An Explainable Fuzzy Logic-Based Approach to Quantum Backend Selection

Giovanni Acampora, Allegra Cuzzocrea and Autilia Vitiello
Department of Physics "Ettore Pancini", *University of Naples Federico II*, Naples, Italy
{giovanni.acampora, allegra.cuzzocrea, autilia.vitiello}@unina.it

***Abstract*—Quantum backend selection is emerging as a critical step toward achieving quantum computing utility. Indeed, the accuracy of quantum circuit execution strongly depends on the specific quantum machine used, each characterized by a distinct computational noise pattern. Currently, machine learning models for quantum backend selection are being proposed to address this challenge effectively. However, these approaches still lack two fundamental aspects: the integration of execution queue information, which impacts waiting time and scheduling efficiency, and the explainability of the selected backend, which is crucial to understand and trust the decision-making process. This paper introduces the first fuzzy-based machine learning approach for quantum backend selection capable of addressing both aspects. Experimental results show that the proposed method achieves competitive performance compared to state-of-the-art crisp classifiers, while selecting the backend by incorporating execution waiting times and exposing the circuit features underlying the selection decision.**

***Index Terms*—Fuzzy Decision Trees, Explainable Artificial Intelligence, Quantum Computing, Quantum Backend Selection.**

I. Introduction

Quantum computing is an increasingly expanding field, with the potential to solve problems intractable to classical computers [1], [2]. The rapid growth in quantum algorithms and applications across multiple domains [3], [4] has driven the development of diverse quantum hardware technologies and cloud-based quantum computing platforms, providing access to a broad set of heterogeneous quantum backends. Despite this rapid progress, current quantum technologies operate in the noisy intermediate-scale quantum (NISQ) era, characterized by quantum devices with a limited number of qubits and significant susceptibility to noise and decoherence [5]. In this context, the performance of quantum algorithms is strongly influenced by hardware-specific properties such as qubit connectivity, coherence times, gate fidelities, and error rates, which vary from one quantum processor to another. For this reason, quantum algorithms must be adapted and optimized for a specific backend before execution through a process known as *quantum transpilation*. Depending on the structure of the algorithm and the characteristics of the target hardware, the same quantum algorithm may be more efficiently transpiled and executed on one backend than on another, leading to significant differences in circuit depth, gate composition, and overall execution fidelity. Consequently, selecting the most suitable quantum device for executing a given quantum algorithm has become a well-recognized challenge in today's NISQ landscape. However, practical and systematic guidelines remain largely unavailable, and identifying the best-performing backend often demands substantial expertise in both quantum algorithms and hardware characteristics, making the task non-trivial and time-consuming. This issue particularly affects non-expert end users, who often rely on a trial-and-error approach by executing the same algorithm on multiple quantum devices, each characterized by its own advantages and limitations, in order to assess performance. Such an approach results in a considerable and often impractical computational overhead. Moreover, due to the increasing number of users submitting quantum jobs to available quantum devices, such resources are often not immediately accessible and are instead characterized by execution queues. Consequently, even when the most suitable backend is identified in terms of hardware characteristics, it cannot be guaranteed that the execution will be completed within an acceptable time frame.

Machine learning (ML) approaches such as reinforcement learning have been proposed to support quantum backend selection. Nevertheless, they do not account for backend availability information and do not provide explanations of the backend selection process. In order to bridge this gap, the objective of this work is to identify the most suitable device for executing a quantum circuit by jointly considering both execution-quality and backend availability indicators, while offering human-understandable decision rules. Specifically, a fuzzy decision tree classifier is employed within a supervised-learning framework to guide the backend selection process. In addition to presenting competitive performance with respect to other crisp ML-based models, the proposed approach offers a key advantage in terms of explainability, providing human-understandable decision rules that enable a clearer explanation of the decision-making process.

The remainder of this paper is organized as follows. Section II reviews existing approaches addressing the problem of quantum backend selection. Section III introduces quantum computing with a particular emphasis on the key characteristics of NISQ devices, focusing on backend-specific error profiles and how they impact circuit performance, thereby underscoring the importance of backend selection. Section IV presents and formalizes the proposed strategy. Before concluding in Section VI, Section V describes the experimental setup and reports

the obtained results, together with an in-depth analysis of the extracted decision rules.

II. Related work

While a large body of literature focuses on improving the execution of quantum algorithms on a given quantum backend or investigate the selection of transpilation strategies [6]–[12], only a limited number of works directly address the problem of selecting the most suitable backend for a given quantum circuit. As an example, work [13] provides an automated analysis to identify suitable implementations of quantum algorithms and quantum devices capable of executing a selected implementation with a given input. Nevertheless, this approach relies on exhaustive transpilation and analysis across all available backends. Similarly, work [14] proposes an automated mechanism to select compatible quantum hardware within the context of quantum workflows, i.e., structured pipelines composed of multiple quantum and classical tasks. However, hardware selection is performed at the workflow level and primarily focuses on feasibility and compatibility, rather than on circuit-level performance optimization. Additionally, work [15] proposes a prioritization of compiled circuits across different backends based on user-defined requirements. It relies on hardware calibration data and circuit-specific metrics to feed a multi-criteria decision-making process. More recently, the MQT tool further explores backend selection by introducing a supervised learning–based compiler that identifies suitable backends through a novel compilation strategy driven by reinforcement learning–based selection of transpilation steps [16]. However, this approach requires training a separate reinforcement learning model for each backend of interest, which may limit scalability and practical deployment. From a system-level perspective, work [17] introduces MTMB, a scheduling and resource management framework designed to handle multiple jobs across multiple quantum backends, with the goal of improving overall throughput and fairness in quantum cloud environments. As a throughput- and fairness-oriented scheduling strategy, MTMB optimizes global system objectives and may assign jobs to backends that are sufficiently good, but not necessarily optimal for a specific quantum circuit. In parallel, recent works have investigated execution-time estimation and queueing behavior of quantum programs on cloud-based quantum devices, highlighting the impact of backend availability and queue-related delays [18].

To the best of our knowledge, none of these approaches provides an explainable and automated backend selection strategy that jointly considers execution quality and queue-related availability indicators for a given quantum circuit. This paper bridges this gap by proposing an explainable model that provides an ordered list of quantum devices by explicitly capturing the trade-off between expected execution quality and backend availability. By providing multiple ranked backend options, the proposed approach enables flexible decision making, as a backend that is slightly less performant than the top-ranked one may still yield competitive execution quality while offering significantly better efficiency in terms of execution latency due to a shorter queue.

III. Background

This section introduces the fundamental concepts required to understand the proposed approach.

A. Quantum computing

Quantum computing exploits quantum-mechanical phenomena to process information in ways that can differ substantially from classical computation. The basic unit of information is the *qubit*, whose state is described by a normalized vector in a two-dimensional complex Hilbert space. In the computational basis, a single-qubit pure state can be written as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are the amplitudes and $|\alpha|^2 + |\beta|^2 = 1$. After a measurement, the quantum state is projected onto a basis state, with probabilities given by the squared magnitudes of the corresponding amplitudes. Multi-qubit systems are described by tensor products of individual qubit spaces, enabling *entangled* states that can not be factorized into independent single-qubit states. Within the widely adopted *quantum circuit model*, a quantum circuit is represented as a sequence of operations acting on specified qubits, typically including unitary operations (gates) and, when required, measurements.

B. The NISQ era

Current quantum hardware operates in the so-called NISQ era, characterized by a limited number of qubits and the presence of non-negligible noise affecting quantum operations. Specifically, each available backend exhibits distinct hardware properties, including qubit count, limited qubit connectivity, native gate set, and hardware-specific error patterns shaped by its control stack and physical implementation. These patterns typically include the following types of errors that vary across devices:

- **gate errors**: errors affecting quantum operations that differ across qubits and are often significantly larger for two-qubit entangling operations than for single-qubit rotations;
- **coherence-driven errors**: errors that arise because qubits naturally lose quantum information over time. Examples are amplitude damping (T1 relaxation) and dephasing (T2), which accumulate with circuit depth and idle times;
- **state-preparation and measurement errors (SPAM)**: errors that capture both imperfect initialization and readout misclassification. In detail, state-preparation errors can be caused by residual thermal excitation, incomplete reset, or pulse miscalibration, leading to incorrect initial states, while readout errors are misassignment of $|0\rangle$ and $|1\rangle$ outcomes due to finite measurement fidelity and are commonly asymmetric and heterogeneous across qubits and can drift with calibration.
- **correlated errors**: noise processes in which errors affecting one qubit or gate are statistically dependent on errors affecting other qubits or gates, so the overall noise cannot

be factorized into independent single-qubit (or per-gate) contributions. These effects can induce spatial and temporal correlations across the device. These errors include crosstalk errors, i.e, unintended perturbations of nearby qubits during control or readout, frequency collisions, i.e., spurious interactions caused by spectral crowding or near-resonant transitions, and spectator effects, i.e., errors induced on idle qubits by residual couplings during multi-qubit operations.
- **leakage errors**: errors that involve population escaping the computational two-level subspace $\{|0\rangle, |1\rangle\}$ into higher-energy (non-computational) states, leading to persistent, non-Pauli errors that can degrade subsequent operations and may generate correlated effects

The described errors are backend-dependent: both their strengths and their dominant mechanisms vary across devices due to differences in physical implementation, control electronics, connectivity, and calibration. As a consequence, the performance of a quantum algorithm strongly depends on how well it matches the specific characteristics of the underlying quantum backend. In fact, transpiling a quantum circuit for different backends, lead to varying transpilation outcomes, such as changes in circuit depth, number of two-qubit gates, and overall noise accumulation. As a result, the same quantum algorithm may yield significantly different performance metrics, such as fidelity or output distribution quality, depending on the selected backend. In addition to hardware-related factors, quantum backends are shared resources that are simultaneously accessed by multiple users. Consequently, each backend is typically characterized by a varying number of pending jobs in its execution queue, which directly impacts the waiting time before a circuit can be executed. This combination of backend-dependent execution quality and backend availability further complicates the backend selection problem in the NISQ era.

IV. Proposed approach

In this scetion, an innovative and explainable strategy to select the best backend for a given quantum circuit is presented, providing a ranking that takes into account backends quality and availability. In this way, users can select the most convenient backends and make more informed and practical execution decisions. Formally, let C denote a quantum circuit and let $B = \{b_1, b_2, \ldots, b_N\}$ be the set of N available quantum backends. Each b_i for $i \in \{1, \ldots, N\}$ is characterized by a set of hardware-specific properties, such as native gate set, noise characteristics, and architectural constraints. The final objective of this work is to provide a ranking $\hat{R}$ that includes all candidate backends, ordered from the best to the worse on which to execute C, while accounting for both execution quality and backend availability. As shown in Fig. 1, the proposed strategy is divided in two stages: the prediction of the best backend for C in terms of quality execution employing a fuzzy logic-based classifier and its combination with backend availability. In the following, these two steps are described in detail:

- **Stage 1**: a feature vector V that represents C is computed by considering the features employed in [16], that are the number of each quantum gate in C, the number of qubits, the depth, the program communication, i.e., the average interaction degree for all qubits, the critical depth, i.e., the number of multi-qubit gates on the longest path, the entanglement ratio, i.e., the number of multi-qubit gates, the parallelism, i.e., the degree of parallelism enabled by simultaneous gate execution, and the liveness, i.e., how frequently qubits remain idle while waiting for subsequent gate executions. Once computed, V is provided to a fuzzy decision tree classifier (FDTC) that predicts the backend B_{best} that guarantees the best expected fidelity, i.e, estimation of the probability that C is executed correctly. Compared to classical decision trees, fuzzy decision trees incorporate fuzzy logic to handle uncertainty and imprecision in data. By modeling attributes through fuzzy sets, they allow soft decision boundaries and partial membership of instances to multiple branches, resulting in increased robustness to noise, smoother decision surfaces, and improved interpretability of the learned rules. FDTC assigns graded confidence levels to multiple fidelity outcomes. These confidence levels are aggregated to derive a scalar score, which naturally induces an ordering among candidate backends. In this way, a quality-oriented ranking of candidate backends R_q is obtained.
- **Stage 2**: practical execution also depends on backend availability, which is influenced by the number of queued jobs. To explicitly account for this additional dimension, we adopt the technique for order preference by similarity to ideal solution (TOPSIS) as a multi-criteria decision-making method to refine the backend ranking [19]. In this context, each backend is evaluated with respect to two complementary criteria: (i) the expected execution quality derived from the fuzzy model, and (ii) a queue-related indicator representing backend availability, that is the number of pending jobs. TOPSIS combines these criteria by measuring the relative closeness of each backend to an ideal solution that maximizes execution quality while minimizing queue-related delays. Thus, given R_q and $J = \{j_1, j_2, \ldots, j_N\}$ as the list involving the pending jobs counts for each backend, TOPSIS can be applied to obtain the final ranking $\hat{R}$ that captures an explicit trade-off between expected execution fidelity and backend availability, enabling more practical and time-efficient backend selection decisions. More formally, let s_i be the quality score of backend b_i and j_i its pending jobs count. First, the criteria to make them comparable are normalized:

$$r_{i,s} = \frac{s_i}{\sqrt{\sum_{k=1}^{N} s_k^2}}, \quad r_{i,j} = \frac{j_i}{\sqrt{\sum_{k=1}^{N} j_k^2}} \tag{1}$$

Then, the positive ideal solution A^+, which represents the best possible performance (high quality, low queue), and the negative ideal solution A^- are determined. The final score for each backend, SC_i, is calculated based on its Euclidean distance from these two points:

$$SC_i = \frac{d_i^-}{d_i^+ + d_i^-} \tag{2}$$

where d_i^+ and d_i^- are the distances from A^+ and A^-, respectively. The final ranking $\hat{R}$ is obtained by sorting the backends according to SC_i in descending order.

V. Experiments and results

This section presents a detailed description of the experimental session aimed at evaluating the proposed approach. Firstly, this section describes the construction of the dataset used for backend selection in terms of execution quality. Secondly, the experimental workflow with the settings is discussed. The performance of the approach are evaluated by considering two different aspects: the achieved accuracy of the classification task and the quality-based ranking predictions of FMDT in terms of the mean reciprocal rank metric. Finally, an explainability analysis is carried out to show the human-understandability of our approach.

A. Dataset creation

The backend selection is driven by an estimate of the expected execution fidelity, which is used as the primary criterion to assess the suitability of a quantum device for a given circuit. In particular, each sample of the datasets used to train and test the FDTC model is created as follows. First, a set of candidate backends is selected $B = \{b_1, b_2, \ldots, b_N\}$. Then, a quantum random circuit C is generated and transpiled on each candidate b_i with $i \in \{1, \ldots, N\}$. Specifically, the expected execution fidelity is computed by aggregating the fidelities associated with the individual operations composing the transpiled circuit. Let T_i denote the transpiled version of circuit C on backend B_i, and let $\mathcal{G}(T_i) = \{g_1, g_2, \ldots, g_M\}$ be the set of quantum operations contained in T_i, excluding non-operational instructions such as barriers. For a given backend b_i, each operation g_j is associated with an error rate obtained from the backend calibration data. In particular, readout errors are considered for measurement operations, while gate error rates are considered for single- and two-qubit gates. The fidelity associated with operation g_j is defined as

$$f(g_j, b_i) = \begin{cases} 1 - \epsilon_{\text{ro}}(q), & \text{if } g_j \text{ is a measurement on qubit } q, \\ 1 - \epsilon_{g_j}(\mathcal{Q}_j), & \text{otherwise,} \end{cases} \tag{3}$$

where $\epsilon_{\text{ro}}(q)$ denotes the readout error of qubit q, and $\epsilon_{g_j}(\mathcal{Q}_j)$ denotes the error rate of gate g_j acting on the set of qubits $\mathcal{Q}_j$. Under the assumption of independent error contributions, the expected execution fidelity of circuit C on backend B_i is then estimated as

$$F_{\text{exp}}(C, b_i) = \prod_{g_j \in \mathcal{G}(T_i)} f(g_j, b_i). \tag{4}$$

Thus, by construction, the expected execution fidelity satisfies

$$F_{\text{exp}}(C, b_i) \in [0, 1], \tag{5}$$

where values closer to 1 indicate a higher expected probability of correct execution, while values closer to 0 correspond to lower expected execution quality. This multiplicative formulation provides a practical and backend-aware proxy for execution quality, enabling a quantitative comparison of different quantum devices for a given circuit.
Once the expected execution fidelities for all N candidate backends are computed, the backend associated with the highest value is selected as the label of the corresponding dataset sample in the supervised learning setting.

B. Experimental setup

The backends selected during the experimental session are three 127-qubit and one 133-qubit Fakebackends from Qiskit SDK [20]: `FakeOsaka`, `FakeSherbrooke`, `FakeWashingtonV2`, and `FakeTorino`. A dataset consisting of 1200 randomly generated quantum circuits was created, with 300 circuits associated with each backend. Specifically, each circuit was generated with a number of qubits randomly sampled in the range $[2, 30]$ and composed of the following gate set: *H*, *X*, *Y*, *Z*, *S*, *T*, *RX*, *RY*, *RZ*, and *CX*. To introduce variability in noise accumulation, the circuit depth was randomly sampled in the range $[2, 50]$. Each quantum circuit was transpiled for all available backends using the maximum optimization level provided by Qiskit in order to obtain highly optimized backend-specific circuit representations. The dataset was partitioned into training and test subsets, with 70% of the data allocated to training and 30% to testing, using a stratified splitting strategy to ensure that both subsets were perfectly balanced with respect to the target classes. Then, a distributed fuzzy decision tree classifier fuzzy multi-way decision tree (FMDT) was used as a fuzzy classifier, as provided in FuzzyML library [21]. Specifically, it extends classical decision trees by incorporating fuzzy logic to handle uncertainty and imprecision in data.

C. Performance Analysis

Classification performance of the trained model was evaluated by testing it on the test set in terms of accuracy. Moreover, its performance was compared with that of well-established crisp classifiers, including its counterpart decision tree, random forest, multi-layer perceptron, and support vector machine. Accuracy values are shown in Tab. I. It is evident that FMDT delivers competitive results compared to the alternatives, with a slightly lower value than random forest and a greater value than all the others, demonstrating its ability to effectively model the backend selection problem and discriminate between devices with similar qubit counts but different noise profiles and connectivity structures. Despite its substantially simpler structure compared to random forest, the FMDT achieves comparable classification performance while offering the key advantage of intrinsic explainability. Owing to its fuzzy formulation, the model performs backend selection in

Fig. 1. From the circuit C, features V are extracted and evaluated via FMDT to obtain an intermediate ranking R_q, which is refined by considering J and using TOPSIS to produce the final ranking $\hat{R}$.

a soft manner, assigning each candidate backend a confidence score that reflects its suitability for executing a given quantum circuit. These scores not only identify the most suitable device, but also naturally induce a ranking over the available backends, enabling a more informative and flexible selection strategy than a single hard decision.

To evaluate the performance of quality-based ranking predictions of FMDT, the mean reciprocal rank (MRR) metric was employed. Specifically, it was possible to assess how effectively the optimal backend is positioned within the predicted ranking. For each quantum circuit, the reciprocal rank is defined as the inverse of the position at which the correct backend appears in the ranked list, and the MRR is obtained by averaging this value over all test samples. This metric provides a comprehensive measure of ranking quality, as it rewards predictions that place the optimal backend in higher positions and penalizes those in which it appears lower in the ranking. FMDT presents an MRR value of 0.81, demonstrating that most of the times the best backend is placed in the first or second position of the predicted ranking. When real quantum devices are considered, additional runtime constraints can be incorporated by accounting for the number of pending jobs associated with each backend. In this case, the confidence-based quality ranking is combined with backend availability information through the TOPSIS multi-criteria decision-making method, yielding a final ranking that jointly considers both expected execution quality and estimated waiting time. A specific case study was conducted to analyze how the initially predicted quality-based ranking evolves when backend availability is taken into account. Specifically, the FMDT was employed to generate a quality-based ranking for a representative sample from the test set. The resulting ranking placed `FakeOsaka` in the first position with a normalized aggregated fuzzy degree of support of 52%, followed by `FakeWashingtonV2` with 26%, `FakeTorino` with 10%, and `FakeSherbrooke` with 9%. Subsequently, a number of pending jobs was associated with each backend, following the same order as the predicted ranking, namely [20,2,2,0]. To jointly account for execution quality and backend availability, the TOPSIS method was applied using a weight of 0.6 for quality and 0.4 for availability. After computing the TOPSIS scores, the final ranking reverses the first two positions. This result indicates that `FakeWashingtonV2`, despite having a slightly lower fuzzy support score than `FakeOsaka`, benefits from significantly lower queue congestion, thereby representing a more favorable choice when execution time is also considered.

TABLE I
CLASSIFICATION ACCURACY ACROSS THE EVALUATED MODELS.

Models				
FMDT	**Decision Tree**	**Random Forest**	**MLP**	**SVM**
0.69	0.66	0.70	0.54	0.64

D. Explainability analysis

A key advantage of the FMDT model lies in its inherent explainability. Owing to its tree-based and fuzzy rule-based structure, FMDT represents the decision-making process through a set of human-understandable IF–THEN rules, where each rule combines fuzzy conditions on circuit features with an associated confidence level. These rules make explicit how different circuit characteristics contribute to the backend selection outcome, allowing practitioners to trace and interpret the reasoning behind each decision. Moreover, the fuzzy formulation enables soft reasoning, in which multiple rules may be partially activated for a given input, providing a transparent representation of uncertainty and trade-offs among candidate backends.

To provide a global interpretation of the learned decision logic, we analyzed the extracted fuzzy rules and computed, for each backend, the frequency with which each input feature appears in the antecedent part of the rules associated with that backend. Specifically, after mining the full set of IF–THEN rules from the trained FMDT, we assign each rule to the backend corresponding to the highest value in its consequent vector. For each backend, we then count how many times each feature index occurs across the antecedents of all rules assigned to it. Finally, features are ranked in descending order of occurrence, and the 5 top-ranked ones are reported as the most frequent features for that backend. Histograms including these occurrences are reported in Fig. 2. The feature frequency analysis reveals both common and backend-specific patterns. In particular, features such as *program communication* and *cx* appear consistently among the most frequent ones across all backends, suggesting that they represent globally relevant circuit characteristics influencing backend suitability. Conversely, other features exhibit backend-dependent relevance. For instance, *critical depth* and *number of qubits* appear more prominently in the rules associ-

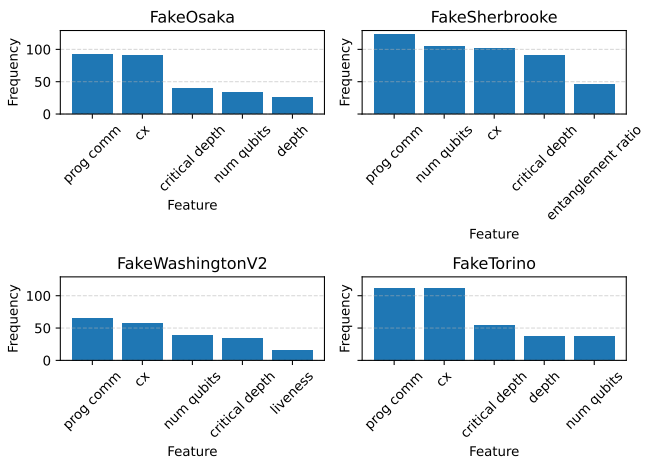

Fig. 2. Frequency of the top-5 features per backend extracted from the FMDT rules.

ated with `FakeSherbrooke`, while liveness emerges among the top features for `FakeWashingtonV2`. These differences indicate that, while the model relies on a shared set of core features, it captures backend-specific decision patterns through the relative importance of certain circuit properties.

VI. Conclusions

In this work, we investigated a fuzzy logic–based model for predicting a candidate backend ranking for the execution of quantum circuits. The proposed approach enables not only the identification of the most suitable device, but also the generation of an ordered list of alternative backends by jointly accounting for expected execution quality and backend availability. By leveraging the fuzzy nature of the model, backend selection is performed in a soft and interpretable manner, thereby providing additional insight into the decision-making process. Experimental results demonstrate competitive performance with respect to crisp ML models, while simultaneously delivering consistent and human-interpretable explanations of the backend selection process across different circuits and devices. Overall, this work paves the way for explainable availability-aware backend selection strategies that can support informed and transparent decision-making in practical quantum computing workflows.

Future research directions include extending the evaluation to structured quantum algorithms and validating the proposed strategy on real quantum hardware to benchmark performance under realistic noise conditions.

References

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2011.

[2] F. Arute et al., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, Oct. 2019. [Online]. Available: https://doi.org/10.1038/s41586-019-1666-5

[3] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica, G. Salton, S. Wang, and F. G. S. L. Brandão, "Quantum algorithms: A survey of applications and end-to-end complexities," 2023. [Online]. Available: https://arxiv.org/abs/2310.03011

[4] G. Acampora, A. Ambainis, N. Ares, L. Banchi, P. Bhardwaj, D. Binosi, G. A. D. Briggs, T. Calarco, V. Dunjko, J. Eisert *et al.*, "Quantum computing and artificial intelligence: status and perspectives," *arXiv preprint arXiv:2505.23860*, 2025.

[5] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018. [Online]. Available: https://doi.org/10.22331/q-2018-08-06-79

[6] P. D. Nation and M. Treinish, "Suppressing quantum circuit errors due to system variability," *PRX Quantum*, vol. 4, p. 010327, Mar 2023. [Online]. Available: https://link.aps.org/doi/10.1103/PRXQuantum.4.010327

[7] Y. Ji, X. Chen, I. Polian, and Y. Ban, "Algorithm-oriented qubit mapping for variational quantum algorithms," *Physical Review Applied*, vol. 23, no. 3, p. 034022, 2025.

[8] A. Zeynali and Z. Bakhshi, "Noise-adaptive quantum circuit mapping for multi-chip nisq systems via deep reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2511.18079

[9] P. Zhu, S. Feng, and Z. Guan, "An iterated local search methodology for the qubit mapping problem," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 8, pp. 2587–2597, 2022.

[10] S. Dangwal, G. S. Ravi, L. M. Seifert, P. Das, J. Sud, and F. T. Chong, "Clifford assisted optimal pass selection for quantum transpilation," 2025. [Online]. Available: https://arxiv.org/abs/2306.15020

[11] D. Kremer, V. Villar, H. Paik, I. Duran, I. Faro, and J. Cruz-Benito, "Practical and efficient quantum circuit synthesis and transpiling with Reinforcement Learning," *arXiv e-prints*, p. arXiv:2405.13196, May 2024.

[12] M. Salm, J. Barzen, F. Leymann, B. Weder, and K. Wild, "Automating the comparison of quantum compilers for quantum circuits," in *Service-Oriented Computing*, J. Barzen, Ed. Cham: Springer International Publishing, 2021, pp. 64–80.

[13] M. O. Salm, J. Barzen, U. Breitenbücher, F. Leymann, B. Weder, and K. Wild, "The nisq analyzer: Automating the selection of quantum computers for quantum algorithms," in *Symposium and Summer School on Service-Oriented Computing*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:263887370

[14] B. Weder, J. Barzen, F. Leymann, and M. Salm, "Automated quantum hardware selection for quantum workflows," *Electronics*, vol. 10, no. 8, 2021. [Online]. Available: https://doi.org/10.3390/electronics10080984

[15] M. Salm, J. Barzen, F. Leymann, and B. Weder, "Prioritization of compiled quantum circuits for different quantum computers," in *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 1258–1265.

[16] N. Quetschlich, L. Burgholzer, and R. Wille, "Mqt predictor: Automatic device selection with device-specific circuit compilation for quantum computing," *ACM Transactions on Quantum Computing*, vol. 6, no. 1, p. 1–26, Jan. 2025. [Online]. Available: http://dx.doi.org/10.1145/3673241

[17] D. Zheng, J. Xv, F. Yue, Q. Du, Z. Wang, and Z. Shan, "A novel scheduling framework for multi-programming quantum computing in cloud environment," *Computers, Materials and Continua*, vol. 79, no. 2, pp. 1957–1974, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S154622182400239X

[18] N. Ma and H. Li, "Understanding and estimating the execution time of quantum circuits," 2025. [Online]. Available: https://arxiv.org/abs/2411.15631

[19] C.-L. Hwang and K. Yoon, *Multiple attribute decision making: methods and applications a state-of-the-art survey*. Springer Science & Business Media, 2012.

[20] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross *et al.*, "Quantum computing with qiskit," *arXiv preprint arXiv:2405.08810*, 2024.

[21] A. Segatori, F. Marcelloni, and W. Pedrycz, "On distributed fuzzy decision trees for big data," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 1, pp. 174–192, 2018.