A Reality Check on Quantum Optimisation: Evidence from an Industrial Case Study

Hila Safi*† Karen Wintersperger*, Oliver von Sicard†, Christoph Niedermeier*, Wolfgang Mauerer†*
* Siemens Foundational Technologies, Munich, Germany
{hila.safi, karen.wintersperger, christoph.niedermeier}@siemens.com
† Technical University of Applied Sciences Regensburg, Laboratory for Digitalisation, Regensburg, Germany
{oliver.vonsicard, wolfgang.mauerer}@oth-regensburg.de

***Abstract*—Quantum Processing Units promise speed-ups for selected computational problems, including combinatorial optimisation, but their industrial utility remains an open challenge. We study an industrial variant of the Job-Shop Scheduling Problem using quantum, quantum-inspired, and classical methods across three platforms: IBM Quantum, the D-Wave Quantum Annealer, and the Fujitsu Digital Annealer. By tailoring formulations to hardware-specific constraints, we show that hardware-software co-design is essential for solution quality and scalability. We benchmark all approaches against an exact classical solver and a MILP formulation, evaluating runtime, solution quality, and scalability. Our results indicate that quantum and quantum-inspired optimisation can support industrial solver selection, integration in classical workflows, modelling decisions, and early proof-of-concept development, while suggesting a potential path towards improved approximations for industrial scheduling.**

***Index Terms*—Optimisation Application, Industrial Scheduling, Job-Shop Scheduling, QUBO, Quantum-inspired Optimisation, Benchmarking, Hardware-Software Co-Design**

I. Introduction

Quantum computers may solve selected computational problems exponentially faster than classical systems, but their practical value can only be assessed on real-world applications [1], [2]. Concurrently, quantum engineering increasingly highlights the need to consider the full computational stack, including algorithm design, problem formulation, hardware constraints, and classical post-processing. In this work, we study how these factors affect solution quality [3] and how quantum and quantum-inspired methods can be integrated into existing classical industrial workflows. Our comparison is application-driven, we focus on a real industrial variant of the Job Shop Scheduling Problem (JSSP), a central optimisation challenge in production logistics and manufacturing, where efficiency and cost reduction are key objectives [4]. The industrial instances provided by Siemens AG allow us to evaluate how problem encodings, solver paradigms, and hardware characteristics influence end-to-end performances and affect practical scheduling solutions. We use this case study to derive transferable design considerations for near-term optimisation, particularly regarding the trade-off between fidelity, constraint density, and hardware tractability, while also contributing to a better understanding of how quantum computing may be applied to real-world optimisation problems as hardware matures. The study therefore serves not only as a benchmark of current performance, but also as a first step towards development and the preparation of future industrial optimisation pipelines. As reference points, we develop a classical exact solver for small tractable instances and a Mixed-Integer Linear Programming (MILP) model for larger ones. While MILP is exact in principle, typically using branch-and-bound and branch-and-cut [5], we solve larger instances under a fixed time budget and use the best solution found as an approximate classical baseline [6]. This provides a scalable and practically relevant comparator for quantum-inspired optimisation. We compare two QUBO formulations for the JSSP. The Single-Constraint Model uses a compact constraint structure, while the Multi-Constraint Model introduces additional constraints to represent scheduling dependencies more accurately. This exposes the trade-off between modelling simplicity, constraint density, and hardware tractability. The models are evaluated on three platforms: IBM gate-based quantum hardware, the D-Wave Advantage quantum annealer, and the Fujitsu Digital Annealer. Our results show that the number of constraints and the degree of hardware proximity in the formulation strongly affect runtime and solution quality across all platforms. More broadly, the findings show that hardware performance cannot be separated from modelling choices and device characteristics. This underscores the importance of hardware-software co-design and systematic design-space exploration across the full quantum stack, from model formulation to device-level execution. This paper is augmented by a reproduction package [7], which is permanently archived on Zenodo at doi:10.5281/zenodo.21363322 (links available in the PDF).

II. Related Work

The Job Shop Scheduling Problem (JSSP) is a fundamental problem in production logistics and operations research and is well known to be computationally hard, even restricted variants are NP-hard, and realistic instances combine machine-capacity constraints, sequence-dependent setup times, and objective trade-offs [4], [8]. Classical research on JSSP and related parallel-machine scheduling has produced a large body of exact and heuristic methods, including branch-and-bound

* Oliver von Sicard conducted this work while employed at Siemens AG.

and MILP formulations. For parallel machines with sequence-dependent setup times (PMSP-SDST), Ying *et al.* provide a recent survey of modelling and algorithmic strategies [9]. Our Classical Approximation Model follows this PMSP-SDST literature and uses a time-limited MILP formulation to obtain practically comparable approximations, similar in spirit to the approach of Schönberger *et al.* for join-order optimisation [10]. On the quantum side, QAOA was introduced by Farhi *et al.* as a variational hybrid algorithm for combinatorial optimisation on NISQ devices [11], and has received intensive scrutiny since (see Refs. [12], [13] and references therein); modifications to the structure of the quantum circuit itself (*e.g.*, Refs. [13]–[17]) and changes to the classical optimisation procedure (*e.g.*, Refs. [18]–[26]) have been proposed to improve performance especially in NISQ scenarios. QAOA and related methods have also been studied for countless graph and scheduling problems, including evaluations on industrial data and the effects of embedding choices [27]. Our work complements this line by studying QAOA-based JSSP formulations using a systematic evaluation approach [28], [29] on IBM hardware. Quantum annealing and quantum-inspired optimisation have also been explored for combinatorial optimisation problems. D-Wave annealers have been applied to scheduling and logistics via Ising and QUBO encodings that must be embedded into sparse hardware graphs [30]–[32] (and are known to possibly benefit from changes in the problem formulation itself [33]–[35]), while Fujitsu's Digital Annealer has been proposed as a quantum-inspired architecture for dense QUBO problems [36], [37]. Our work extends this literature by comparing the Digital Annealer not only with classical MILP and heuristic baselines, but also with a physical quantum annealer (D-Wave) and gate-based quantum devices (IBM), using the same industrial JSSP instances. Scheduling-oriented quantum optimisation has been studied in both annealing and gate-based settings. Venturelli *et al.* introduced an early D-Wave QUBO formulation for JSSP [38], Schworm *et al.* extended annealing approaches to flexible job-shop scheduling in manufacturing [39], Orts *et al.* considered the unrelated parallel machine setting [40], Kurowski *et al.* investigated QAOA for JSSP [41], and Schwenzow *et al.* studied VQE for identical parallel machines with sequence-dependent setup times [42]. Together with earlier work on quantum scheduling by Lu *et al.* [43] and annealing-based load balancing by Rathore *et al.* [44], these studies show that formulation choices and hardware constraints are central to quantum optimisation for scheduling. Finally, our study is closely related to work on hardware-software co-design and quantum scalability [45].

III. Problem Description and Models

This paper considers a JSSP variant based on a real-world industrial Siemens application. We consider N jobs, each executed once, and M identical production machines on which any job can run. Jobs differ in processing time and require a specific tool configuration, referred to as a rig. We consider L possible rigs:

- Jobs: $i \in \{1, 2, \ldots, N\}$
- Machines: $j \in \{1, 2, \ldots, M\}$
- Rigs: $k \in \{1, 2, \ldots, L\}$

Some jobs require the same rig, while others require different rigs. Since rig changes take time and depend on the current and subsequent job, jobs with the same rig should preferably be processed consecutively on the same machine. Each machine is assumed to start with an initial rig from prior production. The objective is to minimise the makespan, defined as the maximum production time across all machines. Modelling the complete scheduling task as a QUBO requires on the order of N^2M variables, as the time domain must also be represented. Moreover, rig-change optimisation is difficult to encode because it requires identifying whether two jobs are executed consecutively on the same machine. With N global time steps, consecutive jobs on one machine do not necessarily correspond to indices t and $t+1$. We therefore split the task into two stages. First, jobs are assigned to machines based on durations and rigs using a quantum or classical solver. Second, the jobs on each machine are ordered to minimise rig changes; for the considered problem sizes, this step can be solved classically and is not studied further. Since the maximum function cannot be directly encoded in a QUBO, we approximate makespan minimisation by balancing total production times across machines. We evaluate two QUBO assignment models, a Single-Constraint and a Multi-Constraint formulation, which differ in objective design and constraint density. Because the task is split, an optimal QUBO solution does not necessarily recover the global optimum of the full scheduling problem. A classical MILP-based Classical Approximation Model is introduced as an additional baseline.

A. Single-Constraint Model

The first model is a formulation with one constraint. We minimise the difference of the overall duration on each machine to the mean production duration D defined as

$$\bar{D} = \frac{1}{M} \sum_{i=1}^{N} d_i, \tag{1}$$

where d_i is the duration of job i. The total duration on each machine includes job durations and rig-change times, including the change from the machine's initial rig. Since the job order is not optimised, the first job is unknown; we therefore approximate this initial setup cost using the average rig-change time between all assigned jobs and the initial rig. Our binary decision variable is

$$x_{ij} = \begin{cases} 1, & \text{if job } i \text{ is processed on machine } j, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The matrix $R \in \mathbb{R}^{N \times N}$ contains the rig-change times between pairs of jobs. The matrix $S \in \mathbb{R}^{N \times M}$ describes the rig-change time difference between each job and the initial rig of each machine. The average rig-change time for machine j is

$$S_j := \frac{1}{N} \sum_{i=1}^{N} S_{ij}. \tag{3}$$

Objective 1 | H_{obj}: Minimise the difference to the mean duration over all machines, including the initial rig-change time

$$H_{\text{obj}} = \sum_{j=1}^{M} \left(\sum_{i=1}^{N} x_{ij}(d_i + S_{ij}) - (\bar{D} + S_j) \right)^2 . \quad (4)$$

Objective 2 | H_{rig}: Minimise rig change time between jobs and between the jobs and the initial rig on each machine

$$H_{\text{rig}} = \sum_{j=1}^{M} \left(\sum_{i<i'}^{N} x_{ij} x_{i'j} R_{ii'} \right) + \left(\sum_{i=1}^{N} x_{ij} S_{ij} \right). \quad (5)$$

Constraint | H_{single}: Each job must be assigned to exactly one machine

$$H_{\text{single}} = \sum_{i=1}^{N} \left(\sum_{j=1}^{M} x_{ij} - 1 \right)^2 . \quad (6)$$

B. Multi-Constraint Model

The Multi-Constraint Model extends the assignment problem of the Single-Constraint Model by explicitly introducing rig configurations and separating the decision of

1) which machine executes each job, and
2) which rig configuration is active on that machine.

To this end, we additionally consider the binary decision variable:

$$y_{kj} = \begin{cases} 1, & \text{if rig configuration } k \text{ is needed on machine } j, \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

Furthermore, we define the following model parameters: Let $R_{\ell k}$ be the rig–change time from configuration ℓ to k with $R_{kk} = 0$. The mean rig–change time associated with rig k is

$$\bar{R}_k = \frac{1}{L-1} \sum_{\ell=1}^{L} R_{\ell k}. \quad (8)$$

Matrix $S_{jk} \in \{0, 1\}$ encodes the initial rig of each machine:

$$S_{jk} = \begin{cases} 1, & \text{if machine } j \text{ has initial rig } k, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

The resulting mean initial rig–change time on machine j is

$$\bar{S}_j = \sum_{k=1}^{L} S_{jk}\, \bar{R}_k. \quad (10)$$

Matrix $r_{ik} \in \{0, 1\}$ indicates the rig requirement of job i:

$$R_{ik} = \begin{cases} 1, & \text{if job } i \text{ is executed with rig configuration } k, \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

Objective 1 | H_{obj}: Minimise the deviation of the total duration on each machine from the mean duration, including the initial rig configuration set on each machine

$$H_{\text{obj}} = \sum_{j=1}^{M} \left(\sum_{i=1}^{N} x_{ij} d_i + \sum_{k=1}^{L} y_{kj} \bar{R}_k - \bar{S}_j - \bar{D} \right)^2 . \quad (12)$$

Constraint 1 | H_{single}: Each job must be assigned to exactly one machine

$$H_{\text{single}} = \sum_{i=1}^{N} \left(\sum_{j=1}^{M} x_{ij} - 1 \right)^2 . \quad (13)$$

Constraint 2 | H_{rig}: Which rig changes take place on which machine

$$H_{\text{rig}} = \sum_{j=1}^{M} \sum_{k=1}^{L} \left(S_{jk} + \sum_{i=1}^{N} R_{ik} x_{ij} \right) (1 - y_{kj}). \quad (14)$$

Constraint 3 | H_{switch}: Minimise rig-change time between the jobs on the same machine

$$H_{\text{pair}} = \sum_{j=1}^{M} \left(\sum_{k=1}^{L} \bar{R}_k y_{kj} - S_j \right)^2 . \quad (15)$$

C. Classical Approximation Model

For the classical implementation, we model the problem as a parallel-machine scheduling problem with sequence-dependent setup times (PMSP-SDST) and flexible job assignment. The formulation is based on established PMSP-SDST literature [9], [46]–[51]. Our approach combines a greedy constructive heuristic, local improvement, SDST-aware insertion, and an iterated greedy destroy-repair procedure. First, jobs are grouped by rig and sorted within each rig by decreasing processing time. Rig blocks are then assigned to machines using a score that balances machine load and setup cost,

$$\lambda \cdot \text{load}(j) + \text{setup}(\text{tail\_rig}(j) \to \text{rig\_block}). \quad (16)$$

The resulting machine sequences are improved by local two-swap search. To further refine solutions, we apply an iterated greedy framework that destroys and reconstructs partial schedules using SDST-aware insertion, with occasional uphill acceptance based on

$$\text{Probability}(\text{accept}) = \exp(\Delta / T). \quad (17)$$

For small instances, we additionally use a disjunctive MILP baseline with sequence variables y_{ijk}, dummy start jobs, and big-M timing constraints. The MILP minimises the makespan $C_{\max}$ subject to standard sequencing constraints of the form

$$C_{\max} \geq S_i + p_i, \qquad S_k \geq S_i + p_i + s_{i \to k} - M(1 - y_{ijk}). \quad (18)$$

Algorithm 1 summarises the overall heuristic framework.

D. QAOA

QAOA is a hybrid variational algorithm for combinatorial optimisation on NISQ hardware [11]. It alternates problem and mixer Hamiltonians over p layers, while a classical optimiser updates the parameters to minimise the problem Hamiltonian expectation value. The resulting output distribution yields an approximate solution to the QUBO instance.

Algorithm 1 Heuristic Framework for PMSP-SDST Scheduling

Require: Instances with jobs, machines, processing times, and SDST matrix
Ensure: A schedule minimising $C_{\max}$

```
Group jobs by rig; sort each rig in descending duration
Initialise empty sequence for each machine
for each rig block in random order do
    for each machine j in parallel do
        Compute score λ · load(j) + setup(tail(j) → rig)
    end for
    Assign rig block to machine with minimum score
end for
for each machine j do
    Perform 2-swap local search on the sequence of j
end for
while termination criterion not met do
    Destroy part of the schedule
    for each removed job do
        Evaluate all insertion positions using SDST marginal cost
        Insert at best position
    end for
    if new solution worse then
        Accept with probability exp(Δ/T)
    end if
end while
return best schedule found
```

IV. Hardware Platforms

A. IBM Quantum (Eagle and Heron Processors)

IBM's gate-based devices use superconducting fixed-frequency transmon qubits operated at millikelvin temperatures and arranged in a heavy-hex topology to reduce crosstalk and improve manufacturability [52]–[54]. Eagle provides 127 qubits, while Heron introduces redesigned couplers and a new control stack for improved two-qubit fidelities [55].

B. D-Wave Quantum Annealer (Advantage System)

D-Wave's *Advantage* quantum annealer solves Ising/QUBO problems through annealing dynamics and contains over 5000 superconducting qubits in the Pegasus connectivity graph [30], [31]. It also offers a hybrid solver combining classical heuristics with quantum annealing, although its internal optimisation process is intransparent.

C. Fujitsu Digital Annealer

The Fujitsu Digital Annealer is a quantum-inspired CMOS accelerator that emulates annealing dynamics at room temperature and performs parallel updates on QUBO formulations [36]. Since it does not require minor-embedding, it can directly represent dense optimisation problems [37].

V. Experimental Setup

To ensure comparability, we evaluate the proposed JSS formulation using a unified experimental setup for quantum, quantum-inspired, and classical solvers [29], [56]. This includes benchmark instances, solver settings, parameters, validation metrics, and integration into the classical industrial workflow [42].

A. Benchmark

All experiments use real industrial JSS instances provided by Siemens. The instances range from small toy problems with 2 machines, 2 rigs, and 2 jobs to industry-scale and larger cases with 32 machines, 85 rigs, and 256 jobs. Each instance specifies machines, jobs, rigs, processing times, and rig setup relations, with multiple instances per size to vary difficulty. For small instances, a branch-and-bound solver provides the optimal makespan and job-to-machine assignment. the largest tractable case has 6 machines, 5 rigs, and 14 jobs. For larger instances, we use the best solution found within a one-hour time limit. We run our experiments on the D-Wave Quantum Annealer, IBM Eagle and IBM Heron processors, the Fujitsu Digital Annealer, and classical hardware equipped with an Intel Xeon E5-2683 v4 CPU (16 cores, 32 threads per socket; 2 sockets, 64 logical CPUs in total) running at up to 3.0 GHz. For the quantum and quantum-inspired evaluations, we implement two assignment-based QUBO formulations, the Single-Constraint and Multi-Constraint Models, described in Sections III-A and III-B. Both encode only the job-to-machine assignment and do not enforce internal job order. For the considered instance sizes, sequencing is efficiently handled classically [57], making this decomposition suitable for a realistic hybrid industrial workflow. To ensure a fair comparison with classical state-of-the-art methods, we also implement a Classical Approximation Model (Section III-C), which approximates the assignment component efficiently on classical hardware.

B. Hardware and Solver Setups

1) D-Wave Annealer: The QUBO models are solved on the D-Wave Advantage and hybrid systems. For each instance, we use 1024 shots and vary the chain strength according to D-Wave guidelines [58].

2) IBM Gate-Based Quantum Processors: QAOA circuits for the Single-Constraint Model are generated in Qiskit 1.4 and executed with 1024 shots at different QAOA depths to identify when noise from additional layers outweighs gains in solution quality. Experiments are conducted on the Eagle processor *Kiev* and the Heron processors *Fez* and *Torino*.

3) Fujitsu Digital Annealer: The third-generation Digital Annealer is used as a reference quantum-inspired architecture. It may be viewed as a lower bound on what future quantum annealers could achieve with improved hardware stability, connectivity, and precision, while acknowledging important architectural differences from physical quantum annealers in the threats to validity section VII-B. Experiments use three QUBO implementations: a `PyQUBO` model, a standard

`BinPol` formulation, and an extended `BinPol` version that separates objective and constraint terms to exploit hardware-supported constraint handling.

4) Classical Approximation: For larger-scale problems, we implement the classical approximation algorithm described in Section III-C under the same assignment constraints as the Single-Constraint and Multi-Constraint Models, using between 100 and 1200 iterations.

C. Evaluation Metrics

We use several metrics to compare performance across all platforms:

1) **Share of best, valid, and invalid solutions**, where best refers to the best feasible solution found within the time limit, valid satisfies all constraints and invalid violate at least one constraint.
2) **Optimality gap** relative to the best-known solution.
3) **Execution time** (end-to-end execution time).
4) **Scalability** with respect to problem and QUBO size.

VI. Results

We next discuss the results across the hardware platforms described in Section IV.

A. D-Wave Annealer Results

Figure 1 shows the performance of both QUBO formulations on the D-Wave Advantage System 6.4. We also run experiments on Advantage System 4.1, but the results are nearly identical and are omitted for clarity. The x-axis lists problem instances in increasing complexity as (M, R, J), and the two plots below show the required logical qubits. The Multi-Constraint Model requires more qubits, scaling as $M \cdot J + M \cdot R$, whereas the Single-Constraint Model uses only $M \cdot J$ variables. The y-axis reports the share of best, valid, and invalid solutions. The Multi-Constraint Model is too constraint-dense for practical execution on both Advantage generations: only very small toy instances yield valid results, best solutions are rare, and invalid samples quickly dominate as problem size increases. In contrast, the Single-Constraint Model performs much better. Both Advantage systems 4.1 and 6.4 show nearly identical behaviour, indicating that the hardware improvements between these generations are minimal for this problem class. For the Single-Constraint Model, valid and best solutions remain frequent up to QUBO sizes of about 100–120 qubits.

Table I shows the D-Wave hybrid solver results for the Multi-Constraint Model on the same instances. Across all problem sizes, the solver returns at least one valid solution and, up to about 15 qubits, the best solution. However, although 1024 shots are executed, only the top solution is returned, so the share of invalid or suboptimal results cannot be determined. Overall, the hybrid solver produces valid schedules reliably across all tested instances.

Figure 2 visualises the average gap to the best-known solution, considering only valid solutions. The goal of this analysis is to quantify, for each platform, how close the valid solutions are to the optimal reference. The two colours represent the two QUBO formulations. Across both Advantage systems (4.1 and 6.4), the Single-Constraint Model produces valid solutions closer to the best-known solution, mainly because the more faithful Multi-Constraint Model becomes too complex for D-Wave and yields too few valid solutions as problem size grows. Later Fujitsu results show that this trend can reverse when the hardware handles added complexity better. The hybrid solver performs better than the pure quantum hardware: its valid solutions exhibit smaller gaps to the best-known reference across most instance sizes. Figure 3 compares the effect of different chain strengths on both models across Advantage 4.1 and 6.4; chain strength cannot be adjusted for the Hybrid Solver. For the Multi-Constraint Model, chain strength influences solution quality, whereas for the Single-Constraint Model the D-Wave default setting (chain strength 0) gives the most stable results. Manual tuning does not yield systematic improvements and often reduces the number of valid or best samples, especially for larger QUBOs. Overall, solvability depends more on the problem formulation than on chain-strength tuning.

Problem	Result	Problem	Result
M=2 R=2 J=3	Best solution	M=3 R=3 J=7	Valid
M=2 R=3 J=3	Best solution	M=3 R=4 J=8	Valid
M=2 R=3 J=4	Best solution	M=3 R=5 J=10	Valid
M=2 R=3 J=5	Best solution	M=3 R=6 J=12	Valid
M=2 R=4 J=4	Best solution	M=4 R=6 J=10	Valid
M=2 R=3 J=6	Valid	M=4 R=6 J=11	Valid
M=2 R=4 J=5	Valid	M=4 R=6 J=13	Valid
M=2 R=3 J=7	Valid	M=4 R=6 J=15	Valid
M=2 R=4 J=6	Valid	M=5 R=6 J=14	Valid
M=3 R=2 J=5	Best solution	M=6 R=5 J=14	Valid
M=2 R=3 J=8	Valid	M=5 R=7 J=18	Valid
M=2 R=4 J=8	Valid	M=6 R=6 J=16	Valid
M=3 R=3 J=6	Valid	M=5 R=8 J=20	Valid
M=2 R=4 J=10	Valid	M=4 R=12 J=30	Valid
M=2 R=2 J=13	Valid	M=5 R=10 J=30	Valid
M=2 R=4 J=11	Valid	M=6 R=7 J=30	Valid

TABLE I
HYBRID SOLVER RESULTS FOR THE MULTI-CONSTRAINT MODEL.

B. IBM Gate-Based Quantum Computing

Figure 4 summarises the results on the IBM Eagle (*Kiev*) and Heron (*Fez*, *Torino*) processors. The x-axis lists problem instances as (M, R, J), the y-axis shows the share of outcomes across all shots, and the facets indicate the number of QAOA layers. We restrict the IBM experiments to the Single-Constraint Model, as the Multi-Constraint Model does not produce meaningful solutions on any tested system. Across all processors, IBM hardware solves only small toy instances, and the probability of obtaining valid or best solutions is only slightly above random guessing. The Heron processors improve over Eagle, but not enough to expand the range of solvable problems substantially. Increasing the number of QAOA layers beyond a small threshold quickly reduces performance due to noise [59], [60]. For problem sizes up to about 10 qubits, up to 8 layers may provide slight benefits, but for larger problems additional layers introduce more noise than improvement. Results for all tested layer settings are available in the reproduction package.

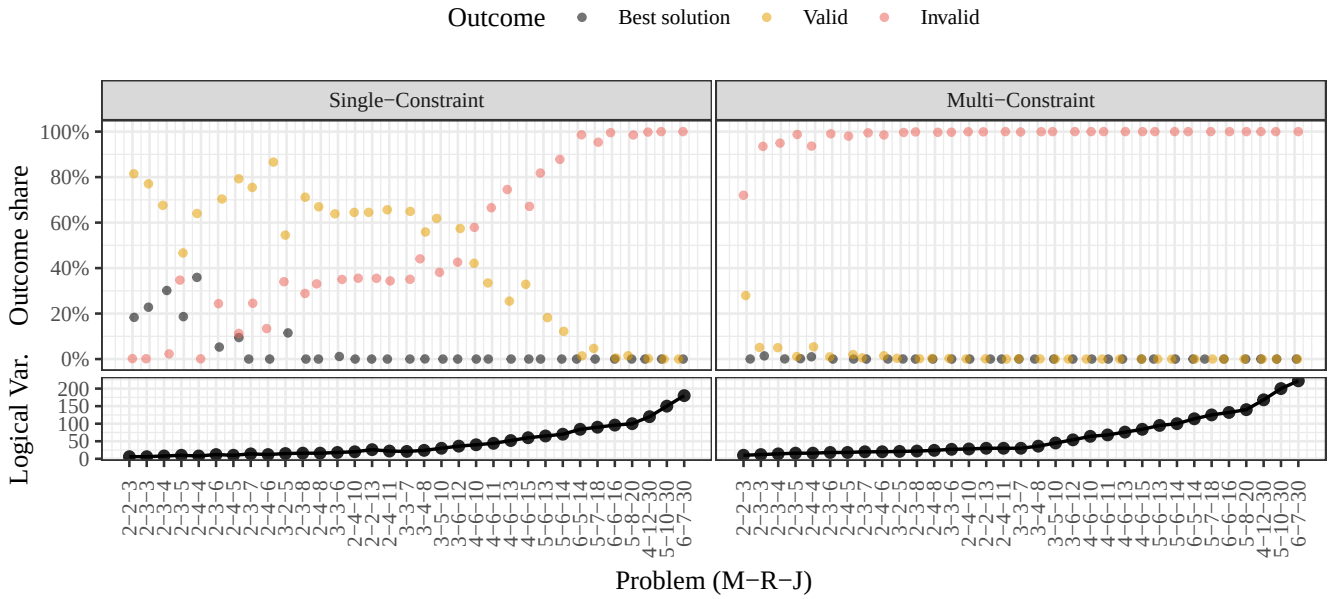

Fig. 1. Scatter plot of best, valid, and invalid solution shares for the Single-Constraint and Multi-Constraint models on Advantage System 6.4. Problem instances on the x-axis are denoted as $(M{-}R{-}J)$ (machines-rigs-jobs). The plots below show the required logical qubits per model, with the Multi-Constraint model exhibiting higher qubit demand due to additional constraints. The y-axis represents the percentage of samples classified as best (black), valid (yellow), or invalid (red).

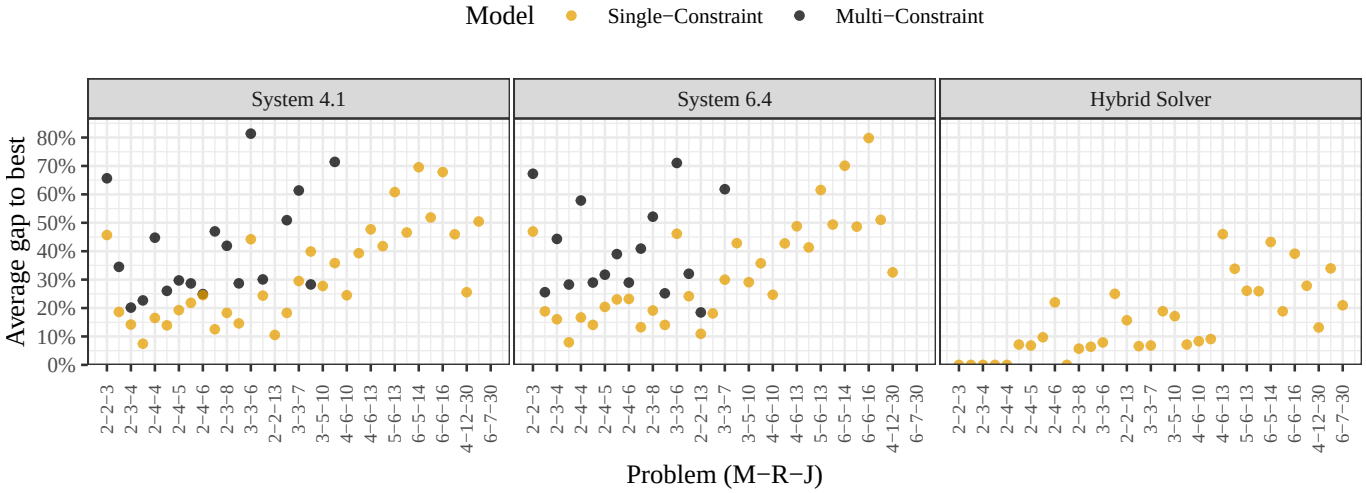

Fig. 2. Average gap (in %) of all valid solutions relative to the best-known solution (y) for varying instances (x) in form $(M{-}R{-}J)$. Panels separate results for Advantage 4.1, Advantage 6.4, and the Hybrid Solver. The x-axis lists all problem instances; invalid and best solutions are excluded.

C. Fujitsu Digital Annealer

Figure 5 summarises the results from the Fujitsu Digital Annealer (DA) across the three QUBO implementations described in Section V-B3. Unlike the quantum devices, the DA supports problem sizes beyond an equivalent of 10,000 logical variables, enabling large-scale experiments for both the Single-Constraint and Multi-Constraint Models. The x-axis shows the number of logical variables, and the y-axis the share of best, valid, and invalid solutions. For the Single-Constraint Model, both the standard `BinPol` and the separated `BinPol` formulation perform consistently well across all tested scales, returning high shares of valid and best solutions even for the largest instances. For the more complex Multi-Constraint Model, the standard `BinPol` formulation is no longer sufficient at larger scales, and the separated `BinPol` implementation is required to obtain valid solutions. By contrast, the `PyQUBO` formulation degrades clearly for both models despite being mathematically equivalent, indicating that internal compilation and preprocessing in the DA software stack significantly affect solution quality. As with the D-Wave hybrid solver, the DA returns only a limited number of top solutions rather than the full sample distribution, but

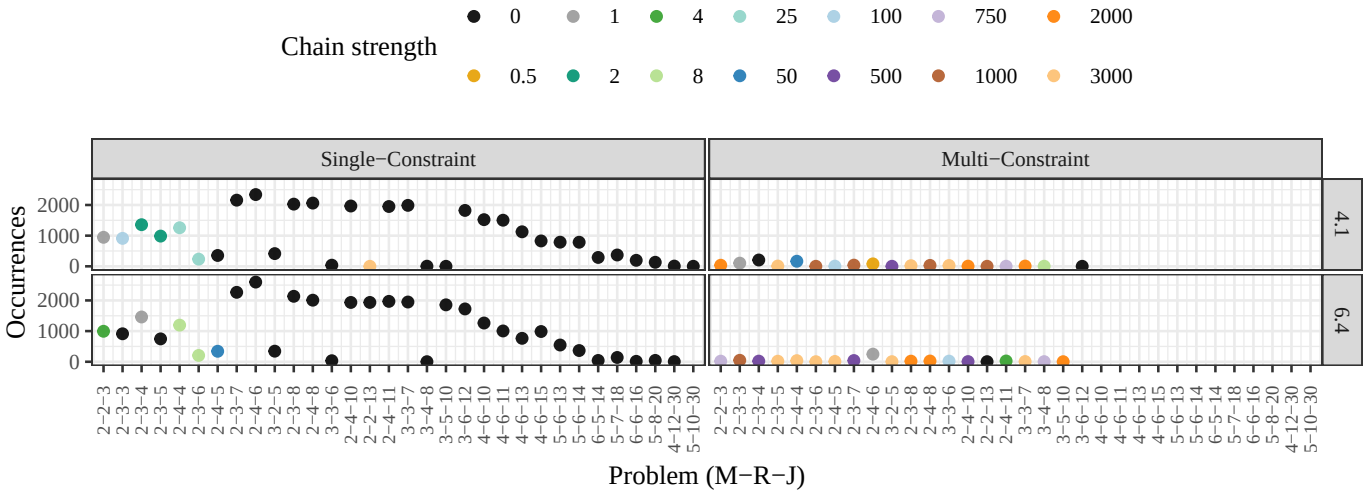

Fig. 3. Impact of chain strength on the number of valid and best solutions for the Single-Constraint and Multi-Constraint models on Advantage 4.1 and 6.4. The x-axis shows problem instances $(M-R-J)$, the y-axis the number of valid and best samples, and colours denote chain strength (0 = hardware default to 3000).

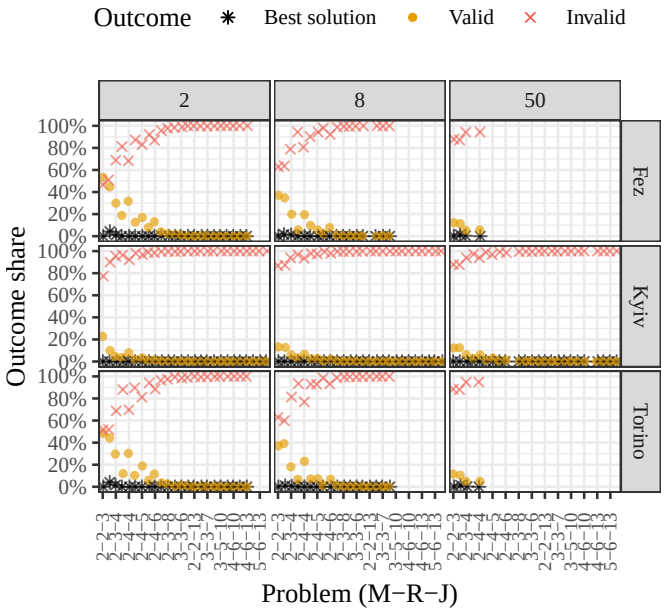

Fig. 4. QAOA outcome shares on the Eagle (*Kiev*) and Heron (*Fez*, *Torino*) processors. The x-axis shows problem instances (M, R, J), the y-axis the percentage of best, valid, and invalid solutions, and each facet a different QAOA depth.

the high rate of valid solutions across large instances shows that it is well suited for the assignment-based formulations studied here. Figure 6 shows the average gap of valid Fujitsu Digital Annealer solutions to the best-known solution. The x-axis gives the problem size in logical variables, the y-axis the average relative deviation, colours denote the three QUBO implementations, and the facets separate the Single-Constraint and Multi-Constraint Models. Best and invalid solutions are excluded to focus on valid assignments. Across both models, the `PyQUBO` formulation produces valid solutions that are consistently farther from the best solution than the two `BinPol`-based implementations. Although the Improved `BinPol` variant performs best at finding optimal solutions, especially for the Multi-Constraint Model, its average gap is larger. We attribute this to the objective/constraint separation, which improves feasibility but appears to reduce fine-grained optimisation performance. The standard `BinPol` formulation achieves the smallest average gap and thus the best valid-solution quality across a wide range of problem sizes. To assess the scaling behaviour of the Fujitsu Digital Annealer, Figure 7 shows the execution time as a function of QUBO size across all implementations and formulations. In all cases, execution time exhibits a clear quadratic growth trend with the number of logical variables.

D. Classical Approximation

Finally, we compare the classical approximation approach with the Fujitsu Digital Annealer, as both solve industry-scale problems. Figure 8 shows the normalised makespan for the Single-Constraint and Multi-Constraint Models, the iterative classical solver, the greedy solver, and the optimal reference as a lower bound. Across nearly all instances, both Fujitsu-based models outperform the classical approximation methods. This is particularly relevant when considering runtime: The classical solvers also exhibit quadratic empirical scaling, similar to the Fujitsu annealer. Even the iterative version with 1000 iterations provides little improvement over the greedy heuristic and remains consistently worse than the annealer-based solutions. Overall, the Digital Annealer yields higher-quality schedules for large-scale problems, when being evaluated under the same runtime scaling setting.

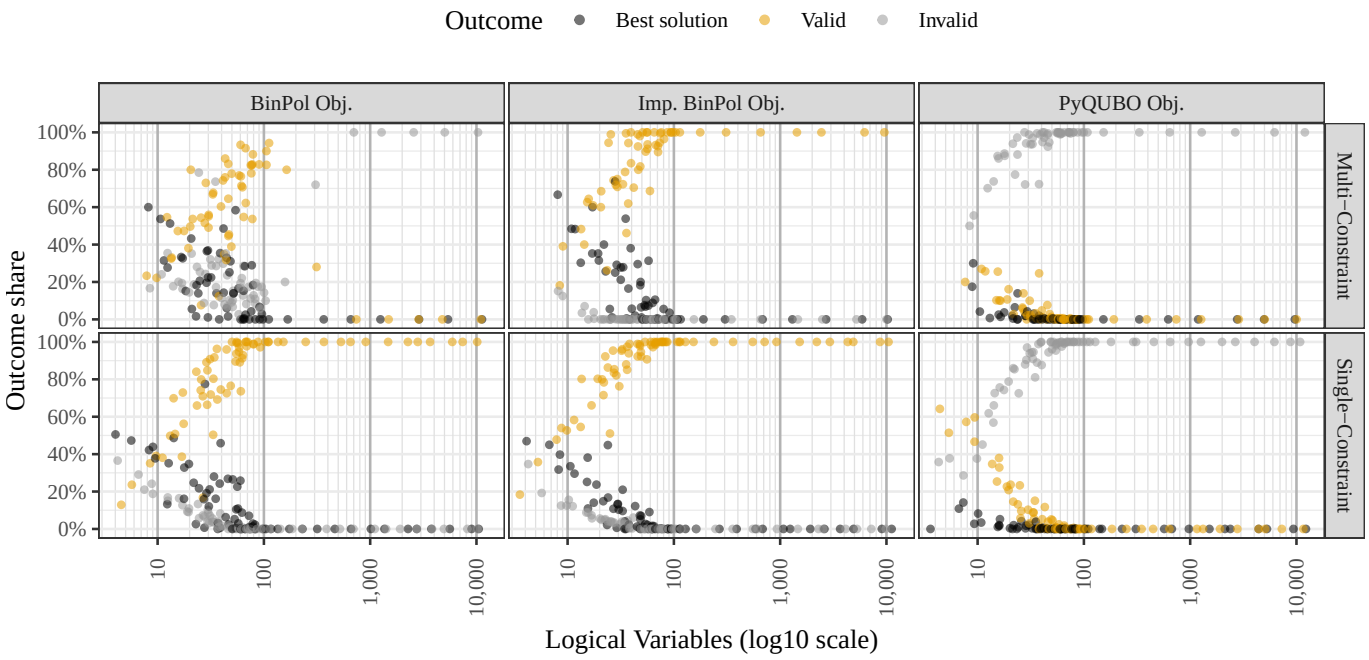

Fig. 5. Overview of Fujitsu Digital Annealer results for the (`BinPol`, improved `BinPol`, and `PyQUBO`) implementation and both problem formulations (Single-Constraint and Multi-Constraint). The x-axis shows the problem size (number of qubits) and the y-axis reports the share of outcomes classified as best, valid, or invalid.

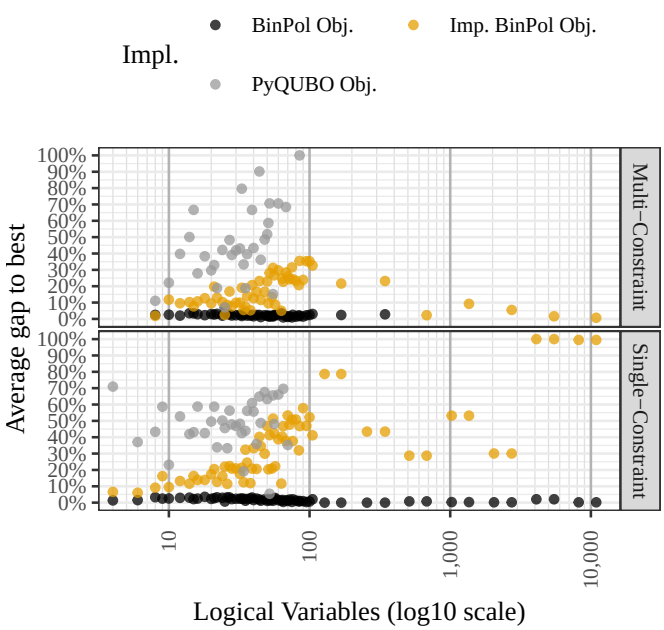

Fig. 6. Average gap to the best-known solution for the Fujitsu Digital Annealer across problem sizes.

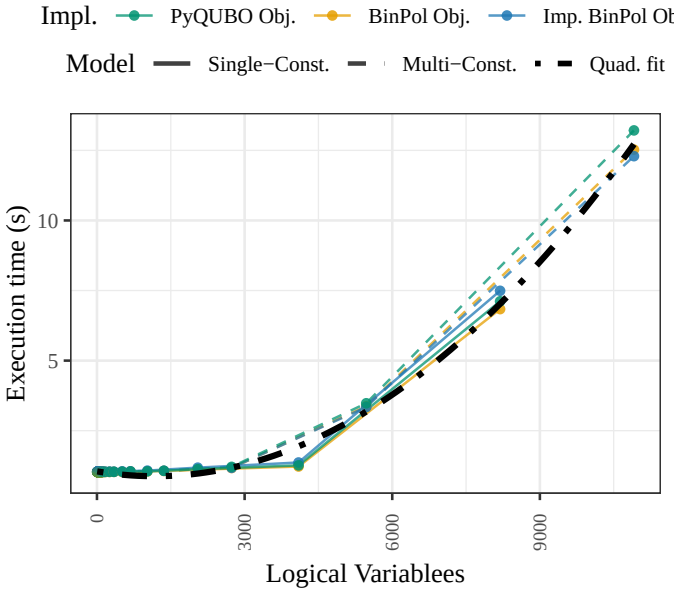

Fig. 7. Execution time on the Fujitsu Digital Annealer as a function of problem size (number of logical variables).

VII. Threats to Validity

These results should be interpreted with some care, as their generalisability may be limited [61], [62].

A. Evaluation baseline

In this study, we deliberately compare solution quality against the true optimal solutions rather than the QUBO optima. This represents a conservative assessment, as the QUBO formulation itself may introduce approximation errors independent of the solver. Our goal was to evaluate practical

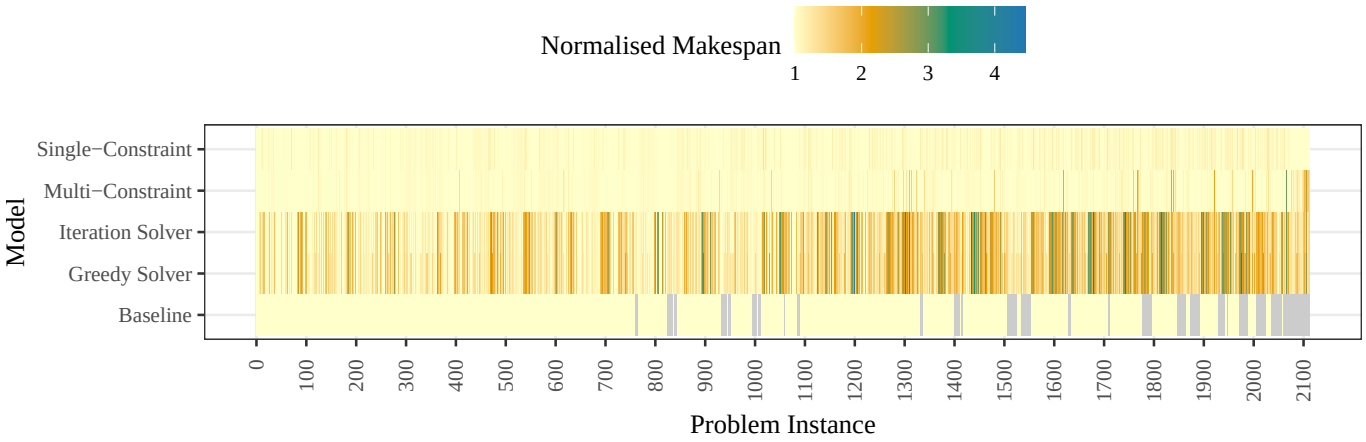

Fig. 8. Comparison of normalised makespan across all problem instances for Fujitsu Digital Annealer (Single-Constraint and Multi-Constraint Model), the classical greedy and iterative solver, and the optimal solution as baseline. Lower values indicate better schedules.

usefulness for real decision making rather than performance relative to the encoded objective alone. Consequently, the reported optimality gaps may overestimate solver limitations but better reflect end-to-end application performance.

B. Internal Validity

One threat stems from our treatment of the Fujitsu Digital Annealer(DA) as a practical lower bound for future fault-tolerant quantum annealing. While this assumption is reasonable from one perspective, the DA's newest generation enables separation of objective terms and penalty constraints, a capability that current physical quantum annealers do not have. To mitigate this risk, we additionally evaluated a full QUBO formulation on the DA. Another concern relates to modelling choices. The JSSP admits multiple QUBO encodings, and different weighting of penalty terms prioritise different industrial challenges. Thus our conclusions may not generalise across all variants of the JSSP or other scheduling problems. Also, both the DA and the D-Wave Hybrid Solver return only the best solutions rather than full sample distributions. While sufficient for optimisation evaluations, this limits transparency and prevents a deeper analysis on solution quality.

C. Construct Validity

Comparisons across computation platforms inherently face challenges because resource notions differ. While we try to ensure fairness by solving identical industrial problems on each platform, certain formulations may naturally benefit from one architecture more than another. Depending on what is considered "fair", this can be viewed as a threat to validity. From an industrial standpoint, however, the size of solvable instances matters more than abstract resource counts. Classical solver configurations introduce an additional threat: The branch-and-bound solver is allowed extended computation time to obtain ground-truth optimal solutions wherever feasible. In contrast, the MILP solver was deliberately time-limited to allow for comparison with quantum and quantum-inspired solvers. While we see this as appropriate for our objectives, this also implies MILP results do not necessarily represent the best possible classical approximations that could be reached using the same algorithm with more compute time. Additionally, splitting the problem introduces a trade-off: while it reduces complexity, it may prevent the global optimum from being reachable in certain cases. We evaluate the distance to the best found solution to quantify the trade-off.

D. External Validity

The evaluation focuses on one industrial JSSP variant and many different instances. Although the instances are highly realistic and reflect constraints encountered in production environments, it remains a representative rather than exhaustive dataset. Different industries—or even different production lines within Siemens—may apply different constraint priorities or operate under different scheduling dynamics. We solved a specialised problem using specific modelling, and the results demonstrate strong interactions between model formulation and hardware behaviour. Although these trends may also apply to other optimisation problems, as suggested by findings in previous work [45], they should still be interpreted with caution, and further verification across additional problem classes is required.

E. Ecological Validity

Although we conceptually simulate the integration of quantum and quantum-inspired solvers into existing classical pipelines, we do not include communication latencies, API overheads, or queueing delays in the runtime measurements. In operational settings, these overheads may influence end-to-end performance, particularly for cloud-based quantum hardware. The absence of such effects may therefore overestimate real-world performance. Finally, the data used in this study are realistic but not directly sourced from a live production system. Real deployments include additional complexities, such as dynamic job arrivals.

VIII. Discussion

Across all platforms, we observe a consistent co-design pattern: more expressive and constraint-rich QUBO formulations capture the industrial problem more accurately, but quickly become intractable on current quantum hardware. For near-term industrial optimisation, the central modelling question is therefore not how to encode the full problem as accurately as possible, but which parts of the problem should remain in the hardware-facing formulation and which should be shifted to hybrid or classical processing stages. This is most visible in the comparison between the Single-Constraint and Multi-Constraint Models. The Multi-Constraint Model, with $M \cdot J + M \cdot R$ variables, places substantially higher constraint pressure on the hardware and yields valid solutions only for toy instances on D-Wave. By contrast, the Single-Constraint Model, with $M \cdot J$ variables, remains solvable across all platforms and is therefore better suited to near-term hardware. A similar pattern appears on the Fujitsu Digital Annealer: although it can process problem sizes beyond $10{,}000$ logical variables, the more constraint-dense formulation still requires a more carefully tailored implementation. Overall, our results show that constraint pressure, at least as much as raw problem size, limits scalability across all evaluated platforms.

A. Trade-Offs and Transferable Insights

A central cross-platform result is the trade-off between solution validity and proximity to the optimum. Simplified formulations increase the probability of obtaining valid solutions, but can lead to larger gaps to the best-known solution. More detailed formulations represent the scheduling problem better, but current hardware often struggles to navigate the resulting constrained landscape efficiently. These findings suggest three transferable design principles for near-term industrial quantum optimisation. First, compact formulations that preserve the dominant optimisation structure while relaxing secondary constraints are often more effective on current hardware than more faithful but constraint-dense encodings. Second, solver assessment should be performed end-to-end, including feasibility, approximation quality, and practical execution effort, rather than only on encoded objective values. Third, the most suitable solver paradigm depends not only on problem size, but on the interaction between constraint density, formulation structure, and hardware capabilities. These observations are likely relevant beyond the present JSSP case to a broader class of industrial assignment and scheduling problems.

B. Platform-Specific Lessons

The IBM processors show architectural improvement from Eagle to Heron, but for this problem class they still solve only toy instances, and additional QAOA layers beyond small depths quickly become ineffective because noise outweighs any gain in solution quality. D-Wave performs substantially better with the compact formulation, while the Multi-Constraint Model remains impractical beyond toy scales. The Fujitsu Digital Annealer illustrates the benefit of specialised hardware most clearly: with a suitable formulation, it handles industry-scale instances and consistently outperforms the classical approximation heuristics in solution quality. From a practical perspective, the main lesson is not that one solver paradigm universally dominates, but that successful industrial proof-of-concept work currently depends on restricted problem scope, hardware-aware formulation, and explicit matching of optimisation goals to platform capabilities. Under present hardware conditions, this is more important than abstract expectations of quantum advantage.

C. Conclusion

We conducted a multi-platform evaluation of quantum annealing, quantum-inspired computing, and classical approximation methods for a real-world JSSP variant. Our results show that current quantum hardware is not yet competitive for large-scale constrained optimisation, although simplified formulations allow non-trivial valid solutions. Quantum-inspired hardware, by contrast, already supports industry-relevant problem sizes for the assignment-based formulations studied here and therefore provides a practically valuable reference point for near-term industrial optimisation workflows. Overall, the study highlights the importance of hardware-software co-design, formulation tuning, and systematic design-space exploration.

IX. Outlook

A next step is to move from case-specific studies towards systematic design-space exploration across formulations, hardware backends, and solver settings. Such work should separate formulation design, solver configuration, and end-to-end integration effects more clearly, and should account explicitly for whether the goal is optimal solutions, high-quality approximations, or fast feasible schedules. Future work should therefore extend the analysis to additional optimisation problems, improved hybrid workflows, and live production settings, including integration overheads and latency, to better understand the practical value of quantum and quantum-inspired optimisation in industrial environments.

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.

References

[1] A. and Bayerstadler, G. Becquin, J. Binder, T. Botter, H. Ehm, T. Ehmer, M. Erdmann, N. Gaus, P. Harbach, M. Hess, J. Klepsch, M. Leib, S. Luber, A. Luckow, M. Mansky, W. Mauerer, F. Neukart, C. Niedermeier, L. Palackal, R. Pfeiffer, C. Polenz, J. Sepulveda, T. Sievers, B. Standen, M. Streif, T. Strohm, C. Utschig-Utschig, D. Volz, H. Weiss, and F. Winter, “Industry quantum computing applications,” *EPJ Quantum Technology*, vol. 8, no. 1, Nov. 2021. [Online]. Available: http://dx.doi.org/10.1140/epjqt/s40507-021-00114-x

[2] T. Yue, W. Mauerer, S. Ali, and D. Taibi, *Challenges and Opportunities in Quantum Software Architecture*. Springer Nature Switzerland, 2023, p. 1–23. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-36847-9_1

[3] T. Gabor, S. Zielinski, S. Feld, C. Roch, C. Seidel, F. Neukart, I. Galter, W. Mauerer, and C. Linnhoff-Popien, *Assessing Solution Quality of 3SAT on a Quantum Annealing Platform*. Springer International Publishing, 2019, p. 23–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-14082-3_3

[4] A. Muluk, H. Akpolat, and J. Xu, "Scheduling problems — an overview," *Journal of Systems Science and Systems Engineering*, vol. 12, pp. 481–492, 01 2003. [Online]. Available: https://doi.org/10.1007/s11518-006-0149-z

[5] J. E. Mitchell, *Integer Programming: Branch-and-Cut Algorithms*. Cham: Springer Nature Switzerland, 2025, pp. 1–8. [Online]. Available: https://doi.org/10.1007/978-3-030-54621-2_287-1

[6] Gurobi Optimization, LLC, "Optimization methods: Choose the right approach," 2026, accessed 2026-04-20. [Online]. Available: https://www.gurobi.com/resources/blog/optimization-methods-choosing-the-right-approach-with-gurobi

[7] W. Mauerer and S. Scherzinger, "1-2-3 reproducibility for quantum software experiments," in *IEEE SANER*, 2022, pp. 1247–1248.

[8] M. R. Garey, D. S. Johnson, and R. Sethi, "The complexity of flowshop and jobshop scheduling," *Mathematics of Operations Research*, vol. 1, no. 2, pp. 117–129, 1976. [Online]. Available: http://www.jstor.org/stable/3689278

[9] K.-C. Ying, P. Pourhejazy, and Z.-R. Lin, "Scheduling with sequence-dependent setup times in short-term production planning: A main path analysis-based review," *Operations Research Perspectives*, vol. 14, p. 100340, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214716025000168

[10] M. Schönberger, I. Trummer, and W. Mauerer, "Hybrid mixed integer linear programming for large-scale join order optimisation," in *Proceedings of the VLDB Endowment*, 12 2025. [Online]. Available: https://arxiv.org/abs/2510.20308

[11] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014. [Online]. Available: https://arxiv.org/abs/1411.4028

[12] K. Blekos, D. Brand, A. Ceschini, C.-H. Chou, R.-H. Li, K. Pandya, and A. Summer, "A review on Quantum Approximate Optimization Algorithm and its variants," *Physics Reports*, vol. 1068, pp. 1–66, Jun. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0370157324001078

[13] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Physical Review X*, vol. 10, no. 2, Jun. 2020. [Online]. Available: http://dx.doi.org/10.1103/physrevx.10.021067

[14] Z. Jiang, E. G. Rieffel, and Z. Wang, "Near-optimal quantum circuit for grover's unstructured search using a transverse field," *Phys. Rev. A*, vol. 95, Jun 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.95.062317

[15] Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, "Xy mixers: Analytical and numerical results for the quantum alternating operator ansatz," *Physical Review A*, vol. 101, no. 1, Jan. 2020. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.101.012320

[16] A. Bärtschi and S. Eidenbenz, "Grover mixers for qaoa: Shifting complexity from mixer design to state preparation," in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2020, pp. 72–82.

[17] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, "Obstacles to variational quantum optimization from symmetry protection," in *Physical Review Letters*, vol. 125, no. 26. APS, 2020, p. 260505.

[18] D. J. Egger, J. Mareček, and S. Woerner, "Warm-starting quantum optimization," *Quantum*, vol. 5, no. 479, p. 479, 2021.

[19] A. Awasthi, A. Bärtschi, S. Eidenbenz, E. Le, and S. Zhu, "Quantum walks with iterated open quantum dynamics," *Communications Physics*, vol. 7, no. 1, Dec. 2023. [Online]. Available: http://dx.doi.org/10.1038/s42005-023-01453-0

[20] R. Tate, M. Moondra, B. Gard, M. Mohseni, S. E. Economou, and E. Barnes, "Bridging classical and quantum with sdp initialized warm-starts for qaoa," *Communications Physics*, vol. 7, no. 1, Nov. 2023. [Online]. Available: http://dx.doi.org/10.1038/s42005-023-01439-y

[21] V. Vijendran, R. S. Das, L. C. L. Hollenberg, C. D. Hill, and J. Thompson, "Large scale quantum approximate optimization algorithm on non-planar graphs with machine learning noise mitigation," 2024. [Online]. Available: https://arxiv.org/abs/2408.02342

[22] J. Sud, M. P. Harrigan, N. Rubin, N. M. Tubman, D. Lidar, E. Knill, and R. Babbush, "Qaoa-in-qaoa: solving large-scale maxcut problems on small quantum machines," 2024. [Online]. Available: https://arxiv.org/abs/2405.05087

[23] B. Montanez-Barrera, K. Kottmann, R. Orús, V. Dunjko, and M. Mauri, "Towards a universal qaoa protocol: Evidence of quantum advantage in solving combinatorial optimization problems," 2024. [Online]. Available: https://arxiv.org/abs/2405.09169

[24] C. Fingar, T. Müller, B. Zanger, and W. Lechner, "A parity quantum approximate optimization algorithm," 2024. [Online]. Available: https://arxiv.org/abs/2406.20084

[25] M. Streif and M. Leib, "Training the quantum approximate optimization algorithm without access to a quantum processing unit," *Quantum Science and Technology*, vol. 5, no. 3, p. 034008, 2020.

[26] T. Krüger and W. Mauerer, "Out of the Loop: Structural Approximation of Optimisation Landscapes and non-Iterative Quantum Optimisation," *Quantum*, vol. 9, p. 1903, Nov. 2025. [Online]. Available: https://doi.org/10.22331/q-2025-11-06-1903

[27] L. Schmidbauer, C. A. Riofrío, F. Heinrich, V. Junk, U. Schwenk, T. Husslein, and W. Mauerer, "Path matters: Industrial data meet quantum optimization," in *Proceedings of the IEEE International Conference on Quantum Computing and Engineering*, 5 2025. [Online]. Available: https://arxiv.org/abs/2504.16607

[28] J. M. Lorenz, T. Monz, J. Eisert, D. Reitzner, F. Schopfer, F. Barbaresco, K. Kurowski, W. van der Schoot, T. Strohm, J. Senellart, C. M. Perrault, M. Knufinke, Z. Amodjee, and M. Giardini, "Systematic benchmarking of quantum computers: status and recommendations," 2025. [Online]. Available: https://arxiv.org/abs/2503.04905

[29] V. Gierisch and W. Mauerer, "QEF: Reproducible and Exploratory Quantum Software Experiments," in *Service-Oriented Computing – ICSOC 2025 Workshops*, ser. Lecture Notes in Computer Science, vol. 16441. Springer Singapore, 2026, forthcoming. [Online]. Available: https://arxiv.org/abs/2511.04563

[30] M. W. e. a. Johnson, "Quantum annealing with manufactured spins," *Nature*, vol. 473, pp. 194–198, 2011.

[31] K. Boothby, P. Bunyk, J. Raymond, and A. Roy, "Next-generation topology of d-wave quantum processors," 2020. [Online]. Available: https://arxiv.org/abs/2003.00133

[32] T. Krüger and W. Mauerer, "Quantum Annealing-Based Software Components: An Experimental Case Study with SAT Solving," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 445–450. [Online]. Available: https://doi.org/10.1145/3387940.3391472

[33] L. Schmidbauer and W. Mauerer, "SAT Strikes Back: Parameter and Path Relations in Quantum Toolchains," in *2025 IEEE International Conference on Quantum Software (QSW)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2025, pp. 1–12. [Online]. Available: https://doi.org/10.1109/QSW67625.2025.00021

[34] L. Schmidbauer, E. Lobe, I. Schaefer, and W. Mauerer, "It's Quick to be Square: Fast Quadratisation for Quantum Toolchains," *ACM Transactions on Quantum Computing*, vol. 7, no. 3, pp. 1–46, 2026. [Online]. Available: https://doi.org/10.1145/3800943

[35] I. Sax, S. Feld, S. Zielinski, T. Gabor, C. Linnhoff-Popien, and W. Mauerer, "Approximate approximation on a quantum annealer," in *Proceedings of the 17th ACM International Conference on Computing Frontiers*, ser. CF '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 108–117. [Online]. Available: https://doi.org/10.1145/3387902.3392635

[36] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. Katzgraber, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," *Frontiers in Physics*, vol. 7, p. 48, 04 2019. [Online]. Available: http://dx.doi.org/10.3389/fphy.2019.00048

[37] F. Laboratories, "Fujitsu digital annealer: Architecture and applications," Fujitsu Technical Whitepaper, 2020.

[38] D. Venturelli, D. J. J. Marchand, and G. Rojo, "Quantum annealing implementation of job-shop scheduling," 2016. [Online]. Available: https://arxiv.org/abs/1506.08479

[39] P. Schworm, X. Wu, M. Glatt, and J. C. Aurich, "Solving flexible job shop scheduling problems in manufacturing with quantum annealing," *Production Engineering*, no. 17, pp. 105 – 115, 2024. [Online]. Available: https://nbn-resolving.de/urn:nbn:de:hbz:386-kluedo-79093

[40] F. Orts, A. M. Puertas, E. M. Garzón, and G. Ortega, "Quantum annealing to solve the unrelated parallel machine scheduling problem," in *Parallel Processing and Applied Mathematics: 14th International Conference, PPAM 2022, Gdansk, Poland, September 11–14, 2022, Revised Selected Papers, Part II*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 165–176. [Online]. Available: https://doi.org/10.1007/978-3-031-30445-3_14

[41] K. Kurowski, T. Pecyna, M. Slysz, R. Różycki, G. Waligóra, and J. Weglarz, "Application of quantum approximate optimization algorithm to job shop scheduling problem," *European Journal of Operational Research*, vol. 310, no. 2, pp. 518–528, None 2023. [Online]. Available: https://ideas.repec.org/a/eee/ejores/v310y2023i2p518-528.html

[42] T. Schwenzow, K. Wintersperger, H. Safi, O. Sicard, C. Niedermeier, C. Liebrecht, J. Franke, and S. Reitelshöfer, "Investigating the variational quantum eigensolver to solve scheduling for identical parallel machines with sequence-dependent setup-times," 08 2024.

[43] F. Lu and D. C. Marinescu, "An r —— cmax quantum scheduling algorithm," *Quantum Information Processing*, vol. 6, no. 3, p. 159–178, Jun. 2007. [Online]. Available: https://doi.org/10.1007/s11128-006-0048-8

[44] O. Rathore, A. Basden, N. Chancellor, and H. Kusumaatmaja, "Load balancing for high performance computing using quantum annealing," *Phys. Rev. Res.*, vol. 7, p. 013067, 01 2025. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.7.013067

[45] H. Safi, K. Wintersperger, and W. Mauerer, "Influence of hw-sw-co-design on quantum computing scalability," in *2023 IEEE International Conference on Quantum Software (QSW)*, 2023, pp. 104–115. [Online]. Available: https://www.lfdr.de/Publications/2023/SaWiMa23.pdf

[46] I. M. Ovacik and R. Uzhoy, "Worst-case error bounds for parallel machine scheduling problems with bounded sequence-dependent setup times," *Operations Research Letters*, vol. 14, no. 5, pp. 251–256, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/016763779390089Y

[47] M. E. Kurz and R. G. Askin, "Heuristic scheduling of parallel machines with sequence-dependent set-up times," *International Journal of Production Research*, vol. 39, no. 16, pp. 3747–3769, 2001. [Online]. Available: https://doi.org/10.1080/00207540110064938

[48] P. M. França, M. Gendreau, G. Laporte, and F. M. Müller, "A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times," *International Journal of Production Economics*, vol. 43, no. 2, pp. 79–89, 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/092552739600031X

[49] F. Sivrikaya-Şerifoğlu and G. Ulusoy, "Parallel machine scheduling with earliness and tardiness penalties," *Computers & Operations Research*, vol. 26, no. 8, pp. 773–787, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054898000902

[50] S. Radhakrishnan and J. A. Ventura, "Simulated annealing for parallel machine scheduling with earliness-tardiness penalties and sequence-dependent set-up times," *International Journal of Production Research*, vol. 38, no. 10, pp. 2233–2252, 2000. [Online]. Available: https://doi.org/10.1080/00207540050028070

[51] M. Bruni, S. Khodaparasti, and E. Demeulemeester, "The distributionally robust machine scheduling problem with job selection and sequence-dependent setup times," *Computers & Operations Research*, vol. 123, p. 105017, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054820301349

[52] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting qubits: Current state of play," *Annual Review of Condensed Matter Physics*, vol. 11, pp. 369–395, 2020.

[53] P. Krantz, M. Kjaergaard, F. Yan, T. Orlando, S. Gustavsson, and W. Oliver, "A quantum engineer's guide to superconducting qubits," *Applied Physics Reviews*, vol. 6, p. 021318, 06 2019. [Online]. Available: http://dx.doi.org/10.1063/1.5089550

[54] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, "Topological and subsystem codes on low-degree graphs with flag qubits," *Phys. Rev. X*, vol. 10, p. 011022, Jan 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.10.011022

[55] I. Quantum, "Ibm heron processor technology overview," https://quantum.cloud.ibm.com/docs/de/guides/processor-types, 2025, accessed: 2025-07-01.

[56] S. Thelen and W. Mauerer, "Predict and Conquer: Navigating Algorithm Trade-Offs with Quantum Design Automation," in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, 2025, pp. 591–602. [Online]. Available: https://doi.org/10.1109/QCE65121.2025.00071

[57] M. Khan, S. Shaheen, and F. Qureshi, "Comparative analysis of five sorting algorithms on the basis of best case, average case, and worst case," *International Journal of Information Technology and Electrical Engineering 2306-708X*, vol. 3, pp. 1–10, 05 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:63715675

[58] D-Wave, "Programming the D-Wave QPU: Setting the chain strength," D-Wave Systems Inc., Burnaby, BC, Canada, Tech. Rep., April 2020, white Paper. [Online]. Available: https://www.dwavequantum.com/media/vsufwv1d/14-1041a-a_setting_the_chain_strength.pdf

[59] F. Greiwe, T. Krüger, and W. Mauerer, "Effects of imperfections on quantum algorithms: A software engineering perspective," in *IEEE QSW*, 2023, pp. 31–42.

[60] S. Thelen, H. Safi, and W. Mauerer, "Approximating under the influence of quantum noise and compute power," in *IEEE QCE*, vol. 02, 2024, pp. 274–279.

[61] M. Wyrich and S. Apel, "Evidence tetris in the pixelated world of validity threats," in *Proceedings of the 1st IEEE/ACM International Workshop on Methodological Issues with Empirical Studies in Software Engineering*, ser. WSESE '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 13–16. [Online]. Available: https://doi.org/10.1145/3643664.3648203

[62] E. Teixeira, L. Fonseca, and S. Soares, "Threats to validity in controlled experiments in software engineering: what the experts say and why this is relevant," in *Proceedings of the XXXII Brazilian Symposium on Software Engineering*, ser. SBES '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 52–61. [Online]. Available: https://doi.org/10.1145/3266237.3266264