UNIVERSITY OF TWENTE
DELFT UNIVERSITY OF TECHNOLOGY

MASTER THESIS

Physics-informed reinforcement learning for process design.

Author:
E.M.T. (Ernst) UIJTHOF

Supervisors:
Prof. Dr. ing. M.B. FRANKE
Dr. ing. A. BANERJEE
Prof. Dr. A.M. SCHWEIDTMANN
Q. GAO

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Chemical Process Engineering

in the

Sustainable Process Technology research group of the faculty of Science and Technology

Friday 1st December, 2023

Enschede, the Netherlands.

"Actions such as his could come only from a robot, or from a very honorable and decent human being. But you see, you just can't differentiate between a robot and the very best of humans."

- Isaac ASIMOV [I, Robot]

Abstract

Physics-informed reinforcement learning for process design.

by E.M.T. (Ernst) UIJTHOF

Recently, reinforcement learning was used to generate sensible flowsheet designs for several applications, including distillation trains and other simplified reaction and separation problems. However, because of a lack of engineering knowledge, these learning agents are unable to discover general physics-related concepts and transfer learned information to new processes, requiring retraining prior to every unseen use-case. This study set out to improve the ability of an agent to learn underlying chemical and physical processes in a simplified distillation separation problem, increasing its applicability across an wider variety of use-cases.

To increase the generality of the model, two additions were made to an existing reinforcement learning infrastructure. First, the learning agent was trained on multiple feed-fraction use-cases instead of only on one, switching between feed-fractions at a set frequency. Second, information regarding the processes in the process units and chemical properties of the components in the streams was supplied to the agent in the form of the feed-composition and the boiling points of the components. Passing the engineering knowledge to the agent was achieved by concatenating the property values at the end of the flowsheet fingerprint that described the state of the flowsheet.

It was found that, for both implementations, the average model performance was worse than an established benchmark. Yet, on some occasions, the integrated models outperformed the benchmarks and generated flowsheets with a higher model scores. However, currently results are not consistent enough for this type of model to be of value in the flowsheet design process. As direct flowsheet fingerprint concatenation was deemed unreliable, future studies should focus on changing the way engineering knowledge is supplied to the reinforcement learning agent. Other information integration techniques such as reward shaping have shown promising results in other fields and could provide a solution here. When an adequate methodology for property integration is found, the efficiency of designing chemical processes could be increased greatly, thereby, contributing to the development of sustainable solutions in the field of chemical engineering.

Acknowledgements

First, I would like to express my sincere appreciation to the University of Twente and the Delft University of Technology for providing the necessary resources as well as a conducive environment for this research. Additionally, I am grateful to my supervisors Qinghe Gao, Meik Franke and Artur Schweidtmann for their valuable insights and support throughout this research journey. Their expertise and mentorship played a pivotal role in shaping the direction and quality of this thesis. I consider myself fortunate to have had such dedicated and supportive supervisors, and their influence will surely impact the rest of my academic and professional pursuits.

Furthermore, I am also deeply thankful for the encouragement and support of my friends and fellow students during the past few months. Their motivation was integral to overcoming challenges and achieving milestones during this project. Finally, I am also indebted to my family for being on my side throughout this academic journey. Your support, understanding, and encouragement have been instrumental in my academic journey during the past seven years.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	**A**rtificial **I**ntelligence
AAD	**A**verage **A**bsolute **D**eviation
CE	**C**hemical **E**ngineering
CEPCI	**C**hemical **E**ngineering **P**lant **C**ost **I**ndex
CNN	**C**onvolutional **N**eural **N**etworks
DSTWU	**D**i**ST**illation-**W**inn-**U**nderwood
FUGK	**F**enske-**U**nderwood-**G**illiland-**K**irkbride
GCN	**G**raph **C**onvolutional **N**etwork
GNN	**G**raphical **N**eural **N**etworks
HK	**H**eavy **K**ey
HNK	**H**eavy **N**on-**K**ey
IEA	**I**nternational **E**nergy **A**gency
IoT	**I**nternet **o**f **T**hings
LK	**L**ight **K**ey
LNK	**L**ight **N**on-**K**ey
MAD	**M**aximum **A**bsolute **D**eviation
MDP	**M**arkov **D**ecision **P**rocess
MLP	**M**ulti**L**ayer **P**erceptrons
ML	**M**achine **L**earning
NIST	**N**ational **I**nstitute of **S**tandards and **T**echnology
NK	**N**on-**K**ey
RL	**R**einforcement **L**earning

Chapter 1

Introduction

In the past decade, advancements in strategic thinking, decision-making, and natural language understanding across different applications have allowed Artificial Intelligence (AI) to outperform human players in areas where this was previously not possible [1], [2]. This transition has helped to catalyze innovative AI research in many fields [1]–[4]. The chemical industry is one of these areas, showing great potential for the implementation of information and computer technology as well as AI. The availability of large amounts of data relating to process and plant operation [5] has made data-driven optimization techniques relatively easy to implement in this field, resulting in the emergence of Industry 4.0 in 2011 [5]–[7]. Constrained by emission limitations, increasing energy prices and raw material scarcity, Industry 4.0 has revolutionized the way companies in the chemical industry develop their products [5]–[7]. In addition to an increase in the connectedness of chemical plants through the use of cloud computing and the Internet of Things (IoT) [8], AI is one of the main tools driving the Industry 4.0 movement, having seen several substantial breakthroughs in the past decade [5]–[7]. By collecting information connected to processes occurring in the physical world and employing algorithms and automation techniques, it has become possible to enable more informed decision making and translate decisions made without human interference from the digital world into the physical world. By this, the plant productivity can be maximized while risks associated with chemical product manufacturing are minimized [9]–[11]. The AI field that focuses on the development, training, and improvement of algorithms to recognize patterns and make predictions or decisions based on data, is called Machine Learning (ML) [12]–[15].

Reinforcement Learning (RL), a subcategory of ML is one of the primary areas of AI advancement in the chemical industry. Here, an agent learns how to interact with a complex dynamic environment based on experience [16]–[18]. The agent determines an optimal policy by observing the response of the environment when random or non-optimal actions are taken. Additionally, the agent learns the reward that is associated with any given state. As the agent learns the rules of the environment, it is able to take informed decisions in an effort to achieve the maximum future reward [16]–[18]. Until now, most RL usage in the chemical industry has been related to automated control, real-time optimization, and supply chain operation, as RL is most applicable in dynamic decision-making settings, where a series of decisions need to be made with the aim to minimize an overall loss function [16], [18]; a feature all three of the mentioned fields share [19]–[21].

Another area where RL has been applied effectively is in the optimization of catalyst structures as well as reaction- and operation conditions. Catalysis is a key process in many chemical reactions, so optimizing the catalyst can greatly increase reaction efficiency and reduce waste. Lan and An demonstrated that RL can be used to automatically design catalysts by iteratively suggesting changes to a catalyst's structure and testing their effects on reaction outcomes [22]. This allows for the identification of optimal catalyst structures in a more efficient and cost-effective manner than

traditional trial-and-error methods. Additionally, RL has been applied for the optimization of chemical reaction conditions. Chemical reactions are affected by a multitude of factors, including temperature, pressure, and reactant concentrations. In a similar manner to the catalysts structure optimization, RL can be used to suggest changes to these variables and test their effects on reaction outcomes to identify optimal conditions for the reaction, as shown by Zhou et al. [23]. This allows for more efficient and cost-effective process development. RL has also been used in the optimization of chemical plant operations. Chemical plants have many factors that can be adjusted to optimize production efficiency, such as flow rates, temperatures, and pressures. Adams et al. showed that RL can be used to learn the optimal set-points for these variables based on current plant conditions and production goals [24]. This can lead to increased plant efficiency, reduced waste, and improved product quality.

Another Chemical Engineering (CE) field that conventionally depends on dynamic decision making and has exhibited a considerable potential for the implementation of RL is process synthesis [25]–[28]. Currently, process designers primarily rely on mathematical optimization techniques, historical data, and engineering expertise [29]–[31] to determine suitable process unit configurations to achieve optimal values for process features, such as costs or environmental impact, while meeting specific constraints regarding product purity or safety regulations. Using RL for process synthesis has proven to allow designers to better understand the chemical and physical processes that occur within the system as well as reduce the time and costs associated with the design process itself [27], [32], [33]. Since, if a computer can learn to master a complex game with well-defined rules, shouldn't it also be able to generate process flowsheets that adhere to thermodynamic principles?

Recently, RL has shown several promising results in the field of process synthesis, where an RL agent can be trained to design (bio-) chemical processes through interaction with an environment, e.g., process simulation software [25]–[28], [32]–[34]. For example, Midgley demonstrated that a deep RL agent was able to produce sensible flowsheet designs of distillation trains for the separation of non-azeotropic mixtures. Here, a soft-actor-critic RL agent first decides whether to add a new distillation column to the intermediate flowsheet and, subsequently selects the operating parameters related to the column [25]. This combination of discrete and continuous decisions defines the hybrid action-space [25], [26]. As described by Stops et al., a similar architecture can be extended to also employ the use of Graph Neural Networks (GNNs) to convert process flowsheets to fingerprints [26]. This method has several advantages over previous works that represented the flowsheet structure using vectors and matrices, passing them through a Convolutional Neural Networks (CNNs) for the observational step [26]. Namely, a fingerprint containing information about the units (nodes) and the connections (edges) is much better suited for the representation of the complex branched connectivity of flowsheets than a matrix.

One of the main limitations of the current architecture is its lack of prior engineering knowledge, resulting in an inability to discover general physics-related concepts and transfer the gained information to new processes. This lack of prior knowledge makes it impossible to reuse the same learned policy on two separate use-cases, requiring the agent to relearn a new policy for every use-case [35], [36]. Additionally, the training process is often slow and prone to errors or failure to converge [36]. These limitations are caused by the agent's limited knowledge about the actual properties and driving forces inherent to the chemical processes it makes decisions based on [35], [37]. To prevent these limitations, this research aims to incorporate chemical composition and property knowledge into a previously developed RL agent for process synthesis by Stops et al. [26] via direct concatenation of the specific property onto the flowsheet fingerprint. This will enable the RL agent to learn general physics-related concepts that can be transferred to different process design tasks.

To demonstrate the advantages of the physics-informed RL for process synthesis, the extended RL agent is applied to multiple problems related to the separation of mixtures of paraffinic hydrocarbons found in petrochemical processes. The viability and reliability of the agent are determined based on two metrics: First, the overall model score of the agent with and without the added property will be observed. Second, the RL agent will be tested on multiple separation problems in succession to test the transferability of the gained knowledge. Thus, this research aims to determine to what degree integrating general physics-related concepts into an existing RL agent structure improves the ability to transfer learned policy information to new processes. Enabling the possibility of knowledge transfer, allowing for the use of the same learned policy for multiple use-cases, would greatly improve the viability of RL agents as a design tool in the field of process synthesis [35]–[38].

In the next section, a brief description of the primary functionalities of RL, as well as an overview of the current state of the agent, are provided. Following that, an overview outlining the general research methodology and the metrics that can be used to evaluate the performance of the RL agent is given. Subsequently, the results are summarized and assessed. Finally, by determining and evaluating the effectiveness and knowledge transferability of the agent prior to and after property integration, the effect of providing additional information on the performance of an RL agent is quantified.

Chapter 2

Theoretical Background

In the following section, first, a brief introduction is given into what distinguishes RL from the other learning algorithms employed within the field of ML. Then, the main theory behind RL is explained, prior to transitioning into an explanation of the specific architecture that was employed during this research.

2.1 Machine learning

The field of ML is traditionally divided into the supervised learning, semi-supervised learning, unsupervised learning and RL sub-divisions. This study focuses on applying an RL approach to a specific challenge. Yet, in order to comprehend the intricacies of RL as a ML methodology, it is essential to gain a fundamental understanding of the other three existing algorithms. Therefore, the subsequent paragraph provides a summary of the core principles underlying the other three methodologies.

In the domain of supervised learning, the primary objective is to deduce a function or relationship based on labeled training data [12], [39], [40]. The training dataset comprises input vectors X and their corresponding output vectors Y, where each label or tag in vector Y is linked to its respective input example in vector X. This pairing of input vectors and their labels provides the algorithm with sufficient information to learn and generalize patterns within the data [39]. The final goal is for the supervised learning model to find the underlying relationships between inputs and outputs, allowing it to make accurate predictions or classifications on new, unseen data.

For unsupervised learning, the idea remains to find hidden patterns in the input data [12], [39], [41]. However, in this case, the supervisors are non-existent and only unlabeled training data X is available. This changes the nature of the relationships that the model attempts to find; unsupervised learning models are mainly utilized for clustering and association analyses, employing the model's ability to group similar data points together and reveal inherent patterns or structures within datasets without explicitly knowing what it is looking at [12], [39], [41].

Semi-supervised learning is a mixture between the supervised and unsupervised learning cases [39], [40]. For this type of ML, the provided data consists of a mixture of both classified and unclassified information. In most cases, an insufficient amount of labeled data is available to develop a model based on purely a supervised learning approach. Therefore, the goal of this type of algorithm is to train a model that outperforms one generated solely with labeled data by incorporating both the labeled and unlabeled data. The objective is to enhance the model's predictive accuracy on future test data, leveraging the benefits of additional unlabeled information for a more robust and effective classification [39], [40].

2.1.1 Reinforcement learning

Where the previous algorithms are based on supplied (and sometimes labeled) information, RL is based on leveraging observations from interactions with an environment to make decisions that either maximize rewards or minimize risks. RL problems revolve around the task of learning what actions to take in different situations [40], [42]. These problems are inherently closed-loop in nature because the actions taken by the learning system influence its subsequent inputs. Unlike the other forms of ML where the system is explicitly told which actions to take, in RL, the learner must discover the most rewarding actions through trial-and-error. To develop intelligent programs, also known as agents, an optimal policy or strategy is found by observing the response of the environment when random or non-optimal actions are taken [18], [42].

2.1.2 Markov decision processes

RL uses the formal framework of Markov Decision Processes (MDP), as shown in the tuple in Equation 2.1 below, to define the interaction between a learning agent and its environment [18], [42]. In the most basic form, the learner and decision maker is called the agent. The party it interacts with, comprising everything outside the agent, is called the environment. These parties interact by the agent selecting actions and the environment responding to these actions and presenting new situations to the agent. The environment additionally gives rise to a reward, a numerical value that the agent seeks to maximize over time through its choice of actions. Based on the obtained reward, the agent alters its policy, changing the probability of choosing a certain action for the specific state it is given [18], [42]. Thus, a MDP is comprised of states $s \in S$, actions $a \in A$ and a transition model T that determines a new state based on the taken actions, according to $SxA \rightarrow S$, as well as a reward function R [42].

$$MDP = \{S, A, T, R\} \tag{2.1}$$

The agent and environment interact at each of a sequence of time steps, $t = 0, 1, 2, 3, ...$ At each time step t, the agent receives a representation of the environment's state S_t and based on its current policy selects an action A_t. The term "policy" refers to the strategy or set of rules that the agent employs to make decisions in different states of the environment. One time step later, as a consequence of the taken action and the state of the environment, the agent receives a numerical reward, R_{t+1}. Simultaneously, a new state S_{t+1} is determined by the transition model T based on how the taken action influenced the environment [16], [18], [42]. Then, this process is repeated giving rise to a sequence that begins like:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, ... \tag{2.2}$$

MDPs are a mathematically idealized form of the RL problem. Additionally, MDPs are an example of sequential decision making, where actions influence not just immediate rewards, but also subsequent states, and through those future rewards. This requires the agent to balance between optimizing immediate gains and considering the long-term consequences of its decisions. Therefore, two fundamental concepts are important in the decision-making process of the agent; exploration and exploitation. First, exploration refers to the agent trying out new or unknown actions to discover their effects and gather information about the environment. The second fundamental concept is exploitation, which revolves around choosing actions that the agent believes to be optimal based on its current knowledge. Both factors need to be balanced for the agent to achieve optimal performance in a specific environment, as overemphasis on exploration may lead to non-optimal short-term rewards, while overemphasis on exploitation may result in a failure to

discover better strategies or changes in the environment [42]. For a more in-depth exploration of MDP, the reader is referred to "Reinforcement Learning: An introduction" by Sutton and Barto [42].

2.1.3 Actor-critic reinforcement learning

In the present study, a slightly more complicated RL variant is used, where the agent is divided into separate actor and critic structures. RL methods of this type feature a separate memory structure that is used to represent the policy independently of the value function [39], [42]. The policy structure, known as the actor, is used for selection of the action, while the estimated value function, the critic, evaluates and critiques the actions taken by the actor [42]. After each action, the critic evaluates the new state to determine whether things have gone better or worse than expected. This helps the actor in refining the policy by learning more directly about the value of the actions it takes [42], [43]. In the previous architecture, the actor is responsible for selecting actions based on the current policy, but it does not directly estimate the value of states or actions. Instead, the actor focuses on the exploration and exploitation of the environment to maximize expected rewards. Actor-critic methods are known to exhibit more stable learning compared to other RL approaches and are especially well-suited for problems with continuous action spaces, where discrete methods might be less effective [44]. The actor-critic architecture is shown in Figure 2.1.

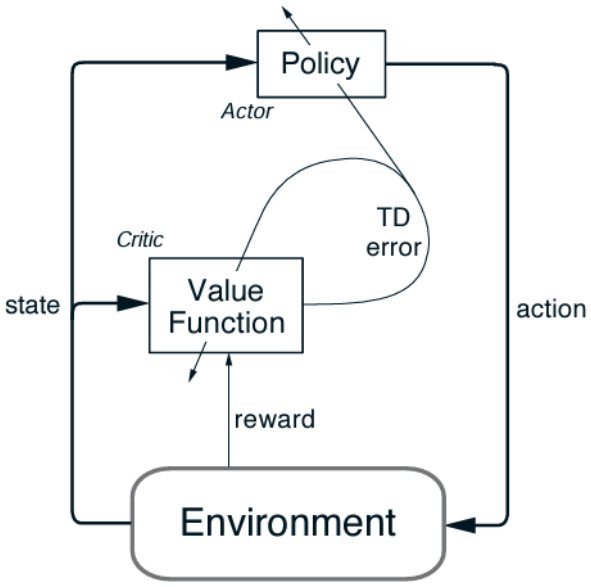

FIGURE 2.1: The actor-critic architecture [42]

2.2 Reinforcement learning for process synthesis

Recently, RL has shown several promising results in the field of process synthesis. For these use-cases, an RL agent can be trained to design chemical processes through interaction with an environment, such as process simulation software [25]–[28], [32]–[34]. In this scenario, the observed state corresponds to a representation of the chemical flowsheet, and the reward driving the learning of the agent could be associated with a financial value. Actions within this context might entail decisions such as placing a process unit with specified operating parameters [25], [26]. Midgley demonstrated that a deep RL agent was able to produce sensible flowsheet designs of distillation

trains for the separation of non-azeotropic mixtures [25]. Here, an actor-critic RL agent first decided whether to add a new distillation column to an intermediate flowsheet and, subsequently selected the operating parameters related to the column [25]. This combination of discrete and continuous decisions defines the hybrid action-space [25], [26].

As shown by Stops et al., a similar architecture can be extended to also employ the use of GNNs to convert process flowsheets into fingerprints [26]. This method has several advantages over the work of Midgley where the flowsheet structure was represented using vectors and matrices, passing them through a CNN for the observational step [25]. Namely, a fingerprint containing information about the units (nodes) and the connections (edges) is much better suited for the representation of the complex branched connectivity of flowsheets. Therefore, the work of Stops et al. [26] was used as the basis for this research. The subsequent sections explain the intricacies of this model.

2.2.1 State representation using graphical neural networks

As mentioned, the main characteristic of the method employed in this study as originally developed by Stops et al. [26] is the representation of states through directed flowsheet graphs. This feature enables the processing of states in GNNs as proposed by Schweidtmann et al. [45], allowing for the incorporation of topological information in the analysis. Figure 2.2 shows an example of the conversion of a flowsheet to a graph, where traditional units, feeds and products are represented as nodes and process streams are represented as the edges between them. The nodes store information about the type of unit operation and design variables, while the edges capture thermodynamic details of process streams, such as the temperature, molar flow, and molar fractions. Lastly, intermediate flowsheets also include "undefined" nodes resulting from the fact that when a new unit operation is added to the flowsheet, the open effluent streams are considered as "undefined". In subsequent steps, when the agent is allowed to place a new unit, these locations represent potential sites for incorporating new unit operations.

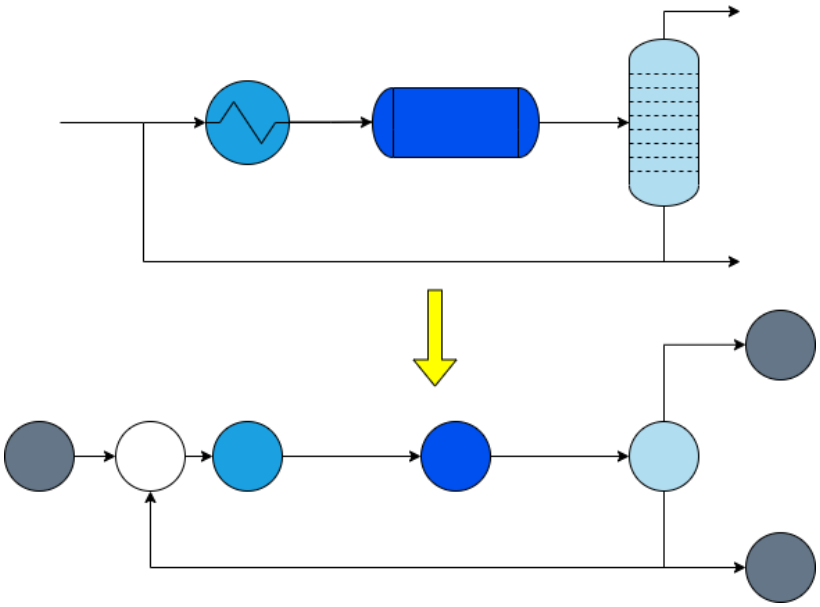

FIGURE 2.2: Flowsheet representation as a graph
(Adapted from Stops et al. [26])

2.2.2 Reinforcement learning agent

The agent is the primary component in the architecture shown in Figure 2.1. In the subsequent section, the specific agent architecture employed for this research as proposed by Stops et al. [26] is shown and discussed. As mentioned, the agent selects actions, and the environment responds by transitioning to a new state and providing feedback in the form of a reward. During this research, each action consisted of the following three sequential decisions:

1. Select an open location within the flowsheet based on the previously discussed graph flowsheet representation.
2. Add a new unit operation to the selected open location.
3. Determine a value for a continuous design variable applicable to the selected process unit in Step 2.

For this research, a simple distillation separation problem was considered, where an RL agent had the opportunity to place a distillation column or define the node as a product stream at any open node in the flowsheet. During this research, for every open node, the agent only had the ability to place either a column or end the stream, but as shown by Stops et al. [26], it is possible to adapt the model to also be able to place heat exchangers, reactors or recycle streams. For the distillation columns, the third-level decision variable was chosen to be the recovery values of the main components. There is no third level decision required for when the agent defines the open node as a product stream. Currently, it is only possible to set one third-level decision variable, but future extensions to the model could allow for additional third-level decision variable implementations.

Hierarchical agent architecture

The employed agent's architecture as proposed by Stops et al. [26] utilizes a hierarchical and hybrid action space structure as introduced by Zhou et al. [46], which entails that both discrete and continuous decisions are included, which are solved as multiple action-selection sub-problems. This structure, as shown in Figure 2.3, employs individual MultiLayer Perceptrons (MLPs) for each decision level, and a single MLP functions as a critic to evaluate the made decisions.

In the top-left corner of the diagram, the "Fingerprint generation" step is presented. Here, a flowsheet graph is transformed into a flowsheet fingerprint using a Graph Convolutional Network (GCN). Subsequently, the obtained fingerprint is passed to the "Actor" block, which comprises an additional GCN structure for the first decision level. This initial actor selects an open stream to extend the flowsheet, using the graph representation, where open streams are represented as "undefined" nodes. The ID of selected node then is concatenated with the previously found flowsheet fingerprint. Subsequently, the concatenated vector is passed to the second- and third-level actors, where decisions regarding the specific unit type and related design variables are made.

The second level actor, which is also implemented as an MLP, returns the probabilities for each type of unit operation to be chosen. Then, for the third-level decision, individual MLPs are used for each specific unit operation. These third-level MLPs take the concatenated vector, including the flowsheet fingerprint and the selected location's ID, as input, and output the beta-distribution parameters α and β. These values describe a beta distribution according to B(α, β), based on which continuous decisions regarding the respective design variables are made.

Additionally, in the top-right corner of the diagram, the critic structure is shown. After the flowsheet fingerprint is generated, it is passed to the "Critic" block, where it is passed through another MLP. The output value of the critic-MLP serves as an estimate of the anticipated reward that the

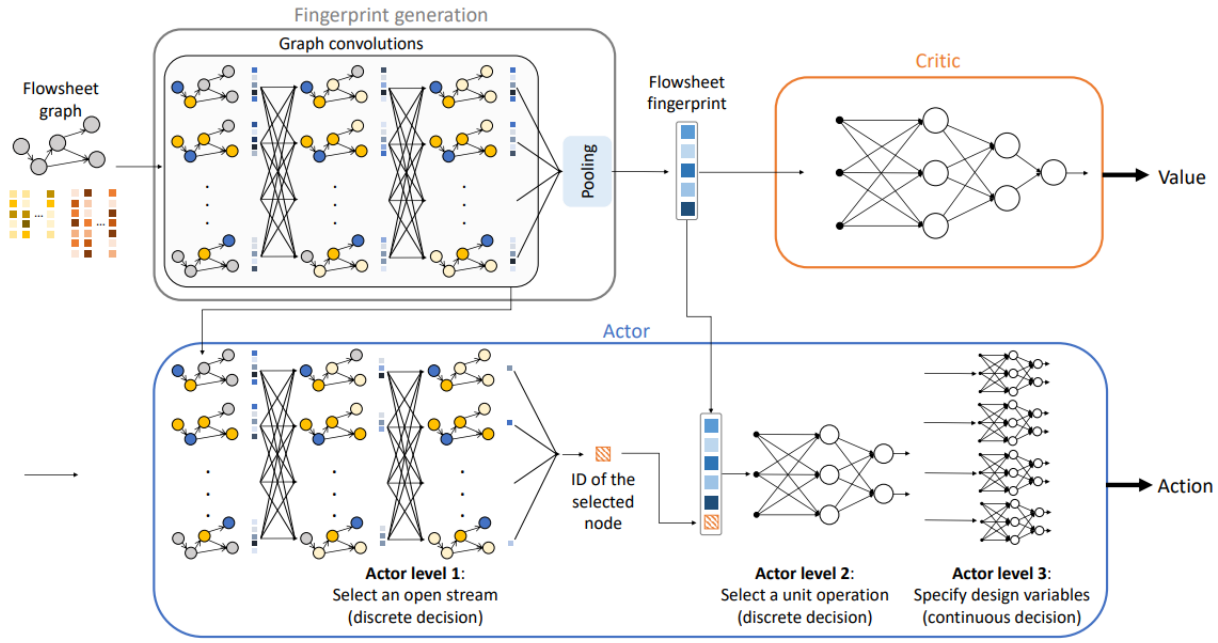

FIGURE 2.3: Full architecture of the deployed actor-critic agent as proposed by Stops et al. [26]

agent is expected to receive until the end of the episode, starting from the given state and employing the current policy. Then, the losses for the actor networks can be determined by computing the generalized advantage estimation $\hat{A}$ as introduced by Schulman et al. [47], which states whether an action outperformed or underperformed expectations. By comparing with the actual obtained rewards, the loss of the critic-MLP is computed. Finally, before the entire process starts again, the weights of the agent's MLPs are updated through gradient descent, employing a loss function derived from the summation and weighting of individual actor losses, their entropies, and the critic's loss.

Agent-environment interaction

The entirety of the agent-environment interaction is showcased in Figure 2.4 above. This figure was obtained from "Flowsheet synthesis through hierarchical reinforcement learning and graph neural networks" by Stops et al. [26].

The process initiates with the environment being set up, incorporating a specific feed-stream, consisting of a feed temperature, pressure, molar flow and the molar fractions of the components in the feed. Subsequently, the flowsheet is systematically generated through iterative steps. The agent observes the current state as provided by the environment (s) and makes decisions (a) at the three levels previously described. The agent then provides the probabilities, selected actions, and the value (v) associated with the state.

Then, the selected actions are applied to the environment, resulting in the computation of the next state (s') by simulating the flowsheet. Additionally, the environment checks for any remaining open streams. If no open streams are left, the flowsheet is completed. Given the random initialization of the agent's network weights, initial training episodes may result in the creation of very large flowsheets. Therefore, the total number of process units is capped at 25 and all remaining open streams at that point are designated as product streams. Second, the environment calculates the reward to give to the agent. A positive net cash flow corresponds to the reward being equal

to the net cash flow, while a negative net cash flow results in the reward being the net cash flow divided by a factor of 10. This is done to incentivize exploration by the agent. Process rewards of zero are assigned to the agent during intermediate steps in the synthesis.

```
done = False
while not done do
    observe state s
    actions a, probs p, value v = AGENT(s)
    new state s', reward r, done = ENV(a)
    store transition (s, a, p, r, done) in memory
end while

function AGENT(state s)
    for level=1,2,3 do
        probs p_level = actor(s)
        action a_level = sample(p_level)
    end for
    value v = critic(s)
    return a, p, v
end function

function ENV(actions a)
    next state s' = SimulateFlowsheet(a)
    if no more open streams then
        done = True
        reward r = NetCashFlow(s')
        if reward r < 0 then
            reward r = reward r / 10
        end if
    else
        reward r = 0 €
    end if
    return s', r, done
end function
```

FIGURE 2.4: Pseudocode of the agent-environment interaction as proposed by Stops et al. [26]

Chapter 3

Research Strategy

This research aimed to incorporate chemical composition and property knowledge into a previously developed RL agent for process synthesis by Stops et al. [26] via direct concatenation of the specific property onto the flowsheet fingerprint. Throughout this study, a simple distillation separation problem was considered, where an RL agent had the opportunity to place a distillation column or end the stream at any open node in the flowsheet, as the first and second-level decisions described in the previous section. Furthermore, as a third-level decision, the RL agent had the capability to choose both the distillate and bottoms product recoveries for each of the distillation columns. To ensure the model's efficacy and practicality, it should be able to generate viable flowsheets for a broad array of scenarios. Therefore, for this analysis, various scenarios were developed for testing the model by modifying the molar fractions of the components in the feed stream.

To start the research, first, a baseline performance was established to which later versions of the model were compared to determine whether changes in the model resulted in significant model performance improvements. This process was initiated by training the model on one specific feed-fraction use-case. Then, by assessing the model's ability to generate viable flowsheets based on the feed-fraction use-case it was trained on, a baseline model performance was established. Additionally, the model was tasked with generating optimal flowsheets for the feed-fraction use-cases it was not trained on initially. Through this methodology, the ability of the model to transfer knowledge it obtained previously for one scenario to other environments could be quantified.

To generalize and improve the performance of the model in environments that it was not explicitly trained on, several additional implementations were made to the model. First, the generality of the model was improved by training on multiple feed-fraction use-cases instead of exclusively on one. This was achieved by switching between the three different feed-fraction use-cases at a set switching frequency. The effect of the switching frequency on the transferability and average model score was also recorded in an effort to increase the understanding of the impact of varying switching frequencies on the overall model performance.

As a second additional implementation to the model, information regarding the actual process and chemical properties of the components in the streams was supplied to the RL agent with the aim of further improving the ability of the model to transfer knowledge it obtained previously for one scenario to other environments. This way, instead of basing its choices purely on factors related to the specific feed-fraction use-case, the agent would be able to consider more general thermodynamic processes and, therefore, make more informed decisions about when and where to place distillation columns in the flowsheet.

Two distinct categories of property information were conveyed to the model. First, the model was informed about the composition of the components in the entry stream. Providing this information to the agent could be consequential, given that fluctuations in component composition induce shifts in the thermodynamic equilibria within the column. Consequently, such shifts play a crucial role in determining the efficacy of column placement and adaptation of the recovery values. Therefore, supplying this information to the agent should allow the agent to make more informed decisions about where and when to place a column in the flowsheet, resulting in higher average model scores. Similarly, as a second step, the model was given information regarding the pure component boiling points of the components at atmospheric pressure. The specific boiling points of the individual components have a significant influence on the boiling point of the mixture in the column as well as the ease of separation. As the last step, the model was given information regarding both the component composition and boiling points to observe whether including both further improved or decreased the effects of the property integration on the model performance. It was expected that both the composition and boiling point integrations would improve the ability of the model to transfer knowledge it obtained from one feed-fraction use-case to other environments, as the agent would be able to make more informed decisions based on the additional thermodynamic information.

The current study aimed to provide valuable insights and methodologies applicable to broader industrial contexts. First, this research contributed to the field of automated process flowsheet design. If supplying thermodynamic information to an RL agent resulted in a more comprehensive understanding of the underlying physical and chemical processes by the agent, the viability of RL agents as a design tool in the field of process synthesis would be greatly improved. This understanding is crucial for designing processes and flowsheets that are robust across a range of conditions. Furthermore, by showing that through the incorporation of additional information the model's adaptability and decision-making precision could be improved, the results from this study can be applied to other areas where actor-critic RL is employed, such as robotics, game-playing algorithms, or the field of finance and trading.

Chapter 4

Methodology

Several adaptations to the model architecture were required before the RL agent would be able to generate viable flowsheets for distillation separation problems. Simultaneously, specific testing standards needed to be formulated to maintain consistency within the results. In the following section, the methodologies related to the development and evaluation of the environment and RL agent are outlined. First, the selected distillation separation use-case is described. Next, the necessary adaptations to the environment as well as the developed distillation environment are elaborated. Finally, the employed testing methodologies for the initial training, transferability testing and switch training are described in detail.

4.1 Use-case selection

Following the use-cases proposed by Stephanopoulos et al. [48] and Porter and Momoh [49], the separation of five paraffinic hydrocarbons via distillation was investigated. The five chosen components as well as the primary properties relevant to the distillation separation process are shown in Table 4.1 below.

Component	**Boiling point *(at 1 bar)* [K]**	**Vapor pressure *(at 293 K)* [Bar]**
Propane	231.1	12.41
n-Butane	262.0	4.61
i-Butane	273.0	3.26
n-Pentane	301.1	1.30
i-Pentane	309.2	0.97

TABLE 4.1: Selected paraffinic components and their distillation properties

Furthermore, as the standard values for each of the scenarios, the total feed was considered to be 100 mol/s. A pressure of 100,000 Pa was chosen for the feed stream and, as pressure effects were not taken into account during the simulation, the pressure remained constant throughout the flowsheet.

Where a previous case study by Jobson [50] only considered one feed composition, in this study, three distinct scenarios, referred to as feed-fraction use-cases, were examined. The first use-case entailed equimolar feed-fractions, where each of the five components had an equal molar fraction of 0.20 in the feed. Two additional cases were designated as the Light and Heavy leaning feed-fraction use-cases. The Light leaning use-case had a higher prevalence of components with lower molar masses in the feed compared to the Equimolar use-case. Conversely, in the Heavy leaning use-case, there was an increased prevalence of components with higher molar masses in the feed

compared to the Equimolar use-case. These three feed-fraction use-cases are summarized in Table 4.2 below.

Feed-fraction use-case	**Propane**	**n-Butane**	**i-Butane**	**n-Pentane**	**i-Pentane**
Light leaning	0.30	0.25	0.20	0.15	0.10
Equimolar	0.20	0.20	0.20	0.20	0.20
Heavy leaning	0.10	0.15	0.20	0.25	0.30

TABLE 4.2: Designated feed-fraction use-cases

4.2 Environment development

The first step in adapting the model so that the RL agent would be able to generate viable flowsheets for distillation separation problems, was to implement a more general distillation model. The previous version implemented by Stops et al. [26] was specifically focused on the production of methyl acetate (MeOAc) and its by-product water (H_2O) via the esterification of acetic acid (HOAc) with methanol (MeOH) and is, therefore, not viable for the separation of paraffinic hydrocarbons. The section below describes the modeling and validation of the distillation column that was implemented in the RL infrastructure.

4.2.1 Distillation column modeling

The modeling of the fractional distillation was done using the Fenske-Underwood-Gilliland-Kirkbride (FUGK) method, which is similar to the DiSTillation-Winn-Underwood (DSTWU) methodology that Aspen Plus employs. The Fenske equation serves to provide an estimation of the theoretical minimum number of plates or equilibrium stages required at total reflux. The Underwood equation estimates the minimum reflux necessary for an infinite number of theoretical equilibrium stages. By incorporating Fenske's minimum plate and Underwood's minimum reflux estimations, the Gilliland method can be used to determine the theoretical plates required for a specific distillation process at a selected reflux. Finally, the Kirkbride equation was employed to predict the optimum feed stage location. The method for using the FUGK method is described in more detail by Coker et al. [31]. Its application in the present process is presented below:

1. **Light key and heavy key determination**
   The elements within a distillation process that have predetermined fractional recoveries for both their distillate and bottoms are referred to as key components. Among these key components, the one with the highest volatility is designated as the Light Key (LK), while the least volatile component is referred to as the Heavy Key (HK). The keys are known as "adjacent" if they are neighbours in a listing of the components in order of volatility, and "split" if other components have a relative volatility in between the keys. For this research, the keys are assumed to always be "adjacent". All other components are classified as Non-Keys (NK). Components with a higher saturated vapor pressure than the LK are referred to as Light Non-Keys (LNK) and components with a lower saturated vapor pressure than the HK are referred are referred to as Heavy Non-Keys (HNK). The initial stage of the distillation column design process involves selecting the LK and HK and establishing their respective distributions in the top and bottom products. Generally, the LK is the component that is desired to keep out of the bottom product, while the HK component is to be kept out of the top product. Often a specified minimum recovery is required for the LK and HK, i.e. at least 95% of the LK should exit the column in the distillate stream or at least 95% of the HK should exit the column in the bottom stream. Subsequently, the fractions of the LK and HK in the other stream can be calculated according to:

$$x_{LK.B} = 1 - x_{LK.D} \tag{4.1}$$

$$x_{HK.D} = 1 - x_{HK.B} \tag{4.2}$$

in which $x_{LK.B}$ is the fraction of the LK in the bottoms (B) and $x_{LK.D}$ is the fraction of the LK in the distillate (D). Similarly, $x_{HK.B}$ is the fraction of the HK in the bottoms (B) and $x_{HK.D}$ is the fraction of the HK in the distillate (D).

2. **Saturated vapor pressure calculation**
The second step is to calculate the saturated vapor pressures of the components using Antoine's equation.

$$log_{10}(P^0) = A - \frac{B}{C + T_{Feed}} \tag{4.3}$$

A, B and C refer to Antoine coefficients that are specific to the substance. Empirically determined values for these parameters for several temperature ranges can be obtained from the National Institute of Standard and Technology (NIST) Chemistry Webbook. Practically, the saturated vapor pressure is a measure of the fugacity of a component or the ability of a component in a liquid mixture to escape, or vaporize, from the mixture. At each temperature, there is an equilibrium between the molar fractions of component i in the liquid and vapor phase, noted as x_i and y_i, respectively. This equilibrium is given by the following equation:

$$K_i = \frac{y_i}{x_i} = \frac{\gamma_i P_i^0}{\phi_i^V P} \tag{4.4}$$

in which γ_i is the the activity coefficient that accounts for the effects of the temperature, pressure and concentration of the components in the system on the deviation from the limiting ideal behavior. P_i^0 is the saturated vapor pressure. ϕ_i^V is the vapor phase fugacity coefficient and P is the pressure in the system. At moderate pressures and if the liquid phase behaves as an ideal solution, both ϕ_i^V and γ_i go to 1, reducing Equation 4.4 to Raoult's Law.

$$K_i = \frac{P_i^0}{P} \tag{4.5}$$

3. **Relative volatility calculation**
Subsequently, the relative volatility with respect to the HK of all components can be estimated according to Equation 4.6.

$$\alpha_i = K_i / K_{HK} \tag{4.6}$$

To be able to employ this equation the system is assumed to be ideal, as only in that case, the relative volatility is numerically equal to ratio of the saturated vapor pressures of the pure component and the HK, as demonstrated in Equation 4.4. The relative volatility can function as an indicator of the separation's degree of difficulty. As the relative volatility of a component compared to the HK becomes larger, the separation becomes easier to achieve. If we combine Equation 4.5 with Equation 4.6, the following relationship can be obtained:

$$\alpha_i = P_i^0 / P_{HK}^0 \tag{4.7}$$

Ordinarily, at this stage, the average volatility of the LK over the column is estimated or calculated by taking the geometric mean of the relative volatility of the LK at the distillate temperature and the bottoms temperature, according to Equation 4.8. This requires prior knowledge about or an estimation of the distillate and bottoms compositions to calculate the required relative volatilities of the mixtures. As these are not generally not known, an iterative procedure may be required to estimate them. To reduce the required computer power and training times, in this case, $\alpha_{LK.avg}$ is assumed to be equal to α_{LK} at the feed temperature, T_{feed}. Another advantage of the Fenske equation is that it is the most reliable relationship for conditions in which the relative volatility is almost constant [51].

$$\alpha_{LK.avg} = \sqrt{\alpha_{LK.D} * \alpha_{LK.B}} \tag{4.8}$$

4. **Fenske equation - Determination of minimum number of stages,** N_{min}
Now, the minimum number of stages can be estimated using the Fenske equation. Since the minimum number of stages needs to be determined, total reflux is assumed for this equation.

$$N_{min} = \frac{\log\left[\left(\frac{x_{LK}}{x_{HK}}\right)_d * \left(\frac{x_{HK}}{x_{LK}}\right)_b\right]}{\log(\alpha_{LK})} \tag{4.9}$$

As mentioned, α_{LK} refers to the relative volatility of the LK component at T_{feed}.

5. **Hengstebeck-Geddes equation - Determination of NK component splits**
Often, a simplification can be made when the amount of the NK components is small, or when the components form near-ideal mixtures. In scenarios where the concentration of NK components is below 10%, they can be combined with the key components. However, when the concentration of NK components is greater, an alternative method proposed by Hengstebeck and Geddes should be employed to convert the system into an equivalent binary system. As the aim is to consider the separation of a multi-component mixture, the overall top and bottom flow rate compositions should be recalculated based on the Hengsteback and Geddes equation.

$$\log\left(\frac{d_i}{b_i}\right) = A + C\log(\alpha_i) \tag{4.10}$$

where the constants A and C are defined as:

$$A = \log\left(\frac{d_{HK}}{b_{HK}}\right) \tag{4.11}$$

$$C = N_{min} \tag{4.12}$$

Second, the following mass balance over the column holds:

$$f_i = b_i + d_i \tag{4.13}$$

Then, by combining Equation 4.10 and Equation 4.13, the molar flow of component i in the bottoms can be calculated according to:

$$b_i = \frac{f_i}{10^{A+C\log(\alpha_i)} + 1} \tag{4.14}$$

Additionally, the molar flow of component i in the distillate follows from the mass balance in Equation 4.13.

6. **Underwood equations - Determination of minimum and actual reflux ratios, R_{min} and R**
Now that the molar flows in the bottoms and distillate streams are known, the minimum reflux ratio, R_{min}, can be estimated using the first and second Underwood equations.

$$1 - q = \sum \frac{\alpha_i x_{i.f}}{\alpha_i - \theta} = 0 \tag{4.15}$$

where, $\alpha_{HK} \leq \theta \leq \alpha_{LK}$ (or since the relative volatilities are determined compared to the HK, $1 \leq \theta \leq \alpha_{LK}$) and q is the quality of feed. In practice, we assume the feed is always at its boiling point, meaning $q = 1$. Now, through a series of trial-and-error measurements, the θ value can be determined so that the summation in Equation 4.15 equals 0. Subsequently, the determined θ value is inserted into the second Underwood equation.

$$R_{min} + 1 = \sum \frac{\alpha_i x_{i.d}}{\alpha_i - \theta} \tag{4.16}$$

Finally, R_{min} is found. The operating reflux ratio, R, is selected as a factor of the minimum reflux ratio, typically factors between 1.2 and 1.5 are considered. In this case, a factor of 1.3 was chosen.

$$R = 1.3 * R_{min} \tag{4.17}$$

7. **Gilliland equation - Determination of the actual number of stages, N**
Then, based on the minimum and operating reflux ratios, R_{min} and R, the actual number of stages, N, is determined based on Gilliland's correlation. The Gilliland correlation utilizes design data to present a graphical correlation, which is viable for X values between 0.02 and 1.00. Several Gilliland correlation model equations are available. In this case, the correlation fit by McCormick et al. (Equation 4.20 [52]) was chosen as it showed the best fit according to the Average Absolute Deviation (AAD), Maximum Absolute Deviation (MAD), Expanded Uncertainty ($\pm$ $U_{95\%}$) and Coefficient of Determination (R^2) values [53].

$$X = \frac{R - R_{min}}{R + 1} \tag{4.18}$$

$$Y = \frac{N - N_m}{N + 1} \tag{4.19}$$

$$Y = 1 - X^{0.00456\ln(X)+0.44} \tag{4.20}$$

8. **Kirkbride equation - Determination of the feedstage location**
   Finally, the feed stage location was determined based on the Kirkbride equation.

$$\frac{m}{p} = \left[\left(\frac{B}{D} \right) \left(\frac{x_{HK}}{x_{LK}} \right)_F \left(\frac{x_{LK.B}}{x_{HK.D}} \right)^2 \right]^{0.206} \tag{4.21}$$

   where, m is number of stages above feed plate and p is number of stages below feed plate. Thus, the following logically holds:

$$N = m + p \tag{4.22}$$

For the final implementation into the model, only the component splits are relevant, as the final reward is not influenced by the number of stages, reflux ratios, and feed-stage location. These additional parameters were determined to be able to validate the accuracy of the distillation column model using the methodology described in the next section. Additionally, as a simplified non-iterative process was employed and, therefore, no temperature profile was available within the column, Antoine's equation (Equation 4.3) could not be used. Instead, the distillation column model was supplied directly with the vapor pressure at 293 K (listed in Table 4.1). Then, the distillation column modeling process resumed from Step 3.

Distillation column model validation

Then, to ensure the validity of the implemented distillation column model, simplified analyses were run in the Aspen Plus modeling software. By setting the same key component recovery values and utilizing the same feed temperature, pressure and feed molar flow and fractions, the results obtained from a DSTWU column in Aspen Plus and the distillation column as described in the previous section could be compared. Primarily the values found for the distillate and bottoms streams needed to be the consistent for both columns for the implemented distillation column model to be considered valid. Additionally, also the obtained values for the number of stages, reflux ratio and, feed-stage location were compared to ensure the full validity of the model. Alterations to the distillation column model would be made if significant inconsistencies were found.

4.2.2 Reward function

As the nature of the environment and the components in the system were changed, alterations to the reward function were also necessary. The reward that is passed to the agent is based on the model score or the net cash flow of the overall process flowsheet in M€ per year. A positive net cash flow corresponds to the reward being equal to the net cash flow, while a negative net cash flow results in the reward being the net cash flow divided by a factor of 10. The total net cash flow of a process flowsheet can be divided into three separate sections: First, the annual cash flows related to the acquisition and sales of the process streams can be derived from the stream purity and pure component prices. Subsequently, the annualized costs of the distillation columns in the flowsheet are found. Finally, the utility costs related to the heating and evaporation of the components in the process equipment are obtained. By combining these three factors, the total net cash flow of the overall process flowsheet is determined. The overall relationship is summarized in the following equation.

$$\text{Net cash flow} = \frac{\text{Revenue} - \text{Annualized capex} - \text{Utility costs}}{10^6} \tag{4.23}$$

Acquisition and sale of process streams

In Table 4.3, an overview is given of the pure component prices of the five chosen components in the system.

Component	Price [€/mole]
Propane	1.30E-2
n-Butane	1.97E-2
i-Butane	2.16E-2
n-Pentane	2.44E-2
i-Pentane	2.68E-2

TABLE 4.3: Pure component prices *(rounded to two decimals)*

To calculate the final selling price of a process stream based on the existing fractions of the components x and their respective pure component costs P, the following relationship can be used:

$$\sum_{i=1}^{C} 0.5 + 0.5 * \tanh(35 * (x_i - 0.5)) * P_i \tag{4.24}$$

in which C is the set of all components and i is one of the components. The relationship between the molar fraction of a component in a stream and the selling price related to that component is visualized in Figure 4.1 below.

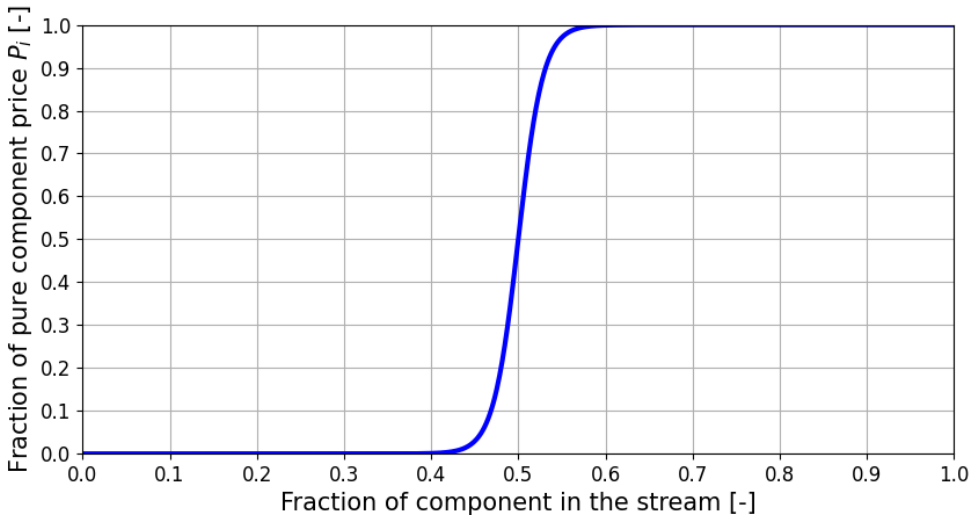

FIGURE 4.1: Component purity-price relationship

During this research, only the revenue based on the effluent streams was considered, as during the analysis comparisons were only made between scenarios where the feed-fractions and input-flows remained constant. Therefore, disregarding the input-streams reduced the possibility of errors related to the pure component prices, as they were only used once in calculations instead of twice.

Annualized fixed capital costs of the process equipment

In addition to costs related to components in the process units, the costs related to the units themselves needed to be considered. For every process unit the capital costs C were estimated based on relationships devised by Smith [54], starting with the determination of the mass capacity c_m of a distillation column, according to:

$$c_m = \frac{150 * \sum_{i=1}^{C} F * f_i * M_{r.i}}{4.6} \tag{4.25}$$

in which F is the molar feed flow, f_i refers to the feed-fraction of component i and $M_{r.i}$ is the relative molar mass of component i. Then, based on the mass capacity, the capital costs C are calculated to be the larger of:

$$C = 6.56 * 10^4 * \left(\frac{c_m}{8}\right)^{0.89} \text{ and: } C = 6.56 * 10^4 * 1^{0.89} \tag{4.26}$$

As these cost relationships are based on 2000 prices, the obtained process unit capital costs needed to be adjusted to a common time basis via multiplication with a factor of 5.8 and the 2019 Chemical Engineering Plant Cost Index (CEPCI). This relationship is shown in Equation 4.27 below.

$$C_I = C * 5.8 * \frac{607.5 \text{ (2019 CEPCI)}}{394.3 \text{ (2000 CEPCI)}} \tag{4.27}$$

Finally, as the yearly depreciation, property taxes and insurance cost were assumed to be equal to 13% of total capital costs, the annualized fixed capital costs of this piece of process equipment was found to be equal to $0.13 * C_I$ [55]. The total annualized fixed capital costs were found by repeating this procedure for every distillation column in the flowsheet.

Utility costs

The total utility of the distillation columns was assumed to be equal to 2.5 times the heat of vaporization of the products. To find the total heat duty Q_T, the heat of evaporation of each component $\Delta H_{vap.i}$ was multiplied by the number moles of that component coming out of the column in the distillate d_i for each distillation column n, according to:

$$Q_T = 2.5 \sum_{n=1}^{N} \sum_{i=1}^{C} \Delta H_{vap.i} * d_{i.n} \tag{4.28}$$

Subsequently, Q_T was annualized by assuming 8000 yearly operational hours. Further, it was assumed that all heating was done using pure water vapor at a temperature of 550 K and a specific heat c_{steam} of 1.996 kJ/kgK. The cost of this type of steam P_{steam} was assumed to be 5.49 € per MT in accordance with a 2019 International Energy Agency (IEA) report [56]. Finally, the total financial cost related to the utility U were calculated according to:

$$U = \frac{Q_T}{c_{steam}} * P_{steam} \tag{4.29}$$

4.2.3 Model parameter selection

Finally, as the environment was changed, the decisions the RL agent had the ability to take were also altered compared to the original work by Stops et al. [26]. In the paragraphs below, an overview is given of the final first-, second- and third-level decisions the agent was able to take.

First-level decision

The first-level decision remained unchanged from the original work. For this decision, The RL agent still selects one of the open locations within the flowsheet.

Second-level decision

As mentioned in Chapter 2: Theoretical Background, during this research, a simple distillation separation problem was considered. Therefore, the second-level decision was changed so that, for every open node, the agent only had the ability to either place a distillation column or declare the stream to be a product stream, ending the flowsheet. For this decision, the original model was also able to place heat exchangers, reactors or recycle streams [26].

Third-level decision

Finally, the original model by Stops et al. [26] used the distillate to feed ratio D/F as the continuous third-level decision related the distillation column. As the used distillation column model was altered significantly, the third-level decision was changed as well. Jobson [50] demonstrated that, for the same paraffin use-case that is considered during this research, it holds that the most ideal distillation train based on the short-cut methods of Fenske, Underwood and Gilliland can be found by selecting the LK and HK components so that the distillate and bottoms molar flows are as close equal as possible. However, where Jobson assumed that the recovery of the key components were 99%, for this research, the agent is allowed to make a choice on the key component recovery, marking it as the third-level agent decision. The agent is able to choose recovery values between 0.5 and 0.99.

4.3 Chemical composition and property integration via fingerprint-concatenation

In the present study, an RL agent was informed about the chemical compositions and properties in the flowsheet via direct concatenation onto the flowsheet fingerprint. After the flowsheet graph is transformed into a flowsheet fingerprint using a GCN as shown in the top-left corner of Figure 2.3, the values of a property are added at the end of the generated fingerprint, prior to it being passed to the "Actor" and "Critic" blocks.

Two categories of property information were conveyed to the RL agent in this way. First, the model was informed about the molar composition of the components in the entry stream. Providing this information to the RL agent could be impactful, given that variations in the component concentrations influence the thermodynamic equilibria within the column. Similarly, as a second option, the model was given information regarding the pure component boiling points of the components at atmospheric pressure. The specific boiling points of the individual components have a significant influence on the boiling point of the mixture in the column as well as the ease of separation, as was shown in the section about distillation column modeling. The boiling points are listed in Table 4.1 and remained constant throughout the experiment. As the last step, the model

was given information regarding both the composition and boiling points to observe whether including both parameters further improved or decreased the effects of the property integration on the overall model performance.

4.4 Testing methodologies

In the upcoming section on testing methodologies, the approaches to assess the performance and capabilities of the RL agent are described. The section starts with a description of the statistical methods that were used to determine whether one model outperformed another. Second, the process behind the generation of the optimal flowsheets and model performance benchmarks is explained. The subsequent section about transferability testing assesses the agent's adaptability across diverse environments. The first implemented alteration to the model's training process is elaborated in the switch training testing section. The initial study, transferability testing and switch training procedures were repeated for each level of property integration. To close out the methodology section, an overview of all the conducted tests is supplied.

4.4.1 Statistical improvement tests

For this research, the independent t-test served as the primary analytical method to determine the statistical significance of the difference in performance between different models. This type of test is commonly employed to determine whether there is a significant difference between the means of two independent groups, where the data in each group is normally distributed and the variance of the two groups is homogeneous. To increase the power of the t-test, and since it is expected that models with additional implementations will exhibit higher average scores, a lower-tailed alternative hypothesis $H_{1.LT}$ is employed, which assumes that the difference between the means of the observations μ_d is less than zero. Then, when the p-value falls below a predefined significance-level α commonly set at 0.05 or 0.10, $H_{1.LT}$ is rejected in favour of the Null hypothesis H_0, which suggests that the true mean difference μ_d is equal to 0. In this case, given the preliminary nature of the research and need for validation of the testing methodologies, an α-value of 0.10 was chosen. It is only tested whether the model with additional implementations performed better than the benchmark models. However, in the case that the p-value is higher than 0.10, it can not be determined whether the two models performed equally or if the benchmark model actually outperformed the model with additional implementations, based on only the results from the lower-tailed alternative hypothesis t-test. Nevertheless, for this research, it is only relevant whether the model with additional implementations outperformed the benchmark model.

If it is unknown which of the two models should exhibit a higher average model performance, which is the case when two different types of model implementations are compared, the two-tailed alternative hypothesis $H_{1.TT}$ is used. This hypothesis assumes that the difference between the means of the observations, μ_d, is not equal to zero. If $H_{1.TT}$ is rejected, again the same Null hypothesis H_0 as before is adopted.

4.4.2 Initial study

The first test that was conducted is the initial study. Initially, for each of the three feed-fraction use-cases, the RL agent was trained for 50,000 episodes and, subsequently asked to present the flowsheet it generated during training that achieved the highest model score. Subsequently, the effectiveness of the flowsheet was evaluated using the optimality model, which is commonly employed in the field of evolutionary biology. In this case, the RL agent can be treated as an organism and the optimal flowsheet it generated during training can be assumed to be its optimal behavior, where the difference between benefits and costs is maximized. The optimality of the learned

behavior by the agent can be confirmed by testing and evaluating different strategies via trial-and-error. For example, additional distillation columns could be added to the flowsheet or the recovery of the existing columns could be adapted to be higher or lower. If the flowsheet is optimal, any alteration in strategy should result in a lower final model score. The optimal flowsheets form the first comparison metric.

Second, the primary model evaluation should be based on the performance after training rather than during training. Therefore, after the training process, the final model weights were saved. Then, using the same weights, the agent was asked to generate an additional 20 flowsheets for the same feed-fraction use-case it was trained on initially, thereby establishing a benchmark to compare any future results to. Because of the stochastic nature of the model, one flowsheet is not sufficient to draw any conclusions and it is, therefore, necessary to generate several flowsheets to capture the model behavior accurately. This process was repeated for each of the three feed-fraction use-cases, resulting in a performance benchmark for the Light leaning, Equimolar, and Heavy leaning feed-fraction use-cases.

4.4.3 Transferability testing

To test whether the agent is able to leverage knowledge gained in one environment to improve its performance in a different but related environment, a transferability test was employed. The transferability of the agent was quantified by tasking the agent with generating 20 flowsheets for the two feed-fraction use-cases it was not explicitly trained on based on the model weights from the initial studies. These flowsheets are then compared to the established benchmarks. This way, the ability of the model to transfer learned knowledge to other use-cases was evaluated by determining to what degree a good model performance on one feed-fraction use-case resulted in a good model-performance on other feed-fractions.

4.4.4 Switch training testing

Then, in an effort to increase the generality of the model, a switch training methodology was employed. This is a form of curriculum learning, where the RL agent is trained on a diverse set of environments to expose the agent to various scenarios and encourage generalization. The aim is to train the model on the essential underlying processes while reducing its reliance on the initial feed-fraction. By training on a set of varying feed-fractions, the RL model is likely to develop more robust policies that can handle unexpected situations and disturbances.

This methodology was implemented by training on all three feed-fractions, switching between the feed-fraction use-cases at a set frequency. Initially, three different switching frequencies were tested; switching between the feed-fractions every 1, every 100 and every 500 episodes. The sequence of the feed-fraction use-cases remained consistent for every test, starting with the Equimolar case, according to:

$$\text{Equimolar} \rightarrow \text{Light leaning} \rightarrow \text{Heavy learning} \rightarrow \text{Equimolar} \rightarrow ... \tag{4.30}$$

In total, the model was again trained for 50,000 episodes as was the case for both the initial study and the transferability tests. After the training process was complete, the model weights were saved, and the model was tasked with making a prediction of the optimal flowsheet for each of the three feed-fraction cases. Because of the stochastic nature of the model, the prediction of the optimal flowsheet was, again, repeated 20 times for each feed-fraction use-case.

Random feed-fraction introduction

Furthermore, a fourth type of switch training testing was attempted. Instead of cycling between only the three established feed-fraction use-cases, a random feed-fraction was introduced. In this experiment, prior to every episode, a new random feed-fraction was obtained by generating five random values and normalizing them so that their sum equaled 1. Consistent with the switch training methodology, the model was again trained for 50,000 episodes. After the training process was complete, the model weights were saved, and the model was tasked with making a prediction of the optimal flowsheet for each of the three feed-fraction cases.

4.4.5 Summary of the conducted tests

As a summary of the testing methodology section, an overview of the considered testing cases is given in Table 4.4. Each entry in the table represents 20 measurements. The same measurements were conducted for the base case, the composition integration case, the boiling point integration case, and the combined composition and boiling point integration case. Additionally, based on the initial study results for the base case, the overall optimal flowsheets were determined.

Initial study	**Transferability testing**	**Switch training**
Light leaning	Equimolar (Light leaning)	Light leaning (1)
Equimolar	Heavy leaning (Light leaning)	Equimolar (1)
Heavy leaning	Light leaning (Equimolar)	Heavy leaning (1)
	Heavy leaning (Equimolar)	Light leaning (100)
	Light leaning (Heavy leaning)	Equimolar (100)
	Equimolar (Heavy leaning)	Heavy leaning (100)
		Light leaning (500)
		Equimolar (500)
		Heavy leaning (500)
		Light leaning (1 (Random)
		Equimolar (1 (Random))
		Heavy leaning (1 (Random))

TABLE 4.4: Overview of the conducted tests

Finally, for the composition integration case, the boiling point integration case, and the combined composition and boiling point integration case, one-sided t-tests were conducted to compare all the measurements in the table above to the initial study results for the base case.

Chapter 5

Results

Throughout this study, a simple distillation separation problem was explored, employing an RL agent to make the critical decisions in the design of a process flowsheet. The research commenced by establishing a benchmark performance by means of training the model on specific feed-fraction use-cases. Then, through several additional implementations, attempts were made to generalize the model and improve its performance in diverse environments. The subsequent section presents and reviews the results of the conducted tests.

5.1 Performance benchmark analysis

The performance benchmark analysis consisted of four parts. First, the optimal flowsheets and related recovery values are shown. Then, the initial model performance benchmark is portrayed for each feed-fraction use-case. Similarly, a benchmark for the ability of the model to transfer knowledge it obtained from one scenario to other environments was established. Finally, a flowsheet topology analysis was conducted to determine to what degree the model without any additional implementations was able to transfer knowledge from one feed-fraction use-case to another.

5.1.1 Optimal flowsheets

To obtain the first performance benchmarks to which later versions of the model could be compared, the model was trained for 50,000 episodes for each of the use-cases. The flowsheets that achieved the highest model scores during the training process were recorded. Then, based on a trial-and-error procedure as described by the optimality model, the three optimal model scores were found to be 21.04, 24.73 and 29.29 for the Light leaning, Equimolar and Heavy leaning feed-fraction use-cases, respectively. In Figures 5.1, 5.2 and 5.3, the optimal flowsheets for each of the feed-fraction use-cases are displayed. The development of the model scores during the training processes are shown in Appendix A for the three use-cases. Additionally, for these three flowsheets, the recovery values that were found to be optimal for each of the columns are shown in Table 5.1.

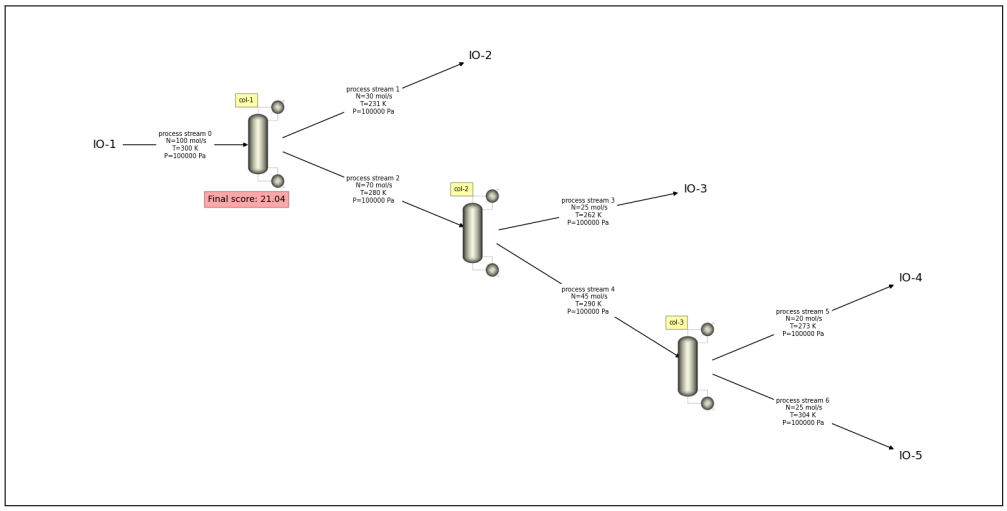

FIGURE 5.1: Optimal flowsheet for the Light leaning feed-fraction use-case

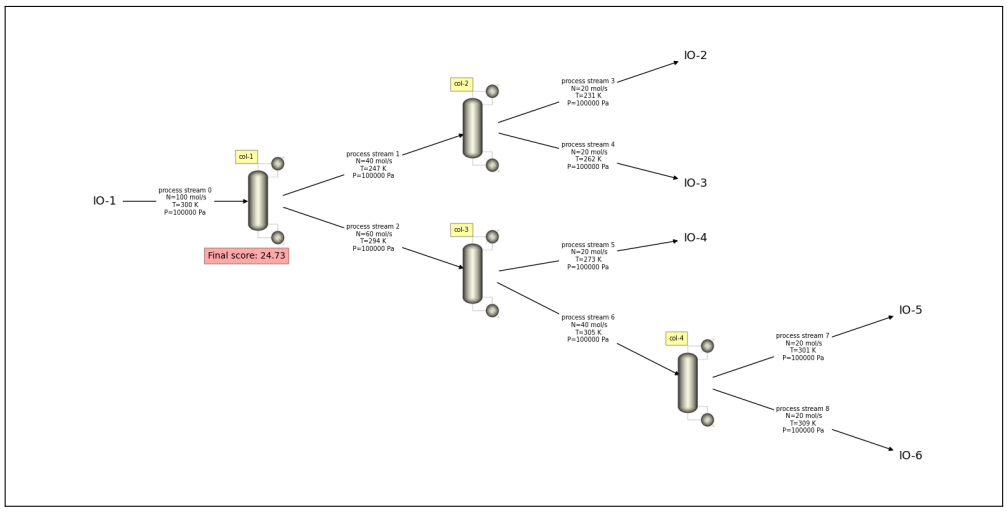

FIGURE 5.2: Optimal flowsheet for the Equimolar feed-fraction use-case

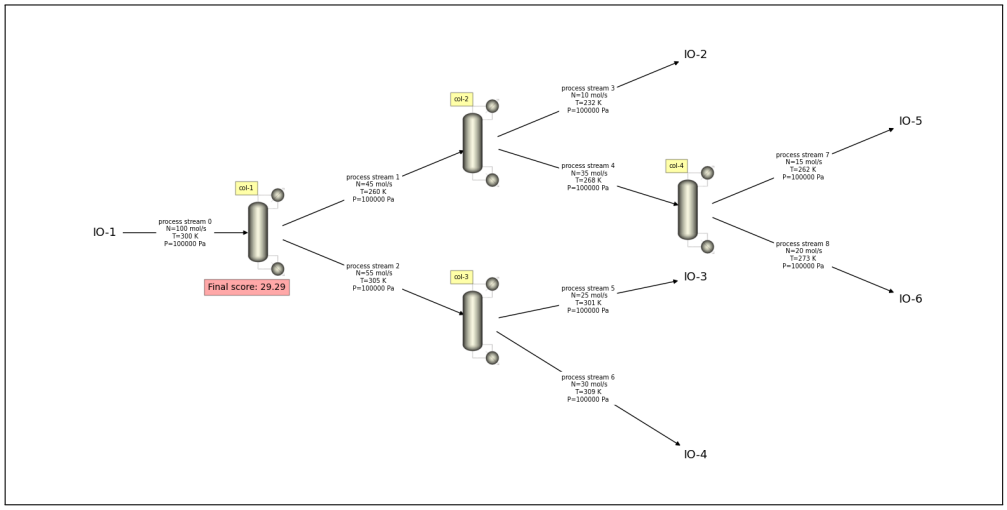

FIGURE 5.3: Optimal flowsheet for the Heavy leaning feed-fraction use-case

	Feed-fraction use-cases		
Process unit	**Light leaning**	**Equimolar**	**Heavy leaning**
col-1	0.99	0.94	0.99
col-2	0.99	0.99	0.99
col-3	0.99	0.99	0.99
col-4	-	0.99	0.99

TABLE 5.1: Optimal recovery values

On first observation, the flowsheet topology is relatively consistent between the three optimal flowsheets. In the Equimolar and Heavy leaning optimal flowsheets, a sufficient number of distillation columns exist to separate the mixture into the near-pure individual components. In the Light leaning optimal flowsheet only three distillation columns exist, as the costs of placing an additional column outweigh the financial benefits of separating the propane and i-butane left in process stream 1. Additionally, based on the results in Table 5.1, it can be observed that distillation columns with key component recovery values of 0.99 are the most financially beneficial. Only for col-1 in the Equimolar optimal flowsheet, a different recovery value was found. During additional analysis, it was found that increasing the recovery value meant changing the split location as the distillate and bottoms molar flows of that column are programmed to be as close as possible to equal. Therefore, further increasing this recovery value would result in a less optimal flowsheet.

5.1.2 Initial model performance benchmarking

On the same training instances from which the optimal flowsheets in the previous section were obtained, the initial model performance benchmarking was conducted. This entailed saving and using the same model weights from the initial training and asking the model to generate flowsheets for the same feed-fraction use-case it was trained on. The findings of this evaluation are presented in Table 5.2. A complete overview of the obtained model scores is given in Appendix B.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	21.04	17.75	25.70
Mean score	21.04	17.75	25.69
Min. score	21.02	17.72	25.57
Standard deviation	0.01	0.01	0.03

TABLE 5.2: Initial model performance benchmark

It is important to note that the model was only able to generate the previously found optimal flowsheet for the Light leaning use-case. For both the Equimolar and Heavy leaning cases, the model consistently generates a flowsheet that does not reach the same model score as the obtained optimal flowsheets. The differences between the average model performance and optimal flowsheet are -6.98 and -3.59 for the Equimolar and Heavy leaning cases, respectively.

5.1.3 Transferability benchmarking

In an effort to gauge the ability of the model to transfer knowledge it obtained from one scenario to other environments, the model was also tasked with generating optimal flowsheets for the feed-fraction use-cases it was not trained on initially. In Tables 5.3, 5.4 and 5.5, the performance of the model on the feed-fraction use-cases it was not trained on is shown using the maximum, mean,

minimum and standard deviation values of the model scores. Again, a complete overview of the model scores obtained during the transferability benchmarking process is given in Appendix C.

	Feed-fraction use-cases	
	Equimolar	**Heavy leaning**
Max. score	6.87	10.89
Mean score	6.81	10.88
Min. score	6.77	10.86
Standard deviation	0.01	0.02

TABLE 5.3: Light leaning transferability benchmark

	Feed-fraction use-cases	
	Light leaning	**Heavy leaning**
Max. score	1.55	25.70
Mean score	1.55	25.70
Min. score	1.54	25.67
Standard deviation	0.00	0.01

TABLE 5.4: Equimolar transferability benchmark

	Feed-fraction use-cases	
	Light leaning	**Equimolar**
Max. score	1.55	17.75
Mean score	1.54	17.74
Min. score	1.50	17.62
Standard deviation	0.01	0.03

TABLE 5.5: Heavy leaning transferability benchmark

5.1.4 Flowsheet topology analysis

As shown by the small value of the standard deviation and the marginal difference between the maximum and minumum model scores for the tests in the performance benchmark analysis, the model is consistently generating the same flowsheet for each feed-fraction use-case. For the transferability benchmark, in addition to finding the same flowsheet for every run of the same feed-fraction use-case, the model finds the same flowsheet for all three feed-fraction cases. This flowsheet is shown in Figure 5.4 below. This flowsheet was obtained for the Equimolar feed-fraction use-case, but the same flowsheet topology was found for the Light and Heavy leaning use-cases. These initial results demonstrated an apparent inability to transfer obtained knowledge from one scenario to other environments. Therefore, additional model implementations aimed at generalizing and improving the performance of the model in environments that it was not explicitly trained on, were required.

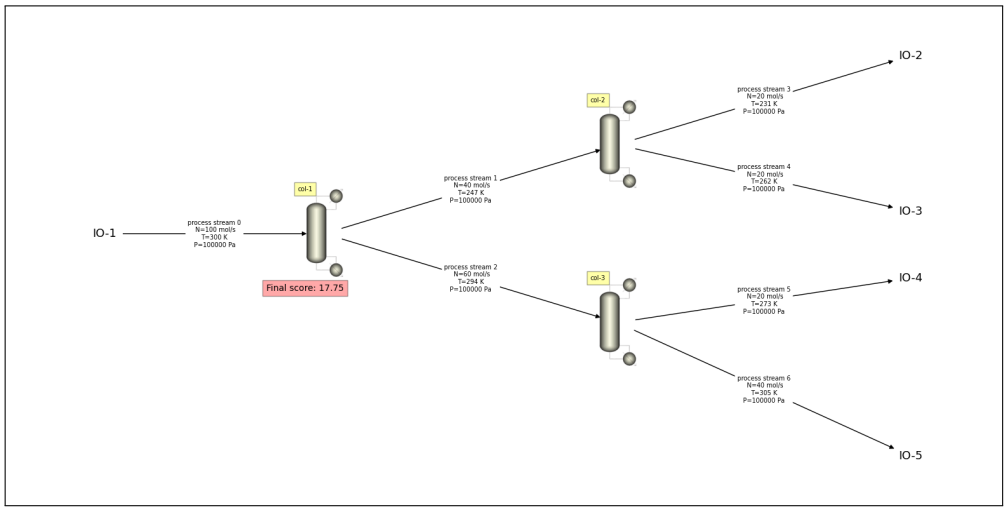

FIGURE 5.4: Main flowsheet obtained during the transferability benchmarking

5.2 Switch training results

An in effort to generalize the model and improve its performance in environments that it was not trained on, a switch training methodology was employed. By switching between the feed-fraction use-cases at a set frequency, the likelihood of the model overfitting on one of them was reduced. Three different switching frequencies were tried: every 1, every 100 and every 500 episodes. Additionally, a version was tried where a random feed-fraction was generated prior to every episode with the aim of reducing the likelihood of the model overfitting on one use-case even further. The results of this analysis are shown in Tables 5.6, 5.7, 5.8 and 5.9.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	20.95	17.39	26.14
Mean score	7.22	9.37	18.34
Min. score	-0.22	-1.59	9.47
Standard deviation	7.56	5.41	4.83

TABLE 5.6: Initial switch training model performance (switching frequency: 1)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	16.69	17.73	25.70
Mean score	2.67	14.78	24.44
Min. score	-0.58	2.11	20.03
Standard deviation	4.15	4.64	2.00

TABLE 5.7: Initial switch training model performance (switching frequency: 100)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	1.55	17.75	25.70
Mean score	1.45	17.67	25.11
Min. score	-0.32	17.50	16.09
Standard deviation	0.41	0.08	2.09

TABLE 5.8: Initial switch training model performance (switching frequency: 500)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	20.46	17.21	25.61
Mean score	3.10	6.89	16.79
Min. score	-1.07	-0.46	7.25
Standard deviation	5.85	5.20	6.78

TABLE 5.9: Initial switch training model performance (switching frequency: 1 (random))

Measurements where the switch training strategy outperformed the initial benchmark are marked in green. If the same result was obtained the table cell is marked in orange. In all other cases, the initial benchmark outperformed the employed switch training strategy. The main observation from these results is that, for the higher switching frequencies, the model does not consistently generate the same flowsheet anymore. It is, therefore, assumed that exposing the agent to each feed-fraction for a smaller amount of time prevented the agent from overfitting on the specific feed-fraction use-case. This conclusion is supported by the fact that when the switching frequency gets lower, the same results are found as for the transferability benchmarking where the same flowsheet topology was generated for each feed-fraction use-case.

Furthermore, to determine whether the employment of a switch training strategy significantly improved the overall model performance and its ability to transfer obtained knowledge, an independent one-sided t-test was employed. The resulting p-values are shown in Table 5.10 below. A p-value below 0.10 reflects that the switch training strategy produced significantly higher model scores than the benchmark case.

	Feed-fraction use-cases		
Switching frequency	**Light leaning**	**Equimolar**	**Heavy leaning**
1	1.000	1.000	1.000
100	1.000	1.000	1.000
500	1.000	0.994	0.993
1 (random)	1.000	1.000	0.881

TABLE 5.10: P-values from the independent one-sided t-test comparing the model scores from the benchmark and switch training strategy

Evidently, the model performance and the model's ability to transfer obtained knowledge were not significantly improved by implementing a switch training strategy. However, as on some occasions, the switch training outperformed the initial benchmark performance, employing the switch training strategy might still be beneficial in practice. Therefore, it was also tried in combination with the property integration.

5.3 Property integration

To further improve the model's ability to transfer knowledge from one feed-fraction use-case to another, the RL agent was supplied with information regarding the processes taking place in the distillation column and the chemical properties of the components in the streams. Initially, the agent was informed about the molar composition of the the feed stream. As a second step, the model was given information regarding the pure component boiling points of the components at atmospheric pressure. As the final integration step, the agent was given information regarding both the feed-composition and boiling points to observe whether including both further improved or decreased the effects of the property integration on the model performance. Finally, at the end of this section, a detailed analysis of the property integration results is given.

For each of these property integration levels, the initial model performance benchmarking, transferability benchmarking and switch training methodologies were repeated. In the interest of clarity, the initial model performance and transferability benchmarks were kept separate in the previous section. However, for the property integration steps, the initial model performance and transferability tables are combined into a single table, as they are all based on inference based on the same model weights. The feed-fraction use-case that the model was trained on is marked in yellow. Additionally, any instance where the property integrated case outperformed the base case, is marked in green and any instance where the model score was found to be the same is marked in orange. In all unmarked cases, the benchmark outperformed the property integrated case.

5.3.1 Composition integration

In Tables 5.11, 5.12 and 5.13, the results from the composition integrated model performance and transferability benchmarks are shown for each of the feed-fraction use-cases.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	21.04	6.82	10.89
Mean score	21.02	6.80	10.87
Min. score	20.90	6.71	10.71
Standard deviation	0.04	0.03	0.04

TABLE 5.11: Light leaning composition integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	1.55	17.75	25.70
Mean score	1.55	17.74	25.49
Min. score	1.54	17.68	21.52
Standard deviation	0.00	0.02	0.91

TABLE 5.12: Equimolar leaning composition integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	1.55	17.75	25.70
Mean score	1.46	17.74	25.69
Min. score	-0.18	17.69	25.63
Standard deviation	0.38	0.02	0.02

TABLE 5.13: Heavy leaning composition integrated model performance and transferability benchmarks

Additionally, an independent one-sided t-test was conducted to determine whether the performance of the composition integrated model was significantly better than the established benchmark case. The results are shown in Table 5.14 below.

	Feed-fraction use-cases		
Base feed-fraction use-case	**Light leaning**	**Equimolar**	**Heavy leaning**
Light leaning	0.985	0.907	0.946
Equimolar	0.500	0.978	0.834
Heavy leaning	0.836	0.253	0.448

TABLE 5.14: P-values from the independent one-sided t-test comparing the model scores from the benchmark and composition integrated case

As can be seen from the p-values in Table 5.14, the integrated model does not outperform the benchmark for either the feed-fraction it was trained on or the feed-fractions it was not trained on. Furthermore, in Tables 5.15, 5.16, 5.17 and 5.18, the results of the switch training analysis are presented.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	16.80	17.63	27.79
Mean score	2.71	9.53	19.24
Min. score	-2.03	-0.09	0.68
Standard deviation	5.45	5.83	6.52

TABLE 5.15: Composition integrated switch training model performance (switching frequency: 1)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	12.75	17.75	25.69
Mean score	1.67	14.78	24.23
Min. score	-0.73	3.71	16.06
Standard deviation	2.72	3.55	2.56

TABLE 5.16: Composition integrated switch training model performance (switching frequency: 100)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	1.55	17.75	25.70
Mean score	1.54	17.27	25.69
Min. score	1.50	14.60	25.65
Standard deviation	0.02	1.03	0.01

TABLE 5.17: Composition integrated switch training model performance (switching frequency: 500)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	5.47	16.32	27.66
Mean score	1.80	6.38	16.94
Min. score	-0.78	-0.59	6.74
Standard deviation	2.41	6.42	5.54

TABLE 5.18: Composition integrated switch training model performance (switching frequency: 1 (random))

Another independent one-sided t-test was employed to determine whether the composition integration improved the overall model performance and its ability to transfer obtained knowledge. P-values below 0.10 reflects that the composition integration produced significantly higher model scores than the benchmark case.

	Feed-fraction use-cases		
Switching frequency	**Light leaning**	**Equimolar**	**Heavy leaning**
1	1.000	1.000	1.000
100	0.989	0.999	1.000
500	0.527	0.971	1.000
1 (random)	1.000	1.000	1.000

TABLE 5.19: P-values from the independent one-sided t-test comparing the model scores from the benchmark and composition integrated switch training strategy

5.3.2 Boiling point integration

The methodologies applied for testing the boiling point integrated model remained consistent with the approach employed in the previous composition integrated case. In Tables 5.20, 5.21 and 5.22, the results from the boiling point integrated model performance and transferability benchmarks are shown for each of the feed-fraction use-cases.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	1.67	3.65	12.65
Mean score	-0.61	-0.21	3.09
Min. score	-1.93	-1.20	-0.64
Standard deviation	0.73	1.01	3.48

TABLE 5.20: Light leaning boiling point integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	18.01	17.75	25.70
Mean score	2.67	14.80	22.99
Min. score	-0.32	-0.62	9.51
Standard deviation	4.98	5.47	4.87

TABLE 5.21: Equimolar leaning boiling point integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	21.04	19.49	29.29
Mean score	3.11	11.82	21.00
Min. score	-1.50	-0.26	7.80
Standard deviation	6.04	7.01	6.04

TABLE 5.22: Heavy leaning boiling point integrated model performance and transferability benchmarks

Again, an independent one-sided t-test was conducted to determine whether the performance of the boiling point integrated model was significantly better than the benchmark case. The results are shown in Table 5.23 below.

	Feed-fraction use-cases		
Base feed-fraction use-case	**Light leaning**	**Equimolar**	**Heavy leaning**
Light leaning	1.000	1.000	1.000
Equimolar	0.169	0.985	0.987
Heavy leaning	0.136	0.999	0.999

TABLE 5.23: P-values from the independent one-sided t-test comparing the model scores from the benchmark and boiling point integrated case

In Tables 5.24, 5.25, 5.26 and 5.27, the results of the switch training analysis are presented.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	6.03	12.09	16.06
Mean score	0.21	2.36	7.44
Min. score	-1.68	-1.37	-0.44
Standard deviation	2.11	3.52	4.44

TABLE 5.24: Boiling point integrated switch training model performance (switching frequency: 1)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	20.28	16.25	28.54
Mean score	16.00	6.57	16.00
Min. score	-0.37	-0.99	-0.37
Standard deviation	9.36	5.60	9.36

TABLE 5.25: Boiling point integrated switch training model performance (switching frequency: 100)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	9.07	5.29	19.28
Mean score	2.49	0.93	8.36
Min. score	-2.01	-1.35	-0.98
Standard deviation	3.59	2.09	5.37

TABLE 5.26: Boiling point integrated switch training model performance (switching frequency: 500)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	4.67	8.41	18.68
Mean score	-0.26	1.88	6.30
Min. score	-2.31	-1.17	-0.40
Standard deviation	1.56	2.80	4.38

TABLE 5.27: Boiling point integrated switch training model performance (switching frequency: 1 (random))

Last, an independent one-sided t-test was employed to determine whether the composition integration improved the overall model performance and its ability to transfer obtained knowledge. P-values below 0.10 reflect that the composition integration produced significantly higher model scores than the benchmark case.

	Feed-fraction use-cases		
Switching frequency	**Light leaning**	**Equimolar**	**Heavy leaning**
1	1.000	1.000	1.000
100	1.000	1.000	1.000
500	1.000	1.000	1.000
1 (random)	1.000	1.000	1.000

TABLE 5.28: P-values from the independent one-sided t-test comparing the model scores from the benchmark and boiling point integrated switch training strategy

5.3.3 Composition and boiling point integration

For the final property integrated case, the model was supplied with information about both the feed-composition and pure component boiling points. In Tables 5.29, 5.30 and 5.31, the results from the composition and boiling point integrated model performance and transferability benchmarks are shown for each of the feed-fraction use-cases.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	10.27	11.88	21.96
Mean score	2.00	2.28	11.24
Min. score	-1.62	-0.73	-0.35
Standard deviation	3.68	4.53	6.03

TABLE 5.29: Light leaning composition and boiling point integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	16.84	17.69	29.17
Mean score	1.99	11.67	22.39
Min. score	-2.09	-0.85	10.44
Standard deviation	4.59	6.91	5.79

TABLE 5.30: Equimolar leaning composition and boiling point integrated model performance and transferability benchmarks

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	16.89	19.38	29.21
Mean score	2.73	12.92	22.26
Min. score	-0.86	-0.22	-0.97
Standard deviation	5.21	5.42	6.69

TABLE 5.31: Heavy leaning composition and boiling point integrated model performance and transferability benchmarks

Additionally, an independent one-sided t-test was conducted to determine whether the performance of the composition and boiling point integrated model was significantly better than the benchmark case. The results are shown in Table 5.32 below.

	Feed-fraction use-cases		
Base feed-fraction use-case	**Light leaning**	**Equimolar**	**Heavy leaning**
Light leaning	1.000	1.000	0.398
Equimolar	0.341	0.999	0.989
Heavy leaning	0.168	0.999	0.981

TABLE 5.32: P-values from the independent one-sided t-test comparing the model scores from the benchmark and composition and boiling point integrated case

In Tables 5.33, 5.34, 5.35 and 5.36, the results of the switch training analysis are presented.

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	5.07	5.66	16.54
Mean score	0.15	2.00	7.70
Min. score	-1.77	-0.64	-1.17
Standard deviation	1.88	2.27	4.83

TABLE 5.33: Composition and boiling point integrated switch training model performance (switching frequency: 1)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	20.86	19.27	29.14
Mean score	2.20	6.78	17.15
Min. score	-2.41	-1.37	-0.23
Standard deviation	5.50	6.47	7.06

TABLE 5.34: Composition and boiling point integrated switch training model performance (switching frequency: 100)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	3.77	3.13	13.20
Mean score	-0.08	-0.06	4.07
Min. score	-1.51	-1.01	-0.59
Standard deviation	1.32	0.90	4.22

TABLE 5.35: Composition and boiling point integrated switch training model performance (switching frequency: 500)

	Feed-fraction use-cases		
	Light leaning	**Equimolar**	**Heavy leaning**
Max. score	20.79	17.12	28.63
Mean score	2.77	4.32	14.58
Min. score	-1.72	-0.58	7.02
Standard deviation	5.84	6.05	5.65

TABLE 5.36: Composition and boiling point integrated switch training model performance (switching frequency: 1 (random))

Similar to the composition integrated case, an independent one-sided t-test was employed as the final analysis step to determine whether the combined composition and boiling point integration improved the overall model performance and its ability to transfer obtained knowledge. As was the case before, P-values below 0.10 reflect that the current integrated model produced significantly higher model scores than the benchmark case.

	Feed-fraction use-cases		
Switching frequency	**Light leaning**	**Equimolar**	**Heavy leaning**
1	1.000	1.000	1.000
100	1.000	1.000	1.000
500	1.000	1.000	1.000
1 (random)	1.000	1.000	1.000

TABLE 5.37: P-values from the independent one-sided t-test comparing the model scores from the benchmark and composition and boiling point integrated switch training strategy

5.3.4 Property integration result analysis

First, based on the p-values results listed in Tables 5.14, 5.19, 5.23, 5.28, 5.32 and 5.37, it was concluded that, in the current state of the model architecture, no version of the model with additional chemical composition and property integration outperforms the established benchmark performance. However, on occasion, versions of the property integrated model outperformed or reached similar results as the benchmark on one or several of the feed-fraction use-cases.

When the results from the composition and boiling point integrated cases were compared, it was found that the composition integrated model was more likely to generate flowsheets with higher model scores. To confirm, another independent one-sided t-test was run. If both the integrated model cases were compared, it was found that the composition integrated model performed better than the boiling point integrated one. Furthermore, it was found that the combined composition and boiling point integrated model performed worse than both the composition integrated model and the benchmark model. Therefore, it was concluded that supplying the RL agent with information regarding the composition or boiling points using the current fingerprint concatenation method produced worse model scores.

Further, similar observations could be made for the switch training analyses that were conducted for the property integrated models as were done for the model without any integration. Again, it was found that for higher switching frequencies the model did not consistently generate the same flowsheet. However, when lower switching frequencies were used, the same results were found as for the original transferability benchmarking. The same conclusion as previously was reached; it is assumed that exposing the agent to each feed-fraction for a smaller amount of time prevented the agent from overfitting on the specific feed-fraction use-case.

5.4 Main findings

- Even when tasked with generating flowsheets for the same feed-fraction use-case it was trained on, the model consistently generated flowsheets that did not reach the same model score as the found optimal flowsheets.
- Lower switching frequencies resulted in higher consistency in finding the same flowsheets as during the initial transferability result analysis. Higher switching frequencies also resulted in worse average model performance, but on some occasions did result in model scores that were higher than the established benchmarks.
- Additional chemical composition and property integration via the direct fingerprint concatenation method resulted in worse model performances compared to the established benchmarks.

Chapter 6

Conclusion

This study aimed to provide valuable insights and methodologies applicable to broader industrial contexts. A comprehensive understanding of the underlying physical and chemical processes by an RL agent is vital in turning the the proposed methodology into a viable design tool in the field of process synthesis. Therefore, throughout this research, several attempts were made to improve the ability of a RL agent to understand the underlying physical and chemical processes involved in a simplified distillation separation process.

First, attempts were made to improve the generality of the model by training the RL agent on multiple feed-fraction use-cases instead of exclusively on one. The findings show that the average switch training model performed worse than a benchmark that was established by training a model on one specific feed-fraction use-case and subsequently assessing the model's ability to generate viable flowsheets based on the same feed-fraction use-case it was trained on. According to the p-values originating from the one-sided t-tests comparing the switch training results and the benchmarks, none of the switch training models outperformed the benchmark. However, if the maximum achieved model scores are compared, it is found that in some cases the same or better model scores as the benchmark are achieved.

In a second attempt to improve the generality of the model, information regarding the actual processes and chemical properties of the components in the streams was supplied to the RL agent. This was done by taking the values of a property and subsequently concatenating them to the end of a flowsheet fingerprint, which described the specifics of a flowsheet in a way that an RL agent could read and act upon it. First, the model was informed about the composition of the components in the entry stream. Second, the model was given information regarding the pure component boiling points of the components at atmospheric pressure. By comparing the benchmark performance and the performances of the integrated models, it was found that the integration of additional chemical composition and property information through the direct fingerprint concatenation method appeared to have a detrimental effect on the model's performance. However, similar to the switch training case, on some occasions, similar or better model scores were obtained in comparison to the benchmark model performance.

In conclusion, the findings highlight several crucial aspects regarding the model's performance when training on multiple feed-fraction use-cases and when supplied with information regarding the actual processes and chemical properties of the components in the streams. It was hypothesized that both the switch training methodology and integration of chemical composition and property information would significantly improve the performance of the model on feed-fraction use-cases it was not explicitly trained on. Despite, in both cases the average model performance being worse than the benchmark, on some occasions, the integrated model outperformed the benchmark and generated a flowsheet with a higher model score. As in practical process synthesis

situations only a single flowsheet is required for each specific use-case and on some occasions better model scores than the benchmark were achieved, this research could still add value to the field of process synthesis. However, to ensure an RL agent comparable to the one discussed during this research could become a viable tool in the field of process synthesis, more consistent results are required. Therefore, additional research should be conducted into the employed property integration methodology so that the optimal parameters and methodology for integration can be found. With this fusion of RL and specific process engineering knowledge, the efficiency of industrial processes can be increased greatly, contributing significantly to the broader goals of sustainability, innovation, and responsible resource management.

Chapter 7

Discussion and outlook

Recently, RL has been utilized to generate sensible flowsheet designs for several applications, including distillation trains [25] and other simplified reaction and separation problems [26]. However, the employed RL agents had to be re-trained for every use-case they were applied to, as a lack of knowledge about physics-related concepts resulted in an inability to transfer the gained information to new processes. To increase the viability of RL as a tool in the field of process synthesis, an RL agent should be able to perform well not only on the training environment itself, but also across similar, but not identical, environments. Therefore, attempts were made to improve the generality of the model both by training on multiple feed-compositions simultaneously as well as by supplying the RL agent with additional information about the chemical feed-composition and the boiling points of the components. The RL agent was informed about the chemical compositions and component properties via direct concatenation onto the flowsheet fingerprint that was generated as a representation of the state of the environment. Finally, this research aimed to determine to what degree switch training and integrating additional properties into an existing RL agent structure improved the ability to transfer learned policy information to new processes.

First, benchmark performances were established by training the agent on one feed-fraction use-case without any additional property integration and subsequently tasking the model with generating flowsheets for specific Light leaning, Equimolar, Heavy leaning feed-compositions. After analysis, it was found that lower feed-fraction switching frequencies resulted in higher consistency in finding the same flowsheets as during the initial transferability result analysis. Conversely, higher switching frequencies resulted in a worse average model performances, but on some occasions did result in model scores that were higher than the benchmark measurements. Further, it was found that the additional chemical composition and property integration via the direct fingerprint concatenation method resulted in worse model performances compared to the established benchmarks. The data collection procedure employed for obtaining these results was thorough. However, given the stochastic nature of RL models, conducting additional measurements could further increase the reliability of the results. Overall, it was found that additional property integration resulted in less consistent model performances, but, on occasion, higher model scores than the benchmark performances. Therefore, it was concluded that to reach more consistent results other property integration methods than the direct fingerprint concatenation should be investigated.

Existing literature on methods for property integration in RL is limited, especially since the specifics of the current flowsheet fingerprint-based model need to be taken into account. However, a technique called reward shaping, which operates by providing feedback through specific rewards informed by prior knowledge to guide the learning process, could offer a solution [57]. A research paper by Radaideh et al. demonstrated that by integrating expert knowledge in the form of game rules and by effectively exploring the search space via reward shaping, RL algorithms were able

to optimize nuclear fuel assemblies more effectively than standard stochastic optimization methods [58]. Shaping the reward function makes targeted optimization possible. This methodology could be particularly beneficial in complex domains where the RL agent cannot explore the entire environment before taking an action [57].

Therefore, for further research, the author recommends to investigate the prospect of applying reward shaping as a method to steer the RL agent to focus its learning onto the underlying physical and chemical processes that exist within chemical processes. Once again, it is suggested to employ the same distillation column model and consider the same paraffinic distillation use-case. In that case, obtained results could easily be compared to the current research as well as to previous work by human actors, such as Jobson [50], Stephanopoulos et al. [48] or Porter and Momoh [49]. Further, the separation of paraffinic compounds is very suitable for distillation separation use-cases as their separation is relatively ideal and additionally well-researched. Then, once reward shaping as a method is proven to work on the current flowsheet fingerprint-based model, additional process units and third-level decisions can be added. Increasing the complexity of the domain, will mean changes are required to the way reward shaping occurs within the model. However, as the concept was proven on a less complex environment, scaling up should be feasible.

Even though the examined methods turned out to be unable to steer the RL agent towards learning general physics-related concepts, promising instances occurred where altered models outperformed the established benchmarks. Especially, the switch training methodology showed promise as technique to steer agent learning by exposing the agent to the same environment for a limited number of episodes at a time. Therefore, it is recommended that future research also experiments with combining any new technique with the switch training methodology as described during this research.

In the end, the current study contributed significantly to guiding research in the field of RL in process synthesis. The implementation and testing of switch training at multiple frequencies as well as several types of property integration have resulted in the establishment of an exceptional benchmark. This benchmark serves as a reference point to which future implementations such as the reward shaping methodology described above could be compared. Additionally, the testing methods that were devised during the current research could be reused to navigate future research projects.

Currently, the model is not suitable for application in practical process synthesis situations. Further, if the proposed methodologies and adaptations turn out to be effective, it is important that the working and intricacies of the model are well documented prior to application in the process synthesis field. Practitioners should be able to understand the underlying mechanisms and decision-making processes of the model to ensure effective application of the technique and mitigate the risks associated with any biases the model could have learned.

To conclude, instances in which property integrated models surpassed benchmark performances in terms of model scores were encouraging and with additional research and the exploration of different property integration methodologies, it is believed that RL agents could be used as a viable design tool in the field of process synthesis. When an adequate methodology for property integration is found, the efficiency of designing chemical processes will be greatly increased, thereby, contributing to the development of sustainable solutions in the field of CE. Finally, the bright future of applied RL not only suggests transformative advancements and breakthroughs in the CE and process synthesis fields, but also offers many possibilities to revolutionize various other fields, where agents can be taught to perform complex tasks by learning from experience and feedback.

Bibliography

[1] M. Campbell, A. Hoane, and F. Hsu, "Deep Blue," *Artificial Intelligence*, vol. 134, no. 1, pp. 57–83, 2002. DOI: 10.1016/S0004-3702(01)00129-1.

[2] L. Greenemeier, "20 Years after Deep Blue: How AI has advanced since conquering chess," *Scientific American*, 2017, [Accessed: 23-03-2023]. [Online]. Available: www.scientificamerican.com/article/20-years-after-deep-blue-how-ai-has-advanced-since-conquering-chess/.

[3] C. Lee, M. Wang, S. Yen, *et al.*, "Human vs. computer go: Review and prospect [discussion forum]," *IEEE Computational Intelligence Magazine*, vol. 11, no. 3, pp. 67–72, 2016. DOI: 10.1109/MCI.2016.2572559.

[4] Z. Zhang, "When doctors meet with AlphaGo: Potential application of machine learning to clinical medicine," *Annals of Translational Medicine*, vol. 4, no. 6, pp. 125–125, 2016. DOI: 10.21037/atm.2016.03.25.

[5] L. Oliveira, R. Dias, C. Rebello, *et al.*, "Artificial Intelligence and cyber-physical systems: A review and perspectives for the future in the chemical industry," *Artificial Intelligence*, vol. 2, no. 3, pp. 429–443, 2021. DOI: 10.3390/ai2030027.

[6] M. Reis and P. Saraiva, "Data-centric process systems engineering for the chemical industry 4.0," in *Systems Engineering in the Fourth Industrial Revolution*. John Wiley & Sons, Ltd, 2019, ch. 6, pp. 137–159. DOI: 10.1002/9781119513957.ch6.

[7] L. Chiang, B. Braun, Z. Wang, and I. Castillo, "Towards artificial intelligence at scale in the chemical industry," *AIChE Journal*, vol. 68, no. 6, e17644, 2022. DOI: 10.1002/aic.17644.

[8] N. Misra, Y. Dixit, A. Al-Mallahi, M. Bhullar, R. Upadhyay, and A. Martynenko, "IoT, big data and artificial intelligence in agriculture and food industry," *IEEE Internet of Things Journal*, vol. 9, no. 9, pp. 6305–6324, 2022. DOI: 10.1109/JIOT.2020.2998584.

[9] V. V. Venkatasubramanian, "The promise of artificial intelligence in chemical engineering: Is it here, finally?" *AIChE Journal*, vol. 65, no. 2, pp. 466–478, 2019. DOI: 10.1002/aic.16489.

[10] M. Taleb-Berrouane and H. Pasman, "Chapter 15 - Integrated dynamic risk management in process plants," in *Methods to Assess and Manage Process Safety in Digitalized Process System*, ser. Methods in Chemical Process Safety, F. Khan, H. Pasman, and M. Yang, Eds., vol. 6, Elsevier, 2022, pp. 525–560. DOI: 10.1016/bs.mcps.2022.05.006.

[11] F. Khan, S., and S. Ahmed, "Methods and models in process safety and risk management: Past, present and future," *Process Safety and Environmental Protection*, vol. 98, pp. 116–147, 2015. DOI: 10.1016/j.psep.2015.07.005.

[12] B. Mahesh, "Machine Learning algorithms - A review," *International Journal of Science and Research (IJSR)*, vol. 9, no. 1, pp. 381–386, 2018. DOI: 10.21275/ART20203995.

[13] I. Udousoro, "Machine Learning: A review," *Semiconductor Science and Information Devices*, vol. 2, no. 2, pp. 381–386, 2020. DOI: 10.30564/ssid.v2i2.1931.

[14] M. Jordan and T. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015. DOI: 10.1126/science.aaa8415.

[15] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107298019.

[16] Y. Li, "Deep reinforcement learning," *arXiv*, 2018. DOI: 10.48550/arXiv.1810.06339.

[17] R. Nian, J. Liu, and B. Huang, "A review on reinforcement learning: Introduction and applications in industrial process control," *Computers & Chemical Engineering*, vol. 139, p. 106 886, 2020. DOI: 10.1016/j.compchemeng.2020.106886.

[18] V. François-Lavet, P. Henderson, R. Islam, M. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Foundations and Trends in Machine Learning*, vol. 11, no. 3-4, pp. 219–354, 2018. DOI: 10.1561/2200000071.

[19] O. Dressler, P. Howes, J. Choo, and A. deMello, "Reinforcement learning for dynamic microfluidic control," *ACS Omega*, vol. 3, no. 8, pp. 10 084–10 091, 2018. DOI: 10.1021/acsomega.8b01485.

[20] K. Powell, D. Machalek, and T. Quah, "Real-time optimization using reinforcement learning," *Computers & Chemical Engineering*, vol. 143, p. 107 077, 2020. DOI: 10.1016/j.compchemeng.2020.107077.

[21] B. Rolf, I. Jackson, M. Müller, S. Lang, T. Reggelin, and D. Ivanov, "A review on reinforcement learning algorithms and applications in supply chain management," *International Journal of Production Research*, pp. 1–29, 2022. DOI: 10.1080/00207543.2022.2140221.

[22] T. Lan and Q. An, "Discovering catalytic reaction networks using deep reinforcement learning from first-principles," *Journal of the American Chemical Society*, vol. 143, no. 40, pp. 16 804–16 812, 2021. DOI: 10.1021/jacs.1c08794.

[23] Z. Zhou, X. Li, and R. Zare, "Optimizing chemical reactions with deep reinforcement learning," *ACS Central Science*, vol. 3, no. 12, pp. 1337–1344, 2017. DOI: 10.1021/acscentsci.7b00492.

[24] D. Adams, D. Oh, D. Kim, C. Lee, and M. Oh, "Deep reinforcement learning optimization framework for a power generation plant considering performance and environmental issues," *Journal of Cleaner Production*, vol. 291, p. 125 915, 2021. DOI: 10.1016/j.jclepro.2021.125915.

[25] L. Midgley, "Deep reinforcement learning for process synthesis," *arXiv*, 2020. DOI: 10.48550/arXiv.2009.13265.

[26] L. Stops, R. Leenhouts, Q. Gao, and A. Schweidtmann, "Flowsheet synthesis through hierarchical reinforcement learning and graph neural networks," *arXiv*, 2022. DOI: 10.48550/arXiv.2207.12051.

[27] A. Khan and A. Lapkin, "Searching for optimal process routes: A reinforcement learning approach," *Computers & Chemical Engineering*, vol. 141, p. 107 027, 2020. DOI: 10.1016/j.compchemeng.2020.107027.

[28] Q. Göttl, D. Grimm, and J. Burger, "Automated synthesis of steady-state continuous processes using reinforcement learning," *Frontiers of Chemical Science*, vol. 16, no. 2, pp. 288–302, 2022. DOI: 10.1007/s11705-021-2055-9.

[29] G. Towler and R. Sinnott, "Chapter 4 - process simulation," in *Chemical Engineering Design*, G. Towler and R. Sinnott, Eds., Second edition, Boston: Butterworth-Heinemann, 2013, pp. 161–250. DOI: 10.1016/B978-0-08-096659-5.00004-3.

[30] G. Towler and R. Sinnott, "Chapter 12 - optimization in design," in *Chemical Engineering Design*, G. Towler and R. Sinnott, Eds., Second edition, Boston: Butterworth-Heinemann, 2013, pp. 525–553. DOI: 10.1016/B978-0-08-096659-5.00012-2.

[31] A. Coker, "Chapter 1 - process planning, scheduling and flowsheet design," in *Ludwig's Applied Process Design for Chemical and Petrochemical Plants*, A. Coker, Ed., Fourth edition, Burlington: Gulf Professional Publishing, 2007, pp. 1–68. DOI: 10.1016/B978-075067766-0/50008-7.

[32] P. Fantke, C. Cinquemani, P. Yaseneva, *et al.*, "Transition to sustainable chemistry through digitalization," *Chemistry*, vol. 7, no. 11, pp. 2866–2882, 2021. DOI: 10.1016/j.chempr.2021.09.012.

[33] S. Sachio, M. Mowbray, M. Papathanasiou, E. del Rio-Chanona, and P. Petsagkourakis, "Integrating process design and control using reinforcement learning," *Chemical Engineering Research and Design*, vol. 183, pp. 160–169, 2022. DOI: 10.1016/j.cherd.2021.10.032.

[34] S. van Kalmthout, L. Midgley, and M. Franke, "Synthesis of separation processes with reinforcement learning," *arXiv*, 2022. DOI: 10.48550/arXiv.2211.04327v1.

[35] R. Glatt, F. Da Silva, and A. Costa, "Towards knowledge transfer in deep reinforcement learning," in *2016 5th Brazilian Conference on Intelligent Systems (BRACIS)*, 2016, pp. 91–96. DOI: 10.1109/BRACIS.2016.027.

[36] L. Helleckes, J. Hemmerich, W. Wiechert, E. von Lieres, and A. Grünberger, "Machine learning in bioprocess development: From promise to practice," *Trends in Biotechnology*, 2022. DOI: 10.1016/j.tibtech.2022.10.010.

[37] M. Mahmoudabadbozchelou, G. Karniadakis, and S. Jamali, "nn-PINNs: Non-Newtonian physics-informed neural networks for complex fluid modeling," *Soft Matter*, vol. 18, pp. 172–185, 1 2022. DOI: 10.1039/D1SM01298C.

[38] K. Weiss, T. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, 9 2016. DOI: 10.1186/s40537-016-0043-6.

[39] M. Mohammed, M. Khan, and E. Bashier, *Machine Learning Algorithms and Applications*. CRC Press, 2017, ISBN: 978-1-119-76924-8.

[40] I. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, 2021. DOI: 10.1007/s42979-021-00592-x.

[41] N. Li, M. Shepperd, and Y. Guo, "A systematic review of unsupervised learning techniques for software defect prediction," *arXiv*, 2019. DOI: 10.48550/arXiv.1907.12027.

[42] R. Sutton and A. Barto, *Reinforcement Learning: An introduction*, Second edition. The MIT Press, 2020. [Online]. Available: www.incompleteideas.net/book/RLbook2020.pdf.

[43] F. Yalaoui and V. Uc-Cetina, "A novel reinforcement learning architecture for continuous state and action spaces," *Advances in Artificial Intelligence*, vol. 2013, p. 492 852, 2013. DOI: 10.1155/2013/492852.

[44] A. Zanette, M. Wainwright, and E. Brunskill, "Provable benefits of actor-critic methods for offline reinforcement learning," *arXiv*, 2021. DOI: 10.48550/arXiv.2108.08812.

[45] A. Schweidtmann, J. Rittig, A. König, M. Grohe, A. Mitsos, and M. Dahmen, "Graph neural networks for prediction of fuel ignition quality," *Energy & Fuels*, vol. 34, no. 9, pp. 11 395–11 407, 2020. DOI: 10.1021/acs.energyfuels.0c01533.

[46] Z. Fan, R. Su, W. Zhang, and Y. Yu, "Hybrid actor-critic reinforcement learning in parameterized action space," *arXiv*, 2019. DOI: 10.48550/arXiv.1903.01344.

[47] J. Schulman, P. Moritz, M. Jordan, and P. Abbeel, "High-dimenesional continuous control using generalized advantage estimation," *arXiv*, 2015. DOI: 10.48550/arXiv.1506.02438.

[48] G. Stephanopoulos, B. Linnhoff, and A. Sophos, "Synthesis of heat integrated sequences.," 74, 1982, pp. 111–130.

[49] K. Porter and S. Momoh, "Finding the optimum sequence of distillation columns - an equation to replace the "rules of thumb" (heuristics)," *The Chemical Engineering Journal*, vol. 46, no. 3, pp. 97–108, 1991. DOI: 10.1016/0300-9467(91)87001-Q.

[50] M. Jobson, "Short-cut evaluation of distillation sequences," *Computers & Chemical Engineering*, vol. 21, S553–S557, 1997. DOI: 10.1016/S0098-1354(97)87560-6.

[51] M. Koçak, "Simple, robust, and fast iterative solution of underwood's equation for minimum reflux," *Chemical Engineering Research and Design*, vol. 89, no. 2, pp. 197–205, 2011. DOI: 10.1016/j.cherd.2010.05.006.

[52] J. McCormick, "A correlation for distillation stages and reflux," *Chemical Engineering*, vol. 95, no. 13, pp. 75–76, 1988.

[53] R. Davis, "Gilliland's correlation: A case study in regression analysis," *Chemical Engineering Education*, vol. 54, no. 4, pp. 213–221, 2020.

[54] R. Smith, *Chemical Process Design and Integration*, Second edition. Wiley, 2005.

[55] W. Seider, *Product and process design priciples*, Third edition. Wiley, 2010.

[56] International Energy Agency (IEA), *World Energy Outlook 2019*. IEA, 2019. [Online]. Available: www.iea.org/reports/world-energy-outlook-2019.

[57] A. Laud, "Theory and application of reward shaping in reinforcement learning," AAI3130966, Ph.D. dissertation, USA, 2004.

[58] M. Radaideh, I. Wolverton, J. Joseph, *et al.*, "Physics-informed reinforcement learning optimization of nuclear assembly design," *Nuclear Engineering and Design*, vol. 372, p. 110 966, 2021. DOI: 10.1016/j.nucengdes.2020.110966.

Appendix A

Optimal model learning curves

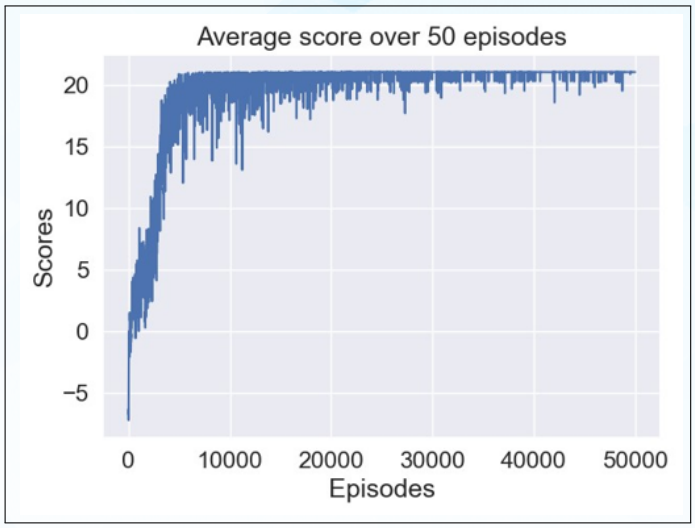

FIGURE A.1: Performance benchmark analysis learning curve for the Light leaning feed-fraction use-case

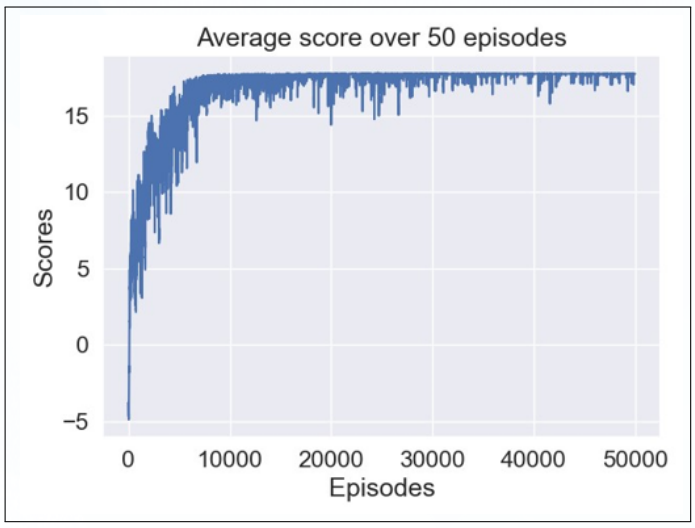

FIGURE A.2: Performance benchmark analysis learning curve for the Equimolar leaning feed-fraction use-case

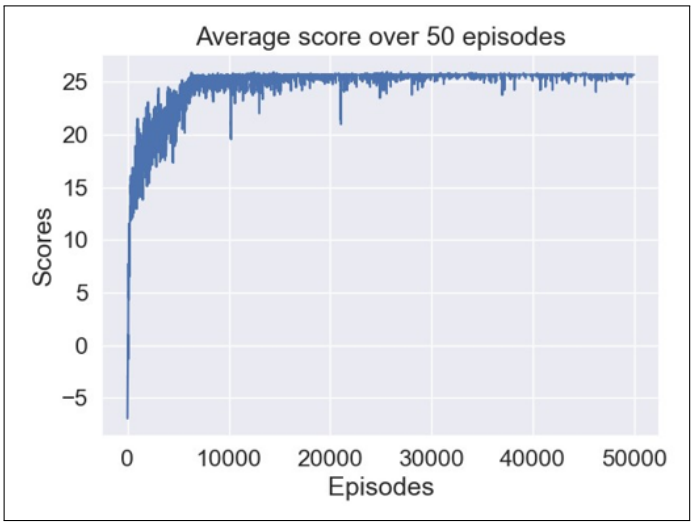

FIGURE A.3: Performance benchmark analysis learning curve for the Heavy leaning feed-fraction use-case

Appendix B

Initial model performance benchmarking measurements

	Feed-fraction use-cases		
Measurement number	**Light leaning**	**Equimolar**	**Heavy leaning**
1	21.04	17.75	25.70
2	21.04	17.75	25.68
3	21.04	17.75	25.70
4	21.03	17.75	25.70
5	21.04	17.75	25.70
6	21.04	17.72	25.70
7	21.04	17.75	25.70
8	21.02	17.74	25.70
9	21.04	17.75	25.57
10	21.03	17.75	25.70
11	21.04	17.75	25.70
12	21.04	17.75	25.70
13	21.04	17.75	25.70
14	21.04	17.75	25.70
15	21.04	17.75	25.70
16	21.03	17.75	25.70
17	21.04	17.75	25.69
18	21.04	17.75	25.70
19	21.04	17.75	25.70
20	21.04	17.75	25.70

TABLE B.1: Initial model performance benchmarking measurements

Appendix C

Transferability benchmarking measurements

	Feed-fraction use-cases	
Measurement number	**Equimolar**	**Heavy leaning**
1	6.81	10.88
2	6.81	10.88
3	6.79	10.89
4	6.81	10.88
5	6.81	10.88
6	6.81	10.89
7	6.81	10.88
8	6.81	10.88
9	6.80	10.88
10	6.78	10.89
11	6.87	10.89
12	6.81	10.89
13	6.81	10.89
14	6.80	10.89
15	6.81	10.88
16	6.81	10.89
17	6.81	10.86
18	6.81	10.86
19	6.77	10.88
20	6.80	10.89

TABLE C.1: Light leaning transferability benchmark measurements

	Feed-fraction use-cases	
Measurement number	**Light leaning**	**Heavy leaning**
1	1.55	25.70
2	1.55	25.70
3	1.55	25.68
4	1.55	25.70
5	1.55	25.70
6	1.55	25.70
7	1.55	25.70
8	1.55	25.69
9	1.55	25.70
10	1.55	25.70
11	1.55	25.70
12	1.55	25.70
13	1.55	25.70
14	1.55	25.70
15	1.55	25.70
16	1.55	25.70
17	1.54	25.67
18	1.55	25.70
19	1.55	25.70
20	1.54	25.70

TABLE C.2: Equimolar transferability benchmark measurements

	Feed-fraction use-cases	
Measurement number	**Light leaning**	**Equimolar**
1	1.55	17.73
2	1.55	17.75
3	1.52	17.75
4	1.55	17.74
5	1.55	17.74
6	1.55	17.75
7	1.55	17.74
8	1.53	17.73
9	1.54	17.74
10	1.55	17.75
11	1.55	17.75
12	1.54	17.75
13	1.55	17.75
14	1.55	17.75
15	1.55	17.62
16	1.50	17.74
17	1.55	17.74
18	1.55	17.75
19	1.55	17.72
20	1.55	17.73

TABLE C.3: Heavy leaning transferability benchmark measurements