Direct implementation of gate operations between non-interacting quantum bits

Rudrayya Chowdary Garigipati
Faculty: Preethika Kumar

Department of Electrical Engineering and Computer Science, College of Engineering

1. Introduction

Most physical realizations of a quantum computer are based on a linear nearest neighbor (LNN) arrangement of qubits where qubits are arranged along an array, and *only* the adjacent qubits interact with each other. To perform an operation between two non-NN (remote) qubits, additional gates are required to bring the qubits adjacent to each other, which can increase the computational overhead. Here, we propose a new scheme using a learning algorithm as a tool, to find the system parameters that perform Controlled-Not (CNOT) gate efficiently between remote qubits in LNN architectures which is an elementary operation in quantum systems.

2. Training method to find the system parameters to implement gate operations

We will now introduce the training process, which will be used in this paper to find the system parameters for realizing the desired gate operations. Following are the steps that will be followed during the training process:
Step 1: Choose a Hamiltonian for an N-qubit system. In this paper, to demonstrate our scheme, we choose a 3-qubit Hamiltonian, H_3:

$$H_3 = \sum_{i=1}^{3}(\Delta_i \sigma_{Xi} + \varepsilon_i \sigma_{Zi}) + \sum_{i=1}^{2} \xi_{i,i+1} \sigma_{Zi} \sigma_{Zi+1} \qquad (1)$$

Here, the terms Δ_i, and ε_i for i =1, 2 and 3, are the tunneling and bias parameters, respectively, and $\xi_{i,i+1}$ is the coupling parameter between qubits Q_i and Q_{i+1}. Also, σ_{Xi} and σ_{Zi} for i =1, 2, and 3, are the Pauli matrices corresponding to qubit Q_i. Here, we use Ising type interactions which are typical of Josephson-junction qubits.

Step 2: Choose a learning rule that will be used to train the quantum system. In this paper, a dynamic learning algorithm based on back propagation technique similar to the one proposed by Behrman *et al.* [1] is used. While in [1], measurement operators were used for training, we use the quantum fidelity condition. This makes our scheme more general. Therefore, the error, E_p, is calculated as:

$$E_p = 1 - F(\rho_{des}, \rho_{out}) = Tr\left(\sqrt{\sqrt{\rho_{des}}\,\rho_{out}\sqrt{\rho_{des}}}\right) \qquad (2)$$

Here, $F(\rho_{des}, \rho_{out})$ is the quantum fidelity, which gives a measure of closeness of two density matrices and varies between 0 and 1 (F = 1 when matrices are identical); ρ_{des} and ρ_{out} are described in Steps 3 and 4.
Step 3: Form training pairs for a specific gate operation we want to implement (**U**). A training pair will comprise of an initial state, $|\psi\rangle_{in}$, and the corresponding output state, $|\psi\rangle_{des}$, that results when applying the gate operation, i.e., $|\psi\rangle_{des} = \mathbf{U}|\psi\rangle_{in}$. The corresponding density matrices for the two states that will be used in training are $\rho_{in} = |\psi\rangle_{in}\langle\psi|_{in}$ and $\rho_{des} = |\psi\rangle_{des}\langle\psi|_{des} = \mathbf{U}|\psi\rangle_{in}\langle\psi|_{in}\mathbf{U}^{-1}$. A minimum of 2^N vector pairs will be used for an N-qubit system, corresponding to all the basis vectors.
Step 4: Train the network for '*m*' iterations to find parameters of the system Hamiltonian that will implement **U**, within a time duration, t_f (time within which we want to implement **U**). The duration of each iteration is equal to t_f. At the start of training, i.e., the first iteration, we randomly assign values to the parameters. Next, the following procedure is followed for each iteration:
(i) Input one of the 2^N input states as the initial state, $\rho_{in} = |\psi\rangle_{in}\langle\psi|_{in}$, corresponding to a training pair, to the network. Allow the state to evolve under the Schrodinger equation for a time t_f. The density matrix for the "actual" output state, ρ_{out}, by propagation of the input state through the network is calculated.

(ii) Calculate $F(\rho_{des}, \rho_{out})$, and the error E_p for the training pair, E_p (Eq. (2)).
(iii) Back-propagate the error (in time) through the network, and adjust the parameters. This can be done by integrating the Schrödinger equation from the final time, t_f, to 0 with the help of a change of variable, $t' = t_f - t$. The network can be set up such that different parameters are adjusted at different "learning" rates.
(iv) Repeat steps (i), (ii), and (iii) until all training pairs are exhausted for the iteration.
At the end of each iteration, the root mean square (RMS) error will be calculated. The training is stopped (after *m* iterations) when the RMS error falls below a certain threshold. A successfully trained network will have RMS error below a certain threshold. When training, as a first step, we will assume all the parameters of the system Hamiltonian to be variables. The network will then be trained to find the system parameters that realize **U** within time t_f.

3. Training Results: CNOT gate operation between next-to-nearest neighbor qubits

CNOT is a two qubit gate operation where one qubit acts as a control and the other acts as a target. Under the CNOT gate operation, the target qubit flips its state when the control qubit is in the $|1\rangle$ state. When the control qubit is in the $|0\rangle$ state, the target does not change its state. Suppose we want to implement a CNOT gate between two next-to-nearest neighbor qubits Q_1 and Q_3 without requiring the qubits to be adjacent to each other. Here, Q_3 is the target qubit. The CNOT gate operation between qubits Q_1 and Q_3 can be described as:

$$|q_1q_2q_3\rangle \to \begin{cases} |q_1q_2q_3\rangle, if\ q_1 = |0\rangle \\ |q_1q_2q_3{}'\rangle, if\ q_1 = |1\rangle \end{cases} \qquad (3)$$

where $|q_i\rangle$ corresponds to state of qubit Q_i, i = 1 to 3, and $|q_3'\rangle$ is the complement of state $|q_3\rangle$ (for instance, if $|q_3\rangle = |1\rangle$, $|q_3'\rangle = |0\rangle$). To implement the gate operation, two additional swap gates are required to bring qubits Q_1 and Q_3 adjacent to each other. If each swap gate is decomposed into 3 CNOT gates, the gate count can increase to 7. In addition to increasing the computational overhead, such multi-gate decompositions can also increase the probability of propagating errors through the circuit. Here, using the dynamic learning algorithm as a tool, we find the system parameters that realize the CNOT gate operation on qubits Q_1 and Q_3 *directly* without requiring swap gates to bring them adjacent to each other. The training set comprises of eight input-output pairs. The parameters of the trained network to realize the desired CNOT gate operation are as follows: Δ_1 = 5 MHz, Δ_2 = 12.6 MHz, Δ_3 = 1.8842 GHz, ε_1 = 1 GHz, ε_2 = 395 MHz, ε_3 = 119.7 MHz, ξ_{12} = 395 and ξ_{23} = 113.1 MHz (t_f = 34.5 ns). The training was stopped when the RMS error was 0.0038. The simulations confirmed the CNOT gate operation with Q_1 as control and Q_3 as target, which was realized up to an overall global phase of 90 degrees that can be ignored. As an example, Fig. 1 shows the probabilities of the qubits A, B, and C in state $|1\rangle$ for the superposition of $|000\rangle$ and $|100\rangle$ states as an input state. After 34.5ns, the input state switches to the superposition of $|000\rangle$ and $|101\rangle$ states, as expected under the CNOT gate operation between qubits Q_1 and Q_3. From the figure, we can see that the final probabilities of the qubits A, B, and C to be in state $|1\rangle$ are 0.5, 0, and 0.5 respectively which confirms the CNOT gate operation between qubits Q_1 and Q_3. In this case, the worst-case fidelity of the gate operation was 99.24% and the overall fidelity was 99.795%.

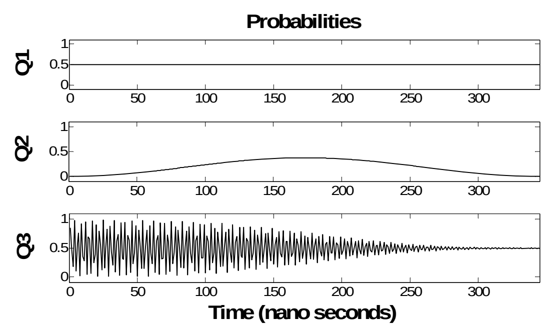

Fig. 1. CNOT gate operation on qubits Q_1, Q_2 and Q_3, with qubit Q_1 as control, and qubit Q_3 as target. The initial state of the system is $(1/\sqrt{2})(|000\rangle+|100\rangle)$. The probability of the qubits Q_1, Q_2 and Q_3 to be in state $|1\rangle$ has been shown, which confirm the CNOT gate operation.

3. Conclusions

In this paper, we introduced a new scheme to implement gate operations *directly*, in a one dimensional linear nearest neighbor array, by using dynamic learning algorithm. We showed how the training algorithm can be used as a tool for finding the parameters for implementing CNOT gates between next-to-nearest neighbor qubits in an Ising-coupled LNN system. The main advantages of our scheme are that, we can reduce the computational overhead of a quantum circuit and all the parameters found using our scheme are scalable, and therefore, can be adjusted to the requirements of a given experimental realization.

[1] Behrman, E.C., Steck, J.E., Kumar, P., Walsh, K.A.: Quantum algorithm design using dynamic learning. Quantum Information and Computation **8**, No. 1&2, 0012–0029 (2008)