12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

Enhancing process flowsheet design through masked hybrid Proximal Policy Optimization

Simone Reynoso-Donzelli, Luis A. Ricardez-Sandoval*

Department of Chemical Engineering, University of Waterloo, ON N2L 3G1, Canada

**Corresponding author e-mail:* laricard@uwaterloo.ca

Abstract: This work introduces a methodology to design and optimize chemical process flowsheets through a novel hybrid Proximal Policy Optimization (HPPO) agent. The novelty of this work lies in the use of masking, a technique never considered before in the design of process flowsheets capable of finding correlations between the composing unit-operations (UOs). The performance of this novel agent is tested using case studies, one of which employs ASPEN Plus. The results obtained from both cases demonstrate significant learning on the part of the agent, yielding outcomes in line with the problem's specifications.

Keywords: flowsheet, design, optimization, HPPO, masking, UOs, correlations

1. INTRODUCTION

Process flowsheets are a fundamental tool in chemical engineering, offering a graphical description of a chemical process, involving the arrangement of unit-operations (UOs), their underlying interconnections through streams, and the materials involved. This visual representation provides an understanding of the process and facilitates the identification of potential challenges and optimization opportunities. Typically, process flowsheet design follows a structured approach, often initiated by focusing on the process's core units followed by the incorporation of purification and separation processes and the addition of units for mass and energy integration. With the development of novel optimization tools, process flowsheet designs underwent significant improvements, enabling engineers to tackle increasingly complex design challenges. When optimal process flowsheet design is attempted, the optimization formulation often results in a Mixed-Integer Nonlinear Programming (MINLP) problem. In this context, discrete decisions often involve UO selection, their placement, and design specifications (e.g., the number of plates in a distillation column); in contrast, continuous decisions consider the specification of operational conditions, e.g., flowrates, temperatures, pressures, as well as equipment sizing, e.g., capacity of mixing tanks. Note that the solution of MINLPs involving optimal flowsheet design is particularly challenging due to the interplay of discrete and continuous variables, the inherent process nonlinearity, consideration of disjunctive decisions, etc. Specialized algorithms and significant computational resources are thus typically required, with no assurance of achieving an optimal solution.

Reinforcement Learning (RL) is a Machine Learning framework that specializes in training agents to make sequential decisions through their interaction with an environment. This framework is often used to optimize complex processes (Sutton and Barto, 2018). Significant advancements have been made in combining RL techniques with chemical process flowsheet design. Midgely (2020) tackled the challenge of non-azeotropic mixture distillation trains by employing a soft actor-critic agent in conjunction with the COCO simulation suite. Khan and Lapkin (2020) introduced a hierarchical RL strategy for process flowsheet design, allowing for improved agent-process interaction and achieving superior results compared to baseline methods. Stops et al. (2022) employed graph neural networks to model process flowsheets and combining them with a hybrid proximal policy optimizer (PPO) agent to generate process flowsheets for an esterification process. Furthermore, van Kalmthout et al. (2022) extended Midgely's work (Midgely, 2020) by adapting their framework to the widely used ASPEN Plus simulation suite, revealing challenges in the complexity of convergence of the distillation trains and the extended training times.

An unexplored area, which is tackled in this work, involves the use of masking, a technique positioned to improve decision-making and strengthen the learning process of the agent. Huang and Ontañón (2022) introduced and applied the concept of masking by employing a masked Proximal Policy Optimization (mPPO) algorithm, shedding light on its efficacy in enhancing RL. This innovative technique allows for the consideration of logically constrained actions, which are often present when performing process flowsheet design. The primary objective of this work is to create an agent capable of autonomously design and optimize process flowsheets through the utilization of deep RL techniques. The major contribution, distinguishing this work from previous studies, revolves around the deployment of a masked hybrid Proximal Policy Optimization (mHPPO) agent. The innovative use of masking not only enhances the efficiency of process optimization but also offers a unique advantage in terms of unraveling the intricate relationships between various UOs within the flowsheet design process. In this work, a RL agent has been trained to make decisions with the goal of performing optimal process flowsheet design. The agent is also tasked with adhering to general process design, operational, environmental and safety constraints while generating feasible designs.

2. PROBLEM STATEMENT

In this section, a general definition of the process flowsheet design problems that can be addressed with the methodology shown in the next section is presented. To illustrate the approach, the minimization of the costs of a sequential process flowsheet is considered. Note that this problem can be readily adapted to consider other types of sequential flowsheets; however, ramifications or the handling of two separate pathways is currently outside the scope of this work.

Problem ($P1$) shown below can be formulated as a MINLP problem, having as a primary goal the minimization of a function $f(\boldsymbol{x_d}, \boldsymbol{x_c})$ that may represent a combination of process economics, environmental requirements, sustainability incentives, etc.

$P1$

$$\min f(\boldsymbol{x_d}, \boldsymbol{x_c}) \qquad (1.a)$$

$$s.t. \qquad \boldsymbol{g}(\boldsymbol{x_d}, \boldsymbol{x_c}) \leq 0 \qquad (1.b)$$

$$\boldsymbol{x_d} \Rightarrow \boldsymbol{x_d} \qquad (1.c)$$

$$\boldsymbol{x_d} \in X_D \subset \{0,1\}^{n_d}, \boldsymbol{x_c} \in X_C \subset \mathbb{R}^{n_c}$$

Variables in this study can be classified into discrete and continuous variables. The discrete variables, $\boldsymbol{x_d}$, are binary decision variables belonging to a bounded set $X_D \subset \{0,1\}^{n_d}$, where n_d represents the number of discrete variables. These variables often specify the inclusion or exclusion of discrete components in the process flowsheet design problem, which in this context represent the UOs to be considered in the flowsheet. The continuous variables, $\boldsymbol{x_c}$, belong to a bounded set $X_C \subset \mathbb{R}^{n_c}$, where n_c represent the number of continuous variables. The process flowsheet design optimization problem is subject to a set of inequality constraints represented by $\boldsymbol{g}(\boldsymbol{x_d}, \boldsymbol{x_c})$. For process flowsheet design problems, these inequalities often involve process design goal requirements, e.g., reactant conversion and product purity, as well as specific constraints related to UO design and operational limitations, e.g., allowed operating pressure or temperature of a UO. Logical and disjunctive constraints that delineate the interdependencies among the UOs are also considered (1.c). Given that these are binary variables formulated in linear constraints, there is no need for any reformulation.

In addition to the specification of the optimization formulation, an educated initial guess is needed to solve $P1$. For the problem at hand this initial data comprises comprehensive information regarding the inlet stream, featuring essential parameters such as temperature, pressure, and the molar flow rates of reactants. After previously addressing the challenges associated with MINLP problems, the inclusion of logical constraints within the formulation introduces an augmented degree of complexity. These complexities, in turn, motivate the exploration of alternative approaches for addressing these problems, with one such approach being the utilization of RL.

3. SOLUTION FRAMEWORK

This section presents a RL approach to solve the flowsheet design problems described in the previous section, delving into the key elements of the implementation: environment and agent. For the former, it will delve deeper into the action space, observation, and reward mechanisms, whereas for the latter, the agent's structure will be elucidated. Both sections emphasize the novel masking technique introduced in this work, describing in detail the respective modifications required for the proper functioning of both the environment and the agent, and the advantages of the proposed framework.

3.1 Environment

The environment serves as the external context for RL agent interactions and represents the part of the algorithm where optimization problem is translated into RL terms to be understandable to the agent (Sutton and Barto, 2018). The action space is defined in the environment and contains all the discrete and continuous actions, i.e., $\mathcal{A} = [\boldsymbol{x_d} \in \mathbb{N}, \boldsymbol{x_c} \in \mathbb{R}]$. For the flowsheet design problem, the discrete actions encompass all the UOs specified in the problem's definition, which in contrast to the optimization formulation, each UO correspond to an integer value. The continuous actions contain all the normalized ranges of $\boldsymbol{x_c}$. Normalizing the values is essential as the RL agent's continuous section operates with a Beta distribution that ranges from 0 to 1, per continuous action. An episode is composed of steps (i) – a single interaction with the environment – and ranges from 0 up until i_{max}, which is a user-defined hyperparameter that determines the maximum number of steps in an episode. At each step, an action, a_i, is sampled from set $\mathcal{A}$ and applied to the environment, note that the continuous part of a_i is interpolated into its respective domains. For instance, the addition and definition of a UO in the flowsheet is equivalent to a step. The observation vector at step i (o_i) shown in (2) is generated by the environment and sent to the agent to produce an action.

$$\begin{gathered} o_i = [T_i, F_{i,s}, i] \\ i \in \{1, i_{max}\} \\ s \in \{reactants, products\} \end{gathered} \qquad (2)$$

Depending on the optimization problem's configuration, the structure of o_i may vary but common features are kept. The information gathered at each step is: a) process operating variables, e.g., temperature (T_i); b) a chemical components (reactants or products $\rightarrow s$) tracker, which could be molar flowrates or molar fractions ($F_{i,s}$); iii) the step of the current observation (i), to add temporal context to the vector and differentiate identical action responses. All the elements in o_i are normalized, thus preventing larger values from having a greater influence on data processing within the agent's neural network. The step reward, r_i, used in this problem is decomposed into m-individual sub-rewards, i.e., $r_i = \sum_m r_m$. To apply this approach, $P1$ is broken down into its constituent objectives and constraints, assigning a separate reward to each of these new elements, as illustrated in Table 1. The constraints dealing with the interdependencies between discrete variables are not considered in distinct sub-rewards, primarily because their influence is embedded within the other functions. On the other hand, an additional sub-reward is considered, referred to as driving force. This term is defined as the difference in one distinctive parameter (Δ), e.g., conversion, molar flowrate. The driving force encourages the agent to convert the reactant into product and avoid remaining in an idle state. To motivate the agent to complete the process in as few steps (i) as possible, an additional sub-reward is introduced. This sub-reward adds the unused steps ($i_{max} - i$) to r_i, ultimately promoting the generation of shorter and more efficient process flowsheets.

Table 1. Reward shaping

Optimization problem	*RL reward function term*
Minimization of $f(\boldsymbol{x_d}, \boldsymbol{x_c})$ (1.a)	The penalty decreases as the values obtained from evaluating the function become less negative.
Process design goal requirement constraints (1.b)	The penalty lowers as the constraint violation approaches the desired value.
Specific design constraints (1.b)	Constant reward provided for constraint satisfaction.
-	Varying reward depending on the magnitude of the driving force.
-	Varying reward depending on the remaining steps.

The total costs obtained from the objective function are set to negative values in order to consider them as a penalty. However, to avoid the agent choosing actions with the lowest penalty, the previously defined driving force is used to balance out the rewards, thus avoiding trivial solutions. Note that the process design goal requirement constraints become relevant only in the final step of the episode. If the agent fails to satisfy these constraints within the stipulated maximum number of steps, it will incur a penalty. Conversely, there are no penalties for successfully meeting these requirements. Specific design constraints, which are only activated for certain actions, offer an additional constant sub-reward to the agent when the constraint is not violated. Treating local constraints as penalties, where the agent is penalized for any violation (similar to a constraint programming approach) may lead to poor learning rates. This formulation hinders the agent's exploration, resulting in convergence toward suboptimal results or, in some cases, it may lead to divergence.

One distinctive feature of this environment lies in its masking feature. This function is subject to the state, the environment's configuration at a specific step (i) and has the capability to activate or deactivate discrete actions within the action space. This means that the action space changes frequently throughout the training process. For example, consider the dependency between variables $x_{d,1}$, $x_{d,2}$ and $x_{d,3}$. The variables $x_{d,2}$ and $x_{d,3}$ can only be chosen if $x_{d,1}$ has been previously selected, i.e.,

$$x_{d,1} \implies (x_{d,2} \wedge x_{d,3}) \tag{3.a}$$
$$x_{d,2} + x_{d,3} \le x_{d,1} \tag{3.b}$$
$$x_{d,1}, x_{d,2}, x_{d,3} \in \{0,1\}$$

In the case of the RL environment, instead of a reformulation, a vector of Boolean variables known as the masking vector is employed. Depending on the state, the masking vector changes values, effectively masking or unmasking the discrete actions. Following the previous example, the starting masking vector would be: $mv_i = [True, False, False]$, corresponding to an action space $\mathcal{A} = [x_{d,1}]$. Once the action $x_{d,1}$ is chosen, mv_i changes to $[False, True, True]$, making $x_{d,2}$ and $x_{d,3}$ available in the environment for the agent to select and changing the action space to $\mathcal{A} = [x_{d,2}, x_{d,3}]$.

3.2 Agent

The agent used in this work is an adaptation of the hybrid Proximal Policy Optimizer (HPPO) proposed by Fan et al. (2019), which incorporates the masking methodology introduced by Huang and Ontañón (2022). The agent's architecture shares most components with the HPPO, with the adjustment made to the policy network (actor) to incorporate the masking technique. In a policy-based method, discrete actions are drawn from a categorical probability distribution (discrete actions policy $\pi_{\theta d}$) based on the current state provided by the environment. In a masked framework, along with the observation, an additional masked vector is supplied to the agent by the environment, i.e.,

$$a_i = \pi_{\theta d}(\cdot | o_i, mv_i) \tag{4}$$

To set the probabilities associated with the undesired actions to zero, the masking function integrated in the agent sets the logit values ($\boldsymbol{l}$) - output values from the second-to-last layer of the neural network - to a large negative number, $\boldsymbol{l} \to -\infty$. These values are subsequently forwarded through the final layer of the actor network, where the *SoftMax* activation function is used to convert these large negative values into the probability domain by setting them to 0, i.e.,

$$SoftMax(\boldsymbol{l}) = \frac{e^{l_m}}{\sum_{j=1}^{K} e^{l_j}} \; \forall \, m \in n_d \text{ and } \boldsymbol{l} \in \mathbb{R}^{n_d} \tag{5}$$

$$\lim_{l \to -\infty} e^{l_{masked}} = 0 \tag{6}$$

The masking technique allows the agent to prioritize and concentrate on the most relevant actions during each step of the learning process. This not only accelerates the agent's learning but also helps to avoid illogical decisions, e.g., starting a flowsheet with a product purification column. Note that masking can only be applied to discrete actions. Hence, continuous actions cannot be masked, and as a result, the agent may persist in selecting certain continuous actions even when their associated discrete actions have been masked.

4. RESULTS AND DISCUSSION

The methodology presented in the previous section was tested using two case studies involving the design of process flowsheets.

4.1 Case study 1

The goal for this case study was to design a process flowsheet that transforms reactant A into product B up to a specified conversion while minimizing the process economics. The UOs involved in this case are as follows: a) *Mixer*: This UO's function is to mix material flows; it has two possible inlet streams and one outlet. The mixer does not consider enthalpy, temperature, or pressure changes during mixing. The mixer is one of the two essential components for performing recycling, i.e., if this UO is not present in the flowsheet, material recirculation cannot be carried out. b) *Reactor*: A Continuous

Stirred-Tank Reactor (CSTR) is used as the reactor type, it is assumed that the reactor is non-isothermal, having a constant flow of coolant; it is also assumed that the reactor operates at steady state and has one inlet and one outlet stream. The CSTR model parameters were taken from Schweiger et al. (1998). The decision variables for the RL agent are the reactor's diameter (D) and height (H). c) *Flash*: This unit aims to separate the A/B mixture, which is assumed to have a relative volatility (α) of 4.5. The equations governing the flash operation are a modification of the Rachford-Rice equations expressed in terms of α. The design variable for this unit is the vapor/feed fraction of the flash tank (q). The flash operation constitutes the second part of the recycling process; in this case, the entire vapor stream is recycled back to the mixer assuming that the stream is subsequently condensed to match the composition of the input flow to the mixer. This UO has one input and two outputs, the tops and bottoms, being the latter, the material stream used to further build the process flowsheet.

The environment for this case study closely follows the description provided in the previous section. The action space ($\mathcal{A}$) consists of $\boldsymbol{x_d} \in$ [Mixer,CSTR,Flash], and $\boldsymbol{x_c} \in \left[\mathbb{R}_D, \mathbb{R}_H, \mathbb{R}_q\right]$ representing the diameter, height and the vapor/feed fraction, respectively. The observation vector returned at each step contains information of the outlet stream of the selected UO: concentration of reactant A, temperature, total molar flowrate, step and achieved conversion, $o_i = [C_A, T, F, i, x_A]$. The reward function consists of four sub-rewards: a step driving-force, defined as the difference of conversion between two consecutive steps; UO capital costs; the process design goal requirement, which seeks to achieve a specific conversion; and the difference between i_{max} and i reward given for short and effective flowsheets. Note that all rewards were normalized to prevent an individual term from having a disproportionate impact, e.g., the capital operating cost of a UO (CC_{UO}). In this case, due to the lack of thermodynamic data for the compounds and the absence of intrinsic constraints on the unit operations, the cost of operation (OC_{UO}) and specific design constraints are not considered.

Regarding the masking, to prevent the agent from opting to select the trivial solution of placing a mixer followed by a flash, a condition to unmask the latter required not only the presence of a mixer in the flowsheet but also a change of concentration between flows in a UO. The latter implies the existence of at least one reactor between the mixer and the flash. Additionally, to avoid confusion for the agent concerning recycling, once the mixer was chosen, that operation was masked and unmasked again only after selecting the flash. If the agent needed to use another flash, a second mixer had to be chosen beforehand. The problem's hyperparameters were specified next. The goal for this problem was for the agent to discover an economically attractive process flowsheet that can achieve a conversion rate of 97.5% or higher while using reactors with diameters and heights within the range of 1.68 to 2.36 meters (5.5-7.75 ft.). The agent has an i_{max} of 10 steps (i), 10 reactors, and all the required mixers and flash tanks to solve the problem in the shortest episodes possible. The agent's architecture features a fully connected neural network divided into two parts. The first part consists of three layers, which encode the input neurons into two output neurons. The second part takes the output of the first part, bifurcates into discrete and continuous components, and decodes (with one hidden layer in between) each section into their respective action sizes. All the hidden layers consist of 64 neurons each and employ *Tanh* activation function. The agent was trained for 75,000 instances, with each episode's steps adding up to this total number of instances, culminating in a total training time of 16 minutes. The network was updated every 2048 instances with a discount factor of 0.99 and optimized using Adam's optimizer with a starting learning rate of 2.5e-4 which was adjusted throughout the training.

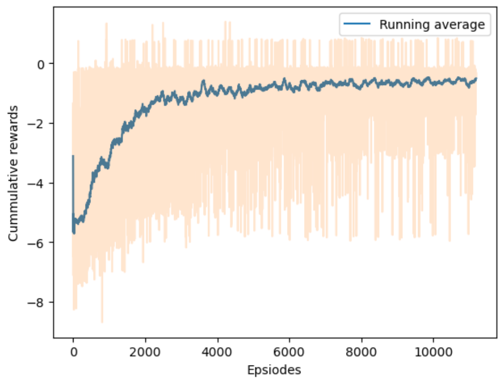


Figure 1. Learning curve based on the rewards case study 1

Figure 1 presents a summary of cumulative rewards at the end of each episode. The solid blue line represents the running average over the last one hundred episodes, while the shaded lines denote the rewards obtained at each step thus representing the variability in the reward function. The running average serves as a reliable indicator of the agent's learning progress, showing significant growth within the initial thousands of episodes, and reaching a plateau at around 7,500 episodes. Beyond this point, the agent struggles to find a better solution, or refines the already found configuration, oscillating around this maximum. From the graph, it is observed that the average cumulative reward obtained fluctuates around -0.310. However, when evaluating the best-performing agent, this value improves further, reaching -0.096.

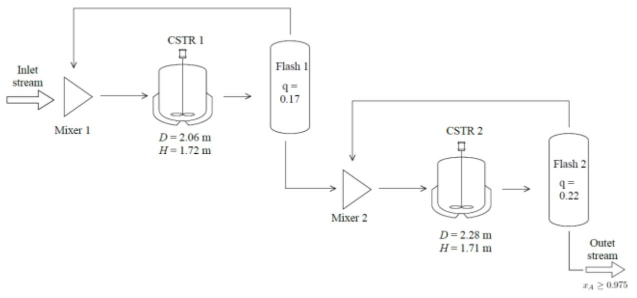


Figure 2. Best process flowsheet case study 1

The results shown in Figure 2 offer significant insights into the agent's decision-making process to find the best process flowsheet for this process. The agent could have reached the conversion objective by only placing reactors in series;

however, the identified flowsheet highlights the agent's proficiency in leveraging recycling, opting for flash tanks alongside reactors to not only reduce costs but also improve conversion rates. This is precisely the key, innovative feature introduced by the masking since it helps to identify these correlations between UOs, or alternatively prevents the agent from being lost in its learning process by making incorrect decisions, guiding the agent to make the correct decisions that lead to the specification of attractive flowsheets. The primary limitation of the hybrid agent lies in the necessity to adjust the learning processes of its discrete and continuous components, with the latter presenting the most pronounced challenges. Regarding the discrete component, the agent, constrained by a finite set of actions, can readily identify an optimal configuration. However, in the case of the continuous component, where an infinite number of decisions are possible, the agent is persistently engaged in the pursuit of improvement and exploration of new options, which significantly amplifies the variability of the overall process. This can be observed in Figure 1, where process variability decreases as the training progresses. Despite the pronounced variability, the running average tends to plateau towards the upper part of the cumulative rewards graph, however it does not attain the highest achievable rewards, indicating the existence of a potential better flowsheet. To demonstrate the masked agent's ability to discover the optimal solution, the continuous variables of the environment were discretized, and a discrete masked PPO was trained within the same environment. The results of this test are not shown here for brevity but it was confirmed that the agent found the expected configuration achieving the highest possible rewards, outputting a process flowsheet composed by one full recycle (mixer-reactor-flash) and an additional reactor at the end.

4.2 Case study 2

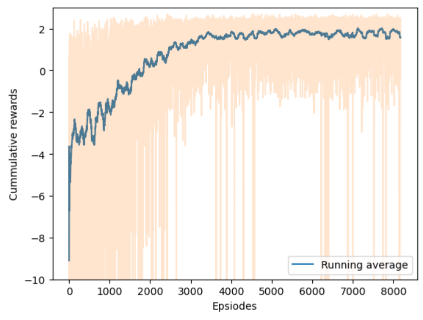
The objective of this case study was to develop a process flowsheet for the production of propylene glycol using water and propylene oxide, using methanol as an inert compound. In contrast to case study 1, this case study features the use of the ASPEN-Plus platform as part of the simulation environment for the propylene glycol production process. The key motivation behind the development of this hybrid platform lay in harnessing the thermodynamic packages and equations governing all declared UOs within the program. Notably, this approach facilitated the incorporation of a broader spectrum of UOs, resulting in processes that closely emulate real-world design procedures. For this case study, the selected thermodynamic package was the Non-Random Two-Liquid (NRTL) model, and the UOs were as follows: Mixer, Plug-Flow Reactor (RPLUG in ASPEN), Distillation column (DSTWU in ASPEN), Splitter and Heat exchanger (HEATER in ASPEN). Note that more rigorous models, such as RADFRAC and HeatX, could have been employed in the current framework to achieve more accurate results and address multicomponent or multiphase situations. However, these models were not explored in this study to simplify the analysis.

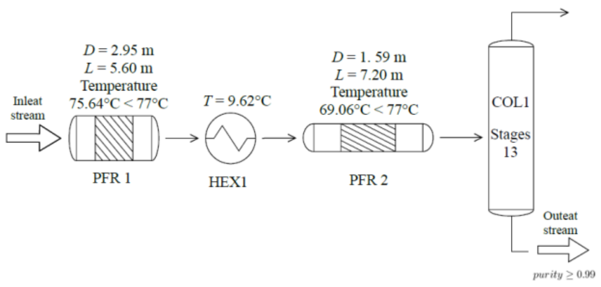
The action space ($\mathcal{A}$) for this case is as follows: $\boldsymbol{x_d} \in [M, PFR, COL, COL+S, HEX]$; note that, $COL+S$ is a DSTWU column coupled with a splitter; similarly, $\boldsymbol{x_c} \in \left[\mathbb{R}_D, \mathbb{R}_L, \mathbb{R}_{nst}, \mathbb{R}_{rr}, \mathbb{R}_{T_{HEX}}\right]$, i.e., reactor's diameter and length, column's number of stages, splitter's reflux ratio and heater's outlet temperature, respectively. The observation vector returned at each step (i) contains information of the outlet stream of the selected UO, i.e., temperature, molar fractions (χ) of water (H_2O), propylene oxide (PO), propylene glycol (PG), and methanol ($MeOH$), and the current iteration; thus, $o_i = \left[T, \chi_{H_2O}, \chi_{PO}, \chi_{PG}, \chi_{MeOH}, i\right]$. The reward function consists of the following terms, i.e.

$$r_i = -CAPEX_{UO} - OPEX_{UO} + \Delta x_A - (conv - x_A) \\ + cte_T + (i_{max} - i) \quad (7)$$

CC_{UO}, user defined function and OC_{UO}, term obtained from the heat requirements of each UO. A step driving force (Δx_A), a goal designed requirement, which seeks to achieve a certain purity of the product while maximizing the conversion of the reactant. Also, a constant reward (cte_T) set for a specific design constraint, which for this case is the operation's temperature under which the reactors should operate to avoid runaway reactions, 77°C, and the step difference reward, for short and effective flowsheets. The masking approach in this case closely resembles that of the previous case study. However, instead of the mixer-flash pair, now a mixer-column with recycle pair is considered. Additionally, a masking mechanism has been introduced for the simple distillation column, triggering its activation once a substantial amount of reactant has been converted into the product. This was implemented to prevent the agent from selecting short sequences of UOs that would result in reactant wastage, particularly given that this column lacks recycling capabilities. For the $COL+S$ a different approach is taken, the UO is unmasked once some reactant has been converted. In order to avoid reactant wastage in the distillate stream, the concentration of reactant in the purge (complementary stream to the recycle) needs to have a certain concentration to end the flowsheet; otherwise, it will be utilized to further extend the flowsheet. Similarly, masking has been applied to the heat exchangers (HEX), where once one is selected, another cannot be chosen consecutively. This restriction serves to deter the agent from placing HEX units in series without a specific purpose. Note that masking does not fix or predefine a chemical process flowsheet. Instead, masking is used solely to simplify the agent's learning process and, most importantly, to prevent simulation errors in ASPEN Plus, which could otherwise hinder the agent's learning.

The objective was for the agent to find the optimal process flowsheet to produce the highest amount of PG at a purity of 99% or higher. The agent had a maximum of 10 steps, 10 reactors and all the required heaters, distillation columns (with or without recycle) and mixers to solve the problem in the shortest episode possible. The agent's specifications were the same as those presented in case study 1, with the sole difference that the agent was trained for 50,000 instances and the network was optimized every 512 instances. Unlike the previous case, the agent's training time lasted approximately one day. This was due to the complexity of the environment, where for each instance, a flowsheet needed to be simulated in ASPEN Plus. The results presented for this case study are shown in Figures 3a and 3b. The plateau reached by the running average shown in Figure 3a closely aligns with the agent's maximum rewards achieved during training, suggesting that the configuration found by the agent is near optimal. The slight deviation from the maximum rewards represents the margin of error in the continuous component of the hybrid agent.





Figures 3a and 3b. Learning curve based on the rewards and best process flowsheet for case study 2

As shown in Figure 3b, the flowsheet found by the agent did not require recycling. This decision is reasonable because, unlike the previous case, the overall costs of a distillation column were much higher than those of a PFR. Thus, the agent chose to fully convert the reactant first, meeting the reactor design constraints, operating the reactors below 77°C, before proceeding with the separation process. A closer look at the flowsheet reveals that the agent aims to maximize reactant conversion in the initial stages. However, to avoid violating the specific design constraints, it acknowledges the need to employ medium sized reactors and incorporate a heat exchanger between the two PFRs. A distillation column is chosen at the end to obtain the desired product with a purity of 99% in the bottoms. Note that tests were conducted with the inclusion of a recycling column. However, the rewards obtained under those conditions were lower, e.g., the agent needed more than just a single column, the reactors were not able to meet the constraints, or the simulation failed. One notable aspect from the learning curve is the significant reduction in reward variability. Unlike the previous case study, once the agent found the optimal configuration, it focused on exploring the continuous part while exploiting the discrete component. To prevent flowsheets that had runs with warnings or errors in ASPEN, a penalty of -10 was assigned to the agent when faced with these issues. Note that as the agent learned, such flowsheets were mitigated, with only a few remaining in the later stages of learning. To improve the computational process, episodes were terminated as soon as an error or warning was encountered, and the entire flowsheet was reset.

5. CONCLUSIONS AND FUTURE WORK

This study presented a novel agent designed for the purpose of chemical process flowsheet optimization, with a specific focus on producing a policy capable of simultaneously minimizing both capacity and operational costs while adhering to the design constraints of the UOs and the overall process conditions. The key innovation in this work lies in the utilization of a masking method, a previously unexplored approach in chemical process flowsheet design. By accounting for the intricate interdependencies among UOs, this technique significantly enhances the agent's decision-making capabilities. Complex simulation environments were utilized during this work allowing the incorporation of diverse UOs into the optimization process, thereby augmenting both solution diversity and framework versatility. Future work will involve the implementation of this method in the design and optimization of flowsheets that are subject to uncertainty. Also, consideration of process dynamics while performing the optimal process flowsheet design is considered as future work.

REFERENCES

Fan, Z., Su, R., Zhang, W., & Yu, Y. (2019). Hybrid Actor-Critic Reinforcement Learning in Parameterized Action Space (Version 3). *arXiv*. https://doi.org/10.48550/ARXIV.1903.01344

Huang, S., & Ontañón, S. (2022). A Closer Look at Invalid Action Masking in Policy Gradient Algorithms. *The International FLAIRS Conference Proceedings* (Vol. 35). University of Florida George A Smathers Libraries. https://doi.org/10.32473/flairs.v35i.130584

Khan, A., & Lapkin, A. (2020). Searching for optimal process routes: A reinforcement learning approach. *Computers & Chemical Engineering* (Vol. 141, p. 107027). Elsevier BV. https://doi.org/10.1016/j.compchemeng.2020.107027

Midgley, L. I. (2020). Deep Reinforcement Learning for Process Synthesis (Version 1). *arXiv*. https://doi.org/10.48550/ARXIV.2009.13265

Schweiger, C.A., Floudas, C.A. (1998). Interaction of Design and Control: Optimization with Dynamic Models. *Optimal Control. Applied Optimization,* vol 15. Springer, Boston, MA. https://doi.org/10.1007/978-1-4757-6095-8_19

Stops, L., Leenhouts, R., Gao, Q., & Schweidtmann, A. M. (2022). Flowsheet generation through hierarchical reinforcement learning and graph neural networks. *AIChE* Journal (Vol. 69, Issue 1). Wiley. https://doi.org/10.1002/aic.1793

Sutton, R. S., Barto, A. G. (2018). *Reinforcement Learning: An Introduction.* The MIT Press.

van Kalmthout, S. C. P. A., Midgley, L. I., & Franke, M. B. (2022). Synthesis of separation processes with reinforcement learning (Version 1). *arXiv*. https://doi.org/10.48550/ARXIV.2211.04327