Stacking the Odds: Full-Stack Quantum System Design Space Exploration

Hila Safi[1,4*†], Medina Bandic[2, 3*†], Christoph Niedermeier[4], Carmen G. Almudever[5], Sebastian Feld[2, 3], Wolfgang Mauerer[1, 4]

[1]Laboratory for Digitalisation,Technical University of Applied Sciences Regensburg, Seybothstr. 2, Regensburg, 93053, Bavaria, Germany.
[2]Quantum & Computer Engineering, Delft University of Technology, Mekelweg 5, Delft, 2628 CD, Netherlands.
[3]QuTech, Lorentzweg 1, Delft, 2628 CJ, Delft, Netherlands.
[4]Foundational Technology, Siemens AG, Friedrich-Ludwig-Bauer-Str. 3, Garching bei München, 85748, Bavaria, Germany.
[5]Computer Engineering, Universitat Politècnica de València, Camí de Vera, València, 46022, Spain.

*Corresponding author(s). E-mail(s): hila.safi@siemens.com; m.bandic@tudelft.nl;
Contributing authors: christoph.niedermeier@siemens.com; cargara2@disca.upv.es; s.feld@tudelft.nl; wolfgang.mauerer@othr.de;
†These authors contributed equally to this work.

Abstract

Design space exploration (DSE) plays an important role in optimising quantum circuit execution by systematically evaluating different configurations of compilation strategies and hardware settings. In this paper, we conduct a comprehensive investigation into the impact of various layout methods, qubit routing techniques, and optimisation levels, as well as device-specific properties such as different variants and strengths of noise and imperfections, the topological structure of qubits, connectivity densities, and back-end sizes. By spanning through these dimensions, we aim to understand the interplay between compilation choices and hardware characteristics. A key question driving our exploration is whether the optimal selection of device parameters, mapping techniques, comprising of initial layout strategies and routing heuristics can mitigate device induced errors beyond standard error mitigation approaches. Our results show that carefully

selecting software strategies (*e.g.*, mapping and routing algorithms) and tailoring hardware characteristics (such as minimising noise and leveraging topology and connectivity density) significantly improve the fidelity of circuit execution outcomes, and thus the expected correctness or success probability of the computational result. We provide estimates based on key metrics such as circuit depth, gate count and expected fidelity. Our results highlight the importance of hardware–software co-design, particularly as quantum systems scale to larger dimensions, and along the way towards fully error corrected quantum systems: Our study is based on computationally noisy simulations, but considers various implementations of quantum error correction (QEC) using the same approach as for other algorithms. The observed sensitivity of circuit fidelity to noise and connectivity suggests that co-design principles will be equally critical when integrating QEC in future systems. Our exploration provides practical guidelines for co-optimising physical mapping, qubit routing, and hardware configurations in realistic quantum computing scenarios.

Keywords: Design Space Exploration (DSE), hardware-software co-design, quantum circuit compilation, NISQ devices

1 Introduction

As quantum computing moves closer to practical application, the performance bottlenecks are no longer defined solely by hardware limitations or algorithmic efficiency, but increasingly by how well both align [1]. Current devices suffer from limited qubit counts, restricted connectivity, and different types of noise, all of which degrade circuit performance and solution quality. Just as classical computing has benefited from decades of hardware-software co-design, quantum computing demands a similar approach [1–3]. In this work, we take a layered perspective on quantum circuit compilation (transpilation)[4], examining how choices made across the hardware-software stack collectively shape performance and impact fidelity and resource efficiency. We apply a Design Space Exploration (DSE) approach to systematically evaluate these strategies. DSE enables the identification of optimal trade-offs by balancing the exploration of novel design configurations with exploitation of known high-performing strategies [5]. As illustrated in Table 1, we span five layers—from the initial circuit setup, through qubit mapping and routing, down to noise modelling and hardware topology. At each layer, we systematically explore the design space using a set of representative benchmark circuits covering algorithmic, variational, arithmetic, and simulation-based circuits to enable a meaningful evaluation. These include well-known algorithms, for instance Quantum Fourier Transform (QFT), Shor's algorithm, or VQE [6–8]. To reflect the ongoing research toward fault-tolerant quantum computing, we extend our evaluations to include quantum error correction codes—thus aligning near-term compilation strategies with future fault-tolerant demands. We begin our investigation with device-level parameters, including back-end size, qubit connectivity, topologies and different noise variants— with a focus on crosstalk. Crosstalk is a dominant and often underestimated source of correlated error [9]. All hardware aspects

influence gate scheduling, circuit depth, error rates, and routing complexity. Building on this, we examine the mapping layer, and evaluate how different initial layout strategies, qubit routing techniques, and optimisation levels (as implemented in Qiskit [4]) impact circuit depth and fidelity. On top of these, we incorporate multiple additional optimisations (*e.g.* gate simplification techniques). Our results confirm that optimal circuit compilation is not only back-end-dependent in terms of architecture, but also strongly influenced by hardware-specific noise characteristics such as decoherence and crosstalk. Moreover, we show that informed decisions on quantum circuit mapping can reduce the effect of noise, achieving improvements in addition to conventional error mitigation techniques like Zero Noise Extrapolation [10]. We present a layered DSE framework for quantum circuit optimisation, highlight the critical impact of connectivity and noise variations, particularly crosstalk, and show that co-optimised mapping settings improve circuit fidelity and resource efficiency. By benchmarking multiple quantum algorithms across this multi-layered stack, our work offers actionable guidelines for future quantum system design. Taken together, our contributions demonstrate the necessity for treating quantum circuit compilation as a full-stack optimisation problem. Rather that tuning individual parameters in isolation, we advocate for a holistic, hardware-aware approach. This layered perspective enables more informed design choices, ultimately leading to higher-fidelity executions and more efficient use of limited quantum resources. The paper is augmented by a reproduction package [11], (link in PDF) that also contains the full set of benchmarks used in each experiment. For long-term reproducibility and archival, we also provide a snapshot via Zenodo.

Table 1: Full-stack design space (parameters and techniques) explored in our work.

Layers	Examples
Algorithmic Design & Circuit Setup	Problem formulation, QAOA layers
Logical Circuit Optimisation	Gate count reduction, optimise Clifford Gates
Qubit Mapping & Routing	Qubit allocation, routing, swap gate insertion
Noise Modelling & Hardware Simulation	Depolarisation, crosstalk
Topology	Heavy-hex, Sycamore

2 Context, Foundation & Prior Work

The execution of quantum circuits on current NISQ (Noisy Intermediate-Scale Quantum) devices presents significant challenges due to hardware limitations, error-prone operations, and restricted qubit connectivity. Addressing these constraints requires a full-stack quantum computing approach, where both the quantum hardware and software stack are co-designed to enhance performance and scalability. Safi *et al.* [12] demonstrate that co-designing quantum processing units (QPUs) for specific applications can significantly improve execution performance, even on relatively simple architectures. Complementary to hardware-focused efforts, recent work [13] introduces

software-layer co-design methodology that automates the selection of quantum-classical algorithms and their parameters based on non-functional requirements, thereby supporting scalable and application-aware quantum software development. Bandic *et al.* [14] argue that end-to-end integration—from qubit control hardware to compilers—is essential for effective noise mitigation and progress toward fault tolerance. Such studies highlight the importance of co-optimisation across the stack to align hardware capabilities with software demands. At the device level, critical parameters such as back-end size, qubit count, native gate set, connectivity, and noise characteristics profoundly influence circuit execution. Optimising these properties has been the focus of several works. For instance, Murali et al. [15] proposed a noise-adaptive strategy that selects qubits based on individual error rates to increase execution fidelity. Li et al. [16] demonstrated that topology-aware design decisions—such as application-specific qubit placement, customised coupling graph topologies, and routing-aware compilation—can reduce the resource overhead needed for realistic workloads. The impact of these parameters extends into the software stack, where hardware-aware compilers leverage device knowledge to guide circuit optimisation [17]. Furthermore, the MQT Predictor framework by Quetschlich *et al.* [18] introduces an automated approach to selecting quantum devices and optimising compilation flows, also demonstrating significant performance gains. Although such techniques are relevant for near-term devices, circuit optimisation remains equally critical for fault-tolerant devices [19]. Our work explores the intersection of hardware-level parameters and compilation strategies to identify and evaluate the optimal balance, that maximised circuit fidelity with realistic architectural constraints.

2.1 NISQ Architecture vs. FTQC

NISQ era, coined by Preskill [20], describes today's quantum devices with 50 to 1000 qubits that suffer from noise and decoherence. Although not fully error-corrected, an intensive discussion has unfolded on the computational advantage of NISQ machines through variational methods [21] or the QAOA class of algorithms [22, 23]. Despite being unsuitable for large-scale fault-tolerant tasks, error mitigation techniques can improve their practical utility [24]. Fault-tolerant quantum computing (FTQC) [25] overcomes these limitations using quantum error correction (QEC), e.g., surface codes [26], which encode logical qubits across many physical ones. FTQC demands high qubit fidelity and scale. Bridging NISQ and FTQC will require hybrid strategies—combining classical and quantum processing— , resilient algorithms, improved connectivity, and crosstalk-tolerant QEC codes [12, 27].

2.2 DSE for Hardware-Software Co-Design

In the pursuit of practical quantum computing applications, hardware-software co-design (HW-SW co-design) is a promising strategy, particularly in light of the current limitations of quantum hardware, such as restricted number of qubits and high error rates—especially in *two-qubit* gate operations [12, 14, 17, 28, 29]. These constraints are a significant challenge for executing complex quantum algorithms in a reliable way. HW-SW co-design involves the collaborative development of quantum hardware and

software, ensuring that both components are optimised in tandem to enhance overall system performance. By aligning the design of quantum algorithms with the specific characteristics of QPUs, and vice versa, co-design facilitates more efficient and effective quantum computations [3, 23]. In our research, we employ design space exploration as a systematic approach to optimise quantum circuit compilation and assess the hardware characteristics necessary for efficient execution of benchmark problems [5]. DSE, in general, enables structured evaluation of architectural and algorithmic design choices—such as in our case qubit mappings, circuit decompositions, and noise-aware strategies for error reduction—with the goal of enhancing performance, scalability and resource efficiency. Within this framework, quantum circuit compilation refers to the process of transforming a high-level circuit into a low-level, hardware-executable form, involving qubit mapping and operations scheduling. Among the key compiler-level challenges, quantum circuit mapping plays a fundamental role in enabling full-stack quantum computing [30, 31]. Efficient mapping is supposed to reduce gate overhead and latency, but becomes increasingly challenging as qubit count grows. To optimise these metrics, various approaches—ranging from heuristics and brute-force methods to graph-based, dynamic programming, and machine learning techniques—have been developed [32–37]. We systematically analyse critical metrics such as circuit depth and gate counts, as well as the impact of various noise sources—particularly crosstalk—on overall circuit performance. We observed performance variations stem from key hardware properties, including topology, layout strategies, qubit routing techniques, and optimisation levels. Table 1 illustrates the key hardware-aware design choices involved in our benchmarking approach, spanning logical circuit optimisation, qubit mapping, noise modelling, and hardware back-end selection. Although we do not modify the circuit design in our study, we incorporate a broad range of circuits, including error-correcting ones, to assess their performance across different compilation and execution strategies. This ensures that our findings remain relevant not only for current quantum devices but also as a preparatory step toward fault-tolerant quantum computing. This structured co-design approach sets the foundation for the next sections, where we dive deeper into device design parameters and compilation strategies.

2.2.1 Quantum Device Design

A comprehensive analysis of device-specific parameters is essential to understand their impact on scalability and the feasibility of achieving practical utility in quantum computing. Key factors include:

- **Device Topology:** Physical arrangement and connectivity of qubits, which impacts quantum circuit mapping and execution efficiency. This includes the size and connectivity of the system back-end.
- **Native Gate Set:** Set of quantum gates natively supported by the respective hardware, which affects gate count and parallelisation.
- **Gate and Measurement Fidelity:** Probability of correctly applying quantum gates and accurately measuring states, which determine computational reliability.

- **Coherence Time:** Temporal stability of qubits before decoherence. Decoherence refers to the loss of a qubit's superposition state over time due to environmental interactions, causing the quantum state to collapse into a classical outcome [38].
- **Noise and Error Models:** The various sources of quantum errors that affect solution quality, including crosstalk, thermal relaxation, depolarisation noise, readout noise, and gate-based noise [39].

These parameters collectively define the operational envelope of a quantum processor. They not only determine which algorithms can be realistically deployed, but also shape compilation strategies, error mitigation techniques, and ultimately the feasibility of achieving quantum advantage on current and near-term hardware [17].

2.2.2 Compiling Quantum Circuits to Quantum Hardware

A systematic approach to quantum circuit compilation is essential to bridge the gap between abstract algorithms and hardware-specific execution. To run a circuit on a given quantum processor, it must be transformed to comply with hardware constraints, which vary across platforms. These constraints, many of which are outlined in Section 2.2.1, introduce complex challenges for efficient execution [5]. Addressing them requires a sequence of interdependent tasks, whose order and implementation depend on the architecture and optimisation goals. The key tasks are:

- **Gate Decomposition:** The gates are transformed into the primitive gate set supported by the quantum processor [40].
- **Operation Scheduling** Scheduling gates in parallelised manner to ensure minimal circuit depth and decoherence effects while respecting all the architecture-dependent shared control constraints [41].
- **Qubit Allocation:** Assigns logical qubits to the physical ones [42].
- **Qubit Routing:** Moves logical qubits into adjacency for *two-qubit* interactions. For instance, by introducing SWAP gates [43–45].
- **Circuit Optimisation:** Performing tasks such as gate commutation or cancellation, transformation, with the aim of achieving the simplest form of the circuit [46].

Our work provides a comprehensive framework for co-optimising quantum software and informing hardware design. While it does not directly optimise hardware components, it identifies problem-dependent requirements and performance bottlenecks that guide the specification of optimal hardware configurations. This enables more informed algorithmic design choices and supports the selection or development of hardware best suited for targeted computational tasks.

3 Methodology

In this section, we outline the methodology employed to evaluate quantum circuit performance under varying device and compilation conditions. We investigate the impact of different parameters as illustrated in Table 2. Our experimental approach comprises two parts: one focusing on varying device parameters, while keeping compilation settings fixed, and the other examining the effects of varying compilation conditions

while maintaining fixed device parameters. This dual approach allowed us to isolate and analyse the effect of device variations independently from compilation variations, thereby providing a comprehensive understanding of these factors and their influence on quantum circuit performance under simulation.

Table 2: Parameter ranges used in our experimental setup.

Parameter	Options
Device Design	
Connectivity Density	[0.013895, 1.0]
back-end	QASM Simulator (Qiskit)
Native Gate Set	'id', 'rz', 'sx', 'x', 'cx', 'swap', 'cz'
Coupling Map	Heavy-Hex, Sycamore
back-end Size (Heavy-Hex)	6×4, 6×5, 8×5
back-end Size (Sycamore)	6×6, 11×11, 12×12
Noise Model	Crosstalk, Thermal Relaxation, Depolarisation
Compiler Design	
Optimisation Level	$0, 1, 2$
Layout Method	SABRE, Dense, Trivial
Routing Technique	SABRE, Stochastic
Additional Opt. Setups	0-5
Scheduling Method	ALAP

3.1 Device Layer

In quantum computing, device parameters such as connectivity, topology, and noise significantly play a critical role in determining the fidelity and efficiency of quantum circuit execution. These factors influence how circuits are mapped, routed, and scheduled, often introducing overheads that significantly affect overall performance. To ensure a comprehensive and systematic evaluation, we examine the impact of device connectivity, topology configurations, back-end sizes and noise models. For experiments that involve varying device parameters, we employed fixed compiler settings as follows: optimisation level 3, qubit routing technique SABRE, layout method SABRE and scheduling method "as late as possible" (ALAP) [4]. A detailed analysis is presented in the subsequent section.

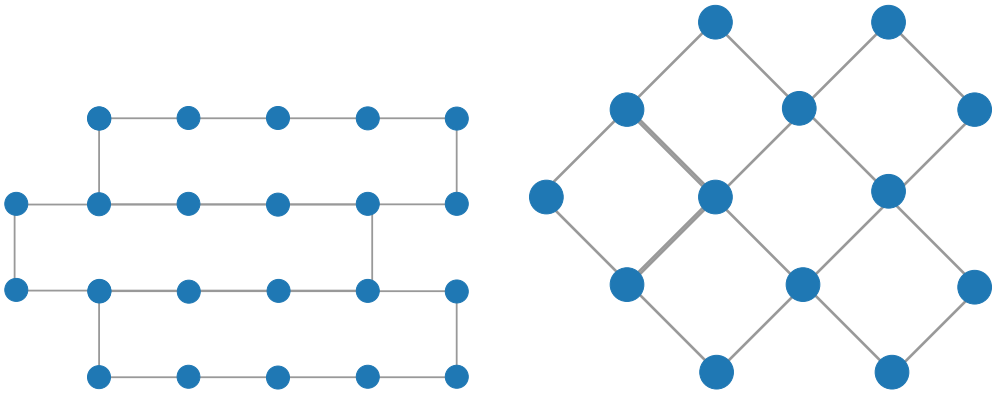
3.1.1 Topology and Connectivity

The quantum circuits were compiled for two hardware architectures: a heavy-hex and Sycamore, both illustrated in Figure 1. Each architecture is characterised by a specific topology—the qubit connectivity graph—and a corresponding physical layout, which refers to the concrete arrangement of qubits on the chip. The heavy-hex topology optimises connectivity while reducing crosstalk by limiting qubit interactions to carefully placed neighbors [47], whereas the grid like topology of the Sycamore chip

uses a 2D grid where each qubit connects to up to four neighbors, enabling efficient gate operations [48]. To compare the topologies we introduce the following metric:

$$\text{Relative depth} = \frac{D_{\text{sycamore}} - D_{\text{heavy-hex}}}{D_{\text{heavy-hex}}}, \tag{1}$$

where D_{sycamore} and $D_{\text{heavy-hex}}$ denote the circuit depths when transpiled to the Sycamore and heavy-hex layouts. This normalised measure captures the proportional increase or decrease in circuit depth due to differences in topology and connectivity. It allows for comparisons by quantifying how circuit execution complexity scales relative to a baseline, independent of the absolute circuit size. We study how varying the ratio of problem size to back-end size affects circuit fidelity by uniformly scaling the physical layout. By systematically adjusting the ratio of problem size to physical qubit count, we analyse how the spatial embedding of a fixed logical circuit affects execution fidelity across different topologies and device sizes. When the number of qubits used to encode a problem is significantly smaller than the size of the quantum processor, multiple logical-to-physical qubit mappings become possible. This flexibility in placement can influence circuit fidelity, as different mappings may lead to variations in gate routing and noise accumulation. To explore this effect, we vary the ratio of encoded problem size to back-end size by uniformly scaling the physical layout. back-end sizes are described in terms of their $2D$ as $(n \times m)$, where n denotes the number of rows, and m the number of columns. Layouts implementing the heavy-hex topology range from configurations with 6×4 (143 physical qubits) to 8×6 (297 physical qubits). Similarly, Sycamore layouts range from 36 qubits (6×6) to 144 qubits (12×12).



(a) Heavy-hex topology

(b) Grid-like topology of the Sycamore layout

Fig. 1: Comparison of hardware topologies.

The connectivity of the back-end is measured in terms of a *connectivity density*

$$c = \frac{N_C}{N_{C,\max}}, \tag{2}$$

with N_C denoting the total number of edges in the hardware graph and $N_{C,\max} = N(N-1)/2$ the maximal number of edges for N qubits. If two qubits are connected, they can physically interact and a *two-qubit* gate can be performed between them. $c = 1$ describes a device with all-to-all connectivity. In the experiments presented below, the connectivity density is increased by randomly adding connections between qubit pairs that are not yet connected, continuing until full connectivity is reached. The average number of nearest neighbours per qubit grows linearly with the connectivity density.

3.1.2 Noise

In quantum computing, noise plays a critical role in limiting circuit reliability and fidelity. We examine common noise sources—crosstalk, thermal relaxation, and depolarisation—and their impact on benchmark circuits. Due to computational limitations, fidelities are approximated analytically using a model that incorporates the number of gates, gate fidelities, and circuit depth.

Crosstalk

refers to unintended interactions where operations on one qubit affect nearby qubits, reducing fidelity. It remains a hardware-specific challenge without a standardised model. Inspired by prior studies on superconducting qubits [49, 50], we evaluate three crosstalk models — the shared qubit, the simultaneous execution, and proximity based model — each capturing different aspects of crosstalk behavior to assess their impact on circuit fidelity.

Shared qubit introduces crosstalk noise whenever two *two-qubit* gates share a common qubit, regardless of whether they occur in the same layer or at different times in the circuit. This model is inspired by studies on superconducting qubits [50–52]. Prior research has shown that crosstalk effects during *two-qubit* gate operations become significant when the gates share a neighboring pair. A neighboring pair occurs when a qubit involved in one *two-qubit* gate is directly connected to a qubit involved in another *two-qubit* gate. Let qubits i and j participate in one *two-qubit* gate operation, and qubits k and l in another. The crosstalk effect $C(i, j, k, l)$ is defined as follows:

$$C(i,j,k,l) = \begin{cases} 1 & \text{No connection between } i,j \text{ and } k,l, \\ C_{01,02} & \text{Connection between } i,j \text{ and } k,l. \end{cases}$$

To quantify the effect of neighboring gates, the fidelity metric $C_{01,02}$ is defined as the harmonic mean of the fidelities of the overlapping *two-qubit* gates:

$$C_{01,02} = \left(\frac{2}{\frac{1}{F_{O_1}} + \frac{1}{F_{O_2}}} \right)^n . \tag{3}$$

Here:

- F_{O_1} and F_{O_2}: Fidelities of the two overlapping operations O_1 and O_2.
- n: Degree of crosstalk amplification, proportional to the number *two-qubit* gate operations executed.

The **simultaneous execution** model surfaces when at least two *two-qubit* gates are executed within the same circuit layer. This model introduces crosstalk when *two-qubit* gates are executed concurrently, regardless of spatial separation, affecting neighboring qubits connected to the active ones [51]. For qubits i and j undergoing simultaneous *two-qubit* operations at time k, the crosstalk effect $C(i, j, k)$ is given by:

$$C(i,j,k) = \begin{cases} 1 & \text{No operation on } i \text{ and } j \text{ at time } k, \\ C_{01,02} & i, j \text{ undergo ops. } O_1, O_2 \text{ at time } k. \end{cases}$$

The fidelity metric for simultaneous operations is similarly to the shared qubit model expressed as:

$$C_{\text{neigh x sim}} = \prod_{(i,j)\in\mathcal{N}} F_{O_m}^n \left(\frac{2}{\frac{1}{F_{O_i}} + \frac{1}{F_{O_j}}} \right)^k , \tag{4}$$

here:

- F_{O_1} and F_{O_2}: Fidelities of the two simultaneously executed *two-qubit* gates.
- k: Amplification factor for number of simultaneous *two-qubit* operations.
- $\mathcal{N}$: Set of all neighboring edges affected by the simultaneous *two-qubit* gates.
- F_{O_m} : Fidelity of the single-qubit operation on qubit .
- n: Degree of influence, proportional to the number of shared qubits.

In addition to affecting the involved *two-qubit* gates, both models—**shared qubit** and **simultaneous execution**—also apply a lower-intensity noise term to nearby *single-qubit* gates to reflect indirect interference.

The proximity based model considers crosstalk when *two-qubit* gate operations are executed within a predefined physical distance on the hardware coupling map. Additionally, single-qubit gates performed on neighboring qubits are also affected by a weaker noise penalty. Let qubits i and j participate in one *two-qubit* gate operation, and qubits k and l in another. If the Euclidean distance between any qubit from the first operation and any qubit from the second operation is within a maximum radius

$r_{max} = 2$, crosstalk noise occurs. The crosstalk effect $C(i, j, k, l)$ is defined as follows:

$$C(i,j,k,l) = \begin{cases} 1 & \text{if } d(i,j,k,l) > r_{\max}, \\ C_{\text{prox}} & \text{if } d(i,j,k,l) \leq r_{\max}. \end{cases}$$

Here, $d(i, j, k, l)$ is the Euclidean distance between any qubit in the first gate operation and any qubit in the second gate operation. The fidelity metric for proximity based crosstalk is given by:

$$C_{\text{total}} = C_{\text{prox}} \times C_{\text{neigh}}, \tag{5}$$

with

$$C_{\text{prox}} = \left(\frac{2}{\frac{1}{F_{O_1}} + \frac{1}{F_{O_2}}} \right)^n, \qquad C_{\text{neigh}} = \prod_{m \in \mathcal{N}} F_{O_m}^n. \tag{6}$$

Here:

- $\mathcal{N}$: Set of all neighbouring qubits executing single-qubit gates.
- F_{O_m}: Fidelity of the single-qubit operation on qubit .
- k: Degree of influence, proportional to the number of affected neighbours.
- F_{O_1} and F_{O_2}: Fidelities of the two topologically close *two-qubit* gates.
- n: Amplification factor based on the proximity and number of affected qubits.

Other Noise Variants

We compare the effect of crosstalk to other prominent noise types— thermal relaxation and depolarisation noise. By conducting a systematic evaluation across these variants, we aim to quantify the relative significance of crosstalk noise in relation to its counterparts. **Thermal relaxation** noise is a non-unital, irreversible process describing the interaction between qubits and their environment as they evolve toward thermal equilibrium. A non-unital process does not preserve the identity operator, meaning it drives the systems toward a specific state—in this case, the ground state $|0\rangle$—rather than maintaining a maximally mixed state. It comprises two primary mechanisms:

- **Relaxation** T_1: The process by which a qubit exchanges energy with its environment, typically decaying from the excited state $|1\rangle$ to the ground state $|0\rangle$.
- **Dephasing** T_2: A process that leads to the decay of quantum coherence without necessarily changing the energy state of the qubit.

These effects are characterised by timescales $T_1^{(q)}$ and $T_2^{(q)}$ for each qubit q, where typically $T_2 \leq 2T_1$ [53]. Dephasing can occur both independently and in conjunction with relaxation [54]. To estimate fidelity loss due to thermal relaxation (amplitude damping and dephasing) in a quantum circuit, we consider the total time each qubit is active during the execution of the circuit. Let $t^{(q)}$ be the total accumulated gate duration for qubit q. Each qubit is characterised by a $T_1^{(q)}$ (energy relaxation) and $T_2^{(q)}$ (dephasing) time constant. The fidelity due to thermal relaxation for qubit q is modeled as:

$$F^{(q)}(t^{(q)}) = e^{-t_q/T_1^{(q)}} \cdot e^{-t_q/T_\phi^{(q)}}, \tag{7}$$

where the pure dephasing time $T_\phi^{(q)}$ is derived from:

$$\frac{1}{T_\phi^{(q)}} = \frac{1}{T_2^{(q)}} - \frac{1}{2T_1^{(q)}}. \tag{8}$$

The total circuit fidelity is estimated as the product of the fidelities over all qubits $q \in Q$:

$$F_{\text{total}} = \prod_{q \in Q} F^{(q)}(t^{(q)}). \tag{9}$$

This method provides a idealised, estimation of circuit fidelity under thermal noise, assuming Markovian relaxation and no gate errors.

Depolarisation noise affects quantum systems, by randomly replacing quantum states with the maximally mixed state. This process results in a complete loss of information about the original state, with the system transitioning to a uniform probability distribution over all possible states. Depolarisation noise for a quantum gate is modelled using

$$F_{\text{gate}} = F_{\text{initial}} \times (1 - p_{\text{depolarisation}}), \tag{10}$$

where F_{initial} is the initial fidelity of the gate, and $p_{\text{depolarisation}}$ is the depolarisation probability associated with the gate. The overall fidelity of a quantum circuit, incorporating all N gates in the transpiled circuit, can be expressed as:

$$F_{\text{circuit}} = \prod_{i=1}^{N} F_{\text{gate},i}, \tag{11}$$

where N is the total number of gates in the circuit [55, 56].

3.2 Compilation Layer

We evaluate the impact of compilation choices—such as optimisation level, qubit mapping, and routing strategy—on circuit fidelity. For the experiments exploring compilation variations, the fixed device configuration are set as follows: a coupling map with 128 qubits with connectivity densities set to [0.013895, 0.03, 0.05, 0.1, 0.3, 0.5, 0.8]. The basis gates are defined as ['x', 'y', 'z', 'rx' 'ry', 'rz', 'cx', 'cy'].

3.2.1 Metrics

The circuit mapping performance metrics are defined as follows: Gate overhead measures the relative increase in the total number of quantum gates after compilation:

$$G_{\text{overhead}} = \frac{G_{\text{after}} - G_{\text{before}}}{G_{\text{before}}}, \tag{12}$$

where G_{before} and G_{after} denote the number of gates before and after compilation. Similarly, depth overhead measures the relative change in circuit depth with D_{before}

and D_{after} representing pre- and post- compilation:

$$D_{\text{overhead}} = \frac{D_{\text{after}} - D_{\text{before}}}{D_{\text{before}}}. \tag{13}$$

In the next expression, F_{before} is the fidelity before compilation and F_{after} is the fidelity after compilation.

$$F_{\text{decrease}} = \frac{F_{\text{before}} - F_{\text{after}}}{F_{\text{before}}}. \tag{14}$$

To measure the impact on solution quality, we utilise the cost improvement metric—as introduced in Arline Benchmarks [57] —combining circuit depth, gate counts and gate fidelities. The cost improvement is defined as the ratio between the initial and final circuit costs. A higher ratio indicates the optimisation of reduced errors. This metric serves as the primary Figure of merit and builds upon the metric set previously introduced in [8].

$$C = \frac{C_{\text{in}}}{C_{\text{out}}}, \tag{15}$$

where:

$$C_{\text{in}} = -D_{\text{before}} \times \log K - N_{1q}^{\text{before}} \times \log F_{1q} - N_{2q}^{\text{before}} \times \log F_{2q}, \tag{16}$$

$$C_{\text{out}} = -D_{\text{after}} \times \log K - N_{1q}^{\text{after}} \times \log F_{1q} - N_{2q}^{\text{after}} \times \log F_{2q}. \tag{17}$$

F_{1q}, F_{2q} denote *one-qubit* and *two-qubit* gate fidelity (default: 0.9982, 0.9765), and N_{1q} respectively N_{2q} represent the number of single- and two-qubit gates before and after compilation. The decoherence fidelity per depth unit, K (default: 0.995), models the loss in fidelity due to idle time and circuit depth, *i.e.*, the longer a qubit remains active within a deep circuit, the greater the chance it suffers from decoherence effects. A value of $K = 0.995$ corresponds to a 0.5% fidelity loss per unit of depth [53].

Although this work primarily focuses on cost improvement metric as a comprehensive indicator for solution quality, we have also evaluated the other three metrics: fidelity decrease, gate overhead, and depth overhead. These supplementary results are included in the full set of experiments provided in our reproduction package (see Section 1). Gate fidelities are derived from Starmon-5 a superconducting quantum processor based on circuit quantum electrodynamics [58]. While our benchmark topologies are inspired by IBM's heavy-hex and Google's Sycamore architectures, we chose Starmon-5 gate fidelities as a more neutral and representative reference point, avoiding direct bias toward a specific commercial platform. Furthermore, Starmon-5 exhibits performance characteristics that fall within the typical range of superconducting qubit platforms making it suitable for general benchmarking without favouring any particular topology or vendor.

3.2.2 Optimisation Level

Qiskit provides four optimisation levels (0–3) that progressively reduce circuit depth and gate count during transpilation, with higher levels applying more aggressive transformations at the cost of longer compilation time [59].

3.2.3 Layout Methods

To map logical qubits to physical qubits on a quantum device, we evaluate three Qiskit layout methods: Trivial, Dense, and SABRE [4]. The *Trivial* method maps logical to physical qubits in numerical order without considering the hardware topology, making it computationally inexpensive but potentially leading to many SWAP operations. The *Dense* method aims to minimise SWAP operations by selecting a subset of physical qubits that closely matches the logical qubit structure. It analyses the device's coupling map to identify a densely connected group of qubits, reducing the distance between interacting qubits but requiring more computational effort. The *SABRE* method uses a heuristic to iteratively refine the mapping as the circuit progresses, effectively minimising SWAP gates and thus circuit depth, especially in larger, more complex circuits.

3.2.4 Qubit Routing Techniques

Qiskit offers different routing techniques to insert SWAPs when logical qubits are not physically adjacent [60]. We examine *Stochastic* and *SABRE* routing: Stochastic uses randomisation and heuristics to minimise circuit depth, while SABRE dynamically adjusts qubit placement to reduce SWAP overhead during execution.

3.2.5 Pass Manager Setups (Circuit Optimisation Passes)

We investigate five pass manager setups, each applying progressively more complex optimisation techniques to improve circuit fidelity.

- **Setup 1**: Optimises *single-qubit* gates and Clifford operations by simplifying commutation relationships.
- **Setup 2**: Decomposes *single-qubit* gates and cancels adjacent CNOT gates.
- **Setup 3**: Extends Setup 1 by removing diagonal gates before measurements.
- **Setup 4**: Decomposes *single-qubit* gates and applies commutative gate cancellation.
- **Setup 5**: Extends Setup 3, applies the Hoare optimiser, followed by (inverse) commutative gate cancellation.

While we utilise Qiskit as compilation, our proposed DSE technique is compiler-agnostic and can be readily applied within other quantum software platforms.

3.3 Benchmarks

For our experiments, we selected an extensive set of quantum circuits that span a wide range of computational paradigms and quantum algorithmic classes. The benchmark set includes circuits from well-known quantum algorithms and standard benchmarking tools such as Quantum Volume, Shor's algorithm, Quantum Fourier Transform (QFT), and circuits generating Greenberger-Horne-Zeilinger (GHZ) states. We include QFT and related circuits because they exhibit well-understood structural and noise sensitivity characteristics, making them suitable candidates for studying residual noise effects and error mitigation techniques [61]. This is particularly important as the transition from NISQ to FTQC is unlikely to be abrupt; intermediate regimes will exhibit partial

error correction and noise resilience, for which techniques like those evaluated in this study are highly relevant [27]. Additionally, we incorporated circuits for arithmetic operations, such as modular addition as well as structurally diverse randomly generated circuits to capture non-algorithmic patterns and stress-test compiler behaviour in less regular configurations. To further enhance the diversity and structural variety of our benchmark set, we ensured that the circuits represent distinct structural categories based on their circuit properties as defined in the paper by Bandic *et al.* [8]. This clustering is based on quantitative features rather than fixed algorithm classes. This approach guarantees that the benchmarks are structurally distinct, enabling more comprehensive comparisons. The benchmark set itself is derived from the qbench benchmark set [6, 7], which offers an extensive collection of quantum circuits, sourced from various platforms and written in different programming languages. To address contemporary challenges we added error correction algorithms like Bosonic, Repetition, Shor, Steane and Surface Code. These circuits enable performance evaluation in the context of FTQC and support the investigation of early-stage error correction and mitigation strategies applicable to near-term devices. The selected benchmarks enable a comprehensive evaluation of gate-based quantum computing performance across diverse circuit features and use cases, including optimisation, factoring and quantum simulation. The full benchmark suite of 30 circuits along with information on each algorithm, its purpose, and implementation specifics are included in the reproduction package referenced in Section 1.

4 Results

We commence with discussing our results along the two parts described in Section 3: Device and compilation characteristics.

4.1 Results of the Device Setup and Parameter Sweep

To isolate the impact of each noise model, we evaluate circuit fidelity and circuit depth under five distinct noise models, considering them individually and independently in an otherwise noiseless setting. The three crosstalk-related models include **shared qubit**, **simultaneous execution**, and **proximity based** (see Section 3.1.2), each representing plausible physical behaviours in multi-qubit gate execution systems. In addition, we include two widely studied standard noise models: **thermal relaxation** and **depolarisation**. Figure 2 compares the effects of all five noise models across varying connectivity densities for a representative set of benchmarks (see Section 3.3). Note that the fidelities are model-estimated and may introduce bias or approximation artifacts, particularly in regimes with correlated errors. We observe that devices with higher connectivity density (right-hand side of the x-axis) are generally more resilient to the simultaneous execution model, as parallel gate execution causes less disruption when the qubit layout is less constrained. In contrast, the shared qubit model highlights a trade-off where more densely connected devices can amplify interference and common qubits. The proximity based model performs more consistently across connectivity densities, indicating its relative independence from overall connectivity density. No single crosstalk model dominates across all benchmarks. Which model performs

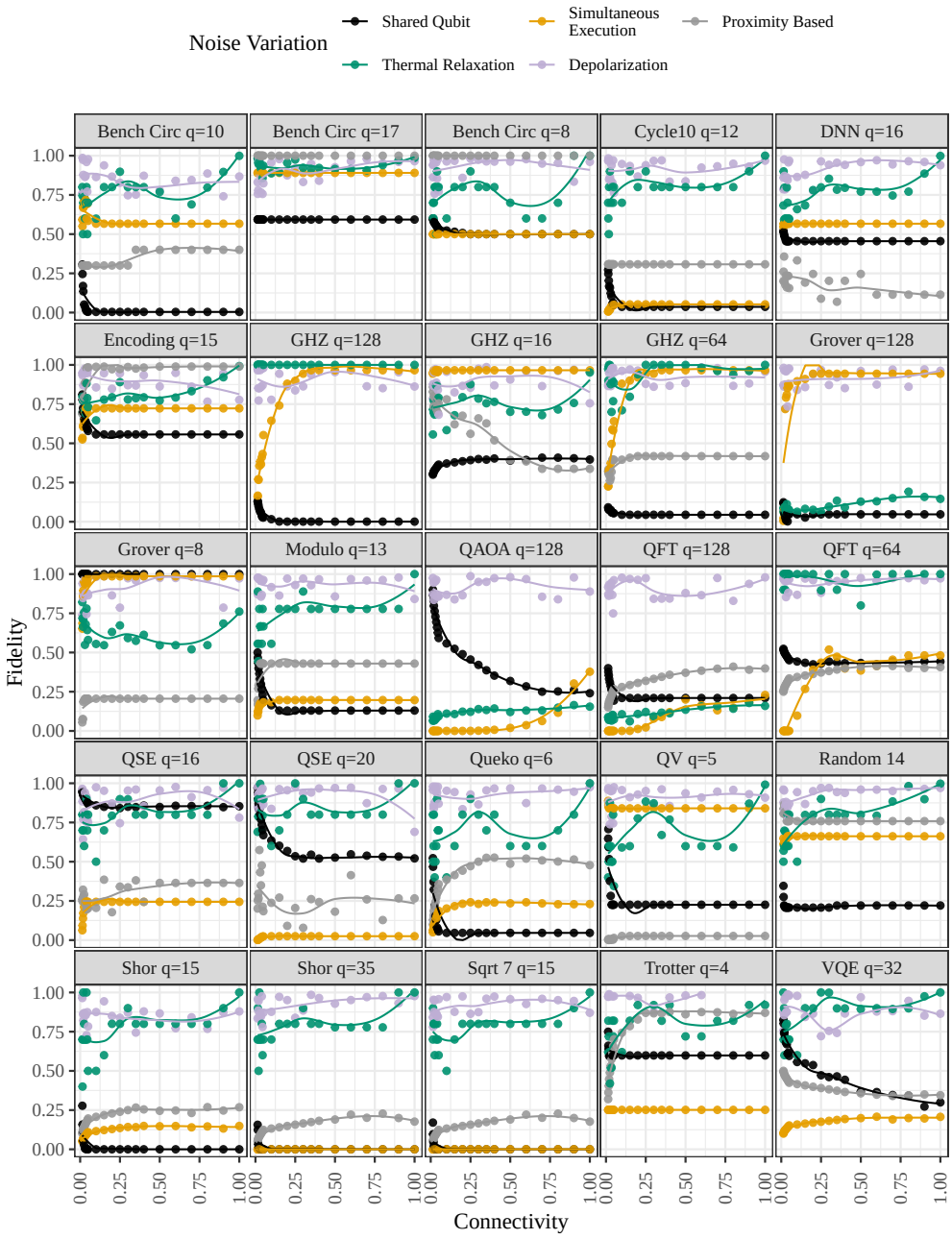


Fig. 2: Illustration of fidelity vs. connectivity across benchmarks as facets comparing three crosstalk models, thermal relaxation, and depolarisation noise for the heavy-hex back-end topology.

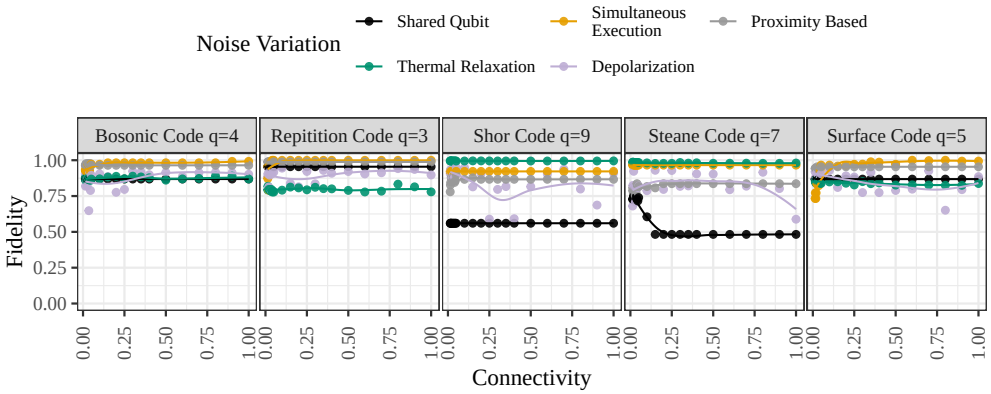


Fig. 3: Extension of Figure 2 with error correcting codes.

best is highly benchmark-specific, reinforcing the need for tailored noise mitigation. That said, the shared qubit model nearly consistently results in the most severe degradation, highlighting it as a key target for error mitigation in quantum architectures with slightly higher connectivity density. Beyond crosstalk, quantum circuits are also affected by thermal relaxation and depolarisation noise. While thermal relaxation and depolarisation are generally less harmful to fidelity than crosstalk—particularly compared to the shared qubit model—exceptions do exist. Notably, in large-qubit benchmarks such as *QAOA q=128*, *Grover q=128*, and *QFT q=128*, thermal relaxation causes greater fidelity loss than any of the other noise variants. Error-correcting codes are essential for achieving fault-tolerant quantum computing (FTQC). To evaluate their resilience to crosstalk, we analyse the fidelity of well-known error-correcting codes, as shown in Figure 3. Proximity based and simultaneous execution models maintain stable fidelities for all codes when connectivity exceeds 0.1. However, the shared qubit model shows a fidelity drop with increasing connectivity, particularly impacting steane code. Shor code performs significantly worse under the shared qubit model than the other two models. These results highlight the need for tailored error-correcting strategies in FTQC, that consider both noise models and hardware topology. Effectively mitigating crosstalk remains a complex challenge. For example, Qiskit's now removed `CrosstalkAdaptiveSchedule` [62] aimed to reduce crosstalk by locally adapting gate scheduling. However, it was found that such local optimisations can have unpredictable global effects on circuit performance by increasing exposure to other noise sources. This illustrates the difficulty of balancing different noise mechanisms and underscores the need for community-wide standards to define how crosstalk is modeled, measured, and mitigated. Nevertheless, new techniques like twirling [63] and dynamical decoupling [64] are emerging as promising techniques to reduce crosstalk error. Tunable couplers [65] have also been proposed as a hardware-level solution to suppress crosstalk by enabling dynamic control over qubit-qubit interactions. To further investigate the spatial aspects of crosstalk, Figure 4 and 5 compare the heavy-hex and Sycamore layouts, each consisting of 143 and 144 qubits, respectively. Positive values indicate better performance on heavy-hex, while negative values favour Sycamore.

Overall, Sycamore shows lower robustness to crosstalk across most models and circuits, with few exceptions. This highlights the importance of back-end-specific co-design. This is particularly evident when considering that Qiskit's transpiler has been shown to perform especially well on the heavy-hex architecture [47, 66]. However, the advantages of adopting a different topology diminish around a connectivity density of 0.8. For certain smaller benchmarks, this effect appears even earlier.

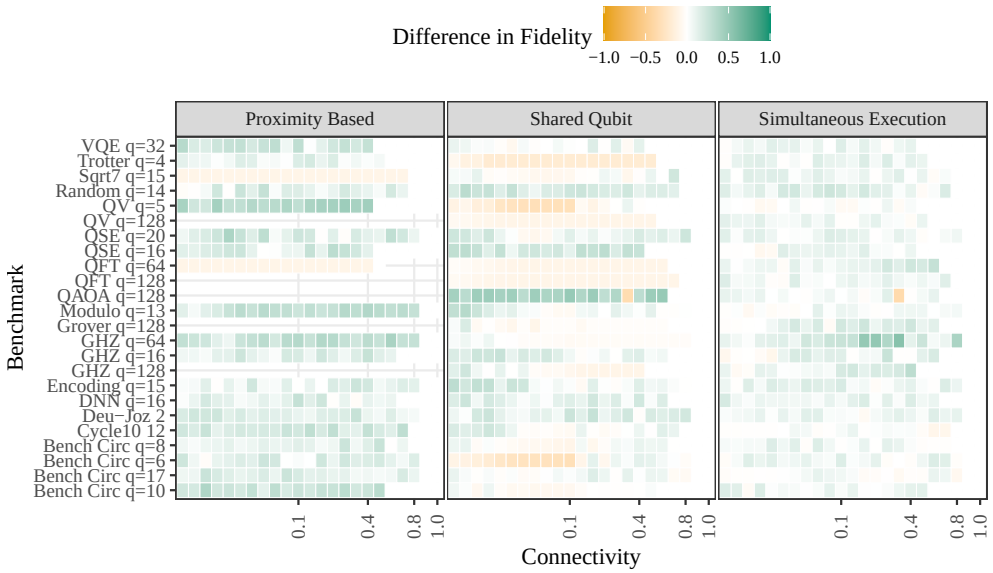


Fig. 4: Difference in fidelity between heavy-hex and the grid-like topology of the Sycamore chip across benchmarks and connectivity.

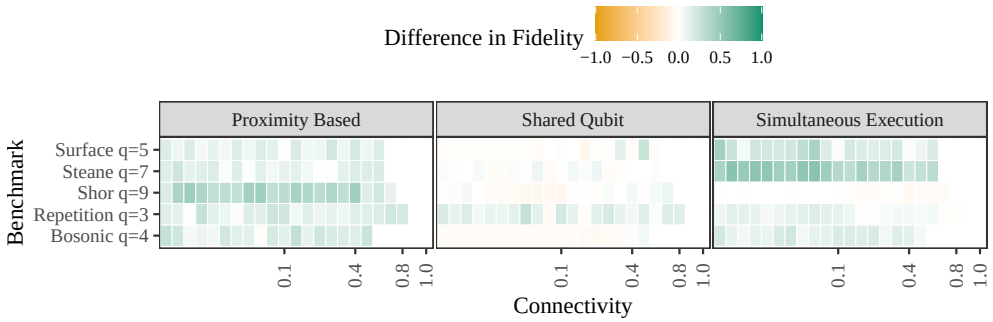


Fig. 5: Extension of Figure 4 with error correcting codes.

To assess further architectural impacts on performance, we examine how different back-end sizes influence fidelity and circuit depth. As seen in 6, different back-end sizes

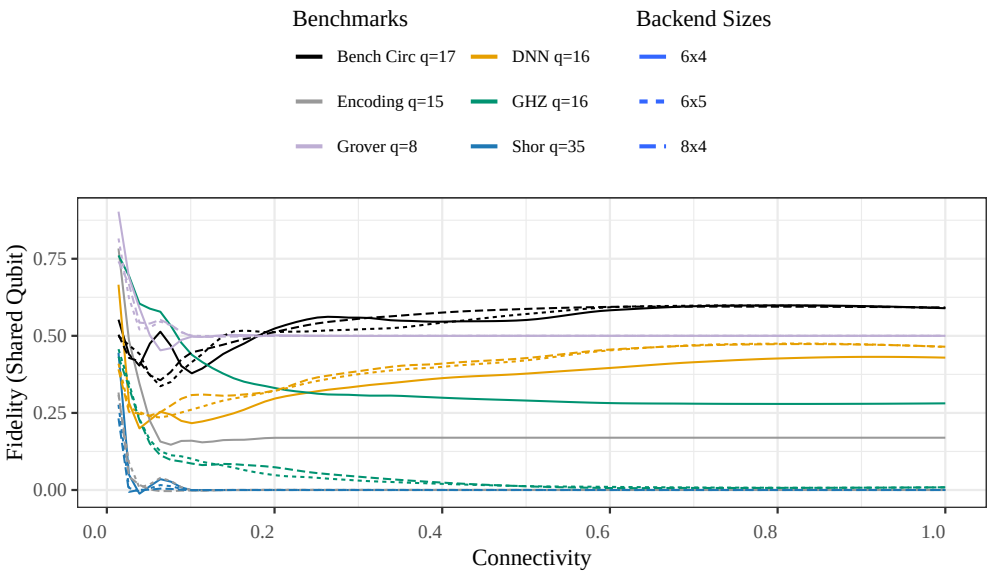


Fig. 6: Impact of back-end size fidelity as a function of connectivity across benchmarks for crosstalk version shared qubit.

across different optimisation levels $(0-3)$ have little to no consistent effect on fidelity across benchmarks. In certain cases, larger back-ends even exhibit a lower fidelity, suggesting that scaling alone is insufficient— if not a disadvantage—for mitigating crosstalk-induced loss.

The results in Figure 7 reinforce the observation that back-end sizes exerts minimal influence on fidelity. Under the simultaneous execution crosstalk model, increasing the back-end size does not yield noticeable improvements in fidelity. While some minor fluctuations are visible, the overall trend indicates that fidelities remain largely unaffected by back-end size scaling. Figure 8 indicates that the depth of most circuits converges near a connectivity of 0.3, regardless of back-end size. This convergence point is also consistent with findings in optimisation-focused problems, as reported by Safi *et al.* [12]. While most larger benchmarks also tend to stabilise around this connectivity threshold, some continue to exhibit slight reductions in circuit depth beyond a connectivity of 0.3. However, these gains are marginal and often come at the cost of increased compilation time. In general, back-end size has a limited impact on depth and fidelity. Selecting a back-end size that matches the scale of the target problem helps strike a balance between computational efficiency and resource allocation.

Overall, the results presented in this section indicate that higher connectivity density improves resilience to crosstalk noise across most models. Nonetheless, fidelity in the shared qubit model can degrade as connectivity increases. The grid-like topology of the Sycamore chip exhibits lower robustness compared to its counterpart, and both fidelity and circuit depth converge independent of back-end size.

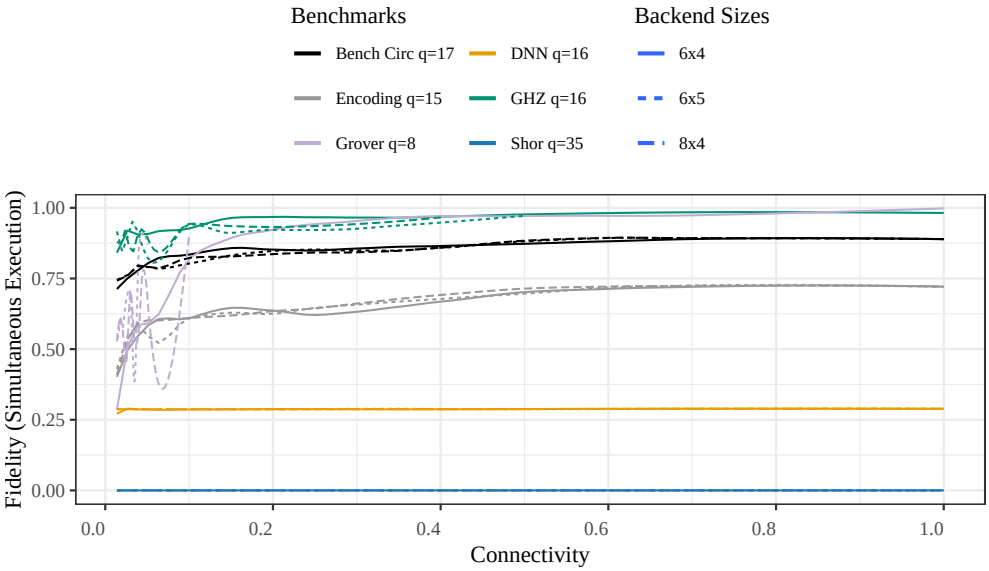


Fig. 7: Impact of back-end size fidelity as a function of connectivity across benchmarks for crosstalk version simultaneous execution.

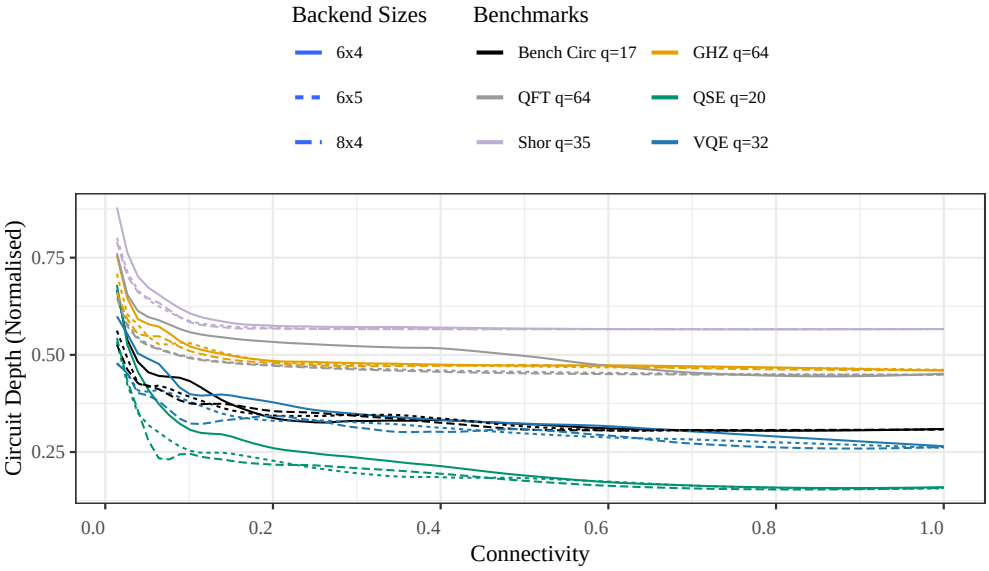


Fig. 8: Impact of back-end size on normalised circuit depth as a function of connectivity across benchmarks.

4.2 Results of the Compilation Parameter Sweep

Having examined device-level parameters, we now focus on compilation parameters. We sweep the full set of benchmarks across all combinations as defined in Section 3.

As shown in Figure 9, we evaluate each benchmark using the cost improvement metric introduced in section methodology (see Equations (15), (16), and (17)), where higher values indicate better performance. Across most benchmarks, the SABRE and Dense layout provide the best improvements, though Trivial performs well for specific cases like GHZ states. The Dense layout shows variable results, often outperforming Trivial and sometimes matching SABRE. While SABRE yields high-quality mappings it also has the longest compilation time. Moreover, its performance deteriorates for variational algorithms such as VQE, where repeated circuit execution favours stable and noise-aware mappings [67]. Therefore, its use should be carefully considered in light of both time constraints and the specific characteristics of the problem.

Regarding qubit routing technique (see Figure 10), the SABRE router consistently outperforms the Stochastic approach, especially for large and structurally complex circuits such as QFT. However, this advantage has trade-off's. The SABRE method typically requires longer compilation time (3.2.4), and for smaller or highly regular circuits, the improvement over Stochastic routing does not justify the additional effort. However unlike the layout methods, where the optimal choice is problem dependent, the comparison of qubit routing techniques reveals a clear trend.

Figure 11 illustrates the impact of various optimisation levels on cost improvement. As optimisation level 2 rarely offers significant advantages over level 1, the additional compile-time overhead and increased variability make optimisation level 1 the preferred choice across most benchmarks.

Although analysing layout method, qubit routing technique, and optimisation level separately is useful, their effects are often interdependent. Certain combinations can perform notably better or worse due to synergies between them. Figure 12 presents the frequency of best and worst-performing configurations across all benchmarks. The most successful combinations typically use SABRE as a qubit routing technique with optimisation level 2, with SABRE|2|SABRE performing best overall, followed by SABRE|2|Trivial and SABRE|2|Dense. The worst results often come from Stochastic|0|SABRE, further validating that strong components alone are not enough—effective performance requires aligned configurations across layout method, qubit routing technique, and optimisation level. Notably, even SABRE|0|SABRE performs poorly, underlining the significant role of the optimisation level. Figure 13 compares the performance of the best and worst-performing initial configurations according to the cost improvement metric across our benchmark algorithms and different connectivity densities. The y-axis indicates the connectivity density of the hardware topology, while the x-axis represents successive passes introducing additional circuit optimisations (as discussed in Section 3.2.5) aimed at reducing depth and gate overhead, as well as improving fidelity. Configurations that already incorporate an effective combination of layout method, optimisation level, and qubit routing technique generally do not benefit from additional circuit transformation passes. Even the worst-performing initial configurations tend to benefit minimally from additional passes, showing that further complexity does not necessarily yield in proportional

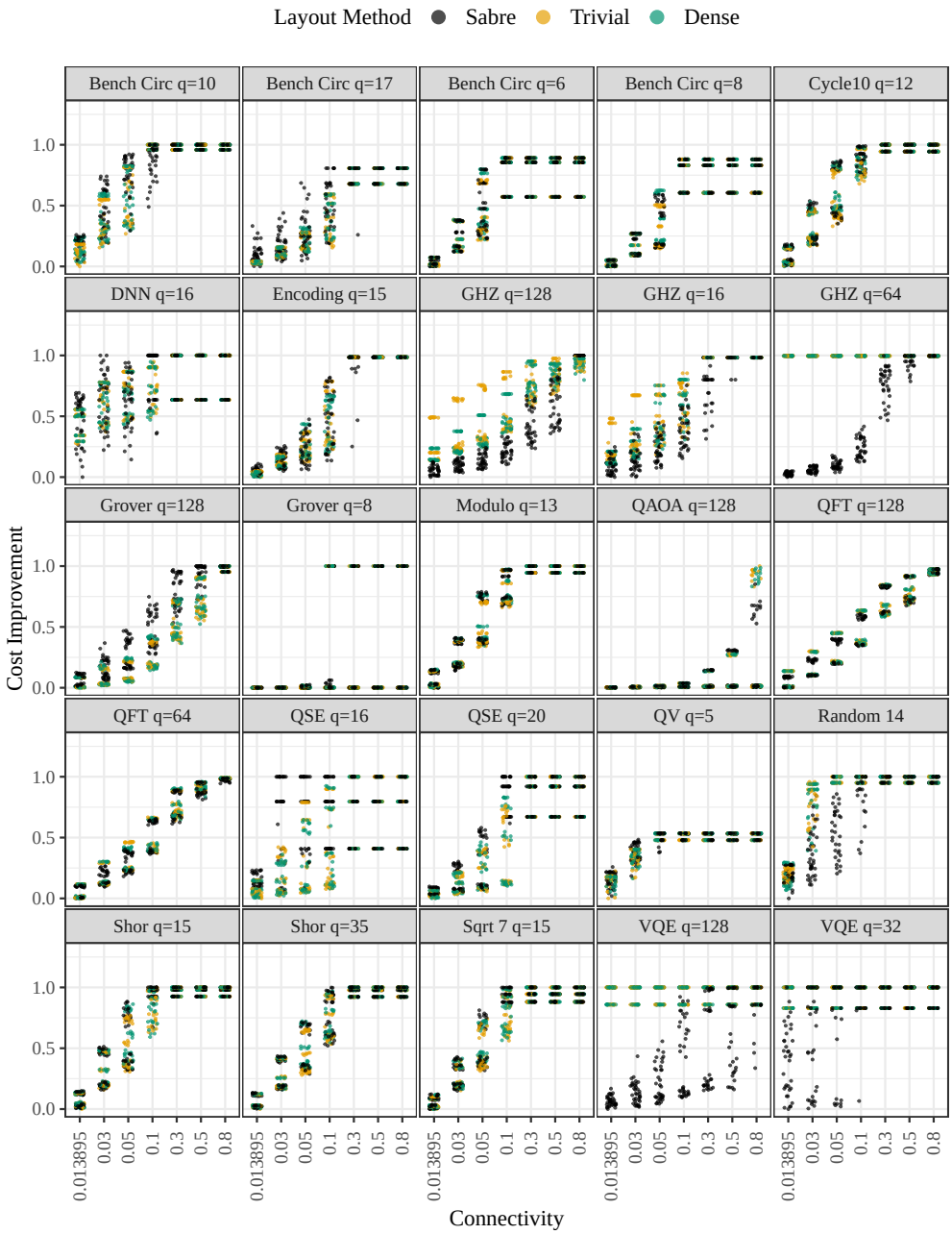


Fig. 9: Cost improvement (see Equations (15) (16), (17)) vs. connectivity across benchmarks for three layout methods.

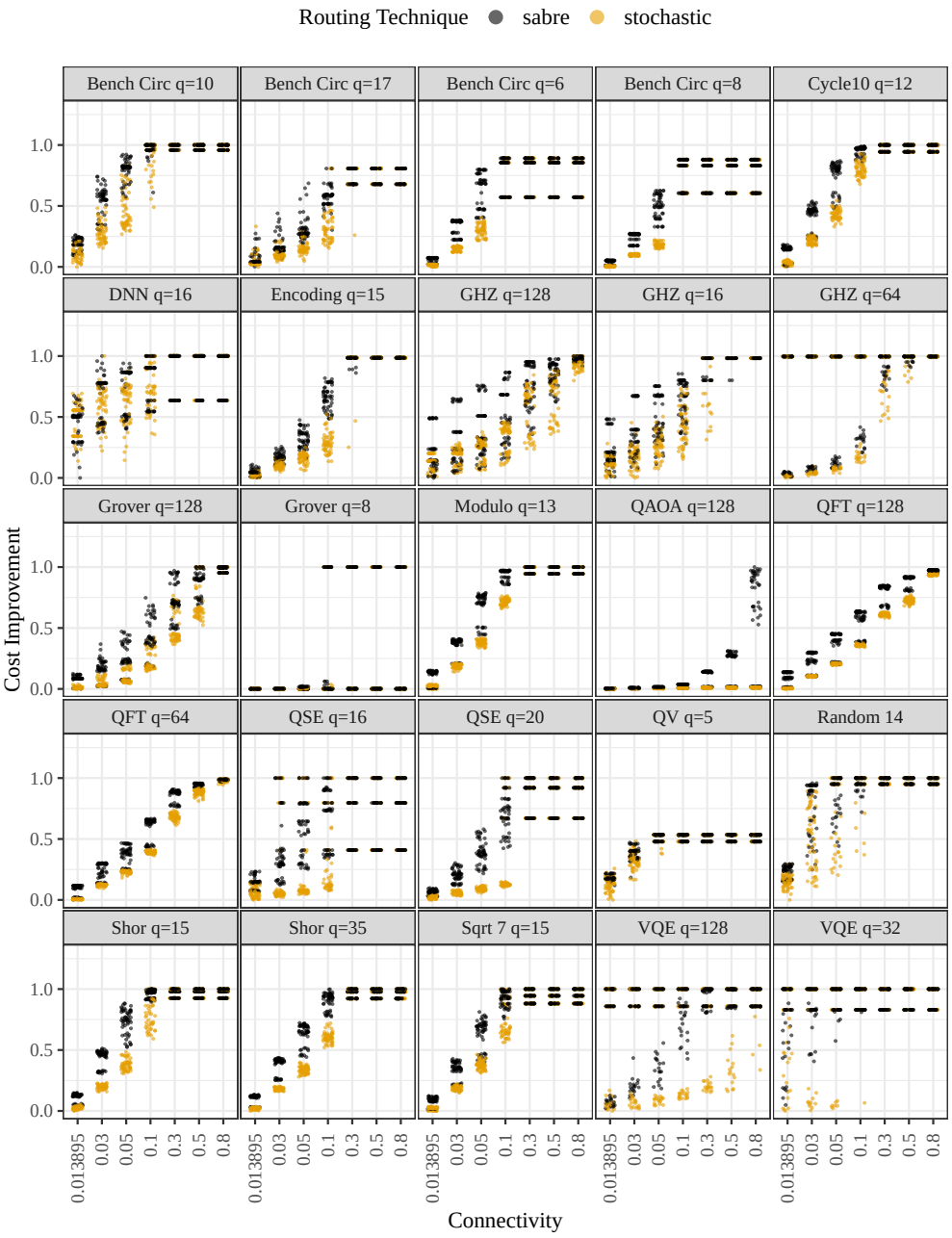


Fig. 10: Illustration of cost improvement as defined in Equations (15) (16), (17) vs. connectivity across benchmarks as facets comparing two routing techniques.

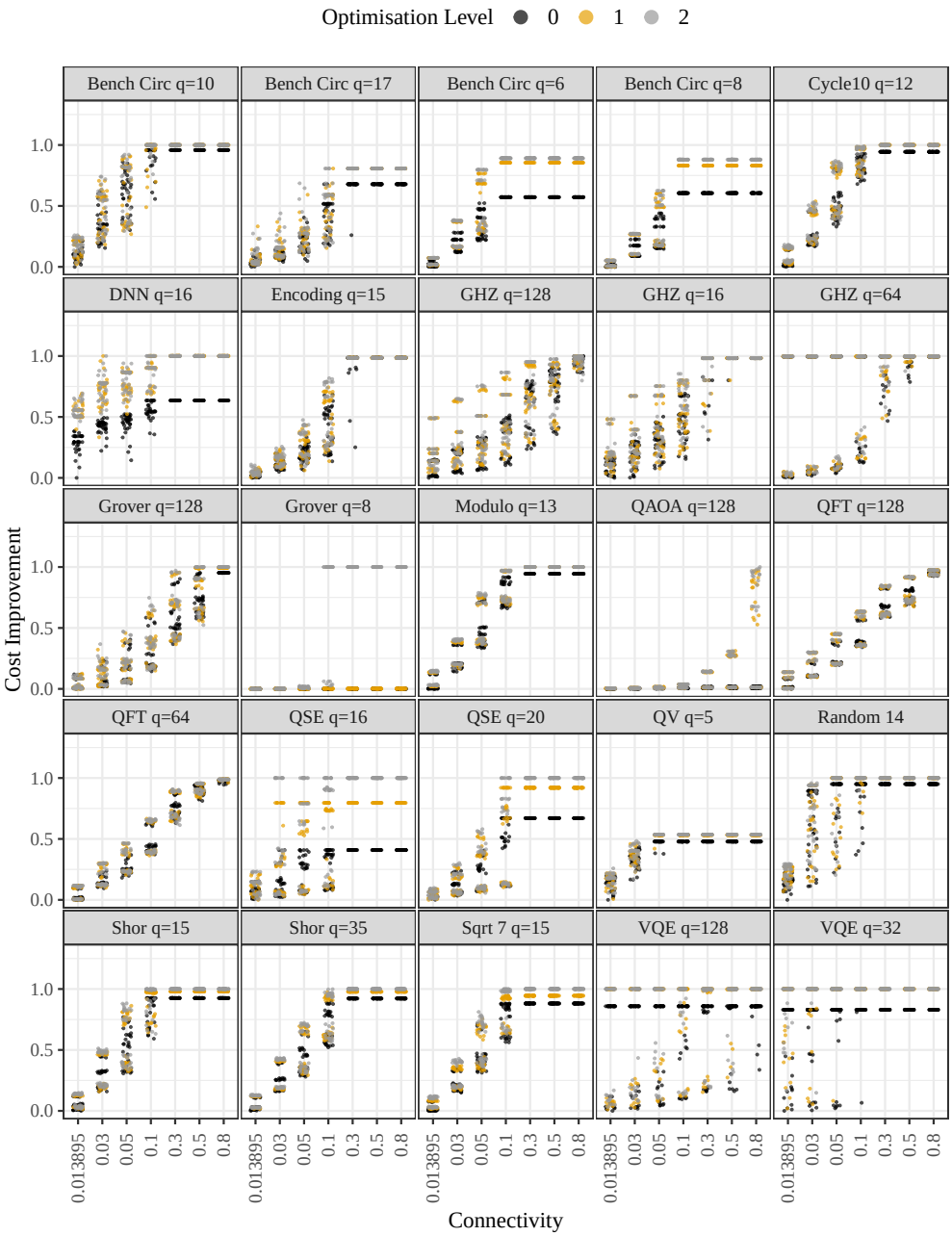


Fig. 11: Illustration of cost improvement as defined in Equations (15) (16), (17) vs. connectivity across benchmarks as facets comparing three optimisation levels.

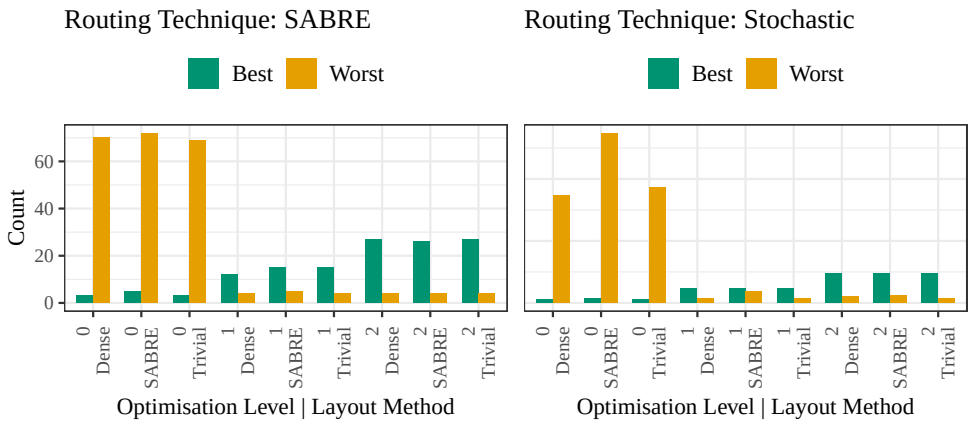


Fig. 12: Frequency of best and worst-performing combinations, based on cost improvement, evaluated across qubit routing technique, layout method, and optimisation level over all benchmarks and connectivity densities

gains. Ultimately, the most significant factor in improving circuit performance is connectivity: the more complex the circuit, the more essential high connectivity becomes. Generally, in combination SABRE consistently delivers the best layout and routing performance but has higher compile times. Thus optimisation level 1 offers a good compromise.

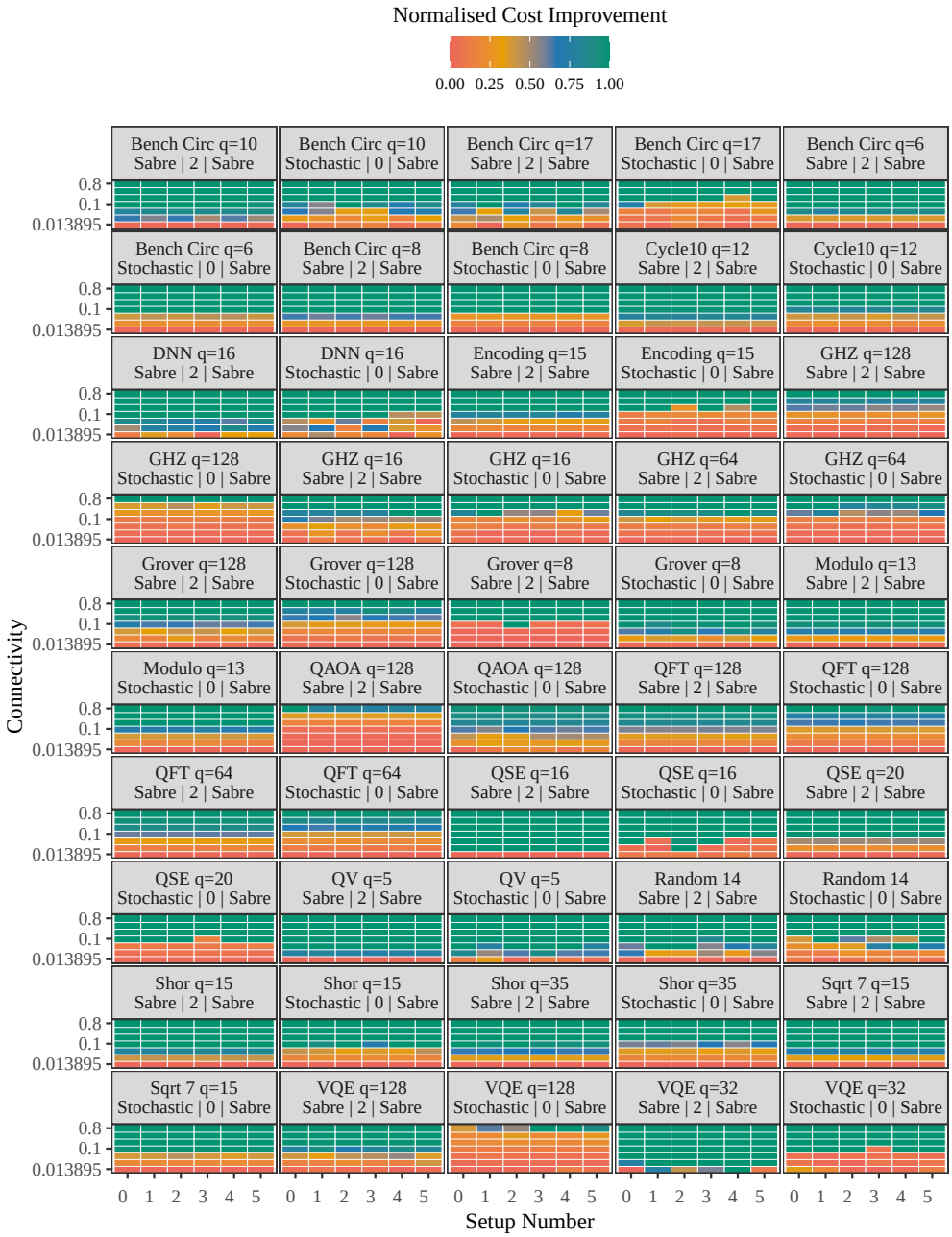


Fig. 13: Cost improvement (see Equations 15 16, 17) vs. connectivity across benchmarks and compilation setups. Improvements are measured relative to the baseline configuration with no additional improvements.

5 Conclusion & Outlook

Our study highlights the importance of a holistic approach to optimising quantum circuit performance, emphasising the interplay between hardware characteristics and compilation strategies. The results demonstrate that both device attributes and compilation choices impact circuit performance. We observe that different crosstalk models affect fidelity in distinct ways, underlining the importance of tailoring error mitigation strategies to the specific noise characteristics of the device. Notably, the shared qubit model consistently exhibits the most detrimental impact on fidelity, while the simultaneous execution model shows a more stable behavior, especially when connectivity exceeds 0.3. However, full connectivity is not required as fidelity converges much sooner. Additionally, back-end-specific optimisations play a crucial role. Heavy-hex topology consistently shows better crosstalk resilience compared to the grid-like topology of the Sycamore chip. In contrast larger back-end sizes have a negligible impact on circuit fidelity. Among compilation strategies, SABRE-based routing and layout methods with moderate optimisation levels yield the best compromise between performance and compile time. Notably, increasing system complexity through excessive optimisation does not lead to proportional improvements in performance, as shown in Figure 13. This suggests diminishing returns beyond setting good parameters for qubit routing, layout method and optimisation level. Importantly, optimal performance emerges from aligning all parameter options. Our findings support the need for end-to-end co-design: quantum system performance is maximised when noise-aware hardware selection, connectivity, and compilation strategies are considered jointly. Key takeaways include:

- Connectivity density is the dominant factor influencing both fidelity and circuit depth.
- The shared qubit crosstalk model presents the greatest fidelity challenge and should be prioritised in mitigation strategies.
- back-end size scaling offers limited to no benefit, suggesting resource-aware deployment is preferable.
- Among compilation parameters, SABRE routing combined with optimisation level 1 provides the best cost-performance balance.
- Adding more circuit transformations beyond already aligned configurations yields marginal or no gains, reinforcing the value of strategic simplicity.

Building on these insights, hardware developers should address some issues at the design level. It is beneficial to establish a standardised definition of crosstalk to ensure more consistent benchmarking. Adaptive compilation strategies that respond to both circuit structure and device properties hold promise for scalable quantum computing. Ultimately, aligning hardware innovations with software design will be the key to achieving the full potential of quantum technologies. In future work, it would be valuable to investigate the trade-offs between circuit performance and the additional compilation time by more advanced circuit improvements.

Acknowledgments

MB thanks H. van Someren, P. le Henaff, R. Turrado Camblor and A. Hesam for insightful discussions. SF and MB would like to acknowledge funding from Intel Corporation. CGA is thankful for support from the QuantERA grant EQUIP with the grant number PCI2022-133004, funded by Agencia Estatal de Investigación, Ministerio de Ciencia e Innovación, Gobierno de España, MCIN/AEI/10.13039/501100011033, and by the European Union NextGenerationEU/PRTR. Also from the Ministry for Digital Transformation and of Civil Service of the Spanish Government through the QUANTUM ENIA project call - Quantum Spain project, and by the European Union through the Recovery, Transformation and Resilience Plan - NextGenerationEU within the framework of the Digital Spain 2026 Agenda. This work was supported by the German Federal Ministry of Education and Research (BMBF), funding program 'quantum technologies—from basic research to market', grant numbers 13N16092 and 13N16093. This work used the Dutch national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-5822. WM acknowledges support by the High-Tech Agenda of the Free State of Bavaria.

References

[1] Li, G., Wu, A., Shi, Y., Javadi-Abhari, A., Ding, Y., Xie, Y.: On the co-design of quantum software and hardware. In: Proceedings of the Eight Annual ACM International Conference on Nanoscale Computing and Communication. NANOCOM '21. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3477206.3477464

[2] Tomesh, T., Martonosi, M.: Quantum codesign. IEEE Micro **41**(5), 33–40 (2021)

[3] Wintersperger, K., Safi, H., Mauerer, W.: Qpu-system co-design for quantum hpc accelerators. In: Schulz, M., Trinitis, C., Papadopoulou, N., Pionteck, T. (eds.) Proceedings of the 35th GI/ITG International Conference on the Architecture of Computing Systems, pp. 100–114. Gesellschaft für Informatik, Heilbronn, Germany (2022). https://doi.org/10.1007/978-3-031-21867-5_7

[4] IBM Quantum: Transpiler optimization level. https://docs.quantum.ibm.com/guides/transpile

[5] Bandic, M., Zarein, H., Alarcon, E., Almudever, C.G.: On structured design space exploration for mapping of quantum algorithms. In: 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS), pp. 1–6 (2020). IEEE

[6] Bandic, M., Almudever, C.G., Feld, S.: Interaction graph-based characterization of quantum benchmarks for improving quantum circuit mapping techniques. Quantum Machine Intelligence **5**(2), 40 (2023)

[7] Bandic, M., Paraskevopulous, N.: qBench benchmark suite. https://github.com/QML-Group/qbench (2021)

[8] Bandic, M., Henaff, P., Ovide, A., Escofet, P., Ben Rached, S., Santiago, R., Someren, H., Abadal, S., Alarcón, E., Almudever, C., Feld, S.: Profiling quantum circuits for their efficient execution on single- and multi-core architectures. Quantum Science and Technology **10** (2025) https://doi.org/10.1088/2058-9565/ada180

[9] Sarovar, M., Proctor, T., Rudinger, K., Young, K., Nielsen, E., Blume-Kohout, R.: Detecting crosstalk errors in quantum information processors. Quantum **4**, 321 (2020)

[10] Giurgica-Tiron, T., Hindy, Y., LaRose, R., Mari, A., Zeng, W.J.: Digital zero noise extrapolation for quantum error mitigation. In: 2020 IEEE International Conference on Quantum Computing and Engineering (QCE), pp. 306–316. IEEE, Denver, CO, USA (2020). https://doi.org/10.1109/qce49297.2020.00045

[11] Mauerer, W., Scherzinger, S.: 1-2-3 reproducibility for quantum software experiments. Q-SANER@IEEE International Conference on Software Analysis, Evolution and Reengineering (2022)

[12] Safi, H., Wintersperger, K., Mauerer, W.: Influence of hw-sw-co-design on quantum computing scalability. In: 2023 IEEE International Conference on Quantum Software (QSW), pp. 104–115 (2023). https://doi.org/10.1109/QSW59989.2023.00022

[13] Mauerer, S.T.W.: Predict and conquer: Navigating algorithm trade-offs with quantum design automation. In: Proceedings of IEEE QCE (2025)

[14] Bandic, M., Feld, S., Almudever, C.G.: Full-stack quantum computing systems in the nisq era: algorithm-driven and hardware-aware compilation techniques. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1–6 (2022). IEEE

[15] Murali, P., Baker, J.M., Javadi-Abhari, A., Chong, F.T., Martonosi, M.: Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers. In: International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 1015–1029 (2019)

[16] Li, G., Ding, Y., Xie, Y.: Towards efficient superconducting quantum processor architecture design. In: Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 1031–1045 (2020)

[17] Murali, P., Linke, N.M., Martonosi, M., Abhari, A.J., Nguyen, N.H., Alderete, C.H.: Full-stack, real-system quantum computer studies: Architectural comparisons and design insights. In: 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA), pp. 527–540 (2019). IEEE

[18] Quetschlich, N., Burgholzer, L., Wille, R.: Mqt predictor: Automatic device selection with device-specific circuit compilation for quantum computing. ACM Transactions on Quantum Computing **6**(1) (2025) https://doi.org/10.1145/3673241

[19] Amy, M., Maslov, D., Mosca, M., Roetteler, M.: A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **32**(6), 818–830 (2013) https://doi.org/10.1109/tcad.2013.2244643

[20] Preskill, J.: Quantum computing in the nisq era and beyond. Quantum **2**, 79 (2018)

[21] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S.C., Endo, S., Fujii, K., McClean, J.R., Mitarai, K., Yuan, X., Cincio, L., et al.: Variational quantum algorithms. Nature Reviews Physics, 1–20 (2021)

[22] Farhi, E., Goldstone, J., Gutmann, S.: A Quantum Approximate Optimization Algorithm (2014). https://arxiv.org/abs/1411.4028

[23] Thelen, S., Safi, H., Mauerer, W.: Approximating under the influence of quantum noise and compute power. In: Proceedings of WIHPQC@IEEE QCE (2024). https://doi.org/10.1109/QCE60285.2024.10291

[24] Temme, K., Bravyi, S., Gambetta, J.M.: Error mitigation for short-depth quantum circuits. Physical Review Letters **119**(18) (2017) https://doi.org/10.1103/physrevlett.119.180509

[25] Shor, P.W.: Fault-tolerant quantum computation. In: Proceedings of 37th Conference on Foundations of Computer Science, pp. 56–65 (1996). https://doi.org/10.1109/SFCS.1996.548464

[26] Fowler, A.G., Mariantoni, M., Martinis, J.M., Cleland, A.N.: Surface codes: Towards practical large-scale quantum computation. Physical Review A **86**(3) (2012) https://doi.org/10.1103/physreva.86.032324

[27] Roffe, J.: Quantum error correction: an introductory guide. Contemporary Physics **60**(3), 226–245 (2019) https://doi.org/10.1080/00107514.2019.1667078

[28] Niu, S., Suau, A., Staffelbach, G., Todri-Sanial, A.: A hardware-aware heuristic for the qubit mapping problem in the nisq era. IEEE Transactions on Quantum Engineering **1**, 1–14 (2020) https://doi.org/10.1109/TQE.2020.3026544

[29] Park, S., Kim, D., Kweon, M., Sim, J.-Y., Kang, S.: A fast and scalable qubit-mapping method for noisy intermediate-scale quantum computers. In: Proceedings of the 59th ACM/IEEE Design Automation Conference. DAC '22, pp. 13–18. Association for Computing Machinery, New York, NY, USA (2022).

https://doi.org/10.1145/3489517.3530402

[30] Li, G., Ding, Y., Xie, Y.: Tackling the qubit mapping problem for NISQ-era quantum devices. In: International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 1001–1014 (2019)

[31] Almudever, C.G., Lao, L., Wille, R., Guerreschi, G.G.: Realizing quantum algorithms on real quantum computing devices. In: 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 864–872 (2020). IEEE

[32] Pozzi, M.G., Herbert, S.J., Sengupta, A., Mullins, R.D.: Using reinforcement learning to perform qubit routing in quantum compilers. ACM Transactions on Quantum Computing **3**(2) (2022) https://doi.org/10.1145/3520434

[33] Zulehner, A., Paler, A., Wille, R.: An efficient methodology for mapping quantum circuits to the IBM QX architectures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2018)

[34] Wagner, F., Bärmann, A., Liers, F., Weissenbäck, M.: Improving quantum computation by optimized qubit routing. Journal of Optimization Theory and Applications, 1–34 (2023)

[35] Bandic, M., Prielinger, L., Nußlein and, J., Ovide, A., Rodrigo, S., Abadal, S., Someren, H.v., Vardoyan, G., Alarcon, E., Almudever, C.G., Feld, S.: Mapping quantum circuits to modular architectures with qubo. In: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), pp. 790–801. IEEE Computer Society, Los Alamitos, CA, USA (2023). https://doi.org/10.1109/QCE57702.2023.00094

[36] Ovide, A., Rodrigo, S., Bandic, M., Van Someren, H., Feld, S., Abadal, S., Alarcon, E., Almudever, C.G.: Mapping quantum algorithms to multi-core quantum computing architectures. Proceedings of the ISCAS '23 (2023)

[37] Cowtan, A., Dilkes, S., Duncan, R., Krajenbrink, A., Simmons, W., Sivarajah, S.: On the qubit routing problem. In: 14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019) (2019). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik

[38] Schlosshauer, M.: Quantum decoherence. Physics Reports **831**, 1–57 (2019) https://doi.org/10.1016/j.physrep.2019.10.001

[39] Harper, R., Flammia, S.T., Wallman, J.J.: Efficient learning of quantum noise. Nature Physics **16**(12), 1184–1188 (2020) https://doi.org/10.1038/s41567-020-0992-8

[40] Rosa, E.C.R., Duzzioni, E.I., Santiago, R.: Optimizing gate decomposition for high-level quantum programming. Quantum **9**, 1659 (2025) https://doi.org/10.

22331/q-2025-03-12-1659

[41] Guerreschi, G.G., Park, J.: Two-step approach to scheduling quantum circuits. Quantum Science and Technology **3**(4), 045003 (2018)

[42] Siraichi, M.Y., Santos, V.F.d., Collange, C., Pereira, F.M.Q.: Qubit allocation. In: Proceedings of the 2018 International Symposium on Code Generation and Optimization. CGO 2018, pp. 113–125. Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3168822

[43] Pant, M., Krovi, H., Towsley, D., Tassiulas, L., Jiang, L., Basu, P., Englund, D., Guha, S.: Routing entanglement in the quantum internet. npj Quantum Information **5**(1), 25 (2019)

[44] Gyongyosi, L., Imre, S.: Routing space exploration for scalable routing in the quantum internet. Scientific reports **10**(1), 1–15 (2020)

[45] Mariella, N., Zhuk, S.: A doubly stochastic matrices-based approach to optimal qubit routing. Quantum Information Processing **22**(7), 264 (2023)

[46] Karuppasamy, K., Puram, V., Johnson, S., Thomas, J.: A comprehensive review of quantum circuit optimization: Current trends and future directions. Quantum Reports **7**, 2 (2025) https://doi.org/10.3390/quantum7010002

[47] IBM Quantum: The IBM Quantum heavy hex lattice. https://research.ibm.com/blog/heavy-hex-lattice

[48] Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J.C., Barends, R., Biswas, R., Boixo, S., Brandao, F.G., Buell, D.A., *et al.*: Quantum supremacy using a programmable superconducting processor. Nature **574**(7779), 505–510 (2019)

[49] Paraskevopoulos, N., Sebastiano, F., Almudever, C.G., Feld, S.: Spinq: Compilation strategies for scalable spin-qubit architectures. ACM Transactions on Quantum Computing **5**(1), 1–36 (2023) https://doi.org/10.1145/3624484

[50] Murali, P., Mckay, D.C., Martonosi, M., Javadi-Abhari, A.: Software mitigation of crosstalk on noisy intermediate-scale quantum computers. In: Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems. ASPLOS '20, pp. 1001–1016. Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3373376.3378477

[51] Zhao, P., Linghu, K., Li, Z., Xu, P., Wang, R., Xue, G., Jin, Y., Yu, H.: Quantum crosstalk analysis for simultaneous gate operations on superconducting qubits. PRX Quantum **3**, 020301 (2022) https://doi.org/10.1103/PRXQuantum.3.020301

[52] Undseth, B., Xue, X., Mehmandoost, M., Rimbach-Russ, M., Eendebak, P.T., Samkharadze, N., Sammak, A., Dobrovitski, V.V., Scappucci, G., Vandersypen, L.M.K.: Nonlinear response and crosstalk of electrically driven silicon spin qubits. Phys. Rev. Appl. **19**, 044078 (2023) https://doi.org/10.1103/PhysRevApplied.19.044078

[53] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, ??? (2010)

[54] Fowler, A.G., Mariantoni, M., Martinis, J.M., Cleland, A.N.: Surface codes: Towards practical large-scale quantum computation. Phys. Rev. A **86**, 032324 (2012) https://doi.org/10.1103/PhysRevA.86.032324

[55] Cross, A.W., Magesan, E., Bishop, L.S., Smolin, J.A., Gambetta, J.M.: Scalable randomised benchmarking of non-clifford gates. npj Quantum Information **2**(1) (2016) https://doi.org/10.1038/npjqi.2016.12

[56] Georgopoulos, K., Emary, C., Zuliani, P.: Modeling and simulating the noisy behavior of near-term quantum computers. Phys. Rev. A **104**, 062432 (2021) https://doi.org/10.1103/PhysRevA.104.062432

[57] Kharkov, Y., Ivanova, A., Mikhantiev, E., Kotelnikov, A.: Arline Benchmarks: Automated Benchmarking Platform for Quantum Compilers (2022). https://arxiv.org/abs/2202.14025

[58] QuTech: Quantum Inspire Starmon-5 Fact Sheet. https://qutech.nl/wp-content/uploads/2020/04/3.-Technical-Fact-Sheet-Quantum-Inspire-Starmon-5.pdf (2020)

[59] IBM Quantum: Qiskit Optimization levels. https://docs.quantum.ibm.com/guides/set-optimization

[60] IBM Quantum: IBM Quantum Routing Documentation. https://docs.quantum.ibm.com/api/qiskit/qiskit.transpiler.Layout

[61] Amy, M., Maslov, D., Mosca, M.: Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **33**(10), 1476–1489 (2014) https://doi.org/10.1109/tcad.2014.2341953

[62] IBM Quantum: Crosstalk Adaptive Schedule. https://docs.quantum.ibm.com/api/qiskit/0.28/qiskit.transpiler.passes.CrosstalkAdaptiveSchedule

[63] IBM Quantum: Twirling Options. https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/options-twirling-options

[64] Niu, S., Todri-Sanial, A., Bronn, N.: Multi-qubit dynamical decoupling for

enhanced crosstalk suppression. Quantum Science and Technology **9** (2024) https://doi.org/10.1088/2058-9565/ad5a37

[65] Mundada, P., Zhang, G., Hazard, T., Houck, A.: Suppression of qubit crosstalk in a tunable coupling superconducting circuit. Phys. Rev. Appl. **12**, 054023 (2019) https://doi.org/10.1103/PhysRevApplied.12.054023

[66] Nation, P.D., Saki, A.A., Brandhofer, S., Bello, L., Garion, S., Treinish, M., Javadi-Abhari, A.: Benchmarking the performance of quantum computing software (2025). https://arxiv.org/abs/2409.08844

[67] Tannu, S.S., Qureshi, M.K.: Not all qubits are created equal: A case for variability-aware policies for nisq-era quantum computers. In: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. ASPLOS '19, pp. 987–999. Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3297858.3304007