Hybrid Feedback and Reinforcement Learning-Based Control of Machine Cycle Time for a Multi-Stage Production System

Chen Li[a], Qing Chang[a,*]

[a]*Department of Mechanical and Aerospace Engineering, University of Virginia, Charlottesville, VA, 22904, US*

Abstract

With the increasing need of flexible manufacturing systems, machine flexibility has become one of the major impact factors. An important aspect of machine flexibility is the ability to change an individual machine's capacity (or cycle time) to improve the overall system efficiency. In this paper, a novel control method is proposed for multi-stage production systems to dynamically change the individual machines' cycle time to improve overall system efficiency. The proposed control method integrates distributed feedback control scheme and a Reinforcement Learning (RL) control scheme based on an extended actor-critic algorithm. The feedback control will determine whether a machine is turned on or off using real-time system status, while the RL control scheme will decide how to increase or decrease a machine's cycle time when a machine is on. An improved actor-critic RL algorithm is developed to add an auxiliary model-based path to the standard model-free RL to enhance the learning performance. To demonstrate the effectiveness of the proposed method, numerical case studies have been performed that clearly show improvements in the overall profits and energy savings compared to other methods.

Keywords: Machine Cycle Control, Production Loss, Deep Reinforcement Learning, Markov Decision Process (MDP)

1. Introduction

The cycle time of a machine is the time it takes to complete a designated process on one part. It is a crucial parameter that determines the efficiency of a machine and, consequently, the overall performance of a production system. Usually, a machine's cycle time is determined based on the production line design, and it is typically unchangeable once determined due to efficiency and quality consideration for mass production [1]. Therefore, a large amount of research has been focused on reducing machine downtime, optimizing resource distribution, and maintenance management to achieve the goal of improving the system efficiency. With the rapid development of variable speed drive techniques (i.e., variable speed drives and variable frequency drives) and the recent advancement in learning-based control techniques, changing the cycle time of each machine in a manufacturing system is increasingly possible. For instance, Siemens launched a series of production automation modules called SINAMICS that allows for the speed of each machine to be adjusted by changing the operating voltage of motors [2]. In addition, machines in many flexible industrial processes such as machining lines and assembly lines can operate with different

*Corresponding Author.
Email address: qc9nq@virginia.edu (Q. Chang)

cycle times for different product types. Thus, changeable cycle time offers machines and production lines considerable flexibility and adaptability, further opening a new pathway to enhance system performance.

However, managing the cycle time change of machines in a production line is not trivial. A serial production line with multiple machines and buffers is a nonlinear (due to limited buffer) and stochastic (due to, e.g., random downtimes) system. There is no closed-form representation for such a system [3]. This makes the classic control theory hard to apply. In addition, the nonlinear and stochastic dynamics of the production system will lead to a vast state space for the control problem, which may include each machine's cycle time, operating status, and intermediate buffer level. A production line with variable cycle time machines will further increase the states and complexity of the system. This makes the traditional mathematical programming methods intractable. A Markov Decision Process (MDP) [4] can be used to formulate control problems like these, and model-free reinforcement learning (RL) has proven to be effective since it does not require detailed state transition models but relies on continual interactions with the environment to improve its policy [5]. However, a pure learning-based control policy may not be able to enjoy the advantage of the system property or expert knowledge about the system, resulting in lower learning performance. It could be time-consuming to train an RL-based control policy considering all parameters and input data, assuming no knowledge about the system.

In view of these challenges, this paper is devoted to developing an intelligent control scheme by changing machines' cycle time and on/off status to facilitate a flexible operation of machines and maximize the profit of the system. The main contributions of this paper are in 1) formulating an MDP problem and developing a hybrid control scheme that combines a quick distributed feedback control and a deep RL (DRL)-based control scheme; 2) utilizing a distributed feedback control scheme for each machine on/off status by adopting the system property of opportunity window [6] based on the manufacturing system-level understanding; 3) developing an extended actor-critic algorithm to improve DRL performance for controlling machines' cycle time change.

The rest of this paper is organized as follows: Section 2 gives a brief overview of related literatures; Section 3 describes the structure of a serial production line with variable cycle time machines. Section 4 briefly introduces the production system model and system property; The hybrid control scheme is presented in Section 5. Section 6 discusses the integration of the feedback control and the improved DRL to obtain a better policy with higher efficiency. A case study is presented in Section 7 to show the effectiveness of the proposed method. Section 8 summarizes the conclusion of this paper and provides future directions.

2. Literature Review

There have been substantial research efforts devoted to improving the overall manufacturing system efficiency through the scheduling and control of machines' operations. With the development in modeling and analytical analysis of manufacturing systems, the analytical- and model-based control strategies have become one of the

mainstream approaches to promoting system efficiency. For example, a state-based feedback control policy has been developed to switch machines on or off for a Bernoulli production line in [7] to reserve energy, where the on/off control is decided based on the occupancy of the buffers using the Markov chain model. In [8], Markovian analysis with aggregation approximation is used in deriving the so-called N-policy to maintain desirable productivity while reducing the energy consumption of the system. These model-based methods typically need strong machine reliability assumptions (e.g., Bernoulli or Geometric machines) and are difficult to generalize to more complicated realistic scenarios.

With advancements in machine/process sensing and big data analytics, data-driven approaches have seen increasing applications in smart manufacturing [9]. Utilizing real-time production data, Liu et al. [10] have quantitively analysed the downtime impact on system real-time performance and overall cost. Paul et al. [11] have formulated a dynamic solution approach that can deal with multiple disruptions to reschedule the production plan on a real-time basis. Based on the sensor reading, Zou et al. [12] have developed an automated distributed feedback production control scheme by switching each machine on or off to improve overall system productivity and profit. In [13], a real-time resilient adaptive control policy is developed to deliver resilient performance against random disruption events based on a real-time system performance diagnostic by unitizing both system properties and sensor information. Li et al. [14] proposed an event-based control methodology that can strategically switch machines to the energy-saving mode for energy-efficient production. To achieve higher production efficiency, Frazzon et al. proposed a data-driven adaptive planning and control approach that uses simulation-based optimization to determine the most suitable dispatching rules under varying conditions [15]. However, in most of these studies, the production systems considered consist of only unchangeable cycle time machines. There has been not enough discussion on smartly adjusting machines' cycle time to improve system performance and flexibility.

Changing the cycle time of machines for a stochastic production line needs to deal with large state space and is a real-time decision-making problem, which can be formulated as a RL problem. MDP is a typical framework that can solve most RL problems [4]. The objective of MDP is to find an optimal policy that gives the best action for each state. The most widely used methods to solve MDP can be categorized into model-free RL and model-based RL.

Model-based RL rely on a system model that evaluates the outcomes of the current action. On the other hand, model-free RL uses real-world experiences obtained from environments and does not require the transition probability distribution (and the reward function). The emergence of deep reinforcement learning (DRL) method allows the standard RL to incorporate deep learning (DL). The state-of-the-art DRL method has demonstrated a great potential to solve complex manufacturing problems, such as NP-hard production scheduling [16]. Recent studies propose to combine the strengths of both model-free and model-based approaches. In [17], an integrated control of production and preventive maintenance has been formulated as a semi-MDP and an infinite time horizon dynamic programming algorithm has been adopted to solve the control problem. Ou et al. [5] have developed an innovative

method, "Q-ADP", which combines a model-free Q-learning and an approximate dynamic programming to optimally assign gantry robots in a work cell. Racanière et al. [18] propose a novel I2As architecture for DRL by combining model-free and model-based techniques to improve data efficiency and robustness of the model. Inspired by these ideas, we propose an improved DRL algorithm that embeds an auxiliary model-based path to the standard model-free DRL algorithm to smartly change machines' cycle time.

Industrial manufacturing is becoming more automated and complex. Therefore, control problems need to deal with increasingly complex environments and handle variability and uncertainty. Hybrid control strategies are emerging as a promising way to solve real world control problems [19]. There are various ways of performing hybrid control. In [20], a problem of solving combined discrete decision variables (e.g., digital outputs) and continuous decision variables (e.g., velocity setpoints) are addressed. To improve the control accuracy and efficiency, Neunert et al. [20] have defined a hybrid RL problem to optimize for discrete and continuous actions simultaneously. Hybrid automation with balanced introduction of humans and robots are also on the rise. Astudillo et al. [21] have introduced a hybrid framework of a manufacturing control system that integrates human activities in the manufacturing process by employing a human decisional entity as a virtual component. These control methods only introduce additional module or component to the conventional or RL-based control scheme without integrating different control methods. In [19], a robotic control problem involving contacts and unstable objects is introduced. It can be difficult for conventional feedback control to account for contact behavior and RL requires a considerable amount of data before converging to adequate performance. Therefore, Johannink et al. [19] have developed a hybrid control methodology that combines the strength of both feedback control and deep RL methods to achieve an assembly task by relying on the engineered controller as a starting point and using RL to correct for the controller's mistakes.

To the best of our knowledge, the applications of hybrid control methods to manufacturing system-level decision makings (e.g., maintenance or machines' cycle time change) have yet to be sufficiently studied, albeit their powerful advantages. Motivated by these prior efforts, this paper presents a hybrid feedback control and an improved DRL-based control scheme to optimally change the cycle time of machines on the production line to achieve a higher system profit.

3. System Description

3.1 Notations and assumptions

A schematic of a multi-stage production line with M machines and $M-1$ buffers is shown in Figure 1, where rectangles represent machines and circles represent buffer. The following notations are adopted in this work:

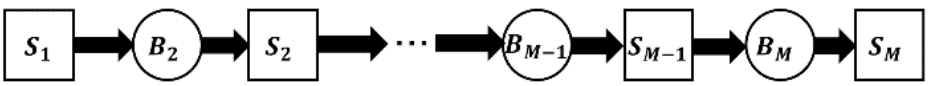

Figure 1. The structure of a serial production line

1) S_i represents the i^{th} machine, $i = 1, 2, \dots, M$;
2) $T_i(t)$ denotes the cycle time of machine S_i at time t, $i = 1, 2, \dots, M$;
3) $S_{M^*}(t)$ denotes the slowest machine of the production line at time t, and $T_{M^*}(t)$ denotes the cycle time of the slowest machine at time t;
4) B_i denotes the i^{th} buffer, $i = 2, 3, \dots, M$, and it is also used to represent the buffer capacity of the i^{th} buffer for convenience;
5) $b_i(t)$ represents the buffer level of B_i at time t;
6) $\vec{e}_i = (j, t_i, d_i)$ denotes i^{th}disruption event that machine S_j is down at time t_i with time duration of d_i, where $i = 1, 2, \dots, n$, and $j = 1, 2, \dots, M$;

We make the following assumptions in this paper:

1) A machine is starved if it is on, and its upstream buffer is empty;
2) A machine is blocked if it is on, and its downstream buffer is full;
3) The cycle time of a machine can only be changed within a range that is constrained by the design of the machine, and therefore $T_i(t) \in [T_{i,L}, T_{i,U}]$, where $T_{i,L}$ and $T_{i,U}$ are the lower bound and the upper bound for machine S_i respectively.

4. Introduction of System Model and System Property

Our previous study [6] has developed a data-enabled model for serial production lines with variable cycle time machines, which efficiently evaluates the real-time states of the system. To keep this paper self-contained, the main conclusions and the modeling methodology are briefly introduced without detailed derivation.

The manufacturing system with variable cycle time machines can be modelled with the following state-space equation:

$$\dot{\boldsymbol{X}}(t) = \boldsymbol{F}(\boldsymbol{X}(t), \boldsymbol{W}(t), \boldsymbol{U}(t)) \tag{1}$$

- $\boldsymbol{X}(t) = [X_1(t), X_2(t), \dots, X_M(t)]'$, where $X_i(t)$ is the production count of machine S_i up to time t;
- $\boldsymbol{W}(t) = [W_1(t), W_2(t), \dots, W_M(t)]'$, where $W_i(t)$ denotes whether machine S_i is suffering a disruption event at time t. If there exists a random disruption event $\vec{e}_k = (i, t_k, d_k)$ and $t \in [t_k, t_k + d_k]$, then $W_i(t) = 1$, otherwise, $W_i(t) = 0$;
- $\boldsymbol{U}(t) = [u_1(t), u_2(t), \dots, u_M(t)]'$ is the control input at time t.

In this paper, the control input of a machine is the machine operation status, including machine on, off, and cycle time increasing and decreasing. We assume each machine's cycle time can be increased or decreased by a ratio φ, where $\varphi \in [0,1]$. Let $\boldsymbol{u^c}(t) = [u_1^c(t), u_2^c(t), \dots, u_M^c(t)]'$ denote the cycle time control input, we have

$$u_i^c(t) = \begin{cases} (1+\varphi), & \text{increase the cycle time of machine } S_i \text{ with the ratio of } \varphi \text{ at time } t \\ (1-\varphi), & \text{decrease the cycle time of machine } S_i \text{ with the ratio of } \varphi \text{ at time } t \end{cases} \quad (2)$$

Let $\boldsymbol{u^e}(t) = [u_1^e(t), u_2^e(t), \dots, u_M^e(t)]'$ denote the on/off control input, we have

$$u_i^e(t) = \begin{cases} 0, & \text{turn machine } S_i \text{ off at time } t \\ 1, & \text{turn machine } S_i \text{ on with its designed cycle time at time } t \end{cases} \quad (3)$$

Therefore, the control input to machine S_i at time t is

$$u_i(t) = u_i^c(t) \cdot u_i^e(t) \quad (4)$$

Machine S_i is operational when there is no disruption event, and it is not turned off. Let $\vartheta(t) = [\vartheta_1(t), \vartheta_2(t), \dots, \vartheta_M(t)]'$ denotes the on/off status of each machine, where

$$\vartheta_i(t) = \big(1 - W_i(t)\big) \cdot u_i^e(t) \quad (5)$$

Based on the conservation of flow, a quick recursive algorithm has been developed to evaluate the system state $\boldsymbol{X}(t)$ and $\boldsymbol{b}(t)$, i.e., production count of each machine and buffer level of each buffer at time t. The analytical derivation of the dynamic functions $\boldsymbol{F}(*)$ can be found in [6].

In order to real-time control the machine operation, we need a metric to gauge the system's dynamic performance. It has been proven from our previous study [6] that not all the downtimes will lead to permanent production loss (PPL) of the system, and only the downtime that causes the stoppage of the slowest machine, S_{M^*}, will result in PPL. To quantify such property, we have developed the concept of opportunity window (OW). The OW for machine S_i, denoted as OW_i, is defined as the longest possible downtime on S_i that will not cause the PPL of the system [6]. It has been shown that OW_i is the time it takes for all the buffers between S_i and the slowest machine S_{M^*} to be empty or full [6]. Based on the definition of OW_i, PPL due to downtimes on machine S_i is defined as the amount of production loss due to an extended downtime that exceeds its threshold (i.e., OW_i). If machine S_i suffers from a downtime event with a duration d, then PPL due to d can be evaluated as $PPL_i = \max\{\frac{d - OW_i(t)}{T_M^*(t)}, 0\}$. We have proven that this production loss is the same for every machine on the line that cannot be recovered in any circumstances [6]. The details of derivation and rigorous proof of OW and PPL can be found in our pervious study [6]. OW_i and PPL_i are important system dynamic properties that we will use in designing the control policy in this paper.

5. Control Problem Formulation

Production systems with changeable cycle time machines offer more flexibility and adaptability, but the overall complexity is increased. Therefore, controlling such systems becomes more challenging. In this paper, the control actions will be to turn each machine on/off and/or change its cycle time as in Eq. (4). Changing machines' cycle time in a stochastic production line is a real-time decision-making problem, which can be formulated as an MDP problem [4]. The target of the control problem is to find an optimal policy under stochastic situations, by mapping from states

to actions (i.e., cycle time change), so as to maximize a reward.

There are various methods to solve MDP problems. In this work, the system state consists of all machines' states and buffers' states, which can grow exponentially with the number of machines and buffers. Furthermore, the system may also have a large action space due to the extra flexibility in machine cycle time adjustments. For example, for a 10-machine production line, the action space will be $4^{10} = 1048576$ assuming each machine has four actions (i.e., on, off, increasing its cycle time, and decreasing its cycle time). Therefore, conventional algorithms, such as dynamic programming, are difficult to apply for controlling the machine's operational status [6]. In addition, although we have developed a fast recursive algorithm to evaluate system real-time output and PPL, there is no closed-form representation for the production lines. In view of this challenge, we propose a hybrid control scheme that combines a quick distributed feedback control and a DRL-based control, as shown in Figure 2.

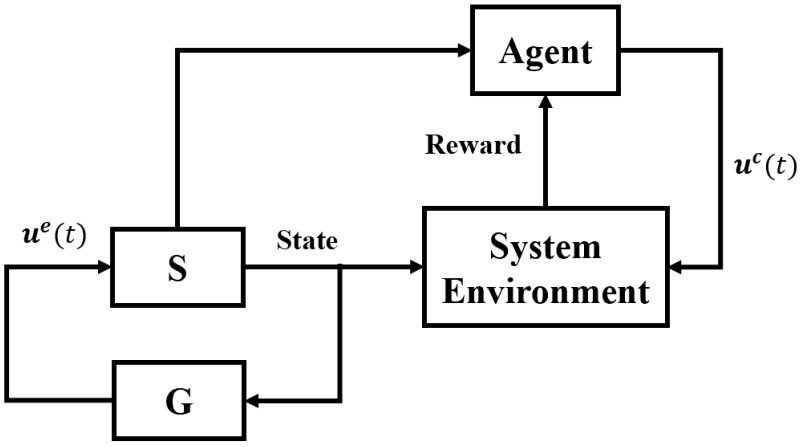

Figure 2. The control diagram of the hybrid control framework

Figure 2 represents the hybrid control structure for this MDP problem. $\boldsymbol{S}$ denotes the state of the system with variable cycle time machines. It includes each machine's cycle time, disruption events, and system buffer levels. The state space dimension depends on the number of machines and buffers. Let $s_t \in \boldsymbol{S}$ denote the system states of each machine at time t, we define

$$\boldsymbol{S} = [\boldsymbol{T}(t), \boldsymbol{W}(t), \boldsymbol{\vartheta}(t), \boldsymbol{b}(t)] \tag{6}$$

where each element of the state is:

- $\boldsymbol{T}(t) = [T_1(t), T_2(t), \dots, T_M(t)]$ denotes the cycle time of each machine at time t;
- $\boldsymbol{W}(t) = [W_1(t), W_2(t), \dots, W_M(t)]$ denotes whether machines are suffering a disruption event at time t;
- $\boldsymbol{\vartheta}(t) = [\vartheta_1(t), \vartheta_2(t), \dots, \vartheta_M(t)]$ denotes the operating status of machines at time t;
- $\boldsymbol{b}(t) = [b_2(t), b_3(t), \dots, b_M(t)]$ denotes the buffer levels of each machine at time t.

For the hybrid control framework, the control inputs $\boldsymbol{u_i}(t)$ include $\boldsymbol{u_i^e}(t)$, i.e., a machine's on/off action, and $\boldsymbol{u_i^c}(t)$, i.e., machines' cycle time change action. To reduce the state and action space, we will leverage the system property to control a machine's on/off input $\boldsymbol{u^e}(t)$. A distributed feedback control will be designed for each machine based on the system property of OW as discussed in Section 4. For machine's cycle time change, i.e., $\boldsymbol{u^c}(t)$, we propose to use DRL due to the NP-hard nature of the problem and a lack of closed-form representation for general production lines. This hybrid control scheme can take advantage of the well-developed feedback control using our understanding of the production system and the flexibility of RL to effectively reduce the action space to facilitate efficient online control.

The goal of the control is to maximize the profit, or in another word, to minimize the profit loss of the system. The cost function of the system can be formulated as:

$$C_{total} = C_{PPL} + C_{energy} + C_{sw} \tag{7}$$

where C_{PPL} stands for the profit loss due to the permanent production loss, C_{energy} denotes the energy cost, and C_{sw} indicates the control cost when executing actions.

The profit loss of the whole system due to PPL within the time period $[0, T]$ can be written as:

$$C_{PPL} = c_p \cdot \sum_{j=1}^{M} PPL_j[0, T] \tag{8}$$

where c_p is the profit per part produced by the system, and PPL_j is evaluated as discussed in Section 5.

Energy cost C_{energy} is evaluated by the cost of energy consumed by all the machines within the system during time period $[0, T]$, and can be represented as

$$C_{energy} = c_e \cdot \int_0^T \sum_{j}^{M} I_j \, \vartheta_j(t) dt \tag{9}$$

where c_e is the electricity rate, θ_j is the on/off status of machine S_j, and I_j is the power consumption rate of machine S_j.

We also define a penalty cost for frequent control action to change the status of a machine. This cost accounts for the potential damage and extra energy consumption when a machine's cycle time or on/off status is changed. Assume the control cost of machine S_j at time t is $C_{j,sw}(t)$, then the control cost of the production line during time period $[0, T]$ is

$$C_{sw} = \int_0^T \sum_{j}^{M} C_{j,sw}(t) \, dt \tag{10}$$

6. Hybrid Control Method Description

Based on the control framework in Section 5, we first introduce the distributed feedback control scheme that takes the system property of OW to determine whether a machine should be turned on or off at time t. Then an extended DRL-based algorithm will be designed to change the cycle time of each machine. Finally, the hybrid control method is developed that integrates the feedback control and DRL-based control.

6.1. Feedback Control for Each Machine

As shown in Figure 2, the control input for the feedback control part $\boldsymbol{u^e}(t)$ are decided by the current states of the production system, and we have:

$$\boldsymbol{u^e}(t) = G\big(\boldsymbol{T}(t), \boldsymbol{W}(t), \boldsymbol{\vartheta}(t), \boldsymbol{b}(t)\big) \tag{11}$$

For the feedback control part, the control inputs are switching each machine on or off. The feedback policy $\boldsymbol{G}(*)$ depends on the current system state described above. Based on the system assumption, the machine $S_i, \forall i \in \{1, \dots, M\}, and\ i \neq M^*$ can be (partially) starved or blocked only when its immediate downstream buffer is full, or its immediate upstream buffer is empty. Thus, S_i can be switched off based on the buffer level of its upstream and downstream buffers, i.e., when, its downstream buffer is full, or its upstream buffer is empty.

In addition, as we discussed in Section 4, the opportunity window indicates the ability of the system to withstand a disruption event without causing PPL. We can set a desired system resilience threshold values $o_1, \dots, o_M$, based on either expert knowledge of the system or the system's historical performance as detailed in [13]. Assuming the existence of a set of threshold values and using this set as a control goal, then conventional distributed feedback control can be applied to each machine. The machine S_i can be turned off when its opportunity window goes beyond the threshold value, i.e., $OW_i(t) \geq o_i$, otherwise, the machine will keep at "on" status. The feedback control scheme is shown in Algorithm 1.

Algorithm 1 Feedback Control Algorithm for Each Machine

For each action time step t:

1. For $j = M^*(t)$, $u_j^e(t) = 1$
2. For $j = 1 \neq M^*(t)$
   - if $b_{j+1}(t) = B_{j+1}$ and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$
   - if $OW_j(t) < o_i$, $b_{j+1}(t) < B_{j+1}$, and $\vartheta_j(t) = 0$, then $u_j(t) = 0$, and $T_j(t) = T_{j,or}$
3. For $j = M \neq M^*(t)$
   - if $b_{j-1}(t) = 0$ and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$
   - if $OW_j(t) < o_i$, $b_{j-1}(t) > 0$, and $\vartheta_j(t) = 0$, then $u_j(t) = 0$, and $T_j(t) = T_{j,or}$
4. For $\forall j \in \{2, \dots, M-1\}$, $j \neq M^*(t)$
   - if $b_{j-1}(t) = 0$ or $b_{j+1}(t) = B_{j+1}$, and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$

- if $OW_j(t) < o_i$, $b_{j-1}(t) > 0$, $b_{j+1}(t) < B_{j+1}$, and $\vartheta_j(t) = 0$, then $u_j^e(t) = 0$, and $T_j(t) = T_{j,or}$

6.2. Reinforcement Learning Based Control

6.2.1. Problem Formulation

According to our hybrid control framework, a reinforcement learning (RL)-based control will be used to control the machines' cycle time change. A commonly used framework for RL is MDP, which is a stochastic decision process for decision making under stochastic environment.

An MDP is represented by a tuple $\langle S, A, P_{ss'}^a, R_{ss'}^a \rangle$, which contains a set of states S, a set of actions A, transition probabilities $P_{ss'}^a$ and reward function $R_{ss'}^a$. At each time step, the process is in some state s, and the decision-maker may choose any action a that is available in state s. At the next time step, the process moves to a new state s' with the transition probability $P_{ss'}^a$ and giving the decision-maker a corresponding reward $R_{ss'}^a$. The MDP problem aims to find out a "policy" $\pi(a|s)$, which indicates the specific action that will be taken in a certain state s. For the speed changing problem, the task is to find the optimal policy π^* that maximizes the cumulative reward. Due to the large state space in this cycle time control problem, a DRL algorithm will be applied.

6.2.2. Action Set

Each machine can either increase, decrease, or stay at its current cycle time. Therefore, let a_t denotes the machine control action at time t, we define

$$a_t = \boldsymbol{u}^c(t) = [u_1^c(t), u_2^c(t), \dots, u_M^c(t)] \tag{12}$$

6.2.3. Reward Function

For the MDP problem, $R(s_{t+1}, s_t, a_t)$ is the immediate reward received after the transition from state s_t to state s_{t+1}, due to action a_t. Therefore, for the serial production line, the overall reward function should be able to map each perceived state-action pair to a certain value. Since the control strategy aims to minimize losses due to production loss, control action, and energy consumption, the reward function can be formulated as the negative form of the total cost of the system. By evaluating the effectiveness of the proposed data-enabled model and the energy economics of the manufacturing system, the reward functions during two consecutive decision points are defined as follows:

$$r_t = -C_{total}\,(t) \tag{13}$$

where $C_{total}(t)$ is the stepwise total cost of the production line, as defined in Eq. (7), at time t.

6.2.4. Extended DRL with Model-Based Value Expansion

Typical RL methods fall into two categories: model-based and model-free methods. The model-based methods use the model's predictions or distributions of the next state and reward to calculate optimal actions. For the model-based method, the performance of the learned policy is strongly dependent on the accuracy of the model. However, for the cycle time changing problem, since the control inputs, $\boldsymbol{u}^{c}(t)$, are highly coupled with the system dynamics, the transition probabilities of the machine states cannot be fully provided, meaning the model is not available.

On the contrary, model-free methods, such as Q-learning and Actor-Critic, are model-free algorithms that are purely sampled from experiences. They rely on actual samples from the environment and never use generated predictions of the following state and next reward to alter behavior. However, the model-free DRL does not take into account some system properties that could influence policy exploration. Thus, its estimated state value is inaccurate in some cases.

To deal with these limitations, we seek a more rigorous and efficient way to control the machine cycle time. We propose an extended DRL control method that takes advantages of both model-based method and model-free method by combining the model-based path and the model-free path, as illustrated in Figure 3.

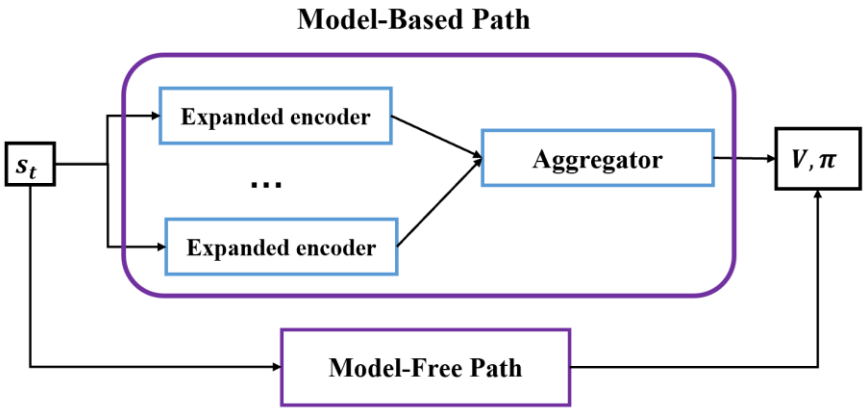

Figure 3. The MVE-DRL architecture

For the model-based path, we propose a model-based value expansion (MVE) method to predict the state value at the next time step conditioned on an action sampled from the expanded policy. Inspired by the idea of I2As, we use an expanded encoder that processes the model-based expansion as a whole and learns to interpret it [18]. Figure 4 shows a zoom-in illustration of the model-based path, where MVE is used to produce n trajectories and n is the number of the available actions. The actions are chosen in each MVE resulting from an expanded policy $\hat{\pi}$. The agent model and the expansion policy $\hat{\pi}$ constitute the MVE to predict the next step. We expand the agent model over multiple steps into the future. In this paper, the encoder uses a long short-term memory (LSTM) architecture to process each trajectory. The features are fed to the LSTM from the end of the trajectory to mimic the dynamic programming, which utilizes the information in reversed order. The model-based path provides additional information and more expressive power. The model-free path only takes the objective observation as input. Therefore,

the extended DRL method can learn to combine information from its model-free path and model-based path.

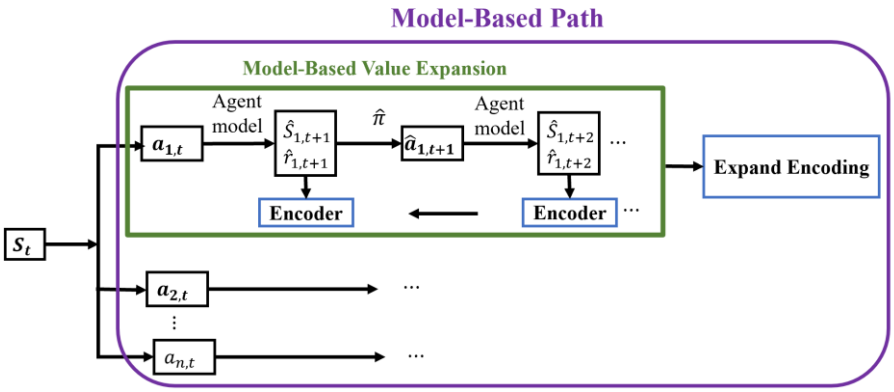

Figure 4. The structure of a single trajectory of Model-based value expansion

6.2.5. Extended Advantage Actor Critic (A2C) Implementation

In this paper, we use the A2C algorithm [22] to implement the above extended DRL algorithm. The extended A2C is proposed to improve the state value estimation by combining a model-based path when updating the advantage function.

The standard model-free A2C algorithm aims to achieve a near-optimal policy by iteratively updating the policy parameter θ. Policy gradients are used to update the parameter:

$$g(\theta) = A(s_t, a_t)\nabla_\theta \log \pi(a_t|s_t, \theta) \tag{14}$$

where $A(s_t, a_t)$ is an estimate of the advantage function, which can be evaluated as:

$$A(s_t, a_t) = r_t - \big(V(s_t; \theta) - \gamma V(s_{t+1}; \theta)\big) \tag{15}$$

The estimation of the value function $V(s_t; \theta)$ of the pure model-free A2C relies on the interactive interaction with the environment. The accuracy of the estimation cannot be guaranteed. However, for the extended A2C algorithm, there is an auxiliary model-based path with the expansion policy $\hat{\pi}$ which sampled from the MVE process for each trajectory. This model-based path is also trained by updating the expansion policy parameter $\hat{\theta}$. Its value function $V(s_t; \hat{\theta})$ is also evaluated as the output of the neural network. For the MVE process with l steps, its corresponding advantage function is formulated as:

$$\hat{A}(s_t, a_t) = \left(\sum_{t<t'\le t+l} \gamma^{t'-t} r_{t'}\right) + \gamma^{l+1} V\big(s_{t+l+1}; \hat{\theta}\big) - V\big(s_t; \hat{\theta}\big) \tag{16}$$

where γ is the discount factor, $r_{t'}$ is the instant reward at time t'. Therefore, the gradient of the expansion policy parameter is:

$$g(\hat{\theta}) = -\hat{A}(s_t, a_t)\partial_{\hat{\theta}} V\big(s_t; \hat{\theta}\big) \tag{17}$$

The cross-entropy loss between the extended A2C policy $\pi(s_t)$ as computed on the current observation, and

the expansion policy $\hat{\pi}(s_t)$ as computed on the same observation, will be added to the total loss function as an auxiliary loss. By imitating the extended A2C policy, the internal expansion will be similar to the trajectories of the agent in the real environment, which would enhance the accuracy of the value function estimation. This also ensures that the expansion corresponds to trajectories with high rewards [18].

To summarize, for the extended A2C algorithm, both the state value and policy are determined based on the aggregation of model-free and model-based paths. Therefore, the proposed extended A2C algorithm can make learning more reliable and closer to a "true optimal" policy and can estimate state value function more accurately than standard A2C. The extended A2C algorithm for machine cycle time control is shown in Algorithm 2.

Algorithm 2 Extended A2C Algorithm

Input: η, θ, $\hat{\theta}$

Initialize θ, $\hat{\theta}$

Repeat

Generate an episode τ following $\pi_\theta(\cdot)$

For each action time step t:

1. Perform action a_t according to policy $\pi_\theta(\cdot)$

2. Get state s_{t+1}, reward r_t from environment

Repeat n times (n: number of trajectories)

Repeat l times (l: number of time steps on a trajectory)

Expand for each time step

Advantage function: $\hat{A}(s_t, a_t) = \left(\sum_{t<t'\leq t+l} \gamma^{t'-t} r_{t'}\right) + \gamma^{l+1} V\left(s_{t+l+1}, \hat{\theta}\right) - V\left(s_t, \hat{\theta}\right)$

Update $g(\hat{\theta}) = g(\hat{\theta}) - \hat{A}(s_t, a_t)\partial_{\hat{\theta}} V\left(s_t, \hat{\theta}\right)$

Expand encoding for each trajectory

3. Aggregate for all trajectories

4. Update $V(s_t; \theta)$

5. Evaluate advantage function: $A(s_t, a_t) = r_t - (V(s_t, \theta) - V(s_{t+1}, \theta))$

6. Update $g(\theta) = g(\theta) + A(s_t, a_t)\nabla_\theta log\pi(a_t|s_t, \theta)$

7. $s_t \leftarrow s_{t+1}$

6.3. Hybrid Control by Integrating Distributed Feedback Control and Extended DRL Control

The extended A2C algorithm is used in managing the machines' cycle time change and the distributed feedback control takes advantage of the system property of OW. These two control schemes are simultaneously trained to derive an optimal control policy. The hybrid control framework can effectively reduce the action space for this control problem. The hybrid integrated control scheme is shown in Algorithm 3.

Algorithm 3 DK-DRL Algorithm for Machine Controlling

Input: η, θ, Q

Initialize θ, Q,$\hat{\pi}$, $\hat{Q}$

Repeat

Generate an episode τ following $\pi_\theta(\cdot)$

For each action time step t:

1. For $j = M^*(t)$, $u_j(t) = 1$
2. For $j = 1 \neq M^*(t)$
   - if $b_{j+1}(t) = B_{j+1}$ and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$
   - if $OW_j(t) < o_i$, $b_{j+1}(t) < B_{j+1}$, and $\vartheta_j(t) = 0$, then $u_j(t) = 0$, and $T_j(t) = T_{j,or}$
   - otherwise, execute steps 1 to 6 of **Algorithm 2**
3. For $j = M \neq M^*(t)$
   - if $b_{j-1}(t) = 0$ and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$
   - if $OW_j(t) < o_i$, $b_{j-1}(t) > 0$, and $\vartheta_j(t) = 0$, then $u_j(t) = 0$, and $T_j(t) = T_{j,or}$
   - otherwise, execute steps 1 to 6 of **Algorithm 2**
4. For $\forall j \in \{2, \dots, M-1\}$, $j \neq M^*(t)$
   - if $b_{j-1}(t) = 0$ or $b_{j+1}(t) = B_{j+1}$, and $\vartheta_j(t) = 1$, then $u_j^e(t) = 0$
   - if $OW_j(t) < o_i$, $b_{j-1}(t) > 0$, $b_{j+1}(t) < B_{j+1}$, and $\vartheta_j(t) = 0$, then $u_j^e(t) = 0$, and $T_j(t) = T_{j,or}$
   - otherwise, execute step 1 to 6 of **Algorithm 2**
5. $s_t \leftarrow s_{t+1}$

7. Case Study

In this section, simulation experiments are performed to validate the effectiveness of the proposed hybrid control scheme. The following case study compares the production performance of the control policies learned from the standard A2C, extended A2C algorithm, and the proposed hybrid control scheme. We intend to show three results: 1) with the same training duration, the policy learned by the proposed hybrid control scheme has the best performance in improving the system profits; 2) the proposed hybrid control scheme outperforms other pure learning based algorithms in the training process (i.e., reaching a stable reward faster); 3) the impact of cycle time changing ratio on training and the execution performance; 4) the impact of control interval on training and the execution performance.

7.1. Parameters setting and training progress

The simulation experiments are carried out on Python platform. We construct and investigate 100 different

serial productions lines, including automotive assembly lines, semi-conductor lines, and battery lines etc. The effectiveness and robustness of the proposed hybrid control scheme have been successfully validated with these 100 different lines. For demonstration purposes, we present a detailed numerical case study on a serial production line with four machines and three buffers, similar to Figure 1. System parameters for machines and buffers are shown in Table I, which are mocked up data based on a real production line for confidential consideration.

Table I. Parameters for the Four-Machine and Three-buffer in the Production Line System

Parameters	**Machines**			
	S_1	S_2	S_3	S_4
Designed cycle time(sec)	50	70	60	40
Cycle time upper bound(sec)	75	105	90	60
Cycle time lower bound(sec)	25	35	30	20
Profit per part c_p ($)	50	50	50	50
electricity rate c_e ($)	5	5	5	5
Power consumption rate I	10	6	8	12
MTBF(min)	24.2	13	24.4	16.1
MTTR(min)	1.3	2	1.4	2
	Buffers			
	B_2		B_3	B_4
Buffer capacity B_i	12		30	48

Given the system parameters, we implement Algorithm 3 to train the control policy. The control decision is triggered for each machine when it finished its current process. The control interval is the number of cycles for each machine between two consecutive control actions. For example, a control interval of two cycles indicates that the control decision is made when a machine completes two parts. During the initial training, the control interval is 4 cycles, and the time step for each trajectory of the MVE process is 1 min. The cycle time changing ratio is set to be $\varphi = 0.1$. The neural network has two fully connected hidden layers, and each layer has 32 hidden units. The reward function is based on Eq. (13), where η is set as 0.00025 after multiple experiments. The discount factor γ for the model value expansion is set to 0.95. The learning rate of actor α_a is set to 0.0005, and the learning rate of the critic α_c is set to 0.001. For the MVE process, we generate $n = 4$ sample trajectories between every two consecutive actions, and each path rollouts $l = 1$ step. With the completion of the training process, a machine control policy is derived based on the updated network parameter θ.

To evaluate the performance of training and the learned policy π based on the proposed hybrid control scheme, two other control schemes are considered, including the standard A2C and an extended A2C schemes.

1) Standard A2C scheme.

In this scenario, a standard model-free advantage actor-critic (A2C) algorithm will be used for training to obtain

a control policy π_{a2c}. All the four actions (i.e., on, off, speed up and slow down) of each machine will be determined by this pure model-free RL-based control scheme.

2) Extended A2C scheme

In this scenario, we use the extended A2C algorithm with an auxiliary model-based path for training to obtain a control policy π_{mve}. All the four actions (i.e., on, off, speed up and slow down) of each machine will be determined by this extended A2C algorithm.

In this case study, a static policy without any control action is used as a baseline. For the training process, we use the reward's improvement with respect to the baseline reward as the metric. In other words, we compare $(R_{base} - R_{base})$, $(R_{standard\ A2C} - R_{base})$, $(R_{extended\ A2C} - R_{base})$ and $(R_{hybrid} - R_{base})$. In order to ensure that the performance of control policies is independent of its initial states, eight sets of initial states are randomly generated to evaluate each policy. With each initial state, these three policies are tested with the three scenarios independently for 10,188 min.

7.2. Results and discussion

Training performance comparison: The training processes for all three algorithms are demonstrated in Figure 5. The x-axis shows the training iteration, and the y-axis represents the rewards as in Eq. (13). It is observed that the extended A2C algorithm and the proposed hybrid control scheme ultimately reach a much higher reward than that of the standard A2C algorithm. The final reward after training for the extended A2C and the hybrid control schemes are comparable to each other. This result confirms that the DRL with the auxiliary model-based path is more competitive in achieving a more accurate state value. Although the training reward of both extended A2C and the hybrid control schemes are similar, the hybrid algorithm achieves the steady reward value (considered as converges) much faster during the training process. This result indicates that the proposed hybrid control scheme can effectively shorten the training time through reducing the action space of the control problem.

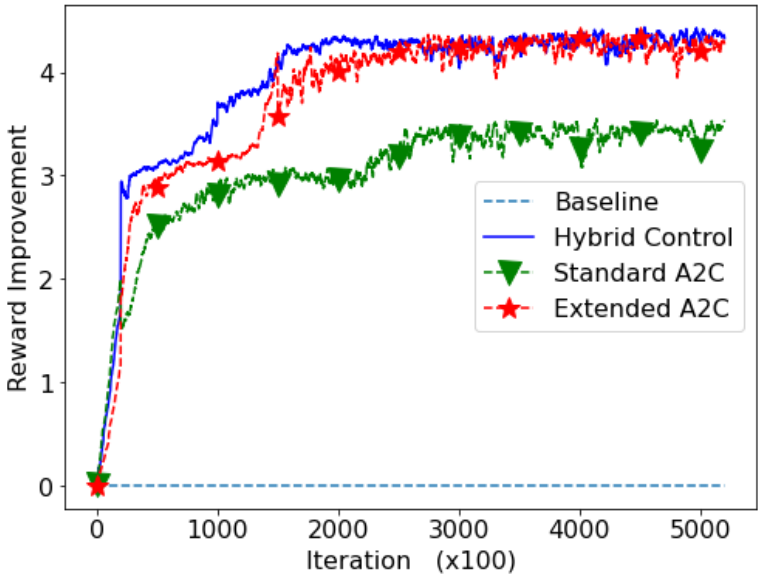

Figure 5. The learning performance comparison within the same training duration

Execution performance comparison: In Figure 6, the average costs and their 95% confidence intervals using the trained policies based on the three algorithms are compared. Eight randomly generated initial state sets have been used. It is evident that both extended A2C and hybrid control schemes outperform the standard A2C in reducing both profit loss and energy costs. In terms of the total cost, the hybrid control scheme performs 29.62% better than the standard A2C. Meanwhile, extended A2C algorithm improves the system performance by 23.4% compared with the standard A2C. This result further confirms the discussion in Section 6.2.5 that with the additional information gained from the model-based path, more accurate state value estimations can be achieved to ensure a higher reward policy. In addition, it is noted that although the hybrid control scheme has resulted the highest action costs, it leads to the best system performance in terms of lowest total cost. This result demonstrates that although the proposed hybrid control algorithm requires more actions during the operational period, the effectiveness of each action is ensured.

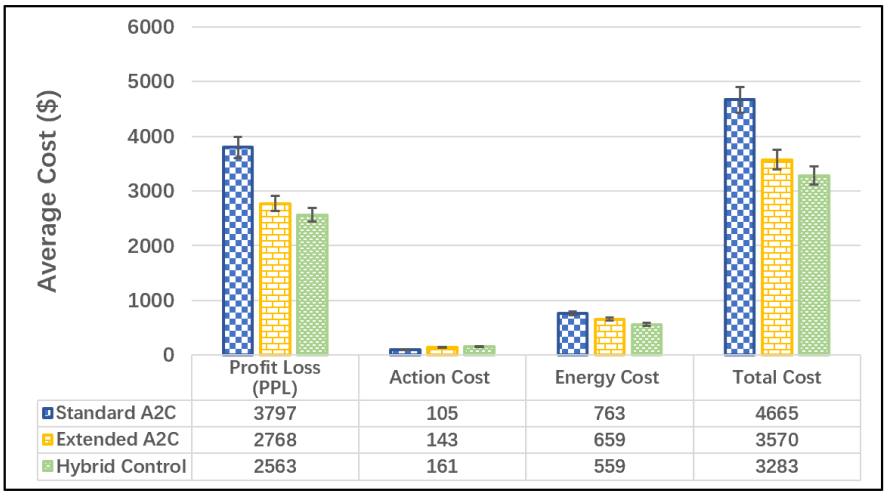

Figure 6. The average cost comparison between different policies

Impact of cycle time changing ratio on training and execution: Based on a realistic machine's setting, the cycle time of each machine can only be changed within a range. In other words, a machine's cycle time cannot exceed its designed upper and lower bound to ensure the machine normal operation. Within the boundary, when control action triggers, each machine's cycle time can be increased or decreased by a cycle time changing ratio φ, i.e., the cycle time of machine S_i can be changed to $T_i(t) = T_i(t-1) * (1 \pm \varphi)$. We perform additional experiments to study the impact of cycle time changing ratio on training and execution. We vary the cycle time changing ratio from 0.05 to 0.3 with an increment of 0.05 and return the training with the same parameters for each variation. As shown in Figure 7, the proposed hybrid control method outperforms the standard A2C method and extended A2C method in achieving a faster convergence and a higher reward value, as expected. It is noticed that for different cycle time changing ratio, there is no significant difference in convergence time for all three algorithms. Figure 8 further demonstrates the average total cost with 95% confidence intervals (CI) resulted from the trained policy based on the hybrid control scheme. It can be observed that there is no statistically significant difference in total cost for different cycle time changing ratio. These experiments suggest that the hybrid control scheme and the system performance are not sensitive to the cycle time changing ratio.

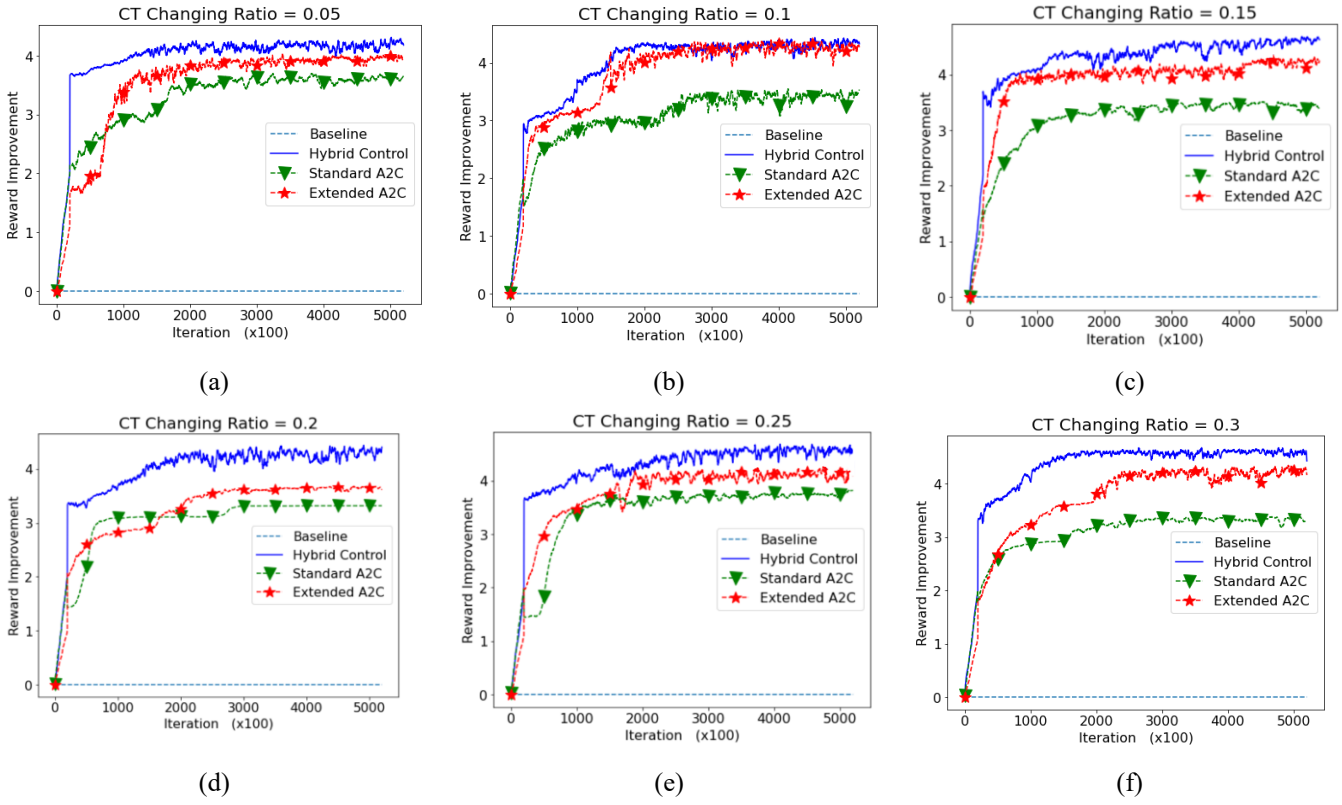

Figure 7. The learning performance comparison with different cycle time changing ratios

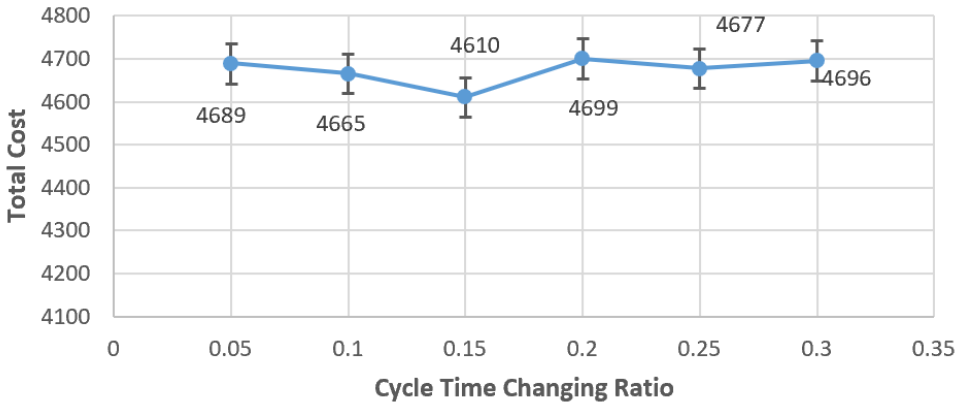

Figure 8. The average total cost and 95% confidence intervals comparison between different cycle time changing ratios

Impact of control interval on training and execution: The control interval is a critical factor that is difficult to analyze analytically in complex manufacturing systems. Therefore, additional experiments have been performed to study the impact of control interval on the system. We vary the control interval to be every one cycle, two cycles, three cycles, four cycles, five cycles, and six cycles of each machine. The training process and execution keep the same as previous experiments. Figure 9 illustrates the training process. It shows that for very short control interval such as one cycle, or relatively longer control interval such as six cycles, although all three algorithms converge with hybrid control scheme converging faster, the final rewards in training for all three schemes are noticeably lower compared to that for other control interval. The same phenomenon can be observed for execution performance. In Figure 10, the total costs with 95% CI resulting from the trained policies based on the proposed hybrid control scheme are compared for different control intervals. It is clear that very short or very long control intervals may lead to lower system performance in terms of higher total cost. It is noticeable that when the control interval is comparable to the general cycle time of the production line (i.e., within the range of two cycles - three cycles for each machine in this case study), the overall system performance will reach its best (meaning lowest total cost), and the advantage of the hybrid control method is most obvious compared with the other two pure DRL-based algorithms. This phenomenon is due to the fact that too short control intervals make the control actions chase the noise without stabilizing the effect of control. On the other hand, a much longer control interval means lesser chances for the agent to interact with the environment within the same training time. Thus, it requires more iterations to gather experience to achieve convergence, leading to poorer training and execution performance. Therefore, it is evident that the manufacturing system and control schemes are sensitive to the control interval, and there exists an optimal range of control interval that can lead to better system performance.

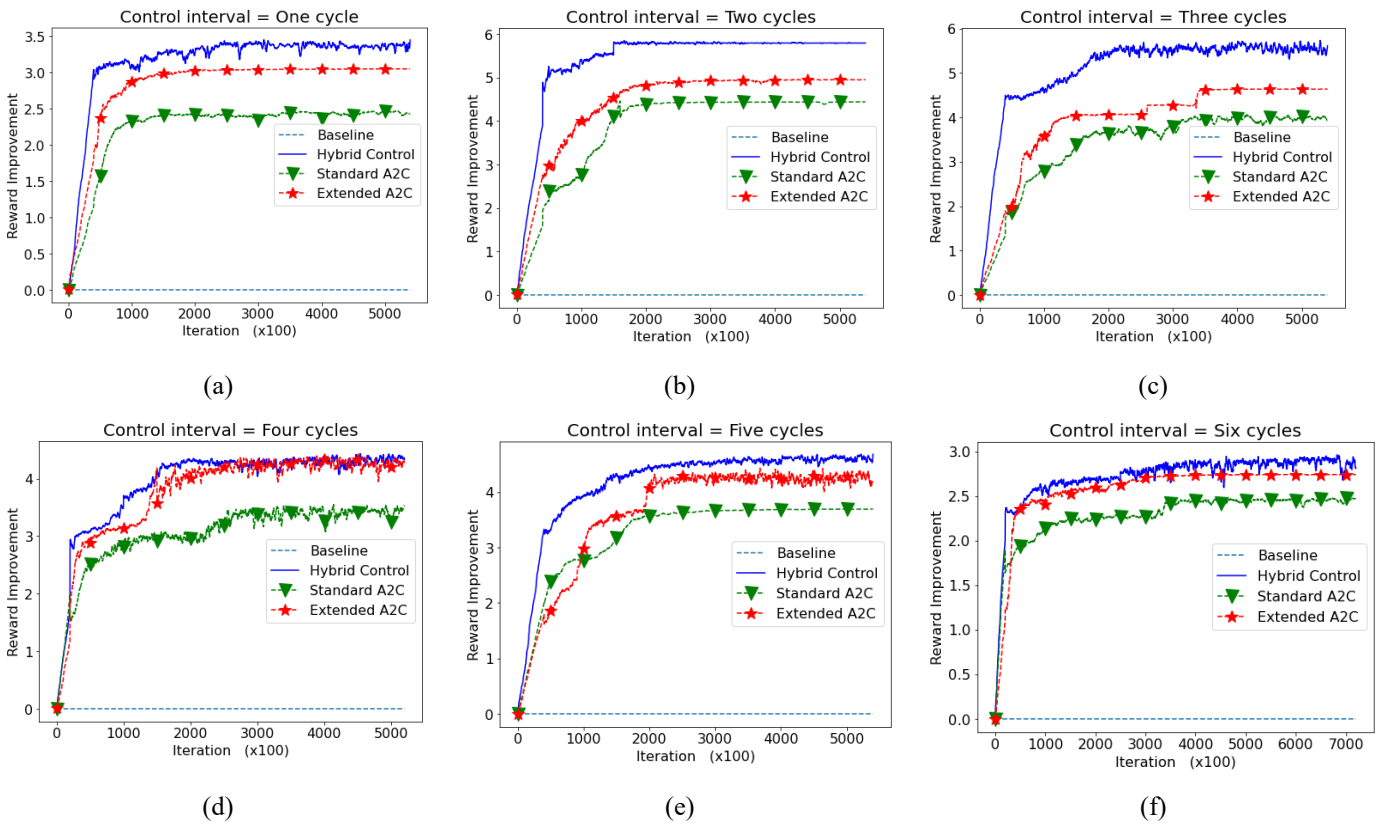

Figure 9. The learning performance comparison with different control intervals

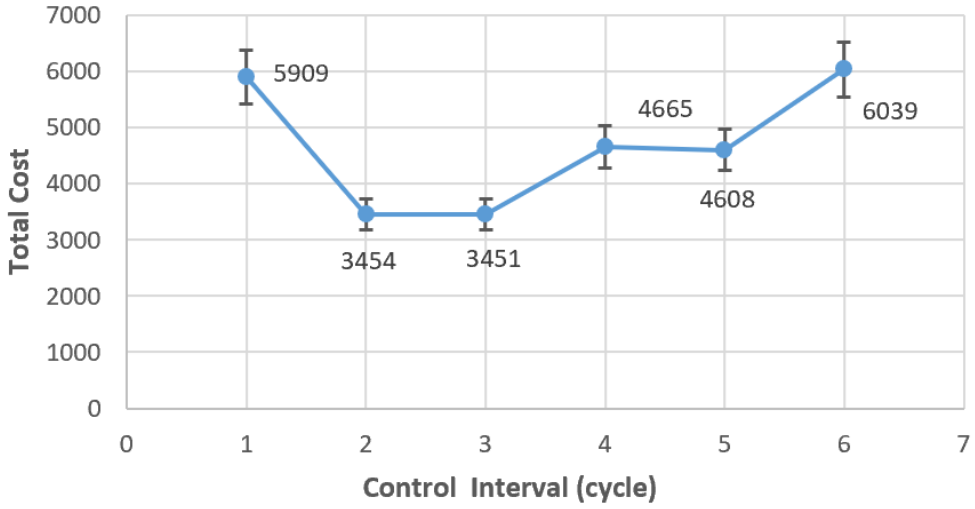

Figure 10. The average total cost and 95% confidence intervals comparison between different control intervals

8. Conclusion and Future Work

Machine cycle control is a complex problem due to its huge state space and complicated machine interactions.

A hybrid control strategy that combines distributed feedback control and extended DRL-based control is developed to solve this problem. The distributed feedback control can quickly determine each machine's on/off status based on the system property of OW, which can effectively reduce the action space for this control problem. The extended DRL algorithm is proposed to adjust each machine's cycle time smartly. In addition, we develop an MVE method to improve the state value estimation for the DRL algorithm by adding an auxiliary model-based path. The case study demonstrates that a good control policy can be obtained by using the proposed hybrid control scheme, which leads to a lower total cost than that obtained from pure learning-based control policy. In the future, we plan to investigate more about the real-world constrains of the production line and further refine our control methods. We will also consider embedding the process-level knowledge (e.g., quality issue) into our control framework.

Acknowledgments

This work is supported by National Science Foundation Grants 1853454.

Reference

[1] P. Prajakta and D. Prajakta., "Reduction in Cycle Time using Lean Manufacturing," Technique: A Typical Case Study 1 Patil Prajakta R and 2 Dr. Inamdar K.H, 2019.

[2] S. Huang, Y. Guo, D. Liu, S. Zha, and W. Fang. "A Two-Stage Transfer Learning-Based Deep Learning Approach for Production Progress Prediction in IoT-Enabled Manufacturing," IEEE Internet Things J 2019; 6:10627–38.

[3] J. Li, and S. M. Meerkov, "Production System Engineering." New York, NY, USA: Springer, 2009.

[4] E. A. Feinberg and A. Shwartz, "Handbook of Markov Decision Processes: Methods and Applications," Boston, MA, USA: Springer, 2002.

[5] X. Ou, Q. Chang, and N. Chakraborty, "A Method Integrating Q-Learning with Approximate Dynamic Programming for Gantry Work Cell Scheduling," IEEE Transactions on Automation Science and Engineering, vol. 18, no. 1, pp. 85-93, Jan. 2021, doi: 10.1109/TASE.2020.2984739.

[6] C. Li, J. Huang, and Q. Chang, "Data-Enabled Permanent Production Loss Analysis for Serial Production Systems with Variable Cycle Time Machines," IEEE Robotics and Automation Letters, vol. 6, no. 4, pp. 6418-6425, Oct. 2021, doi: 10.1109/LRA.2021.3093012.

[7] Z. Jia., L. Zhang, J. Arinez, and G. Xiao, "Performance Analysis for Serial Production Lines with Bernoulli Machines and Real-Time WIP-Based Machine Switch-On/Off Control," International Journal of Production Research, 54:21, 6285-6301, 2016, DOI: 10.1080/00207543.2016.1197438.

[8] P. Cui, J. Wang, Y. Li, and F. Yan, "Energy-Efficient Control in Serial Production Lines: Modeling, Analysis and Improvement," Journal of Manufacturing Systems, Volume 60, 2021, Pages 11-21, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2021.04.002.

[9] F. Tao, Q. Qi, A. Liu, and A. Kusiak, “Data-Driven Smart Manufacturing,” Journal of Manufacturing Systems, Volume 48, Part C, 2018, Pages 157-169, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2018.01.006.

[10] J. Liu, Q. Chang, G. Xiao, and S. Biller, “The Costs of Downtime Incidents in Serial Multi-Stage Manufacturing Systems,” Journal of Manufacturing Science & Engineering 2012; 134(2): 021016. http://dx.doi.org/10.1115/1.4005789.

[11] S. Paul, R. Sarker, and D.Essam, “Managing Disruption in an Imperfect Production Inventory System,” Computers & Industrial Engineering 2015; 84:101e12. http://dx.doi.org/10.1016/j.cie.2014.09.013.

[12] J. Zou, Q. Chang, J. Arinez, and G. Xiao. "Data-Driven Modeling and Real-Time Distributed Control for Energy Efficient Manufacturing Systems." Energy 127 (2017): 247-257.

[13] J. Zou, Q. Chang, X. Ou, J. Arinez, and G. Xiao, “Resilient adaptive control based on renewal particle swarm optimization to improve production system energy efficiency”, Journal of Manufacturing Systems, Volume 50, 2019, Pages 135-145, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2018.12.007.

[14] Y. Li, J. Wang, and Q. Chang. "Event-Based Production Control for Energy Efficiency Improvement in Sustainable Multistage Manufacturing Systems." ASME. Journal of Manufacturing Science & Engineering February 2019; 141(2): 021006. https://doi.org/10.1115/1.4041926.

[15] E. Frazzon, M. Kück, and M. Freitag, “Data-Driven Production Control for Complex and Dynamic Manufacturing Systems”, CIRP Annals, Volume 67, Issue 1, 2018, Pages 515-518, ISSN 0007-8506, https://doi.org/10.1016/j.cirp.2018.04.033.

[16] W. Bernd, R. André, B. Lenz, A. Thomas, B. Thomas, K. Alexander, and K. Andreas, “Optimization of Global Production Scheduling with Deep Reinforcement Learning,” Procedia CIRP, Volume 72, 2018, Pages 1264-1269, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2018.03.212.

[17] K. Kang and V. Subramaniam, “Integrated Control Policy of Production and Preventive Maintenance for a Deteriorating Manufacturing System,” Computers & Industrial Engineering, vol. 118, pp. 266–277, Apr. 2018.

[18] S. Racanière, T. Weber, David P. Reichert, L. Buesing, A. Guez, D. Rezende, A. Badia, O. Vinyals, N. Heess and Y. Li, “Imagination-Augmented Agents for Deep Reinforcement Learning”, NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, December 2017 Pages 5694–5705.

[19] T. Johannink et al., "Residual Reinforcement Learning for Robot Control," 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 6023-6029, doi: 10.1109/ICRA.2019.8794127.

[20] M. Neunert, A. Abdolmaleki, M. Wulfmeier, T. Lampe, T. Springenberg, R. Hafner, F. Romano, J. Buchli, N. Heess and M. Riedmiller, “Continuous-Discrete Reinforcement Learning for Hybrid Control in Robotics”, Proceedings of the Conference on Robot Learning, PMLR 100:735-751, 2020.

[21] Y.A. Paredes-Astudillo, JF. Jimenez, G. Zambrano-Rey, and D. Trentesaux, “Human-Machine Cooperation for the Distributed Control of a Hybrid Control Architecture”, Studies in Computational Intelligence, vol 853. Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-27477-18.

[22] S. Li, S. Bing, and S. Yang, “Distributional Advantage Actor-Critic,” arXiv preprint arXiv:1806.06914 (2018).