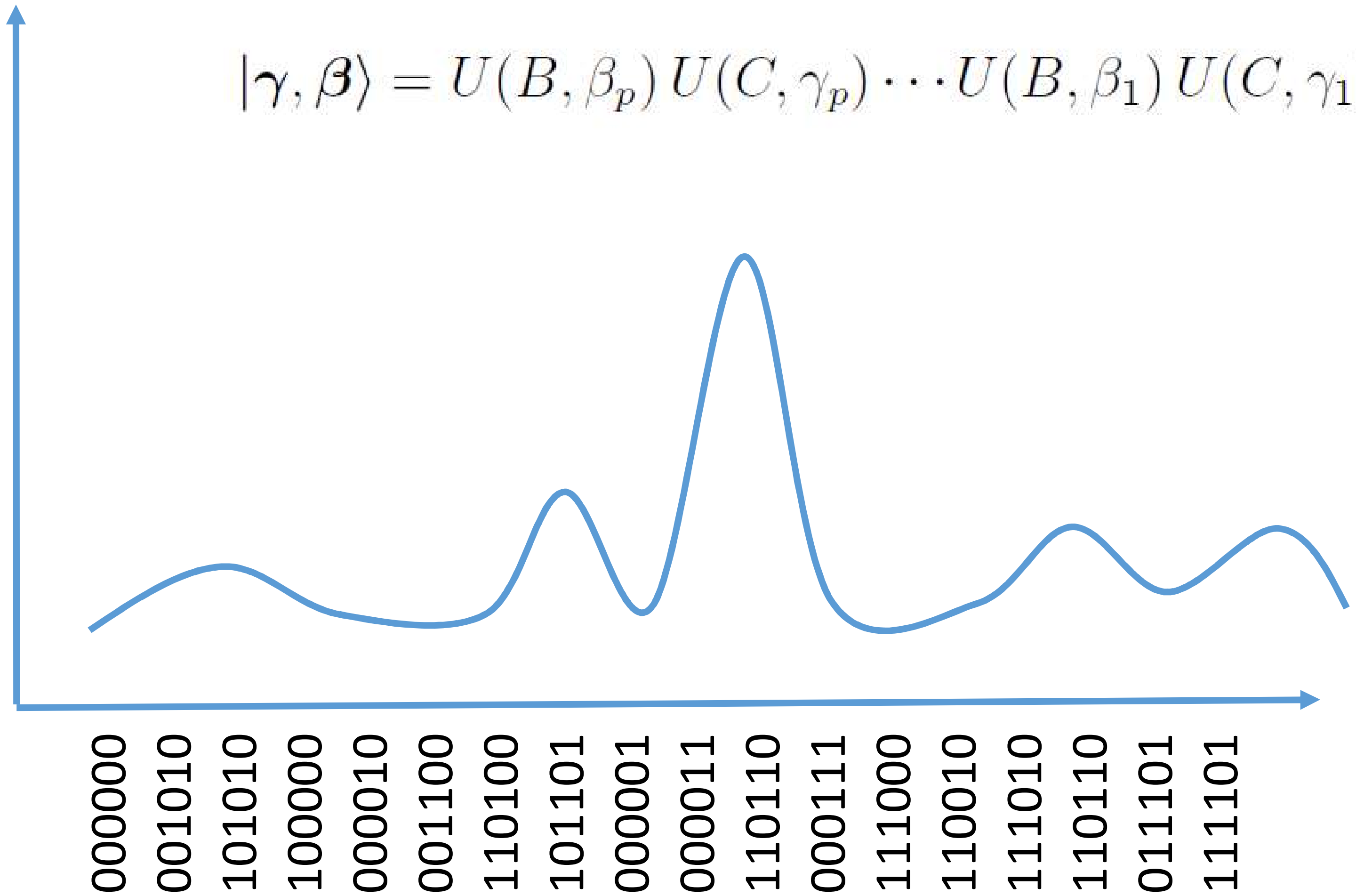
Get better parameters γ, β

(Classically)

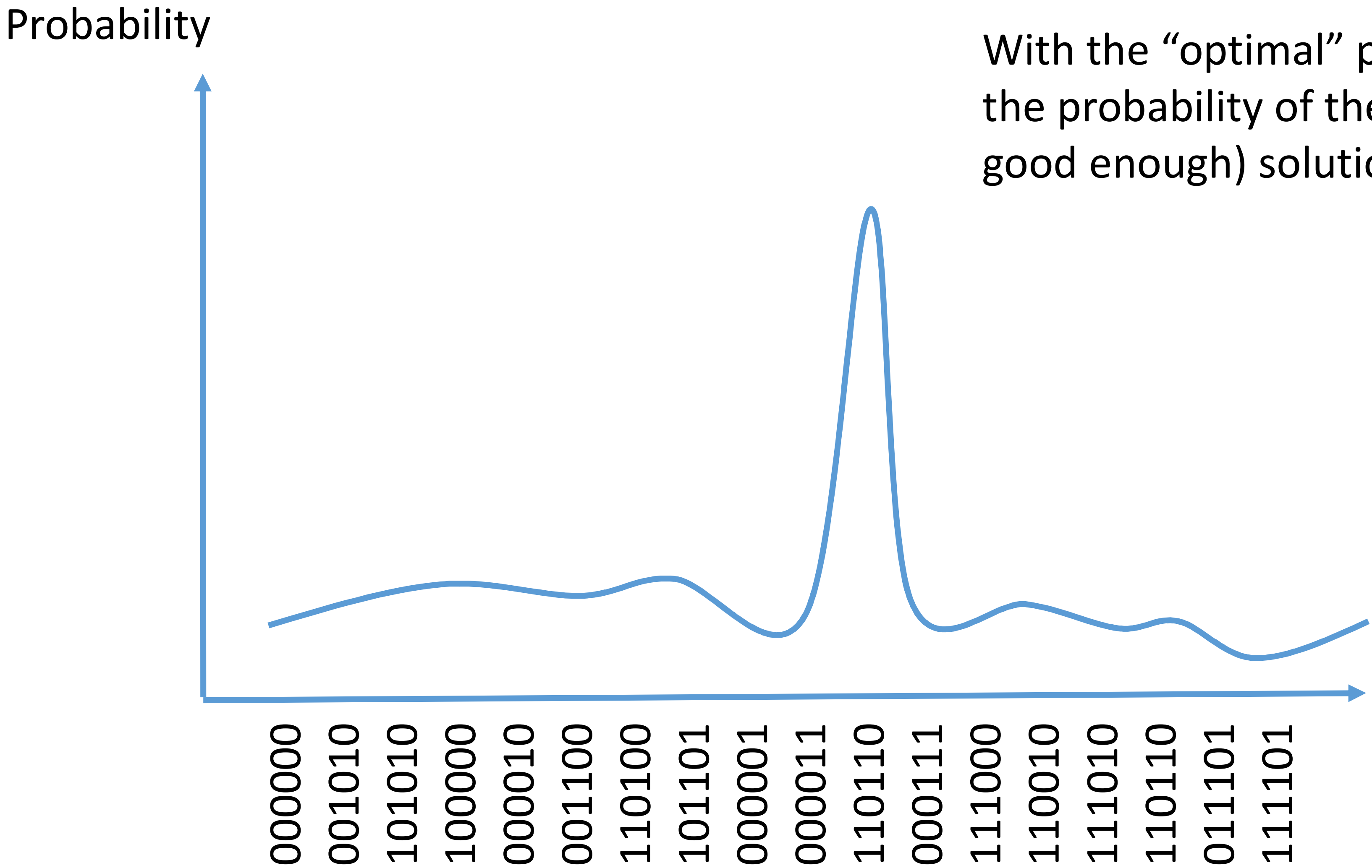
Probability

Sample with better parameters

$$|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = U(B, \beta_p)\, U(C, \gamma_p) \cdots U(B, \beta_1)\, U(C, \gamma_1)\; |s\rangle$$

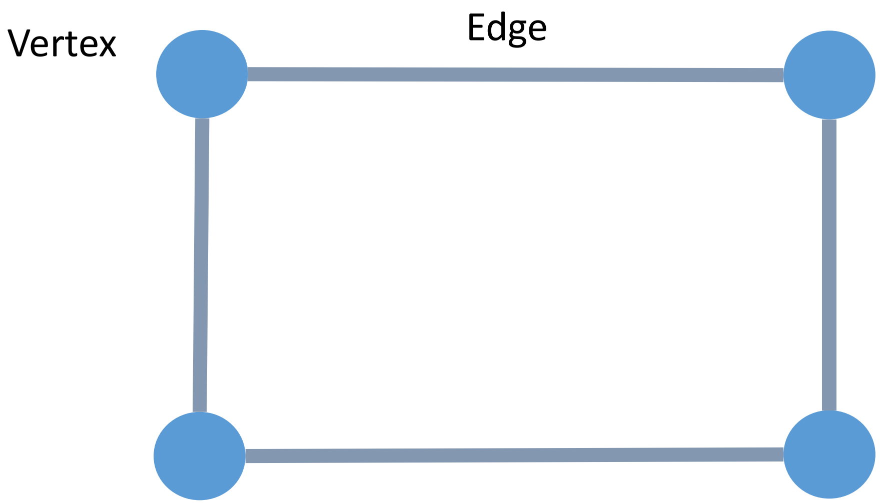


With the “optimal” parameters the probability of the optimal (or good enough) solution increases



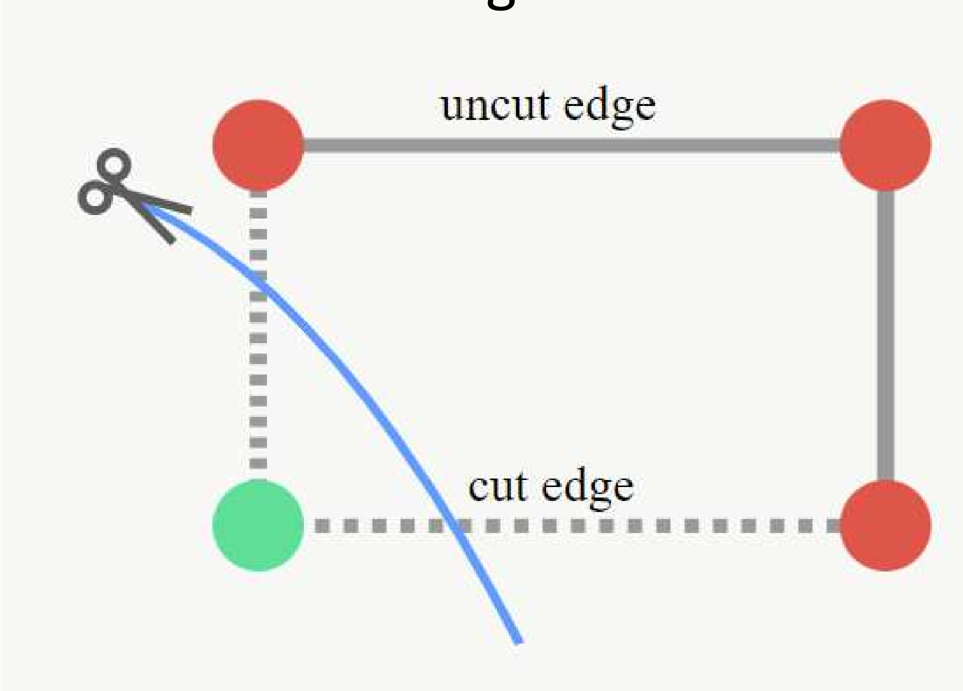
MaxCut problem example

- NP-hard optimization problem from graph theory
- Applications in fields such as network design, statistical physics, circuit layout design, data clustering.
- Find two subsets of vertices such that the number of edges between the two subsets gets maximized



MaxCut problem example

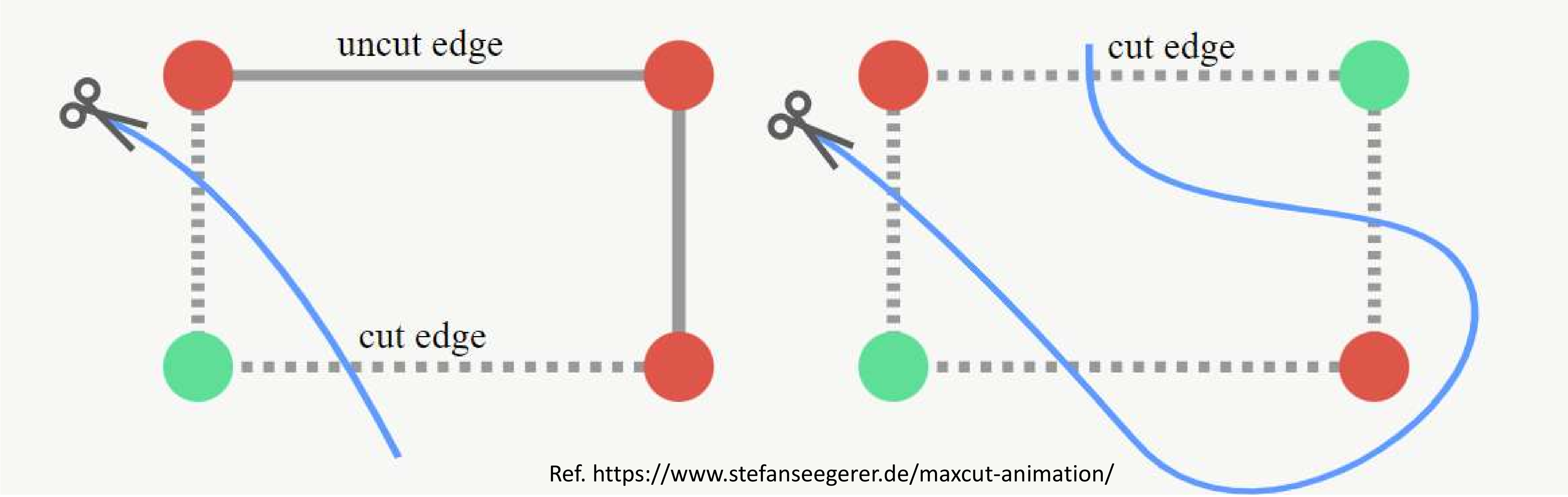
- NP-hard optimization problem from graph theory
- Applications in fields such as network design, statistical physics, circuit layout design, data clustering.
- Find two subsets of vertices such that the number of edges between the two subsets gets maximized



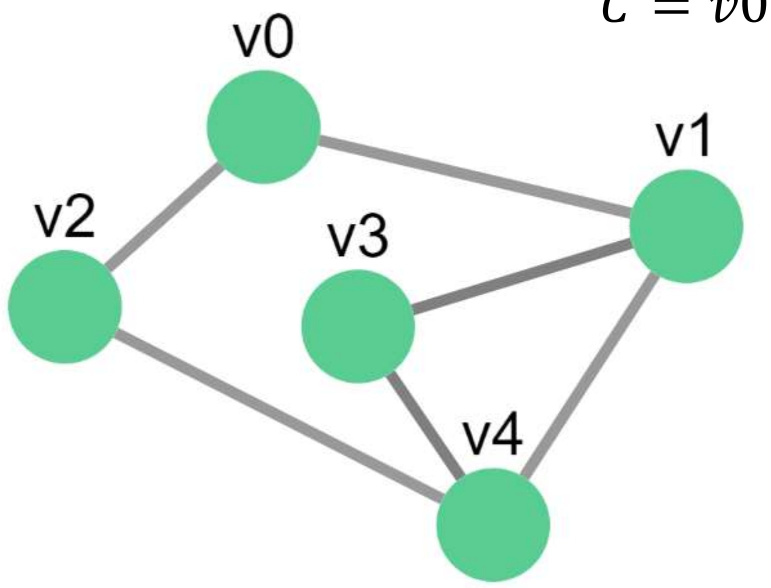
MaxCut problem example

- NP-hard optimization problem from graph theory
- Applications in fields such as network design, statistical physics, circuit layout design, data clustering.
- Find two subsets of vertices such that the number of edges between the two subsets gets maximized

Maximum cut



Ref. https://www.stefanseegerer.de/maxcut-animation/

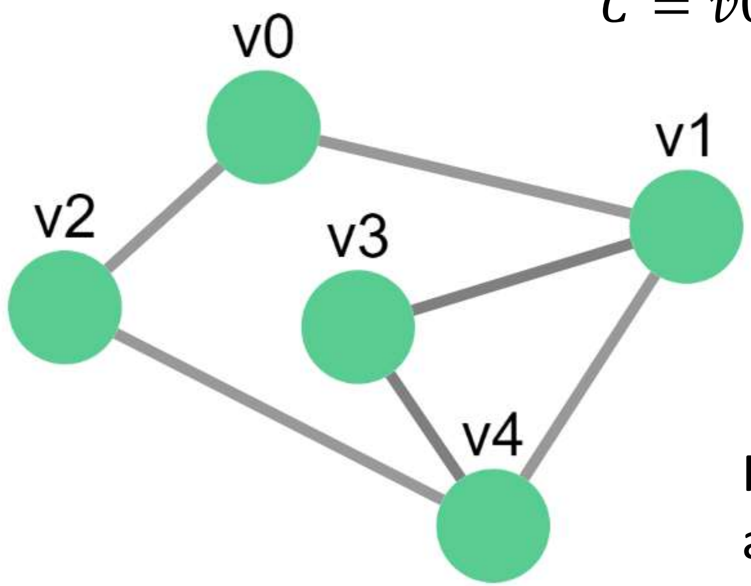


Assign a binary variable to each node to create the objective function:

$$C = v0 \oplus v1 + v0 \oplus v2 + v1 \oplus v3 + v1 \oplus v4 + v2 \oplus v4 + v3 \oplus v4$$

1, if $v0 \neq v1$
0, if $v0 = v1$

(Here a weight of 1 is used for each link, for real-world tasks weights can be anything and represent e.g. the link cost)



Assign a binary variable to each node to create the objective function:

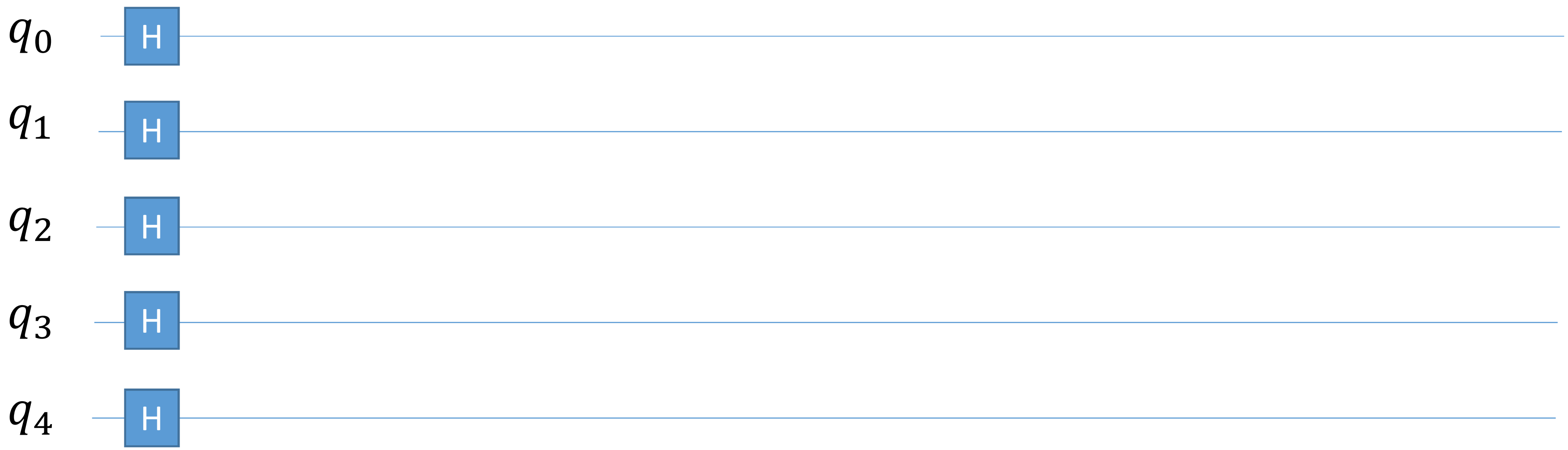
$$C = v0 \oplus v1 + v0 \oplus v2 + v1 \oplus v3 + v1 \oplus v4 + v2 \oplus v4 + v3 \oplus v4$$

1, if $v0 \neq v1$
0, if $v0 = v1$

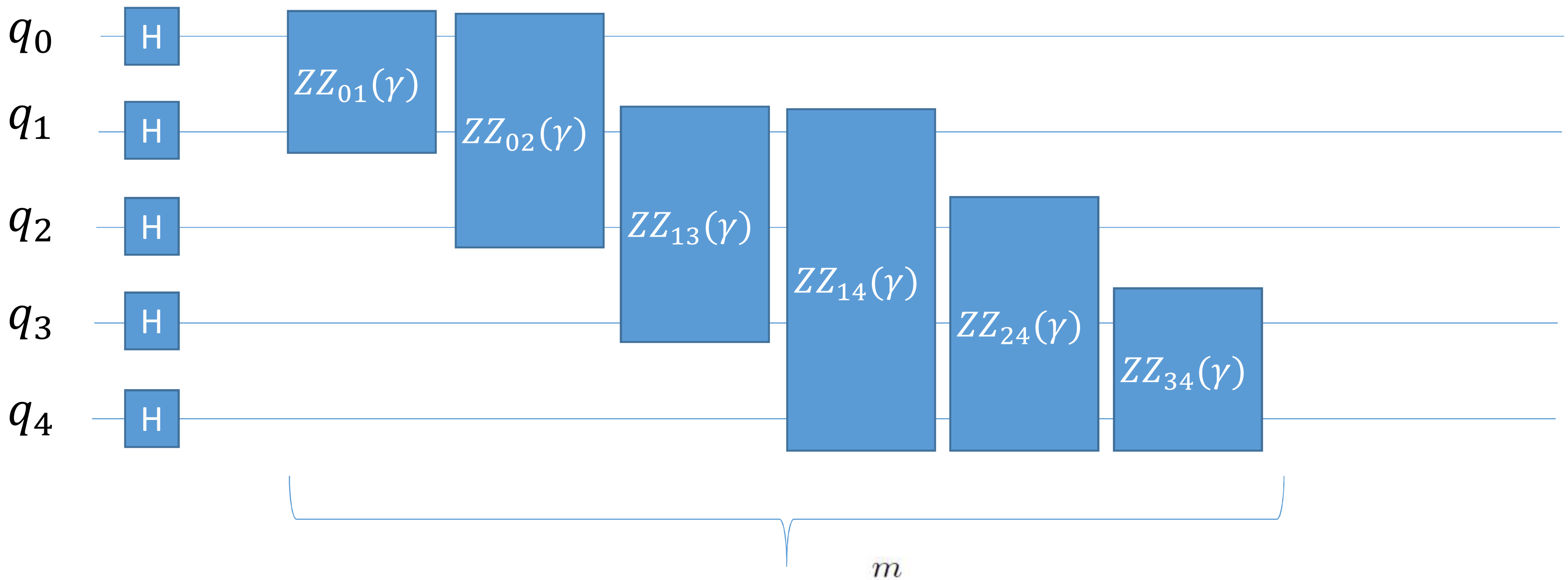
Replace the classical variables with qubits and map the objective function to Hamiltonian:

$$H = -\left(\frac{1-Z_0Z_1}{2} + \frac{1-Z_0Z_2}{2} + \frac{1-Z_1Z_3}{2} + \frac{1-Z_1Z_4}{2} + \frac{1-Z_2Z_4}{2} + \frac{1-Z_3Z_4}{2}\right)$$

$$|s\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle$$

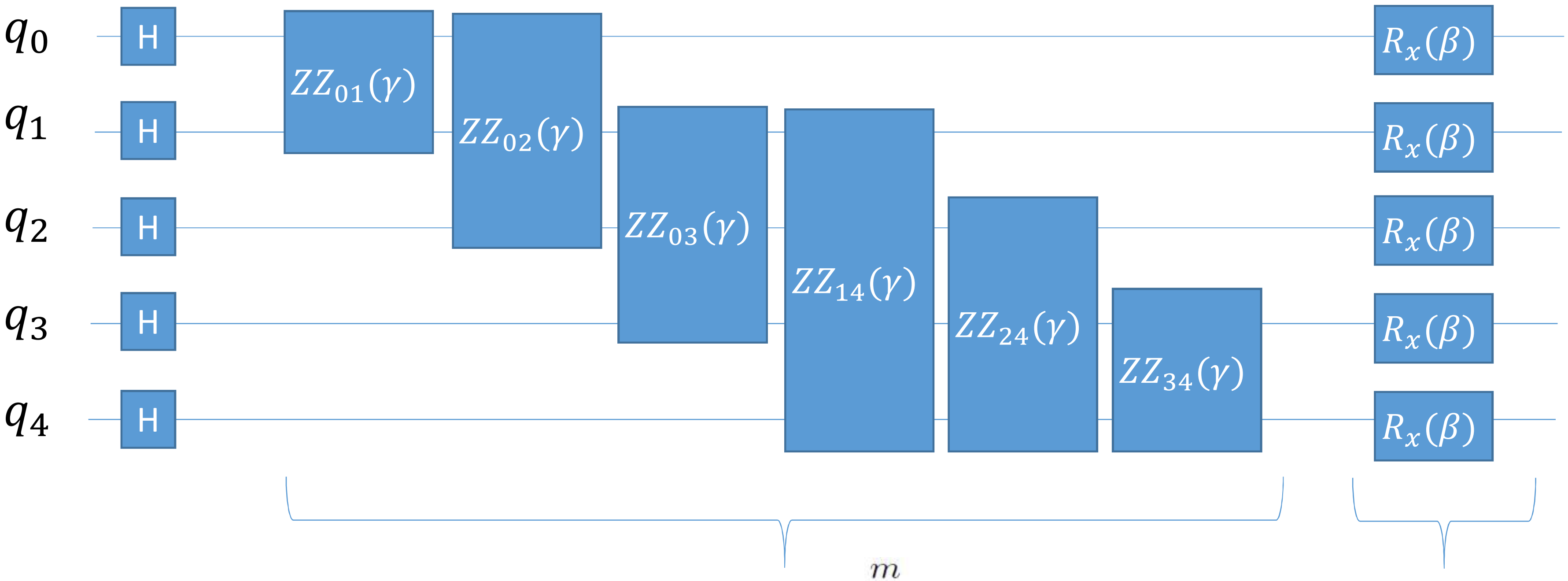


$$H = -\left(\frac{1-Z_0Z_1}{2} + \frac{1-Z_0Z_2}{2} + \frac{1-Z_1Z_3}{2} + \frac{1-Z_1Z_4}{2} + \frac{1-Z_2Z_4}{2} + \frac{1-Z_3Z_4}{2}\right)$$

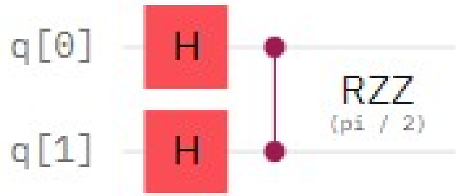


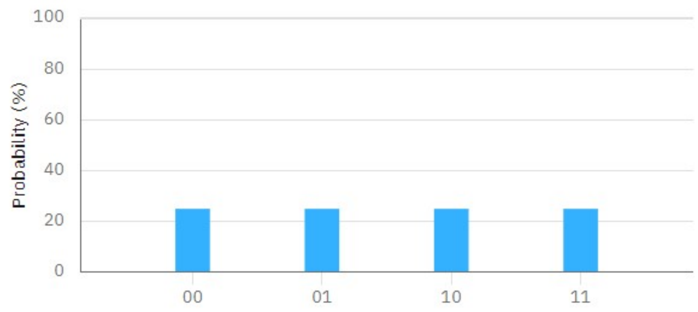
$$U(C,\gamma) = e^{-i\gamma C} = \prod_{\alpha=1}^{m} e^{-i\gamma C_\alpha}$$

$$H = -\left(\frac{1-Z_0Z_1}{2} + \frac{1-Z_0Z_2}{2} + \frac{1-Z_1Z_3}{2} + \frac{1-Z_1Z_4}{2} + \frac{1-Z_2Z_4}{2} + \frac{1-Z_3Z_4}{2}\right)$$

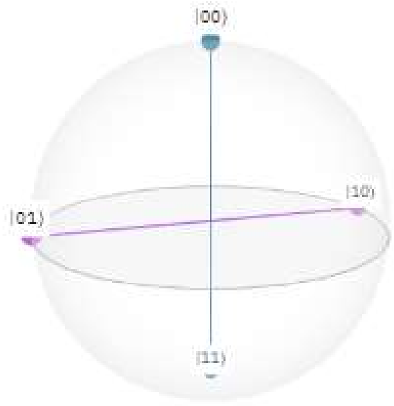


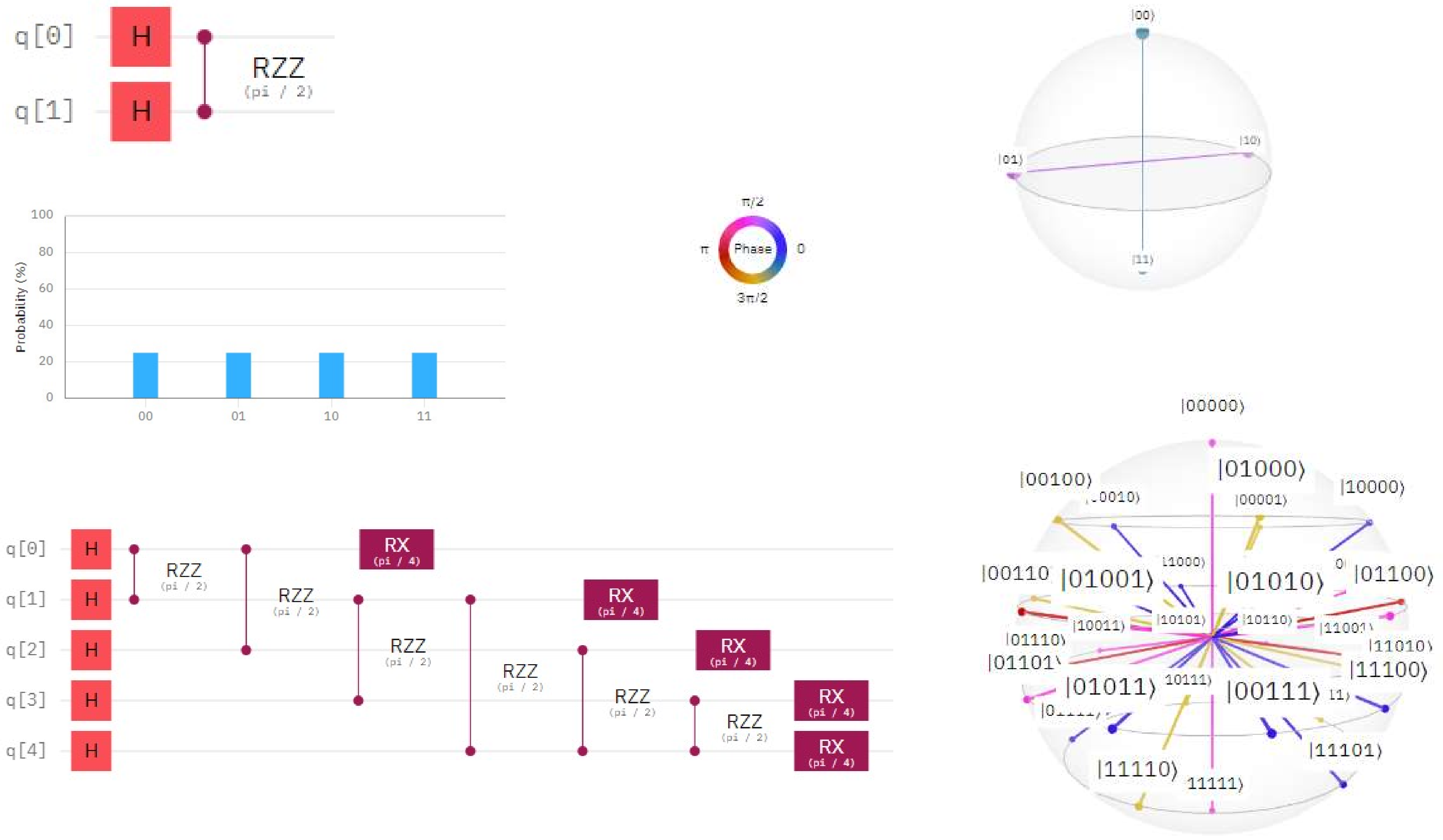
$$U(C,\gamma) = e^{-i\gamma C} = \prod_{\alpha=1}^{m} e^{-i\gamma C_\alpha} \qquad U(B,\beta) = e^{-i\beta B} = \prod_{j=1}^{n} e^{-i\beta\sigma_j^x}$$


q[0]
q[1]
H
H
RZZ
(pi / 2)

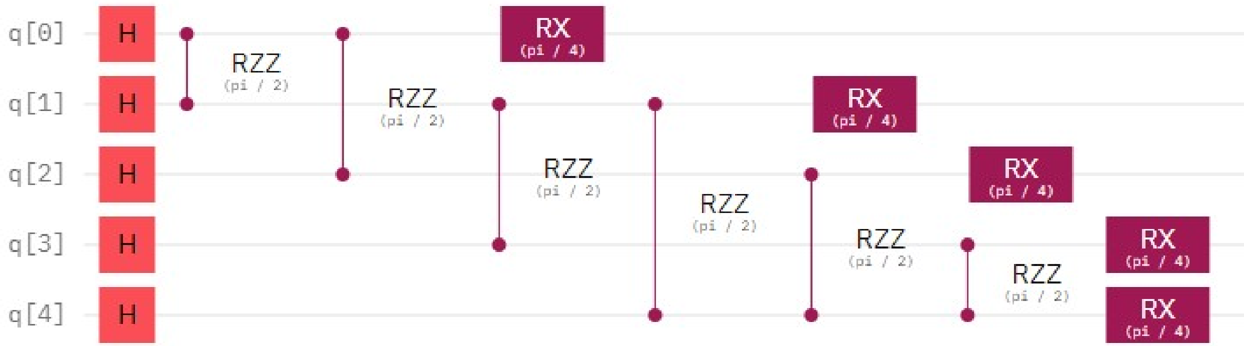

100
80
60
40
20
0
Probability (%)
00
01
10
11

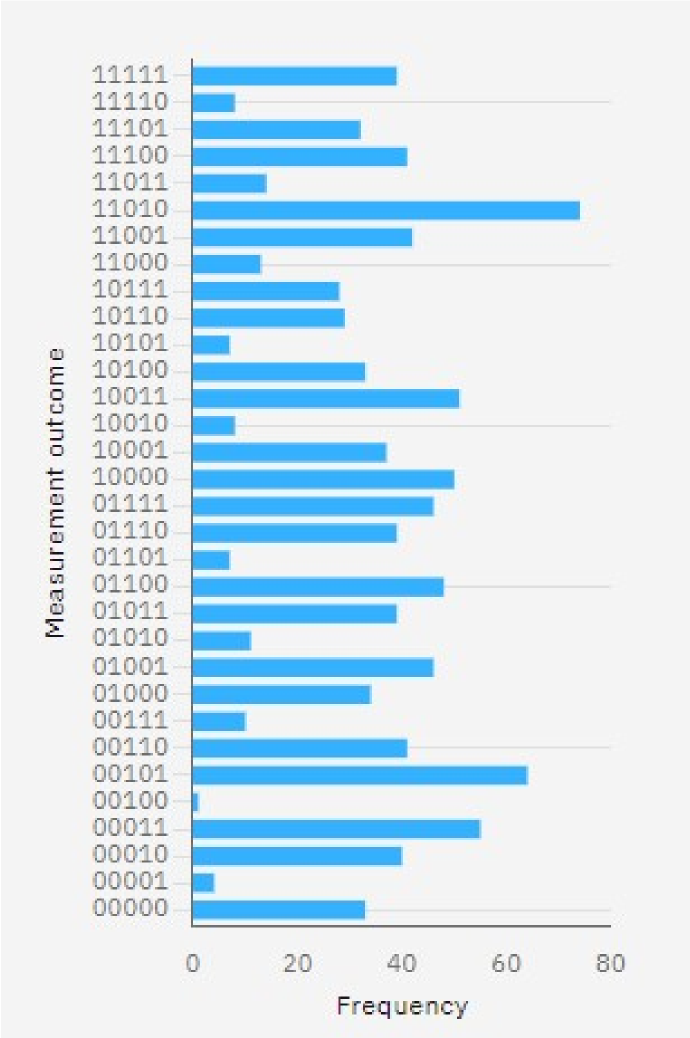

π/2
π
Phase
0
3π/2

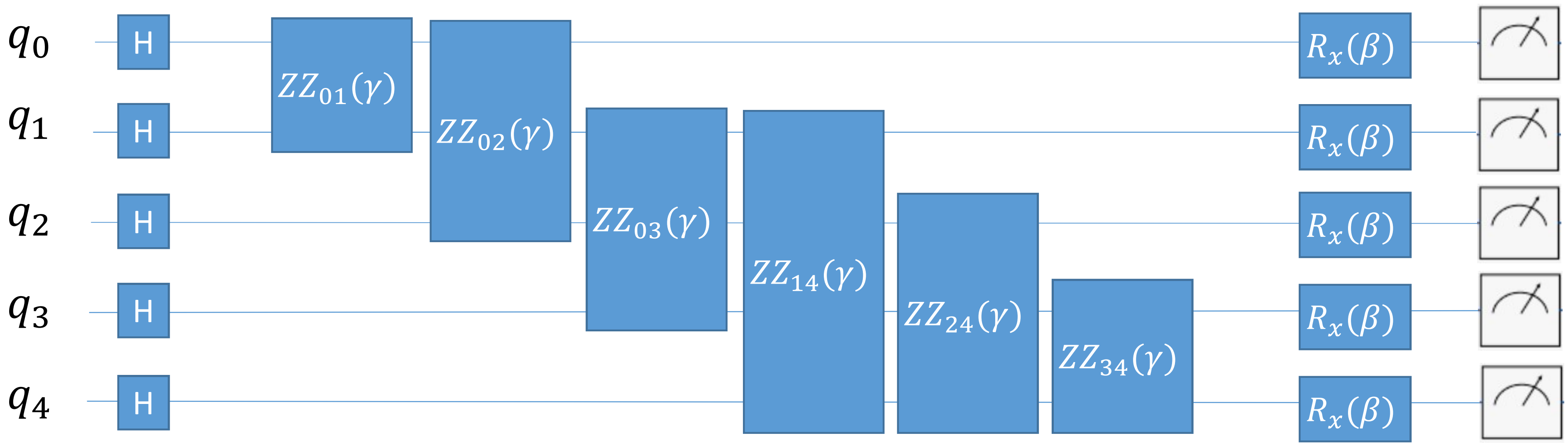

|00⟩
|01⟩
|10⟩
|11⟩



q[0]
q[1]
H
H
RZZ
(pi / 2)
100
80
60
40
20
0
Probability (%)
00
01
10
11
π/2
π
Phase
0
3π/2
|00⟩
|01⟩
|10⟩
|11⟩
q[0]
q[1]
q[2]
q[3]
q[4]
RZZ
(pi / 2)
RX
(pi / 4)
|00000⟩
|00100⟩
|01000⟩
|10000⟩
|00001⟩
|00110
|01001⟩
|01010⟩
|01100⟩
|10011⟩
|10101⟩
|10110⟩
|01110⟩
|01101⟩
|11010⟩
|11100⟩
|01011⟩
|10111⟩
|00111⟩
|11101⟩
|11110⟩
|11111⟩


q[0]
q[1]
q[2]
q[3]
q[4]
H
RZZ
(pi / 2)
RX
(pi / 4)


Measurement outcome
Frequency
11111
11110
11101
11100
11011
11010
11001
11000
10111
10110
10101
10100
10011
10010
10001
10000
01111
01110
01101
01100
01011
01010
01001
01000
00111
00110
00101
00100
00011
00010
00001
00000
0
20
40
60
80



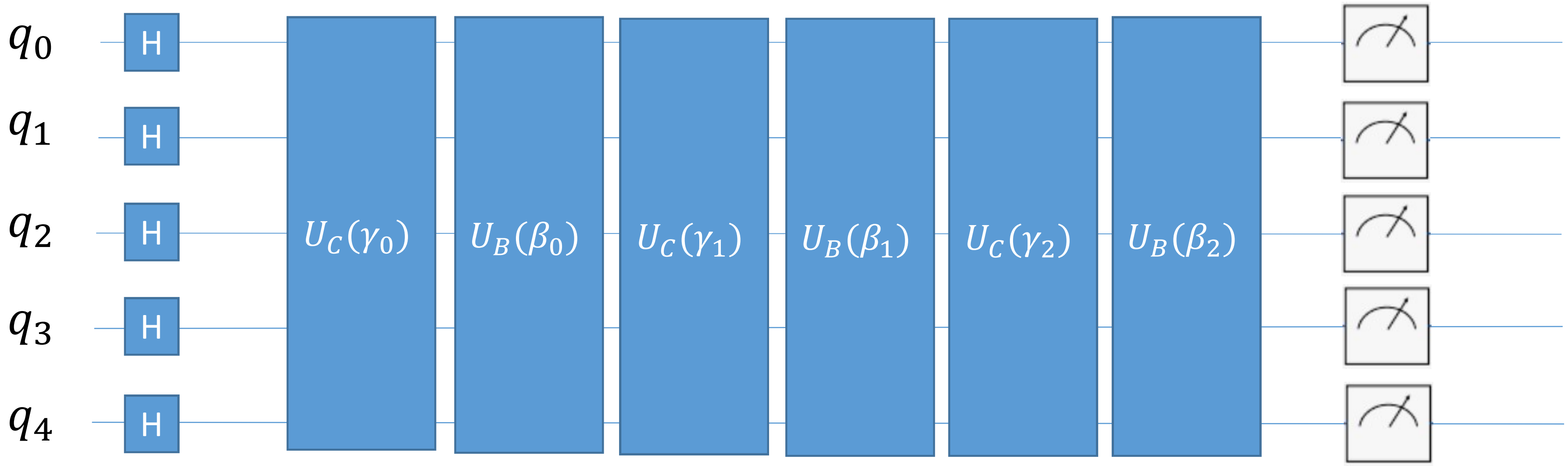
Measure with enough repetitions to compute the expectation value

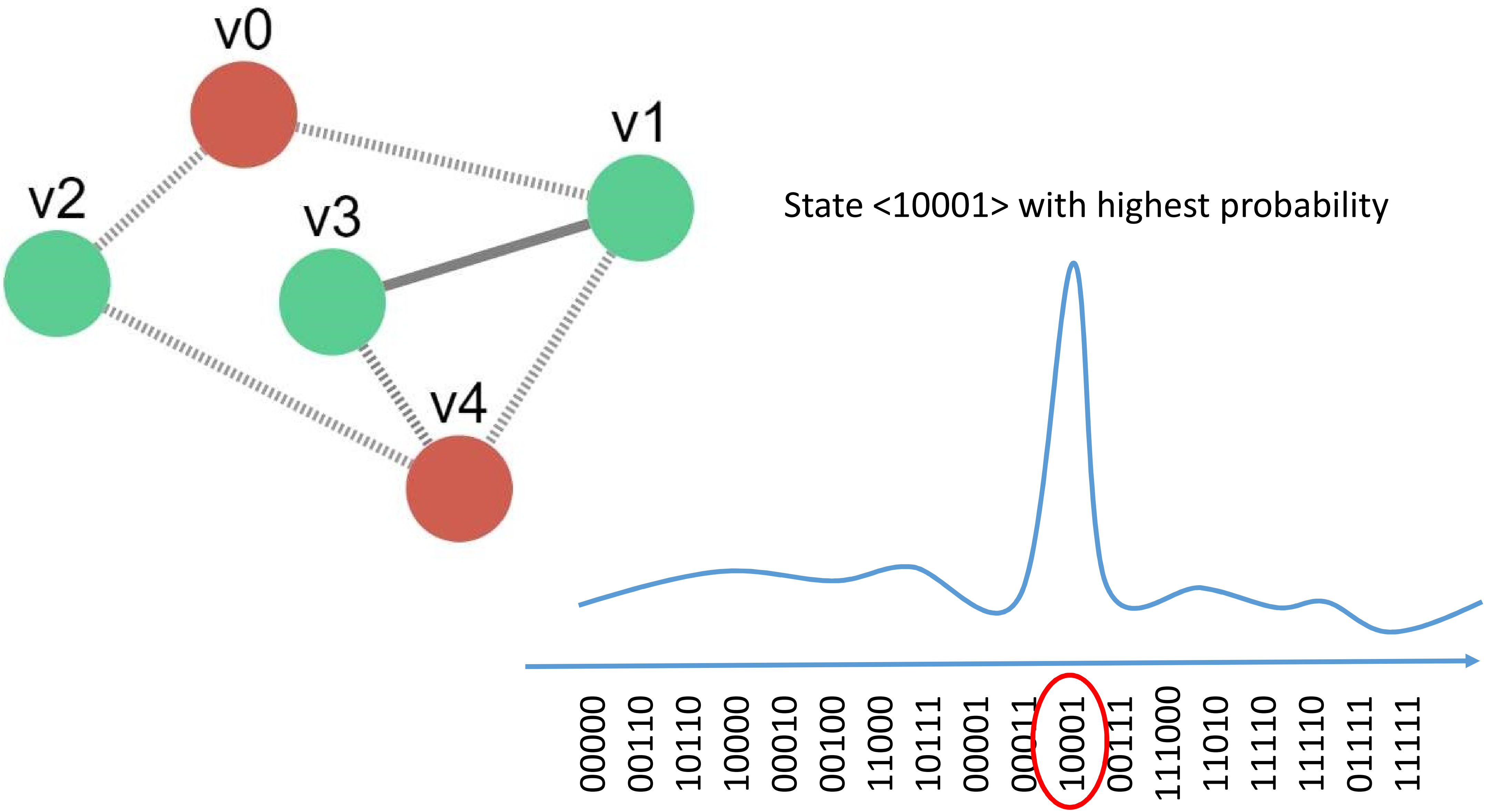
$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | C | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle$$

Use classical optimisation to find better parameters

$$M_p = \max_{\boldsymbol{\gamma}, \boldsymbol{\beta}} F_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$$

Accuracy can be improved by increasing p thus by having more $U_C(\gamma)$ and $U_B(\beta)$ pairs. However, this increases the circuit depth and the complexity of the classical optimisation as the number of parameters increases.




v0
v1
v2
v3
v4
State <10001> with highest probability
00000
00110
10110
10000
00010
00100
11000
10111
00001
00011
10001
00111
111000
11010
11110
11110
01111
11111

Conclusions

- Quantum Approximate Optimization Algorithm (QAOA) is a variational algorithm for combinatorial optimisation
- Suitable for noisy intermediate-scale quantum (NISQ) era devices
- Is it better than classical? .. To find out we'll have to wait for better devices and test.