Systems & Control Transactions

Original Research Article
Peer Reviewed Conference Proceeding
ESCAPE 36 - European Symposium on Computer Aided Process Engineering
Sheffield, UK. 21-24 June 2026

PSE PRESS

Process Flowsheet Synthesis via Quantum Reinforcement Learning with Improved Scalability

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 present quantum reinforcement learning algorithms for process flowsheet synthesis. Particularly, we discuss the implementation of encoding strategies to improve the algorithmic scalability. Reinforcement learning (RL)-driven flowsheet synthesis techniques provide a promising approach for conceptual process design, in addition to traditional optimization-based methods. These RL-based strategies identify the optimal flowsheet configurations from a maximum set of available processing units, without requiring to pre-postulate an interconnected superstructure. However, the resulting combinatorial design space for RL can scale extensively with the increased number of available processing units, which can render the algorithms to be computationally intensive or even intractable. To address this challenge, our prior work has introduced a quantum-enhanced approach to RL-driven process synthesis. However, this algorithm was limited in its capacity to solve larger flowsheeting problems as a large number of qubits were needed. To reduce qubit requirements and improve scalability, this work presents two different encoding strategies to embed information into the gates of quantum circuits instead of embedding it into qubits alone. We demonstrate the efficacy of these algorithms on three different scenarios of a flowsheet synthesis problem and interpret the results obtained from the IonQ quantum computing platform.

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

1. INTRODUCTION

There has been increasing interest in RL-driven process synthesis methods [1-3], which strive to determine the optimal design solutions while minimizing the impact of prior expert knowledge and the solution of large-scale combinatorial optimization algorithms (e.g., generalized disjunctive programming, mixed integer nonlinear programming). However, a significant challenge for these strategies is the large process design space which can be highly computationally intensive or even intractable for RL to identify the truly optimal design solution.

RL-driven process synthesis involves training an agent to build optimal flowsheet designs for a given process application. This is posed as a discrete decision-making process. The RL agent is usually represented via a neural network (NN). This parameterization allows a deep RL algorithm to adjust tunable parameters within the NN-based agent. In this way, the agent iteratively becomes better at developing process flowsheets. Several techniques have been developed by the research community in recent years [1]. These include graph-based [2] or matrix-based [3] flowsheet representations, leveraging transfer learning models instead of expensive simulators [4], and simultaneously addressing design and control decisions [5].

To potentially speed up the RL-driven process synthesis, our prior work [4] presented the use of Quantum computing to leverage the theoretical advantages and continuous improvements of quantum hardware [5-6]. However, this quantum-accelerated methodology was constrained to very small flowsheet synthesis problems due to the poor scaling relationship between the required number of qubits and the problem size (e.g., the number of unit operations considered for process design). In this work, we improve this methodology by applying new

information encoding strategies to reduce the number of qubits needed in the quantum algorithm. This allows larger process design problems to be solved.

The remaining sections are organized as follows: Section 2 provides an overview of RL-driven process synthesis and covers the fundamentals of quantum RL. Section 3 introduces the re-uploading parameterized quantum circuit (PQC) approach used in our prior work and the proposed two new encoding strategies for qubit reduction. Section 4 showcases the new algorithms performance in solving three different scenarios of a process flowsheet synthesis problem. Section 5 discusses the outlook for next-step implementation on current-scale quantum hardware. Section 6 gives the concluding remarks and directions of ongoing research.

2. BACKGROUND

In this section, we review the basics of: (i) the RL-driven process synthesis approach adopted in this work, and (ii) quantum machine learning (QML).

2.1 RL-driven Process Synthesis

In this work, we adopt the matrix-based method developed by Wang et al. [3], which was also used in our prior works [4, 7].Fig. 1 provides a conceptual overview of this approach [3]. The flowsheet is represented by a stream matrix which represents the interconnectivity between potential unit operations using binary (0-1) values. It is important to note that not all available unit operations must be used. This stream matrix representation is then used as the input to an NN-based agent. The agent serves as a Q-function approximator by assigning a Q-value to each potential action. In the context of process synthesis, the action involves adding, changing, or removing a connecting stream within a process flowsheet. By taking the action that corresponds to the highest Q-value, the agent has made the theoretically optimal decision.

In this algorithm, the agent is limited to making only one action per step (i.e., only one stream connection can be changed). The RL agent can thus determine all the mixed-integer variables through iterations with the environment, which define the flowsheet structure. Afterwards, the resulting flowsheet is simulated using the Institute for the Design of Advanced Energy Systems (IDAES) platform and the continuous process variables are optimized using nonlinear optimization solvers (e.g., IPOPT) [8-9].

This nonlinear optimization is conducted by maximizing a reward or objective function defined by the user. Examples of the reward functions include maximizing economic profits, productivity, sustainability, etc. This function is also used within the RL algorithm to update the tunable parameters of the agent. At each step of the algorithm, a gradient-based constraint-free optimization technique is used to improve the agents decision-making capability. The RL method used here is the Deep Q-Network algorithm.

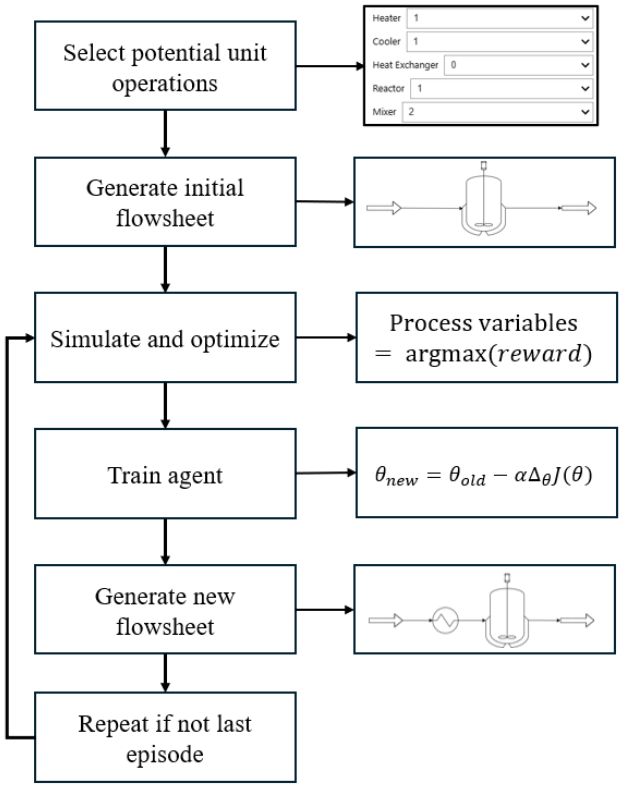

Figure 1. An overview of RL-driven process synthesis.

2.2 Quantum Reinforcement Learning

Quantum RL relies on QML which implements ML algorithms on quantum computing platforms and manipulates information encoded in qubits. The fundamental component of QML is the use of parameterized quantum circuits (PQCs), which can be described as the quantum analog of classical NNs [10].

PQCs consist of quantum circuits that take in classical or quantum data typically as quantum states and produce outputs via measurement combinations known as observables. The quantum gates have adjustable parameters that are trainable. Figure 2 illustrates an example of PQC with three qubits. Each qubit is sequentially acted upon by single-qubit rotation gates R_X, R_Y, and R_Z, followed by an entangling architecture composed of controlled-Z (CZ) gates. The rotation gates implement transformations about the x, y, and z axes of the Bloch sphere representation of the qubit's quantum state. The rotation angles are defined by the trainable parameters θ_i. The CZ gates introduce conditional phase shifts that depend on the state of another qubit, thereby generating quantum entanglement among the qubits. In this work, we will refer to a sequence of x, y, z rotation gates followed by CZ

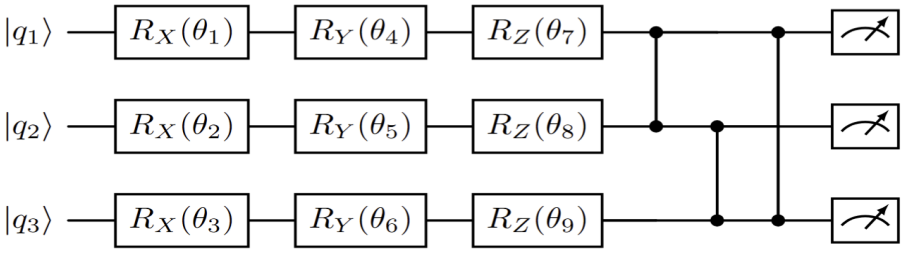

Figure 2. PQC with measurments: a 3-qubit example.

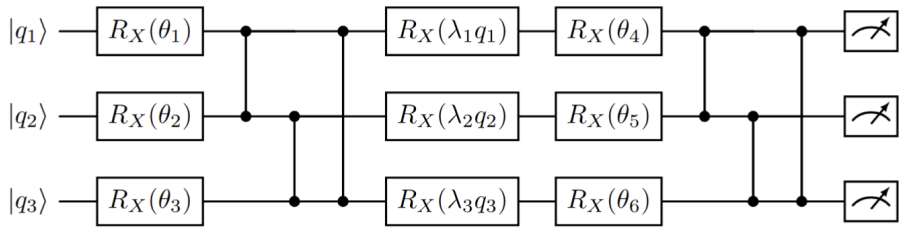

Figure 3. 3-qubit re-uploading PQC with measurements.

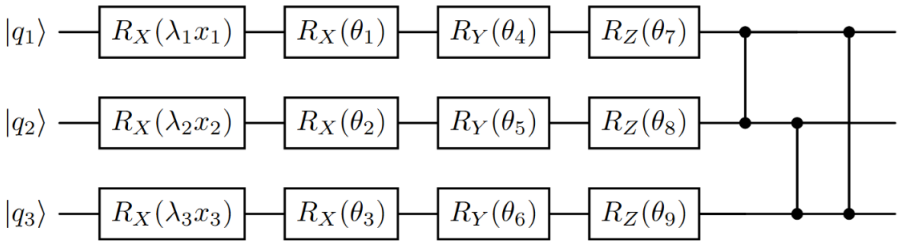

Figure 4. 3-qubit PQC using "X-only" encoding: an example of one PQC layer.

gate entanglement as a "layer" of a PQC.

For quantum RL, these PQCs can replace NNs in deep RL algorithms, as that developed in our prior work [4]. These would act as quantum agents to be trained. In addition, different with the data fed to NNs, the data fed to PQCs may need to be encoded as a quantum state. All other components of the classical RL algorithms may remain the same for quantum RL. Essentially any modern RL algorithm (e.g., DQN, DDPG) can be modified to use quantum agents by replacing NNs with PQCs [10].

3. QUANTUM RL-DRIVEN SYTHESIS

In this section, we first provide a brief review of the re-uploading PQC algorithm for the quantum RL-driven process synthesis approach presented in our prior work. We then present two new encoding approaches for qubit reduction and scalability improvement.

3.1 Quantum RL with Re-Uploading PQC

Our prior work utilized a special type of PQC, called a re-uploading PQC as shown in Fig. 3 [4]. This type of PQC increases expressibility by reinjecting the input data into the circuit after sets of CZ gates. This reinjection is accompanied with its own set of tunable parameters, which is represented by λ in Fig. 3. The data re-encoding step can be repeated many times, thus forming a quantum circuit with several layers. Despite the advantage, re-uploading PQC does require that the number of qubits in the circuit equals the size of the input data. The many layers of entanglement may also be expensive to compute.

For the RL-driven process synthesis algorithm, the input dimension is determined by the number of available unit operations. This is not an issue for classical RL, since NNs can handle extremely large input dimensions. However, for PQCs the circuit complexity increases vastly with more qubits. This necessitates the need for new quantum-based approaches which can reduce the number of qubits needed and therefore solve larger process synthesis problems.

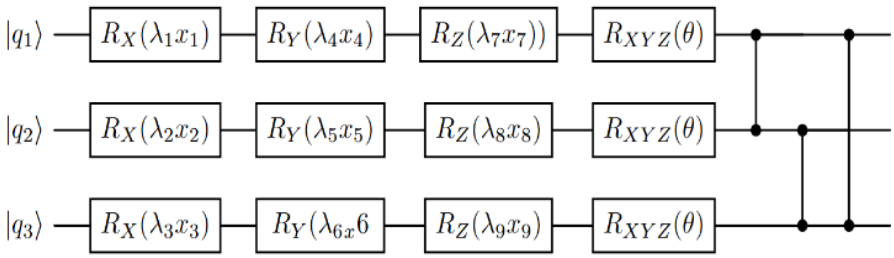

Figure 5. 3-qubit PQC using "XYZ" encoding: an example of one PQC layer.

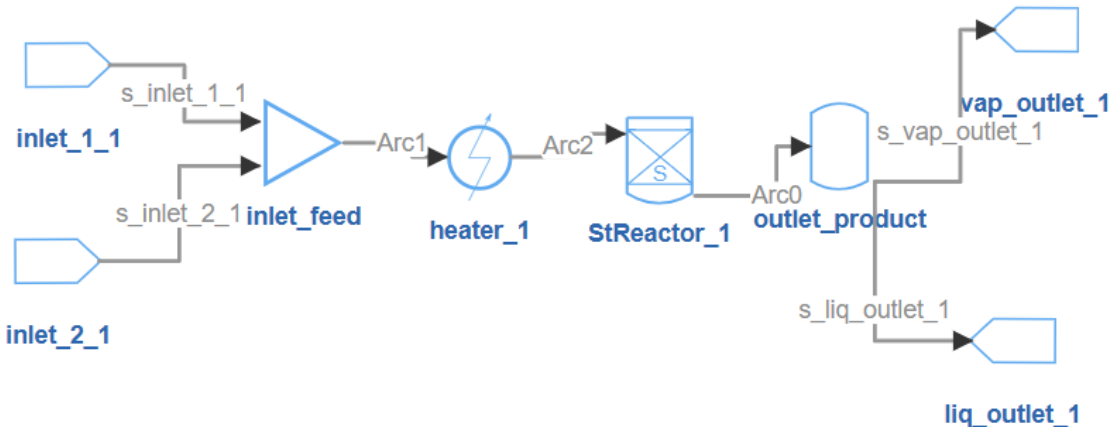

Figure 6. Optimal flowsheet corresponding to the maximum acheivable reward.

3.2 Improved Algorithm with Encoding

Motivated by the re-uploading PQC and other works in the field of QML [11-12], our proposed approaches below adopt new encoding strategies. These strategies take advantage of the fact that input data can be encoded into the angle of rotation gates. This differs with the previous algorithm which only encodes the input information to the qubits. In this way, larger problems can be solved using fewer qubits and less complex circuits.

3.2.1 X-only Encoding

In the "X-only" encoding strategy, the input features are injected directly into rotational gate angles with an additional trainable parameter λ used as a multiplier (Fig. 4). The input to each qubit of the circuit first enters an x rotation gate. It is then processed by a PQC layer, i.e. a sequence of x, y, z rotation gates followed by CZ gate entanglement. Since each qubit can only encode one extra input per layer, the number of qubits used in the circuit plus the number of layers needed should be equal to the size of the input. In other words, the number of qubits used or available determines how many layers should be added.

3.2.2 XYZ Encoding

The "XYZ" encoding strategy is similar to the "X-only" but adds additional y and z rotation gates before the qubit reaches the remaining parts of the PQC layer (Fig. 5). This allows more information to be encoded into each layer which can drastically save computational time. Again, the number of qubits used determines the number of layers needed since there is an encoding limit on each qubit per layer. This algorithm theoretically saturates the amount of information that can be embedded per layer using rotational gates, since x, y, and z are the only possible rotations.

4. CASE STUDIES

The section applied the improved encoding strategies on a case study, solving three process synthesis problem with increasing sizes to investigate algorithmic scalability. We also compare the performance of the quantum RL algorithm with classical RL.

4.1 Problem Description

Consistent with our prior works, this case study focuses on the flowsheet design of a hydrodealkylation (HDA) process which converts toluene ($C_6H_5CH_3$) into the product benzene (C_6H_6). This reaction is carried out in the presence of hydrogen (H_2) as described by Eq. 1. The reaction is solely active in the vapor phase and is

considered to occur without producing any byproducts.

$$H_2 + C_6H_5CH_3 \rightarrow H_2 + C_6H_6, \Delta H = -1.08 \times 10^5 \frac{\text{J}}{\text{mol}} \quad (1)$$

The HDA process feed conditions are summarized in Table 1 and consist of two inlet streams: (i) a liquid-phase stream with hydrogen and methane, and (ii) a vapor-phase stream composed entirely of toluene. A range of process units is available for flowsheet development, including a heater, cooler, heat exchanger, reactor, mixer, flash separation, splitter, compressor, expander, and a turbine. Each unit involves their own operational constraints, such as maintaining outlet temperatures in the range of 500-600 K. All phase separations within the process are assumed to obey ideal vapor-liquid equilibrium. The final product of the process must be a vapor-phase stream with a benzene mole fraction greater than 0.55.

In the following scenarios, the amount of available process units varies. The primary design goal is to maximize the vapor product stream flowrate and purity, forming a productivity-based reward function. The optimal flowsheet for this problem only uses a reactor and a heater with no other processing units, as shown in Fig. 6. The goal of each scenario is to build this optimal flowsheet given a set of potential process unit operations.

4.2 Scenario Analyses

4.2.1 Scenario 1

This scenario matches with the case study presented in our previous work [4], except the amount of training episodes has been extended from 1, 500 to 30, 000. The extended number of episodes is made possible by the newly proposed algorithms which only requires 4 qubits. The prior approach required 12 qubits. There are only two available unit operations: (i) reactor, and (ii) heater. This set represents the smallest possible subset of unit operations where the optimal flowsheet design can be found. The results of each algorithm for this scenario are provided in Table 2. Note that "classical" refers to the classical RL algorithm without quantum computing. "X-only" and "XYZ" refer to the quantum RL algorithms respectively with X-only encoding and XYZ encoding.

4.2.2 Scenario 2

In this scenario, there are three potential unit operations: (i) reactor, (ii) heat exchanger, and (iii) heater. Each algorithm is run for a total of 30, 000 episodes. In our previous algorithm with re-uploading PQC, this scenario would have required twenty qubits which is approaching the limit for simulation capability. The newly proposed encoding algorithms allow to solve this problem via quantum RL, only needing 5 qubits. The results of this scenario are highlighted in Table 3.

Table 1: Compositions and inlet conditions for feed 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

Table 2. Algorithm results for Scenario 1.

	Classical	X-only	XYZ
Max Reward Solutions	>100	>100	>100
Unique Flowsheets	3	3	3
Feasible Designs	1	1	1
Tunable Parameters	2, 374	64	40
Qubits Required	n/a	4	4
Number of Layers	2	3	1
Total Time (hr)	0.056	1.029	0.499

Table 3. Algorithm results for Scenario 2.

	Classical	X-only	XYZ
Max Reward Solutions	20	12	11
Unique Flowsheets	7	11	11
Feasible Designs	2	1	1
Tunable Parameters	2, 374	100	80
Qubits Required	n/a	5	5
Number of Layers	2	5	2
Total Time	0.0599	1.433	0.959

Table 4. Algorithm results for Scenario 3.

	Classical	X-only	XYZ
Max Reward Solutions	6	1	0
Unique Flowsheets	31	19	19
Feasible Designs	1	1	0
Tunable Parameters	2, 374	144	96
Qubits Required	n/a	6	6
Number of Layers	2	5	2
Total Time	0.086	3.040	2.315

4.2.3 Scenario 3

For this scenario, there are four potential unit operations: (i) reactor, (ii) cooler, (iii) heat exchanger, and (iv) heater. The number of episodes is extended to 60, 000, because no algorithm (including the classical RL algorithm) can find the optimal solution within 30, 000 episodes. For our previous algorithm with re-uploading PQC, this study would have required multiple simulations of a 30-qubit circuit which is not feasible. The results of this scenario are given in Table 4.

4.3 Results and Discussion

The ability of the quantum algorithms to use fewer parameters but achieve similar performance to classical algorithms is evident through the three scenarios. The quantum algorithms use an order of magnitude less tunable parameters. All three algorithms perform exceptionally well for Scenario 1 and find the optimal flowsheet over one-hundred times within the total training episodes (Table 1). In Scenario 2, an interesting result occurs in which the quantum algorithms propose more unique flowsheet designs than the classical algorithm, yet they find the maximum reward solution at only about half of the rate (Table 2). Additionally, the classical algorithm find two feasible designs while the quantum algorithms only find one each. For Scenario 3, the classical algorithm outperforms the quantum algorithms by discovering the optimal flowsheet six times compared to one and zero times for the quantum approaches, while also proposing significantly more unique flowsheets (Table 3).

Of the two newly proposed algorithms, the "X-only" encoding approach significantly outperforms the "XYZ" approach. This is particularly evident in Scenario 3 when the "X-only" method finds the optimal flowsheet while the "XYZ" method does not. This could be explained by the difference in trainable parameters. In these studies, the minimum number of layers are used. By design, the "X-only" encoding scheme must use more layers than "XYZ" encoding. It is possible that the superior performance is indicative on only the increased circuit complexity but not the different encoding schemes.

The classical RL algorithm appears to perform better than the quantum approaches especially when considering the computational time taken. However, there are two important aspects to consider: (i) the classical algorithm does use significantly more tunable parameters, and (ii) the computational time of the quantum algorithms are for simulated circuits run on a classical computer. Adding additional trainable parameters in the quantum circuits via increasing qubit or layer amounts may allow the quantum agents to become more expressive and therefore potentially as competitive as the classical agent. The computational time taken for the quantum approaches may be reduced when moving to quantum hardware.

5 CURRENT-SCALE QUANTUM HARDWARE OUTLOOK

The full end-to-end RL algorithm cannot currently run on quantum hardware due to prohibitive cost restrictions. It is technically possible, but the sampling costs of current-scale quantum circuits render this approach infeasible because of the large amount of total training episodes needed. However, it is still critical to understand whether the trained quantum agents could be realized via real-world quantum machines. To do this, we ran our trained circuits on IonQ's quantum computing simulator which mathematically represents their quantum hardware. These results are provided in Figs. 7 and 8 for the "X-only" algorithm in Scenario 1.

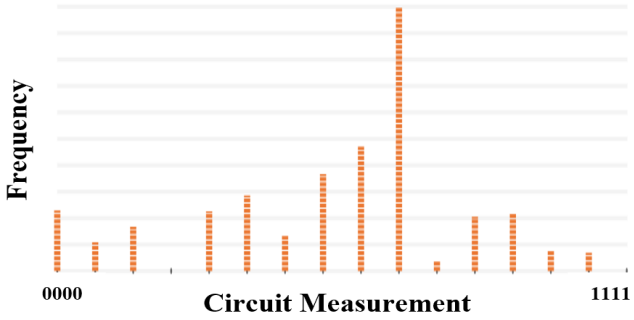

Figure 7. Scenario 1 X-only agent before training.

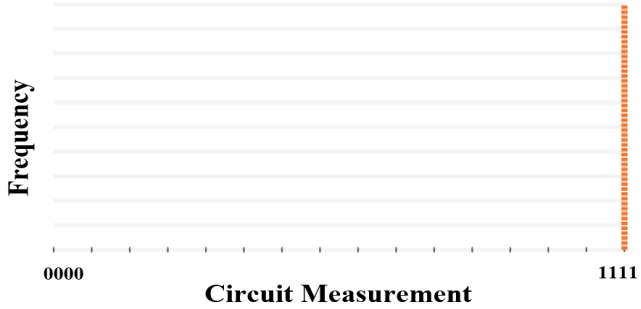

Figure 8. Scenario 1 X-only agent after training.

The results from the IonQ simulator give a probability distribution over the expected circuit measurement. These results are for the same random flowsheet before and after training. Before training the agent is random in its decision-making on which action to take while after training the probability distribution collapses to a single measurement. This shows that the agent's decision-making becomes deterministic in selecting the optimal action.

6 CONCLUDING REMARKS

In this work we presented two new quantum-accelerated RL-driven process synthesis algorithms. These algorithms were introduced to overcome a limitation of our previous approach which scaled poorly with problem size. The algorithms were demonstrated through three different scenarios of varying problem sizes. It was shown that the quantum algorithms can solve these problems while needing fewer parameters than classical RL algorithms without quantum computing. Ongoing efforts will include trying to reduce the algorithm complexity further to shorten the computational time needed on both simulated and real-world quantum hardware.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the financial

support from NSF GRFP DGE-1102689, NSF EPSCoR RII Track-4 OIA-2327303, and West Virginia University.

AUTHOR IDENTIFIERS

Author ORCIDs:
Braniff A: 0009-0006-6835-320X
You F: 0000-0001-9609-4299
Tian Y: 0000-0001-7937-5785

REFERENCES

1. Gao Q, Schweidtmann AM. Deep reinforcement learning for process design: review and perspective. Current Opinion in Chemical Engineering 44:101012 (2024). https://doi.org/10.1016/j.coche.2024.101012
2. Gao Q, Yang H, Theisen MF, Schweidtmann AM. Accelerating process synthesis with reinforcement learning: transfer learning from multi-fidelity simulations and variational autoencoders. Computers & Chemical Engineering 201:109192 (2025). https://doi.org/10.1016/j.compchemeng.2025.109192
3. Wang D, Bao J, Zamarripa-Perez MA, Paul B, Chen Y, Gao P, Ma T, Noring AA, Iyengar AKS, Schwartz DT, Eggleton EE, He Q, Liu A, Marina OA, Koeppel B, Xu Z. A coupled reinforcement learning and IDAES process modeling framework for automated conceptual design of energy and chemical systems. Energy Adv. 2:1735-1751 (2023). https://doi.org/10.1039/d3ya00310h
4. Stops L, Leenhouts R, Gao Q, Schweidtmann AM. Flowsheet generation through hierarchical reinforcement learning and graph neural networks. AIChE Journal 69: (2022). https://doi.org/10.1002/aic.17938
5. Reynoso-Donzelli S, Ricardez-Sandoval LA. An integrated reinforcement learning framework for simultaneous generation, design, and control of chemical process flowsheets. Computers & Chemical Engineering 194:108988 (2025). https://doi.org/10.1016/j.compchemeng.2024.108988
6. Braniff A, You F, Tian Y. Enhanced reinforcement learning-driven process design via quantum machine learning. Systems and Control Transactions 4:1403-1408 (2025). https://doi.org/10.69997/sct.149501
7. Ajagekar A, You F. New frontiers of quantum computing in chemical engineering. Korean J. Chem. Eng. 39:811-820 (2022). https://doi.org/10.1007/s11814-021-1027-6
8. Bernal DE, Ajagekar A, Harwood SM, Stober ST, Trenev D, You F. Perspectives of quantum computing for chemical engineering. AIChE Journal 68: (2022). https://doi.org/10.1002/aic.17651
9. Tian Y, Akintola A, Jiang Y, Wang D, Bao J, Zamarripa MA, Paul B, Chen Y, Gao P, Noring A, Iyengar A, Liu A, Marina O, Koeppel B, Xu Z. Reinforcement learning-driven process design: a hydrodealkylation example. Systems and Control Transactions 3:387-393 (2024). https://doi.org/10.69997/sct.119603
10. Lee A, Ghouse JH, Eslick JC, Laird CD, Siirola JD, Zamarripa MA, Gunter D, Shinn JH, Dowling AW, Bhattacharyya D, Biegler LT, Burgard AP, Miller DC. The idaes process modeling framework and model library—flexibility for process simulation and optimization. J Adv Manuf & Process 3: (2021). https://doi.org/10.1002/amp2.10095
11. Biegler LT, Zavala VM. Large-scale nonlinear programming using IPOPT: an integrating framework for enterprise-wide dynamic optimization. Computers & Chemical Engineering 33:575-582 (2009). https://doi.org/10.1016/j.compchemeng.2008.08.006
12. Skolik A, Jerbi S, Dunjko V. Quantum agents in the gym: a variational quantum algorithm for deep q-learning. Quantum 6:720 (2022). https://doi.org/10.22331/q-2022-05-24-720
13. Jerbi, S., Gyurik, C., Marshall, S. C., Briegel, H. J., & Dunjko, V. (2021). Parametrized quantum policies for reinforcement learning (arXiv:2103.05577).
14. Chen, S. Y.-C., Yang, C.-H. H., Qi, J., Chen, P.-Y., Ma, X., & Goan, H.-S. (2019, June 30). *Variational Quantum Circuits for Deep Reinforcement Learning*. arXiv.Org.
15. Pérez-Salinas, A., Cervera-Lierta, A., Gil-Fuster, E., & Latorre, J. I. (2019, July 3). *Data re-uploading for a universal quantum classifier*. arXiv.Org.