Accelerated quantum circuit simulation on reconfigurable devices

Thomas Parks

Abstract

Quantum simulation is hard for classical computers, and with high memory bandwidth requirements has not been efficiently implemented on a reconfigurable architecture. We aim to show that quantum simulations can be carried out at large scale and lower cost over FPGA networks as compared to conventional supercomputers.

Quantum circuit simulation

A large-scale quantum computer with sufficient error correction is capable of efficiently computing problems contained within the BQP complexity class, and so are strictly more powerful than computers based on classical logic. In order to express algorithms for a quantum computer, mathematicians use quantum circuit diagrams.

A quantum circuit describes a sequence of unitary operations to be performed on a register of qubits. All states and operations in the circuit are understood in terms of the computational basis. The computational basis is formed over 2 basis states per qubit, $|0\rangle$ and $|1\rangle$. As the systems are quantum mechanical, a full description of a single qubit is given by $\alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers that, when squared, give the classical probability of observing that qubit in the associated basis state[1].

The full state register for a n qubit circuit is given by the superposition of all 2^n states with the form $|x_1 \dots x_n\rangle$, where $x_1 \dots x_n$ represent every n-bit string. Some states have a properly of separability. A separable state can be formed as a product of individual qubit states, requiring only $2n$ coefficients rather than 2^n. This is only possible for unentangled states with no mutual information. It is conjectured and widely expected to be the case that the relative power of a quantum computer is related to the amount of entanglement as any qubit that does not participate in a non-product state can be simply simulated separately, exponentially reducing the effort required to simulate a quantum computer.

[1]A measurement is performed by forming a bra-ket with the linear operator associated with the quantity of interest. If we wish to measure the basis state a system is in, the operator is the identity.

A quantum circuit has a n-qubit state register and a series of gates that operate on individual qubits. Gates are unitary operations that normally involve up to 3 qubits.

Matrix formalization

A quantum circuit can be transformed into a matrix multiplication. A single gate can be transformed into a unitary operation on the full state vector by expanding using the tensor product. Operations on the same qubit can be combined before taking the product by the mixed-product property, reducing the computation effort needed to compute the full operator.

The resulting matrix will be Hermitian, but the most recent effort to simulate a large system achieved a 49qubit simulation using 0.5 petabytes of memory. It is widely expected that a full simulation significantly larger than 50 qubits will never be possible on a classical computer.[2]

Current largest simulation

The most recent evaluation of methods for quantum circuit simulation is given by [1]. In this work, tensor slicing, simple circuit transformations, and entangling gate deferral are combined to perform a full simulation of a 49-qubit random circuit using the Vulcan supercomputer at Lawrence Livermore. Over 32TB of RAM was needed for this simulation, but only 5TB was needed for the state vector.

Quick thoughts: The tensor representation allows for gate deferral and slicing out non-interacting subsets of the full state, and with memory being much more available on the Von Neumann servers this makes sense. The tensor iteration does expose parallelism within a single step, but it is iterative - the full previous state is required. Possibly by repeated substitution of the iteration, we could find a hyper-volume that systolic programming could explore?

The final state is given by a recursive tensor contraction

$$\forall t = 1, \ldots, d \quad \psi^t_{i_1 \ldots i_n} = \sum_{j_1, \ldots, j_n \in \{0,1\}} \left(\prod_{k \in V^1_t} U^{t,k}_{i_k, j_k} \prod_{(h,k) \in E_t} U^{t,(h,k)}_{i_h i_k, j_h j_k} \psi^{t-1}_{j_1 \cdots j_n} \right)$$

[2]Assuming a computer was created capable of writing the elements of the state vector at the Landauer limit, and the sun's energy output was dedicated for a second to writing this vector, it would be possible to write down a 150 qubit state. Hopefully, by the time humanity becomes a Type II Kardashev civilization we will be able to use a QC to compute the state directly!

Space-efficient simulation

[2] introduces a method that collapses any superposition after every gate. This allows the intermediate state to be represented as a single bitstring in $\mathcal{O}(n)$ space, rather than the $\mathcal{O}(2^n)$ space needed for the superposition of all possible basis vectors that describe an arbitrary state. The algorithm proceeds by selecting a random basis vector after every gate in proportion to the probability of observing that state.

If the circuit expresses a deterministic computation, i.e. the final state is a computational basis state, this method will find that state in one iteration. Otherwise, multiple runs will be required in order to find the full output state probability distribution. Termination can be detected by observing the sum of output value probabilities becoming 1.

Gates that may produce entanglement require further attention. When a 2-qubit gate is encountered, the probability mass for all possible states that can interact with this gate is calculated. This is a recursive calculation that expands the calculation by a 1-bit hamming distance ball in space at every non-trivial level of the circuit prior. If all the probabilities calculated are cached, the algorithm is equivalent to the matrix product version, but there exist a continuum of possible amounts of caching that can optimise utilisation of the computing devices.

A clear optimisation would be to combine groups of gates into larger, 5-10 qubit operations (supergates). This can be done at compile time by tracking possible entanglement as described in [1], and would make more efficient use of FPGA compute resources by trading off computational efficiency for control flow complexity.

A possible avenue for restructuring the recursive calculation step would be to determine the full set of needed gate operations for an output level gate and rewrite this as a nested loop over all prior gates.

Supergates for work distribution

In order to scale, we could place individual supergates on separate FPGA blades. After a collapse sample, the state is in a product of basis states and so can be transmitted using 2n values. Determining the probability and amount of data that will need to be transferred during the recursive step would be needed for good device allocation.

References

[1] E. Pednault *et al.*, “Breaking the 49-qubit barrier in the simulation of quantum circuits,” *arXiv preprint arXiv:1710.05867*, 2017.

[2] M. P. Frank, U. H. Meyer-Baese, I. Chiorescu, L. Oniciuc, and R. A. Van Engelen, “Space-efficient simulation of quantum computers,” in *Proceedings of the 47th annual southeast regional conference*, 2009, p. 83.