Compiling Quantum Programs Using Genetic Algorithms

Rodney Van Meter
Graduate School of Science and Technology,
Keio University
3-14-1 Hiyoushi, Kohoku-ku, Yokohama-shi,
Kanagawa 223-8522, Japan
rdv@tera.ics.keio.ac.jp

Kevin Binkley
Graduate School of Science and Technology,
Keio University
3-14-1 Hiyoushi, Kohoku-ku, Yokohama-shi,
Kanagawa 223-8522, Japan
kbinkley@soft.ics.keio.ac.jp

Quantum computer architectures have special characteristics that complicate the process of compiling efficient programs for them. We are exploring the use of genetic algorithms as one method for the storage assignment (and gate execution location assignment) phase of a compiler back end.

Many proposed quantum computer technologies have the feature that quantum bits, or qubits, are stored in specific, static locations, and the quantum gates that form a circuit, or algorithm, come to where the qubits are in the form of e.g. microwave pulses. Thus, quantum circuit design is more like classical program compilation than circuit design.

On many of these technologies, two-qubit gates (or, if you prefer, two-operand instructions) can only have *neighboring* qubits as operands. When two operands that are not next to each are scheduled to be arguments to an instruction, they must be brought together by *swapping* qubit values (or variables) with their neighbors until the arguments are next to each other, and the algorithmically specified gate can be performed. The swap instructions are pure overhead, to be optimized away whenever possible.

The topology in which the qubits are physically laid out has a big impact on the performance [6]. Many of these technologies, including the all-silicon NMR device [4], permit only a one-dimensional line of qubits. Some, such as the optical lattice [1], are two-dimensional arrays. Oskin *et al*'s proposed architecture based on Kane's technology supports a loose lattice of lines with occasional four-way intersections. The scalable ion trap computer [3] is one of the only proposals in which storage areas and interaction areas are separated; ions carrying qubits are literally shuffled around using magnetic fields.

Other researchers have begun exploring GAs to find the analog microwave pulse sequences to implement specific gates [5], but our interest lies at a slightly higher level, the assignment of application-level variables and gates to locations in the machine.

The assignment problem is more than simply *storage* location assignment; the location of variables changes over the course of the execution of the program, as we shuffle qubits around. Our compilation problem, then, is effective assignment of the instruction *execution* locations. For example, consider a program consisting of one thousand instructions on ten qubits. The assignment problem is not assigning the ten variables to ten locations, which would be a maximum of 10! possibilities. Rather, the problem is assigning each of the thousand instructions to one of the ten locations, a search space of 10^{1000}.

We have begun working on this assignment problem and the associated back-end compilation using a genetic algorithm (GA) [2]. Preliminary results are that a GA run in a few seconds on a PC produces a better layout than hand-compiled programs for a 90-instruction program on 32 qubits.

We suspect that adequate heuristics for this problem exist for the one-dimensional line topology, without the need to introduce stochastic mechanisms such as GAs. However, for the more complex topologies, including the lattice and especially the scalable ion trap, we suspect no adequate heuristics will be easily found.

1. REFERENCES

[1] G. K. Brennen, C. M. Caves, P. S. Jessen, and I. H. Deutsch. Quantum logic gates in optical lattices. *Physical Review Letters*, 82(5):1060–1063, Feb. 1999.

[2] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[3] D. Kielpinski, C. Monroe, and D. J. Wineland. Architecture for a large-scale ion-trap quantum computer. *Nature*, 417:709–711, 2002.

[4] T. D. Ladd, J. R. Goldman, F. Yamaguchi, Y. Yamamoto, E. Abe, and K. M. Itoh. All-silicon quantum computer. *Physical Review Letters*, 89(1), July 2002.

[5] M. J. Rethinam, A. K. Javali, E. C. Behrman, J. E. Steck, and S. R. Skinner. A genetic algorithm for finding pulse sequences for NMR quantum computing. http://arXiv.org/quant-ph/0404170, Apr. 2004.

[6] R. Van Meter. Communications topology and distribution of the quantum Fourier transform. In *Proc. Tenth Symposium on Quantum Information Technology (QIT10)*, pages 19–24, May 2004.