A Variational Quantum Algorithm for the Permutation Flow Shop Scheduling Problem

Marco Baioletti*
marco.baioletti@unipg.it
University of Perugia, Department of Mathematics and Computer Science
Perugia, Italy

Fabrizio Fagiolo*
fabrizio.fagiolo@istc.cnr.it
National Research Council (CNR), Institute of Cognitive Sciences and Technologies (ISTC)
Rome, Italy

Angelo Oddi*
angelo.oddi@istc.cnr.it
National Research Council (CNR), Institute of Cognitive Sciences and Technologies (ISTC)
Rome, Italy

Riccardo Rasconi*
riccardo.rasconi@istc.cnr.it
National Research Council (CNR), Institute of Cognitive Sciences and Technologies (ISTC)
Rome, Italy

Abstract

In this paper we present a Variational Quantum Algorithm (VQA) for the Permutation Flow Shop Scheduling Problem (PFSSP). This scheduling problem is challenging to be solved with a quantum algorithm because the objective function is quite difficult to be expressed by means of a Hamiltonian, and a large number of qubits is generally required. For this reason, we propose to use a variational approach where the solution is represented with a limited number of qubits and the objective function can be computed without explicitly synthesizing a Hamiltonian operator. The algorithm has been implemented and tested on a simulator; the preliminary experimental results demonstrate that our solution could represent a viable alternative to solve the PFSSP on real quantum devices.

ACM Reference Format:
Marco Baioletti, Fabrizio Fagiolo, Angelo Oddi, and Riccardo Rasconi. 2025. A Variational Quantum Algorithm for the Permutation Flow Shop Scheduling Problem. In *Genetic and Evolutionary Computation Conference (GECCO '25 Companion), July 14–18, 2025, Malaga, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3712255.3734312

1 Introduction

One of the most important applications of Quantum Computing is optimization, which can use two approaches: employing Quantum Annealers, which are specifically designed for this purpose, or implementing some optimization algorithms on gate-based quantum computers.

In this paper we only focus on the second approach. The two most popular algorithms for optimizations are the Variational Quantum Eigensolver (VQE) and the Quantum Approximate Optimization Algorithm (QAOA). Both algorithms use a variational approach, in which a parametrized quantum circuit is optimized by a classical continuous optimization procedure. Moreover, both algorithms are well suited for near-term quantum computers because they generally require a limited number of qubits and employ shallow circuits composed only of unary and binary quantum gates.

VQE is a hybrid quantum-classical algorithm used to estimate the minimum eigenvalue of a quantum Hamiltonian, which corresponds to finding the fundamental state energy of a quantum system. Originally introduced by Peruzzo et al. [19], VQE combines classical optimization techniques with quantum state preparation, using parameterized quantum circuits known as *ansatzes*. Initially applied to quantum chemistry problems [15], VQE has ever since been used to address a variety of problems, including quantum magnetism, molecular simulations and combinatorial optimization [7, 15]. The main goal of the VQE algorithm is to optimize the θ parameters of the ansatz, so as to minimize the expectation value of the Hamiltonian $\hat{H}$:

$$E(\theta) = \langle \psi(\theta) | \hat{H} | \psi(\theta) \rangle$$

This hybrid quantum-classical structure makes the VQE particularly suitable for the current Noisy Intermediate-Scale Quantum (NISQ) devices, characterized by relatively small numbers of qubits, limited coherence times, and moderate error rates [5, 20].

Similarly, the Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical approach introduced by Farhi, Goldstone, and Gutmann to find solutions for combinatorial optimization problems using quantum computers [10]. QAOA leverages principles from adiabatic quantum computing [2], utilizing a structured ansatz that alternates between two distinct Hamiltonians: a problem-specific Hamiltonian ($\hat{H}_C$), encoding the objective function of the problem, and a mixing Hamiltonian ($\hat{H}_B$), responsible of exploring the solution space of the problem. Formally, the quantum state prepared by the QAOA ansatz can be described as:

$$|\gamma, \beta\rangle = U_B(\beta_p)\, U_C(\gamma_p) \ldots U_B(\beta_1)\, U_C(\gamma_1)\, |+\rangle^{\otimes n},$$

*All authors contributed equally to this research.

GECCO '25 Companion, July 14–18, 2025, Malaga, Spain

ACM ISBN 979-8-4007-1464-1/2025/07
https://doi.org/10.1145/3712255.3734312

where the initial state $(|+\rangle^{\otimes n})$ represents a symmetric superposition of all (2^n) computational basis states in a system composed of n qubits. The operators $(U_C(\gamma_k))$ and $(U_B(\beta_k))$ are defined as:

$$U_C(\gamma_k) = e^{-i\,\gamma_k \hat{H}_C}, \quad U_B(\beta_k) = e^{-i\,\beta_k \hat{H}_B}, \quad \hat{H}_B = \sum_{j=1}^{n} X_j$$

where X_j denotes the Pauli-X operator acting on the j-th qubit. The variational parameters γ_k and β_k are classically optimized to the aim of minimizing the expected value of the problem Hamiltonian $\hat{H}_C$. Differently from VQE, we observe that QAOA specifically utilizes an ansatz structure that is more closely related to the specific problem's Hamiltonian, enabling targeted exploration of combinatorial solution spaces.

The Permutation Flowshop Scheduling Problem (PFSSP) is an important combinatorial optimization problem in operations research, due to both its theoretical complexity and practical significance in production and logistics [11]. In PFSSP, a set $J = \{1, \ldots, n\}$ of n jobs must be sequentially processed by a set $M = \{1, \ldots, m\}$ of m machines in a fixed order, such that: (i) each job is executed on all m machines; (ii) the processing sequence of jobs remains identical across all machines. A job $j \in J$ assigned to a machine $i \in M$ has a known processing time $p_{i,j}$, and the overall goal is to find an optimal permutation $\pi = \langle \pi(1), \ldots, \pi(n) \rangle$ of jobs that minimizes a determined performance criterion. In this study, we wish to find the job permutation π that minimizes the overall *makespan*, formally defined as the completion time of the last job on the final machine, expressed as:

$$f(\pi) = c(m, \pi(n))) \tag{1}$$

where the completion time $c(i, \pi(k))$ of job $\pi(k)$ on machine i is recursively defined as:

$$c(i, \pi(k)) = p_{i,\pi(k)} + \max\{c(i-1, \pi(k)),\ c(i, \pi(k-1))\} \tag{2}$$

PFSSP with makespan minimization is NP-hard problem, therefore making heuristic and metaheuristic approaches popular choices in its resolution [11]. Its inherent complexity continues to inspire the exploration of innovative strategies such as quantum computing techniques to develop more efficient and effective solution methods.

An important point of decision when approaching PFSSP's resolution using QAOA or VQE is how to represent a solution of the problem. Since it is necessary to express the makespan (or other objective functions) as a Hamiltonian, a very large number of qubits is usually required. An interesting example is provided in [6], where a solution is represented with the start times of each operation to be scheduled; this encoding seems quite expensive as the number of qubits is not polynomial with respect to the size of the problem, being it proportional to the values of the processing times. For instance, the smallest instance handled in [6] is a 2×2 (2 jobs and 2 machines) and requires at least 70 qubits to be represented, while for the largest instance (7×5) 1474 qubits are needed, which is over the theoretical possibility of current gate-based quantum computers.

It is evident that any other alternative formulation of the problem in terms of QUBO (or a Ising model) would require a large number of qubits. However it is important to note that a solution of the PFSSP, being just a permutation of jobs, can be encoded in several ways; for instance, if a one-hot encoding approach is used, only $O(n^2)$ qubits are necessary.

The approach used in this paper indeed exploits the fact that a solution of the PFSSP can be encoded in an even more concise way, using only $O(n \log n)$ qubits. When the average value of the objective function has to be computed, an empirical estimation is employed, consisting in: (i) extracting N solutions, (ii) computing the objective function for each solution, and finally (iii) taking the average. Therefore, it is possible to create an ansatz for the PFSSP that produces only valid solutions, and use the variational quantum scheme to optimize the parameters. The objective function of the optimization process is provided by the empirical average computation.

The paper is organized as follows. Section 2 describes the related work. Section 3 introduces the scheme of the approach proposed in this paper. Section 4 describes and comments the obtained experimental results. Section 5 ends the paper by providing some concluding remarks as well as possible future lines of research.

2 Related work

The PFSSP has traditionally been tackled using classical approaches, ranging from heuristics to advanced metaheuristics, due to its NP-hard nature [22].

In recent years, quantum approaches have gained attention for scheduling problems, particularly through Quantum Annealing, where Quadratic Unconstrained Binary Optimization (QUBO) or Ising formulations allow scheduling constraints to be mapped onto physical qubits [1, 9, 26]. While most of these studies focus on the Job Shop Scheduling Problem (JSSP) and Flow Shop Scheduling Problem (FSSP) rather than PFSSP, they demonstrate the feasibility of quantum-based techniques for multiple instances.

Alongside Quantum Annealing, gate-based quantum algorithms have also started to be explored for scheduling applications. The Quantum Approximate Optimization Algorithm (QAOA), originally introduced by Farhi et al. [10], has indeed been applied to scheduling tasks [17]. However, the practical effectiveness of QAOA remains limited to very small problem instances due to the currently existing quantum hardware constraints. Hadfield et al. [12] significantly expanded this original framework, introducing the Quantum Alternating Operator Ansatz, which generalizes QAOA to systematically encode and solve constrained combinatorial optimization problems, including several scheduling problems. Despite its theoretical attractiveness and flexibility, the Quantum Alternating Operator Ansatz often requires a large number of qubits as well as multi-qubit mixing gates, making it challenging to test on the currently available quantum hardware, especially when considering problem instances of medium.large size [12].

More recently, the Variational Quantum Eigensolver (VQE), originally developed for quantum chemistry applications [19], has attracted attention as another gate-based quantum algorithm for combinatorial optimization. VQE has already shown promising results in addressing scheduling problems such as the Job-Shop Scheduling Problem (JSSP) [3]. However, its application to the Flow Shop Scheduling Problem (FSSP) has received much less attention, and most of the current studies are still based on quantum annealers rather than gate-based quantum computers [9]. A recent

contribution by Schmid et al. [23] introduces an efficient encoding strategy for Job-Shop Scheduling Problems, which significantly reduces the number of required qubits and enhances the performance of VQE-based approaches.

To date, no experimental study has successfully demonstrated the application of VQE specifically for the PFSSP. Given the growing interest in hybrid quantum-classical computing methodologies, our work aims to fill this gap by applying VQE to the PFSSP, and rigorously evaluating its performance on standard instances of the Taillard [24] benchmark.

3 Proposed solution

We propose a new approach for solving the PFSSP with a simple Variational Quantum Algorithm: a parametric quantum circuit $\mathcal{A}$ (the ansatz) is used to produce solutions of the problem and a classical optimization procedure is employed to adjust the parameters of $\mathcal{A}$ in order to improve the quality of the produced solutions.

The main idea behind our approach is that it is possible to represent a PFSSP solution, i.e., a permutation of jobs, in a very compact fashion by means of the Lehmer's code [18]. The Lehmer's code of a permutation π of n objects is represented by an integer number L_π between 0 and $n! - 1$. Both the encoding procedure, which returns L_π from π, and the decoding procedure, which returns π from L_π are fast and take $O(n)$ steps. In particular, the decoding procedure needed in our approach is depicted in Algorithm 1.

Algorithm 1 Lehmer decoding procedure

```
1:  function LehmerDecoding(L)
2:      R ← [1, 2, ..., n]
3:      for i ← n downto 1 do
4:          j ← 1 + (L mod i)
5:          π(n + 1 − i) ← R_j
6:          remove R_j from R
7:          L ← ⌊L/i⌋
8:      end for
9:      return π
10: end function
```

The number of qubits required by this representation is then $\lceil \log_2(n!) \rceil$, which by Stirling's formula, is $O(n \log n)$. For example, in a problem instance with $n = 4$ jobs, the number of permutations is 24; since $\log_2(24) \approx 4.585$, 5 qubits are sufficient to solve the PFSSP instance. Note that this encoding is independent from both the number of machines m and the jobs' processing times.

The procedure proposed in this paper is described in Algorithm 2, where $\mathcal{P}$ contains the data of the PFSSP to be solved (n, m and the matrix p_{ij}) and $\mathcal{A}$ is the ansatz. We denote by $\mathcal{A}.np$ the number of parameters necessary to express $\mathcal{A}$.

Algorithm 2 The Variational Quantum Algorithm proposed

```
1:  function VQA(P, A)
2:      for i ← 1 to 100 do
3:          η^(i) ←Sample(A.np)
4:          E_i ←ComputeAverageEnergy(A, η^(i))
5:      end for
6:      Sort the sequence of all (η^(i), E_i) with respect to E_i
7:      for i ← 1 to 10 do
8:          η̄^(i) ← Optimize(A, η^(i))
9:          Ē_i ←ComputeAverageEnergy(A, η̄^(i))
10:     end for
11:     return (η̄^(i), Ē_i) with the smallest value of Ē_i
12: end function
```

The function Sample randomly generates a vector that contains the values of all the parameters of $\mathcal{A}$. The function ComputeAverageEnergy computes the average of the solution makespans produced by $\mathcal{A}$ when the parameters have the value $\eta^{(i)}$. The ansatz $\mathcal{A}$ with parameter values $\eta^{(i)}$ is run N times (usually $N = 1024$), obtaining N bitstrings $b_1, \ldots, b_N$. Each bitstring b_i is interpreted as an integer number and then decoded to obtain the permutation π_i using the decoding procedure in Alg. (1). Note that the procedure returns a permutation even if the number exceeds $n! - 1$. Using the formulas (1) and (2) the makespan $f(\pi_i)$ is computed. Finally, the makespan returned by the function is the average of all the values $f(\pi_i)$.

Note that this step replaces the computation of the Hamiltonian as done in the usual VQE or QAOA approaches.

The Optimize procedure is then used to classically optimize the parameters of $\mathcal{A}$, starting from the chosen guess ($\eta^{(i)}$), with respect to the function ComputeAverageEnergy. Initially, the **SPSA** (Simultaneous Perturbation Stochastic Approximation) algorithm is applied, which performs a global search (up to 1000 iterations) [14]. Next, the results are refined with **COBYLA** (Constrained Optimization By Linear Approximations), also limited to 1000 iterations, for a more accurate local search [13]. Iteratively repeating these steps, the bitstring distribution tends to shift toward permutations characterized by a lower makespan, thus reducing the average cost.

The ansatz $\mathcal{A}$ chosen for this paper is the *RealAmplitudes* [25] because of the small number of required parameters. The *RealAmplitudes* ansatz is composed by r repetitions of two blocks (see Figure 1). In the first block, a rotation gate R_Y is applied to each qubit, while in the second block a sequence of CNOT is applied to some pairs of qubits. For this work, we have chosen the ansatz's standard form (reverse_linear) in which the CNOT is applied to the pair $(q-2, q-1), (q-3, q-2), \ldots, (0, 1)$, where q is the number of qubits. The last block of the circuit is composed by R_Y rotations applied on each qubit.

Since the only parametrized gates are the rotations, the total number of parameters required is $(r + 1)q$. In our experiments we set $r = 1$.

Looking at Algorithm 2, the VQA function starts by generating **100 random vectors** of angular parameters in the range $[0, 2\pi]$ and computing their individual energy as the average of the makespan. Subsequently, **the best 10 vectors**, i.e., those with the lowest energy values, are used as starting points for the Optimize procedure. The

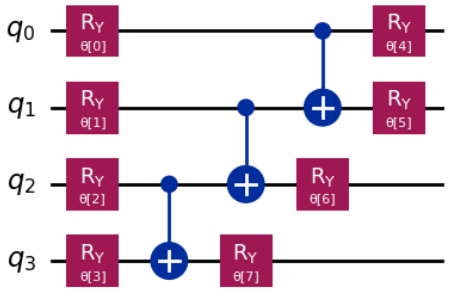

Figure 1: *RealAmplitudes* ansatz with 4 qubit.

procedure terminates returning the best post-optimization vector of parameters.

4 Experiments

In this section we describe the experiments performed to solve the PFSSP problem with our VQE approach.

4.1 Problem setup

We selected some instances from the benchmark collection proposed by Taillard [24]. Each instance is composed of: (i) the number of jobs n, (ii) the number of machines m, (iii) the processing time matrix p_{ij}.

The Taillard benchmark contains instances whose number of jobs n is between 20 and 500 and the number of machines m is between 5 and 20. Solving large instances using a quantum approach would require a very large number of qubits, even with our compact representation, making it impractical with current quantum hardware or simulators.

Therefore, we have taken all the 10 instances *tai20_5_X.tsp* and from each of them we have extracted the processing times of the first n jobs for $n = 2, 3, 4, 5, 6, 7$, keeping the number of machines fixed at $m = 5$. This procedure thus generated 6×10 sub-instances.

In this way, we can also calculate the optimal **makespan** by a complete enumeration of all permutations, which can be used to compare the performance of our VQA procedure with the known optimal value.

Table 1 summarizes how many qubits and ansatz parameters are required, depending on the instance size.

4.2 Experimental Setup

For each instance *tai20_5_X.tsp* and each value of n, we have performed **5 independent runs** of the VQE procedure. In each run we have stored the final average of the makespan $\bar{M}$, i.e., the average of all the makespan values found by the best optimized circuit in 1024 shots, as well as the number of bit-strings N_{opt} whose makespan is equal to the optimal value (e_{min}) found by the enumeration method.

Table 1: Qubit count required for different grid sizes.

Grid Size	Qubits Used	ansatz parameters
2×5	1	2
3×5	3	6
4×5	5	10
5×5	7	14
6×5	10	20
7×5	13	26

4.3 Results

4.3.1 Best run results. Table 2 shows the ratio between the minimum value of $\bar{M}$ (among the 5 runs) and the known optimal makespan e_{min}, for each instance and for each number of jobs (2 to 7).

Table 2: Minimum ratio between $\bar{M}$ and e_{min} for each instance and number of jobs

Job / Instance	2	3	4	5	6	7
tai20_5_0.fsp	1.0000	1.0000	1.0000	1.0000	1.0044	1.0030
tai20_5_1.fsp	1.0000	1.0000	1.0000	1.0000	1.0008	1.0036
tai20_5_2.fsp	1.0000	1.0000	1.0000	1.0038	1.0014	1.0449
tai20_5_3.fsp	1.0000	1.0000	1.0000	1.0090	1.0002	1.0063
tai20_5_4.fsp	1.0000	1.0000	1.0000	1.0000	1.0000	1.0004
tai20_5_5.fsp	1.0000	1.0000	1.0000	1.0046	1.0007	1.0230
tai20_5_6.fsp	1.0000	1.0000	1.0000	1.0022	1.0031	1.0228
tai20_5_7.fsp	1.0000	1.0000	1.0000	1.0001	1.0170	1.0197
tai20_5_8.fsp	1.0000	1.0000	1.0000	1.0197	1.0031	1.0643
tai20_5_9.fsp	1.0000	1.0000	1.0000	1.0001	1.0038	1.0092

Values equal to 1 indicate that the best run always reached a makespan equal to the optimal value. For larger problem sizes (6 and 7 jobs), small deviations above 1 appear, indicating that in some runs, the circuit returned solutions with a slightly worse makespan than the optimum.

Table 3 gives the largest observed probability (% of samples) of obtaining the global optimum (p_{opt}) among the 5 runs, for each instance and each value of n. A value of 100% means that in at least

Table 3: Largest observed probability (%) of sampling the optimal makespan (p_{opt}) across five runs, for each instance and number of jobs

Job / Instance	2	3	4	5	6	7
tai20_5_0.fsp	100.0	100.0	100.0	100.0	99.7	99.4
tai20_5_1.fsp	100.0	100.0	100.0	100.0	99.9	99.4
tai20_5_2.fsp	100.0	100.0	100.0	99.9	99.7	0.0
tai20_5_3.fsp	100.0	100.0	100.0	100.0	100.0	0.0
tai20_5_4.fsp	100.0	100.0	100.0	100.0	100.0	99.8
tai20_5_5.fsp	100.0	100.0	100.0	99.9	99.7	98.3
tai20_5_6.fsp	100.0	100.0	100.0	100.0	99.5	0.0
tai20_5_7.fsp	100.0	100.0	100.0	100.0	99.8	0.0
tai20_5_8.fsp	100.0	100.0	100.0	100.0	100.0	0.0
tai20_5_9.fsp	100.0	100.0	100.0	100.0	99.8	0.0

one run, the circuit almost exclusively sampled optimal solutions. For $n = 2, 3, 4$ this value has been reached for all instances, while for $n = 5, 6$ the values were very close or equal to 100%. Only for

$n = 7$, we had the value 0% for 6 instances, while for the other 4 instances the value is very close to 100%.

Table 4 shows how many times (out of a possible 5) the final sampling distribution had a high probability (at least 75%) of producing the global optimum for each instance and value of n. As

Table 4: Number of runs where $N_{opt} \geq 75\%$ for each instance and value of n

Job / Instance	2	3	4	5	6	7
tai20_5_0.fsp	5	5	5	5	1	3
tai20_5_1.fsp	5	5	5	5	4	4
tai20_5_2.fsp	5	5	5	2	4	0
tai20_5_3.fsp	5	5	5	4	5	0
tai20_5_4.fsp	5	5	5	5	5	5
tai20_5_5.fsp	5	5	5	4	5	1
tai20_5_6.fsp	5	5	5	4	2	0
tai20_5_7.fsp	5	5	5	5	3	0
tai20_5_8.fsp	5	5	5	4	4	0
tai20_5_9.fsp	5	5	5	5	4	0

the tables shows, smaller problems $n \leq 4$ achieve the highest value (5), demonstrating that the circuit consistently returns high-quality solutions. As the number of jobs grows, the results become more variable and frequently fall below the maximum count, reflecting the increased difficulty of exploring larger permutation spaces.

4.3.2 Average Results. Table 5 summarizes the average ratio between $\bar{M}$ and *e_min*, across the runs. As visible, for problem sizes

Table 5: Average ratio between $\bar{M}$ and e_{min} for each instance and number of jobs

Job / Instance	2	3	4	5	6	7
tai20_5_0.fsp	1.0000	1.0000	1.0000	1.0000	1.0003	1.0002
tai20_5_1.fsp	1.0000	1.0000	1.0000	1.0000	1.0000	1.0001
tai20_5_2.fsp	1.0000	1.0000	1.0000	1.0000	1.0000	1.0335
tai20_5_3.fsp	1.0000	1.0000	1.0000	1.0000	1.0000	1.0026
tai20_5_4.fsp	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
tai20_5_5.fsp	1.0000	1.0000	1.0000	1.0000	1.0004	1.0019
tai20_5_6.fsp	1.0000	1.0000	1.0000	1.0000	1.0001	1.0079
tai20_5_7.fsp	1.0000	1.0000	1.0000	1.0000	1.0003	1.0072
tai20_5_8.fsp	1.0000	1.0000	1.0000	1.0000	1.0004	1.0478
tai20_5_9.fsp	1.0000	1.0000	1.0000	1.0000	1.0008	1.0068

up to 5 jobs, the average ratio values remain at 1, indicating that the circuit's final distribution revolve around optimal schedules. As the number of jobs rises to 6 there are some instances where the value is slightly above 1. For $n = 7$, certain instances show larger deviations (e.g., (1.0335) or (1.0478); yet, the circuit was able to produce very good quality solutions even for these instances.

It is interesting to highlight that the four tables agree on the fact that in the $n = 7$ case, the hardest instances are *tai20_5_2.fsp*, *tai20_5_3.fsp*, *tai20_5_6.fsp*, *tai20_5_7.fsp*, *tai20_5_8.fsp*, and *tai20_5_9.fsp*. Indeed, these instance correspond to the 0 values in Tables (3) and (4) and also to the largest values in Tables (2) and (5).

4.3.3 Gantt chart visualization. In this section, we present a graphical example of the optimal solutions obtained using the VQE algorithm. Specifically, we utilize a Gantt chart to visually represent the sequence and scheduling of jobs on different machines. In the Gantt chart, machines are indicated along the vertical axis (from $M1 \ldots M5$), while the horizontal axis shows the elapsed processing time. Each colored rectangle corresponds to the execution of a specific job on a given machine, with rectangle lengths proportional to actual processing times. These visualizations intuitively illustrate the optimal scheduling sequences, clearly highlighting the absence of time overlaps among the jobs scheduled on the same machine (hence proving the correctness of the returned solution) as well as the machine idle times. This chart provides a clear visualization of

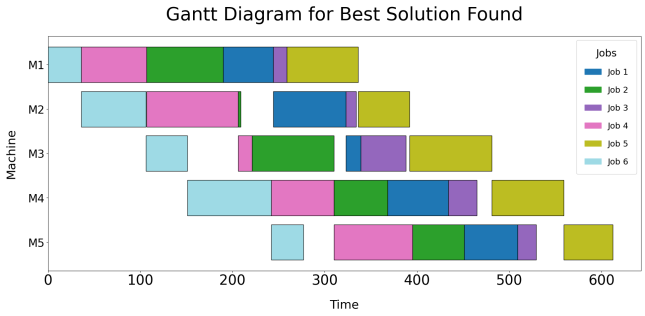

Figure 2: Gantt diagram representing the optimal solution obtained with our VQE-based approach for the PFSP with $n = 6$ jobs and $m = 5$ machines.

how the algorithm efficiently allocates job sequences to minimize the makespan, demonstrating the effectiveness of the proposed VQE methodology.

4.3.4 Energy visualization. In this section we present an example graph illustrating the convergence curve of a found solution. More precisely, Figure 3 shows the energy convergence curve for the tai20_5_0.fsp instance with $n = 6$ jobs. This graph is intended to demonstrate how the energy typically decreases and stabilizes during the optimization process, thanks to the "tandem" utilization of both SPSA and COBYLA classical optimizers. As the figure shows, the energy rapidly decreases in the early steps and stabilizes after approximately 800 iterations. This example highlights how the optimization strategy can guide the quantum circuit toward high-quality solutions.

5 Conclusions and Future Work

In this work we explored the use of the Variational Quantum Algorithm applied to the Permutation Flow Shop Scheduling Problem. We demonstrated how the adoption of a compact representation based on Lehmer's code can solve the PFSSP using a reduced number of qubits (of order $O(n \log n)$), while retaining the possibility of empirically calculating the objective function through a hybrid quantum-classical approach.

For the construction of the ansatz, we adopted the *RealAmplitudes* ansatz, whose structural simplicity and small number of parameters make it particularly suitable for execution on NISQ

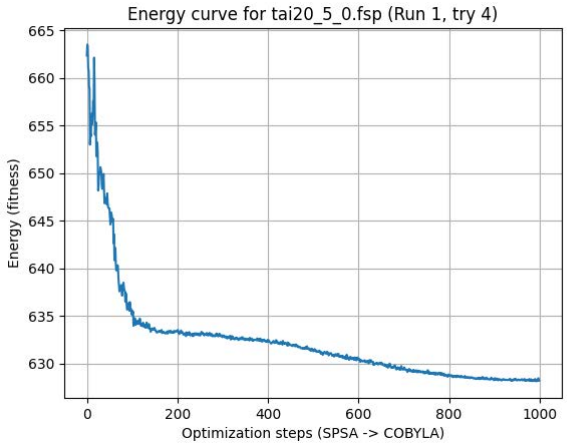

Figure 3: Energy curve for tai20_5_0.fsp with 6 jobs, showing fast convergence during SPSA and refinement via COBYLA.

quantum devices. The classical optimization phase of the ansatz parameters was carried out using a combination of the SPSA and COBYLA algorithms, showing good effectiveness in converging to optimal solutions for small problems.

As future developments, we intend to investigate other types of ansatz that are better suited to the problem and can further improve the exploratory ability of quantum algorithms on complex combinatorial search spaces. In addition, the same methodology based on compact representation using Lehmer's code could be applied to other combinatorial optimization problems characterized by permutations, such as the Traveling Salesman Problem (TSP) [8], the Minimum Spanning Tree Problem (MSTP) [16] or Linear Ordering Problem [21].

Moreover, another interesting research direction involves incorporating risk-sensitive metrics such as the Conditional Value-at-Risk (CVaR) instead of the total mean to evaluate results [4]. This approach would allow for solutions that not only optimize the expected outcome but also explicitly consider and mitigate worst-case scenarios, thus enhancing the robustness of solutions.

Finally, the proposed technique could be especially effective where objective functions are difficult or impossible to encode directly in the form of efficient Hamiltonians. In these cases, the approach based on empirical evaluation of the objective could allow the representative limitation of the objective function to be circumvented, thus opening up new application possibilities in the field of variational quantum algorithms.

Acknowledgments

M. Baioletti was partially funded by the European Union - NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem, grant ECS00000041 - VITALITY - CUP J97G22000170005.

References

[1] Riad Aggoune and Samuel Deleplanque. 2023. *A Quantum Annealing Solution to the Job Shop Scheduling Problem.* Springer, 421–428. doi:10.1007/978-3-031-37105-9_28

[2] Tameem Albash and Daniel A. Lidar. 2018. Adiabatic quantum computation. *Reviews of Modern Physics* 90, 1 (Jan. 2018). doi:10.1103/revmodphys.90.015002

[3] David Amaro, Matthias Rosenkranz, Nathan Fitzpatrick, Koji Hirano, and Mattia Fiorentini. 2022. A case study of variational quantum algorithms for a job shop scheduling problem. *EPJ Quantum Technology* 9, 1 (02 2022), 5. doi:10.1140/epjqt/s40507-022-00123-4

[4] Panagiotis Kl. Barkoutsos, Giacomo Nannicini, Anton Robert, Ivano Tavernelli, and Stefan Woerner. 2020. Improving Variational Quantum Optimization using CVaR. *Quantum* 4 (April 2020), 256. doi:10.22331/q-2020-04-20-256

[5] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek, and Alán Aspuru-Guzik. 2022. Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics* 94, 1 (Feb. 2022). doi:10.1103/revmodphys.94.015004

[6] Wojciech Bożejko, Radosław Idzikowski, Dariusz Jagielski, Ryszard Klempous, Sergii Trotskyi, Mateusz Tykierko, Mariusz Uchroński, and Mieczysław Wodecki. 2024. Hybrid Quantum Approximate Algorithms for Solving Flow Shop Scheduling Problem. In *2024 IEEE 24th International Symposium on Computational Intelligence and Informatics (CINTI).* 117–122. doi:10.1109/CINTI63048.2024.10830818

[7] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. 2021. Variational quantum algorithms. *Nature Reviews Physics* 3, 9 (Aug. 2021), 625–644. doi:10.1038/s42254-021-00348-9

[8] G. Dantzig, R. Fulkerson, and S. Johnson. 1954. Solution of a Large-Scale Traveling-Salesman Problem. *Journal of the Operations Research Society of America* 2, 4 (1954), 393–410. doi:10.1287/opre.2.4.393 arXiv:https://doi.org/10.1287/opre.2.4.393

[9] Yousra Farhani, Malak Saiem, Taha Arbaoui, and Faicel Hnaien. 2024. Enhanced QUBO Formulations for The Flow Shop Scheduling Problem. In *Proceedings of the 2024 Conference on Artificial Intelligence and Optimization.* 1942–1949. doi:10.1145/3638530.3664150

[10] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A Quantum Approximate Optimization Algorithm. arXiv:1411.4028 [quant-ph] https://arxiv.org/abs/1411.4028

[11] Jatinder N.D. Gupta and Edward F. Stafford. 2006. Flowshop scheduling research after five decades. *European Journal of Operational Research* 169, 3 (2006), 699–711. doi:10.1016/j.ejor.2005.02.001

[12] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. 2019. From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz. *Algorithms* 12, 2 (2019). doi:10.3390/a12020034

[13] IBM Quantum. 2024. Qiskit Algorithms: COBYLA Optimizer. https://qiskit-community.github.io/qiskit-algorithms/stubs/qiskit_algorithms.optimizers.COBYLA.html

[14] IBM Quantum. 2024. Qiskit Algorithms: SPSA Optimizer. https://qiskit-community.github.io/qiskit-algorithms/stubs/qiskit_algorithms.optimizers.SPSA.html

[15] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. 2017. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* 549, 7671 (Sept. 2017), 242–246. doi:10.1038/nature23879

[16] Joseph B. Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.* 7, 1 (1956), 48–50. doi:10.1090/S0002-9939-1956-0078686-7

[17] Krzysztof Kurowski, Tomasz Pecyna, Mateusz Slysz, Rafał Różycki, Grzegorz Waligóra, and Jan Węglarz. 2023. Application of quantum approximate optimization algorithm to job shop scheduling problem. *European Journal of Operational Research* 310, 2 (2023), 518–528. doi:10.1016/j.ejor.2023.03.013

[18] D. H. Lehmer. 1960. Teaching combinatorial tricks to a computer. In *Proceedings of the Symposium on Applied Mathematics*, Vol. 10. 179–193. doi:10.1090/psapm/010

[19] Alberto Peruzzo, Jarrod Mcclean, Peter Shadbolt, Man Hong Yung, Xiaoqi Zhou, Peter Love, Alán Aspuru-Guzik, and Jeremy O'Brien. 2013. A variational eigenvalue solver on a quantum processor. *Nature communications* 5 (04 2013). doi:10.1038/ncomms5213

[20] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (Aug. 2018), 79. doi:10.22331/q-2018-08-06-79

[21] Gerhard Reinelt. 1985. *The linear ordering problem: algorithms and applications.* doctoralthesis. Universität Augsburg.

[22] Valentino Santucci, Marco Baioletti, and Alfredo Milani. 2015. Algebraic Differential Evolution Algorithm for the Permutation Flowshop Scheduling Problem With Total Flowtime Criterion. *IEEE Transactions on Evolutionary Computation* 20 (01 2015), 1–1. doi:10.1109/TEVC.2015.2507785

[23] Mathias Schmid, Sarah Braun, Rudolf Sollacher, and Michael J Hartman. 2024. Highly efficient encoding for job-shop scheduling problems and its application on quantum computers. *Quantum Science and Technology* 10, 1 (Dec. 2024), 015051. doi:10.1088/2058-9565/ad9cba

[24] E. Taillard. 1993. Benchmarks for basic scheduling problems. *European Journal of Operational Research* 64, 2 (1993), 278–285. doi:10.1016/0377-2217(93)90182-M

Project Management anf Scheduling.
[25] Qiskit Development Team. 2023. RealAmplitudes. https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.RealAmplitudes
[26] Davide Venturelli, Dominic J. J. Marchand, and Galo Rojo. 2016. Quantum Annealing Implementation of Job-Shop Scheduling. arXiv:1506.08479 [quant-ph] https://arxiv.org/abs/1506.08479