Scientific Research Publishing

Journal of Quantum Information Science, 2026, 16(3), 425-448
https://www.scirp.org/journal/jqis
ISSN Online: 2162-576X
ISSN Print: 2162-5751

Quantum Reinforcement Learning Framework for Multi-Rig Drilling Scheduling under Operational Constraints

Sulaiman Ureiga, Moodhi Aljouali

Drilling and Production, Saudi Aramco, Dhahran, Saudi Arabia
Email: sulaiman.ureiga@aramco.com, Moodhi.aljouali@aramco.com

How to cite this paper: Ureiga, S. and Aljouali, M. (2026) Quantum Reinforcement Learning Framework for Multi-Rig Drilling Scheduling under Operational Constraints. *Journal of Quantum Information Science*, **16**, 425-448.
https://doi.org/10.4236/jqis.2026.163015

Received: August 12, 2026
Accepted: September 18, 2026
Published: September 21, 2026

Abstract

The purpose of this research is to show the capabilities that quantum computing has to offer for AI. Therefore, the scheduling problem was chosen in order to demonstrate and compare the classical reinforcement learning (CRL) agent with the quantum computing reinforcement learning (QC-RL) agent. This constrained combinatorial optimization problem involves scheduling an oil field with several drilling rigs which are limited in capacity, area limitations, mandatory maintenance periods, weather restrictions and a mobilization costs which is considered along with well priorities and deadlines. Deep reinforcement learning (DRL) can learn adaptive scheduling policies, but value-based agents such as Deep Q-Networks (DQN) are known to exhibit high variance across random initializations. This paper investigates whether a hybrid quantum-classical value function can improve this reliability. We formulate realistic multi-rig drilling scheduling as a Markov Decision Process with action masking over an eight-constraint simulator (24 wells, 6 rigs, 3 areas, 80-day horizon) and compare a classical DQN against a hybrid Quantum DQN in which the value network's core is a variational quantum circuit via a 4-qubit ZFeatureMap encoding followed by a RealAmplitudes ansatz with 12 trainable quantum parameters, executed using a PyTorch-Qiskit interface. Both agents share an identical training and evaluation protocol (3 seeds × 30 episodes, greedy evaluation) to ensure a controlled comparison. Under greedy evaluation, the hybrid Quantum DQN completes 22.3/24 wells on average versus 17/24 for the classical DQN, achieves an 12.87% higher mean reward (9.12M vs 8.08M), and most significantly reduces the across-seed reward standard deviation by roughly 6.8× (0.202M vs 1.38M). These results indicate that the structurally constrained quantum value function acts as an implicit regularizer, yielding scheduling policies that are markedly more consistent across initializations than a substantially larger classical network. This simulated quan-

tum computing experiment demonstrates the potential for quantum computing in the field of AI.

Keywords

Quantum Reinforcement Learning, Variational Quantum Circuits, Deep Q-Network, Drilling Rig Scheduling, Combinatorial Optimization, Hybrid Quantum-Classical Machine Learning, Action Masking, Qiskit

1. Introduction

Developing an oilfield means drilling many wells with a limited fleet of rigs over a planning horizon that can stretch from weeks to months. Deciding which rig drills which well, and when, is known as the rig scheduling problem (RSP). It's a constrained combinatorial optimization problem that is difficult to solve computationally, which makes it a strong candidate for benchmarking CRL against QC-RL. Rigs are confined to specific geographic areas, go offline during maintenance, can't mobilize in bad weather, and incur travel and setup costs whenever they relocate, all while balancing well priority and deadlines against operating and mobilization costs. As the number of feasible rig-well-time assignments grows combinatorially, exact optimization quickly becomes intractable at field scale, which is what motivates heuristic and learning-based approaches.

Reinforcement learning (RL) fits sequential scheduling problems well because an agent can learn an adaptive policy that maps the current state of the field to the next assignment, instead of re-solving an optimization problem every time conditions change. Deep RL has already been applied across several petroleum domains, including well-control and production optimization, and more recently to drilling-location sequencing using graph neural networks combined with deep RL [1].

Value-based methods, in particular the Deep Q-Network (DQN) [2], are a natural fit here since assigning a specific rig to a specific well is inherently a discrete action. The problem is that DQN agents are known for high variance: the policy an agent ends up learning depends heavily on random initialization and exploration, so two runs that differ only in seed can produce very different deployed performance. For an operational tool, this kind of instability is a real problem, since planners need a schedule that is reliably good, not one that performs well on average but occasionally fails.

At the same time, quantum machine learning has emerged as a new route to function approximation. In the noisy intermediate-scale quantum (NISQ) era, variational quantum circuits (VQCs), parameterized quantum circuits whose gate angles are trained through classical optimization, have been used as drop-in replacements for neural networks inside RL agents [3]. Hybrid VQC-based DQN agents have already learned control and navigation tasks such as frozen-lake while using far fewer trainable parameters than an equivalent classical network [3] [4],

and later work using data re-uploading and output scaling shows that VQC-enhanced agents can match classical performance, with the circuit's structured parameter space playing a key role.

This paper connects these two threads, value-based reinforcement learning and quantum machine learning, by pairing a constraint-rich drilling scheduling environment with a direct comparison between classical and hybrid quantum-classical deep RL agents. Specifically, we ask whether a hybrid quantum-classical value function can make scheduling policies more robust and reliable, meaning less sensitive to random initialization, on a realistic multi-rig drilling problem. To answer this, we build a constraint-rich drilling scheduling simulator, formulate it as a Markov Decision Process (MDP) with action masking, and train two agents that are identical in every respect except their value network: a classical multilayer perceptron and a hybrid quantum network built around a 4-qubit VQC.

Contributions. The main contributions of this work are:

- **A constraint-rich drilling-scheduling environment.** We develop an open, Gym-style simulator for multi-rig drilling scheduling (24 wells, 6 rigs, 3 areas, 80-day horizon) that models eight operational constraints: area assignment, well release dates, rig maintenance, weather windows, shared crews, rig-up time, mobilization/idle costs, and deadlines, together with a feasibility-guaranteeing horizon and an action-masking interface.
- **A hybrid quantum DQN for scheduling.** We design a hybrid quantum-classical Q-network in which a 12-parameter variational quantum circuit replaces the hidden representation of a classical DQN, implemented using a PyTorch-Qiskit interface and trained end-to-end with masked Q-learning.
- **A fair, multi-seed evaluation protocol.** We compare the quantum and classical agents under an identical training and greedy-evaluation procedure, isolating the effect of the value network and reporting mean, variance, and worst-case performance rather than best-run results.

The remainder of the paper is organized as follows. Section 2 reviews related work in rig scheduling, deep RL, and quantum RL. Section 3 formulates the scheduling MDP. Section 4 describes the classical and hybrid quantum agents and the evaluation protocol. Section 5 details the experimental setup, and Section 6 presents results. Section 7 discusses the findings and limitations, and Section 8 concludes.

2. Related Work

This work sits at the intersection of three research threads: optimization and learning for drilling and rig scheduling, deep reinforcement learning for sequential decision-making, and quantum reinforcement learning with variational quantum circuits. We review each in turn and then position our contribution.

2.1. Drilling and Rig Scheduling

The rig scheduling problem (RSP) is about allocating well activities across a scarce

and expensive fleet of rigs, with the goal of avoiding production-delay losses while making efficient use of resources. It's been studied in operations research since linear-programming production-planning models first emerged in the early 1960s. In their systematic review, [5] proposed a classification that splits the RSP into four major classes: the drilling rig scheduling problem (DRSP), workover planning with and without routing, field development planning, and resource planning, where rigs are scheduled jointly with auxiliary resources such as offshore support vessels. Our formulation falls under the DRSP class. Specifically, it addresses scheduling drilling activities across a heterogeneous rig fleet under operational constraints, treated as an isolated decision separate from broader field development.

Within these RSP classes, mathematical programming forms an important strand of the literature, alongside a large body of heuristic and metaheuristic methods. Mixed-integer linear programming (MILP) formulations have been applied to workover rig scheduling with heterogeneous fleets, release dates, and rig-eligibility constraints, including arc-time-indexed formulations solved through branch-price-and-cut [6]. Related work on scheduling general maintenance tasks for oil and gas production assets has extended discrete-time MILP formulations to multitask work orders with precedence relations, parallel heterogeneous resources, and crews with differing capabilities and work shifts [7]. Since service and drilling durations are uncertain, stochastic extensions have also been proposed: deterministic integer-programming models for rig fleet sizing and scheduling have been paired with two-stage stochastic programs that optimize fleet size under intervention-time uncertainty [8]. Despite these advances, a persistent theme is that realistic instances scale poorly for exact solvers, motivating a large body of heuristic and metaheuristic methods [6].

Alongside optimization-based methods, data-driven approaches have also emerged, using historical records through text mining and regression to produce realistic estimates of intervention durations as inputs to optimization models [6]. A recognized gap in this literature is the scarcity of methods validated in real-world settings: a systematic review found that only nine of the surveyed studies had actually been implemented in practice, which limits collaboration between academia and industry [5]. Heuristic optimization has also been applied to drilling schedule generation, where [9] proposed two algorithms, a greedy economic-selection procedure and a search-space-reduction method, whose schedules achieved net present values exceeding at least 95% of randomly generated schedules and outperformed conventional economic-indicator well ranking. More recently, machine-learning approaches have started to appear as well, including a Markov-chain model that infers rig capabilities from historical rig movements and computes transition probabilities between well classes to automate drilling schedule generation [10].

Reinforcement learning (RL) extends this shift further by learning sequential decision policies for problems with long decision horizons. RL has been shown to

improve net present value (NPV) in well-production control compared to conventional optimization-based strategies, and it has also been applied to geosteering and other sequential drilling decisions under geological uncertainty [11]. Closest to our setting, Cárdenas Pantoja *et al.* [1] formulate drilling-schedule optimization as a Markov decision problem and solve it using a Deep Q-Network augmented with a soft-update mechanism, reporting higher and more stable NPV than a standard DQN baseline on a synthetic field-development case under subsurface uncertainty. These studies establish that learned policies work well for petroleum scheduling, but they remain entirely classical: none of them employs a quantum or hybrid quantum-classical value function, and, importantly, none systematically reports the across-seed reliability of the resulting policies, which is the specific gap our work targets.

2.2. Deep Reinforcement Learning and Its Variance

The Deep Q-Network (DQN) showed that a neural network trained with experience replay and a periodically updated target network can approximate the action-value function stably enough to reach human-level control on high-dimensional tasks [2]. DQN is a natural fit for scheduling because the decision at each step is simply assigning a specific rig to a specific well, or waiting. That said, value-based deep RL is well known to be sensitive to random seeds, initialization, and hyperparameters. Independently seeded runs can converge to policies of markedly different quality, and reporting only the single best run substantially overstates what typical deployed performance actually looks like. This reproducibility concern is exactly why rigorous evaluation trains across multiple seeds and reports the full distribution of outcomes (mean, variance, and worst case) rather than just the maximum. Our protocol follows this practice, treating variance as a primary object of investigation rather than a nuisance to be averaged away. A related challenge is the large discrete action space that's typical of assignment problems: at any given step, most rig-well pairs are infeasible, whether because the rig is busy, in maintenance, in the wrong area, or the well itself is unavailable. Simply penalizing invalid actions floods the agent with negative reward and destabilizes learning. Action masking, which restricts the policy to only the currently feasible actions, is a standard remedy that improves both stability and sample efficiency in large discrete action spaces, and we adopt it as the interface between our environment and both agents, regardless of their underlying value function.

2.3. Quantum Reinforcement Learning with Variational Quantum Circuits

Variational quantum circuits (VQCs), which are parameterized quantum circuits whose gate angles are optimized classically, are the dominant model of quantum machine learning in the noisy intermediate-scale quantum (NISQ) era, where they act as function approximators in place of neural networks. Within value-based RL, Chen *et al.* were the first to use a VQC as the action-value approximator in a

DQN, learning discrete environments such as FrozenLake with far fewer trainable parameters than an equivalent classical network [3]. Lockwood and Si extended hybrid VQC agents to continuous state spaces such as CartPole [4], and later attempted to scale a hybrid neural-network/VQC model to Atari environments, where the limited results exposed just how difficult it is to encode large observation spaces and to attribute learning between the classical and quantum components [12]. Skolik, Jerbi, and Dunjko refined value-based quantum agents further with data re-uploading and trainable output scaling on CartPole and FrozenLake, showing that the structured, low-dimensional parameter space of the circuit plays a central role in its behavior, and that output-range scaling is essential for value-based VQCs [13]. On the policy-gradient side, Jerbi *et al.* introduced parametrized quantum policies and analyzed their expressivity, and in certain special cases established provable separations from classical policies [14]. Work in this area continues on circuit design and optimization for QRL, including QCNN-inspired, noise-robust circuit variants for quantum deep Q-learning that improve robustness under device noise while adopting repeated-run evaluation to stabilize comparisons across random seeds [15].

A central theoretical question for these models is the trade-off between expressibility and trainability. Highly expressive, deeply entangled circuits tend to fall into barren plateaus, regions where gradients vanish exponentially as system size grows, which impedes trainability [16]. This isn't just an empirical observation either; Holmes *et al.* formally connected the two properties, showing that the more expressive an ansatz is, the flatter its cost landscape becomes, and therefore the harder it is to train [17]. On the other hand, the restricted and structured hypothesis class defined by a shallow variational quantum circuit can actually support favorable generalization. Caro *et al.* proved that quantum models can generalize from relatively few training samples, with error bounds that scale with the number of trainable gates rather than the dimension of the Hilbert space [18]. This tension, where too much capacity harms trainability and too little harms expressivity, yet a compact circuit can still generalize well, is precisely the lens through which we interpret our results: a small, 12-parameter circuit turns out to be expressive enough to represent good scheduling value functions while remaining constrained enough to yield policies that stay consistent across random initializations.

2.4. Positioning of This Work

Across these threads, three observations stand out. First, classical rig scheduling is dominated by mathematical programming and, more recently, classical RL, none of which examines quantum value functions or policy reliability. Second, DQN's seed sensitivity is widely acknowledged, but it's usually mitigated procedurally rather than treated as a target for improvement in its own right. Third, VQC-based agents are almost always evaluated on standard control benchmarks, and their prospective advantage tends to be framed in terms of parameter or sample efficiency rather than robustness. Our work departs from this on all three

counts: we apply a hybrid quantum DQN to a realistic, constraint-heavy drilling-scheduling problem, we adopt a controlled multi-seed protocol that makes across-seed variance a primary metric, and we identify variance reduction, meaning reliable completion of the field regardless of initialization, as the main practical benefit of the quantum value function, grounding this claim in the generalization and expressibility theory discussed above. To our knowledge, this is the first application of hybrid quantum reinforcement learning to drilling-rig scheduling.

3. Problem Formulation

We model multi-rig drilling scheduling as a finite-horizon Markov Decision Process (MDP) in which, on each simulated day, an agent either assigns a rig to a well or chooses to wait. The formulation follows the DRSP class of the RSP taxonomy, treating drilling as an isolated scheduling decision under operational constraints. This section covers the field and fleet, the state representation, the action space and masking mechanism, the operational constraints, the transition dynamics, the reward function, and the optimization objective along with its associated feasibility guarantee.

3.1. Field and Fleet

The field contains 24 wells, partitioned into three geographic areas (A, B, C) of eight wells each. A fleet of six rigs (two per area) drills the wells over a planning horizon of T = 80 days. Each well has a location, a drilling duration of 3 to 6 days, a release day (its earliest permissible start), an integer priority from 1 to 4 (higher is more valuable), a soft due day, and a hard deadline equal to the horizon T. Each rig has a travel speed, a daily operating cost, an assigned area, and a set of scheduled maintenance (down) days. A depth attribute is retained per well for descriptive purposes only and is not enforced as a constraint.

3.2. State

The observation on day t exposes every quantity on which feasibility and future cost depend. It comprises the normalized planning day; five values per rig (an availability flag equal to 1 if the rig is free and not in maintenance, the normalized remaining busy time, and the normalized x and y position of the rig, together with a maintenance flag equal to 1 on a scheduled down day); one occupancy flag per operating area, equal to 1 while the shared crew in that area is committed; and one status value per well. Well status is encoded on a single channel taking the value 0 for a well not yet started, 0.5 for a well whose drilling is in progress, and 1 for a well whose drilling has been completed, so that a committed well still being drilled is distinguishable from one not yet begun. The state dimension is:

$$StateSize = Normalized\ day + 5 \times N_{rigs} + N_{areas} + N_{wells} = 1 + 30 + 3 + 24 = 58$$

This gives the agent a fully observable summary of field progress: rig locations and remaining busy times, on which feasibility and future mobilization costs depend, are included directly in the observation.

3.3. Action Space and Action Masking

The action space contains one assignment action for every rig-well pair, plus one idle action:

$$ActionSpaceSize = \left(N_{rigs} \times N_{wells}\right) + 1 = \left(6 \times 24\right) + 1 = 145$$

Because most rig-well pairs are infeasible on any given day, the environment exposes a Boolean feasibility mask of length 145 indicating which actions are currently valid, and the agent selects only among feasible actions. This action-masking interface, standard for large discrete action spaces in which many actions are invalid [2], prevents the agent from being flooded with invalid-action penalties and is applied identically to both agents compared in this research.

3.4. Constraints

Eight operational constraints govern feasibility and dynamics:

1) Area: A rig may drill only wells located in its assigned area.

2) Release: A well cannot start before its release day.

3) Maintenance: Each rig has scheduled down days on which it is out of service. An assignment is infeasible if the rig is in maintenance on the decision day, or if any down day falls within the drilling interval; the start is otherwise deferred to the earliest clear window. Travel and rig-up days are not tested against maintenance.

4) Weather: No job may start on a weather-blocked day, the start is deferred to the next clear day. Once started, a job may span such days.

5) Shared crew: At most one active drilling job is permitted per area at any time, reflecting a shared crew or key equipment unit limit within each area.

6) Rig-up time: A fixed setup time of 1 day precedes drilling at a newly assigned well.

7) Costs: Operating cost, travel/delay cost, per-kilometer mobilization plus a fixed move-setup cost, and a daily idle spread cost charged for each free rig.

8) Deadlines: A soft penalty applies to completion after the due day. A well that cannot finish by the hard deadline T is infeasible to assign and exactly by the action mask.

3.5. Transition Dynamics

When a feasible assignment of a rig to a well is selected on day t, the environment first computes the travel time from the rig's current location to the well:

$$\boldsymbol{TravelTime} = \frac{\boldsymbol{distance}}{\boldsymbol{rig\_speed}}$$

The earliest start day accounts for travel, rig-up time, the rig's current availability, and weather:

$$\boldsymbol{Start} = \boldsymbol{next\_clear\_day}\left(\boldsymbol{MAX}\left(\left(\boldsymbol{t} + \boldsymbol{travel}_{\boldsymbol{time}} + \boldsymbol{rig}_{\boldsymbol{up}}\right), \boldsymbol{busy\_until}\right)\right)$$

where the resulting interval would overlap a scheduled down day of the assigned

rig, the start is deferred to the earliest day admitting a clear window of length duration, so a drilling job is never interrupted by maintenance. The well completion day is:

$$\boldsymbol{End = start + duration}$$

Here, next_clear_day() advances the start past any weather-blocked day, and busy_until is the day the rig becomes free. After assignment, the rig's busy_until marker is set to end, its location is updated to the well, and the well is recorded as committed. A well is marked complete only once its drilling interval has elapsed, that is, on the first planning day d ≥ end. Assignment and completion are therefore distinct events: between the assignment day and end the well is in progress, is excluded from the feasible action set, and does not yet count toward the completion total. Assignments are irrevocable, and the simulated day then advances by one. Constraints are checked over intervals rather than at the decision day alone. The busy interval recorded for a rig runs from the assignment day to completion and spans travel, rig-up, and drilling, so the shared-crew constraint permits at most one commitment per area throughout. Maintenance is evaluated over the drilling interval, and weather applies only to the start day. Because the same routine computes the interval for both the action mask and the state transition, the mask cannot admit an action that the transition would then execute in violation of a constraint; a separate audit routine re-verifies every realized schedule and reports no violations for any schedule in this paper. An episode terminates once every well has completed or the horizon T is reached. Wells still in progress at the horizon are not counted as complete, and the hard-deadline check guarantees that all committed jobs finish within T.

3.6. Reward

Each feasible assignment yields the immediate reward:

$$\begin{aligned} Reward = &\left(P \times priority\right) - \left(c \times day_{cost} \times duration\right) - \left(lambda \times \left(start - t\right)\right) \\ &- \left(mu \times distance + mu0\right) - \left(Lpen \times MAX\left(0, end, due\right)\right) \end{aligned}$$

With the parameter values:

$$P = 200000\left(priority\ value\ per\ unit\right),$$

$$c = 0.5\left(operating\ cost\ scale\right),$$

$$lambda = 100\left(delay\ cost\ per\ day\right),$$

$$mu = 500\left(mobilization\ cost\ per\ km\right),$$

$$mu0 = 20000\left(fixed\ move\ setup\ cost\right),$$

$$Lpen = 2000\left(late\ penalty\ per\ day\ past\ the\ soft\ due\ day\right)$$

In addition, each free rig incurs a daily idle spread cost:

$$IdleCostPerDay = 1500 \times \left(N\_free\_rigs\right)$$

At episode termination, a proportional completion bonus is awarded:

$$CompletionBonus = B\left(\left(wells\_completed\right)/\left(total\_wells\right)\right), with\ B = 6000000$$

The proportional form rewards partial progress rather than only a perfect schedule, and is intended to keep the marginal value of drilling each well positive so that completing the field is encouraged. We do not, however, prove that every feasible well yields a positive marginal return under all schedules and parameter settings, so full-field completion is not a guaranteed property of the reward-optimal policy. Feasibility of full completion is established by the heuristic planner in Section 3.7 rather than implied by the reward structure. In practice, this shaping largely reduces the problem to sequencing: the agent decides which feasible well to drill next, and when, to minimize operating, mobilization, idle, and lateness costs. Scheduling quality is therefore measured jointly by the number of wells completed and the total reward.

3.7. Objective and Feasibility Guarantee

The agent seeks a policy that maximizes the expected episodic return, defined as the sum of per-step rewards over an episode:

$$Return = sum\ of\ rewards\ over\ all\ steps$$

To ensure the problem is solvable rather than merely difficult, a constraint-aware heuristic planner establishes feasibility by assigning, in each area, the free rig to the highest-priority available well. For the problem instance studied in this paper, this planner completes all 24 wells with a makespan of 64 days. We set the horizon to:

$$T = 1.2 \times 64 + 3 = 80\ days$$

The instance is thus provably feasible: a sufficiently capable policy can complete every well within the horizon. Full completion of the field is therefore attainable, and the completion total is a well-defined quantity against which scheduling policies can be compared.

4. Methodology

We compare two agents that are identical in every respect except their value network, so that any difference in performance is attributable to the network alone. Both use the same environment, action masking, replay buffer, exploration schedule, optimizer, and training and evaluation procedures. Section 4.1 describes the shared deep Q-learning framework, 4.2 the classical baseline, 4.3 the hybrid quantum network, and 4.4 the controlled evaluation protocol.

4.1. Shared Deep Q-Learning Framework

Both agents are Deep Q-Networks (DQN). A Q-network estimates the action-value function Q(s, a), which represents the expected return obtained by taking action **a** in state **s**. The agent acts greedily with respect to Q among the currently feasible actions. Training uses standard DQN machinery: an experience-replay buffer, a target network updated by soft (Polyak) averaging, and epsilon-greedy

exploration restricted to feasible actions.

Using masked action selection, given state **s** and feasibility mask **m**, the local Q-network produces Q-values for all 145 actions. Infeasible actions are set to negative infinity so they can never be selected. With probability (1 - epsilon), the agent picks the highest-valued feasible action; with probability epsilon, it picks a random feasible action. This confines both exploration and exploitation to valid rigwell assignments.

Masked temporal-difference target. For a sampled transition (**s**, **a**, **r**, **s_next**, **done**) with next-state mask **m_next**, the bootstrap target is:

$$y = \boldsymbol{r} + gamma \times \left(\max over\ \ feasible\ a\_next\ of\ \ Q\_targets\left(s\_next, a\_next\right)\right) \times \left(1 - done\right)$$

Infeasible next-state actions are masked out before the maximum is taken. The local network is updated by minimizing the mean-squared error between its prediction and this target:

$$Loss = \left(Q\_local\left(\boldsymbol{s}, \boldsymbol{a}\right) - y\right)^2$$

where gamma is the discount factor and Q_target is the target network. Rewards are divided by a fixed scale factor before entering the buffer to keep regression targets numerically stable.

Figure 1 illustrates the architectures of the classical DQN and the proposed hybrid quantum-classical DQN, highlighting the replacement of the classical value-function approximator with a variational quantum circuit in the hybrid model.

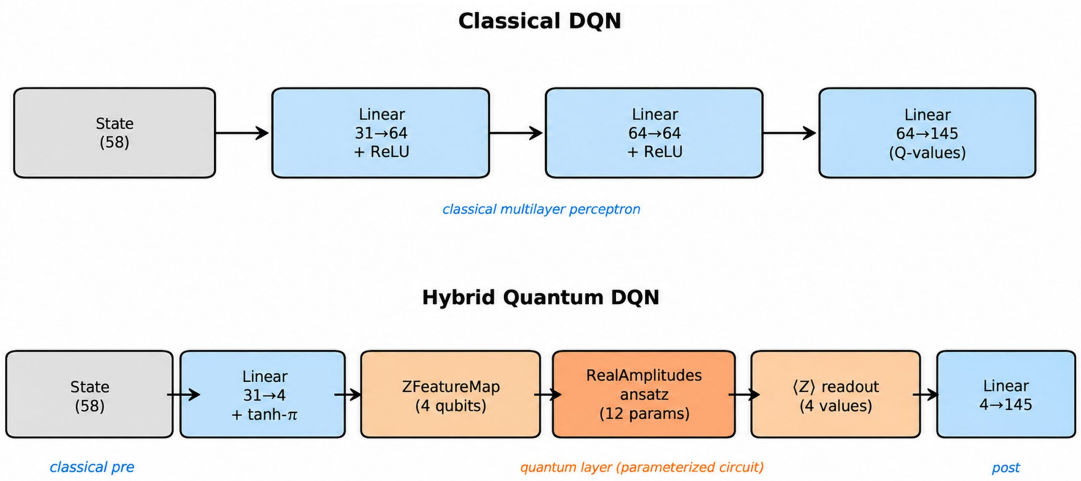

Figure 1. Classical and hybrid DQN.

4.2. Classical Baseline (Classical DQN)

The classical value network is a fully connected multilayer perceptron with the layer sizes:

58 -> 64 -> 64 -> 145

That is, an input layer of size 58, two hidden layers of 64 units each with ReLU activations, and an output layer of size 145 (one Q-value per action). This network

contains on the order of 17,300 trainable parameters and serves as the baseline against which the quantum agent is measured.

4.3. Hybrid Quantum Network (Quantum DQN)

The hybrid network replaces the hidden representation of the classical MLP with a variational quantum circuit (VQC), while keeping thin classical layers at the input and output. It has three stages arranged as a classical-quantum-classical sandwich.

1) Classical pre-layer. A linear layer maps the 58-dimensional state to 4 values, one per qubit. Each value is squashed by the hyperbolic tangent and scaled by pi, producing four rotation angles in the range [−pi, pi]:

$$Angle = \tanh\left(linear\left(state\right)\right)\times\pi$$

2) Quantum layer. The four angles are processed by a 4-qubit VQC consisting of two parts:

- a ZFeatureMap that encodes the four input angles into the qubits as data-dependent rotations (fixed, not trained); and
- a RealAmplitudes ansatz of trainable single-qubit Y-rotations interleaved with entangling CNOT gates, using two repetitions. The circuit is read out through the Pauli-Z expectation value on each qubit, yielding four real outputs in the range [−1, 1]. The number of trainable quantum parameters and the circuit depth are:

$$trainable\ quantum\ parameters = 12,\ circuit\ depth = 10$$

3) Classical post-layer. A linear layer maps the four measured expectation values to the 145 Q-values: 4 → 145

The VQC is implemented in Qiskit [19] and exposed to the classical training loop through a PyTorch interface, so the entire hybrid network, meaning the classical pre-layer, the quantum parameters, and the classical post-layer, is trained end-to-end by backpropagation using the same masked Q-learning loss as the classical agent. Gradients with respect to the quantum parameters are obtained using the parameter-shift rule via the framework's estimator. The quantum register runs on a state-vector simulator, with no hardware noise model applied. The idea behind this design is that the quantum core, with only 12 trainable parameters compared to roughly 17,300 in the classical MLP, defines a compact and structured hypothesis class. As discussed in Section 2.3, this kind of constrained class is associated with favorable generalization [18] while still staying shallow enough to avoid the barren-plateau training pathologies that come with highly expressive circuits [16].

4.4. Controlled Evaluation Protocol

To get a fair comparison and to capture reliability rather than just peak performance, both agents are trained and evaluated under an identical protocol.

Using multi-seed training, each agent is trained independently with three ran-

dom seeds (0, 1, 2), for 30 episodes per seed. All stochastic components, including network initialization, exploration, and replay sampling, are governed by the seed, so the classical and quantum runs differ only in the value network itself.

Once training is complete, each learned policy is evaluated greedily with exploration disabled (epsilon = 0), which ensures deterministic behavior. Starting from the initial field state, the agent selects the highest-valued feasible action at every step until the episode ends. We record two metrics from this greedy rollout: the number of wells completed (out of 24) and the total reward. Since the greedy rollout reflects how the learned policy would actually be deployed in practice, it gives a more faithful measure of policy quality than simply taking the best training episode.

For each agent, we report the mean and standard deviation of greedy-evaluation reward and wells completed across the three seeds, along with the worst-case (minimum) reward. Reporting the full distribution rather than just the single best run matters here, given the known seed sensitivity of value-based deep RL (Section 2.2), and it allows across-seed variance to serve as a primary measure of policy reliability.

5. Experimental Setup

This section specifies the problem instance, the software and hardware environment, the run configuration, and the metrics and figures used to evaluate the two agents.

5.1. Problem Instance

All experiments use a realistically synthetic field instance generated with a fixed random seed so that the classical and quantum agents face exactly the same problem. Table 1 shows a summary of the key instance parameters. The instance comprises:

number of wells = 24 (8 per area) number of areas = 3 (A, B, C) number of rigs = 6 (2 per area) planning horizon = 80 days

Well attributes are drawn from fixed ranges (Table 2 shows the full per-well specifications): drilling duration of 3 to 6 days, release day of 0 to 8, priority of 1 to 4, and a soft due day set to the release day plus the duration plus a random offset of 20 to 40 days. The due days therefore span roughly days 26 to 50, which is loose enough that all wells can be completed on time given the shared-crew queue. Each area is served by two rigs with slightly different travel speeds and daily operating costs, and each rig has one or two scheduled maintenance days; Table 3 presents the rig speeds, operating costs, and maintenance days. Also, four weather-blocked days are imposed across the horizon (days 9, 10, 23, 24), on which no drilling job may start.

Feasibility of the instance is established by the constraint-aware heuristic planner. The field layout consists of three well-separated clusters, one per area, each with its own pair of rigs positioned at an area base.

Table 1. Drilling-scheduling problem instance summary.

Parameter	Value
Wells/Areas/Rigs	24 wells, 3 areas, 6 rigs
Planning horizon (days)	80
Heuristic planner feasibility	All wells scheduled: True (24/24)
Heuristic planner makespan (days)	64
Heuristic planner reference spend	6,173,573
Action space | State size	145 | 58
Due-day range	26 .. 50

Table 2. Well specifications.

Well	Area	X (km)	Y (km)	Dur.	Rel.	Prio.	Depth	Due
W1	A	12.8	45.3	6	7	3	7	48
W2	A	42.7	37.4	6	4	4	8	43
W3	A	37.9	43	4	2	4	2	35
W4	A	18.5	41.1	3	1	1	8	33
W5	A	28.4	57.6	3	1	1	5	26
W6	A	36.9	63.1	5	2	2	3	32
W7	A	42.7	51.4	5	8	4	7	50
W8	A	37.1	46.6	6	1	4	1	42
W9	B	115.5	58	6	7	2	3	42
W10	B	114.8	65.8	4	2	2	2	37
W11	B	85.9	54.4	4	8	1	5	41
W12	B	131.7	57.1	6	2	2	6	34
W13	B	128	54.7	6	8	1	8	49
W14	B	110.5	58.5	6	8	1	3	41
W15	B	125.5	37.3	3	5	3	1	33
W16	B	120.4	50.8	4	1	1	8	26
W17	C	71.7	116.6	4	8	1	4	44
W18	C	81	118	6	2	3	2	40
W19	C	76.6	128.9	3	8	3	2	46
W20	C	88.6	115.8	6	5	2	2	35
W21	C	63.4	126.7	4	3	1	4	36
W22	C	53.6	116.9	6	5	3	4	33
W23	C	44.7	109.2	3	2	4	8	38
W24	C	77.2	117.7	6	2	3	3	40

Table 3. Rig specifications.

Rig	Area	Speed (km/day)	Day cost	Maintenance days
R1	A	31.8	100071	(11, 12)
R2	A	32.9	115601	(25)
R3	B	32.2	104494	(6)
R4	B	33.1	118457	(18, 19)
R5	C	31.4	104123	(14)
R6	C	34	114076	(28, 29)

5.2. Implementation

The environment is implemented as a custom Gym-style simulator in Python. The classical Q-network and the training loop use PyTorch. The quantum layer is built with Qiskit [19] and Qiskit Machine Learning, using a ZFeatureMap for data encoding and a RealAmplitudes ansatz, wrapped so that it can be trained inside PyTorch through the framework's neural-network connector. Quantum expectation values and their gradients are computed on a state-vector estimator primitive using the parameter-shift rule; no device-noise model is applied, so results reflect the ideal (noiseless) behavior of the circuit.

5.3. Run Configuration

Both agents are trained under the identical protocol of Section IV-D:

$$N_{seeds} = 3, (seeds\ 0,1,2)$$

$$episodes\ per\ seed = 30$$

$$maximum\ steps\ per\ episode = horizon + 2 = 82$$

For each seed, the agent is trained for 30 episodes, the best-reward network weights are retained, and the resulting policy is then evaluated greedily with exploration disabled. The quantum agent is executed on the same machine as the classical agent, on the state-vector simulator.

5.4. Metrics

Two primary metrics are reported, both obtained from the greedy (deployment) rollout:

1) Wells completed: the number of wells drilled by the greedy policy, out of 24. This measures whether the policy schedules the full field.

2) Total reward: the sum of per-step rewards over the greedy rollout, including all cost terms and the completion bonus. This measures schedule quality (priority captured minus operating, mobilization, idle, and lateness costs).

For each metric, we report the mean and standard deviation across the three seeds, and for reward, we additionally report the worst-case (minimum) value across seeds. The standard deviation across seeds is treated as the primary indicator of policy reliability.

6. Results

The experimental setup and results are presented through the following figures.

Figure 2 shows the spatial layout of the instance: 24 wells in three well-separated clusters, one per area, with each area's rig bases marked, so that mobilization cost is driven mainly by the ordering of wells within an area rather than by travel between them. Figure 3 reports greedy-evaluation reward and wells completed per seed; neither agent completes the field on any seed, and the classical results vary considerably more across seeds than the quantum results, with seed 2 accounting for most of that spread. Figure 4 overlays the training-reward curves, which begin near 9.9 million in the opening episodes, when exploration is close to random, and decline over the 30-episode budget for both agents, the two curves overlapping throughout with a somewhat narrower band for the quantum agent in later episodes.

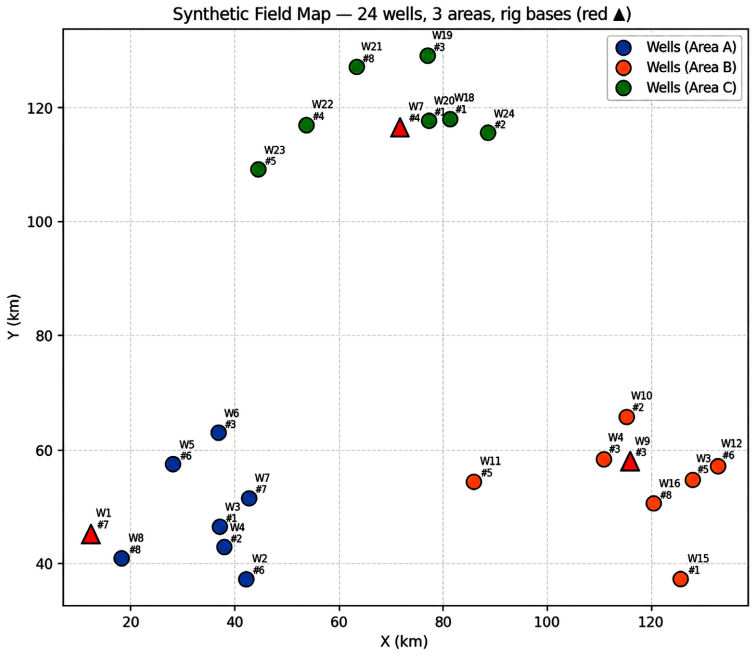

Figure 2. Field map: the spatial layout of the 24 wells in three areas with rig bases.

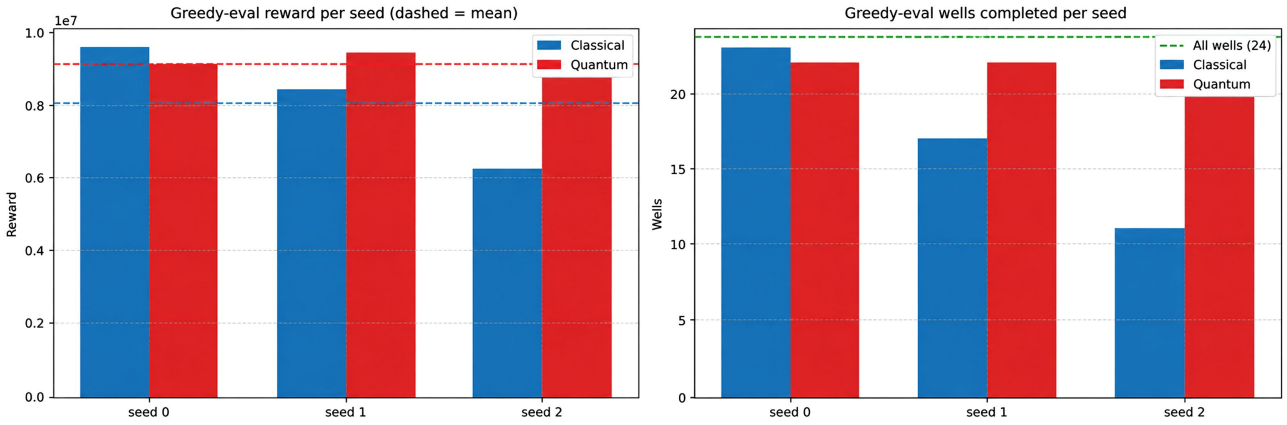

Figure 3. Greedy evaluation.

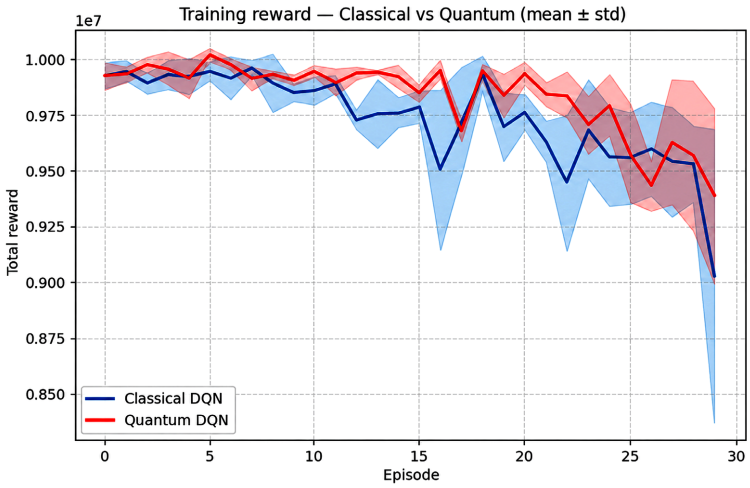

Figure 4. Quantum VS classical training reward.

6.1. Instance Feasibility and Baselines

The constraint-aware heuristic planner completes all 24 wells within a makespan of 64 days, so the horizon is set to 80 days accordingly. The Gantt chart in **Figure 5** shows all 24 drilling jobs placed within the horizon while respecting weather-blocked days and rig maintenance, which confirms the instance is solvable by construction. One notable structural feature is that the heuristic schedule only uses one rig per area at a time. This follows directly from the shared-crew constraint, which serializes drilling within each area and leaves the second rig as a backup for maintenance periods. As a result, the agent's main optimization lever is the timing and ordering of wells within each area, rather than parallel rig operation. The action mask guarantees that every executed assignment satisfies all eight operational constraints, so no policy can produce an infeasible schedule. Differences between methods therefore arise not from constraint satisfaction but from the ordering and timing of assignments, which determine mobilization distance, idle rig-days, and lateness against soft due dates.

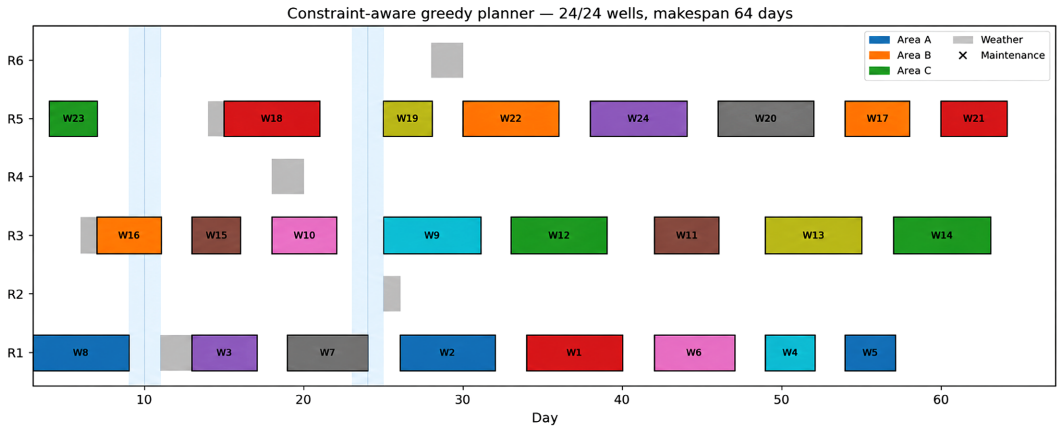

Figure 5. Gantt chart of the constraint-aware heuristic planner schedule. All 24 wells are completed within a 64-day makespan, within the 80-day planning horizon.

6.2. Training Behavior

Both agents were trained for three seeds of 30 episodes each. The overlaid comparison in Figure 4 shows the two training-reward curves interleaving closely throughout, beginning near 9.9 million in the opening episodes, when exploration is close to random, and declining over the training budget to roughly 9.0 million for the classical agent and 9.4 million for the quantum agent, with the quantum curve showing a slightly tighter band in later episodes. Neither curve indicates convergence within the 30-episode budget. Judged by training performance alone, the two agents look comparable, the differences between them appear under greedy evaluation.

6.3. Greedy-Policy Evaluation

Table 4 summarizes the greedy (deployment) evaluation, in which exploration is disabled, and each trained policy is rolled out deterministically.

Table 4. Greedy-evaluation results (3 seeds).

Method	Greedy reward (mean ± std)	Wells (mean/best)
Classical DQN	8.08M ± 1.38M	17/24
Quantum DQN	9.12M ± 0.20M	22.3/24

The per-seed results reveal the source of this gap (Figure 3, per-seed bars). The classical agent completes 23 wells on seed 0 but falls to 17 on seed 1 and 11 on seed 2, with reward declining correspondingly from roughly 9.5 million to 6.2 million. The quantum agent completes 22, 22, and 23 wells across the three seeds, with rewards clustered near 9.1 million. Neither agent completes the full field on any seed.

The head-to-head differences (Quantum minus Classical) are:

$$reward\_difference = +1,039,865\left(+12.87\%\right)$$

$$wells\_difference = +5.30\ wells\left(mean, out\ of\ 24\right)$$

$$reward\ std = classical\ 1{,}375{,}812\ vs\ quantum\ 202{,}359$$

6.4. Robustness

The most significant result is the difference in variance. The across-seed reward standard deviation is 1.38 million for the classical agent and 0.202 million for the quantum agent—a reduction of approximately 6.8 times. The box plot in Figure 6 shows the classical greedy-evaluation reward spanning from roughly 6.2 million to 9.5 million, whereas the quantum reward is compact around 9.1 million. The robustness plot in Figure 7 highlights the worst-case (minimum) reward for each method: the classical worst case is 6.52 million (seed 2), while the quantum worst case is 8.89 million (seed 1). The quantum policy therefore has a substantially higher performance floor.

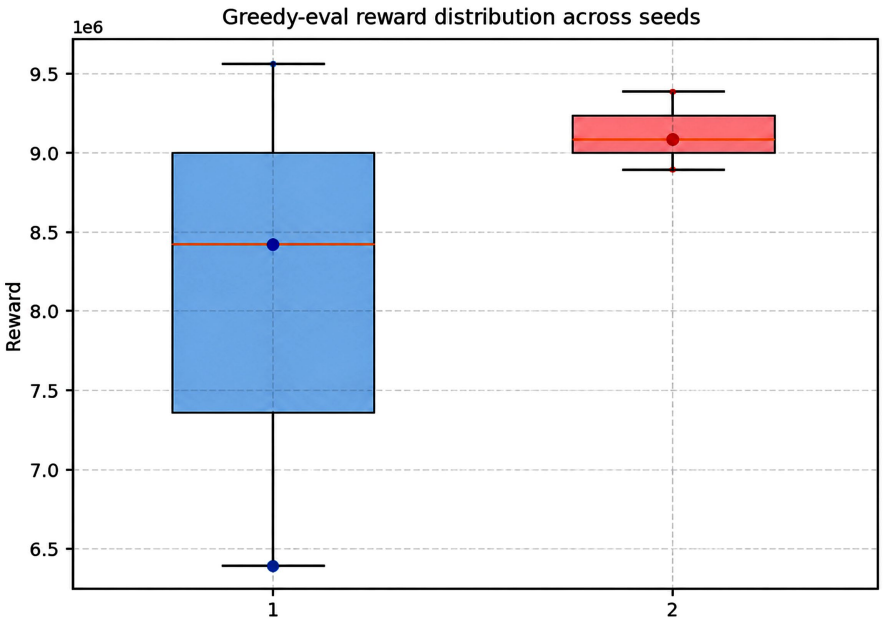

Figure 6. Distribution of greedy-evaluation rewards across random seeds for the classical and hybrid quantum-classical agents.

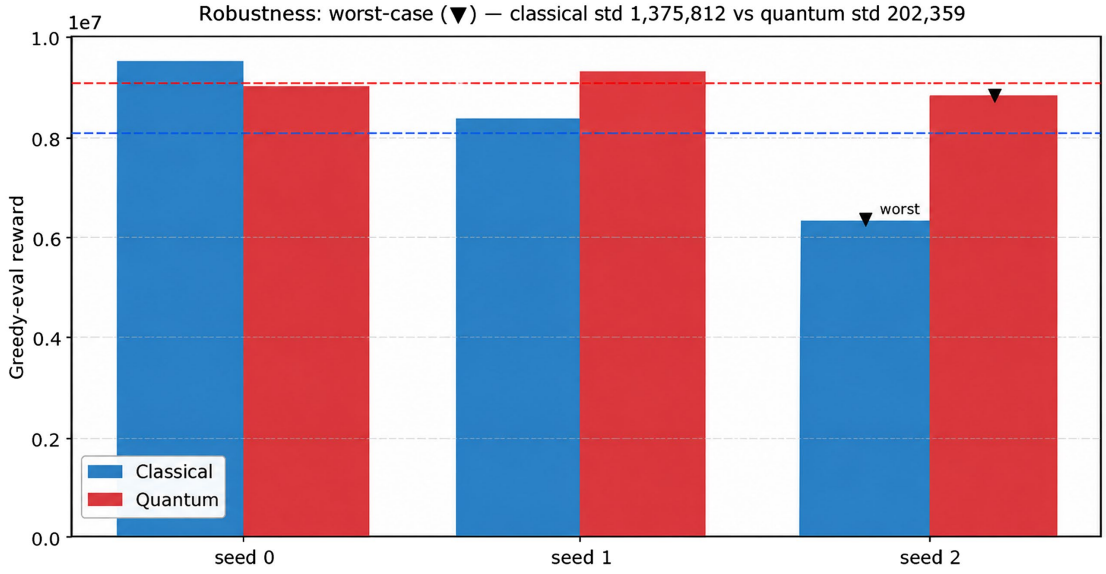

Figure 7. Robustness diagram.

6.5. Schedule and Cost Analysis

To see how the two policies differ in practice, we replayed the best-seed rollout for each agent and broke down its reward. The decompositions are closely matched in structure: priority value, operating cost, and completion bonus are of similar magnitude for both agents, with only a small lateness penalty separating them. The difference between the methods therefore does not lie in the cost efficiency of an individual schedule, but in how consistently each agent performs across seeds.

6.6. Quantum Circuit

For completeness, the trained quantum model uses a 4-qubit circuit with 4 encoding parameters (ZFeatureMap), 12 trainable parameters (RealAmplitudes, two repetitions), and a compiled depth of 10 (Figure 8, circuit and architecture diagrams). The full hybrid network—classical pre-layer, quantum core, and classical post-layer—is therefore dominated in parameter count by its two thin classical layers, while the quantum core that shapes the value function's structure has only 12 trainable parameters, compared with roughly 17,300 in the classical baseline MLP.

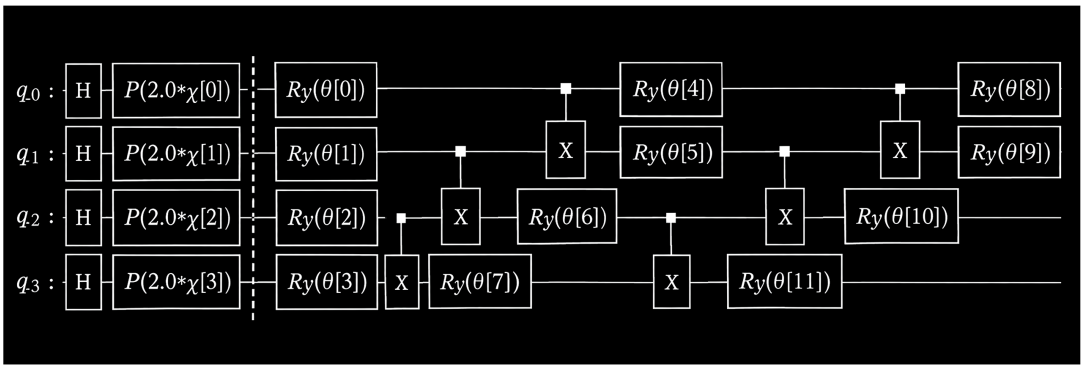

Figure 8. Circuit and architecture diagrams.

Summary of Results

In summary: i) the instance is feasible, with all 24 wells schedulable within 80 days by the constraint-aware heuristic planner; ii) neither agent completes the field under greedy evaluation, with the classical agent completing 17.0 wells on average and the quantum agent 22.3; iii) the quantum agent earns 12.9% higher mean reward (9.12M vs 8.08M) and shows lower across-seed reward variance, with a standard deviation of 0.20M against 1.38M, and a higher worst case (8.89M vs 6.24M). Across these three seeds, the quantum value function's main difference from the classical baseline is consistency rather than peak performance.

7. Discussion

This section interprets the results, explains the likely mechanism behind the quantum agent's advantage, states the limitations, and identifies threats to validity.

7.1. Interpretation: Reliability, Not Peak Performance

The difference between the two agents on this problem lies in consistency rather than peak capability. On their best runs the classical and quantum policies produce schedules of comparable quality, with closely matched reward composition (Section 6). The two methods diverge in how consistently they reach that quality: across three seeds, the classical policy's greedy-evaluation reward has a standard deviation of 1.38 million against 0.20 million for the quantum policy, and a worst case of 6.24 million against 8.89 million. Neither agent completes the field on any

seed, the classical agent averaging 17.0 of 24 wells and the quantum agent 22.3. For an operational scheduling tool a dependable floor matters, but with three seeds and no hypothesis test this difference is a description of these runs rather than an established property of either method.

This tempers what the present results can claim for quantum machine learning on this class of problem. We do not claim quantum superiority, which noiseless small-scale simulation cannot support. Nor do the results yet establish a reliability advantage in any statistical sense. What they show is that a value network built around a 12-parameter variational circuit reached performance comparable to a classical network with roughly 17,300 parameters on this instance, which is a feasibility result for the formulation rather than a demonstration of advantage.

7.2. Why the Quantum Core Regularizes

One candidate explanation for the observed spread is the compact and structured hypothesis class defined by the variational quantum circuit. The quantum core has only 12 trainable parameters against roughly 17,300 in the classical multilayer perceptron. A network with that much capacity can fit a high-reward exploratory trajectory closely, and may then produce a deterministic policy that is sharp but brittle. The circuit constrains what the value network can represent, since its parameters enter only as rotation angles on a small entangled register, and this may act as an implicit regularizer. Such an interpretation would be consistent with theory relating the small, structured parameter space of shallow circuits to favorable generalization, where bounds are governed by the number of trainable gates rather than the dimension of the Hilbert space [18], and the circuit is shallow enough (depth 10, four qubits) to remain trainable, avoiding the barren-plateau regime that afflicts deeply entangled circuits [16].

7.3. Limitations

Despite these promising results, several limitations are worth considering when interpreting the findings and assessing how broadly they apply. There are four main ones: the use of simulation rather than real quantum hardware, the evaluation of a single modest-scale instance, the limited number of seeds and training episodes, and the absence of controlled experiments that would pin down the mechanism behind the observed reliability advantage.

1) All quantum computations are performed on a noiseless state-vector simulator. Real quantum hardware introduces gate noise, decoherence, and measurement error, all of which would degrade the circuit's behavior. The results reported here should therefore be seen as an idealized upper bound on quantum performance, not a hardware-validated one.

2) Our work uses a single synthetic field instance (24 wells, 6 rigs, 3 areas) and a 4-qubit circuit. Whether the reliability advantage holds up across many instances, larger fields, and larger circuits still remains to be shown.

3) Results are reported over three seeds and 30 episodes per seed, without con-

fidence intervals, hypothesis tests, or a formal variance-comparison test. Three seeds are insufficient to establish either a difference in means or a difference in variance between the two methods, so the differences reported in Section 6 should be read as descriptive of these particular runs rather than as statistically established. Seed counts adequate for inference, together with Welch and Brown-Forsythe tests, are left to future work.

4) Model weights are retained for each seed by the highest ε-greedy training reward rather than by a separate greedy validation rollout. A high exploratory return does not imply a strong deterministic policy, so the retained checkpoint may not be the best-performing one, and this choice contributes to the dispersion of the reported results. Implementing validation-based checkpoint selection properly requires held-out validation instances, since selecting on a greedy rollout of the deployment instance itself would be circular.

7.4. Future Work

Future work should extend the research in four directions: i) evaluation across many field instances and larger problem sizes to test whether the reliability advantage scales; ii) execution on noisy simulators and real quantum hardware to assess robustness to device noise; iii) capacity-matched ablations—for example, shrinking the classical network or enlarging the circuit—to test the regularization hypothesis directly; and iv) exploration of alternative encodings and ansatz designs, and of policy-gradient quantum agents, to determine which architectural choices drive the variance reduction.

8. Conclusions

This paper investigated whether a hybrid quantum-classical value function can improve the reliability of deep reinforcement learning policies for multi-rig drilling scheduling. We formulated the problem as a finite-horizon Markov Decision Process with action masking over an eight-constraint simulator (24 wells, 6 rigs, 3 areas, 80-day horizon) whose feasibility was established by a constraint-aware heuristic planner, and compared a classical Deep Q-Network against a hybrid Quantum DQN whose value network's core is a 4-qubit variational quantum circuit with only 12 trainable parameters. The two agents were identical in every respect except the value network, and both were trained and evaluated under the same controlled, multi-seed protocol.

Under greedy evaluation, neither agent completed the full field: the quantum agent completed 22.3 of 24 wells on average against 17.0 for the classical baseline, and earned 12.9 percent higher mean reward (9.12 million versus 8.08 million). The across-seed reward standard deviation was 0.20 million for the quantum agent against 1.38 million for the classical one, with a higher worst case (8.89 million versus 6.24 million). With three seeds and no hypothesis test, these differences describe the runs we performed rather than establishing a general property of either method. A compact hypothesis class may act as an implicit regularizer,

but separating a circuit effect from the effect of reduced parameter count would require the parameter-matched classical control we identify in Section 7.3.

The practical interest of this work lies less in a performance claim than in the formulation itself: a value network built around 12 quantum parameters reached performance comparable to a classical network with roughly 17,300 on this instance. To our knowledge, this is the first application of hybrid quantum reinforcement learning to drilling-rig scheduling. Our setup relies on a noiseless simulator, a single synthetic instance, and a small number of seeds, so future work will test whether the observed differences persist across many instances and larger fields, under realistic device noise and on actual quantum hardware, with seed counts sufficient for statistical inference and capacity-matched ablations. We hope this research encourages further examination of robustness, rather than raw performance, as a practical direction for quantum machine learning in industrial optimization.

Use of Generative AI

During the preparation of this work, the author used a generative AI assistant to help draft and edit the manuscript text and to assist in developing and documenting the accompanying code. All technical content, experimental design, code execution, results, and interpretations were by the authors.

Author Contributions

Sulaimn Uriega developed and implemented the quantum circuit, while Moodhi Aljouali developed and implemented the experimental environment. Both authors contributed equally to the analysis, interpretation of the results, and writing and revision of the manuscript.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Cárdenas Pantoja, N.J., Abdin, A., Baraldi, P., Pinciroli, L., Forello, A., Dovera, L., *et al.* (2025) Optimal Drilling Scheduling in Field Development Planning by Deep Reinforcement Learning. *SPE Reservoir Simulation Conference*, Galveston, 25-27 March 2025, SPE-223917-MS. https://doi.org/10.2118/223917-ms

[2] Chen, S.Y., Yang, C.H., Qi, J., Chen, P., Ma, X. and Goan, H. (2020) Variational Quantum Circuits for Deep Reinforcement Learning. *IEEE Access*, **8**, 141007-141024. https://doi.org/10.1109/access.2020.3010470

[3] Lockwood, O. and Si, M. (2020) Reinforcement Learning with Quantum Variational Circuit. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, **16**, 245-251. https://doi.org/10.1609/aiide.v16i1.7437

[4] Santos, I.M., Hamacher, S. and Oliveira, F. (2021) A Systematic Literature Review for the Rig Scheduling Problem: Classification and State-of-the-Art. *Computers & Chemical Engineering*, **153**, Article ID: 107443. https://doi.org/10.1016/j.compchemeng.2021.107443

[5] Santos, I.M., Hamacher, S. and Oliveira, F. (2023) A Data-Driven Optimization Model for the Workover Rig Scheduling Problem: Case Study in an Oil Company. *Computers & Chemical Engineering*, **170**, Article ID: 108088. https://doi.org/10.1016/j.compchemeng.2022.108088

[6] Achkar, V.G., Cafaro, V.G., Méndez, C.A. and Cafaro, D.C. (2019) Discrete-Time MILP Formulation for the Optimal Scheduling of Maintenance Tasks on Oil and Gas Production Assets. *Industrial & Engineering Chemistry Research*, **58**, 8231-8245. https://doi.org/10.1021/acs.iecr.9b00861

[7] Fernández Pérez, M.A., Oliveira, F. and Hamacher, S. (2018) Optimizing Workover Rig Fleet Sizing and Scheduling Using Deterministic and Stochastic Programming Models. *Industrial & Engineering Chemistry Research*, **57**, 7544-7554. https://doi.org/10.1021/acs.iecr.7b04500

[8] Lamas, L.F., Botechia, V.E., Schiozer, D.J. and Delshad, M. (2017) Optimization for Drilling Schedule of Wells in the Development of Heavy Oil Reservoirs. *Brazilian Journal of Petroleum and Gas*, **11**, 165-173. https://doi.org/10.5419/bjpg2017-0014

[9] Nooruddin, H., Zahrani, A., Shahri, M. and Alsousy, A. (2022) Automatic Generation of Drilling Schedules Using Machine Learning: A Paradigm Shift in Planning and Resource Allocation. *International Petroleum Technology Conference*, Riyadh, 21-23 February 2022, IPTC-22147-MS. https://doi.org/10.2523/iptc-22147-ms

[10] Muhammad, R.B., Alyaev, S. and Bratvold, R.B. (2026) Optimal Sequential Decision-Making in Geosteering: A Reinforcement Learning Approach. *Geoenergy Science and Engineering*, **258**, Article ID: 214304. https://doi.org/10.1016/j.geoen.2025.214304

[11] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., *et al.* (2015) Human-Level Control through Deep Reinforcement Learning. *Nature*, **518**, 529-533. https://doi.org/10.1038/nature14236

[12] Lockwood, O. and Si, M. (2021) Playing Atari with Hybrid Quantum-Classical Reinforcement Learning. arXiv: 2107.04114. http://arxiv.org/abs/2107.04114

[13] Skolik, A., Jerbi, S. and Dunjko, V. (2022) Quantum Agents in the Gym: A Variational Quantum Algorithm for Deep Q-Learning. *Quantum*, **6**, 720. https://doi.org/10.22331/q-2022-05-24-720

[14] Jerbi, S., Gyurik, C., Marshall, S.C., Briegel, H.J. and Dunjko, V. (2021) Parametrized Quantum Policies for Reinforcement Learning. arXiv: 2103.05577. http://arxiv.org/abs/2103.05577

[15] Yu, L., Yu, W., Chen, Y. and Zhang, C. (2026) QCNN-Inspired Variational Circuits for Enhanced Noise Robustness in Quantum Deep Q-Learning. *Information*, **17**, Article 250. https://doi.org/10.3390/info17030250

[16] Larocca, M., Thanasilp, S., Wang, S., Sharma, K., Biamonte, J., Coles, P.J., *et al.* (2025) Barren Plateaus in Variational Quantum Computing. *Nature Reviews Physics*, **7**, 174-189. https://doi.org/10.1038/s42254-025-00813-9

[17] Holmes, Z., Sharma, K., Cerezo, M. and Coles, P.J. (2022) Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus. *PRX Quantum*, **3**, Article ID: 010313. https://doi.org/10.1103/prxquantum.3.010313

[18] Caro, M.C., Huang, H., Cerezo, M., Sharma, K., Sornborger, A., Cincio, L., *et al.* (2022) Generalization in Quantum Machine Learning from Few Training Data. *Nature Communications*, **13**, Article No. 4919. https://doi.org/10.1038/s41467-022-32550-3

[19] Javadi-Abhari, A., *et al.* (2024) Quantum Computing with Qiskit. arXiv: 2405.08810.