Progamming and Simulation of Quantum Computers

Bernhard Ömer[1] and Ian Glendinning[2]

[1] Institute for Theoretical Physics
Vienna University of Technology, Austria
oemer@tph.tuwien.ac.at
[2] VCPC
European Centre for Parallel Computing at Vienna
Institute for Software Science, University of Vienna, Austria
ian@vcpc.univie.ac.at

Unlike quantum circuits, quantum Turing machines or the algebraic definition of unitary transformations, programming languages allow the complete description of quantum algorithms, including their classical control structure, for arbitrary input sizes and hardware architectures.

The work presented here has investigated how the classical formalism of structured programming can be adapted to the field of quantum computing, based on the machine model of a universal computer with a quantum oracle which allows the application of unitary gates and the measurement of single qubits. In analogy to classical programming languages, the concept of structured quantum programming languages (QPLs) has been developed, and illustrated by the experimental language QCL.

A QPL is called *imperative* if it provides quantum registers *(quantum variables)*, elementary gates and single qubit measurements. A *procedural* QPL additionally supports unitary subroutines and non-classical concepts like the reverse execution of code to derive the adjoint operator, and the unitary "uncomputing" of temporary quantum registers back to their initial states *(scratch space management)*. A procedural QPL is called *structured* it also allows the use of qubits and boolean expression of qubits *(quantum conditions)* in structured flow-control statements *(quantum if-statements)*.

A QCL interpreter for the Linux operating system as well as a numerical simulator for arbitrary quantum oracles are available as free software from

`http://tph.tuwien.ac.at/~oemer/qcl.html`

The simulator takes the form of a C++ library called QC-lib, which simulates quantum computers at an abstract functional level. In order to allow more qubits to be simulated in the same time, or the same number to be simulated in less time, we are implementing a parallel version of the library.[3] Currently, parallel versions of the general single qubit gate and the controlled-NOT gate

[3] This work is partially supported by the Special Research Program SFB F011 "AURORA" of the Austrian Science Fund FWF.

have been implemented, allowing arbitrary quantum computations to be simulated, though not necessarily in the most efficient way possible. As a test case we have implemented Grover's quantum search algorithm in terms of QC-lib, and we have made performance measurements on a cluster of Linux systems, obtaining reasonable speedups of up to 11.2 on 16 processors.