Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale,
Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Enhanced Reinforcement Learning-driven Process Design via Quantum Machine Learning

Austin Braniff[a], Fengqi You[b], and Yuhe Tian[a]*

[a] Department of Chemical and Biomedical Engineering, West Virginia University, Morgantown, WV, United States
[b] Smith School of Chemical and Biomolecular Engineering, Cornell University, Ithaca, NY, United States
* Corresponding Author: yuhe.tian@mail.wvu.edu.

ABSTRACT

In this work, we introduce a quantum-enhanced reinforcement learning (RL) framework for process design synthesis. RL-driven methods for generating process designs have gained momentum due to their ability to intelligently identify optimal configurations without requiring pre-defined superstructures or flowsheet configurations. This eliminates reliance on prior expert knowledge, offering a comprehensive and robust design strategy. However, navigating the vast combinatorial design space poses computational challenges. To address this, a novel approach integrating RL with quantum machine learning (QML) is proposed. QML leverages theoretical advantages over classical methods to accelerate searches in large spaces. Built upon our prior work, the approach begins with a maximum set of available unit operations, represented in a flowsheet structure using an input-output stream matrix as RL observations. A Deep Q-Network (DQN) algorithm trains a parameterized quantum circuit (PQC) in place of a classical neural network (NN). The design structures generated by the RL agent are optimized using the IDAES Process Systems Engineering Framework, with the optimization objectives used as rewards to RL (e.g., cost, productivity). This quantum-enhanced algorithm is performed on a hydrodealkylation process case study, showcasing its efficiency and improved performance in navigating complex design spaces.

Keywords: Process Design, Process Synthesis, Quantum Computing, Reinforcement Learning

1. INTRODUCTION

In the modern era of digitalization, industries are undergoing rapid transformation, driven by advancements in artificial intelligence (AI) and machine learning (ML). These technologies have fundamentally reshaped how chemical and energy processes are designed and optimized, enabling greater efficiency, adaptability, and innovation. Among the many applications of ML, reinforcement learning (RL) has emerged as a particularly promising approach in process systems engineering (PSE) [1-2].

RL-driven process synthesis offers a systematic methodology to identify optimal process configurations without relying on predefined superstructures or flowsheet configurations. By allowing an agent to automatically and intelligently explore and evaluate design decisions, RL minimizes the influence of prior assumptions and heuristics. Despite the potential, RL-driven process synthesis faces significant challenges including the vast combinatorial design space, which often renders the search computationally expensive or even intractable.

Quantum computing presents an exciting opportunity to overcome these limitations by harnessing quantum machine learning (QML) [3-4]. QML algorithms have demonstrated the potential for computational speedups in solving problems that are traditionally constrained by the limitations of classical computation [5]. Leveraging advances in quantum hardware and algorithms, QML can accelerate the search process and address bottlenecks in RL-driven process synthesis. Building on our prior research, this work proposes a novel integration of QML with RL to enhance the efficiency, robustness, and performance of process design frameworks.

This paper introduces the principles, methodologies, and innovations underpinning our quantum-enhanced RL framework for process flowsheet synthesis (Fig. 1). Section 2 reviews the methodology for classical RL-driven process design, touching on key features of

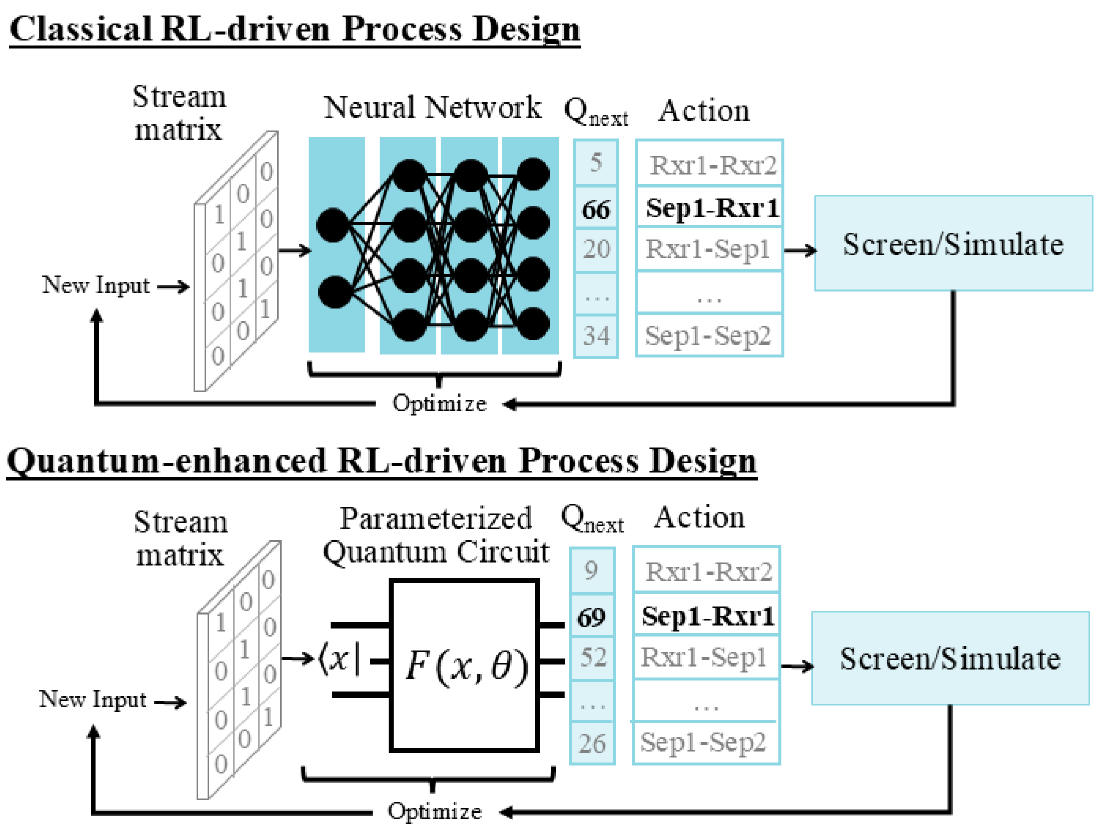

Figure 1: Conceptual overview distinguishing between the classical and quantum methodologies.

the algorithm. Section 3 introduces the topic of quantum machine learning and presents our integrated quantum-enhanced framework. Section 4 applies the framework to a representative case study on hydrodealkylation process design, highlighting the methodological advantages over conventional RL approaches. Section 5 concludes with a discussion of ongoing work.

2. RL-DRIVEN PROCESS SYNTHESIS

In this section, we briefly review the fundamentals of Deep Q-Network (DQN) algorithm and the prior work [2,7] on RL-driven process synthesis which set up the foundation for the current work.

2.1 Deep Q-Network

The DQN algorithm is a form of Q-Learning and is essential to model-free RL. It is designed to address the challenges of high-dimensional state-action spaces using neural networks (NNs). The traditional Q-Learning algorithm approximates the optimal action-value function $Q^*(s,a)$, which represents the maximum cumulative reward achievable from a given state s when taking action a and following the optimal policy thereafter. However, as the state space becomes more complex, approximating the Q-function $Q(s,a)$ becomes computationally intensive. DQN overcomes this limitation by employing an NN as a function approximator to estimate the Q-function instead as $Q(s,a,\theta)$ where θ denotes the network parameters. The NN serves as a computationally friendly alternative over other methods such as tabular Q-Learning.

Central to this algorithm is the Bellman equation, expressed in Eq. 1 where s and a are the current state action pair, s' and a' are future state-actions, r is the instantaneous reward, and γ is a discount factor for future rewards. This equation recursively relates present state-action values to the rewards of future values enabling optimal decision making that considers all future paths.

$$Q(s,a) = r + \gamma \max_{a'} Q\,(s',a') \tag{1}$$

In DQN, the Q-functions on the right and left of Eq. 1 are approximated by NNs. The right being the target network which is periodically updated and the left being the evaluation network which is updated at each transition step. This update is performed by minimizing the mean squared error (MSE) between the current Q-value estimate and the target value, as seen in Eq. 2 where $\mathcal{L}$ is the loss function (in this case MSE), θ is the vector of tunable parameters in the evaluation network, and θ^- is the vector of tunable parameters in the target network.

$$\min_{\theta} \mathcal{L}(\theta) = E\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\right)^2\right] \quad (2)$$

When the target network is updated, it becomes a copy of the evaluation network at that time (i.e., $\theta^- = \theta$). This approach helps to achieve stability and mitigate bias in the algorithm. In the landmark paper for DQN [6], the authors have established a few other key techniques that help make this algorithm efficient and successful such as experience replay and an epsilon-greedy approach to action selection. The experience replay allows the expectation to be taken over a randomly sampled batch of stared state-action-reward transitions which reduces both recency bias and correlations between consecutive transitions. The epsilon-greedy policy is utilized as a method to balance the tradeoffs between exploration and exploitation. In the early stages of learning random actions are taken more often to explore the state-space while later, once the Q-function is better approximated, the predicted optimal actions are taken more frequently.

2.2 Classical RL-Driven Process Synthesis Framework

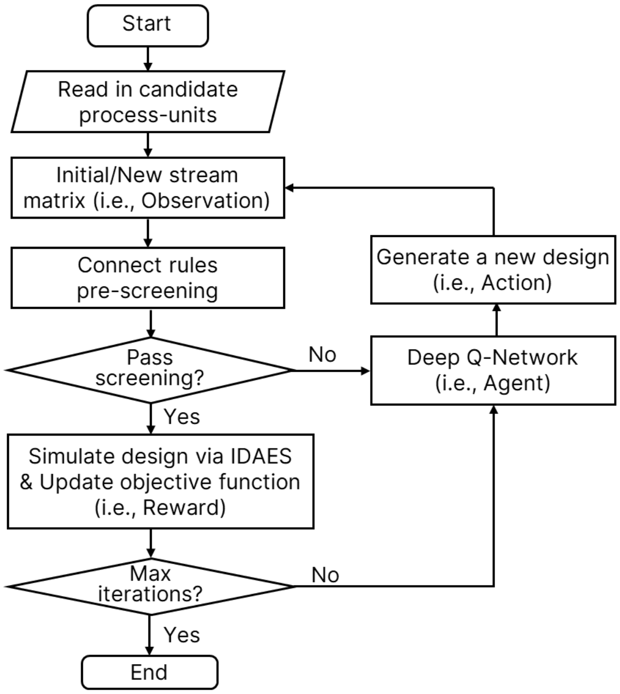

Figure 2. RL-driven process synthesis.

Wang et al. [2] developed a RL-driven process synthesis method as summarized in Fig. 2, based on which our previous work [7] enhanced the method with subspace search to more effectively address the combinatorial design space. For a brief review of this classical approach, RL is integrated with the IDAES platform [8] leveraging its extensive process modeling and simulation capabilities to enhance the design and optimization process. This approach begins with the definition of a pool of candidate process units, including reactors, heaters, and flash drums, with user-specified limits on the maximum number of units available. Unlike classic superstructure-based process synthesis approaches, RL-driven process synthesis eliminates the need for prior pre-postulations regarding flowsheet connectivity, allowing for a more thorough exploration of design spaces.

A critical step in this approach is the representation of process flowsheets as stream matrices, where binary variables encode the inlet and outlet relationships between unit operations. These matrices provide compact and efficient representations for the RL agent, enabling it to analyze and manipulate flowsheet structures. The update of these input-output stream matrices, as determined by the RL agent, synthesizes the new flowsheet structure. To ensure feasibility, generated flowsheets are pre-screened against connectivity rules, such as avoiding self-loops and ensuring proper sequencing of unit operations (e.g. a cooler does not directly connect to a heater).

The framework utilizes a DQN algorithm (as summarized in Section 2.1) to train an RL agent in generating optimal process designs. The agent takes stream matrices as inputs to NNs that are used to predict Q-values that correspond to actions such as adding, removing, or reconfiguring unit operations. The new flowsheet is then simulated and optimized using the IDAES platform, which determines the optimal operating variables using nonlinear optimization. The resulting rewards are used to update the parameters of the evaluation network, enabling continuous improvement over successive iterations.

3. QUANTUM RL-DRIVEN PROCESS SYNTHESIS

In this section, we introduce quantum machine learning, summarize the key techniques that are pertinent to this work, and present our novel methodology that leverages these for RL-driven process synthesis.

3.1 Quantum Machine Learning

QML is an exciting research field that explores the intersection of quantum computing and classical machine learning by implementing classical, quantum, or hybrid algorithms on quantum hardware which performs operations on qubits. The core of QML revolves around parameterized quantum circuits (PQCs), which are regarded as the quantum counterpart of NNs [9]. PQCs (Fig 3.) are quantum circuits that encode data as quantum states, have tunable parameters (e.g., rotation angles within quantum gates), and provide outputs through their observables. These are analogous to input features, weights and biases, and outputs in NNs where the tunable parameters are optimized with respect to a loss function (e.g., backpropagation). The observables of a circuit are representations of measurements taken from each qubit at the end of the circuit. Common expressions for these relate the Pauli sums or Pauli products of qubits in their final measured states.

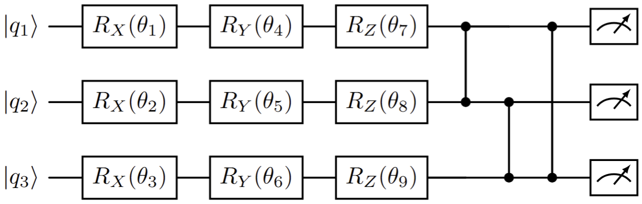

Figure 3. Parameterized quantum circuit.

Fig. 3 depicts a three qubit PQC with each being operated on by R_X, R_Y, and R_Z rotation gates followed by a daisy chaining of CZ gates. The rotation gates perform a manipulation in their respective x, y, or z axis by an angle of rotation specified by θ_i. CZ gates flip the phase of a qubit depending on the state of another qubit thereby imparting entanglement. PQCs similarly serve as universal function approximators and can approximate more efficiently and accurately than their classical counterparts [5]. A common way to increase the expressiveness or efficiency of these circuits is to re-encode data throughout the circuit. This idea is quite similar to the use of skip connections in residual NNs (ResNets) except instead of directly injecting the data into a layer along with that layer's inputs, the data is used in a single qubit gate and can feature its own tunable multiplier (Fig. 4) behavior more like a weight. These circuits are referred to as re-uploading PQCs.

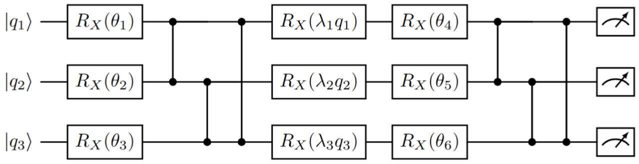

Figure 4. Re-uploading parameterized quantum circuit.

Fig. 4 illustrates a re-uploading PQC with three qubits. Each with an R_X gate and then a series of CZ gate operations before reaching another set of R_X gates where the angles are instead a function of the input data rather than a tunable parameter. Here, the angle of rotation is specified by $\lambda_i q_j$ where λ_i is a tunable multiplier and q_j is the input data encoded into the jth qubit. In addition to data re-uploading, tunable output weights can be used to increase performance and may be required if the PQC output range cannot sufficiently represent the desired output range (e.g. unconstrained continuous spaces).

3.3 Quantum-Enhanced Process Synthesis

Consistent with existing research efforts in quantum reinforcement learning [10], we remove the NNs in our method and incorporate PQCs in their place. This is highlighted in Fig. 1 where the transition is conceptualized. The same underpinning concepts of DQN algorithm are employed with the only distinction being that PQCs are used for the Q-function approximators instead of NNs.

The optimization problem (Eq. 2) is still performed but the vector θ corresponds to all tunable parameters for the evaluation circuit. In general, this adjustment to the algorithm requires special attention to data encoding and Q-value mapping. For data encoding, it is critical that input data be transformed or scaled in a way that it can be represented by a quantum state. In the context of flowsheet matrices, the binary values can easily be mapped to a quantum state. The observables (circuit outputs) must correspond to an action in the action space which greatly relies on the current problem. In our algorithm, we formulate a unique mapping from one observable to a corresponding action.

Our quantum-enhanced framework broadly follows Fig. 2 by integrating QML into RL-driven process design, replacing classical NNs with PQCs within the DQN architecture. The RL agent similarly uses input-output stream matrices to observe flowsheet structures, and PQCs are trained to approximate Q-functions that implicitly relate to optimal flowsheet designs. The quantum-enhanced DQN retains algorithmic principles to prevent model divergence and biases. This approach is implemented using the IDAES-PSE software for flowsheet simulation and optimization, where operational cost and productivity serve as performance rewards for the RL agent. This approach takes advantage of the novel and promising abilities of quantum computing to outpace classical methods in machine learning and optimization while also utilizing the established principles of RL algorithms.

4. HYDRODEALKYLATION CASE STUDY

In this section, we demonstrate the effectiveness and potential advantages of our quantum-enhanced RL-driven process design framework through a hydrodealkylation process benchmarked by the classical RL-driven approach.

4.1 Process Description

In line with the prior work [2,7], this study aims to design a hydrodealkylation (HDA) process which uses hydrogen (H_2) and toluene ($C_6H_5CH_3$) to form benzene (C_6H_6) as shown in Eq. 3. This reaction occurs in the vapor phase and is assumed to occur without any side reactions for the simplification of this analysis.

$$\mathrm{H_2 + C_6H_5CH_3 \rightarrow H_2 + C_6H_6}, \Delta H = -1.08 \times 10^5 \frac{\mathrm{J}}{\mathrm{mol}} \quad (3)$$

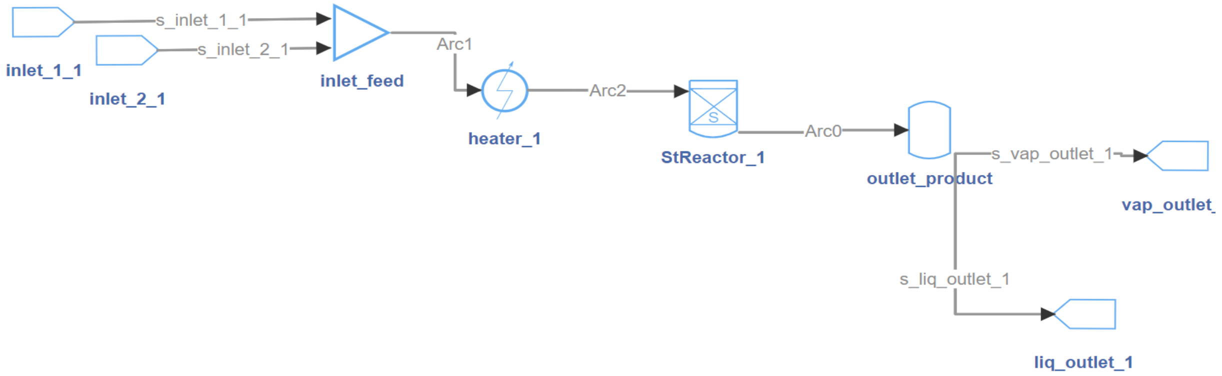

Figure 5: Optimal flow sheet design.

The feed streams, detailed in Table 1, include a vapor stream composed of hydrogen and methane, along with a liquid stream of toluene. The available process units are as follows: feed (combined with a mixer), product (integrated with flash operation), heater, cooler, stoichiometric reactor, mixer, flash, splitter, and compressor. The operating constraints of each unit are specified, such as maintaining an outlet temperature between 500–600 K. The product specification requires achieving a vapor product with a benzene purity exceeding 0.55. Mixture separations are assumed to follow ideal vapor-liquid equilibrium behavior. The design objective is to identify processes that optimize the vapor product's flowrate.

Table 1: Feed conditions for inlet process streams.

	Liquid Feed	**Vapor Feed**
Temperature (K)	303.2	303.2
Pressure (kPa)	350	350
Flowrate (mol/s)		
Hydrogen	0.30	0.00
Toluene	0.00	0.30
Methane	0.02	0.00
Benzene	0.00	0.00

For this study, the available unit ops to be selected are a heater and a reactor. This provides the minimum necessary conditions to reach the optimal flowsheet design. The selection of more units leads the quantum algorithm to become computationally intractable when being simulated on a classical machine. It is of ongoing work to investigate more capable simulators or move to real-world quantum hardware to provide additional studies.

4.2 Performance Comparison

Both the classical and the quantum-enhanced method were able to converge to the optimal flow sheet design in very few iterations. As given in Fig. 5, this flowsheet consists of a heater with an outlet temperature of 536.9 K, followed by a reactor with an outlet temperature of 897.8 K and a flash unit that generates process effluent at 705.7 K and approximately 1 atm. The operational variables of interest are a benzene purity of 0.75, a benzene flowrate of 0.225 mol/s, and an annual revenue of \$7.65×10^5 with the cost data provided by the IDAES simulation platform. This solution is generated on the 63rd epoch for both methods.

Plots of instantaneous reward for each epoch are shown in Figs. 6-7 for the classical and quantum approaches respectively. A large reward (~1500) implies that a suitable stream matrix has been found. This occurs 9 times for the classical approach whereas the quantum-enhanced algorithm discovers 13. The reward plots are nearly identical for around the first 800 epochs and then they start to diverge. This can be attributed to the epsilon-greedy policy from the DQN algorithm prioritizing random exploration in the initial stages. Both methods are performed using the same random generator seeds, so they explore the same scenarios at the same rate.

Not only is the PQC able to provide more solutions over the given amount of training steps, but it does so with less parameters. The PQC has a total of 88 trainable parameters whereas the NN has 1,092. The simulation time for the classical and quantum techniques were 0.58 and 9.06 minutes respectively. These results are summarized in Table 2. The calculation time for the quantum-enhanced approach has the potential to be greatly reduced when moving to real quantum machines and it is important to note that there is little correlation between simulated and true quantum hardware time. Regardless of current computational limitations, the quantum-enhanced framework shows promise in more efficiently searching the design space as compared the prior classical approach.

Table 2: Algorithm and model performance indicators.

	Classical	**Quantum**
Max Reward Solutions	9	13
Tunable Parameters	1,092	88
Computational Time (min)	0.58	9.06

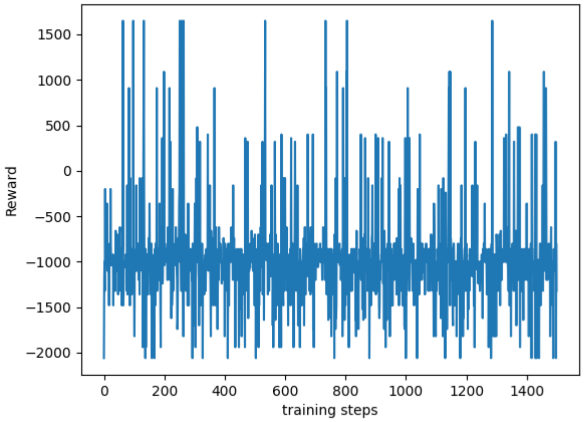

Figure 6. Classical RL agent reward for each episode.

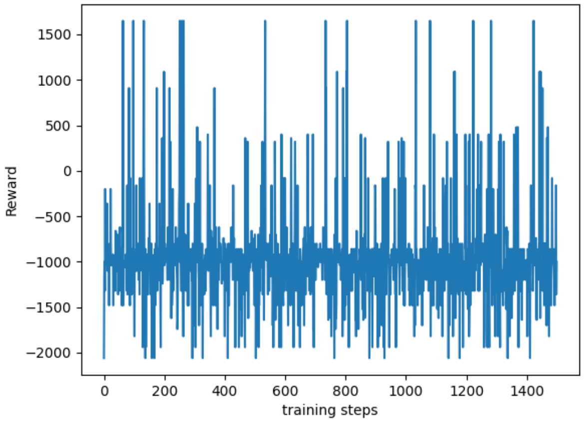

Figure 7. Quantum RL agent reward for each episode.

5. CONCLUDING REMARKS

This work presents a novel quantum-enhanced RL-driven process design framework, addressing challenges of large combinatorial design spaces. We demonstrate significant potential for improving efficiency and performance for RL algorithms in process synthesis by integrating QML techniques. Ongoing research focuses on expanding the range and scale of applications and scenarios by investigating better quantum simulators and exploring quantum hardware capabilities. Additionally, more efforts will be put into investigating how the hyperparameters of both PQCs and NNs affect the performance of these RL agents.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the financial support from NSF GRFP DGE-1102689, NSF EPSCoR RII Track-4 OIA-2327303, and West Virginia University.

REFERENCES

1. Stops, L., Leenhouts, R., Gao, Q., & Schweidtmann, A. M. (2023). Flowsheet generation through hierarchical reinforcement learning and graph neural networks. AIChE Journal, 69(1), e17938.
2. Wang, D., Bao, J., Zamarripa-Perez, M. A., Paul, B., Chen, Y., Gao, P., ... & Xu, Z. (2023). A coupled reinforcement learning and IDAES process modeling framework for automated conceptual design of energy and chemical systems. Energy Advances, 2(10), 1735-1751.
3. Bernal, D. E., Ajagekar, A., Harwood, S. M., Stober, S. T., Trenev, D., & You, F. (2022). Perspectives of quantum computing for chemical engineering. AIChE Journal, 68(6), e17651.
4. Ajagekar, A., & You, F. (2022). New frontiers of quantum computing in chemical engineering. Korean Journal of Chemical Engineering, 39(4), 811–820.
5. Yu, Z., Chen, Q., Jiao, Y., Li, Y., Lu, X., Wang, X., & Yang, J. Z. (2024). *Non-asymptotic Approximation Error Bounds of Parameterized Quantum Circuits* (arXiv:2310.07528).
6. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.
7. Tian, Y., Akintola, A., Jiang, Y., Wang, D., Bao, J., Zamarripa, M. A., Paul, B., Chen, Y., Gao, P., Noring, A., Iyengar, A., Liu, A., Marina, O., Koeppel, B., & Xu, Z. (2024). Reinforcement Learning-Driven Process Design: A Hydrodealkylation Example. 387–393.
8. Lee, A., Ghouse, J. H., Eslick, J. C., Laird, C. D., Siirola, J. D., Zamarripa, M. A., ... & Miller, D. C. (2021). The IDAES process modeling framework and model library - Flexibility for process simulation and optimization. Journal of advanced manufacturing and processing, 3(3), e10095
9. Jerbi, S., Gyurik, C., Marshall, S. C., Briegel, H. J., & Dunjko, V. (2021). Parametrized quantum policies for reinforcement learning (arXiv:2103.05577).
10. Skolik, A., Jerbi, S., & Dunjko, V. (2022). Quantum agents in the Gym: A variational quantum algorithm for deep Q-learning. Quantum, 6, 720.