Genetic Algorithm for Qubits Initialisation in Noisy Intermediate-Scale Quantum Machines: The IBM Case Study

Zakaria Abdelmoiz Dahi*
zakaria.dahi@{uma.es,univ-constantine2.dz}
ITIS Software, University of Malaga
Fac. NTIC, University of Constantine 2
Malaga/Constantine, Spain/Algeria

Francisco Chicano
chicano@lcc.uma.es
ITIS Software, University of Malaga
Malaga, Spain

Gabriel Luque
gabriel@lcc.uma.es
ITIS Software, University of Malaga
Malaga, Spain

Enrique Alba
eat@lcc.uma.es
ITIS Software, University of Malaga
Malaga, Spain

ABSTRACT

Discrete-variable gate-model quantum machines are promising quantum systems considering their wide applicability. Being in their noisy-intermediate-scale era, they allow executing only circuits of limited complexity and fitting the machines' features. Thus, such systems implement a key and unavoidable tailoring process to produce the most possible compact and device-compliant circuit. The qubits' initialisation is a primary and complex step that can ease/jeopardise the tailoring process and restrict/extend the machine's computational capacities. Ultimately, this bottleneck can be responsible of making quantum leaps like quantum supremacy. As a step towards the former, this work investigates how evolutionary algorithms can enhance the qubits' initialisation by tackling it as a single-objective problem using a genetic algorithm. The experiments used instances representing 19 real IBM quantum machines with 7 to 65 qubits and 9 different qubit topologies. Also, 76 GHZ circuits of sizes 7-65 qubits and 25%-100% of entanglement were created and studied. Extensive standard and statistical comparisons have been made against the IBM qubit initialiser that is currently used in real quantum machines. Results showed that the proposal outperforms IBM in 64 instances and is similar to it in 10 ones, with an average circuit-compression gain up to 46%.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**; • **Mathematics of computing → Combinatorial optimization**.

KEYWORDS

Quantum computation, quantum circuit initialisation, evolutionary algorithms

GECCO '22, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9237-2/22/07…$15.00
https://doi.org/10.1145/3512290.3528830

ACM Reference Format:
Zakaria Abdelmoiz Dahi, Francisco Chicano, Gabriel Luque, and Enrique Alba. 2022. Genetic Algorithm for Qubits Initialisation in Noisy Intermediate-Scale Quantum Machines: The IBM Case Study. In *Genetic and Evolutionary Computation Conference (GECCO '22), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3512290.3528830

1 INTRODUCTION

Quantum computers are based on peculiar principles of quantum mechanics (e.g. superposition, entanglement, etc.) and counter-intuitive blocks of information (e.g. quantum bits, or qubits) to achieve a computational speedup compared to their classical counterparts. This quantum advantage over classical systems is promised to be revolutionary and widely applicable in real-life domains (e.g. drugs discovery), which could allow many breakthroughs in the short, mid, and long terms. Quantum calculation is a very precise computation, but most of today's quantum devices are noise-sensitive with limited qubits. Such devices are usually denoted as Noisy Intermediate-Scale Quantum systems (NISQ) [6]. Several quantum computation paradigms can be implemented on such devices. The Discrete Variable Gate-Model (DVGM) is one of the most popular ones considering their broader applicability than other paradigms (e.g. adiabatic).

DVGM machines express quantum processes as circuits that consists in applying a series of operations, called quantum gates, to the qubits. To be able to run on a target machine, a quantum circuit needs to fit the machine's features/capacities (e.g. basis gates). To do so, a very complex tailoring process is needed, where each step is in charge of fulfilling a given system's constraint. Considering the number/difficulty of such constraints, the resulting tailored circuit is often more complex than the original one. Thus, the tailoring's efficiency decides on whether a circuit can (or not) be executed on the target machine and rules the reliability of the computation it performs. The challenge lies in producing the most compact hardware-compliant circuit. Indeed, optimal tailoring would allow executing more complex and reliable circuits (i.e. calculations), which could allow breakthroughs in both quantum and classical fields (e.g. quantum supremacy [10, 14]).

One of the preliminary and most critical steps during the circuit tailoring is the qubits' initialisation. It consists in assigning the abstract qubits of the circuit to physical ones on the target machine.

The efficiency of this step can jeopardise or enhance the efficiency of the whole tailoring [8]. Finding the ideal qubits' allocation is proven to be NP-complete [17], which makes it intractable for classical techniques. Considering these facts, this work explores the contribution that Evolutionary Algorithms (EA) can provide for enhancing the quality of qubits' initialisation in DVGM, the tailoring process in general and finally, the computational limits of NISQ machines. To do so, the Qubits' Initialisation is tackled as a single objective Problem (abbreviated QIP) using a Genetic Algorithm (GA). A simple GA has been used on purpose instead of advanced ones (e.g. linkage-learning [15]) to demonstrate the potential future applicability that evolutionary computation could have on such a problem. Extensive experiments have been performed using 19 real IBM quantum machine topologies of sizes 7, 16, 20, 27, 53 and 65 qubits. Also, 76 GHZ circuits of sizes 7, 16, 20, 27, 53 and 65 qubits and entanglement of 25%, 50%, 75% and 100% have been created and studied. The comparison was made against IBM tailoring routine currently used in real IBM quantum machines. The results have been analysed using six standard metrics and statistical tests.

The rest of the paper is structured as follows. In Section 2, the basic background of quantum computing and the qubits' initialisation are given. Then, in Section 3, the problem modelling as well as the devised approach to solve it are both presented. Later, Section 4 presents the experimental study conducted to assess the quality of the proposal. Finally, Section 5 concludes the paper.

2 BACKGROUND

This section presents essential background related to quantum computing and qubits' initialisation in DVGM NISQ machines.

2.1 Quantum Computing Preliminaries

The Quantum Computation (QC) paradigm was born by including quantum mechanics phenomena (e.g., entanglement, superposition, etc.) into classical computation. The substantial theory lies behind QC (e.g., universality, certainty conservation, etc.), and several paradigms exist (e.g., adiabatic [5], gate-based [6], etc.), where each one differs on the theory it is based on (e.g. adiabatic theorem [5], circuits [6], etc.), the quantum information representation (e.g., qumode, qubits, etc.), the technology it uses (e.g., Ion trapped, silicon, etc.) and its applicability range. This work focuses on DVGM quantum machines using Superconducting qubits (DVGM-S) considering the interests they are getting in industry and academia due to their current and future broad applications. Indeed, leading companies in the field such as IBM, Google and Rigetti use this paradigm/technology (e.g., IBM System One [3], Google Sycamore [6], etc.).

QC in DVGM manipulates a particular piece of information named qubit that, until it is observed, can have a state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ which is the superposition of two quantum states $|0\rangle$ and $|1\rangle$ with amplitude probabilities $\alpha, \beta \in \mathbb{C}$. The value $|\alpha|^2$ represent the probability of measuring 0, while $|\beta|^2$ is the probability of observing a 1, and $|\alpha|^2 + |\beta|^2 = 1$. A quantum state lives in Hilbert space, which is a vector space with additional properties. Thus, we can use explicit vectors to represent the states using Dirac notation (e.g., $|0\rangle$) or linear algebra notation (e.g., $\begin{bmatrix}1\\0\end{bmatrix}$). Both will be interchangeably employed in this work. Moreover, a qubit state can be geometrically represented using the Bloch sphere. The state of d qubits can be represented with a vector in the tensor product of d 1-qubit vector spaces, that is, $\mathbb{C}^{2^d}$.

In DVGM, the computation (i.e., programs) is expressed as quantum circuits describing a series of operations, called quantum gates, applied on qubits. Quantum gates are unitary matrices respecting $U^\dagger \cdot U = U \cdot U^\dagger = I$, where $U^\dagger$ is the Hermitian adjoint of the matrix U and I is the identity matrix. For example, the matrices defined in Equations (1) and (2) represent the Hadamard (H) and Controlled-Not (CNOT) gates, respectively.

$$\mathtt{H} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \quad (1) \qquad \mathtt{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2)$$

The Hadamard gate is applied to one qubit, and the Controlled-Not is applied to two qubits. Many other quantum gates exist, and in real quantum machines, circuits might even include non-gates operations (e.g., measurement) or unconventional implementations (e.g., barrier). Figure 1 illustrates a 3-qubit circuit where 1 Hadamard, 3 CNOT and 3 measurement gates are applied.

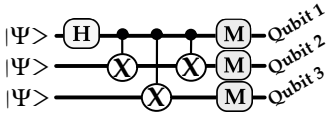

Figure 1: Example of 3-qubits quantum circuit

2.2 Qubits Initialisation in DVGM-S Machines

The quantum circuits' tailoring generally translates the virtual circuit into an *equivalent* physical one that fits the features (e.g., basis gates, hardware topology, etc.) of the machine where it will be executed. As a general rule, 2-qubit gates can be only executed if the involved qubits are physically connected (i.e., adjacent). For several reasons, the qubits' connectivity topology is an important factor for performance. Nonetheless, when developing a quantum program (i.e., designing a virtual circuit), such information is either not accessible or too complex to be considered. Thus, virtual circuits often involve gates not respecting the qubits' adjacency. The tailoring module implements a mapping process where a set of SWAP gates are inserted in order to move the qubit(s) state(s) of non-adjacent qubit(s) to two adjacent physical qubit(s) on the target machine. It is known that SWAP gates induce additional computation time and increase the circuit error. This is often due to an increase in the circuit's complexity [11], where one SWAP will be converted into three CNOT gates. Thus, inefficient mapping makes the circuit impossible to run on the target machine or provide unreliable results due to the excessive error rate. To cope with this, the tailoring module generally includes also a post-processing step as an attempt to reduce the complexity of the tailored circuit (e.g., number of gates, etc.). Here again, the efficiency and feasibility of the compression mechanisms (e.g., gates' cancellation, fusion and commutation) depends on how complex the mapped circuit is. The quality of each tailoring step is quite dependent on the goodness of the step that precedes it.

Of course, the tailoring process might probably include additional steps such as 3-qubit gates decomposition and native gates translation. The workflow might also differ from one quantum machine to another. This work considers a typical tailoring process likely to be common among several DVGM machines.

One of the primary and most critical steps that usually precedes the tailoring steps (e.g., mapping, compression, etc.) is the circuit initialisation. This phase is proved to be critical and has a vast impact on the entire tailoring process [8]. In this phase, the abstract qubits in the virtual circuit are initialised (i.e., assigned) to real physical qubits in the quantum machine. Figure 2(a.II) drafts an example of a 3-qubit virtual circuit to be executed on a 3-qubit machine with the qubits' connectivity shown in Figure 2(a.I). Figure 2(b.I) shows that the first configuration of qubits' initialisation makes that no SWAP gates are inserted during the mapping step. Therefore, no optimisation will be required at the optimisation step. On the other hand, the second qubits' initialisation given in Figure 2(b.II) induced a SWAP gate to move the state of physical qubit 1 to the qubit 2. This will increase the circuit complexity, including its execution time and error rate. Both might compromise the feasibility of the execution and the correctness of the results.

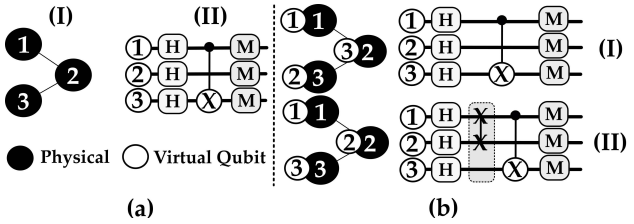

Figure 2: Circuit tailoring: (a) qubits' connectivity and virtual circuit, (b) qubits' initialisation and physical circuit

Having d qubits, $d!$ possible qubits' initialisations are possible [9]. Finding the optimal qubits' initialisation is proven to be NP-complete [17]. Regarding the review in [8], the literature can be classified into works (I) neglecting the qubits' initialisation when tackling the tailoring process and this by assuming that a simple or random initialisation is given and (II) work that solves the quantum initialisation problem jointly with the remaining steps of the tailoring process (e.g., circuit mapping). The second class of works were proven to make the whole problem significantly more complex. Thus, the devised techniques were found to be successful/tractable for relatively small circuits. But, programs consisting of millions of gates will render most of them unfeasible. Thus, this work goes in the same line as the one done in [1, 8, 13] where the QIP is tackled separately. The aim is to build techniques that provide relevant qubits' initialisation for current and future DVGM-S. Ultimately, this will help build tailoring pipelines that extend the machines capacities, enhance the reliability of the computation so as to achieve advances in the quantum domain, including reaching the quantum milestones such as the supremacy regime.

3 THE DEVISED APPROACH

This section presents both the QIP mathematical modelling and formulation considered in this work, and the GA-based routine devised for solving it.

3.1 Problem Modelling and Formulation

In this work the QIP is tackled as a combinatorial optimization problem, where the aim is to find the adequate (*virtual*, *physical*) pairings. In particular, having a virtual circuit involving d qubits $V = \{v_1, \ldots, v_d\}$ to be executed on a quantum machine with m qubits $Q = \{p_1, \ldots, p_m\}$, where $d \leqslant m$, the goal is to find an assignment $\pi : V \rightarrow Q$ of each virtual qubit to a physical qubit: $\pi(v_i) \in Q$ with $\pi(v_i) \neq \pi(v_j)$ for $v_i \neq v_j$. In the proposed modelling, (I) the physical qubits are assigned to virtual ones in this order $1, \ldots, d$, (II) the order of physical qubits matters and no repetitions are allowed. Thus, the total number of possible physical qubits assignments is ${}_mP_d = m!/(m-d)!$ partial permutations. Figure 3(a.I) represents a standard representation of a QIP solution, where we use the notation $x_i = \pi(v_i)$, while Figure 3(a.II) represents a possible solution to a QIP initialising a 4-qubits circuit on a 4-qubits machine. The virtual qubits 1-4 are assigned to the physical ones 2, 4, 3 and 1, respectively. On the other hand, Figure 3(b) illustrates the initialisation provided by such solution. The used QIP modelling is made in order to fit naturally the requirements of IBM tailoring process subject to qubits'-initialisation-format restriction.

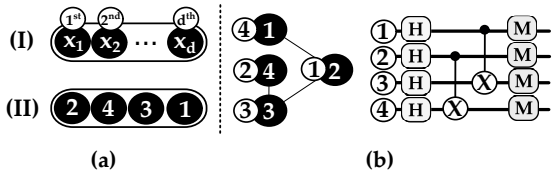

Figure 3: QIP: (a) solution and (b) illustration

Solving the QIP aims at enhancing the tailoring pipeline, which ultimately seeks to produce the most compact and hardware-compliant circuit. The compactness of the circuit could be interpreted in several ways (e.g. size, width, etc.). In the context of quantum circuit tailoring, it is usually reflected by the circuit's *depth*. Indeed, when considering works tackling the QIP individually [8] or the whole circuit tailoring jointly [2], the depth is unavoidably considered as a solution quality metric. Going in this same line of thoughts, in this work, the depth is also considered as a fitness function of the QIP solutions. Mathematically, it can be described as the longest path (i.e. critical path) in a directed acyclic graph $G(V, E)$, where V is the vertices of the graph representing the quantum gate applied on a given qubit, while E are the edges representing the execution dependency between different gates to be executed on the same qubit or independent ones. From a quantum-circuit perspective, it can be seen as the longest sequence of gates' execution a quantum circuit needs to accomplish in sequence (serially). A more simplified interpretation might be to consider the circuit depth as how many quantum gates' layers executed in parallel it takes to complete the computation defined by the circuit. In all cases, the depth metric emphasises that the longer the execution series, the higher the error rate. Thus, in this work, the objective is to minimise the depth of the tailored circuit. Figures 4(a) and (b) represent a layer and graph-based calculation of the circuit depth, which is four.

Having a QIP solution x, it will lead the tailoring process to generate a circuit with some gates' execution dependency at each time step. This results in w possible series of executions $Y = (y_1, \ldots, y_w)$,

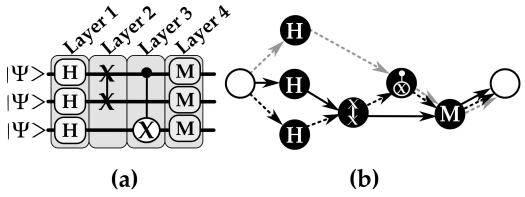

Figure 4: Depth illustration: (a) circuit and (b) DAG

where $y_i = (g_{i,1}, \ldots, g_{i,k_i})$ having $i = 1, \ldots, w$ and k_i the number of gates the i^{th} execution series is made of. The circuit's depth, to be *minimised*, would be the number of gates composing the execution series with the largest gates' count. In practice, the longest path in a Directed Acyclic Graph (DAG) can be computed in several ways including a dynamic programming-based one. In this present work, we use the layer-based IBM depth calculator implemented in Qiskit [7].

3.2 The Proposed Genetic Algorithm

We used in this work a simple genetic algorithm (GA) instead of an advanced EA such as linkage-based ones (e.g. [15], [16] and [18]) to explore and encourage any potential contribution/applicability that evolutionary computation might have in this problem. Considering this perspective, basic configurations and settings are used in each proposal's steps. A whole circuit tailoring pipeline is used, where the GA takes charge of the qubits' initialisation, while the remaining steps, including gate decomposition/translation, circuit mapping/optimisation are the original ones used currently by the IBM Qiskit [7]. Such design methodology has been followed to do a comparison with the IBM tailoring routine, and isolate the effect of the qubit initialisation provided by the GA. Thus, the GA treats the rest of the tailoring process as a black box, which helps the incorporation of the proposed approach in any other tailoring module (e.g. different machines, different manufacturers, etc.).

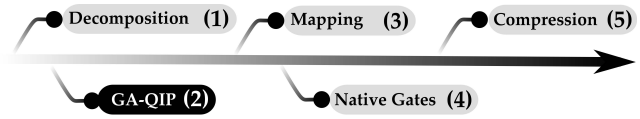

Figure 5: The GA-based tailoring routine

Besides the repair function to ensure the validity of the generated qubits' initialisation, a typical GA is being applied starting by an initialisation, selection, crossover, mutation and finally an evaluation of the solutions and replacement of the population. The process is iteratively executed during T iterations (Algorithm 1).

The GA starts by randomly initialising a population B of n QIP's solutions. Once this is done, the GA selects Z parents to form a mating pool S of $Z/2$ pairs that will go through the crossover and mutation. Two-point crossover with probability P_c of being applied is used. In the mutation, for each gene, a value is randomly chosen from $\{1, \ldots, m\}$ with probability P_m.

As explained in Section 3.1, for a solution to be valid, it must be a partial permutation (see Figure 3). The variation operators used in the GA can produce non-feasible solutions (e.g. two or more virtual qubits are initialised on the same physical one). In this case, applying a repair function on the population of offspring M is mandatory. The proposed mechanism consists in identifying the virtual qubits that have been assigned to the same physical qubit and the physical qubits that have not been assigned to any virtual one. As a second step, an iterative replacement of duplicate physical qubits by unassigned ones is performed until all the virtual qubits are assigned to one and only one physical qubit. This step produces a repaired population R of Z solutions. In the case the virtual circuit is smaller than the target machine capacity ($d \leqslant m$), the same process is applied by replacing the redundant physical qubit by the first unused one registered on the quantum machine.

Finally, the quality of the generated feasible solutions is evaluated using IBM circuit depth calculator, and a new population is built using the $(\mu + \lambda)$ replacement operator.

Algorithm 1 Pseudo-code of the Genetic Algorithm

```
B ← Generate_Population(n,d)
B′ ← Repair(B)
for T iterations do
    S ← Random_Selection(B′,Z)
    C ← Two_Point_Crossover(S)
    M ← Uniform_Mutation(C)
    R ← Repair(M)
    Circuit_Depth_Evaluation(R)
    B′ ← (μ + λ)_Replacement(B′,R)
end for
```

4 EXPERIMENTATION AND DISCUSSION

This section describes the experimental study performed to assess the efficiency of the proposal. This includes the hardware and software settings as well as the obtained results and their discussion. It is worth stating that the *realistic* aspect is emphasised in this work to maximise the applicability of the proposed approach as well as the results. This has been done by studying up-to-date quantum technologies (i.e. superconducting qubits), comparing with cutting edge techniques (i.e. IBM routine currently used in quantum computers [7]), and using real benchmarks (i.e. real IBM machines and circuits). The scalability aspect is also considered to assess the proposal by dealing with machines and circuits of different sizes and complexities.

4.1 Quantum Machines and Circuits

The difficulty of circuit tailoring comes from the complexity and diversity of two components: 1) the quantum hardware constraints it attempts to fulfill, and 2) the quantum circuit it tries to tailor. Thus, the experimentation has been built to include complex and diverse-enough configurations of these two aspects to challenge the GA's efficiency and scalability and ultimately unveil any bias, weakness and strength. Considering the quantum hardware constraints, 19 real IBM quantum machines instances of 7, 16, 20, 27, 53 and 65 qubits with 9 different qubit connectivity topologies were studied (see Figures 7(a)-(i)). The studied machines use one of two sets of native gates and various quantum-related configurations (e.g. gates'

errors/fidelity, qubits' errors, etc.) A complete list of the quantum hardware machines is presented in Table 1.

Table 1: The studied quantum machines

Qubits	Machine Name	Basis Gates
7	IBMQ LegacyCasablanca	id, rz, sx, x, cx, reset
	IBMQ Lagos	
	IBMQ Jakarta	
	IBMQ Casablanca	
16	IBMQ LegacyRueschlikon	u1, u2, u3, cx, id
	IBMQ Rueschlikon	
	IBMQ Guadalupe	id, rz, sx, x, cx, reset
20	IBMQ Singapore	id, u1, u2, u3, cx
	IBMQ Poughkeepsie	
	IBMQ LegacyTokyo	
	IBMQ LegacySingapore	
27	IBMQ Toronto	id, rz, sx, x, cx, reset
	IBMQ Sydney	
	IBMQ Paris	
	IBMQ Mumbai	
53	IBMQ Rochester	id, u1, u2, u3, cx
	IBMQ LegacyRochester	
65	IBMQ Manhattan	id, rz, sx, x, cx, reset
	IBMQ Brooklyn	

Regarding the second aspect of the experiments, the complexity of the circuits to be tailored is mainly influenced by the overhead of the qubits' entanglement it includes (i.e. amount of entangled qubits) and the complexity of the gates it is made of. A GHZ circuit is studied considering that the entanglement can be easily controlled. The studied circuits have several widths (7, 16, 20, 27, 53 and 65 qubits) and different entanglement percentages (25%, 50%, 75% and 100%). They are built up using minimum but sufficient single-qubit Hadamard (H) and 2-qubits Controlled-Not gates (CX) considering that the former is one of the most common native gates among the IBM quantum machines and DVGM-NISQ systems (see Table 1). Additionally, we add a measurement to observe the classical information carried by the qubit and simulate a real-world quantum computation. The experiments are performed by forcing the tailoring routine to tailor the sub-circuit generating the entanglement by using a `barrier` gate. Figure 6, represents the case of a GHZ circuit applied on an 6-qubits machine with 50% entanglement. One should keep in mind that the benchmarks are organised as {machine, entanglement percentage} pairs. Overall, 76 instances will be studied in this work, where quantum circuits are simulated using IBM Qiskit [7]. It is worth stating that the IBM tailoring routine provides four low level QIP solvers named "`trivial`", "`dense`", "`noise_adaptive`" and "`sabre`". In this work, the comparison is made against the default IBM QIP solver because (I) this is the default setting and it is likely that most users will use it. (II) since this work presents a trivial QIP solver, it could be equitable to compare it with the default IBM QIP solver. (III) during this work, the three IBM QIP solvers (except "`noise_adaptive`" that required unavailable quantum machine settings) were tested offline on one of the largest 65-qubits machines (i.e. IBMQ Manhattan) with a 100% entangled GHZ circuit. The difference found between the IBM QIP solvers was of the order of ±18.5 depth circuit difference on average.

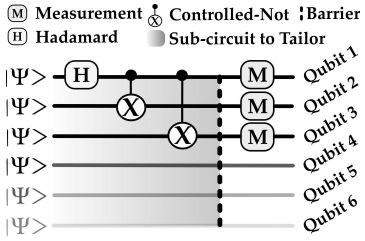

Figure 6: 50%-entanglement-GHZ circuit on 6-qubits machine

4.2 Classical Hardware and Software Settings

The implementation was done using Python 3.9.7, while a Linux Enterprise Server 15.2 has been used during the execution. The experiments were run in a cluster managed by `Slurm` and having the configuration summarised in Table 2. The experiments done on each pair of (machine, entanglement) are launched as independent `jobs`, where each execution is a `task` requiring 30 CPUs.

Table 2: Components of classical hardware cluster used

Nodes	CPU / GPU	RAM	InfiniBand	Localscratch
126 × SD530	56 × Intel Xeon Gold 6230R @ 2.10GHz	200GB	HDR100	950 GB
24 × Bull R282-Z90	128 × AMD EPYC 7H12 @ 2.6GHz	2TB	HDR200	3.5TB
168 × IBM dx360 M4	16 × Intel E5-2670 @ 2.6GHz	32GB	FDR40	400GB
4 × DGX-A100	8 × A100 Tensor Core	1TB		14 TB

The proposed genetic algorithm has a population of $n = 40$ individuals, $P_c = 0.5$, $P_m = 0.1$ and a mating pool of Z=50%. For each benchmark instance in each quantum machine configuration, algorithms have been run 30 times, where in each one the proposed approach is executed during 30 iterations and 1200 fitness evaluations are done in total. The comparison has been made against the IBM tailoring routine [7] that is currently implemented and used in IBM quantum machines. Regarding the stochastic nature of the IBM tailoring pipeline, when evaluating an individual in the proposed approach, the circuit is executed 30 times using the qubits' assignment it provides. The fitness of the individual is the median of the circuit depth throughout the 30 simulations. The same is applied to the IBM routine. Five metrics are reported and chosen to avoid unreliable results due to outliers that might bias the comparison: the Inter-Quartile Range (IQR), the `Median`, Median Absolute Deviation (MAD), the statistical test of the results obtained over 30 executions, and the median time in seconds needed for the proposed approach or IBM routine to perform one execution. Finally, the Mann-Whitney U test was applied using 5% significance.

4.3 Numerical Results and Discussion

Table 3 presents the results obtained using quantum GHZ circuits with 25%-100% of entanglement and machines with 7-65 qubits. Table 4 presents the percentage of the circuit's depth compression the GA could gain with respect to the one achieved by the IBM routine. Table 4 contains also the results of the statistical test performed to investigate the results in Table 3. Considering the metric `Median` in Tables 3 and 4, the best results are highlighted in colour, where light-gray designates that the GA is outperforming the IBM routine, while dark-gray represents the opposite scenario. Finally, black means that both approaches achieve equal results. It can be

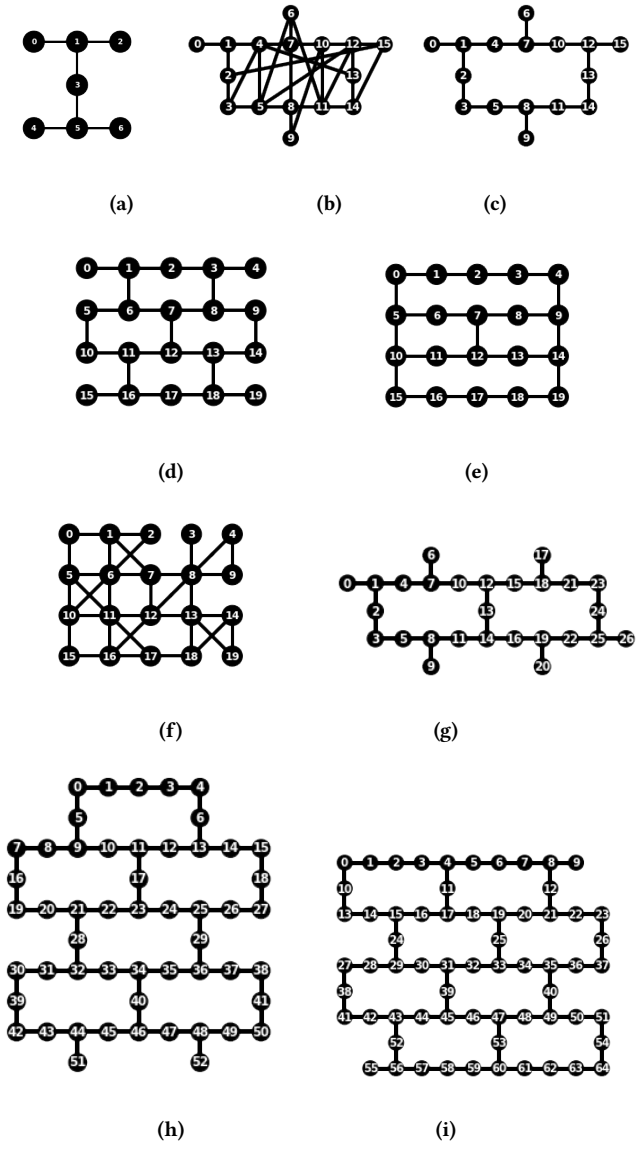

Figure 7: Qubits connectivity of physical qubits: (a) IBMQ LegacyCasablanca, Lagos, Jakarta, Casablanca (b) LegacyRueschlikon, Rueschlikon, (c) Guadalupe, (d) Singapore, LegacySingapore, (e) Poughkeepsie, (f) LegacyTokyo, (g) Toronto, Sydney, Paris, Mumbai, (h) Rochester, LegacyRochester and (i) Manhattan, Brooklyn

seen that the GA-based routine outperforms the IBM one in 64 instances, while in 10 benchmarks it achieves similar results to it, and finally, the IBM surpasses the GA in 2 instances. The results of the statistical tests in Table 4 confirm those presented in Table 3 except for the case of the machines `LegacyRueschlikon`, `Guadalupe` and `Rueschlikon` using 25% entanglement, where the statistical test shows that the results of both IBM and the proposed GA are not statistically significant. This could be explained by the fact that even when considering the `Median` metric in Table 3, both approaches have similar values, while the `MAD` difference is very low.

Overall, the GA demonstrates a scalable efficiency by outperforming IBM on benchmarks of sizes 7-65 qubits and entanglement of 25%-100%. Both IBM and the GA have similar results principally on 7-qubits machines using 25% and 50% entanglement. This could be explained by one or a combination of several factors related to each step of the tailoring pipeline (see Figure 5). One could quickly enumerate some of them such as `(I)` those machines have the same basis gates (see Table 1), which might affect the decomposition and gate translation phases and ultimately the circuit tailoring. `(II)` These benchmarks are the smallest machines using the smallest entanglement (25%) which reduces the problem to ${}_2P_7$ possible permutations of qubits configurations to be explored, which is easily solvable. `(III)` Unlikely, but possibly, these machines have simple and increasingly-ordered physical qubits' connectivity, which might ease a trivial i^{th}-virtual to i^{th}-physical qubit initialisation. This could also ease the mapping step later. `(IV)` The machines might have a calibration (e.g. gate/qubits errors/fidelity) that makes the tailoring easier. The similarity of the results obtained by both approaches in benchmarks of 16 qubits emphasises the hypothesis that such likeness between the GA and IBM results is due to the low complexity of instances. Although, the similarity of the results when tackling the 20-qubits `LegacyTokyo` machine suggests that other(s) factor(s) are at the origin of this resemblance. Nonetheless, to provide a sustainable explanation for this, further low-level monitoring of the tailoring process would be needed (e.g. computing the circuit depth at each tailoring step), as well as a QIP fitness landscape analysis or a GA search trajectory (e.g. using [12]).

Figures 8(a)-(d) display the `Median` (represented by a line) and `MAD` (indicated by a cloud) of the circuit depth given by the best, worst solution as well as all the population throughout 30 executions. This set of figures is made for the largest machine being studied (i.e. 65-qubits `IBMQ Manhattan`) since it was found to be a good representative of the remaining benchmarks. These figures point two pieces of information; `(I)` as the search process evolves (i.e. iterations pass by), the GA's population converges to solutions with the same or close circuit depth. This could be linked to the small number of individuals the population is composed of, which makes crossover to be applied to a restricted set of QIP configurations, reducing diversity. `(II)` We can also observe that as the percentage of entanglement decreases, the convergence speed increases as well. This could be explained by the fact that the entanglement percentage controls the size of the problem to be solved. Thus, the lower the entanglement, the simpler the problem, the quicker the GA solves it, the faster the best individuals' QIP pattern take over the population. For a more concrete and precise justification, concepts such as the takeover time and the growth curve in the GA could be analysed (e.g. [4]).

Another fact is that the proposed GA is slower than the IBM technique. This could be explained by the straightforward/lightweight qubits' initialisation implemented in IBM, which stands in checking two alternative scenarios: it verifies if there is a sub-graph that satisfies all the 2-qubit gates perfectly; if not, it looks for the qubits having the smallest readout metric and Controlled-Not gate with the highest fidelity. On the other hand, the GA is population-based,

Table 3: Results using quantum machines with 7, 16, 20, 27, 53 and 65 qubits

	100% Entanglement					75% Entanglement				50% Entanglement				25% Entanglement			
Qubits	**Solver**	**IQR**	**Median**	**MAD**	**Time**	**IQR**	**Median**	**MAD**	**Time**	**IQR**	**Median**	**MAD**	**Time**	**IQR**	**Median**	**MAD**	**Time**
	IBMQ LegacyCasablanca																
	GA	0.00	14.00	0.00	1422.10	0.00	9.00	0.00	1301.79	0.00	**7.00**	0.00	748.83	0.00	**5.00**	0.00	446.10
	IBM	0.38	19.00	0.00	1.94	0.50	11.00	0.00	1.58	0.00	**7.00**	0.00	0.46	0.00	**5.00**	0.00	0.31
	IBMQ Lagos																
	GA	0.75	14.00	0.00	1399.24	0.00	9.00	0.00	1295.37	0.00	**7.00**	0.00	775.92	0.00	**5.00**	0.00	447.04
7	**IBM**	0.50	19.00	0.00	1.94	0.00	11.00	0.00	1.56	0.00	**7.00**	0.00	0.49	0.00	**5.00**	0.00	0.30
	IBMQ Jakarta																
	GA	1.00	14.00	0.00	1379.55	0.00	9.00	0.00	1277.40	0.00	**7.00**	0.00	758.73	0.00	**5.00**	0.00	449.61
	IBM	0.50	19.00	0.00	1.81	1.00	11.00	0.00	1.57	0.00	**7.00**	0.00	0.47	0.00	**5.00**	0.00	0.30
	IBMQ Casablanca																
	GA	0.00	14.00	0.00	1390.81	0.00	9.00	0.00	1288.42	0.00	**7.00**	0.00	764.66	0.00	**5.00**	0.00	437.14
	IBM	0.38	19.00	0.00	1.95	0.50	11.00	0.00	1.59	0.00	**7.00**	0.00	0.46	0.00	**5.00**	0.00	0.30
	IBMQ LegacyRueschlikon																
	GA	6.75	60.00	3.00	21001.35	3.50	42.00	2.00	13166.79	2.00	23.00	1.00	6933.74	1.00	7.00	1.00	2248.44
	IBM	4.00	93.00	2.00	16.33	3.25	54.00	1.50	7.90	1.00	32.50	0.50	5.05	0.00	7.00	0.00	0.94
	IBMQ Guadalupe																
16	**GA**	4.50	47.00	2.00	6989.00	2.88	29.00	1.00	4578.52	2.00	17.00	1.00	2524.29	0.75	**7.00**	0.00	1068.85
	IBM	2.00	68.00	1.00	8.85	2.00	49.50	1.00	5.39	1.75	29.75	0.75	3.23	0.00	**7.00**	0.00	0.45
	IBMQ Rueschlikon																
	GA	8.75	61.00	4.50	19524.83	5.75	42.00	3.00	12888.83	3.00	22.50	1.50	6924.30	2.75	7.00	1.00	2225.44
	IBM	4.50	93.00	2.25	15.11	2.00	54.00	1.00	8.12	1.50	32.75	0.75	5.05	0.00	7.00	0.00	0.96
	IBMQ Singapore																
	GA	4.00	37.00	2.00	6487.53	2.00	26.00	1.00	4541.55	2.00	17.00	1.00	2878.81	0.00	7.00	0.00	1247.96
	IBM	1.50	55.00	1.00	7.03	2.25	38.25	1.25	5.01	0.00	23.00	0.00	3.13	0.88	9.00	0.00	1.55
	IBMQ Poughkeepsie																
	GA	6.75	46.50	3.50	8452.16	2.00	32.00	1.00	6394.80	2.50	18.00	1.00	3949.23	1.00	7.00	0.00	1906.64
20	**IBM**	2.00	65.00	1.00	8.95	1.00	43.50	0.50	6.29	1.50	26.00	1.00	3.91	0.50	10.00	0.50	2.08
	IBMQ LegacyTokyo																
	GA	1.75	27.00	1.00	5180.09	2.00	20.00	1.00	4085.51	1.00	12.50	0.50	2877.54	0.00	**6.00**	0.00	1670.64
	IBM	1.88	38.50	0.50	11.54	0.88	28.00	0.50	8.70	0.50	18.00	0.00	5.57	0.00	**6.00**	0.00	1.06
	IBMQ LegacySingapore																
	GA	3.00	36.50	1.50	6432.36	2.00	27.00	1.00	4554.24	1.00	17.00	1.00	2705.60	0.00	7.00	0.00	1220.11
	IBM	1.00	55.25	0.75	7.04	1.38	38.50	0.50	4.93	1.00	23.00	0.50	3.01	0.50	9.00	0.00	1.51
	IBMQ Toronto																
	GA	4.00	66.00	2.00	14755.84	4.00	48.00	2.25	10303.56	4.00	33.50	1.50	6736.06	2.00	14.00	1.00	2691.49
	IBM	3.25	120.25	1.75	15.40	3.25	82.50	1.50	10.25	2.50	52.50	1.25	6.31	2.25	22.00	1.00	3.09
	IBMQ Sydney																
	GA	6.13	64.00	2.00	14479.17	5.00	49.00	2.00	10527.43	4.00	35.00	1.50	6641.84	1.75	13.00	0.00	2608.58
27	**IBM**	3.38	120.50	1.50	15.60	2.38	83.75	0.75	10.36	2.00	53.00	1.00	6.37	1.00	21.25	0.75	2.99
	IBMQ Paris																
	GA	6.75	64.50	3.50	14224.01	4.75	48.00	2.00	10464.67	4.00	34.00	2.00	6625.23	3.00	14.00	1.00	2574.66
	IBM	2.88	120.00	1.25	15.64	2.00	83.00	1.00	10.40	2.00	52.00	1.00	6.28	2.50	21.50	1.50	2.90
	IBMQ Mumbai																
	GA	5.00	65.00	3.00	14430.26	4.00	49.00	2.00	10171.78	4.75	33.50	2.50	6723.36	2.00	14.00	1.00	2695.61
	IBM	3.38	120.00	1.50	15.56	2.00	83.00	1.00	10.24	2.50	52.50	1.50	6.25	1.88	22.25	1.00	3.02
	IBMQ Rochester																
	GA	6.75	153.00	3.75	45670.98	5.75	104.00	3.00	28129.64	4.75	65.00	2.00	17572.91	7.00	32.50	3.50	7717.71
53	**IBM**	5.25	219.50	2.25	43.11	3.38	157.50	1.50	28.94	4.00	94.00	1.50	16.81	2.75	41.25	1.25	7.27
	IBMQ LegacyRochester																
	GA	9.50	152.25	5.25	44258.61	4.00	103.50	2.50	28259.78	6.75	64.00	3.00	17183.83	3.75	33.00	2.00	7680.75
	IBM	5.00	220.75	2.75	42.75	3.25	158.50	1.5	28.72	3.88	93.75	1.50	16.98	2.00	41.00	1.00	7.33
	IBMQ Manhattan																
	GA	7.00	187.75	4.00	69401.99	8.25	137.00	4.00	39539.34	5.75	83.50	2.50	29274.50	3.75	42.00	2.00	12049.30
65	**IBM**	7.38	321.00	4.00	82.97	4.25	231.50	2.50	42.97	3.00	143.50	2.00	34.97	2.38	64.00	1.25	11.74
	IBMQ Brooklyn																
	GA	8.50	187.50	5.25	68670.28	8.75	139.00	4.75	39590.89	5.00	85.00	3.50	29429.46	4.75	41.00	2.00	11600.49
	IBM	6.88	321.00	3.00	82.92	3.00	230.25	1.75	42.90	4.88	143.25	2.25	35.11	2.88	63.50	1.25	11.75

which requires executing the same experimental protocol that is done one time in the IBM routine on each of the individuals in the population. This increases the computation time by n times, where n is the population's size. Further execution time improvements can be done by including the problem properties to design light-weight heuristics, applying a constructive technique, etc. However, execution time is not the main focus of this work and we defer its improvement to future work.

Now, when looking at Figures 9(a)-(d), it can be seen that the best topology of qubits' initialisation for the 65-qubits `IBMQ Manhattan` achieved by the IBM routine is more densely connected, while the one provided by the GA is a bit scattered (especially in 25% entanglement). This would suggest that the IBM routine will yield better circuit-depth compression than the GA. Indeed, the IBM topology would suppose that during the mapping step (see Figure 5), it would require fewer `SWAP` gates to fit the circuit's 2-qubit gates interactions into the machine's topology and ultimately obtain a more compact circuit. However, when looking at the circuit depth achieved by IBM (see Table 3), the opposite scenario happened. Figure 8(a) tends to support such remark since in the case of 100%-entanglement circuits ($d == m$) on a 65-qubits machine, the GA's initial random population seem to attain a lower circuit depth than the one provided by the IBM routine. The GA's qubits' initialisation in that case is dense and with no gaps. This contradictory relationship between the qubits' initial topology and the circuit depth could be explained by the hypothesis that the current IBM tailoring pipeline considers two (or more) contradictory criteria that require contrasting initial qubits' assignment (i.e. dense vs scattered). In other words, a first tailoring step(s) (e.g. mapping) would require a dense and connected qubits' initialisation in order to minimise the swapping, and later in a more advanced tailoring phase(s), another step(s) would require a sparse qubits' layout to fulfill the second criterion(a) (e.g. gates

fidelity), which will exacerbate the circuit's depth increase. To make firm claims, further experimentation and software analysis on the IBM tailoring pipeline and Qiskit [7] are needed.

Table 4: Colored: compression gain, neutral: Mann-Whitney U test

Qubits	Entanglement Percentage 100%	75%	50%	25%	Qubits	Entanglement Percentage 100%	75%	50%	25%
7	IBMQ LegacyCasablanca				20	IBMQ Singapore			
	26.32%	18.18%	0.00%	0.00%		32.73%	32.03%	26.09%	22.22%
	1.46E-12	5.04E-13	1.00	1.00		2.66E-11	2.53E-11	1.44E-11	3.04E-12
	IBMQ Lagos					IBMQ Poughkeepsie			
	26.32%	18.18%	0.00%	0.00%		28.46%	26.44%	30.77%	30.00%
	5.06E-12	1.57E-13	1.00	1.00		2.70E-11	2.45E-11	2.37E-11	4.25E-11
	IBMQ Jakarta					IBMQ LegacyTokyo			
	26.32%	18.18%	0.00%	0.00%		29.87%	28.57%	30.56%	0.00%
	6.65E-12	3.99E-13	1.00	1.00		2.43E-11	2.22E-11	1.25E-11	0.081
	IBMQ Casablanca					IBMQ LegacySingapore			
	26.32%	18.18%	0.00%	0.00%		33.94%	29.87%	26.09%	22.22%
	4.74E-12	3.16E-13	1.00	1.00		2.61E-11	2.55E-11	1.80E-11	1.64E-12
16	IBMQ LegacyRueschlikon				27	IBMQ Toronto			
						45.11%	41.82%	36.19%	36.36%
	35.48%	22.22%	29.23%	0.00%		2.87E-11	2.83E-11	2.71E-11	2.36E-11
	2.90E-11	2.81E-11	3.11E-11	0.40		IBMQ Sydney			
	IBMQ Guadalupe					46.89%	41.49%	33.96%	38.82%
						2.92E-11	2.76E-11	2.68E-11	1.63E-11
	30.88%	41.41%	42.86%	0.00%		IBMQ Paris			
	2.78E-11	2.69E-11	2.04E-11	2.69E-2		46.25%	42.17%	34.62%	34.88%
	IBMQ Rueschlikon					2.80E-11	2.84E-11	2.70E-11	2.38E-11
	34.41%	22.22%	31.30%	0.00%		IBMQ Mumbai			
						45.83%	40.96%	36.19%	37.08%
	2.94E-11	3.62E-11	2.48E-11	0.43		2.86E-11	2.85E-11	2.83E-11	2.53E-11
53	IBMQ Rochester				65	IBMQ Manhattan			
	30.30%	33.97%	30.85%	21.21%		41.51%	40.82%	41.81%	34.38%
	2.96E-11	2.91E-11	2.84E-11	1.36E-10		2.97E-11	2.90E-11	2.92E-11	2.80E-11
	IBMQ LegacyRochester					IBMQ Brooklyn			
	31.03%	34.70%	31.73%	19.51%		41.59%	39.63%	40.66%	35.43%
	2.95E-11	2.89E-11	2.88E-11	2.65E-11		2.94E-11	2.95E-11	2.85E-11	2.84E-11

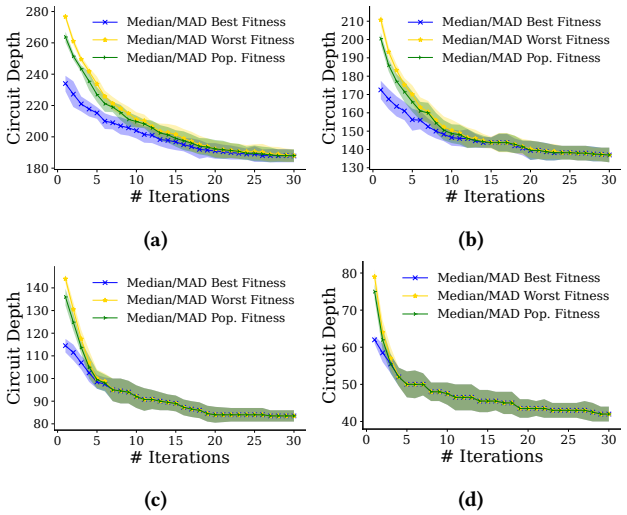

Figure 8: Circuit depth evolution when tackling 65-qubits `IBMQ Manhattan`: (a)-(d) `Median` and `MAD` of the best, worst and population fitness at each iteration throughout 30 executions and this for GHZ circuits of 100, 75, 50 and 25% entanglement

5 CONCLUSIONS AND FUTURE WORKS

As a step towards more efficient quantum computation in NISQ-DGVM machines, this work explores the add-in that evolutionary computation can provide in enhancing the quantum circuit tailoring process. This has been done by tackling the qubits' initialisation task as a mono-objective optimisation problem using a genetic algorithm. Diverse experimentation has been performed to assess the efficiency of the proposal, and this by tackling 19 real IBM quantum machines with 7 to 65 qubits and 9 different qubit topologies. Also, 76 GHZ circuits of sizes 7 to 65 qubits and 25%-100% of entanglement were generated and studied. Extensive comparisons using standard metrics and null-hypothesis tests were made against the IBM qubit initialiser currently implemented in real IBM quantum machines. Results demonstrated that the GA outperforms IBM in 64 instances while achieving similar results in 10 ones. Also, the GA could attain an average circuit-compression gain of 18-46% which could allow executing the circuits of almost twice depth that the one that would be feasible using IBM routine.

As future research lines, it is planned to devise advanced and lightweight/fast solvers combining heuristic and linkage-learning evolutionary algorithms. It is also intended to apply the experiments on larger realistic quantum machines provided by different manufacturers. Finally, efforts will be made to test the proposal with other tailoring pipelines such as the one of Google via `Circ`.

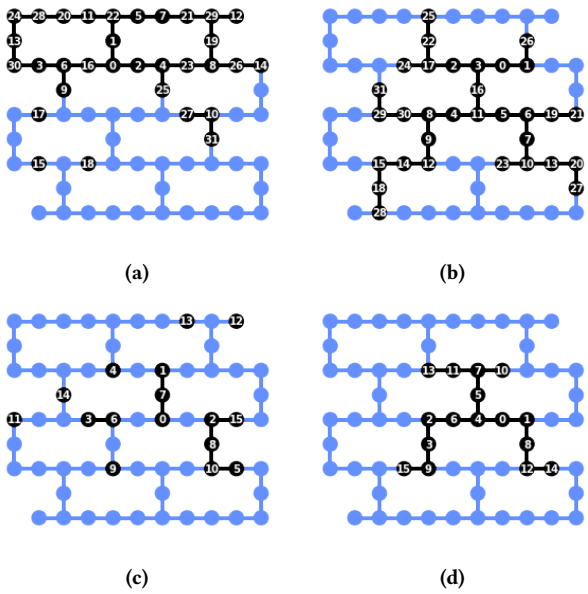

Figure 9: Best QIP solution for `Manhattan` machine using circuits of 50% and 25% entanglement: (a, c) GA and (b, d) IBM.

ACKNOWLEDGMENTS

This research is partially funded by the Universidad de Málaga, Consejería de Economía y Conocimiento de la Junta de Andalucía and FEDER under grant number UMA18-FEDERJA-003 (PRECOG); under grant PID 2020-116727RB-I00 (HUmove) funded by MCIN/AEI/ 10.13039/501100011033; and TAILOR ICT-48 Network (No 952215) funded by EU Horizon 2020 research and innovation programme. The views expressed are purely those of the writer and may not in any circumstances be regarded as stating an official position of the European Commission.

REFERENCES

[1] Giovanni Acampora and Roberto Schiattarella. 2021. Deep neural networks for quantum circuit mapping. *Neural Computing and Applications* 33 (2021), 13723–13743. https://doi.org/10.1007/s00521-021-06009-3

[2] Mahabubul Alam, Abdullah Ash-Saki, and Swaroop Ghosh. 2020. Circuit Compilation Methodologies for Quantum Approximate Optimization Algorithm. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 215–228. https://doi.org/10.1109/MICRO50266.2020.00029

[3] Zakaria Abdelmoiz Dahi and Enrique Alba. 2022. Metaheuristics on quantum computers: Inspiration, simulation and real execution. *Future Generation Computer Systems* 130 (2022), 164–180. https://doi.org/10.1016/j.future.2021.12.015

[4] Zakaria Abdelmoiz Dahi, Enrique Alba, and Gabriel Luque. 2022. A takeover time-driven adaptive evolutionary algorithm for mobile user tracking in pre-5G cellular networks. *Applied Soft Computing* 116 (2022), 107992. https://doi.org/10.1016/j.asoc.2021.107992

[5] Prasanna Date and Thomas Potok. 2021. Adiabatic quantum linear regression. *Scientific Reports* 1 (2021), 2045–2322. https://doi.org/10.1038/s41598-021-01445-6

[6] Frank Arute et al. 2019. Quantum Supremacy using a Programmable Superconducting Processor. *Nature* 574 (2019), 505–510. https://www.nature.com/articles/s41586-019-1666-5

[7] MD SAJID ANIS et al. 2021. Qiskit: An Open-source Framework for Quantum Computing. https://doi.org/10.5281/zenodo.2573505

[8] Blake Gerard and Martin Kong. 2021. Exploring Affine Abstractions for Qubit Mapping. In *2021 IEEE/ACM Second International Workshop on Quantum Computing Software (QCS)*. 43–54. https://doi.org/10.1109/QCS54837.2021.00009

[9] Blake Gerard and Martin Kong. 2021. String Abstractions for Qubit Mapping. *CoRR* abs/2111.03716 (2021). arXiv:2111.03716 https://arxiv.org/abs/2111.03716

[10] Yong (Alexander) Liu, Xin (Lucy) Liu, Fang (Nancy) Li, Haohuan Fu, Yuling Yang, Jiawei Song, Pengpeng Zhao, Zhen Wang, Dajia Peng, Huarong Chen, Chu Guo, Heliang Huang, Wenzhao Wu, and Dexun Chen. 2021. Closing the "Quantum Supremacy" Gap: Achieving Real-Time Simulation of a Random Quantum Circuit Using a New Sunway Supercomputer. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 3, 12 pages. https://doi.org/10.1145/3458817.3487399

[11] Fingerhuth Mark, Babej Tomáš, and Wittek Peter. 2018. Open source software in quantum computing. *PLoS ONE* 13 (2018). https://doi.org/10.1371/journal.pone.0208561

[12] Gabriela Ochoa, Katherine M. Malan, and Christian Blum. 2021. Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics. *Applied Soft Computing* 109 (2021), 107492. https://doi.org/10.1016/j.asoc.2021.107492

[13] Alexandru Paler. 2017. On the Influence of Initial Qubit Placement During NISQ Circuit Compilation. , 207–217 pages.

[14] Edwin Pednault, John A. Gunnels, Giacomo Nannicini, Lior Horesh, and Robert Wisnieff. 2019. Leveraging Secondary Storage to Simulate Deep 54-qubit Sycamore Circuits. arXiv:1910.09534 [quant-ph]

[15] Michal W. Przewozniczek and Marcin M. Komarnicki. 2020. *Empirical Linkage Learning*. Association for Computing Machinery, New York, NY, USA, 23–24. https://doi.org/10.1145/3377929.3397771

[16] Michal W. Przewozniczek, Marcin M. Komarnicki, Peter A. N. Bosman, Dirk Thierens, Bartosz Frej, and Ngoc Hoang Luong. 2021. *Hybrid Linkage Learning for Permutation Optimization with Gene-Pool Optimal Mixing Evolutionary Algorithms*. Association for Computing Machinery, New York, NY, USA, 1442–1450. https://doi.org/10.1145/3449726.3463152

[17] Marcos Yukio Siraichi, Vinícius Fernandes dos Santos, Caroline Collange, and Fernando Magno Quintao Pereira. 2018. Qubit Allocation. In *Proceedings of the 2018 International Symposium on Code Generation and Optimization* (Vienna, Austria) *(CGO 2018)*. Association for Computing Machinery, New York, NY, USA, 113–125. https://doi.org/10.1145/3168822

[18] Adam M. Zielinski, Marcin M. Komarnicki, and Michal W. Przewozniczek. 2019. Parameter-Less Population Pyramid with Automatic Feedback. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (Prague, Czech Republic) *(GECCO '19)*. Association for Computing Machinery, New York, NY, USA, 312–313. https://doi.org/10.1145/3319619.3322052