Computer Physics Communications ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

OptQC v1.3: An (updated) optimized parallel quantum compiler

T. Loke, J.B. Wang[*]

School of Physics, The University of Western Australia, Crawley, WA 6009, Australia

ARTICLE INFO

Article history:
Received 25 May 2016
Accepted 26 May 2016
Available online xxxx

Keywords:
Quantum circuit
Quantum compiler
Optimization

ABSTRACT

We present a revised version of the *OptQC* program of Loke et al. (2014) [1]. We have removed the simulated annealing process in favour of a descending random walk. We have also introduced a new method for iteratively generating permutation matrices during the random walk process, providing a reduced total cost for implementing the quantum circuit. Lastly, we have also added a synchronization mechanism between threads, giving quicker convergence to more optimal solutions.

New version program summary

Program title: OptQC v1.3

Catalogue identifier: AEUA_v1_3

Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AEUA_v1_3.html

Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland

Licensing provisions: Standard CPC licence, http://cpc.cs.qub.ac.uk/licence/licence.html

No. of lines in distributed program, including test data, etc.: 240903

No. of bytes in distributed program, including test data, etc.: 632395

Distribution format: tar.gz

Programming language: Fortran, MPI.

Computer: Any computer with Fortran compiler (not gfortran4.9 or earlier) and MPI library.

Operating system: Linux.

Classification: 4.15.

Catalogue identifier of previous version: AEUA_v1_3

Journal reference of previous version: Comput. Phys. Comm. 185(2014)3307

External routines: Intel MKL LAPACK routines and MPI routines.

Does the new version supersede the previous version?: Yes

Nature of problem:
It aims to minimize the number of quantum gates required to implement a given unitary operation.

Solution method:
It utilizes a descending random walk to select permutation matrices P and Q for a given unitary matrix U such that the number of gates in the quantum circuit of $U = Q^T P^T U' P Q$ is minimized, where U' is equivalent to U up to a permutation. The decomposition of a unitary operator is performed by recursively applying the cosine–sine decomposition.

Reasons for new version:
Simulated annealing process was found to give suboptimal results compared to a normal descending random walk. Computation time was also bloated by the necessity of running the CS decomposition thrice (for U', P and P^T) for each iteration of the optimization process.

[*] Corresponding author.
E-mail address: jingbo.wang@uwa.edu.au (J.B. Wang).

http://dx.doi.org/10.1016/j.cpc.2016.05.028
0010-4655/© 2016 Elsevier B.V. All rights reserved.

Summary of revisions:

- Simulated annealing process was replaced by a descending random walk (equivalent to setting the threshold value β to 0), due to poor convergence of the simulated annealing process, and due to the lack of pathological instances of local minima in this particular search space.
- Introduced an iterative method for generating the general permutation matrix P, in which we start with $P = I$ and build up the quantum circuit (and modify P correspondingly) by adding gates onto the existing circuit. This removes the requirement of running the CS decomposition on P and P^T to find the quantum circuit implementation, since it is already known by construction.
- Added a synchronization mechanism between threads (after some prescribed number of iterations) in which the current state of the top 10% processes with the fittest solutions is copied over to the remaining 90%. This works so as to discard the less fit solutions and focuses the searching algorithm in the state space with the fittest solutions.

Additional comments:
The program contains some Fortran2003 features and will not compile with gcc4.9 or earlier.

Running time:
As before, running time increases with the size of the unitary matrix, as well as the prescribed maximum number of iterations for qubit permutation selection and the descending random walk. All simulation results presented in this paper are obtained from running the program on the Fornax supercomputer managed by iVEC@UWA with Intel Xeon X5650 CPUs. A comparison of running times is also given in Table 1.

Table 1
Result summary for the decomposition of various matrices: m is the dimension of the given unitary matrix U, N_0 denotes the number of gates required to implement U, and $N_{\min}$ is the total number of gates obtained after the optimization process (post-reduction).

Matrix *	m *	N_0 *	Before $N_{\min}$	Update CPU (min)	After $N_{\min}$	Update CPU (min)
Random real unitary	8	29	22	0.333	18	0.250
S_8graph	16	34	23	1.083	21	0.467
3rd generation 3-Cayley tree	42	996	300	13.583	216	8.233
Quantum Fourier transform	64	4095	3508	20.95	3144	13.250
Shor's algorithm 8	128	8285	5621	75.700	4085	42.917

References:

[1] T. Loke, J.B. Wang, Y.H. Chen, Comput. Phys. Comm. 185 (2014) 3307.