Tuning the Molecular Weight Distribution from Atom Transfer Radical Polymerization Using Actor Critic Methods

Haichen Li

May 2018

Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Geoffrey Gordon, Advisor
David Yaron

Submitted in partial fulfillment of the requirements for the degree of Master of Science.

May 5, 2018

Abstract

We devise a novel technique to control the shape of polymer molecular weight distributions (MWDs) in atom transfer radical polymerization (ATRP). This technique makes use of recent advances in both simulation-based, model-free reinforcement learning (RL) and the numerical simulation of ATRP. A simulation of ATRP is built that allows an RL controller to add chemical reagents throughout the course of the reaction. The RL controller incorporates fully-connected and convolutional neural network architectures and bases its decision upon the current status of the ATRP reaction. The initial, untrained, controller leads to ending MWDs with large variability, allowing the RL algorithm to explore a large search space. When trained using an actor-critic algorithm, the RL controller is able to discover and optimize control policies that lead to a variety of target MWDs. The target MWDs include Gaussians of various width, and more diverse shapes such as bimodal distributions. The learned control policies are robust and transfer to similar but not identical ATRP reaction settings, even under the presence of simulated noise. We believe this work is a proof-of-concept for employing modern artificial intelligence techniques in the synthesis of new functional polymer materials.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Most current approaches to development of new materials follow a sequential, iterative process that requires extensive human labor to synthesize new materials and elucidate their properties and functions. Over the next decades, it seems likely that this inherently slow and labor intensive approach to chemical research will be transformed through the incorporation of new technologies originating from computer science, robotics, and advanced manufacturing.[7, 27] A central challenge is finding ways to use these powerful new technologies to guide chemical processes to desired outcomes.[65] Recent advances in reinforcement learning (RL) have enabled computing systems to guide vehicles through complex simulation environments,[57] and select moves that guide games such as Go and chess to winning conclusions.[91, 117, 118, 119] For chemical problems, RL has been used to generate candidate drug molecules in a *de novo* manner,[101, 106] and to optimize reaction conditions for organic synthesis.[152] This work investigates the benefits and challenges of using RL to guide chemical reactions towards specific synthetic targets. The investigation is done through computational experiments that use RL to control a simulated reaction system, where the simulation models the chemical kinetics present in the system.

1.1 Atom transfer radical polymerization

In this work, the simulated reaction system is that of atom transfer radical polymerization (ATRP).[42, 82, 84, 85] ATRP is among the mostly widely used and effective means to control the polymerization of a wide variety of vinyl monomers. ATRP allows the synthesis of polymers with predetermined molecular weights, narrow molecular weight distributions (MWDs),[25] and adjustable polydispersity.[35, 73, 74, 75, 76, 77, 105] The high degree of control allows the synthesis of various polymeric architectures [83] such as block copolymers,[12, 80, 81, 88] star polymers,[33, 68, 89] and molecular brushes.[34] Temporal and spatial control has also been applied in ATRP to further increase the level of control over the polymerization.[18, 108, 140, 141] More recently, chemists have been working on ways to achieve MWDs with more flexible forms,[13, 35] as this may provide a means to tailor mechanical and processability of the resulting plastics.[56]

In addition to its importance, ATRP is well suited to the computational experiments carried out here. The chemical kinetics of ATRP are shown schematically in Figure 1.1. Control of the

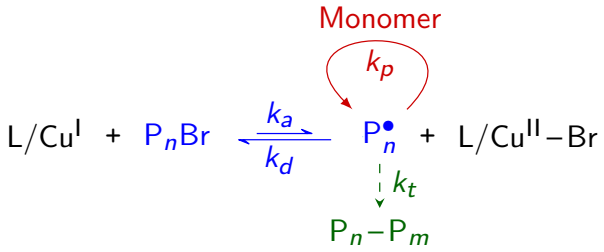

Figure 1.1: Reaction mechanism of ATRP. Polymer species include radical chains $P^{\bullet}_n$ and dormant chains P_nBr with reduced chain length n and chains that terminated through recombination P_n-P_m. L/Cu^{I} and L/Cu^{II}−Br are ATRP catalysts, where L represents the ligand. k_p, k_a, k_d, and k_t are kinetic rate constants for chain propagation, activation, deactivation, and termination, respectively.

polymerization process is related to the activation, k_a, and deactivation, k_d, reactions which interconvert dormant chains, P_nBr, and active, free radical chains, $P^{\bullet}_n$. The active chains grow in length through propagation reactions, k_p. The equilibrium between dormant and active chains can be used to maintain a low concentration of active chains, leading to more controlled growth and a reduction in termination reactions, k_t, that broaden the final MWD. These kinetics are sufficiently well understood[39, 127] that simulations provide reliable results.[24, 26, 59, 60, 107, 134, 135, 144, 151] It is also computationally feasible to carry out a large number of simulated reactions. Figure 1.2 shows how the MWD evolves in a single reaction simulation, which finishes in about 1 minute on a 2.4 GHz CPU core. MWDs will be shown as the fraction of polymer chains (vertical axis) with a specific reduced chain length (horizontal axis), where the reduced chain length refers to the number of monomers incorporated into the chain.

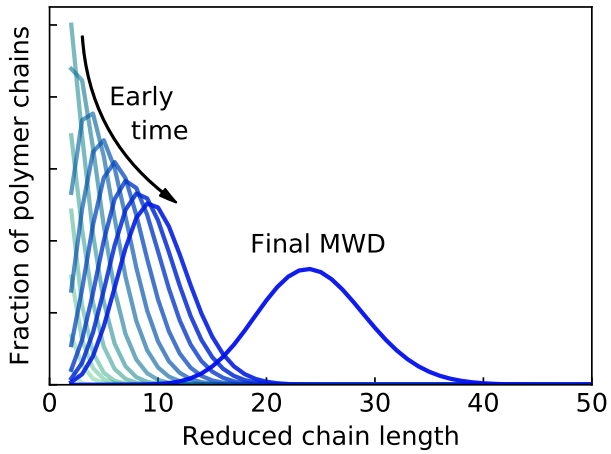

Figure 1.2: Evolution of polymer MWD in a simulated ATRP reaction.

1.2 Formulating controlled ATRP as a reinforcement learning problem

ATRP reactions can also be manipulated in a large variety of ways because of the multiple interacting chemical reactions, and the shape of the MWD provides a diverse set of targets. This makes the system a good choice for evaluating the degree to which RL can guide a chemical process to a desired synthetic target. ATRP reactions are typically carried out by creating an initial mixture of chemical reagents and keeping the temperature and other reaction conditions steady. However, a greater diversity of MWDs can be obtained by taking actions, such as adding chemical reagents, throughout the polymerization process.[35] Here, we use RL to decide which actions to take, based on the current state of the reaction system. In this manner, it is analogous to having a human continuously monitor the reaction and take actions that guide the system towards the target MWD. This use of a state-dependent decision process is a potential advantage of using RL. Consider an alternative approach in which the simulation is used to develop a protocol that specifies the times at which to perform various actions. Such a protocol is likely to be quite sensitive to the specific kinetic parameters used in the simulation. The RL controller may lower this sensitivity by basing its decisions on the current state of the reaction system. Below, the current state upon which the RL controller makes its decisions includes the current MWD. The controller is then expected to succeed provided the correct action to take at a given time depends primarily on the difference between the current MWD and the target MWD (Figure 1.2), as opposed to the specific kinetic parameters. Ideally, an RL algorithm trained on a simulated reaction may be able to succeed in the real laboratory with limited additional training, provided the simulated reaction behaves like the actual one. Such transfer from simulated to real-world reactions is especially important given the potentially large number of reaction trials needed for training, and the inherent cost of carrying out chemical experiments. In our computational experiments, we assess the sensitivity to the simulation parameters by including noise in both the kinetic parameters used in the simulation and in the states of the current reaction system.

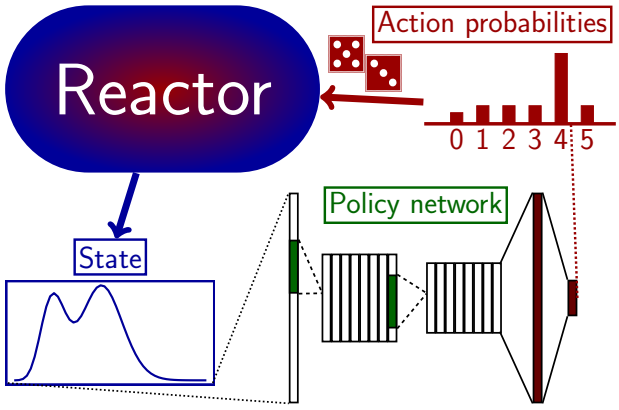

Figure 1.3: Flow chart showing how the policy network of the RL controller selects actions to apply to the simulated ATRP reactor.

Figure 1.3 provides a schematic view of the RL controller. The current state is fed into the RL controller (policy network), which produces a probability distribution for each of the

available actions. An action is then drawn from this probability distribution, and performed on the reactor. The design of the RL controller is inspired by recent advances in deep reinforcement learning,[2, 43, 67] which use neural networks for the policy network and other components. The combination of modern deep learning models, represented by convolutional neural networks,[22, 58, 61, 115] and efficient RL algorithms[37, 38] such as deep Q-learning,[69, 91, 132] proximal policy methods,[116] and asynchronous advantage actor-critic (A3C)[31, 92] has lead to numerous successful applications in control tasks with large state spaces.[27, 71, 110] The computational experiments presented here examine the use of modern deep reinforcement learning techniques to guide chemical synthesis of new materials.

1.3 Related works

There have been many studies that control the state and dynamics of chemical reactors based on classical control theory.[100] Model-based controllers,[6] some of which employ neural networks,[49] have been developed for a number of control tasks involving continuous stirred tank reactors,[3, 32, 70, 143, 147, 148] batch processes,[99, 121, 122] hydrolyzers,[72] bioreactors,[8, 15, 19] pH neutralization processes,[44, 79, 94, 98] strip thickness in steel-rolling mills,[113] and system pressure.[131] Model-free controllers trained through RL also exist for controlling chemical processes such as neutralization[125] and wastewater treatment[126] or chemical reactor valves.[20]

Due to its industrial importance, polymer synthesis has been a primary target for the development of chemical engineering controllers.[14] Some of these make use of neural networks to control the reactor temperature in the free radical polymerization of styrene.[48] McAfee et al. developed an automatic polymer molecular weight controller[87] for free radical polymerization. This controller is based on online molar mass monitoring techniques[30] and is able to follow a specific chain growth trajectory with respect to time by controlling the monomer flow rate in a continuous flow reactor. Similar online monitoring techniques have recently enabled controlling the modality of free radical polymerization products,[63] providing optimal feedback control to acrylamide-water-potassium persulfate polymerization reactors,[36] and monitoring multiple ionic strengths during the synthesis of copolymeric polyelectrolytes.[145] However, none of these works attempted to control the precise shape of polymer MWD shapes, nor did they use an artificial intelligence (AI) driven approach to design new materials. The significance of this work lies in that it is a first trial of building an AI agent that is trained *tabula rasa* to discover and optimize synthetic routes for human-specified, arbitrary polymer products with specific MWD shapes. Another novel aspect of the current work is the use of a simulation to train a highly-flexible controller, although the transfer of this controller to actual reaction processes, possibly achievable with modern transfer learning[5, 16, 102, 128, 139] and imitation learning techniques,[109, 123] is left to future work.

Chapter 2

Controlling ATRP Simulation

2.1 Simulate ATRP by solving ordinary differential equations

We select styrene ATRP as our simulation system. Simulation of styrene ATRP may be done by solving the ATRP chemical kinetics ordinary differential equations (ODEs) in Table 2.1,[66, 107, 137, 144] by method of moments,[153] or by Monte Carlo methods.[1, 95, 96, 104, 129, 130] This work directly solves the ODEs because this allows accurate tracking of the concentration of individual polymer chains while being more computationally efficient than Monte Carlo methods.

Table 2.1: ATRP kinetics equations. Cu^{I} and Cu^{II} stand for the ATRP activator and deactivator L/Cu^{I} and $L/Cu^{II}-Br$, respectively.

Monomer	$[\mathrm{M}]' = -k_p[\mathrm{M}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Activator	$[\mathrm{Cu^I}]' = k_d[\mathrm{Cu^{II}}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}] - k_a[\mathrm{Cu^I}]\sum_{i=1}^{N}[\mathrm{P_iBr}]$
Deactivator	$[\mathrm{Cu^{II}}]' = k_a[\mathrm{Cu^I}]\sum_{i=1}^{N}[\mathrm{P_iBr}] - k_d[\mathrm{Cu^{II}}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Dormant chains	$[\mathrm{P_nBr}]' = k_d[\mathrm{Cu^{II}}][\mathrm{P_n^\bullet}] - k_a[\mathrm{Cu^I}][\mathrm{P_nBr}],\ 1 \le n \le N$
Smallest radical	$[\mathrm{P_1^\bullet}]' = -k_p[\mathrm{M}][\mathrm{P_1^\bullet}] + k_a[\mathrm{Cu^I}][\mathrm{P_1Br}] - k_d[\mathrm{Cu^{II}}][\mathrm{P_1^\bullet}] - 2k_t[\mathrm{P_1^\bullet}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Other radicals	$[\mathrm{P_n^\bullet}]' = k_p[\mathrm{M}]([\mathrm{P_{n-1}^\bullet}] - [\mathrm{P_n^\bullet}]) + k_a[\mathrm{Cu^I}][\mathrm{P_nBr}] - k_d[\mathrm{Cu^{II}}][\mathrm{P_n^\bullet}]$ $- 2k_t[\mathrm{P_n^\bullet}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}],\ 2 \le n \le N$
Terminated chains	$[\mathrm{T_n}]' = \sum_{i=1}^{n-1} k_t[\mathrm{P_i^\bullet}][\mathrm{P_{n-i}^\bullet}],\ 2 \le n \le 2N$

In the ODEs of Table 2.1, M is monomer; $\mathrm{P_n^\bullet}$, $\mathrm{P_nBr}$, and $\mathrm{T_n}$ represent length-n radical chain, dormant chain, and terminated chain, respectively. $\mathrm{P_1Br}$ is also the initiator of radical polymerization. k_p, k_a, k_d, and k_t are propagation, activation, deactivation, and termination rate constants, respectively. N is the maximum allowed dormant/radical chain length in the numerical simulation. Consequently, the maximum allowed terminated chain length is $2N$, assuming

styrene radicals terminate via combination.[97] We set $N = 100$ in all ATRP simulations in this work. This number is sufficiently large for our purpose as the lengths of dormant or terminated chains do not exceed 75 or 150, respectively, in any of the simulations. We used a set of well-established rate constants based on experimental results of the ATRP of bulk styrene at 110 °C (383.15 K) using dNbpy as the ligand[85, 86, 103, 138]: $k_p = 1.6 \times 10^3$, $k_a = 0.45$, $k_d = 1.1 \times 10^7$, and $k_t = 10^8$ (units are $\mathrm{M}^{-1}\mathrm{s}^{-1}$). It was assumed the reactor remained at this temperature for the duration of the polymerization. Although the rate constants depend on the degree of polymerization,[41] we assumed the same rate constants for polymer chains with different lengths. This assumption does not bias the nature of ATRP qualitatively and has been practiced in almost all previous ATRP simulation research.[66, 107, 136, 137, 144] In some of our simulations, we altered the rate constants by up to $\pm 30\%$ to account for possible inaccuracies in the measurement of these values and other unpredictable situations such as turbulence in the reactor temperature. We employed the VODE[9, 11, 46, 47, 50] integrator implemented in SciPy 0.19 using a maximum internal integration step of 5000, which is sufficient to achieve final MWDs with high accuracy. We chose the "backward differentiation formulas" integration method because the ODEs are stiff.

In practice, styrene ATRP is close to an ideal living polymerization,[86, 103] with termination playing only a small role in establishing the final MWD. Excluding termination from the simulation reduces the the total number of ODEs by about $2/3$ and substantially reduces the computer time needed for the simulation. Therefore, in most of the cases, we train the RL agents on *no-termination* environments to save computational cost. Note that we still evaluate their performance on *with-termination* environments. Moreover, this strategy allows us to test the transferability of control policies learned by the RL agent onto similar but not identical environments, which could be of great importance in later works where we need to apply control policies learned with simulated environments to real, physically built reactors.

We assume that the volume of the system is completely determined by the amount of solvent and the number of monomer equivalents (including monomers incorporated in polymer chains). To calculate the system volume, we use a bulk styrene density of 8.73 mol/L as reported in early works[144] and a solvent density of 1.00 mol/L.

2.2 Using RL to control the ATRP reactor simulation

A reinforcement learning problem is usually phrased as an agent interacting with an environment (Figure 2.1). In our case, the agent is an RL controller and the environment is the ATRP reactor simulator. The agent interacts with the simulation at times separated by constant intervals, t_{step}. The interaction between the agent and the environment consists of three elements, each of which is indexed by the timestep (shown as a subscript t):

State ($\boldsymbol{s_t}$) At each timestep, the agent is given a vector, s_t, that is interpreted as the current state of the reaction system. The state vector is used by the agent to select actions. Here, s_t includes: (i) the concentrations of the non-trace species: monomer, dormant chains ($\mathrm{P_1Br}$, $\cdots$, $\mathrm{P_NBr}$), and Cu-based ATRP catalysts, (ii) the volume of the solution, and (iii) binary indicators of whether each of the addable reagents has reached its budget. Note that we include the monomer quantity into the state vector by adding it onto the quantity

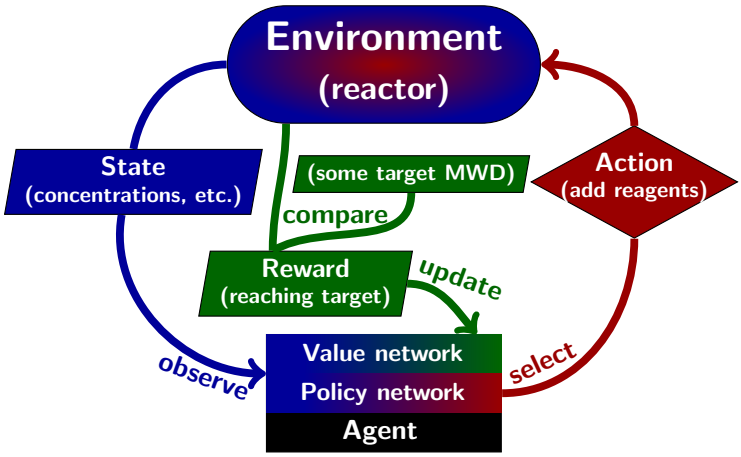

Figure 2.1: A schematic diagram of applying deep reinforcement learning in the ATRP reactor control setting.

of the initiator, or the shortest dormant chain.

Action (a_t) The agent is given a set of actions, $\mathcal{A}$, from which to select an action, a_t, to apply at timestep t. The set of actions is fixed and does not change throughout the simulation. Here, the actions correspond to the addition of a fixed amount of a chemical reagent. The set of actions, $\mathcal{A}$, also includes a *no-op*, selection of which means that no action is taken on the reaction simulation environment. The addable reagents are listed in Table 2.2, along with the amount that is added when the action is selected and the budget. When a reagent reaches its budget, the agent may still select the corresponding action, but this action becomes a no-op and does not alter the reaction simulation environment. Although the simulation allows addition of solvent, the effects of this action are not examined here. A very small amount of solvent is, however, used to initialize the simulation with a non-zero volume of a non-reactive species. Inclusion of other actions, such as changes in temperature, are possible but these are also not examined here.

Reward (r_t) At each timestep, the agent is given a reward, r_t, that indicates the degree to which the agent is succeeding at its task. In many RL problems, rewards may accrue at any time point. Here, however, the reward is based on the final MWD and so the agent receives a reward only when the reaction has run to completion. In practice, we allow the agent to interact with the simulation until all addable reagents have reached their budgets. The simulation then continues for a terminal simulation time of $t_{terminal} = 10^5$ seconds. The simulation environment then provides a reward to the agent based on the difference between the ending dormant chain MWD and the target MWD. This reward is defined in a two-level manner: when the maximum absolute difference between the normalized ending MWD and target MWD is less than 1×10^{-2} the agent obtains a reward of 0.1, and when this difference is less than 3×10^{-3}, the agent obtains a reward of 1.0. This two-level reward structure was determined empirically, with the lower first-level reward helping guide the agent in the early stages of training.

A single simulated ATRP reaction corresponds, in RL, to a single *episode*. Each episode

begins with a small amount of solvent (Table 2.2) and iterates through steps in which the agent is given the current state, s_t, the agent selects an action a_t that is applied to the simulation, and the simulation then runs for a time t_{step}. When all addable reagents have reached their budgets, the simulation continues for $t_{terminal} = 10^5$ seconds and returns a reward based on the difference between the ending dormant chain MWD and the target MWD.

Table 2.2: The initial amounts, addition unit amounts, and budget limits used for simulating styrene ATRP in this work. All quantities are in units of *mol*.

Addable reagents	Initial	Addition unit	Budget limit
Monomer	0	0.1	10.0
Activator	0	0.004	0.2
Deactivator	0	0.004	0.2
Initiator	0	0.008	0.4
Solvent	0.01	0	0

To train the agent, we use the A3C algorithm, a recent advance in actor-critic methods[21] that achieved state-of-the-art performance on many discrete-action control tasks.[111] Actor-critic[55] algorithms are a subclass of RL algorithms based on simultaneous training of two functions:

Policy ($\pi_{\theta_p}(s_t)$) The policy is used to select actions, e.g., which chemical reagent to add at time t. As shown schematically in Figure 1.3, actions are drawn from a probability distribution. The policy function generates this probability distribution, $\pi_{\theta_p}(a_t|s_t)$, which specifies, given the state of the ATRP reactor s_t, the probability that action a_t should be selected. The subscript θ_p represents the set of parameters that parameterize the policy function. In A3C, where a neural network is used for the policy, θ_p represents the parameters in this neural network.[40, 124]

Value ($V_{\theta_v}(s_t)$) Although the policy function is sufficient for use of the RL controller, training also involves a value function, $V_{\theta_v}(s_t)$. Qualitatively, this function is a measure of whether the reaction is on track to generate rewards. More precisely, we define a *return* $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ which includes not only the reward at the current state, but also future states up to timestep T. This is especially relevant here, as rewards are based on the final MWD and so are given only at the end of a reaction. A factor γ, which is greater than 0 and less than 1, discounts the reward for each step into the future, and is included to guarantee convergence of RL algorithms. The value function, $V_{\theta_v}(s_t)$, approximates the expected return, $\mathbb{E}[R_t|s_t]$, from state s_t. A3C uses a neural network for the value function, and θ_v represents the parameters in this network.

Below, we compare results from two different neural network architectures, labeled FCNN and 1D-CNN (see Section 2.3).

During training, A3C updates the parameters, θ_p and θ_v, of the policy and value functions. The actor-critic aspect of A3C refers to the use of the value function to critique the policy's ability

to select valuable actions. To update θ_p, policy gradient steps are taken according to the direction given by $\nabla_{\theta_p} \log \pi_{\theta_p}(a_t|s_t)\big(R_t - V_{\theta_v}(s_t)\big)$. Note that the current value function, $V_{\theta_v}(s_t)$, is used to update the policy, with the policy gradient step being in a direction that will cause the policy to favor actions that maximize the expected return. This may be viewed as using the value function to critique actions being selected by the policy. Moreover, the policy gradient becomes more reliable when the value function estimates the expected return more accurately. To improve the value function, the parameters θ_v are updated to minimize the ℓ^2 error $\mathbb{E}\big(R_t - V_{\theta_v}(s_t)\big)^2$ between the value function, $V_{\theta_v}(s_t)$, and the observed return, R_t. The observed return is obtained by using the current policy to select actions to apply to the reaction simulation environment.

The training therefore proceeds iteratively, with the current value function being used to update the policy and the current policy being used to update the value function. The parameter updates occur periodically throughout the course of an episode, or single polymerization reaction. The current policy is first used to generate a length-L sequence of state transitions $\{s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots, s_{t+L}\}$. This length-$L$ sequence is referred to as a *rollout*. At the end of each rollout, the information generated during the rollout is used to update θ_p and θ_v. To take advantage of multi-core computing architectures, the training process is distributed to multiple asynchronous parallel learners. A3C keeps a global version of θ_p and θ_v. Each learner has access to a separate copy of the reaction simulation environment and a local version of θ_p and θ_v. After a learner performs a rollout, it generates updates to θ_p and θ_v. These updates are then applied to the global versions of θ_p and θ_v, and the learner replaces its local version with the global version. In this manner, each learner periodically incorporates updates generated by all learners.

2.3 Implementation details

The neural networks used for the policy and value functions share a common stack of hidden layers, but use separate final output layers. We compare results from two different network architectures for the hidden layers. The first architecture, FCNN, is a simple fully-connected neural network with two hidden layers containing 200 and 100 hidden units, respectively. The second architecture, 1D-CNN, is convolutional. In 1D-CNN, the input feature vector is fed into a first 1D convolutional layer having 8 filters of length 32 with stride 2, followed by a second 1D convolutional layer having 8 filters of length 32 with stride 1. The output of the second 1D convolutional layer is then fed into a fully-connected layer with 100 units. All hidden layers use rectifier activation. The final layer of the value network produces a single scalar output that is linear in the 100 units of the last hidden layer. The final layer of the policy network is a softmax layer of the same 100 hidden units, with a length-6 output representing a probability distribution over the 6 actions. For a crude estimate of model complexity, FCNN and 1D-CNN contain 42607 and 9527 trainable parameters, respectively.

We implemented the A3C algorithm with 12 parallel CPU learners.[92] The discount factor in the return is $\gamma = 0.99$, and the maximum rollout length is 20. The length of a rollout may be shorter than 20 when the last state in the sequence is a terminal state. After a learner collects a length-L rollout, $\{s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots, s_{t+L}\}$, it generates updates for θ_p and θ_v by performing stochastic gradient descent steps for each $t' \in \{t, \cdots, t + L - 1\}$. Define the boot-

strapped multi-step return $R'_{t'} = I_{t+L}\gamma^{t+L-t'}V_{\theta'_v}(s_{t+L}) + \sum_{i=t'}^{t+L}\gamma^{i-t'}r_i$ where $I_{t+L} = 0$ if s_{t+L} is the terminal state and 1 otherwise. The prime on θ'_v in $V_{\theta'_v}(s_{t+L})$ indicates that the value function is evaluated using the local copy of the network parameters. The update direction of θ_p is set according to

$$d\theta_p = -\nabla_{\theta'_p}\log\pi_{\theta'_p}(a_{t'}|s_{t'})\big(R'_{t'} - V_{\theta'_v}(s_{t'})\big) + \beta\nabla_{\theta'_p}H\big(\pi_{\theta'_p}(s_{t'})\big).$$

$H\big(\pi_{\theta'_p}(s_{t'})\big)$ is the entropy of $\pi_{\theta'_p}(s_{t'})$ and acts as a regularization term that helps prevent $\pi_{\theta'_p}(s_{t'})$ from converging to sub-optimal solutions. β is the regularization hyperparameter, for which we use $\beta = 0.01$. θ_v is updated according to the direction of

$$d\theta_v = \nabla_{\theta'_v}\big(R'_{t'} - V_{\theta'_v}(s_{t'})\big)^2.$$

Updates of the network parameters are done using the ADAM optimizer[54] with a learning rate of 1×10^{-4}.

Additionally, after each action is drawn from the probability distribution generated by the policy, the agent repeats the action for 4 times before selecting the next action. This repetition shortens the length of a full episode by a factor of 4 from the RL agent's perspective and so prevents the value function from exponential vanishing.[90]

2.4 Results and discussion

2.4.1 Targeting Gaussian MWDs with different variance

Our first goal is to train the RL controller against some MWDs with simple analytic forms, for which Gaussian distributions with different variances seem a natural choice. Seemingly simple, Gaussian MWDs exemplify the set of symmetric MWDs the synthesis of which requires advanced ATRP techniques such as activators regenerated by electron transfer (ARGET).[73] Living polymerization produces a Poisson distribution with a variance that depends only on the average chain length, which is set by the monomer-to-initiator ratio. The variance from the ideal living polymerization provides a lower limit to the variance of the MWD. Here, we choose Gaussian distributions with variances ranging from near this lower limit to about twice that limit. Increasing the variance of the MWD can have substantial effects on the properties of the resulting material.[78]

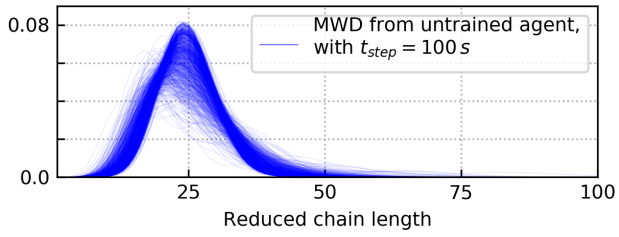

Figure 2.2: Superposition of 1000 ending MWDs from untrained agents when the time interval between actions is 100 seconds. Vertical axis is fraction of polymer chains.

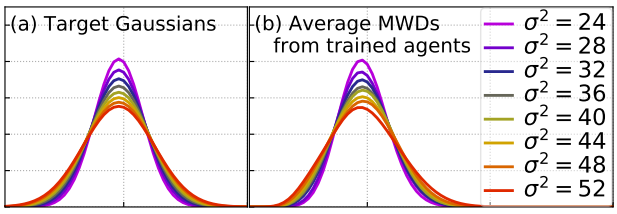

Figure 2.3: Comparison of the human-specified target Gaussian MWDs with the average ending MWDs given by trained 1D-CNN agents, with averaging being over 100 episodes. The horizontal and vertical spacings between dotted line grids are 25 and 0.02, respectively.

For this task, we set the time interval between two actions to 100 seconds. This setting was chosen for two main reasons. First, due to the choice of the addition unit amounts and budget limits of addable reagents, it typically takes $300\sim400$ simulator steps to finish one episode, and so this choice of time interval corresponds to $\sim$10 hours of real reaction time before the terminal step. More importantly, it allows an untrained RL controller to produce a widely variable ending MWD, as illustrated by the 1000 MWDs of Figure 2.2. A widely variable ending MWD is necessary for RL agents to discover strategies for target MWDs through self-exploration.[51, 53]

As specific training targets, we select Gaussian MWDs with variances (σ^2's) ranging from 24 to 52, which covers the theoretical lower limit of the variance to a variance of more than twice this limit. Figure 2.3(a) shows the span of these target MWDs. A summary of the trained 1D-CNN agents' performance on this task is shown in Figure 2.3(b). Each ending MWD is an average over 100 episodes, generated using the trained 1D-CNN controller. Note that this MWD averaging is equivalent to blending polymer products generated in different reactions,[63] a common practice in both laboratory and industrial polymerization.[23, 52, 62, 149] The trained 1D-CNN agent used in these test runs is that which gave the best performance in the training process, i.e., the neural network weights are those that generated the highest reward during the training process. During training, termination reactions are not included in the simulation, but during testing, these reactions are included. For all 8 target Gaussian MWDs, the average ending MWDs are remarkably close to the corresponding targets. The maximum absolute deviation from the target MWD is an order of magnitude less than the peak value of the distribution function. These results show that control policies learned on simulation environments that exclude termination transfer well to environments that include termination. This is perhaps not surprising because ATRP of styrene is close to an ideal living polymerization, with less than 1% of monomers residing in chains that underwent a termination reaction. Tests on changing other aspects of the polymerization simulation are given in the following sections.

Transferability tests on noisy environments

To test the robustness of the learned control policies, the trained 1D-CNN agents were evaluated on simulation environments that include both termination reactions and simulated noise.[4, 28, 45] We introduce noise on the states as well as actions. On states, we apply Gaussian noise with standard deviation 1×10^{-3} on every observable quantity. (The magnitude of the observable quantities range from 0.01 to 0.1.) In the simulation, we introduce three types of noise. First, the

time interval between consecutive actions is subject to a Gaussian noise, whose standard deviation is 1% of the mean time interval. Gaussian noise is also applied to the amount of chemical reagent added for an action, again with a standard deviation that is 1% of the addition amount. Lastly, every kinetics rate constant used in non-terminal steps is subject to Gaussian noise, with the standard deviation being 10% of the mean value. Note that we crop the Gaussian noise in the simulation at $\pm 3\sigma$ to avoid unrealistic physics, such as negative time intervals, addition of negative amounts, or negative kinetic rate constants. Once all budgets have been met, the simulation enters its terminal step and the RL agent no longer has control over the process. During this terminal step, we do not apply noise.

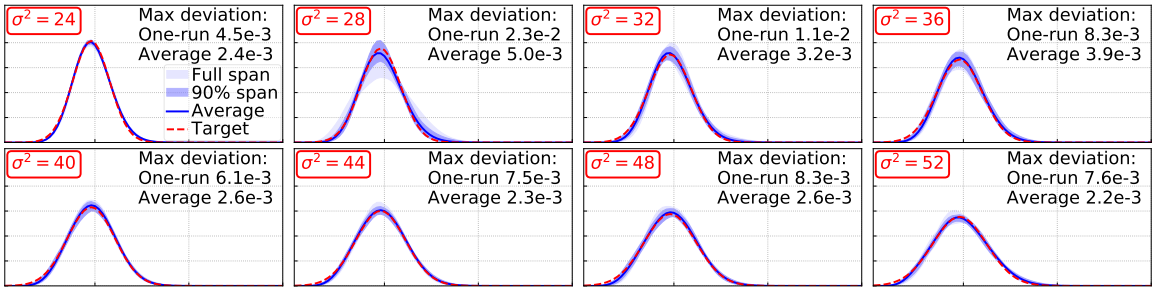

Figure 2.4: Performance of 1D-CNN agents trained on the target Gaussian MWDs of Figure 2.3 on simulation environments that include both termination reactions and noise. In each subplot, the horizontal axis represents the reduced chain length and runs from 1 to 75, and the vertical axis represents fraction of polymer chains and runs from 0.0 to 0.11.

Performance of the 1D-CNN agents, trained against the target Gaussian MWDs of Figure 2.3, on noisy environments is shown in Figure 2.4. The trained agent is used to generate 100 episodes and the statistics of final MWDs are reported in a variety of ways. The average MWD from the episodes is shown as a solid dark blue line. The light blue band shows the full range of the 100 MWDs and the blue band shows, at each degree of polymerization, the range within which 90 of the MWDs reside. The control policies learned by 1D-CNN agents seem to be robust. The deviation of the average MWD is an order of magnitude less than the peak value of the MWD. Deviations of the MWD from a single episode can vary more substantially from the target MWD, but the resulting MWDs are still reasonably close to the target MWD. On average, the maximum absolute deviation between a one-run MWD and the target is still less than 5% of the peak MWD value.

2.4.2 Targeting MWDs with diverse shapes

Beyond Gaussian MWDs, we also trained the 1D-CNN agent against a series of diverse MWD shapes. We have chosen bimodal distributions as a challenging MWD to achieve in a single batch process. Such bimodal distributions have been previously studied as a means to controlling the microstructure of a polymeric material.[112, 146, 150]

To enable automatic discovery of control policies that lead to diverse MWD shapes, it is necessary to enlarge the search space of the RL agent, which is related to the variability in the ending MWDs generated by an untrained agent. We found empirically that a larger time

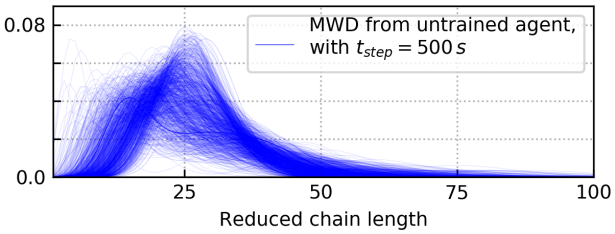

Figure 2.5: Superposition of 1000 ending MWDs from untrained agents when the time interval between actions is 500 seconds. Vertical axis is fraction of polymer chains.

interval between actions leads to wider variation in the MWDs obtained with an untrained agent. Throughout this section, the time interval between actions t_{step} is set to 500 seconds. Figure 2.5 shows 1000 superimposed ending MWDs given by the untrained agent with this new time interval setting, and the span is much greater than in Figure 2.2 where $t_{step} = 100$ seconds.

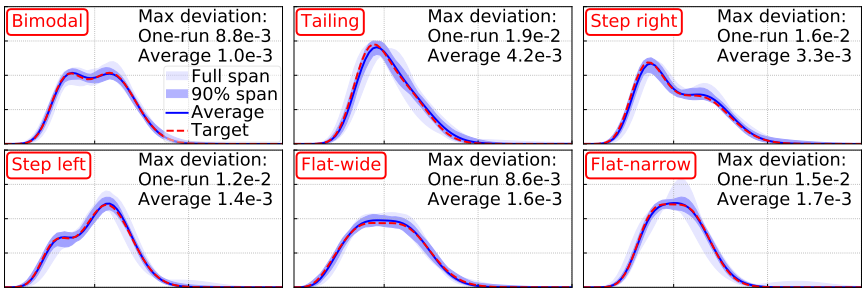

Figure 2.6: Performance of trained 1D-CNN agents on noisy, with-termination environments targeting diverse MWD shapes. In each subplot, the horizontal axis represents the reduced chain length and runs from 1 to 75, and the vertical axis is fraction of polymer chains and runs from 0.0 to 0.08.

The target MWDs with diverse shapes are manually picked from 1000 random ATRP simulation runs (i.e., episodes under the control of an untrained agent). Agents trained on these targets have satisfactory performance. The average MWDs over 100 batch runs match the targets nearly perfectly. In addition, there is a large probability (90%) that a one-run ending MWD controlled by a trained agent falls into a thin band whose deviation from the target is less than 1×10^{-2} (Figure 2.6). All these agents are trained on noisy, no-termination environments and evaluated on noisy, with-termination environments. The parameters specifying the noise are identical to those used in the earlier sections. The results indicate that a simple convolutional neural network with less than 10^4 parameters can encode control policies that lead to complicated MWD shapes with surprisingly high accuracy. Again, adding noise to the states, actions, and simulation parameters does not degrade the performance of the RL agents significantly. This tolerance to noise may allow transfer of control policies, learned on simulated reactors, to actual reactors.

Chapter 3

Sample Efficiency Improvements

3.1 Application of actor-critic with experience replay (ACER) and observation space thermometer encoding

As an on-policy online RL algorithm, A3C typically requires many simulator steps to train. In the above training processes, more than 10^5 simulated chemical experiments must be performed in order to train towards a synthetic target. It is therefore desired to improve the sample efficiency of the training algorithm. We attempt to improve the sample efficiency from both an algorithmic perspective and a feature engineering perspective.

Actor-critic with experience replay (ACER)[142] is a variant of actor-critic that makes use of off-policy training on an experience replay containing past transitions. By storing experienced transitions in a replay memory and utilizing truncated importance sampling for off-policy correction, ACER greatly improves the sample efficiency comparing with A3C. Generally, ACER replaces the state-dependent critic function $V_{\theta_v}(s_t)$ with a state-action-dependent critic function $V_{\theta_v}(s_t,\, a_t)$ and performs off-policy corrections to transitions sampled from the experience replay using a variant of the Retrace(λ) correction scheme.[93] ACER achieved state-of-the-art results on a number of discrete control tasks such as Atari games. We therefore experiment its applicability on the ATRP synthesis tasks. In terms of implementation details, each training thread contains a replay memory that can hold 1000 batches at maximum, and off-policy training begins once the replay memory contains 100 batches. For each online batch learning step, we drew a random integer from Poisson(4) and performed this number of offline, off-policy batch learning steps. All the other settings are as same as our A3C settings.

Thermometer encoding is a dimension-expansion feature engineering method commonly used in supervised learning tasks.[17] Recently, it has been applied to make neural networks more robust against adversarial examples.[10] Thermometer encoding converts any real-valued quantity to a vector of predefined length and therefore expands the dimension of the feature space by 1. A raw observation from the simulated ATRP reactions is a 1D vector of concentrations of species. After encoding, the observation space becomes a 2D array, and as a result we use 2D convolutional neural networks for both the policy network and the value network. In contrast to the discrete thermometer encoding scheme used in Ref. [10], here we use a smoothed encoding scheme based on the sigmoid function σ. Using notations similar to Ref. [10], we pick k equally-

spaced quantization values $0 < b_1 < b_2 < \cdots < b_k = 0.06$. For a real-valued quantity v, the thermometer-encoded vector is:

$$\left[\sigma(\frac{v - b_1}{h}),\ \sigma(\frac{v - b_2}{h}),\ \cdots,\ \sigma(\frac{v - b_k}{h})\right],$$

where h is a hyperparameter controlling the bin-width. In practice, we convert the concentrations of polymer chains of various lengths into thermometer encoded vectors, and concatenate them in the original order before feeding into the 2D convolutional layers of the policy and value networks. Observation entries that are not concentrations bypass the convolutional layers and directly go into the first layer of the fully-connected part of the networks. We use $k = 84$ quantization values, encode concentrations for polymer chains of length 1 to 84, and set the bin-width hyperparameter $h = 0.001$. With the observation space being thermometer encoded, the corresponding policy/value networks becomes 2D convolutional neural networks. The first convolutional layer contains 16 filters of size 8×8 with strides 4×4. The second convolutional layer contains 32 filters of size 4×4 with strides 2×2. The third convolutional layer contains 32 filters of size 3×3 with strides 1×1. The output of the third convolutional layer is fed into a fully-connected hidden layer with 128 units, and the hidden layer is then connected to the final policy/value outputs. All layers use rectifier activation.

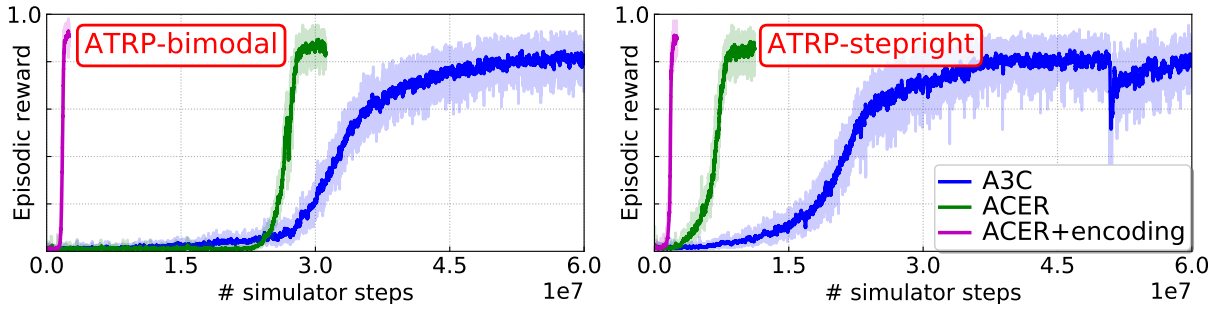

Figure 3.1: Effect of ACER and thermometer encoding on the training sample efficiency on the `ATRP-bimodal` and `ATRP-stepright` environments.

We evaluate the effect of ACER and thermometer encoding on two selected simulated ATRP synthesis learning environments, `ATRP-bimodal` and `ATRP-stepright`, corresponding to the "Bimodal" and "Step right" synthetic targets in Figure 2.6. Noticeably, the combination of ACER and observation space thermometer encoding greatly reduces the number of simulator steps needed to reach peak performance, by a factor that is greater than 10. ACER by itself improves sample efficiency by a factor of 2 to 4 on the training environments we tested.

3.2 ACER with prioritized experience replay

To further improve sample efficiency, we make the algorithmic contribution of adding prioritized experience replay (PER)[114] to ACER. Originally, PER was proposed for being used with 1-step, value-based TD-learning methods. On the other hand, ACER and other actor-critic based deep-RL algorithms such as A3C and IMPALA[29] makes use of multi-step, policy-based TD-learning. The use of PER in ACER therefore requires a proper treatment of sequence

prioritization,[114] as well as a prioritization scheme that is compatible with policy-based methods. It has been proposed that the absolute value of the TD-error can be used to prioritize Q-learning, and the KL-divergence between the online distribution and the target distribution can be used to prioritize distributional Q-learning. Yet it remains to be defined what quantities can be used to prioritize off-policy actor-critic methods.

We propose two potential prioritization quantities for ACER. The first one is the naive approach of using the absolute value of the TD-error of the critic, or the value network. Since TD-error is only defined for one transition, to make this approach compatible with multi-step learning it is required to handle sequence prioritization. We use the simple method of using the average of the absolute TD-error over the entire sequence for prioritizing the sequence.

The second one, which we refer to as "differential prioritization", is an approach that in theory generalizes to any TD-learning algorithms and is therefore also applicable to ACER. We propose to use the ℓ^1-norm of the gradient of the loss with respect to the output of the neural network (referred to as "output-norm" from now on) being optimized. In our current experiments, the output of the neural network refers to the linear output of the last layer before activation. It remains to see whether it is better to use inactivated or activated output. Note that in 1-step Q-learning, this "output-norm" reduces down to the absolute TD-error, if we directly use the linear output of the neural network as the state-action-dependent Q-value. To understand this choice of prioritization scheme, consider from an optimization perspective that we are trying to optimize the weights in a neural network by training on batched samples coming from a population using stochastic gradient descent.[114] Batches corresponding to large norms of the gradient of the loss with respect to the neural network weights (referred to as "weight-norm") lead to large improvements of the minimization process (on average), whereas batches corresponding to small "weight-norms" lead to small improvements. In principle, "weight-norm" should be a very natural choice of the prioritization quantity. However, computing this norm requires a full backpropagation through the entire neural network and is considered computationally expensive in most application scenarios. On the other hand, due to the nature of the backpropagation algorithm, "output-norm" is usually a good proxy for "weight-norm". Therefore, "output-norm" may be used as a prioritization quantity that is essentially applicable to any TD-learning algorithm that involves training a neural network. Another benefit of using the gradient norm of the loss as a prioritization quantity is that it eliminates the need for the treatment of sequence prioritization. One can compute a scalar loss value naturally from a sequence of transitions, and differentiating this loss with respect to the batched output computed from this sequence always gives a well-defined gradient vector, so long as the loss is differentiable.

A comparison of the effect of the proposed prioritization schemes is shown in Figure 3.2. We evaluate their effects on various types of discrete control environments including a classical control environment `CartPole`, a Box2D simulator environment `LunarLander`, and simulated ATRP synthesis environments `ATRP-bimodal` and `ATRP-stepright`. Using a uniform replay memory as in the original ACER algorithm leads to the lowest sample efficiency in any of the four environments. On `CartPole`, uniform replay memory leads to unstable training. On `LunarLander`, the RL agent consistently converges to a sub-optimal policy when trained with uniform replay. On the two ATRP synthesis environments, prioritization improves sample efficiency by a factor of $\sim$1/3. Noticeably, on both `CartPole` and `LunarLander`, while prioritizing by critic absolute TD-error or by differential lead to similar sample efficiency and

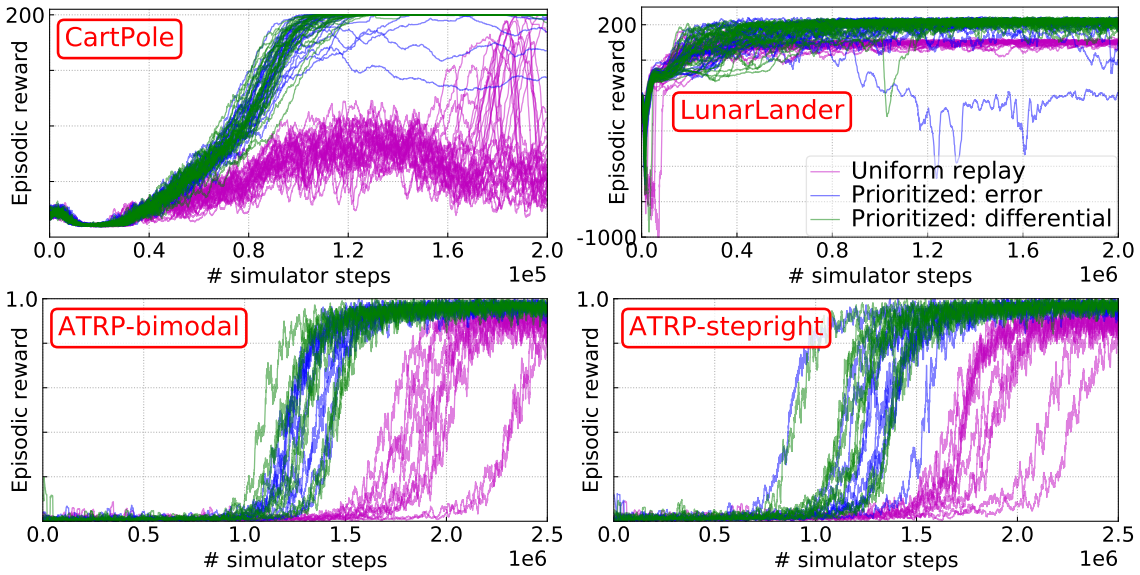

Figure 3.2: Comparison of the effect of different prioritization schemes on the learning curves. For `CartPole` and `LunarLander`, learning curves from 30 independent runs are presented. For `ATRP-bimodal` and `ATRP-stepright` we present learning curves from 10 independent runs.

final optimal solution, the differential prioritization scheme is less susceptible to performance collapses, as the agent trained with differential prioritization can consistently maintain its peak performance in any of the 30 independent runs.

In terms of the details of the above experiments, the PER hyperparameters[114] are set to $\alpha = 0.6$ and $\beta = 1.0$. The other settings on the replay memory is the same as in the ACER uniform replay memory setting described in the above section. On `CartPole`, we set the sequence length in multi-step learning to 2 and used a batch containing 16 sequences, with learning rate 10^{-4}. On `LunarLander`, we initialized the neural network weights to the same across different training runs in order to establish comparable learning curves, scaled the rewards by 1/10 during training and set multi-step learning sequence length to 8 and used a batch containing 4 sequences, with learning rate 10^{-3}. Sequences that are to be evicted from the PER are sampled each time according to the inverse of the priority. The neural networks used for the ATRP synthesis environments are the same as in the previous section. On `CartPole` and `LunarLander`, we used a simple 1-layer fully-connected neural network with 100 hidden units.

Chapter 4

Conclusion

This paper introduces a general methodology for using deep reinforcement learning techniques to control a chemical process in which the product evolves throughout the progress of the reaction. A proof-of-concept for the utility of this approach is obtained by using the controller to guide growth of polymer chains in a simulation of ATRP. ATRP was chosen because this reaction system allows detailed control of a complex reaction process. The resulting controllers are tolerant to noise in the kinetic rate constants used in the simulation, noise in the states on which the controller bases its decisions, and noise in the actions taken by the controller. This tolerance to noise may allow agents trained on simulations of the reaction to be transferred to the actual laboratory without extensive retraining, although evaluation of this aspect is left to future work. This approach, of carrying out initial training of a controller on a simulation, has been successfully applied in other domains such as robotics and vision-based RL.[16, 64, 111] Additional work is also needed to better understand the extent to which the controller can achieve synthetic targets when decisions are based on less detailed information regarding the state of the reactor. The ability of the approach to target multiple properties,[120, 133] such as targeting MWD and viscosity simultaneously, or targeting more complex architectures, such as gradient or brush polymers, also remains to be explored. Our efforts to optimize the reinforcement learning methodology is still ongoing, and we hope to apply similar approaches to guide other chemical reactions.

A developmental open-source implementation of our approach is freely available on GitHub (`https://github.com/spring01/reinforcement_learning_atrp`) under the GPL-v3 license.

Bibliography

[1] Mamdouh Al-Harthi, João BP Soares, and Leonardo C Simon. Dynamic monte carlo simulation of atom-transfer radical polymerization. *Macromol. Mater. Eng.*, 291(8):993–1003, 2006. 2.1

[2] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. A brief survey of deep reinforcement learning. *arXiv preprint*, page arXiv:1708.05866, 2017. 1.2

[3] M Bahita and K Belarbi. Model reference neural-fuzzy adaptive control of the concentration in a chemical reactor (cstr). *IFAC-PapersOnLine*, 49(29):158–162, 2016. 1.3

[4] Bram Bakker. Reinforcement learning with long short-term memory. In *Advances in Neural Information Processing Systems*, pages 1475–1482, 2002. 2.4.1

[5] Samuel Barrett, Matthew E Taylor, and Peter Stone. Transfer learning for reinforcement learning on a physical robot. In *Ninth International Conference on Autonomous Agents and Multiagent Systems-Adaptive Learning Agents Workshop (AAMAS-ALA)*, 2010. 1.3

[6] Thomas Binder, Luise Blank, H Georg Bock, Roland Bulirsch, Wolfgang Dahmen, Moritz Diehl, Thomas Kronseder, Wolfgang Marquardt, Johannes P Schlöder, and Oskar von Stryk. Introduction to model based optimization of chemical processes on moving horizons. In *Online Optimization of Large Scale Systems*, pages 295–339. Springer, 2001. 1.3

[7] Byron Boots, Sajid M Siddiqi, and Geoffrey J Gordon. Closing the learning-planning loop with predictive state representations. *Int. J. Robotics Res.*, 30(7):954–966, 2011. 1

[8] Jovan D Bošković and Kumpati S Narendra. Comparison of linear, nonlinear and neural-network-based adaptive controllers for a class of fed-batch fermentation processes. *Automatica*, 31(6):817–840, 1995. 1.3

[9] Peter N Brown, George D Byrne, and Alan C Hindmarsh. Vode: A variable-coefficient ode solver. *SIAM J. Sci. Comput.*, 10(5):1038–1051, 1989. 2.1

[10] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *Submissions to International Conference on Learning Representations*, 2018. 3.1

[11] George D. Byrne and Alan C. Hindmarsh. A polyalgorithm for the numerical solution of ordinary differential equations. *ACM Trans. Math. Softw.*, 1(1):71–96, 1975. 2.1

[12] Anna Carlmark and Eva E Malmström. Atrp grafting from cellulose fibers to create block-

copolymer grafts. *Biomacromolecules*, 4(6):1740–1745, 2003. 1.1

[13] R Nicholas Carmean, Troy E Becker, Michael B Sims, and Brent S Sumerlin. Ultra-high molecular weights via aqueous reversible-deactivation radical polymerization. *Chem*, 2 (1):93–101, 2017. 1.1

[14] C Chatzidoukas, JD Perkins, EN Pistikopoulos, and C Kiparissides. Optimal grade transition and selection of closed-loop controllers in a gas-phase olefin polymerization fluidized bed reactor. *Chem. Eng. Sci.*, 58(16):3643–3658, 2003. 1.3

[15] Tibor Chovan, Thierry Catfolis, and Kürt Meert. Neural network architecture for process control based on the rtrl algorithm. *AIChE J.*, 42(2):493–502, 1996. 1.3

[16] Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint*, page arXiv:1610.03518, 2016. 1.3, 4

[17] Christopher R Collins, Geoffrey J Gordon, O Anatole von Lilienfeld, and David J Yaron. Constant size molecular descriptors for use with machine learning. *arXiv preprint arXiv:1701.06649*, 2017. 3.1

[18] Sajjad Dadashi-Silab, Xiangcheng Pan, and Krzysztof Matyjaszewski. Photoinduced iron-catalyzed atom transfer radical polymerization with ppm levels of iron catalyst under blue light irradiation. *Macromolecules*, 50(20):7967–7977, 2017. 1.1

[19] J Fernandez de Canete, Pablo del Saz-Orozco, Roberto Baratti, Michela Mulas, A Ruano, and Alfonso Garcia-Cerezo. Soft-sensing estimation of plant effluent concentrations in a biological wastewater treatment plant using an optimal neural network. *Expert Syst. Appl.*, 63:8–19, 2016. 1.3

[20] Marco Antnio de Souza L. Cuadros, Celso J Munaro, and Saul Munareto. Novel model-free approach for stiction compensation in control valves. *Ind. Eng. Chem. Res.*, 51(25): 8465–8476, 2012. 1.3

[21] Thomas Degris, Patrick M Pilarski, and Richard S Sutton. Model-free reinforcement learning with continuous action in practice. In *American Control Conference (ACC), 2012*, pages 2177–2182. IEEE, 2012. 2.2

[22] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8599–8603. IEEE, 2013. 1.2

[23] Paul J DesLauriers, Max P McDaniel, David C Rohlfing, Rajendra K Krishnaswamy, Steven J Secora, Elizabeth A Benham, Pamela L Maeger, AR Wolfe, Ashish M Sukhadia, and Bill B Beaulieu. A comparative study of multimodal vs. bimodal polyethylene pipe resins for pe-100 applications. *Polym. Eng. Sci.*, 45(9):1203–1213, 2005. 2.4.1

[24] Dagmar R D'hooge, Dominik Konkolewicz, Marie-Françoise Reyniers, Guy B Marin, and Krzysztof Matyjaszewski. Kinetic modeling of icar atrp. *Macromol. Theory Simul.*, 21 (1):52–69, 2012. 1.1

[25] Fabio di Lena and Krzysztof Matyjaszewski. Transition metal catalysts for controlled radical polymerization. *Prog. Polym. Sci.*, 35(8):959–1021, 2010. 1.1

[26] Marco Drache and Georg Drache. Simulating controlled radical polymerizations with mcpolymera monte carlo approach. *Polymers*, 4(3):1416–1442, 2012. 1.1

[27] Anca D Dragan, Geoffrey J Gordon, and Siddhartha S Srinivasa. Learning from experience in manipulation planning: Setting the right goals. In *Robotics Research*, pages 309–326. Springer, 2017. 1, 1.2

[28] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016. 2.4.1

[29] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018. 3.2

[30] Fabio Herbst Florenzano, Roland Strelitzki, and Wayne F Reed. Absolute, on-line monitoring of molar mass during polymerization reactions. *Macromolecules*, 31(21):7226–7238, 1998. 1.3

[31] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. *arXiv preprint*, page arXiv:1706.10295, 2017. 1.2

[32] Mosè Galluzzo and Bartolomeo Cosenza. Control of a non-isothermal continuous stirred tank reactor by a feedback–feedforward structure using type-2 fuzzy logic controllers. *Inf. Sci.*, 181(17):3535–3550, 2011. 1.3

[33] Haifeng Gao and Krzysztof Matyjaszewski. Synthesis of star polymers by a combination of atrp and the click coupling method. *Macromolecules*, 39(15):4960–4965, 2006. 1.1

[34] Haifeng Gao and Krzysztof Matyjaszewski. Synthesis of molecular brushes by grafting onto method: combination of atrp and click reactions. *J. Am. Chem. Soc*, 129(20):6633–6639, 2007. 1.1

[35] Dillon T Gentekos, Lauren N Dupuis, and Brett P Fors. Beyond dispersity: deterministic control of polymer molecular weight distribution. *J. Am. Chem. Soc*, 138(6):1848–1851, 2016. 1.1, 1.2

[36] Navid Ghadipasha, Wenbo Zhu, Jose A Romagnoli, Terry McAfee, Thomas Zekoski, and Wayne F Reed. Online optimal feedback control of polymerization reactors: Application to polymerization of acrylamide–water–potassium persulfate (kps) system. *Ind. Eng. Chem. Res.*, 56(25):7322–7335, 2017. 1.3

[37] Geoffrey J Gordon. Stable function approximation in dynamic programming. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 261–268, 1995. 1.2

[38] Geoffrey J Gordon. Reinforcement learning with function approximation converges to a

region. In *Advances in Neural Information Processing Systems*, pages 1040–1046, 2001. 1.2

[39] Atsushi Goto and Takeshi Fukuda. Kinetics of living radical polymerization. *Prog. Polym. Sci.*, 29(4):329–385, 2004. 1.1

[40] Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *J. Mach. Learn. Res.*, 5(Nov):1471–1530, 2004. 2.2

[41] AA Gridnev and SD Ittel. Dependence of free-radical propagation rate constants on the degree of polymerization. *Macromolecules*, 29(18):5864–5874, 1996. 2.1

[42] Craig J Hawker. Molecular weight control by a "living" free-radical polymerization process. *J. Am. Chem. Soc*, 116(24):11185–11186, 1994. 1.1

[43] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *arXiv preprint*, page arXiv:1709.06560, 2017. 1.2

[44] AW Hermansson and S Syafiie. Model predictive control of ph neutralization processes: a review. *Control Eng. Pract.*, 45:98–109, 2015. 1.3

[45] Todd Hester and Peter Stone. The open-source texplore code release for reinforcement learning on robots. In *Robot Soccer World Cup*, pages 536–543. Springer, 2013. 2.4.1

[46] AC Hindmarsh and GD Byrne. Episode: an effective package for the integration of systems of ordinary differential equations.[for stiff or non-stiff problems, in fortran for cdc or ibm computers; tstep, core integrator routine; convrt, to change between single and double precision coding]. Technical report, California Univ., Livermore (USA). Lawrence Livermore Lab., 1977. 2.1

[47] Alan C Hindmarsh. Odepack, a systematized collection of ode solvers, rs stepleman et al.(eds.), north-holland, amsterdam,(vol. 1 of), pp. 55-64. *IMACS Transactions on Scientific Computation*, 1:55–64, 1983. 2.1

[48] Mohammad Anwar Hosen, Mohd Azlan Hussain, and Farouq S Mjalli. Control of polystyrene batch reactors using neural network based model predictive control (nnmpc): An experimental investigation. *Control Eng. Pract.*, 19(5):454–467, 2011. 1.3

[49] Mohamed Azlan Hussain. Review of the applications of neural networks in chemical process controlsimulation and online implementation. *Artif. Intell. Eng.*, 13(1):55–68, 1999. 1.3

[50] Kenneth R Jackson and Ron Sacks-Davis. An alternative implementation of variable step-size multistep formulas for stiff odes. *ACM Trans. Math. Softw.*, 6(3):295–318, 1980. 2.1

[51] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11(Apr):1563–1600, 2010. 2.4.1

[52] Renata Jovanović, Keltoum Ouzineb, Timothy F McKenna, and Marc A Dubé. Butyl acrylate/methyl methacrylate latexes: adhesive properties. In *Macromolecular Symposia*, volume 206, pages 43–56. Wiley Online Library, 2004. 2.4.1

[53] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Mach. Learn.*, 49(2-3):209–232, 2002. 2.4.1

[54] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*, page arXiv:1412.6980, 2014. 2.3

[55] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pages 1008–1014, 2000. 2.2

[56] Veronika Kottisch, Dillon T Gentekos, and Brett P Fors. shaping the future of molecular weight distributions in anionic polymerization. *ACS Macro Lett.*, 5(7):796–800, 2016. 1.1

[57] Jan Koutník, Jürgen Schmidhuber, and Faustino Gomez. Evolving deep unsupervised convolutional networks for vision-based reinforcement learning. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 541–548. ACM, 2014. 1

[58] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 1.2

[59] Pawel Krys and Krzysztof Matyjaszewski. Kinetics of atom transfer radical polymerization. *Eur. Polym. J.*, 89:482–523, 2017. 1.1

[60] Pawel Krys, Marco Fantin, Patrícia V Mendonça, Carlos MR Abreu, Tamaz Guliashvili, Jaquelino Rosa, Lino O Santos, Arménio C Serra, Krzysztof Matyjaszewski, and Jorge FJ Coelho. Mechanism of supplemental activator and reducing agent atom transfer radical polymerization mediated by inorganic sulfites: experimental measurements and kinetic simulations. *Polym. Chem.*, 8(42):6506–6519, 2017. 1.1

[61] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015. 1.2

[62] Marcelo Kaminski Lenzi, Michael F Cunningham, Enrique Luis Lima, and José Carlos Pinto. Producing bimodal molecular weight distribution polymer resins using living and conventional free-radical polymerization. *Ind. Eng. Chem. Res.*, 44(8):2568–2578, 2005. 2.4.1

[63] R Leonardi, C Natalie, Rick D Montgomery, Julia Siqueira, Terry McAfee, Michael F Drenski, and Wayne F Reed. Automatic synthesis of multimodal polymers. *Macromol. React. Eng.*, page 1600072, 2017. 1.3, 2.4.1

[64] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Robotics Res.*, page 0278364917710318, 2016. 4

[65] Steven V Ley, Daniel E Fitzpatrick, Richard Ingham, and Rebecca M Myers. Organic synthesis: march of the machines. *Angew. Chem. Int. Ed.*, 54(11):3449–3464, 2015. 1

[66] Xiaohui Li, Wen-Jun Wang, Bo-Geng Li, and Shiping Zhu. Kinetics and modeling of solution arget atrp of styrene, butyl acrylate, and methyl methacrylate. *Macromol. React. Eng.*, 5(9-10):467–478, 2011. 2.1, 2.1

[67] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint*, page

arXiv:1701.07274, 2017. 1.2

[68] Zhibo Li, Ellina Kesselman, Yeshayahu Talmon, Marc A Hillmyer, and Timothy P Lodge. Multicompartment micelles from abc miktoarm stars in water. *Science*, 306(5693):98–101, 2004. 1.1

[69] Yitao Liang, Marlos C Machado, Erik Talvitie, and Michael Bowling. State of the art control of atari games using shallow reinforcement learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 485–493. International Foundation for Autonomous Agents and Multiagent Systems, 2016. 1.2

[70] G Lightbody and GW Irwin. Direct neural model reference adaptive control. *IEE Proc.-Control Theory Appl.*, 142(1):31–43, 1995. 1.3

[71] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint*, page arXiv:1509.02971, 2015. 1.2

[72] JS Lim, Mohamed Azlan Hussain, and MK Aroua. Control of a hydrolyzer in an oleochemical plant using neural network based controllers. *Neurocomputing*, 73(16):3242–3255, 2010. 1.3

[73] Jessica Listak, Wojciech Jakubowski, Laura Mueller, Andrzej Plichta, Krzysztof Matyjaszewski, and Michael R Bockstaller. Effect of symmetry of molecular weight distribution in block copolymers on formation of metastable morphologies. *Macromolecules*, 41 (15):5919–5927, 2008. 1.1, 2.4.1

[74] Nathaniel A Lynd and Marc A Hillmyer. Influence of polydispersity on the self-assembly of diblock copolymers. *Macromolecules*, 38(21):8803–8810, 2005. 1.1

[75] Nathaniel A Lynd and Marc A Hillmyer. Effects of polydispersity on the order-disorder transition in block copolymer melts. *Macromolecules*, 40(22):8050–8055, 2007. 1.1

[76] Nathaniel A Lynd, Benjamin D Hamilton, and Marc A Hillmyer. The role of polydispersity in the lamellar mesophase of model diblock copolymers. *J. Polym. Sci. B*, 45(24): 3386–3393, 2007. 1.1

[77] Nathaniel A Lynd, Marc A Hillmyer, and Mark W Matsen. Theory of polydisperse block copolymer melts: Beyond the schulz- zimm distribution. *Macromolecules*, 41(12):4531–4533, 2008. 1.1

[78] Nathaniel A Lynd, Adam J Meuler, and Marc A Hillmyer. Polydispersity and block copolymer self-assembly. *Prog. Polym. Sci.*, 33(9):875–893, 2008. 2.4.1

[79] Sanaz Mahmoodi, Javad Poshtan, Mohammad Reza Jahed-Motlagh, and Allahyar Montazeri. Nonlinear model predictive control of a ph neutralization process based on wiener–laguerre model. *Chem. Eng. J.*, 146(3):328–337, 2009. 1.3

[80] Pawel W Majewski and Kevin G Yager. Millisecond ordering of block copolymer films via photothermal gradients. *ACS Nano*, 9(4):3896–3906, 2015. 1.1

[81] Pawel W Majewski, Atikur Rahman, Charles T Black, and Kevin G Yager. Arbitrary lattice symmetries via block copolymer nanomeshes. *Nat. Commun.*, 6:7448, 2015. 1.1

[82] Krzysztof Matyjaszewski. Atom transfer radical polymerization (atrp): current status and future perspectives. *Macromolecules*, 45(10):4015–4039, 2012. 1.1

[83] Krzysztof Matyjaszewski and James Spanswick. Controlled/living radical polymerization. *Mater. Today*, 8(3):26–33, 2005. 1.1

[84] Krzysztof Matyjaszewski and Nicolay V Tsarevsky. Macromolecular engineering by atom transfer radical polymerization. *J. Am. Chem. Soc*, 136(18):6513–6533, 2014. 1.1

[85] Krzysztof Matyjaszewski and Jianhui Xia. Atom transfer radical polymerization. *Chem. Rev.*, 101(9):2921–2990, 2001. 1.1, 2.1

[86] Krzysztof Matyjaszewski, Timothy E Patten, and Jianhui Xia. Controlled/living radical polymerization. kinetics of the homogeneous atom transfer radical polymerization of styrene. *J. Am. Chem. Soc*, 119(4):674–680, 1997. 2.1

[87] Terry McAfee, Natalie Leonardi, Rick Montgomery, Julia Siqueira, Thomas Zekoski, Michael F Drenski, and Wayne F Reed. Automatic control of polymer molecular weight during synthesis. *Macromolecules*, 49(19):7170–7183, 2016. 1.3

[88] Ke Min, Haifeng Gao, and Krzysztof Matyjaszewski. Preparation of homopolymers and block copolymers in miniemulsion by atrp using activators generated by electron transfer (aget). *J. Am. Chem. Soc*, 127(11):3825–3830, 2005. 1.1

[89] Yutaka Miura, Atsushi Narumi, Soh Matsuya, Toshifumi Satoh, Qian Duan, Harumi Kaga, and Toyoji Kakuchi. Synthesis of well-defined ab20-type star polymers with cyclodextrin-core by combination of nmp and atrp. *J. Polym. Sci. A*, 43(18):4271–4279, 2005. 1.1

[90] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint*, page arXiv:1312.5602, 2013. 2.3

[91] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 1, 1.2

[92] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016. 1.2, 2.3

[93] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016. 3.1

[94] EP Nahas, MA Henson, and DE Seborg. Nonlinear internal model control strategy for neural network models. *Comput. Chem. Eng.*, 16(12):1039–1057, 1992. 1.3

[95] Mohammad Najafi, Hossein Roghani-Mamaqani, Mehdi Salami-Kalajahi, and Vahid Haddadi-Asl. A comprehensive monte carlo simulation of styrene atom transfer radical polymerization. *Chin. J. Polym. Sci.*, 28(4):483–497, 2010. 2.1

[96] Mohammad Najafi, Hossein Roghani-Mamaqani, Vahid Haddadi-Asl, and Mehdi Salami-Kalajahi. A simulation of kinetics and chain length distribution of styrene frp and atrp: Chain-length-dependent termination. *Adv. Polym. Tech.*, 30(4):257–268, 2011. 2.1

[97] Yasuyuki Nakamura, Tasuku Ogihara, and Shigeru Yamago. Mechanism of cu (i)/cu (0)-mediated reductive coupling reactions of bromine-terminated polyacrylates, polymethacrylates, and polystyrene. *ACS Macro Lett.*, 5(2):248–252, 2016. 2.1

[98] Ali Nejati, Mohammad Shahrokhi, and Arjomand Mehrabani. Comparison between backstepping and input–output linearization techniques for ph process control. *J. Process Control*, 22(1):263–271, 2012. 1.3

[99] Yisu Nie, Lorenz T Biegler, and John M Wassick. Integrated scheduling and dynamic optimization of batch processes using state equipment networks. *AIChE J.*, 58(11):3416–3432, 2012. 1.3

[100] Paul Nomikos and John F MacGregor. Monitoring batch processes using multiway principal component analysis. *AIChE J.*, 40(8):1361–1375, 1994. 1.3

[101] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *J. Cheminform.*, 9(1):48, 2017. 1

[102] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data. Eng.*, 22(10):1345–1359, 2010. 1.3

[103] Timothy E Patten, Jianhui Xia, Teresa Abernathy, and Krzysztof Matyjaszewski. Polymers with very low polydispersities from atom transfer radical polymerization. *Science*, 272 (5263):866, 1996. 2.1

[104] Kevin A Payne, Dagmar R Dhooge, Paul HM Van Steenberge, Marie-Francoise Reyniers, Michael F Cunningham, Robin A Hutchinson, and Guy B Marin. Arget atrp of butyl methacrylate: utilizing kinetic modeling to understand experimental trends. *Macromolecules*, 46(10):3828–3840, 2013. 2.1

[105] Andrzej Plichta, Mingjiang Zhong, Wenwen Li, Andrea M Elsen, and Krzysztof Matyjaszewski. Tuning dispersity in diblock copolymers using arget atrp. *Macromol. Chem. Phys.*, 213(24):2659–2668, 2012. 1.1

[106] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de-novo drug design. *arXiv preprint*, page arXiv:1711.10907, 2017. 1

[107] João GD Preturlan, Roniérik P Vieira, and Liliane MF Lona. Numerical simulation and parametric study of solution arget atrp of styrene. *Comput. Mater. Sci.*, 124:211–219, 2016. 1.1, 2.1, 2.1

[108] Thomas G Ribelli, Dominik Konkolewicz, Stefan Bernhard, and Krzysztof Matyjaszewski. How are radicals (re) generated in photochemical atrp? *J. Am. Chem. Soc*, 136(38):13303–13312, 2014. 1.1

[109] Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011. 1.3

[110] Nicholas Roy and Geoffrey J Gordon. Exponential family pca for belief compression in

pomdps. In *Advances in Neural Information Processing Systems*, pages 1667–1674, 2003. 1.2

[111] Andrei A Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. *arXiv preprint*, page arXiv:1610.04286, 2016. 2.2, 4

[112] Traian Sarbu, Koon-Yee Lin, John Ell, Daniel J Siegwart, James Spanswick, and Krzysztof Matyjaszewski. Polystyrene with designed molecular weight distribution by atom transfer radical coupling. *Macromolecules*, 37(9):3120–3127, 2004. 2.4.2

[113] D Sbarbaro-Hofer, D Neumerkel, and K Hunt. Neural control of a steel rolling mill. *IEEE Control Syst.*, 13(3):69–75, 1993. 1.3

[114] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015. 3.2, 3.2

[115] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. 1.2

[116] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint*, page arXiv:1707.06347, 2017. 1.2

[117] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 1

[118] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint*, page arXiv:1712.01815, 2017. 1

[119] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550 (7676):354–359, 2017. 1

[120] Nathan Sprague and Dana Ballard. Multiple-goal reinforcement learning with modular sarsa (0). In *IJCAI*, pages 1445–1447, 2003. 4

[121] Balasubramaniam Srinivasan, Dominique Bonvin, Erik Visser, and Srinivas Palanki. Dynamic optimization of batch processes: Ii. role of measurements in handling uncertainty. *Comput. Chem. Eng.*, 27(1):27–44, 2003. 1.3

[122] Balasubramaniam Srinivasan, Srinivas Palanki, and Dominique Bonvin. Dynamic optimization of batch processes: I. characterization of the nominal solution. *Comput. Chem. Eng.*, 27(1):1–26, 2003. 1.3

[123] Wen Sun, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell. Deeply aggrevated: Differentiable imitation learning for sequential prediction. *arXiv preprint*, page arXiv:1703.01030, 2017. 1.3

[124] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 2000. 2.2

[125] S Syafiie, Fernando Tadeo, and E Martinez. Model-free learning control of neutralization processes using reinforcement learning. *Eng. Appl. Artif. Intell.*, 20(6):767–782, 2007. 1.3

[126] S Syafiie, Fernando Tadeo, E Martinez, and Teresa Alvarez. Model-free control based on reinforcement learning for a wastewater treatment problem. *Appl. Soft Comput.*, 11(1): 73–82, 2011. 1.3

[127] Wei Tang and Krzysztof Matyjaszewski. Effect of ligand structure on activation rate constants in atrp. *Macromolecules*, 39(15):4953–4959, 2006. 1.1

[128] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.*, 10(Jul):1633–1685, 2009. 1.3

[129] Carolina Toloza Porras, Dagmar R D'hooge, Paul HM Van Steenberge, Marie-Françoise Reyniers, and Guy B Marin. A theoretical exploration of the potential of icar atrp for one-and two-pot synthesis of well-defined diblock copolymers. *Macromol. React. Eng.*, 7 (7):311–326, 2013. 2.1

[130] Salomon Turgman-Cohen and Jan Genzer. Computer simulation of concurrent bulk-and surface-initiated living polymerization. *Macromolecules*, 45(4):2128–2137, 2012. 2.1

[131] P Turner, G Montague, and J Morris. Neural networks in dynamic process state estimation and non-linear predictive control. *4th International Conference on Artificial Neural Networks*, pages 284–289, 1995. 1.3

[132] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, pages 2094–2100, 2016. 1.2

[133] Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *J. Mach. Learn. Res.*, 15(1):3483–3512, 2014. 4

[134] Paul HM Van Steenberge, Dagmar R Dhooge, Yu Wang, Mingjiang Zhong, Marie-Francoise Reyniers, Dominik Konkolewicz, Krzysztof Matyjaszewski, and Guy B Marin. Linear gradient quality of atrp copolymers. *Macromolecules*, 45(21):8519–8531, 2012. 1.1

[135] Roniérik P Vieira and Liliane MF Lona. Optimization of reaction conditions in functionalized polystyrene synthesis via atrp by simulations and factorial design. *Polym. Bull.*, 73 (7):1795–1810, 2016. 1.1

[136] Ronierik P Vieira, Andreia Ossig, Janaína M Perez, Vinícius G Grassi, Cesar L Petzhold, Augusto C Peres, Joao M Costa, and Liliane MF Lona. Styrene atrp using the new initiator 2, 2, 2-tribromoethanol: Experimental and simulation approach. *Polym. Eng. Sci.*, 55(10): 2270–2276, 2015. 2.1

[137] Roniérik Pioli Vieira and Liliane Maria Ferrareso Lona. Simulation of temperature effect on the structure control of polystyrene obtained by atom-transfer radical polymerization. *Polímeros*, 26(4):313–319, 2016. 2.1, 2.1

[138] Jin-Shan Wang and Krzysztof Matyjaszewski. Controlled/"living" radical polymerization. atom transfer radical polymerization in the presence of transition-metal complexes. *J. Am. Chem. Soc*, 117(20):5614–5615, 1995. 2.1

[139] Xuezhi Wang and Jeff Schneider. Flexible transfer learning under support and model shift. In *Advances in Neural Information Processing Systems*, pages 1898–1906, 2014. 1.3

[140] Zhenhua Wang, Xiangcheng Pan, Lingchun Li, Marco Fantin, Jiajun Yan, Zongyu Wang, Zhanhua Wang, Hesheng Xia, and Krzysztof Matyjaszewski. Enhancing mechanically induced atrp by promoting interfacial electron transfer from piezoelectric nanoparticles to cu catalysts. *Macromolecules*, 50(20):7940–7948, 2017. 1.1

[141] Zhenhua Wang, Xiangcheng Pan, Jiajun Yan, Sajjad Dadashi-Silab, Guojun Xie, Jianan Zhang, Zhanhua Wang, Hesheng Xia, and Krzysztof Matyjaszewski. Temporal control in mechanically controlled atom transfer radical polymerization using low ppm of cu catalyst. *ACS Macro Lett.*, 6(5):546–549, 2017. 1.1

[142] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016. 3.1

[143] N Watanabe. A comparison of neural network based control strategies for a cstr. *IFAC Proc. Vol.*, 27(2):377–382, 1994. 1.3

[144] Emily Daniels Weiss, Racquel Jemison, Kevin JT Noonan, Richard D McCullough, and Tomasz Kowalewski. Atom transfer versus catalyst transfer: Deviations from ideal poisson behavior in controlled polymerizations. *Polymer*, 72:226–237, 2015. 1.1, 2.1, 2.1

[145] Aide Wu, Zifu Zhu, Michael F Drenski, and Wayne F Reed. Simultaneous monitoring of the effects of multiple ionic strengths on properties of copolymeric polyelectrolytes during their synthesis. *Processes*, 5(2):17, 2017. 1.3

[146] Jiajun Yan, Tyler Kristufek, Michael Schmitt, Zongyu Wang, Guojun Xie, Alei Dang, Chin Ming Hui, Joanna Pietrasik, Michael R Bockstaller, and Krzysztof Matyjaszewski. Matrix-free particle brush system with bimodal molecular weight distribution prepared by si-atrp. *Macromolecules*, 48(22):8208–8218, 2015. 2.4.2

[147] YY Yang and DA Linkens. Adaptive neural-network-based approach for the control of continuously stirred tank reactor. *IEE Proc.-Control Theory Appl.*, 141(5):341–349, 1994. 1.3

[148] BE Ydstie. Forecasting and control using adaptive connectionist networks. *Comput. Chem. Eng.*, 14(4):583–599, 1990. 1.3

[149] Min Zhang and W Harmon Ray. Modeling of living free-radical polymerization processes. ii. tubular reactors. *J. Appl. Polym. Sci.*, 86(5):1047–1056, 2002. 2.4.1

[150] Yang Zheng, Yucheng Huang, and Brian C Benicewicz. A useful method for preparing mixed brush polymer grafted nanoparticles by polymerizing block copolymers from sur-

faces with reversed monomer addition sequence. *Macromol. Rapid Commun.*, 38(19): 1700300, 2017. 2.4.2

[151] Mingjiang Zhong, Yu Wang, Pawel Krys, Dominik Konkolewicz, and Krzysztof Matyjaszewski. Reversible-deactivation radical polymerization in the presence of metallic copper. kinetic simulation. *Macromolecules*, 46(10):3816–3827, 2013. 1.1

[152] Zhenpeng Zhou, Xiaocheng Li, and Richard N Zare. Optimizing chemical reactions with deep reinforcement learning. *ACS Cent. Sci.*, 3(12):1337, 2017. 1

[153] Shiping Zhu. Modeling of molecular weight development in atom transfer radical polymerization. *Macromol. Theory Simul.*, 8(1):29–37, 1999. 2.1