ISSN 2464-4617 (print)
ISSN 2464-4625 (online)
Quantum Informatics, Computing & Technology 2025
QC-Horizon 2025
https://www.qc-horizon.eu/

Studying how changes on the transpilation and the number of shots can be used to optimize quantum circuits

Elena Desdentado
Institute of Technology and Information Systems, University of Castilla-La Mancha
Camino Moledores
13005, Ciudad Real, Spain
elena.dfernandez@uclm.es

Macario Polo
Institute of Technology and Information Systems, University of Castilla-La Mancha
Camino Moledores
13005, Ciudad Real, Spain
macario.polo@uclm.es

Coral Calero
Institute of Technology and Information Systems, University of Castilla-La Mancha
Camino Moledores
13005, Ciudad Real, Spain
coral.calero@uclm.es

M. Angeles Moraga
Institute of Technology and Information Systems, University of Castilla-La Mancha
Camino Moledores
13005, Ciudad Real, Spain
mariaangeles.moraga@uclm.es

ABSTRACT

Context: Quantum computing is an emerging technology with the potential to address problems that are unsolvable for classical systems. However, adapting quantum software to current hardware constraints remains a significant challenge, particularly when aiming for efficient and sustainable executions.
Objective: This study explores how two strategies applied during the preparation and execution stages of quantum software development can affect the results and energy consumption of the executions of a simple quantum circuit.
Method: We analyse the impact of (1) the explicit selection of physical qubits within a quantum device, and (2) the number of individual executions per measurement. For each strategy, 12 executions are performed. From each execution, the outcomes of the measurements (to calculate the success rate of each case), and the execution times are collected.
Results: Our analysis shows that the selection of physical qubits has a significant effect, showing a 20% drop in success rate and an increase in execution time (and therefore energy consumption) between a suboptimal qubit allocation versus a better allocation. Furthermore, the number of shots influences the quality of the outcomes, the results suggest that there might be an optimal number of shots for each circuit.
Conclusions: The strategies disclosed in this study seem to affect the final outcomes and runtime required to execute our simple circuit. These strategies should be analysed for more complex circuits, and we should also check how their combination might affect the final results.

Keywords

Quantum computing, Quantum software, Quantum technology, Energy efficiency, Energy consumption

1 INTRODUCTION

The promise of quantum computing lies in its ability to address problems that are unsolvable for classical systems. Although theoretical advances continue, the practical deployment of quantum algorithms remains constrained by the limitations of current hardware. Moreover, quantum devices must be kept at temperatures close to absolute zero to function properly, which requires energy-intensive cooling systems.

Although quantum computing is often evaluated in terms of metrics such as performance, execution time, fidelity, or quantum speedup, energy consumption is not considered one of them ([Des25b]), despite the significant operational demands of these systems, as outlined above. In a previous study, [Des24], we observed that quantum computing can consume up to 150,000 times more energy than its classical counterpart when solving equivalent simple problems, indicating the urgent need to incorporate energy efficiency into the evaluation criteria for this technology.

In this context, optimization strategies become essential to adapt computations to the actual hardware while improving execution efficiency. Beyond functional correctness, these strategies are increasingly driven by sustainability concerns. As quantum systems scale up, reducing the energy consumption of these computations becomes a priority. We could distinguish two moments of intervention for these strategies [Des25a], during *circuit design and development*, and during *circuit preparation and execution*. This article explores how two strategies applicable during *circuit preparation and execution* can contribute to more sustainable quantum computing.

2 OPTIMIZATION STRATEGIES: EXPERIMENTAL INSIGHTS

In this work, two possible optimization strategies applied to quantum gate-based computation are explored.

These two strategies involve: (1) the selection of the qubits employed from a quantum computer, and (2) the number of individual executions (shots) of a circuit. The objective is to determine whether changes on these two aspects can affect the outcomes obtained and the execution time, and therefore, the energy consumption required to execute a quantum circuit. Let us recall that quantum computers have a constant power consumption, which corresponds to the energy required to maintain the temperature necessary for their operation. Therefore, the energy consumption of a circuit can be directly determined by the time it takes to execute.

Transpilation and physical layout

In digital computation, the development of circuits, firmware, or low-level software must consider the processor's architecture to take advantage of all its characteristics. For this purpose, in quantum computing, the process of transpilation consists on translating the program or circuit written by the developer into a physical circuit adapted to the actual architecture and set of gates of the target QPU [IBM25a]. In general, transpilation is transparently performed by a classical program offered by the quantum computing provider, which generates the tailored code before sending it to execute on the actual computer.

Thus, transpilation takes into account the set of gates available on the target QPU and the physical layout of its physical qubits, with the aim of transforming the original circuit into an equivalent one that can run efficiently on real quantum hardware.

The physical layout of a quantum computer determines how the qubits within this computer are organized and connected, which lets us know where the qubits are and how they can interact with each other. Considering the layout of the quantum computer selected to execute our algorithms is very important, as not all qubits can communicate with each other. The graphical representation of this layout is called coupling map. Figure 1 shows the coupling map of the "*ibm_brisbane*" QPU. Then, the physical layout: refers to the connections between qubits inside the quantum computer; it depends on the design of the computer; and cannot be modified.

Besides the physical layout, there is also a virtual layout. The virtual layout of a quantum computer is a logical representation that is employed while developing a quantum circuit, and it does not represent the real connection between qubits (modified during transpilation to make a better fit for the computer). Thus, the virtual layout serves as an abstraction that simplifies programming.

Consider the "*2 taxis, 3 people problem*", consisting of the allocation of three people in two taxis, the only restriction is that passenger one does not stand the others and must travel alone. To solve it we need 3 qubits,

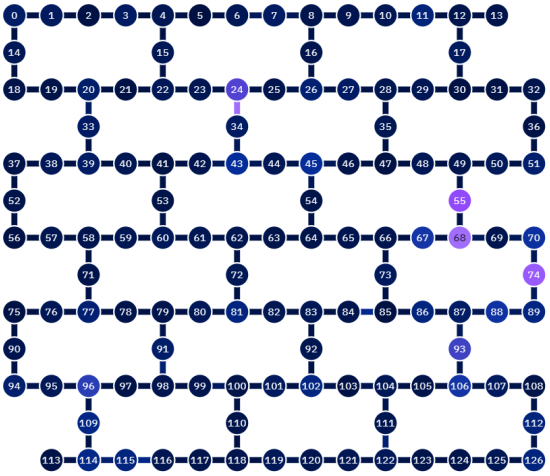

Figure 1: Coupling map of the IBM Quantum's computer "*ibm_brisbane*".

each representing the final allocation of the person in a taxi. Figure 2 shows a graphical representation of the two possible valid outcomes for this algorithm:"*011*" and "*100*", depending on the assigned taxi.

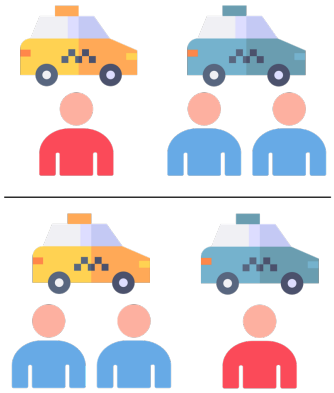

Figure 2: Representation of the possible solutions of the "*2 taxis, 3 people problem*".

With this simple problem, we want to check the effect of modifying the transpilation process, by explicitly indicating the physical qubits we want to employ, on: (1) the success rate obtained, and (2) the energy consumption of the executions. In Figure 4 (top) we can see the logic circuit which solves the "2 taxis, 3 people problem", and, as it can be observed, the three qubits needed are connected 1-2 and 2-3.

Experimentation

To analyse how the physical qubits allocation affects the success and energy consumption, we have executed the algorithm on IBM's computer "*ibm_brisbane*", in three different ways:

1. As is: i.e., letting the system assign physical to logical qubits. For transpiling, the target computer offers four optimization levels, 0 (no optimization) to

3 (high optimization), being level 2 the default setting.

2. HQS (*High Qubit Separation*): during transpilation, IBM's system allows the quantum developer to select the physical qubits to place the logical ones. For this execution configuration, we have selected qubits 0, 63 and 126: as noted in Figure 3, they are placed as far as possible.

3. BQS (*Best Qubit Selection*): for this execution configuration, we have selected qubits 75, 76 and 90. In this case, we checked the coupling map to select qubits with a lower error individually and a better connection (less connection error) than the ones selected by default in the "*as is*" version.

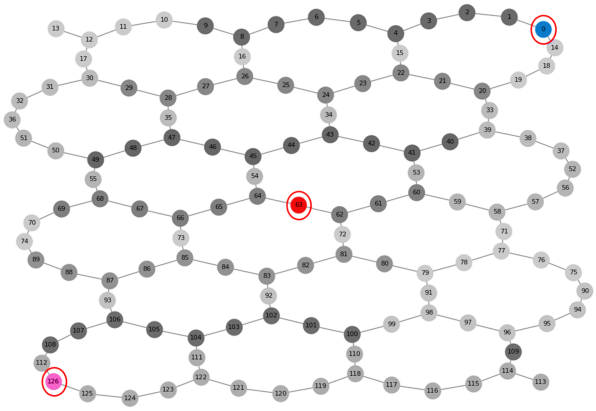

Figure 3: Qubit selection for the HQS version of the "*2 taxis, 3 people problem*".

To put the differences between these three versions into perspective, we will use two metrics: size and depth. Size refers to the total number of quantum operations applied, including logic gates and measurements. Depth represents the minimum number of sequential steps needed to execute the circuit, considering which operations can be applied in parallel.

The "*as is*" and the BQS versions present a size of 8 and a depth of 6, while the HQS version reaches a size of 487 and a depth of 194. Therefore, by separating the physical qubits as much as possible, the transpiler needs to add 471 logic gates (size) and 186 execution layers (depth) to execute the HQS version in a real quantum computer.

Figure 4 includes the logic circuit which solves the "*2 taxis, 3 people problem*" (top), the transpiled circuit of the "*as is*" version (middle), and the transpiled circuit of the BQS version (bottom). Note that the image of the transpiled circuit of the HQS configuration is not included due to its excessive size, which prevents it from being represented as an image.

All versions were executed twelve times on IBM Quantum's quantum computer "*ibm_brisbane*". For each execution, we used the default number of shots (1024) and optimization level (2) defined by the platform. Once the execution finished, we recorded the Qiskit runtime usage and the results obtained. We then filtered the results obtained to take into account only the two possible good results ("011" and "100") with the aim of calculating the success rate percentage. Figure 5 presents the average success rates of all versions. As we can see, the physical distribution of qubits selected matters. In this case, if we compare the optimized (BQS) and the suboptimal selection of qubits, we can see that there is a 19% drop in success rate.

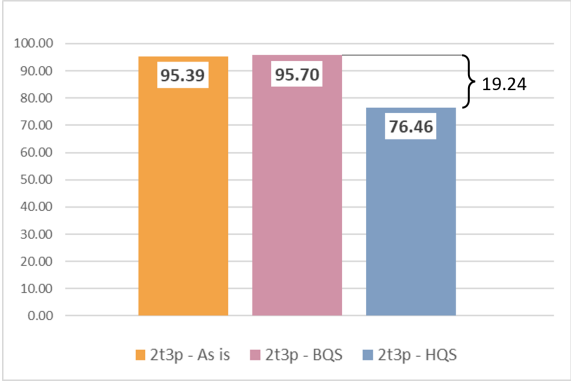

Figure 5: Success rates (%) obtained for all versions of the "*2 taxis, 3 people problem*" circuit.

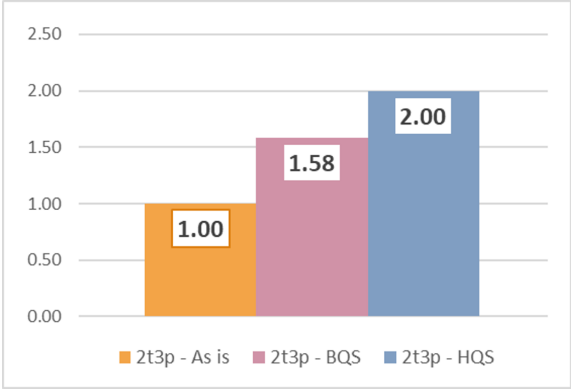

Figure 6: Average execution times (s) obtained for both versions of the "*2 taxis, 3 people problem*".

Moreover, Figure 6 compares the average execution times (in seconds) of the executions performed. The High Qubit Separation (HQS) version requires twice as much execution time as the "*as is*" version, and therefore also consumes twice as much energy. The BQS configuration requires 0.58 seconds more time to execute than the "*as is*" version. It is worth noting that the time values provided by the platform are rounded figures. This implies that the exact execution time is unknown, and that the actual value may differ from the reported one. Nevertheless, these are the only time measurements available, and even if they are not precise, they are sufficient to determine which execution

is longer and, therefore which one consumes the most energy. In the case of the BQS version, the time value is obtained calculating the mean time of the 12 executions. As in some cases the execution time was 1 second, and in other cases 2 seconds, the average of all of them 1.58 seconds.

The differences observed between all versions measured reinforce our initial idea: optimizing the layout can improve the success rate, although this can come at the cost of a slightly increased execution time. We could observe that the suboptimal version (HQS) performs the worst in both dimensions, achieving a success rate 19% lower and requiring twice the execution time compared to the "*as is*" version. Therefore, finding an optimized selection of qubits can help us to achieve a better balance between success rate and execution time, as our goal is not always to minimize energy consumption, but rather find said balance.

Impact of the number of shots

The quantum computers we can currently use are prone to errors due to quantum noise (small deviations when manipulating subatomic particles) and decoherence (the system cannot completely isolate itself from its environment), which lead to inaccuracies and errors in the results provided by the quantum computer.

To address this, the most common solution is to run the quantum algorithm several times per measurement, known as number of *shots*. This section focuses on observing if the selected number of shots influences on the success rate and the energy consumption required to execute a circuit.

Experimentation

In theory, and in general belief, the more shots taken, the closer the obtained probability distribution is to the actual solution. We have executed the "*2 taxis, 3 people problem*" circuit twelve times on IBM Quantum's quantum computer "*ibm_brisbane*", with 50; 1,024 (default); 4,500; 10,000; and 24,500 shots. For each execution, we used the default optimization level (2) defined by the platform.

Figure 7 shows a different reality even for such this simple problem: in fact, the success rate (i.e., the percentage of right solutions) tends to grow as the number of shots increases, but starts to decrease when the number of shots seems "too high".

Employing more shots, as we can see, is not always the best option, likely because as the more shots you run, the more times the circuit is exposed to the noisy environment of quantum computers.

It seems that there could be an optimum number of shots to be employed, a value that could change depending on the characteristics of the quantum circuit to be executed.

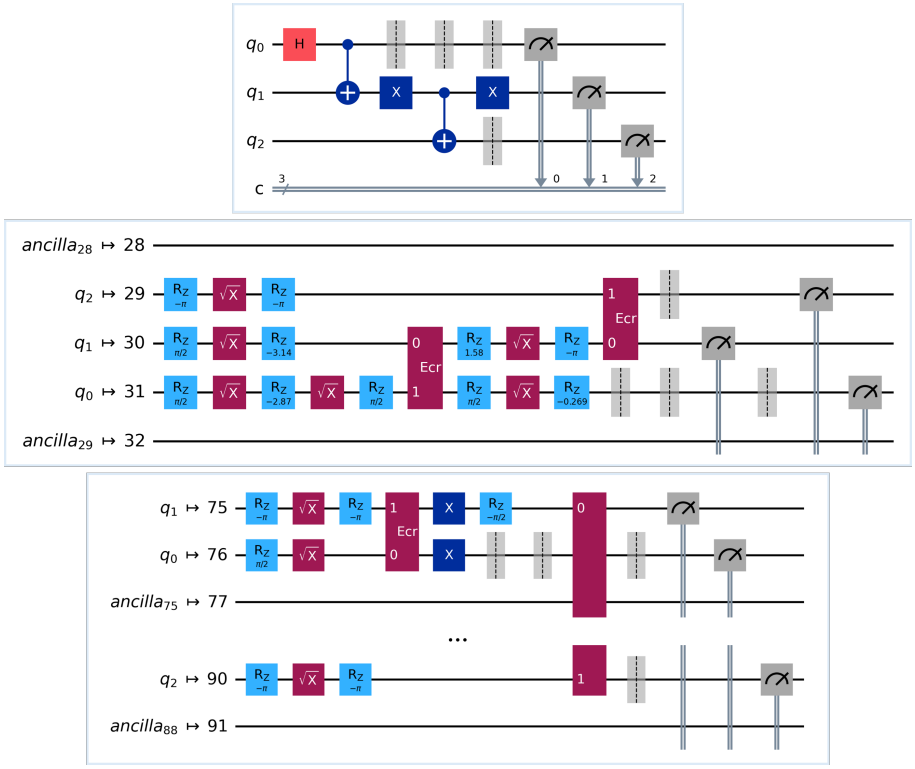

Figure 4: Logical circuit for the "*2 taxis, 3 people problem*" (top); transpiled "*as is*" version circuit with 2 as optimization level (middle); and transpiled BQS version with 2 as optimization level (bottom).

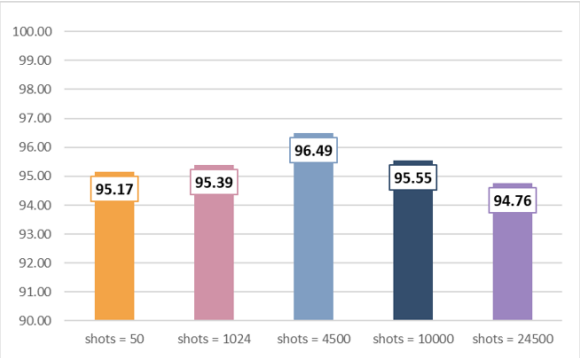

Figure 7: Success rates (%) obtained for 50, 1,024, 4,500, 10,000, and 24,500 shots.

It is also interesting to analyse the relationship between the execution times with respect to the number of shots employed (Figure 8): it is clear that the more shots employed in the execution, the more execution time and therefore the more energy consumption is needed for the quantum computer to run it. In line with our recommendation to look for a balance between time (and therefore energy consumption) and success rate, in this case, it appears that choosing 1,024 shots over 50 is preferable, as both require the same execution time but 1,024 yields a higher success rate. Nevertheless, the best option might be 4,500 shots, since although it takes more than twice the time compared to 1,024 shots, it results in an increase of over one percentage point in the success rate.

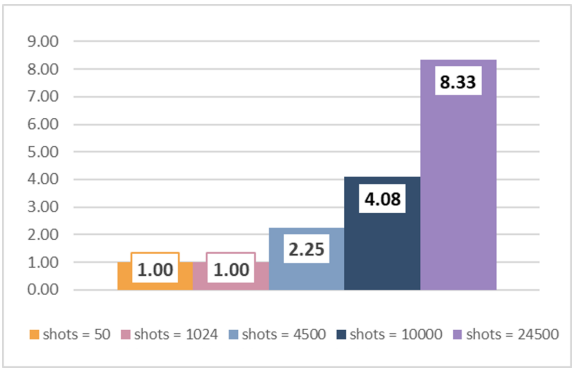

Figure 8: Execution times (s) obtained for 50, 1,024, 4,500, 10,000, and 24,500 shots.

3 CONCLUSIONS AND FUTURE WORK

The analysis of the two strategies presented in this paper, applied to the *preparation and execution* stage of quantum software development, reflects firstly that the physical qubits selected to perform the execution of a quantum circuit can be crucial. As we could see comparing the "*as is*" version, the BQS (Best Qubit Selection) version, and the HQS (High Qubit Separation) version executions, a suboptimal allocation of resources, in this case of qubits, greatly affects the results. Largely impacting the success rate percentages obtained, and also the time needed and energy consumption to perform the execution itself. And that a better allocation of qubits can obtain a better success rate percentage, with a slight increase of execution time, suggesting the need to find a balance between both dimensions.

And, secondly, that the number of shots employed is also crucial, with results suggesting that there might be an optimal number of shots to be used, that might change from one circuit to another.

In the future we would like to expand this study, employing these two strategies with more complex circuits, experiment with other strategies identified, such as the simplification of circuits, the replacement of error-prone gates, or shortening the search space tree for algorithms that produce specific probability distributions (i.e. Grover's algorithm [Gro96]). We also plan to evaluate the combined use of these strategies. As each technique focuses on one aspect of quantum circuit design, development and execution processes, the combination of two or more strategies could lead us to more robust, adaptable, and scalable quantum solutions.

4 ACKNOWLEDGMENTS

This work is part of, and has received financial support from, the following projects:

- OASSIS: PID2021-122554OB-C31/AEI/10.13039/501100011033/FEDER, UE.

- PLAGEMIS: Grant TED2021-129245B-C22 funded by MCIN/AEI/ 10.13039/501100011033 and European Union NextGenerationEU/PRTR.

- EMMA: Project SBPLY/21/180501/000115, funded by CECD (JCCM) and FEDER funds. Financial support for the execution of applied research projects, within the framework of the UCLM Own Research Plan, co-financed at 85% by the European Regional Development Fund (FEDER) UNION (2022-GRIN-34110)

In addition, the first author has a pre-doctoral FPI (Research Staff Training) contract at the University of Castilla-La Mancha, Spain, for which funding has been received from the European Union through the European Social Fund Plus (ESF+).

5 REFERENCES

[Des25a] Calero C., Polo M., Desdentado E., and Moraga M.A. Consider the energy consumption of your quantum circuits *Nature Reviews Physics* 7, 352-353, June 2025. https://doi.org/10.1038/s42254-025-00846-0

[Des25b] Desdentado E., Calero C., Moraga M.A., and Garcia F. Quantum computing software solutions, technologies, evaluation and limitations: a systematic mapping study. *Computing*, 107(5):110, April 2025. https://doi.org/10.1007/s00607-025-01459-2

[Des24] Desdentado E., Calero C., Moraga M.A., Serrano M., and Garcia F. Exploring the trade-off between computational power and energy efficiency: An analysis of the evolution of quantum computing and its relation to classical computing. *Journal of Systems and Software*, 217:112165, July 2024. https://doi.org/

[IBM25a] IBM. Introduction to transpilation. IBM Quantum Documentation, 2025. https://docs.quantum.ibm.com/guides/transpile

[IBM25b] IBM. Set transpiler optimization level. IBM Quantum Documentation, 2025. https://docs.quantum.ibm.com/guides/set-optimization

[Gro96] Grover, L.K. A fast quantum mechanical algorithm for database search, in Proc. 28th Annual ACM Symposium on Theory of Computing, Philadelphia, Pennsylvania, USA, ACM Press, pp. 212-219, 1996.