5.4 Quantum Annealing

The quantum annealing is an heuristic quantum algorithm based on the adiabatic theorem. It aims at solving hard combinatorial optimisation problems using the Ising Hamiltonian as a target Hamiltonian.

The algorithm exploits an Hamiltonian transformation of the form

$$\hat{H}(t) = (1 - s(t))\,\hat{H}_0 + s(t)\hat{H}_{\mathrm{C}}, \tag{5.26}$$

where $s(t)$ is a suitable smooth function of time with $s(0) = 0$ and $s(\tau) = 1$, and $\hat{H}_{\mathrm{C}}$ is the Ising Hamiltonian in Eq. (5.1) whose ground state encodes the solution of the problem. Moreover, one requires that H_0 has a non-degenerate ground state that is easy to prepare and that $[\hat{H}_0, \hat{H}_1] \neq 0$. A simple choice is

$$\hat{H}_0 = -\sum_{i=1}^{n} \hat{\sigma}_x^{(i)}, \tag{5.27}$$

which has $|+\rangle^{\otimes n}$ as ground state. This can be easily prepared via Hadamard gates: $|+\rangle^{\otimes n} = \hat{H}^{\otimes n}\,|0\rangle^{\otimes n}$.

If the quantum annealing time τ is not sufficiently long (i.e. the transformation is not sufficiently adiabatic), which is essentially always the case, then one reach a state $|\psi(\tau)\rangle$, which has a probability p of being the solution of the problem. Such a probability (of success) is given by $p = |\,\langle z_{\mathrm{sol}}|\psi(\tau)\rangle\,|^2$, where $|z_{\mathrm{sol}}\rangle$ is the state encoding the exact solution. To obtain the solution with a 99% certainty, one has to repeat the annealing procedure m times. Indeed,

$$P_{\mathrm{succ}}^{m} = 1 - (1 - p)^m = 0.99. \tag{5.28}$$

The corresponding total time required is given by

$$T_{99\%} = m\tau = \frac{\ln(1 - 0.99)}{\ln(1 - p)}\tau \tag{5.29}$$

A strong challenge for the quantum annealing is the full connectivity of the qubits. Indeed, in a quantum computer, the qubits interactions, which are parameterised by J_{ij}, are typically null beyond nearest-neighbour sites. This strongly limits the scaling of universally annealing where one can suitably tune all the values of J_{ij}.

5.5 Quantum Approximate Optimisation Algorithm (QAOA)

The Quantum Approximate Optimisation Algorithm (QAOA) is a hybrid quantum-classical algorithm that allows for optimising a cost function and finding an approximated solution. It is an application of the adiabatic theorem, similarly as the quantum annealing, which is run on a quantum computer, while a classical computer optimises the cost function.

We start from a quantum annealing Hamiltonian of the form

$$\hat{H}(t) = (1 - s(t))\hat{H}_{\mathrm{M}} + s(t)\hat{H}_{\mathrm{C}}, \tag{5.30}$$

where $s(t)$ is an arbitrary function such that $s(0) = 0$ and $s(\tau) = 1$ with τ being the total time of the algorithm. The initial Hamiltonian $\hat{H}_{\mathrm{M}}$ is such that its ground state can be prepared easily. $\hat{H}_{\mathrm{C}}$ is instead the cost Hamiltonian whose ground state encodes the solution to the problem. The QAOA is based on the observation that the best way to implement the annealing Hamiltonian in Eq. (5.30) is a Trotter procedure. Namely, this is to consider the unitary evolution with respect to $\hat{H}(t)$ and decompose it in small time steps. Then, we have

$$\hat{U}(\tau) = \mathrm{T}\exp\left[-\tfrac{i}{\hbar}\int_0^{\tau} \mathrm{d}t\,\hat{H}(t)\right] \simeq \prod_{k=1}^{p} \exp\left[-\tfrac{i}{\hbar}\hat{H}(k\Delta t)\Delta t\right], \tag{5.31}$$

where T indicates the time ordering, one assumes a large number of steps $p \gg 1$ of length $\Delta = \tau/p$. Owing that for $[\hat{A}, \hat{B}] \neq 0$ one has

$$e^{i(\hat{A}+\hat{B})\Delta t} = e^{i\hat{A}\Delta t} e^{i\hat{B}\Delta t} + \mathcal{O}((\Delta t)^2), \tag{5.32}$$

and since we require that $[\hat{H}_{\rm C}, \hat{H}_{\rm M}] \neq 0$, one has that at each time step the following approximation is valid to the order $(\Delta t)^2$:

$$\hat{U}(\tau) \simeq \prod_{k=1}^{p} \exp\left[-\tfrac{i}{\hbar}\left(1 - s(k\Delta t)\right)\hat{H}_{\rm M}\Delta t\right] \exp\left[-\tfrac{i}{\hbar} s(k\Delta t)\hat{H}_{\rm C}\Delta t\right]. \tag{5.33}$$

Now, the key idea of QAOA is to redefine the time dependence in the following way:

$$\tfrac{1}{\hbar}\left(1 - s(k\Delta t)\right)\Delta t \to \beta_k, \quad \text{and} \quad \tfrac{1}{\hbar} s(k\Delta t)\Delta t \to \gamma_k. \tag{5.34}$$

Thus, we have

$$\hat{U}(\tau) \simeq \prod_{k=1}^{p} \exp\left[-i\beta_k \hat{H}_{\rm M}\right] \exp\left[-i\gamma_k \hat{H}_{\rm C}\right], \tag{5.35}$$

where the parameters $\beta = (\beta_1, \ldots, \beta_p)$ and $\gamma = (\gamma_1, \ldots, \gamma_p)$ become the variational parameters to be optimised. Crucial difference with respect to the quantum annealing case is that one optimises over a set of $2p$ parameters instead of a fixed time segments. Finally, one constructs the variational state

$$|\gamma, \beta\rangle = \prod_{k=1}^{p} e^{-i\beta_k \hat{H}_{\rm M}} e^{-i\gamma_k \hat{H}_{\rm C}} |\text{init}\rangle, \tag{5.36}$$

where the initial state $|\text{init}\rangle$ is the ground state of $\hat{H}_{\rm M}$. In the case of $\hat{H}_{\rm M}$ being equal to Eq. (5.27), one has

$$|\text{init}\rangle = \hat{H}^{\otimes n} |0\rangle^{\otimes n}. \tag{5.37}$$

In the computational basis, the variational state reads

$$|\gamma, \beta\rangle = \sum_{z=0}^{2^n-1} d_z(\gamma, \beta) |z\rangle, \tag{5.38}$$

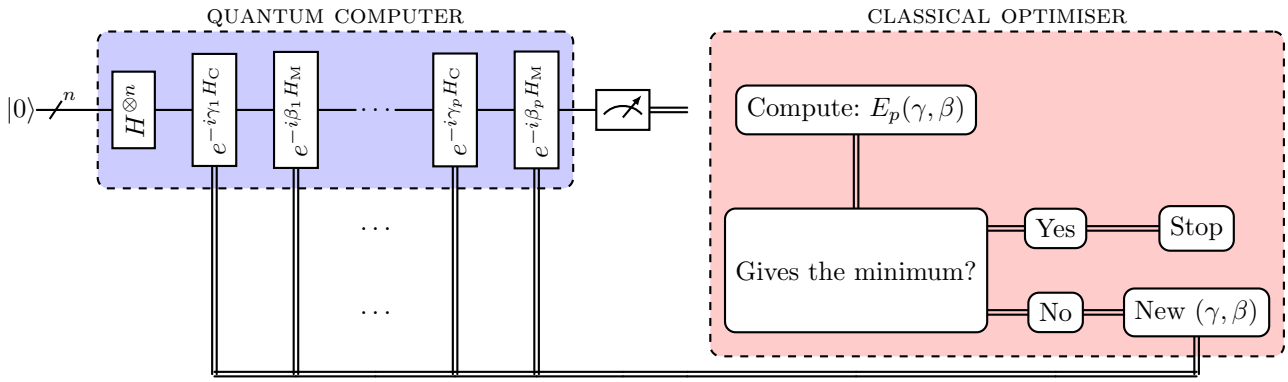
where $d_z(\gamma, \beta)$ defines the superposition in the Z basis. Notably, since $\hat{H}_{\rm C}$ encodes in its ground state the solution of the problem, one needs to minimise the expectation value of $\hat{H}_{\rm C}$ computed on the variational state. Namely

$$E_p(\gamma, \beta) = \langle \gamma, \beta | \hat{H}_{\rm C} | \gamma, \beta \rangle = \sum_{z=0}^{2^n-1} P_z(\gamma, \beta) C(z), \tag{5.39}$$

where $P_z(\gamma, \beta) = |d_z(\gamma, \beta)|^2$ is the probability of having the $|z\rangle$ state and $C(z) = \langle z | \hat{H}_{\rm C} | z \rangle$ is the corresponding cost. The best (γ, β) are such that

$$(\gamma^*, \beta^*) = \arg\min_{\gamma,\beta} E_p(\gamma, \beta). \tag{5.40}$$

Such an optimisation is performed on classical computer (classical optimiser). The circuit representation of the QAOA is



(5.41)

Exercise 5.2
Derive the explicit expression of the cost fucntion $C(z)$ in terms of the coefficients J_{ij} and h_i.

The single k step of the QAOA, when considering $\hat{H}_M$ as in Eq. (5.27) and $\hat{H}_C$ being the Ising Hamiltonian, is implemented as the following. First we consider $\hat{H}_M$,

$$\hat{H}_M = -\sum_{i=1}^{n} \hat{\sigma}_x^{(i)}. \tag{5.42}$$

Then, the corresponding unitary acts independently on each qubit

$$e^{-i\beta_k \hat{H}_M} = e^{i\beta_k \sum_{i=1}^{n} \hat{\sigma}_x^{(i)}} = \prod_{i=1}^{n} e^{i\beta_k \hat{\sigma}_x^{(i)}}. \tag{5.43}$$

Then, the corresponding action can be implemented with a rotation on the single i-th qubit. The circuit implementing it is

$$\boxed{R_X(-2\beta_k)} \tag{5.44}$$

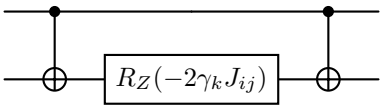
Indeed a rotation around $\mathbf{n}$ by an angle θ is defined a $\hat{R}^{\mathbf{n}}(\theta) = e^{-i\theta \mathbf{n}\cdot\hat{\boldsymbol{\sigma}}/2}$. The implementation of the unitary related to $\hat{H}_C$ can be divided in two steps, indeed the two terms of $\hat{H}_C$ in Eq. (5.1) commute. Then, one writes

$$e^{-i\gamma_k \hat{H}_C} = e^{i\gamma_k \left(\sum_{i=1}^{n} h_i \hat{\sigma}_z^{(i)} + \sum_{1\le i<j\le n} J_{ij} \hat{\sigma}_z^{(i)} \hat{\sigma}_z^{(j)}\right)} = \prod_{1\le i<j\le n} e^{i\gamma_k J_{ij} \hat{\sigma}_z^{(i)} \hat{\sigma}_z^{(j)}} \prod_{i=1}^{n} e^{i\gamma_k h_i \hat{\sigma}_z^{(i)}}. \tag{5.45}$$

Here, the single qubits factors act as rotations, with a circuit being

$$\boxed{R_Z(-2\gamma_k h_i)} \tag{5.46}$$

On the other hand, the two qubits interactions are 2-local gates, which can be implemented via a rotation between two CNOT gates. Namely, the corresponding circuit will read



(5.47)

thus becoming very easy to be implemented.

5.6 Variational Quantum Eigensolver (VQE)

Similarly as the QAOA, the Variational Quantum Eigeinsolver (VQE) is an heuristic approach to solve combinatorial optimisation problem that exploits a combination of quantum computation and classical optimisation. In particular, the QAOA can be seen as a specific implementation of the VQE algorithm.

The algorithm is designed to solve problems that can be stated as finding the ground state energy E_0 of n qubit Hamiltonian. Namely, to find the configuration corresponding to the state $|\Psi_0\rangle$ that

$$\hat{H}\,|\Psi_0\rangle = E_0\,|\Psi_0\rangle\,. \tag{5.48}$$

The generality with respect to QAOA comes in the form of the cost Hamiltonian $\hat{H}_{\mathrm{C}}$. Indeed, one assumes for it the most general form, which is

$$\hat{H}_{\mathrm{C}} = \sum_{\alpha} h_\alpha \hat{P}_\alpha = \sum_{\alpha} h_\alpha \bigotimes_{j=1}^{n} \hat{\sigma}_{\alpha_j}^{(j)}, \tag{5.49}$$

where h_α are coefficients and the $\hat{P}_\alpha$ are called Pauli strings. The latter are product of n single-qubit Pauli matrices (including the identity). Thus, compared to QAOA (which exploits the Ising model), this Hamiltonian is not limited to two qubit interactions only, but can consider n qubit interactions. This is particularly relevant when considering more complex systems where the Ising model fails to describe the entire complexity of the problem.

Then, the steps of VQE are the following:

1. Map the problem in a cost Hamiltonian $\hat{H}_{\mathrm{C}}$ so that the solution is embedded in its ground state.
2. Prepare the initial state as the ground state of $\hat{H}$.
3. Generate the trial state $|\Psi(\theta)\rangle$, which is determined by a set of parameters θ.
4. Measure the expectation values of the Pauli strings in the Hamiltonian, i.e. $\langle\Psi(\theta)|\hat{P}_\alpha|\Psi(\theta)\rangle$. This is the end of the computation on the quantum computer
5. Compute the corresponding energy, i.e. $E(\theta) = \sum_\alpha h_\alpha \langle\Psi(\theta)|\hat{P}_\alpha|\Psi(\theta)\rangle$
6. Update or accept the values of θ based on the result.
7. If updated, one goes back to point 2.

Notably, when searching for the ground state energy of the cost Hamiltonian, there are several pitfalls that the update step must deal with. For example, the parameter landscape may have local minima where one does not want to remain stacked.