Multiobjective quantum circuit compilation

Lis Arufe[1], Riccardo Rasconi[2], Angelo Oddi[2], Ramiro Varela[1] and Miguel Ángel González[1]

[1]*Department of Computing, University of Oviedo. Campus of Gijón, 33204, Gijón, Spain*
[2]*Istituto di Scienze e Tecnologie della Cognizione (ISTC-CNR). Via S. Martino della Battaglia, 44, Rome, Italy*

Abstract
The compilation of quantum circuits on a given quantum architecture is one of the critical issues when developing quantum algorithms. The related problem is known as the Quantum Circuit Compilation Problem (QCCP). It is known that the current quantum chip technology (Noisy Intermediate-Scale Quantum) is affected by noise that can be originated by different sources and significantly limits the circuit's usability. Generally, one important source of noise is the onset of *decoherence* during the quantum circuit's execution. Decoherence can be described as the difficulty of a quantum state to hold its superposition in time; therefore, quantum circuits should be characterized by a limited depth (i.e., makespan) in order to terminate their execution before qubits start to decohere. The other important source of noise is due to the presence of quantum binary gates (e.g., CNOTs); it is in fact known that the noise produced by binary gates is an order of magnitude greater than the noise produced by unary gates. Hence, trying to minimize the number of the circuit's binary gates can also positively affect the stability of the circuit. In this work, we analyze how a multiobjective approach can be useful to reduce the circuit's noise thus improving its stability, focusing on the minimization of both the circuit's makespan and the number of *swap* gates that are synthesized as a product of the compilation process, as every *swap* gate is a binary gate composed of a sequence of three CNOT gates. In particular, we tackle instances of the Quantum Approximate Optimization Algorithm (QAOA) for the resolution of the MaxCut problem, and provide some preliminary results combining an existing genetic algorithm with two well-known multiobjective approaches.

Keywords
Quantum circuit compilation, Genetic algorithms, Multiobjective

1. Introduction and problem definition

In the paradigm of gate-based model, a quantum algorithm is expressed in terms of a set of quantum gates that must be executed on a quantum hardware, fulfilling some constraints. This gives rise to the Quantum Circuit Compilation Problem (QCCP). We consider the model presented in [1], characterized by: (i) the class of Quantum Approximate Optimization Algorithm (QAOA) applied to the MaxCut problem [2], and (ii) the specific hardware architectures depicted in Figure 1, which shows four quantum chip designs inspired by Rigetti Computing Inc. [3] with different number of qubits ($N = 4, 8, 21, 40$) distributed in weighted and undirected graphs. We also consider larger quantum chips with $N = 72$ (Google Bristlecone) and $N = 127$ (IBM Eagle).

AIQxQIA2023 Workshop
arufelis@uniovi.es (L. Arufe); mig@uniovi.es (M. : González); riccardo.rasconi@istc.cnr.it (R. Rasconi); angelo.oddi@istc.cnr.it (A. Oddi); ramiro@uniovi.es (R. Varela)

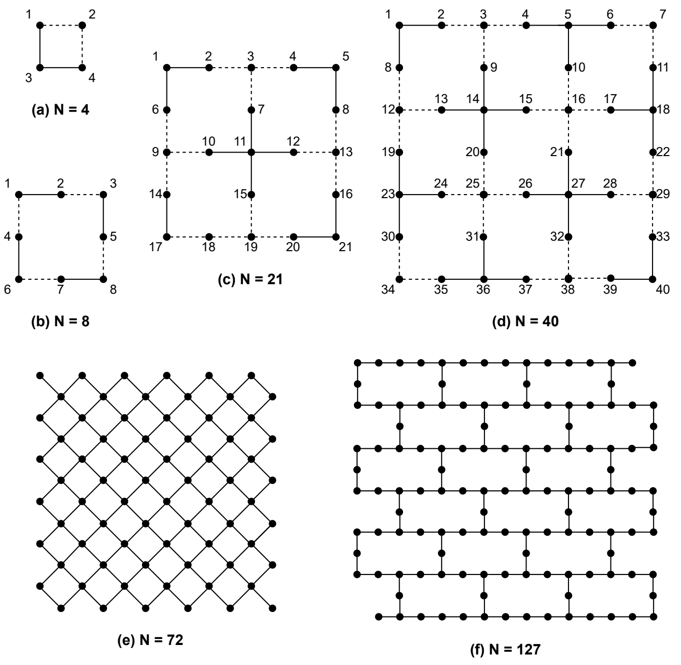


Figure 1: Six quantum chip designs with different number of qubits.

In the QAOA model, the same set of gates must be applied over a number of p rounds to obtain a near optimal solution to the MaxCut problem. Figure 2 shows a MaxCut instance and one solution, considering a single round, compiled on the smallest of the circuits in Figure 1. In this work we assume that each qstate q_i is initially located in qubit n_i. The distribution of quantum gates on qubits is subject to two constraints: (1) no two quantum gates can operate on the same qubit simultaneously, and (2) binary quantum gates can only operate on adjacent qubits. The first constraint requires to organize the gates in a number of steps with the objective of minimizing the circuit depth, while the second constraint often requires introducing additional *swap* gates (which exchange the qstates in two adjacent qubits and are denoted by edges with × symbols in both extremes) to move the quantum states of a binary gate to adjacent qubits. In the example of Figure 2, a *swap* gate was introduced between qubits n_2 and n_4 to ensure that the qstates q_2 and q_3 are adjacent, in qubits n_4 and n_3 respectively, before executing the last binary gate on these qstates.

The study performed in this work is based on the currently available gate-based quantum chip technology, known as Noisy Intermediate-Scale Quantum (NISQ) [4, 5]. NISQ chips are basically quantum processors containing up to about 1000 qubits (https://www.ibm.com/quantum/blog/quantum-roadmap-2033) that are neither sufficiently advanced for fault-tolerance, nor large enough to prove quantum supremacy over classical computing. Moreover, these processors are susceptible to a phenomenon known as *quantum decoherence*, as well as sensitive to their environment (i.e., they are noisy). Decoherence can be described as the difficulty of a quantum state to hold its superposition in time; in other words, for a quantum circuit to return reliable results, the computation should terminate before the pro-

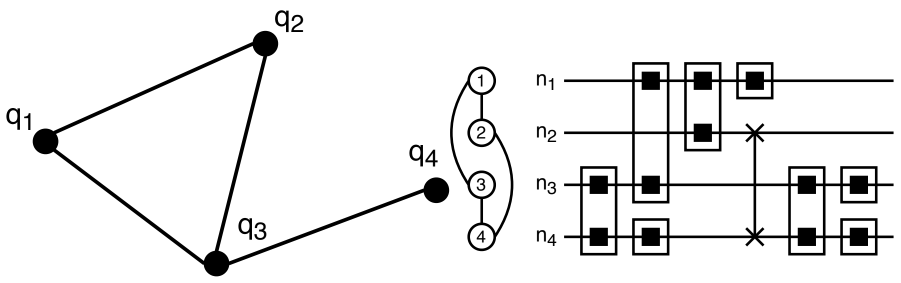


Figure 2: From left to right, a MaxCut instance, a quantum hardware with 4 qubits and a quantum circuit that represents a possible solution of the instance on the quantum hardware.

cessor's qubits start to decohere. Hence, quantum circuits should be characterized by a limited depth (i.e., makespan) in that the shallower the circuit, the sooner the quantum computation ends, and the smaller the decoherence effect suffered by the circuit. Another source of noise is caused by the presence of quantum binary gates (e.g., CNOTs); it is in fact known that the noise produced by binary gates is an order of magnitude greater than the noise produced by unary gates. Hence, trying to minimize the number of the circuit's binary gates also represents an important issue to positively affect the stability of the circuit.

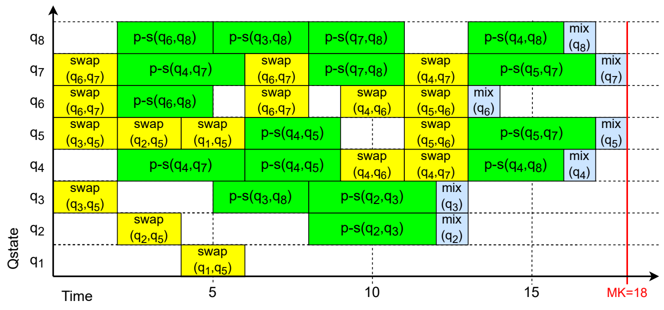
However, minimizing the circuit's depth and minimizing the number of binary gates are generally mutually conflicting objectives [6]; this is also illustrated in Figure 3, where we show two different solutions for an instance of an 8-qubit hardware where we must execute a single round of the following 8 $p-s$ gates: (q_2,q_3), (q_3,q_8), (q_4,q_5), (q_4,q_7), (q_4,q_8), (q_5,q_7), (q_6,q_8), (q_7,q_8). These solutions are shown by means of Gantt charts, representing the operations on each qstate over time. In this work it is assumed that $p - s$ gates take 3 time units on continuous edges and 4 time units on dashed edges, that the *swap* gates require 2 time units on all edges, and the *mix* gates require 1 time unit.

The solution depicted in Figure 3(a) has depth 18 and requires 8 *swap* gates, whereas the solution depicted in Figure 3(b) has depth 25 but only requires 5 *swap* gates. For this same instance it also exists a solution with depth 19 and 6 *swap* gates (not pictured), which can be considered a good compromise between both objectives, hence motivating a multiobjective approach.

2. Solving methods

In this work, we start from the Decomposition Based Genetic Algorithm (DBGA-X) proposed in [7]. In each iteration $r \in \{1, \dots, p\}$ it solves the subproblem defined by rounds $1, \dots, r$, starting from solutions to the subproblem defined by rounds $1, \dots, r-1$ and inserting the $p-s$ and *mix* gates corresponding to round r, also inserting *swap* gates when necessary.

We keep the coding and decoding schemas and genetic operators. In particular, chromosomes for round r are triplets (sg_{r-1}, ch_1, ch_2) where sg_{r-1} is a solution graph for subproblem $1, \dots, r-1$, ch_1 is a permutation of the set of $p-s$ gates of round r, and ch_2 indicates how to insert swaps before inserting the corresponding $p-s$ gate of ch_1. In the decoding step, when


(a) Solution with depth 18 and 8 *swap* gates.

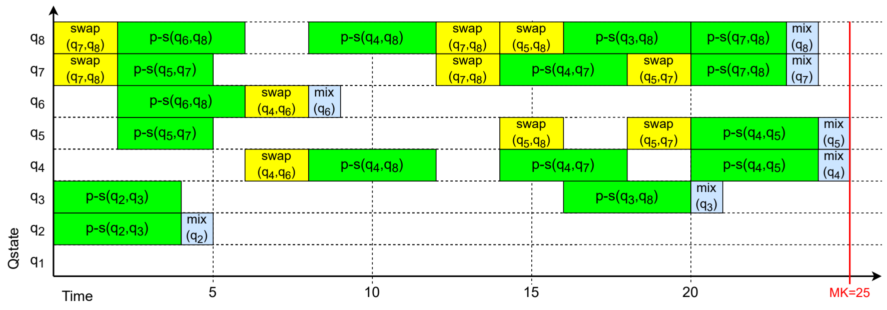

(b) Solution with depth 25 and 5 *swap* gates.

Figure 3: Two example solutions for an instance of an 8-qubit hardware represented with Gantt charts.

inserting a gate $p - s(q_i, q_j)$, a set of *swap* gates are introduced to move concurrently q_j towards q_i and q_i towards q_j through one of the shortest paths on the hardware between the qubits where they are currently located. ch_2 codifies in which edge of that shortest path the $p - s$ gate will be inserted. We also keep using the extension of the partial mapping crossover (PMX) operator and a mutation operator that randomly modifies the ch_2 part of the chromosome. We refer the interested reader to [7] for further details of the method.

In order to extend DBGA-X to deal with two objectives at once, we consider two evolution strategies borrowed from well-known multiobjective approaches:

- The first is the Nondominated Sorting Genetic Algorithm II (NSGA-II) [8], which differs from a standard genetic algorithm mainly in the replacement step, in which we choose individuals based on the concepts of dominance and crowding distance. We enhance the diversity of the method by penalizing solutions with duplicated values of both objective functions before the replacement step.
- The second is the Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D) [9], which divides the whole problem into a set of single-objective problems, each defined by a scalarization function based on different weights. A neighborhood relation between problems is defined so that they can help each other during the evolution.

3. Experimental results

The benchmark instances are taken from [7], and the solving methods are implemented in C++ and run in a Intel Core i5-7400 CPU at 3.00GHz with 16 GB RAM. As for parameter setting, in both multiobjective versions we choose a population size of 1000 chromosomes (so in the case of MOEA/D we solve 1000 single-objective subproblems), a stop condition of 100 generations without adding a new non-dominated solution to the Pareto front, a 100% crossover probability and a 0.05% mutation probability for each position of each chromosome.

Specifically for MOEA/D, we have tried Tchebycheff and weighted sum scalarization functions and the first one seems to perform better experimentally. Additionally, we use a neighborhood size of 100 for each solution (i.e. the 100 solutions from the "closest" single-objective problems). In the crossover step all solutions are mated, choosing as the other parent a solution from the neighborhood with 95% probability, and any solution with 5% probability.

Figure 4 shows some preliminary results of the multiobjective versions of DBGA-X in three instances with 40, 72 and 127 qubits. We show the Pareto fronts obtained by the NSGA-II version and the MOEA/D version. For comparison reasons, we also run single-objective versions of DBGA-X to either minimize makespan (i.e. circuit depth) or to minimize the number of *swap* gates.

Each method is run 10 times, and in the figures we plot the best result from all runs in the case of single-objective, and the Pareto front with the best hypervolume [10] from all runs in the case of multiobjective. The average running time of the multiobjective methods is reasonable, of about 30 seconds for instances with 40 qubits, 2 minutes for 72 qubits and 8 minutes for 127 qubits.

It can be seen that both multiobjective versions are able to reach approximate Pareto sets with a variety of solutions that obtain a good compromise between both objectives. In these instances, the Pareto fronts obtained by the NSGA-II version clearly outperform those of the MOEA/D version, particularly on the larger instances. It is also interesting that neither NSGA-II or MOEA/D are able to reach the best single-objective solutions, although this was the expected behavior, as multiobjective approaches are efficient at obtaining good compromise solutions, but usually less efficient at the extremes of the Pareto front.

4. Conclusions and future work

This paper is a first step in expanding our previous research in the QCCP by considering the minimization of several objectives at once, in particular the circuit depth and the number of added *swap* gates. Minimizing both objectives is useful to deal with the decoherence problem, however we illustrated its conflicting nature and so motivated a multiobjective approach. We combined a previous genetic algorithm from the literature with two well-known multiobjective approaches and we have seen that a NSGA-II strategy obtains a good variety of solutions, and better Pareto fronts than a MOEA/D strategy.

For future work, we first plan to perform an exhaustive experimental analysis using many instances of several sizes and also considering additional rounds of QAOA (as the experiments shown in this paper consider a single round), in order to obtain proper conclusions. However,

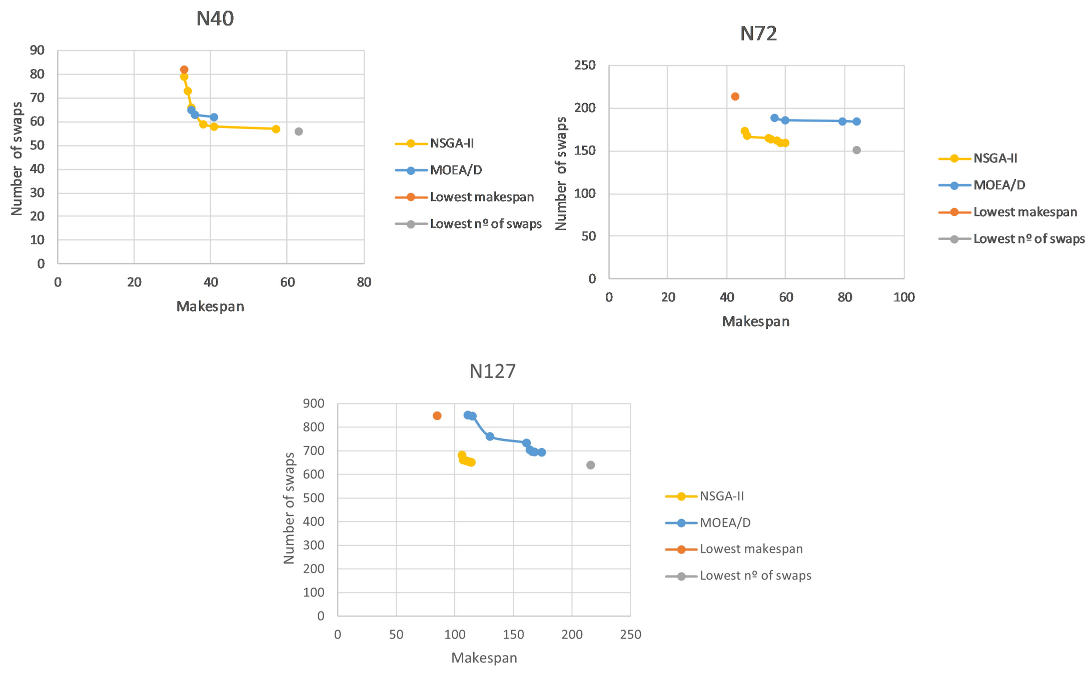


Figure 4: Example Pareto fronts on a 40-qubit circuit (upper left), a 72-qubit circuit (upper right) and a 127-qubit circuit (lower).

from the preliminary results it seems that the spread of solutions of the obtained Pareto fronts can be improved, so we plan to devise some enhancements to the multiobjective procedures. Applying QAOA to different problems, as the Minimum Vertex Cover Problem or the Graph Coloring Problem is also an interesting avenue for future work.

Acknowledgments

This research has been supported by the Spanish Government under research projects PID2019-106263RB-I00 and PID2022-141746OB-I00. A. Oddi and R. Rasconi were supported by the ESA Contract No. 4000112300/14/D/MRP "Mars Express Data Planning Tool MEXAR2 Maintenance".

References

[1] D. Venturelli, M. Do, E. G. Rieffel, J. Frank, Temporal planning for compilation of quantum approximate optimization circuits., in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI 2017), 2017, pp. 4440–4446.

[2] E. Farhi, J. Goldstone, S. Gutmann, A quantum approximate optimization algorithm, 2014. arXiv:1411.4028.

[3] E. A. Sete, W. J. Zeng, C. T. Rigetti, A functional architecture for scalable quantum

computing, in: 2016 IEEE International Conference on Rebooting Computing (ICRC), 2016, pp. 1–6.

[4] J. Preskill, Quantum Computing in the NISQ era and beyond, Quantum 2 (2018) 79.

[5] M. Brooks, Beyond quantum supremacy: the hunt for useful quantum computers, Nature 574 (2019) 19–21.

[6] G. Li, Y. Ding, Y. Xie, Tackling the qubit mapping problem for nisq-era quantum devices, in: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 1001–1014.

[7] L. Arufe, R. Rasconi, A. Oddi, R. Varela, M. A. González, New coding scheme to compile circuits for quantum approximate optimization algorithm by genetic evolution, Applied Soft Computing 144 (2023) 110456.

[8] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Transactions on Evolutionary Computation 6 (2002) 182–197.

[9] Q. Zhang, H. Li, Moea/d: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (2007) 712–731.

[10] E. Zitzler, L. Thiele, Multiobjective optimization using evolutionary algorithms — a comparative case study, in: Parallel Problem Solving from Nature — PPSN V Proceedings, Springer Berlin Heidelberg, 1998, pp. 292–301.