STRUCTURE-DRIVEN DEEP REINFORCEMENT LEARNING FOR WIRELESS NETWORKED CONTROL

By

Jiazheng Chen

SUBMITTED IN FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
CENTRE OF EXCELLENCE IN TELECOMMUNICATIONS
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
FACULTY OF ENGINEERING
THE UNIVERSITY OF SYDNEY

DECEMBER 2024

To my beloved parents Mingzhi Chen, Ruipin Huang,
and loving wife Binbing Yang.

Acknowledgements

My Ph.D experience has been a memorable and transformative period in my life. Here, I want to express my most profound appreciation to the people who have given me much support and guidance during this formative period.

First and foremost, I would like to express my sincere gratitude to my head supervisor, Prof. Yonghui Li, for his invaluable guidance, unwavering support, and patient instruction throughout my Ph.D period. He always gave me precious advice for the future direction. I would like to thank Prof. Branka Vucetic. Her passion and charm have encouraged me to conduct academic research.

I am deeply grateful to my auxiliary supervisor, Dr. Wanchun Liu. I could never complete the thesis without her support. Her profound knowledge, genuine concern, and insightful feedback for my academic have deepened my understanding of my research area and played an essential role in shaping this thesis.

I would also like to thank Prof. Liqun Fu from FUNLAB of Xiamen University. Her kindness and valuable support during the unprecedented challenges posed by COVID-19 enabled me to overcome the obstacles and progress toward my research.

My heartfelt appreciation goes to the members of the Telecommunications Lab at the University of Sydney for fostering a stimulating and collaborative research environment. They have enriched my Ph.D journey and left precious memories in the past years.

To my beloved wife, Binbing Yang, words cannot adequately express my profound gratitude for your unwavering support, patience, and understanding. Your love and encouragement have been the pillars that sustained me through the most challenging and sad times of this journey.

Finally but certainly not least, I am eternally indebted to my parents, Mingzhi Chen and Ruipin Huang, for their unconditional love, endless support, and unwavering

belief in my capabilities over the past twenty-seven years. Their sacrifices, encouragement, and life lessons have been the driving force behind my pursuit of knowledge and personal growth.

Jiazheng Chen

Sydney, NSW

December 2024

Statement of Originality

The work presented in this thesis is the result of original research carried out by myself, in collaboration with my supervisors, while enrolled in the School of Electrical and Computer Engineering at the University of Sydney as a Ph.D. candidate.

These studies were conducted under the supervision of Prof. Yonghui Li and Dr. Wanchun Liu. It has not been submitted for any other degree or award in any other university or educational institution.

Jiazheng Chen
School of Electrical and Computer Engineering
The University of Sydney
December 2024

Authorship Attribution Statement

The primary contributions of this thesis are presented in Chapters 2, 3, and 4. Each of these chapters corresponds to a respective paper [J1], [J2], and [J3]:

- Chapter 2 is based on [J1]. I designed the study, conducted the simulation, collected and analyzed data, and drafted and revised the manuscript. [C1] is a conference paper that is part of the work of [J1].
- Chapter 3 is based on [J2]. I designed the study, conducted the simulation, collected and analyzed data, and drafted and revised the manuscript.
- Chapter 4 is based on [J3] that is to be submitted. I designed the study, conducted the simulation, collected and analyzed data, and drafted and revised the manuscript.

In addition to the statements above, in cases where I am not the corresponding author of a published item, permission to include the published material has been granted by the corresponding author.

Jiazheng Chen,
December 2024

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Yonghui Li,
December 2024

Abstract

Wireless networked control systems (WNCSs) consisting of plants, distributed sensors, remote estimator actuators, and controllers are a key component for Industry 4.0 and have been widely applied in many areas such as industrial automation, vehicle monitoring systems, building automation, and smart grids. In particular, providing a high quality real-time remote estimation of dynamic system plant states is important in ensuring control performance and stability of WNCSs. For large-scale WNCSs, the limited communication resources and unreliable transmission in wireless communications have a significant effect on the remote estimation quality. Therefore, it is challenging to design a transmission scheduling of wireless sensors to guarantee the remote estimation performance. Most existing works on transmission scheduling in wireless communication systems merely use general methods such as value iteration and deep reinforcement learning (DRL) without considering the inherent features of the sensor scheduling problems that are different from other problems. This thesis derives several structural properties of the sensor scheduling problem and then develops new DRL algorithms to minimize the remote estimation mean square error (MSE) by using these properties. First, we derive the threshold structure of the optimal scheduling policy and then propose a structure-driven deep Q-network (DQN) and structure-driven deep deterministic policy gradient (DDPG). Second, we prove the monotonicity of the Q function and use it to develop different types of monotonic critic neural networks (NNs) that can be applied to the baseline DDPG and its variants. Finally, we obtain the convexity of the optimal value function and the greedy structure of the optimal scheduling policy. To find the optimal scheduling policy in large-scale systems, a structure-guided unified dual on-off policy (SUDO) DRL framework is proposed based on the derived structural properties. Numerical results illustrate that our proposed DRL algorithms can significantly improve the convergence speed and estimation performance when compared to general DRL algorithms.

Contents

List of Figures

List of Acronyms

AD	action-difference
AM	action monotonicity
AoI	age of information
AR	augmented reality
DDPG	deep deterministic policy gradient
DQN	deep Q-network
DRL	deep reinforcement learning
FCNN	fully connected neural network
GAE	generalized advantage estimation
i.i.d.	independent and identically distributed
IIoT	Industrial Internet of Things
IoT	Internet of Things
MA-DDPG	monotonic architecture-based DDPG
MDP	Markov decision process
MRI-DDPG	type I monotonicity-regularized DDPG
MRII-DDPG	type II monotonicity-regularized DDPG
MSE	mean-square error
NEC	number of episodes for convergence
NN	neural network
NOMA	non-orthogonal multiple-access
OMA	orthogonal multiple-access

PPO	proximal policy optimization
QoS	quality of service
SAC	soft actor-critic
SE	structure-enhanced
SG	structure-guided
SUDO	Structure-guided Unified (Dual) On-off policy
TD	temporal difference
TD3	Twin Delayed DDPG
TRPO	trust region policy optimization
VM	value-monotonicity
XR	extended reality

List of Symbols and Notations

$\mathbb{N}$	the set of positive integers
$\mathbb{R}$	the set of real numbers
$\mathbb{E}[\cdot]$	the expectation operator
$\mathcal{N}(\cdot, \cdot)$	the Gaussian distribution
$(\cdot)^\top$	the vector or matrix transpose operator
$(\cdot)^{-1}$	the matrix inverse operator
$\mathrm{Tr}(\cdot)$	the matrix trace operator
$\mathbb{1}(\cdot)$	the indicator function
$\rho(\mathbf{A})$	the spectral radius of the square matrix $\mathbf{A}$
$\Pr(A)$	the probability of the event A
$\Pr(A\|B)$	the conditional probability of the event A given the event B

List of Publications

The following is a list of publications in refereed journals and conference proceedings produced during my Ph.D. candidature. In some cases, the journal papers contain materials overlapping with the conference publications.

Journal Papers

[J1] J. Chen, W. Liu, D. E. Quevedo, S. R. Khosravirad, Y. Li and B. Vucetic, "Structure-Enhanced DRL for Optimal Transmission Scheduling," *IEEE Transactions on Wireless Communications*, vol. 23, no. 1, pp. 379-393, May. 2023.

[J2] J. Chen, W. Liu, D. E. Quevedo, Y. Li and B. Vucetic, "Semantic-Aware Transmission Scheduling: A Monotonicity-Driven Deep Reinforcement Learning Approach," *IEEE Wireless Communications Letters*, vol. 27, no. 12, pp. 3260-3264, Oct. 2023.

[J3] J. Chen, W. Liu, D. E. Quevedo, "Structure-Guided DRL Combining Off-policy and On-policy Framework for Goal-oriented Transmission Scheduling," to be submitted to *IEEE Transactions on Wireless Communications*.

Conference Papers

[C1] J. Chen, W. Liu, D. E. Quevedo, Y. Li and B. Vucetic, "Structure-Enhanced Deep Reinforcement Learning for Optimal Transmission Scheduling," in *Proc. IEEE International Conference on Communications (ICC)*, Rome, Italy, 2023, pp. 1-7.

Chapter 1

Introduction

This chapter first introduces the background of the research and describes the foundations of remote state estimation systems. Then, it reviews related works in the existing literature. Finally, it summarizes the motivation and highlights the main contributions of this thesis.

1.1 Background

Wireless networked control systems (WNCSs) consisting of plants, distributed sensors, remote estimator actuators, and controllers are a key component for Industry 4.0 [1, 2] and have been widely applied in many areas such as industrial automation [3–6], vehicle monitoring systems [7], building automation [8], and smart grids [9]. Each plant is measured by a sensor, which pre-processes the collected measurement and transmits the local estimation to a remote estimator. The remote estimator estimates the plants' states based on the collected local estimations. The remote controller generates control signals, and the wireless actuators receive the control signals and control the processes.

Traditional control systems and WNCSs differ significantly in their design, implementation, and challenges. Traditional control systems typically rely on direct, wired connections between system components, ensuring low-latency and highly reliable communication. This wired infrastructure minimizes the risk of data loss and disturbances, making such systems more predictable and stable in performance. However, their reliance on physical wiring limits flexibility and scalability, particularly in large-scale or dynamic environments. In contrast, WNCSs utilize wireless communication networks to interconnect sensors, controllers, and actuators. This networked approach enables greater flexibility, scalability, and ease of deployment, especially in applications where wiring is impractical or cost-prohibitive. Nonetheless, it introduces new challenges, primarily due to the constraints of the underlying communication networks. Wireless networks are inherently less reliable and more prone to interference, latency, and packet loss compared to their wired counterparts. Limited bandwidth in wireless communication further exacerbates the difficulty of ensuring timely and accurate data transmission. These factors pose significant challenges to the design and implementation of WCNCs, particularly in guaranteeing real-time remote estimation and control.

To ensure the performance and stability of the WNCSs, it is essential to provide high quality real-time remote estimation of dynamic system plant states [10, 11]. Unlike conventional control systems, which benefit from extremely low-latency and highly reliable data transmission of wired networks, WNCSs face unique challenges. Due to the limited bandwidth and unreliable nature of wireless networks, it is challenging to design a transmission scheduling policy for wireless sensors to guarantee remote estimation performance. Inaccurate remote estimation can adversely affect

the computation of control signals, potentially destabilizing the controlled plants and significantly increasing the risk of catastrophe. In the open literature, many heuristic policies such as round-robin scheduling [12], quality of service (QoS) based scheduling [13], and event-triggered scheduling [14] have been proposed to schedule the transmission without considering the meaning and importance of packets from different sensors. Clearly, in wireless networked control applications, packets carrying different sensing information have different importance in the overall performance of WNCSs. Many studies have also investigated the optimal scheduling problems by formulating the problems as a Markov decision process (MDP) problem and solving them with conventional model-based solutions such as the policy-iteration algorithm and value-iteration algorithm [15, 16]. However, these methods are not feasible in large-scale scheduling problems because of the curse of dimensionality caused by high-dimensional state and action spaces. Therefore, deep reinforcement learning (DRL) has been developed to deal with large MDPs in large-scale systems by using deep neural networks (NNs) as function approximators [17–19]. Furthermore, to improve the performance of the scheduling policy, the properties of the optimal policy should be used in policy design rather than using conventional general methods.

1.2 Foundation of WNCSs

This section introduces some basic elements and concepts of WNCSs in detail.

1.2.1 Kalman Filter

Each sensor in the WNCS employs a classic Kalman filter, which is one of the most popular methods to generate the state estimation in the presence of noisy measurements and uncertain system dynamics [20, 21]. The Kalman filter generates state estimates through a two-step process: prediction and update. In the prediction step, the system model is used to project the current state estimate forward in time, creating an a priori estimate. In the update step, it incorporates new measurement data to refine this prediction, producing an a posteriori estimate. This recursive process allows the filter to continuously refine its state estimate, adapting to changing system dynamics and measurement quality in real time. The Kalman filter has been proven to be the optimal linear estimator in terms of minimum mean square error (MMSE), provided that the covariance matrices of both the process and measurement noise are accurately known [22]. The Kalman filter's optimality relies on the assumptions of linear system dynamics and Gaussian noise distributions. To address diverse processes and network models, various modified Kalman filters have been developed. [23, 24].

1.2.2 Wireless Channel

In the wireless network, the data packet from each plant is transmitted through several frequency channels. In contrast to wired connections, wireless channels are inherently stochastic, subject to time-varying fading, multipath propagation, and interference. This results in packet losses, delays, and errors in data transmission between sensors and the remote estimator, which presents a significant impact on system performance and stability. The channel's behavior is often modeled using statistical approaches, such as Rayleigh or Rician fading models, to capture its dynamic

nature [25, 26].

1.2.3 Remote Estimator

Due to packet losses, transmission delays, and limited bandwidth, not all local state estimations can be successfully received by the remote estimator. Thus, the remote estimator considers the state estimation of each process for computing control signals based on the current and historical received data. In WNCSs, the minimum mean square error estimator is normally used, which requires knowledge of dynamic processes [11, 27].

1.2.4 Scheduler

For generality, the wireless communication resources are limited, and the number of sensors is assumed to be less than the number of sensors. Thus, the scheduler is a key component in networked control systems that manage access to shared wireless communication resources. Its primary role is to determine which sensors can transmit their measurements at each time instant, effectively coordinating multiple data streams over limited network channels. The scheduler receives commands from the controller to make informed decisions about transmission scheduling, ensuring that critical sensor measurements get priority when needed. Without proper scheduling, network congestion could lead to packet collisions, data loss, and degraded control performance. The scheduler thus acts as a traffic manager, optimizing network utilization while maintaining the quality of control for all processes.

1.2.5 Remote Controller and Stability

The remote controller is developed to stabilize the overall system to have a desirable steady state. There are many different types of controllers, including Proportional-Integral-Derivative (PID) controllers, Linear Quadratic Regulator (LQR) controllers, and Model Predictive Control (MPC). Stability is a fundamental requirement in remote state estimation systems, particularly in the context of wireless networked control [28]. In this thesis, the stability of the remote state estimation system encompasses both the stability of the plant being monitored and the stability of the remote estimator. The stability of the plant is measured by the magnitude of its steady state, which is bounded in the stable systems [29]. The remote estimator is said to be stable if the mean square error (MSE) of estimation is bounded and vice versa [30–32].

1.2.6 Performance Evaluation

In contrast to the conventional data-oriented performance evaluation, such as reliability, data rate, delay, and latency, the performance in remote estimation is usually measured by the MSE or estimation error covariance as it aims to provide an accurate estimation [33–38]. In remote estimation, plants with diverse characteristics have different effects on the system performance, which implies that the information from sensors has different usefulness for the goal of data exchange. Therefore, various goal-oriented metrics are also proposed for the different goals in the remote state estimation system [39, 40].

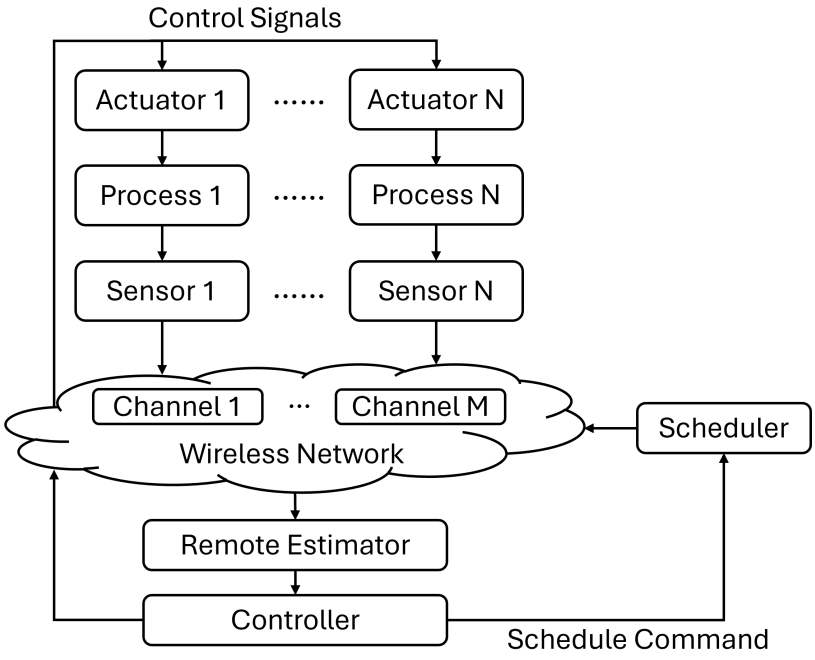


Figure 1.1: General system diagram of WNCS, the number of channels is assumed to be less than the number of sensors, i.e., $M < N$.

1.2.7 General System Model

The general system model of the WNCS in this thesis is illustrated in Fig 1.1, where the data from sensors are transmitted to the controller through a wireless network, and the transmission is regulated by a scheduler. Our goal is to optimize the remote estimation performance by designing the transmission scheduler effectively based on the structural properties of the optimal scheduling policy. We clarify these properties into three aspects: the threshold structure of the optimal policy, the monotonicity of the Q value function, and the convexity of the optimal value function. Comprehensive studies on these three topics are discussed in Chapters 2–4, respectively.

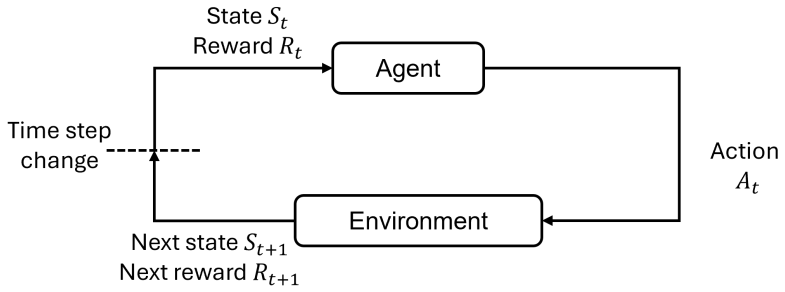


Figure 1.2: The reinforcement learning control loop.

1.3 Foundation of DRL

This section introduces some basic elements and concepts of DRL in detail.

1.3.1 Agent, Environment, and Interaction

The key components of a reinforcement learning algorithm are the agent, environment, and the interaction between them, as shown in Fig 1.2 [41]. During each time step of the interaction, the agent observes the current state of the environment and then takes action by using a policy, which is a state–action mapping function. After that, the environment transits to the next state based on the received action and then returns this next state and the reward to the agent. This loop repeats until the environment terminates, such as when the time step or the state exceeds the given constraint. The goal of the agent in a reinforcement learning algorithm is to maximize the cumulative sum of the received reward signal [42].

1.3.2 Markov Decision Processes

Reinforcement learning problems are usually formulated as Markov decision processes (MDPs), which are required to be a sequential decision-making problem that

satisfies the Markovian properties [43]. The Markovian properties mean that the probability of the environment transitioning to the next state only depends on the current state and action rather than the history of previous states and actions. Therefore, to formulate an MDP, we need to define four-tuple: states, actions, transition probabilities, and rewards.

1.3.3 Primary Functions in Reinforcement Learning

The first primary function is the policy of the agent, which is a function that maps the state to the action. The objective of the agent in an MDP is to find the optimal policy that maximizes the expected sum of the long-term reward. Next, for each policy, there are two functions evaluating how well the policy achieves this objective, as given below [44, 45].

(1) Value function: It evaluates the quality of a given state estimation by estimating the expected sum of the future reward starting from this state and then following the current policy. By using the optimal policy, we derive the optimal value function as defined follows:

$$V^*(\mathbf{s}_t) = \max_{\pi} \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r\left(\mathbf{s}_{t'}\right)\right], \tag{1.3.1}$$

where $\mathbf{s}_t$ is the state at time step t, π is the policy, γ is the discount factor. and $r(\cdot)$ is the reward function.

(2) Q function: It represents the expected sum of the future reward achieved by the current policy after executing a given action with the given state. Thus, the Q function is used to compare the value of different state–action pairs. The

Q function is defined as:

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(\mathbf{s}_{t'}) \Big| \mathbf{a}_t, \mathbf{a}_{t'} = \pi^*(\mathbf{s}_{t'}), \forall t' > t\right], \tag{1.3.2}$$

where $\mathbf{a}_t$ is the executed action at time step t and π^* is the optimal policy.

1.3.4 Exploration and Exploitation

A critical challenge in reinforcement learning is to balance the trade-off between exploration and exploitation [46]. To achieve higher rewards, the agent exploits the acquired knowledge to select the action that is more effective in generating the reward based on past experience. However, to discover more effective actions that have not been executed, the agent also needs to explore the state and action space. Thus, the exploration consumes more time during the training but is important for better action selections in the future. The exploration–exploitation dilemma is that focusing solely on either exploration or exploitation will lead to failure in the task [41]. Therefore, this trade-off is essential for the agent to learn an optimal policy efficiently.

1.3.5 Transitioning to Deep Reinforcement Learning

The conventional reinforcement learning methods are not feasible in MDPs with high-dimensional or continuous state space and action space because of the curse of dimensionality [47]. For example, the reinforcement learning algorithm cannot solve the power allocation problem or the sensor scheduling problem with a large number of sensors and channels [48, 49]. This limitation prompted the integration of deep learning with reinforcement learning, resulting in the emergence of deep reinforcement

learning (DRL). In DRL, deep neural networks (NNs) are used as function approximators to represent value functions, Q functions, or policies, allowing DRL agents to deal with complex and high-dimensional MDPs [50].

1.3.6 Value-based and Policy-based DRL

DRL algorithms can be categorized into two approaches: value-based and policy-based methods [51]. Value-based methods focus on approximating the optimal value function or Q function. The policy is derived implicitly by selecting actions that maximize these learned values [52]. For example, the deep Q network (DQN) algorithm uses an NN to estimate the Q function of the given state–action pair and then select the action with the highest Q value [53]. This method is sample efficient but can only handle the problem with discrete and small action spaces. Policy-based methods, such as soft actor–critic (SAC) [54] and trust region policy optimization (TRPO) [55], learn the policy function directly without computing the value function. This method parameterizes the policy with the NN and updates it to maximize the expected return. Without estimating the value function, it is often more stable during training and can deal with large and continuous action spaces.

1.3.7 Off-policy and On-policy DRL

Two distinct approaches to training a DRL agent are off-policy methods, where the agent learns from past experiences, and on-policy methods, where the agent learns from current experiences. The off-policy method (e.g., DQN and deep deterministic policy gradient (DDPG) [56]) updates the current policy based on data generated from the past policy. By reusing the past data, the off-policy DRL has a higher sample

efficiency and exploration effectiveness, but it also results in bias and instability during the training as the behavior of current policy and past policies can be very different. In contrast to the off-policy DRL, the on-policy DRL (e.g., TRPO and proximal policy optimization (PPO)), which is more stable in the stochastic environment because the agent learns from current experiences, has been used to solve more complex and larger MDPs than the off-policy method. The disadvantage of the on-policy method is that although the current generated data are discarded after the policy update to reduce training bias and instability, it also leads to poor data efficiency as well as insufficient exploration, and unbiased sampling may even be harmful to the performance [57].

1.4 Literature Review

This section presents a literature review of the existing works on transmission scheduling. This topic has attracted significant research interest in both academic and industrial fields, leading to numerous studies focused on solving the sensor scheduling problem. To summarize them clearly, we classify these works into three categories: communication transmission paradigm, derivation method for optimal scheduling policy, and structural features in the transmission scheduling problem.

1.4.1 Communication Transmission Paradigm

There are two different levels in communication transmission: technical (or data-oriented communications); and semantics (or goal-oriented communications) [58, 59]. The goal of the technical level is to accurately transmit the information bit-by-bit, while the semantic level aims to minimize the distortion and timing mismatch of the

data between plants and the remote estimator [60].

Data-oriented Communications

We first focus on the works related to the transmission scheduling in wireless communication systems that optimize the quality of service (QoS) performance, such as throughput [61], transmission delay [62], latency [63], and reliability [64]. In [61], a low-complexity and throughout-optimal scheduling algorithm was proposed for the downlink multi-channel wireless networks. [62] used a dynamic programming approach to find the delay-optimal scheduling policy for a multi-user-multi-channel system. [64] discussed cross-layer optimization methods for delay-aware and reliability-aware scheduler design. However, these conventional transmission scheduling methods are communications-centric and are agnostic to upper-layer applications, i.e., a scheduling policy does not take into account the meaning and importance of each packet. Since packets contain different sensing information, different generated packets are of different importance in the overall WNCS performance. Also, the transmission scheduling objective should be the WNCS performance rather than communications QoS.

Goal-oriented Communications

In WNCSs, which are also a kind of cyber-physical system, goal-oriented transmission aims to minimize the distortion and timing mismatch of the data between the edge devices and the remote destination [60]. Thus, the freshness of messages is critical, and monotonic functions of the age of information (AoI) are often introduced as cyber-physical system performance metrics during dynamic scheduling [65, 66] and

have attracted much attention [67]. The instantaneous remote estimation quality and control performance of the cyber-physical systems in [10, 65, 68] were derived as functions of individual devices' AoI states.

In [69], the optimal scheduling problem of a multi-loop WNCS with limited communication resources was investigated to minimize transmission power consumption under a WNCS stability constraint. In [65, 68], optimal sensor scheduling problems of remote estimation systems were investigated to achieve the best overall estimation MSE. The recent work [70] reduces the computation complexity for solving optimal scheduling problems through approximate dynamic programming. In [10], a joint sensor–controller (uplink) and controller–actuator (downlink) transmission scheduling problem was established and solved by the classic MDP framework. AoI-aware dynamic scheduling of cyber-physical systems over limited communication resources has attracted much attention in communications research [71]. In [65], an optimal scheduling problem of a multi-sensor remote estimation system was considered to minimize the overall long-term estimation mean square error. The instantaneous mean square error of each sensor is a function of the corresponding AoI. The scheduling problem was formulated as a Markov decision process that can be solved by conventional value and policy iteration methods. Furthermore, an optimal AoI minimization scheduling problem was considered under an average transmission power constraint in [16].

1.4.2 Derivation Method of Optimal Scheduling Policy

There are some heuristic scheduling policies, such as round-robin, greedy, and event-triggered policies, which can be used in different kinds of systems. However,

these methods have not considered the differences between transmitted data and wireless channel states and have not used the Markovian property of the scheduling problem to obtain the optimal scheduling policy. Thus, to solve the scheduling problem that can be formulated as a Markov decision problem, some conventional model-based and artificial intelligence methods have been proposed to derive the optimal policy.

Conventional Model-Based Algorithm

Many conventional algorithms, including dynamic programming, policy iteration, and value iteration, have been used to solve the scheduling problem in WNCSs. To solve the power scheduling problem in a simple single-loop networked control system, dynamic programming was used to derive the optimal scheduling policy in [72]. Sometimes, the remote estimation systems are attacked by an eavesdropper, so a dynamic programming algorithm was used to solve the transmission scheduling problem in a single-sensor single-channel system under this scenario [73]. For more complex remote state estimation systems with multiple sensors and a temporally correlated channel, an optimal scheduling policy over the finite time horizon was developed through policy iteration in [74]. In [75], the authors allowed multiple sensors to transmit over the same channel simultaneously and used an enhanced-exploration greedy least squares temporal difference algorithm, which is a kind of value iteration to approximate the optimal value function of the optimal scheduling policy. However, the conventional model-based solutions used in these works are not feasible in large-scale scheduling problems because of the curse of dimensionality caused by high-dimensional state and action spaces.

Artificial Intelligence Algorithm

In recent years, deep learning based artificial intelligence technology has been developed to solve complex problems that cannot be solved by conventional methods. Deep reinforcement learning (DRL) has been introduced to deal with large MDPs by using deep NNs as function approximators [76, 77]. Some works [78–81] have used the deep Q-network (DQN), a simple DRL method, to solve multi-sensor-multi-channel scheduling problems in different remote estimation scenarios. In particular, sensor scheduling problems for systems with 6 sensors have been solved effectively, providing significant performance gains over heuristic methods in terms of estimation quality. Advanced DRL algorithms with an actor–critic framework (consisting of an actor NN and a critic NN), such as soft actor–critic (SAC) and deep deterministic policy gradient (DDPG), were introduced to deal with large-scale semantic-aware scheduling problems that cannot be handled by DQN [54, 56]. The more recent work [82] used the proximal policy optimization (PPO) algorithm and novel action mapping scheme to obtain optimal scheduling policy at a much larger scale compared with other works.[1]

[1]The works [78–81] adopted the orthogonal multiple-access (OMA) scheme in the remote estimation system, while [82] considered non-orthogonal multiple-access (NOMA) schemes. For OMA, each channel can only be allocated to a single sensor; for NOMA, each frequency channel can be assigned to multiple sensors simultaneously, and the packets from different sensors are detected by processing the received superimposed signals [83].

1.4.3 Structural Properties in Transmission Scheduling Problem

Many works have investigated the properties of the transmission scheduling policy in the WNCSs or remote estimation systems (e.g., threshold structure, stability condition) and then utilized these derived properties to develop novel algorithms [84, 85].

The key features this thesis exploits include the threshold structure of the optimal scheduling policies, the monotonicity of the Q value function, and the convexity of the optimal V value function.

The threshold structure means that there are switching boundaries dividing the state space into multiple regions for different scheduling actions. In other words, an optimal policy has a structure where the action only changes at the switching boundaries of the state space. In particular, [86] focuses on an energy-constrained single-sensor-single-channel system and proves that the optimal policy has a threshold in terms of the sensor's AoI, determining whether the sensor will be scheduled or not. In [87], the authors considered a multi-sensor-multi-channel system, where all channels are static, and each sensor has a constant packet-drop probability at all frequency channels. This work also showed that the optimal scheduling policy has a multi-dimensional threshold structure in terms of all sensor AoI. As an extension, the work in [65] (a two-sensor system), assumed that different sensors could have different numbers of packets for carrying each measurement. The demonstrated threshold property of the optimal policy is related to the sensor AoI and the remaining packet numbers of each sensor. However, to satisfy the prerequisite of proving these threshold structures, there are two limitations of the channel models adopted in the above studies: 1) fading channel models are commonly adopted in practice, where channel

states are time-varying, and 2) wireless propagation via different frequency bandwidths has different properties, leading to different channel qualities. Beyond remote estimation systems, in [88], the threshold structure of an optimal sensor scheduling policy has also been identified for minimizing the average sum AoI. However, this work only considered a single-channel system, and the transmission success or failure was determined before a scheduling action.

1.5 Research Problems and Contributions

Although many works on remote state estimation have been reported in the literature to derive the optimal scheduling policy to maximize the overall estimation performance with different methods, these works merely use the general frameworks to solve specific scheduling problems without considering unique features that distinguish sensor scheduling problems from other MDPs. We also note that the drawback of general DRL is that it often cannot perform policy exploration effectively for specific tasks [89], which can result in convergence to local minima or even complete failure. This thesis aims to use the structural properties to guide DRL algorithms to effectively solve optimal scheduling problems. We introduce our research problems and contributions based on the subsequent chapters.

Our first research problem, presented in Chapter 2, is how to prove the threshold structure of the optimal scheduling policy and then tailor the action selection method during the DRL algorithm design to find the optimal policy effectively based on the proved optimal policy structure. The optimal scheduling policy has been proven to have many structural properties [65, 87, 90–93]. In addition, using the threshold structure of the optimal policy during the training of algorithms has been proven

to be an effective method for improving the performance in many different kinds of systems, see for instance [86, 88, 94, 95]. However, these existing works consider the threshold structure under a simple system or impractical assumptions and have not solved scheduling problems in large-scale systems. The DRL has shown its potential to solve MDPs with large state and action space and can also effectively improve the performance of remote estimation when compared with the heuristic policy and conventional model-based algorithm [17, 96, 97]. Thus, we naturally ask: is it possible to investigate the structure of the optimal policy of a specific scheduling problem first and then use the policy structure to tailor a DRL algorithm design to find the optimal policy effectively? The main contributions of Chapter 2 are summarized as follows.

- We prove that the optimal sensor scheduling policy has a threshold structure in terms of both the AoI states of all sensors and the corresponding channel states, where the channel states of different sensors at different frequencies are different. To the best of our knowledge, this is the first structural result of optimal scheduling policies over fading channels in the literature. In addition, we show that such a structural property exists in a wide range of dynamic scheduling and resource allocation problems, not limited to remote state estimations. Interestingly, we also give a counterexample to show when such a property does not exist.

- We first formulate the sensor scheduling problem as an MDP, and then develop novel structure-enhanced DRL algorithms for solving the problem, building on the derived threshold properties of the optimal policy. In particular, we design a structure-enhanced action selection method, which tends to select actions that obey the threshold structure. Such an action selection method can explore the

action space more effectively and enhance the learning efficiency of DRL agents. Furthermore, we introduce a structure-enhanced loss function to add penalties to actions that do not follow the threshold structure. The new loss function guides the DRL to converge to the optimal policy structure quickly. We apply the proposed action selection method and the novel loss function to redesign the most commonly adopted DRL frameworks for scheduling, i.e., DQN and deep deterministic policy gradient (DDPG), referred to as the structure-enhanced DQN and DDPG algorithms.

- Our extensive numerical results illustrate that the proposed structure-enhanced DRL algorithms can reduce training time by 50% while reducing the remote estimation mean square error by 10% to 25% compared to benchmark DRL algorithms. Importantly, the structure-enhanced DRL algorithms can converge and perform well under some system settings that cannot be solved by any of the benchmark DRL algorithms.

My second research problem, discussed in Chapter 3, focuses on developing a deeper understanding of the structural properties of optimal scheduling policies (e.g., the monotonicity of the critic neural network). These properties can be utilized to enhance the approximation of the critic network to the Q-value function, thereby enabling faster and more effective convergence during DRL algorithm training. In the first research problem, we only investigate the properties of the optimal policy itself rather than some functions related to the optimal policy in this kind of MDP. There are two basic value functions of the optimal policy: the optimal value function and the state action value function, measuring the value of the current state and state–action pair, respectively. In different kinds of MDPs, these two value functions

have been proven to have some structural features, such as monotonicity [98–100] and convexity [101, 102]. In DRL algorithms with actor–critic frameworks, the critic NN is used to estimate the optimal value or state action value. Therefore, we aim to find some characteristics of the value function of the optimal sensor scheduling policy and then use them to develop a novel actor–critic DRL algorithm to improve the performance of the derived scheduling policy by regularizing the critic NN during the training. The main contributions of Chapter 3 are summarized as follows.

- First, we mathematically prove that the state action value function of the optimal scheduling policy is a monotonic function in terms of the AoI states and channel states of the system. Such a property has never been established in the open literature. These results illustrate that a well-trained critic NN of the scheduling policy should hold the same monotonicity.

- Second, by leveraging this theoretical property, we propose two monotonic critic NN training methods: network architecture-enabled monotonicity and regularization-based monotonicity. Specifically, the network architecture-enabled method achieves the monotonic NN by limiting the weights and the architecture of the NN and the regularization-based method obtains the monotonicity by introducing the monotonic penalty term in the loss function. The proposed training methods can be directly applied to the baseline DDPG and its variants.

- Last, our numerical results show that the proposed monotonicity-enforced DDPG algorithms can reduce training time and improve training performance significantly compared to the baseline DDPG.

Our third research problem, presented in Chapter 4, is how to prove that the

optimal value function of the optimal scheduling policy is convex and then use all the derived structural features of the optimal scheduling policy to develop a structure-guided unified on-off policy DRL, which can solve large MDPs as on-policy DRL while achieving similar performance as off-policy DRL. The main contributions of this chapter are summarized as follows.

- We formulate the goal-oriented scheduling problem as an MDP and then derive some structural properties of the formulated MDP. Firstly, we prove the monotonicity of the optimal V function w.r.t channel states when the cost function is monotonic. In addition, the optimal V function holds the asymptotic convexity w.r.t AoI states if the cost function of the problem is asymptotic convex, which is proven to exist in some goal-oriented communication systems in this paper. Moreover, we also prove the greedy structure of the optimal policy under the gateway conditions, indicating some actions cannot be optimal when given a state, which is used to regularize the action selection of the scheduling policy. To the best of our knowledge, these derived properties have not been investigated in the existing literature.

- We propose the structure-guided unified on-off policy DRL based on the derived properties and some structural features from the existing works. To improve the exploitation performance, we introduce NNs' structural evaluation terms, structure score schemes, and prioritized replay buffer. The structural evaluation terms are used to evaluate whether the NNs follow the derived structural properties at the given state. Based on these evaluation terms, we further introduce a structure score scheme that assigns scores for different generated trajectories to determine their values. Then, building on the derived score, we develop a

prioritized replay buffer to store the past experience with different priorities. This scheme and the new replay buffer can improve the sample efficiency and training stability by reusing high-quality past data, which is more likely to be the optimal policy.

- In addition to the exploitation, to improve the exploration performance of NNs, we first develop a structure-guided action selection method, which discards the action violating the greedy structure. This method can increase the exploration effectiveness of actor NN as there is no need to waste time exploring the action space that is certainly not optimal. Moreover, for the critic NN, we formulate the structure-guided loss function to increase the convergence speed by utilizing the evaluation terms to penalize the value estimations of the critic NN that do not follow the proven structure.

- The comprehensive numerical experience shows that the developed structure-guided unified on-off policy DRL algorithm can reduce by 25% to 45% remote state estimation mean square error (MSE) while saving convergence time by about 40% compared with benchmark on-policy DRL algorithm. By incorporating on-policy training methods, the proposed algorithm successfully addresses scheduling problems in systems with up to 40 devices and 20 channels, a scale at which benchmark off-policy DRL algorithms fail to converge. Furthermore, the algorithm achieves a higher structure score compared to the benchmark on-policy DRL algorithm, which implies that the derived scheduling policy is closer to the optimal policy.

1.6 Thesis Outline

The rest of this thesis is structured as follows: In Chapter 2, we introduce the threshold structure of the optimal scheduling policy and develop the structure-enhanced DRL by using this structure to solve the optimal scheduling problem. In Chapter 3, we prove the monotonicity of the Q value function and propose a monotonic critic NN by leveraging this property. In Chapter 4, we derive the convexity of the optimal value function and develop structure-guided unified (dual) on-off policy DRL based on all of the derived structural properties. Finally, in Chapter 5, we summarize the main contributions and results of this thesis, followed by some possible extensions of our research in future work.

Chapter 2

Structure-Enhanced DRL for Optimal Transmission Scheduling

2.1 Introduction

Real-time remote state estimation is important in many networked control applications of Industry 4.0 such as industrial automation, vehicle monitoring systems, building automation, and smart grids [103]. In particular, providing high-quality real-time remote estimation of dynamic system plant states plays an important role in ensuring the overall performance of these networked control applications [10, 11]. For large-scale WNCSs, transmission scheduling of wireless sensors over limited bandwidth needs to be properly designed to guarantee remote estimation performance.

Different from the conventional evaluating metrics in wireless communication systems that aim to optimize QoS performance, remote estimation systems focus on providing accurate state estimation [61, 62]. The remote estimator generates the state estimation based on the latest received local state estimation, so it is difficult to precisely estimate the current state of the process if the remote estimator has not

received the information for a long time because of the limited communication resources and packet loss [104, 105]. To measure the freshness of the information, the AoI, which is defined as the time elapsed from the latest information was successfully received, is used in many works as an evaluating metric [106]. However, the AoI has not considered the inherent features of different processes and cannot show the performance of the remote estimation directly. Therefore, the mean square error (MSE) is the more appropriate metric for the remote state estimation system.

The DRL technology has been widely used to derive the scheduling policy to minimize the MSE, and during the generation of the scheduling action, one needs to take into account the AoI states and channel states [107, 108]. In this chapter, we first prove that the optimal policy has a threshold structure w.r.t these two states and then develop a structure-enhanced DRL (SE-DRL) algorithm for optimal scheduling of the system to achieve the minimum overall estimation MSE.

Notation: Matrices and vectors are denoted by capital and lowercase upright bold letters, e.g., $\mathbf{A}$ and $\mathbf{a}$, respectively. $\Pr(\cdot)$ and $\Pr(\cdot|\cdot)$ denote the probability and conditional probability, respectively.

2.2 System Model

We consider a remote estimation system with N dynamic processes, each measured by a sensor, which pre-processes the raw measurements and sends its state estimates to a remote estimator through one of M wireless channels, as illustrated in Fig. 2.1.

2.2.1 Dynamic Process Model and Local State Estimation

Each dynamic process n is modeled as a discrete-time linear time-invariant (LTI) system as [78, 109, 110]

$$\begin{aligned} \mathbf{x}_{n,t+1} &= \mathbf{A}_n \mathbf{x}_{n,t} + \mathbf{w}_{n,t}, \\ \mathbf{y}_{n,t} &= \mathbf{C}_n \mathbf{x}_{n,t} + \mathbf{v}_{n,t}, n \in \{1, \dots, N\}, t \in \{1, \dots\}, \end{aligned} \tag{2.2.1}$$

where $\mathbf{x}_{n,t} \in \mathbb{R}^{l_n}$ is process n's state at time t, and $\mathbf{y}_{n,t} \in \mathbb{R}^{e_n}$ is the state measurement of the sensor n, $\mathbf{A}_n \in \mathbb{R}^{l_n \times l_n}$ and $\mathbf{C}_n \in \mathbb{R}^{e_n \times l_n}$ are the system matrix and the measurement matrix, respectively, $\mathbf{w}_{n,t} \in \mathbb{R}^{l_n}$ and $\mathbf{v}_{n,t} \in \mathbb{R}^{e_n}$ are the process disturbance and the measurement noise modeled as independent and identically distributed (i.i.d) zero-mean Gaussian random vectors $\mathcal{N}(\mathbf{0}, \mathbf{W}_n)$ and $\mathcal{N}(\mathbf{0}, \mathbf{V}_n)$, respectively. *We assume that the spectral radius of* $\mathbf{A}_n, \forall n$*, is greater than one, which means that the dynamic processes are non-stationary,*[1] making the remote estimation problem more interesting (see [66, 111]).

Due to the presence of noise in (2.2.1), each sensor n executes a classic Kalman filter to pre-process the raw measurement and generate state estimate $\mathbf{x}^s_{n,t}$ at each time t as [66, 109, 112]

$$\mathbf{x}^s_{n,t|t-1} = \mathbf{A}_n \mathbf{x}^s_{n,t-1} \tag{2.2.2a}$$

$$\mathbf{P}^s_{n,t|t-1} = \mathbf{A}_n \mathbf{P}^s_{n,t-1} \mathbf{A}_n^\top + \mathbf{W}_n \tag{2.2.2b}$$

$$\mathbf{K}_{n,t} = \mathbf{P}^s_{n,t|t-1} \mathbf{C}_n^\top (\mathbf{C}_n \mathbf{P}^s_{n,t|t-1} \mathbf{C}_n^\top + \mathbf{V}_n)^{-1} \tag{2.2.2c}$$

$$\mathbf{x}^s_{n,t} = \mathbf{x}^s_{n,t|t-1} + \mathbf{K}_{n,t}(\mathbf{y}_{n,t} - \mathbf{C}\mathbf{x}^s_{n,t|t-1}) \tag{2.2.2d}$$

$$\mathbf{P}^s_{n,t} = (\mathbf{I}_n - \mathbf{K}_{n,t}\mathbf{C}_n)\mathbf{P}^s_{n,t|t-1} \tag{2.2.2e}$$

[1] *When the spectral radius of* $\mathbf{A}_n$ *is greater than* 1*, the plant state,* $\mathbf{x}_{n,t}$*, grows exponentially fast and hence the distribution of* $\mathbf{x}_{n,t}$ *changes over time.*

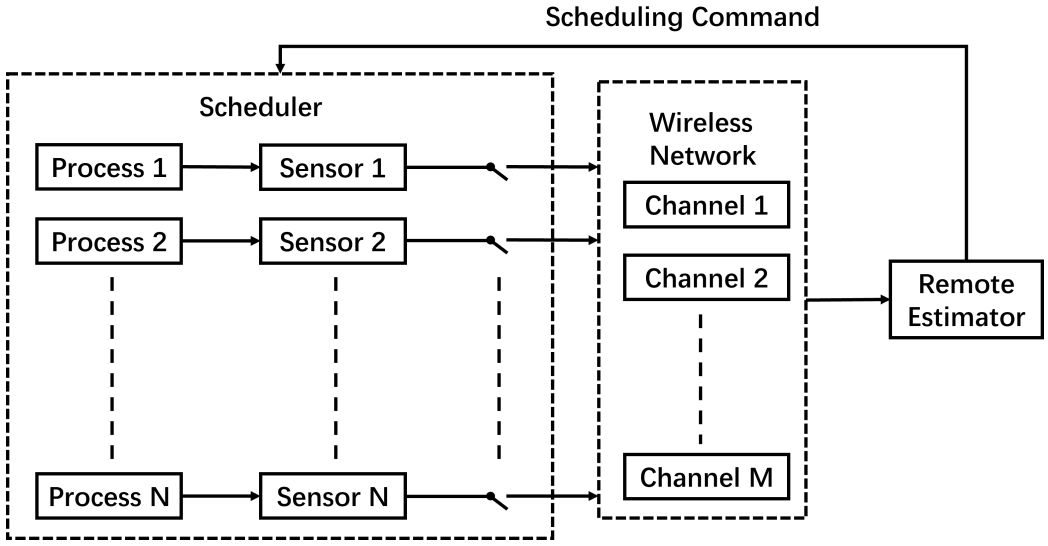


Figure 2.1: Remote state estimation system with N processes and M channels.

where $\mathbf{x}^s_{n,t|t-1}$ and $\mathbf{P}^s_{n,t|t-1}$ are the prior state estimate and the corresponding estimation error covariance of sensor n, respectively, $\mathbf{x}^s_{n,t}$ and $\mathbf{P}^s_{n,t}$ are the posterior state estimate and the corresponding estimation error covariance of sensor n at time t, respectively [29].

In particular, sensor n sends the estimate $\mathbf{x}^s_{n,t}$ to the remote estimator (not $\mathbf{y}_{n,t}$) as a packet, once scheduled, and the local state estimation error covariance matrix is defined as

$$\mathbf{P}^s_{n,t} \triangleq \mathbb{E}\left[(\mathbf{x}^s_{n,t} - \mathbf{x}_{n,t})(\mathbf{x}^s_{n,t} - \mathbf{x}_{n,t})^\top\right]. \tag{2.2.3}$$

$\mathbf{K}_{n,t}$ is the Kalman gain of sensor n, and $\mathbf{I}_n$ is an identity matrix. Note that (2.2.2a) and (2.2.2b) present the prediction steps while (2.2.2c), (2.2.2d), and (2.2.2e) are the updating steps. We note that local Kalman filters are commonly assumed to operate in the steady state mode in the literature (see [66] and references therein[2]).

[2]The nth local Kalman filter converges to steady state if $(\mathbf{A}_n, \mathbf{C}_n)$ is observable and $(\mathbf{A}_n, \sqrt{\mathbf{W}_n})$ is controllable. [113]

We thus assume that the error covariance matrix has converged to a constant, i.e., $\mathbf{P}^s_{n,t} = \bar{\mathbf{P}}_n, \forall t, n$.

2.2.2 Wireless Communications and Remote State Estimation

There are only M wireless channels (e.g., subcarriers) for the N sensors' transmissions, where $M \leq N$. We consider independent and identically distributed (i.i.d.) block fading channels, where the channel quality is fixed during each packet transmission and varies packet by packet, independently. Let the $N \times M$ matrix $\mathbf{H}_t$ denote the channel state of the system at time t, where the element in the nth row and mth column, say $h_{n,m,t} \in \mathcal{H} \triangleq \left\{1, 2, \ldots, \bar{h}\right\}$, represents the channel state between sensor n and the remote estimator at channel m. In particular, there are $\bar{h}$ quantized channel states in total.[3] The distribution of $h_{n,m,t}$ is given as

$$\Pr(h_{n,m,t} = i) = q_i^{(n,m)}, \forall t, \tag{2.2.4}$$

where $\sum_{i=1}^{\bar{h}} q_i^{(n,m)} = 1, \forall n, m$. The instantaneous full channel state $\mathbf{H}_t$ is available at the remote estimator based on standard channel estimation methods [114–117]. The estimated channel state will be used for effectively generating the scheduling command, see e.g., [118–121]. In addition, we also assume perfect synchronization among all network nodes for theoretical analysis tractability.

The packet drop probability at channel state $i' \in \mathcal{H}$ is denoted as $\tilde{p}_{i'}$. Without loss of generality, we assume that $\tilde{p}_1 \geq \tilde{p}_2 \geq \cdots \geq \tilde{p}_{\bar{h}}$. We also define the packet success rate for the channel state $h_{n,m,t}$ as $p_{n,m,t} \in \{p_1, \ldots, p_{\bar{h}}\}$, where $p_{i'} \triangleq 1 - \tilde{p}_{i'}$.

[3]We note that the discrete channel states are quantized from continuous fading channels in practice, such as Rayleigh, Rican, and Nakigami channels; the channel state distributions can be obtained by discretizing the corresponding continuous distribution.

Due to the limited communication channels, only M out of N sensors can be scheduled at each time step. Let $a_{n,t} \in \{0, 1, 2, \ldots, M\}$ represent the channel allocation for sensor n at time t, where

$$a_{n,t} = \begin{cases} 0, & \text{if sensor } n \text{ is not scheduled} \\ m, & \text{if sensor } n \text{ is scheduled to channel } m. \end{cases} \tag{2.2.5}$$

In particular, we assume that each sensor can be scheduled to at most one channel and that each channel is assigned to one sensor [82, 122]. Then, the constraints on $a_{n,t}$ are given as

$$\sum_{m=1}^{M} \mathbb{1}\left(a_{n,t} = m\right) \leq 1, \quad \sum_{n=1}^{N} \mathbb{1}\left(a_{n,t} = m\right) = 1, \tag{2.2.6}$$

where $\mathbb{1}(\cdot)$ is the indicator function.

Considering schedule actions and packet dropouts, sensor n's estimate may not be received by the remote estimator in every time slot. We define the packet reception indicator as

$$\eta_{n,t} = \begin{cases} 1, & \text{if sensor } n\text{'s packet is received at time } t \\ 0, & \text{otherwise.} \end{cases}$$

Considering the randomness of $\eta_{n,t}$ and assuming that the remote estimator performs state estimation at the beginning of each time slot, the remote state estimate that minimizes the estimation MSE follows the stochastic recursion:

$$\hat{\mathbf{x}}_{n,t+1} = \begin{cases} \mathbf{A}_n \mathbf{x}_{n,t}^s, & \text{if } \eta_{n,t} = 1 \\ \mathbf{A}_n \hat{\mathbf{x}}_{n,t}, & \text{otherwise,} \end{cases} \tag{2.2.7}$$

where $\mathbf{A}_n$ is the system matrix of process n defined in (2.2.1). If sensor n's packet is not received, then the remote estimator propagates its estimate in the previous time slot to estimate the current state. From (2.2.1) and (2.2.7), we derive the estimation

error covariance as

$$\mathbf{P}_{n,t} \triangleq \mathbb{E}\left[\left(\hat{\mathbf{x}}_{n,t} - \mathbf{x}_{n,t}\right)\left(\hat{\mathbf{x}}_{n,t} - \mathbf{x}_{n,t}\right)^{\top}\right] \tag{2.2.8}$$

$$= \begin{cases} \mathbf{A}_n \bar{\mathbf{P}}_n \mathbf{A}_n^{\top} + \mathbf{W}_n, & \text{if } \eta_{n,t} = 1 \\ \mathbf{A}_n \mathbf{P}_{n,t} \mathbf{A}_n^{\top} + \mathbf{W}_n, & \text{otherwise,} \end{cases} \tag{2.2.9}$$

where $\bar{\mathbf{P}}_n$ is the local estimation error covariance of sensor n defined under (2.2.3).

Let $\tau_{n,t} \in \{1, 2, \dots\}$ denote the age-of-information (AoI) of sensor n at time t, which measures the amount of time elapsed since the latest sensor packet was successfully received. Then, we have

$$\tau_{n,t+1} = \begin{cases} 1, & \text{if } \eta_{n,t} = 1 \\ \tau_{n,t} + 1, & \text{otherwise.} \end{cases} \tag{2.2.10}$$

If a sensor is frequently scheduled at good channels, then the corresponding average AoI is small. However, due to the scheduling constraint (2.2.6), this is often not possible.

From (2.2.9) and (2.2.10), the error covariances can be written in terms of the AoI as

$$\mathbf{P}_{n,t} = f_n^{\tau_{n,t}}(\bar{\mathbf{P}}_n), \tag{2.2.11}$$

where $f_n(\mathbf{X}) = \mathbf{A}_n \mathbf{X} \mathbf{A}_n^{\top} + \mathbf{W}_n$ and $f_n^{\tau+1}(\cdot) = f_n(f_n^{\tau}(\cdot))$. It has been proved that the estimation MSE, i.e., $\mathrm{Tr}(\mathbf{P}_{n,t})$, *monotonically increases with the AoI state* $\tau_{n,t}$ *[66].*

2.3 Problem Formulation and Threshold structure

In this section, we aim to find a dynamic scheduling policy $\pi(\cdot)$ that uses the AoI states of all sensors, as well as the channel states to minimize the expected total discounted estimation MSE of all N processes over the infinite time horizon.

Problem 2.3.1.

$$\max_{\pi} \lim_{T\to\infty} \mathbb{E}\left[\sum_{t=1}^{T}\sum_{n=1}^{N} -\gamma^t \operatorname{Tr}(\mathbf{P}_{n,t})\right], \tag{2.3.1}$$

where $\gamma \in (0,1)$ *is a discount factor.*

Problem 2.3.1 is a Markovian sequential decision-making problem. This is because the instantaneous estimation MSE, $\mathbf{P}_{n,t}$, only depends on the AoI state $\tau_{n,t}$ in (2.2.11), which is Markovian (2.2.10), and the channel states are i.i.d.. Therefore, we formulate Problem 2.3.1 as an markov decition process (MDP).[4]

2.3.1 MDP Formulation

Given the observable state $\mathbf{s}_t$, including both the channel state and the AoI state, the scheduler generates an action $\mathbf{a}_t$ based on a policy π which is a state-action mapping, i.e., $\mathbf{a}_t = \pi(\mathbf{s}_t)$. An immediate reward is generated to measure the overall estimation quality at the current time step, i.e., the negative sum of estimation MSE, i.e., $\sum_{n=1}^{N} -\operatorname{Tr}(\mathbf{P}_{n,t})$, which is determined by $\mathbf{s}_t$. Due to the Markovian property of the decision-making process, the probability of state transitioning from one to the next only relies on the current state and action, which can be written as $\Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$. Under the state transition rule, the overall target of the MDP problem is to optimize the policy $\pi(\cdot)$ for maximizing the expected total discounted reward over the infinite time horizon. The formal definition of the MDP are given below:

1) The state of the MDP, taking into account both the AoI and channel states, is defined as $\mathbf{s}_t \triangleq (\boldsymbol{\tau}_t, \mathbf{H}_t) \in \mathcal{S} \triangleq \mathbb{N}^N \times \mathcal{H}^{N\times M}$, where $\boldsymbol{\tau}_t = (\tau_{1,t}, \tau_{2,t}, \ldots, \tau_{N,t}) \in \mathbb{N}^N$ is

[4]Not all MDPs for transmission scheduling of remote estimation systems have a feasible solution. However, once the remote estimation stability condition in terms of the dynamic process parameters and the channel statistics is satisfied, the MDP has a solution. We assume that the remote estimation stability condition of our system is satisfied, and only focus on the optimal solution of the MDP in the rest of the chapter. The detailed stability condition can be found in our previous work [82].

the AoI state vector.

2) The overall schedule action of the N sensors is defined as $\mathbf{a}_t = (a_{1,t}, a_{2,t}, \ldots, a_{N,t}) \in \mathcal{A} \triangleq \{0, 1, 2, \ldots, M\}^N$ under the constraint (2.2.6). There are $N!/(N-M)!$ actions of the N-choose-M problem in total.

3) The transition probability $\Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ is the probability of the next state $\mathbf{s}_{t+1}$ given the current state $\mathbf{s}_t$ and the action $\mathbf{a}_t$. Since the state transition is independent of the time index given the action $\mathbf{a}$ and the state $\mathbf{s}$, we drop the subscript t here and use $\mathbf{s}$ and $\mathbf{s}^+$ to represent the current and the next states, respectively. Due to the i.i.d. fading channel states, we have

$$\Pr(\mathbf{s}^+|\mathbf{s}, \mathbf{a}) = \Pr(\boldsymbol{\tau}^+|\boldsymbol{\tau}, \mathbf{H}, \mathbf{a}) \Pr(\mathbf{H}^+), \tag{2.3.2}$$

where $\Pr(\mathbf{H}^+)$ can be obtained from (2.2.4), and $\Pr(\boldsymbol{\tau}^+|\boldsymbol{\tau}, \mathbf{H}, \mathbf{a}) = \prod_{n=1}^{N} \Pr(\tau_n^+|\tau_n, \mathbf{H}, a_n)$ and

$$\Pr(\tau_n^+|\tau_n, \mathbf{H}, a_n) = \begin{cases} p_{n,m}, & \text{if } \tau_n^+ = 1, a_n = m \\ 1 - p_{n,m}, & \text{if } \tau_n^+ = \tau_n + 1, a_n = m \\ 1, & \text{if } \tau_n^+ = \tau_n + 1, a_n = 0 \\ 0, & \text{otherwise,} \end{cases} \tag{2.3.3}$$

which is derived based on (2.2.4) and (2.2.10).

4) The immediate reward at time t is defined as the negative sum estimation MSE, i.e., $\sum_{n=1}^{N} -\operatorname{Tr}(\mathbf{P}_{n,t})$, to measure the overall remote estimation quality. Since $\mathbf{P}_{n,t}$ defined in (2.2.11) is a function of the AoI state $\tau_{n,t}$, the reward is denoted as $r(\mathbf{s}_t)$, *monotonically decreasing with each AoI state*.

Note that the formulated MDP problem cannot be solved by conventional algorithms such as value and policy iterations due to the high computation complexity even for small-scale systems (e.g., $N = 6$ and $M = 3$) [123].

2.3.2 Threshold Structure of the Optimal MDP Solution

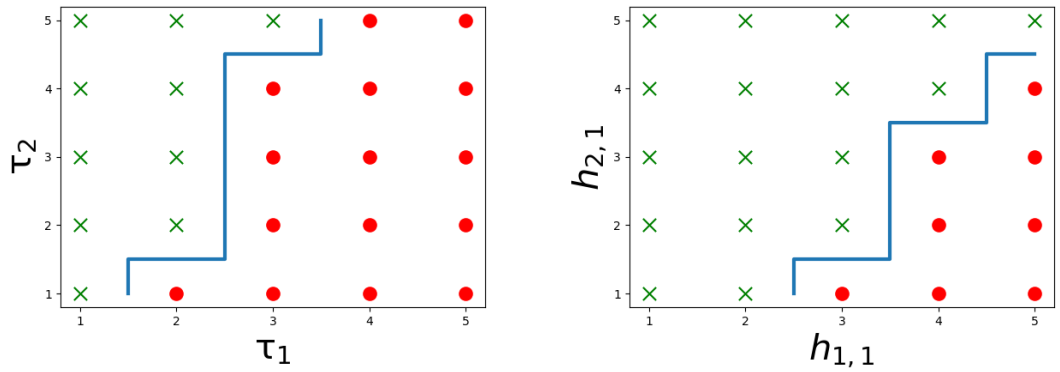
Unlike the existing works [65, 87], which only considered oversimplified systems over static channels, we aim to derive the structural property of the optimal policy of the general multi-sensor-multi-channel system over fading channels as below.

Definition 2.3.1 (Channel-State Threshold Policy)**.** *For a channel-state threshold scheduling policy, if channel m is assigned to sensor n at the state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, then for state $\mathbf{s}' = (\boldsymbol{\tau}, \mathbf{H}'_{n,m})$, where $\mathbf{H}'_{n,m}$ is identical to $\mathbf{H}$ except the sensor-n-channel-m state with $h'_{n,m} > h_{n,m}$, then channel m is still assigned to sensor n.*

Definition 2.3.2 (AoI-State Threshold Policy)**.** *For an AoI-state threshold scheduling policy, if channel m is assigned to sensor n at the state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, then for state $\mathbf{s}' = (\boldsymbol{\tau}'_{(n)}, \mathbf{H})$, where $\boldsymbol{\tau}'_{(n)}$ is idential to $\boldsymbol{\tau}$ except sensor n's AoI with $\tau'_n \geq \tau_n$, then either channel m or a better channel is assigned to sensor n.*

Definition 2.3.1 states that sensor n is scheduled at channel m at a certain state, if the channel quality of $h_{n,m}$ improves while the AoI and the other channel states are the same, then the threshold policy still assigns channel m to sensor n. For Definition 2.3.2, if the AoI state of sensor n is increased while the other states remain the same, the policy must schedule sensor n to a channel that is no worse than the previous one.

To illustrate that an optimal policy may have a threshold structure, we find the optimal policy of a two-sensor-single-channel system by solving the MDP with the conventional value iteration algorithm, which is illustrated in Fig. 2.2. We see that action-switching curves exist in both AoI and channel state spaces, and the properties in Definitions 2.3.1 and 2.3.2 are observed. Inspired by the above result, in



(a) Schedule actions at AoI states (b) Schedule actions at channel states

Figure 2.2: Structure of the optimal scheduling policy with $N = 2$ and $M = 1$, where $\bullet$ and $\times$ represent the schedule of sensor 1 and 2, respectively.

the following, we will prove that the optimal policy has the structural properties in Definitions 2.3.1 and 2.3.2.[5]

2.4 Proof of the Threshold Structure of Optimal Policies

We derive the structural properties of the optimal scheduling policy by using value iteration concepts, which require the definition of the optimal value function, $V^*(\mathbf{s}_t) : \mathcal{S} \to \mathbb{R}$, and the state-action value function, $Q(\mathbf{s}_t, \mathbf{a}_t) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as below.

Given the current state $\mathbf{s}_t = (\boldsymbol{\tau}_t, \mathbf{H}_t)$, the optimal value function is the maximum expected discounted sum of the future reward, i.e., achieved by the optimal policy

[5]We note that in [65, 87], only threshold structural results in terms of the AoI states are proved over static channels. Moreover, the proofs have some limitations: 1) In [65], the authors proved the threshold structure for a multi-dimensional state and action scenario based on Theorem 8.11.3 in [123], which, however, can only be used for single-dimensional scenarios. 2) In [87], the proof of the optimal policy's structural property only considered a part of the action space; however, the full action space must be examined to show the optimality.

$\pi^*(\cdot)$:

$$V^*(\mathbf{s}_t) = \max_{\pi} \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r\left(\mathbf{s}_{t'}\right)\right]. \tag{2.4.1}$$

The optimal value function satisfies the Bellman equation:

$$V^*(\mathbf{s}_t) = r(\mathbf{s}_t) + \gamma \max_{\mathbf{a}_t \in \mathcal{A}} \left[\sum_{\mathbf{s}_{t+1}} \Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) V^*(\mathbf{s}_{t+1})\right], \tag{2.4.2}$$

where the optimal action $\mathbf{a}^*$ is obtained by the optimal policy $\pi^*(\cdot)$, i.e.,

$$\mathbf{a}_t^* \triangleq \pi^*(\mathbf{s}_t) = \arg\max_{\mathbf{a}_t \in \mathcal{A}} \left[\sum_{\mathbf{s}_{t+1}} \Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) V^*(\mathbf{s}_{t+1})\right]. \tag{2.4.3}$$

Given the current state-action pair $\mathbf{s}_t$ and $\mathbf{a}_t$, the state-action value function, which is also called the Q-value function, measures the expected discounted sum of the future cost under the optimal policy $\pi^*(\cdot)$ as

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t) + \gamma \sum_{\mathbf{s}_{t+1}} \Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) V^*(\mathbf{s}_{t+1}). \tag{2.4.4}$$

From (2.4.2) and (2.4.4), it directly follows that the optimal value function and the Q-value function satisfy:

$$V^*(\mathbf{s}_t) = Q(\mathbf{s}_t, \mathbf{a}_t^*) \geq Q(\mathbf{s}_t, \mathbf{a}_t). \tag{2.4.5}$$

For notation simplicity, we use $\mathbf{a}, \mathbf{s}, \mathbf{s}^+$ to represent $\mathbf{a}_t, \mathbf{s}_t, \mathbf{s}_{t+1}$ in the following.

We prove the channel-state and the AoI-state threshold properties of the optimal scheduling policy based on the classical value iteration algorithm, as it can achieve the optimal solution [124, 125]. In the value iteration, the initial value function and its $\tilde{t}$-th iteration are $V^0(\mathbf{s}) \in \mathcal{V}$ and $V^{\tilde{t}}(\mathbf{s}) \in \mathcal{V}$, respectively, where $\mathcal{V}$ is the set of any measurable function, i.e., $\mathcal{V} : \mathcal{S} \to \mathbb{R}$. At the $\tilde{t}$-th iteration, we have $V^{\tilde{t}+1} = \mathsf{B}\left[V^{\tilde{t}}\right]$, where $\mathsf{B}[\cdot] : \mathcal{V} \to \mathcal{V}$ is the Bellman operator:

$$\mathsf{B}\left[V^{\tilde{t}}\right](\mathbf{s}) = r(\mathbf{s}) + \gamma \max_{\mathbf{a} \in \mathcal{A}} \left[\sum_{\mathbf{s}^+} \Pr(\mathbf{s}^+|\mathbf{s}, \mathbf{a}) V^{\tilde{t}}(\mathbf{s}^+)\right]. \tag{2.4.6}$$

We next elucidate the convergence and the optimality of value iterations.

Lemma 2.4.1 ([124, 125]). *If the optimal policy exists, then the operator* B *has a unique fixed point* $V^* \in \mathcal{V}$ *and for all* $V^0 \in \mathcal{V}$*, the sequence* $\{V^{\tilde{t}}\}$ *defined by* $V^{\tilde{t}+1} = \mathsf{B}[V^{\tilde{t}}]$ *converges in norm to* V^**, i.e.*

$$\lim_{\tilde{t}\to\infty} V^{\tilde{t}} = V^*. \tag{2.4.7}$$

Before proceeding further, we derive the following technical lemma about the monotonicity of the optimal value function.

Lemma 2.4.2 (Monotonicity). *Consider states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$*, where* $\boldsymbol{\tau}'_{(i)} = (\tau_1, \ldots, \tau'_i, \ldots, \tau_N)$ *and* $\tau'_i \geq \tau_i$*. The following holds*

$$V^*(\mathbf{s}') \leq V^*(\mathbf{s}). \tag{2.4.8}$$

Proof. See Appendix A.1. □

Lemma 2.4.2 will assist in the comparison of value functions with different AoI states in the proof of the channel-state threshold and the AoI-state threshold properties.

2.4.1 Channel-State Threshold Property

In this part, we completely prove that the optimal policy of a general multi-sensor-multi-channel system is of the channel-state threshold property.

Theorem 2.4.1. *The optimal policy* $\pi^*(\cdot)$ *of the multi-sensor-multi-channel system has the channel-state threshold property in Definition 2.3.1.*

Proof. From the Q-value definition in (2.4.4), the theorem can be translated as: if for state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, the inequality $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a}), \forall \mathbf{a} \in \mathcal{A}$ exists, where $a_i^* = m$, then for state $\mathbf{s}' = (\boldsymbol{\tau}, \mathbf{H}'_{i,m})$, where $\mathbf{H}'_{i,m}$ and $\mathbf{H}$ are identical except the element

$h'_{i,m} > h_{i,m}$, the following inequality holds $Q(\mathbf{s}', \mathbf{a}'^*) \geq Q(\mathbf{s}', \mathbf{a})$, where $\mathbf{a}'^*$ is the optimal action at the state $\mathbf{s}'$ with $a_i'^* = m$. Since $Q(\mathbf{s}', \mathbf{a}'^*) \geq Q(\mathbf{s}', \mathbf{a}^*)$, we only need to prove $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a}')$, where $a'_i \neq m$. To prove $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a}')$, we will prove $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a}^*)$ and then $Q(\mathbf{s}, \mathbf{a}') = Q(\mathbf{s}', \mathbf{a}')$. In the following, we use $\mathbf{H}'$ to represent $\mathbf{H}'_{i,m}$ for the notation simplicity.

First, from (2.2.4) and (2.2.10), we have

$$\Pr(\boldsymbol{\tau}^+|\boldsymbol{\tau}, \mathbf{H}, \mathbf{a}) = \prod_{n=1}^{N} \Pr(\tau_n^+|\tau_n, \mathbf{h}_n, a_n) = \Pr(\tau_i^+|\tau_i, \mathbf{h}_i, a_i) \Pr(\boldsymbol{\tau}_{\backslash i}^+|\boldsymbol{\tau}_{\backslash i}, \mathbf{H}_{\backslash i}, \mathbf{a}_{\backslash i}), \quad (2.4.9)$$

where $\boldsymbol{\tau}_{\backslash i} = (\tau_1, \ldots, \tau_{i-1}, \tau_{i+1}, \ldots, \tau_N)$ and $\mathbf{a}_{\backslash i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_N)$ represent the AoI states and the actions of all sensors except sensor i, respectively, and $\mathbf{h}_i = (h_{i,1}, h_{i,2}, \ldots, h_{i,M})$ is the vector channel states between sensor i and the remote estimator and $\mathbf{H}_{\backslash i} = (\mathbf{h}_1, \ldots, \mathbf{h}_{i-1}, \mathbf{h}_{i+1}, \ldots, \mathbf{h}_N)$. By using (2.3.2) and (2.4.9), it can be derived that

$$\begin{aligned} Q(\mathbf{s}, \mathbf{a}) =& r(\mathbf{s}) + \gamma \sum_{\mathbf{s}^+} \Pr(\mathbf{s}^+|\mathbf{s}, \mathbf{H}, \mathbf{a}) V^*(\mathbf{s}^+) \\ =& r(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}^+} \Pr(\mathbf{H}^+) \Pr(\boldsymbol{\tau}^+|\boldsymbol{\tau}, \mathbf{H}, \mathbf{a}) V^*(\mathbf{s}^+) \\ =& r(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}_{\backslash i}^+} \sum_{\tau_i^+} \Pr(\mathbf{H}^+) \Pr(\boldsymbol{\tau}_{\backslash i}^+|\boldsymbol{\tau}_{\backslash i}, \mathbf{H}_{\backslash i}, \mathbf{a}_{\backslash i}) \Pr(\tau_i^+|\tau_i, \mathbf{h}_i, a_i) V^*(\mathbf{s}^+). \end{aligned} \quad (2.4.10)$$

Based on (2.4.10), we have

$$\begin{aligned} Q(\mathbf{s}', \mathbf{a}^*) &= r(\mathbf{s}') + \gamma \sum_{\mathbf{H}'^+} \sum_{\boldsymbol{\tau}_{\backslash i}^+} \sum_{\tau_i^+} \Pr(\mathbf{H}'^+) \Pr(\boldsymbol{\tau}_{\backslash i}^+|\boldsymbol{\tau}_{\backslash i}, \mathbf{H}_{\backslash i}, \mathbf{a}_{\backslash i}^*) \Pr(\tau_i^+|\tau_i, \mathbf{h}'_i, a_i^*) V^*(\mathbf{s}'^+) \\ &\geq r(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}_{\backslash i}^+} \sum_{\tau_i^+} \Pr(\mathbf{H}^+) \Pr(\boldsymbol{\tau}_{\backslash i}^+|\boldsymbol{\tau}_{\backslash i}, \mathbf{H}_{\backslash i}, \mathbf{a}_{\backslash i}^*) \Pr(\tau_i^+|\tau_i, \mathbf{h}_i, a_i^*) V^*(\mathbf{s}^+) = Q(\mathbf{s}, \mathbf{a}^*), \end{aligned} \quad (2.4.11)$$

where the inequality is derived by replacing the parameter $\mathbf{H}'^+$ with $\mathbf{H}^+$, and then

using $r(\mathbf{s}') = r(\mathbf{s})$, $h'_{i,m} > h_{i,m}, a_i^* = m$, and the following inequality

$$\begin{aligned} & p'_{i,m} V^*(1, \boldsymbol{\tau}_{\setminus i}^+, \mathbf{H}^+) + (1 - p'_{i,m}) V^*(\tau_i + 1, \boldsymbol{\tau}_{\setminus i}^+, \mathbf{H}^+) \\ & \geq p_{i,m} V^*(1, \boldsymbol{\tau}_{\setminus i}^+, \mathbf{H}^+) + (1 - p_{i,m}) V^*(\tau_i + 1, \boldsymbol{\tau}_{\setminus i}^+, \mathbf{H}^+) \end{aligned} \tag{2.4.12}$$

achieved by $p'_{i,m} \geq p_{i,m}$ and $V^*(1,\boldsymbol{\tau}_{\setminus i}^+,\mathbf{H}^+) \geq V^*(\tau_i+1,\boldsymbol{\tau}_{\setminus i}^+,\mathbf{H}^+)$ from Lemma 2.4.2.

Second, we derive that

$$\begin{aligned} Q(\mathbf{s}, \mathbf{a}) = & r(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}_{\setminus i}^+} \sum_{\tau_i^+} \Pr(\mathbf{H}^+) \Pr(\boldsymbol{\tau}_{\setminus i}^+ | \boldsymbol{\tau}_{\setminus i}, \mathbf{H}_{\setminus i}, \mathbf{a}'_{\setminus i}) \Pr(\tau_i^+ | \tau_i, \mathbf{h}_i, a'_i) V^*(\mathbf{s}^+) \\ = & r(\mathbf{s}') + \gamma \sum_{\mathbf{H}'^+} \sum_{\boldsymbol{\tau}_{\setminus i}^+} \sum_{\tau_i^+} \Pr(\mathbf{H}'^+) \Pr(\boldsymbol{\tau}_{\setminus i}^+ | \boldsymbol{\tau}_{\setminus i}, \mathbf{H}_{\setminus i}, \mathbf{a}'_{\setminus i}) \Pr(\tau_i^+ | \tau_i, \mathbf{h}'_i, a'_i) V^*(\mathbf{s}'^+) = Q(\mathbf{s}', \mathbf{a}), \end{aligned} \tag{2.4.13}$$

where the second equality is derived by replacing the parameter $\mathbf{H}'^+$ with $\mathbf{H}^+$, and then using $h'_{i,m} > h_{i,m}, a'_i \neq m$, and $\Pr(\tau_i^+ | \tau_i, \mathbf{h}_i, a'_i) V^*(\mathbf{s}^+) = \Pr(\tau_i^+ | \tau_i, \mathbf{h}'_i, a'_i) V^*(\mathbf{s}'^+)$.

□

2.4.2 AoI-State Threshold Property

We consider three network scenarios with different numbers of sensors and channels as below.

Two-sensor-single-channel systems

As presented in Theorem 2.4.2, we prove that the optimal policy of a two-sensor-single-channel system is an AoI-state threshold policy in Definition 2.3.2. In other words, if the optimal action at a state is to schedule sensor n, then the optimal action is still to schedule sensor n at all states that only increase sensor n's AoI.

Theorem 2.4.2. *The optimal policy $\pi^*(\cdot)$ of the two-sensor-single-channel system has*

the AoI-state threshold property in Definition 2.3.2: suppose that for state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$*, the optimal action is to schedule sensor* i*, i.e.* $a_i^* = 1$*, then for state* $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$*, where* $\tau'_i \geq \tau_i$*, the optimal action is still* $a_i'^* = 1$*.*

Remark 2.4.1 (Analytical Challenges)**.** *Although Theorem 2.4.2 looks simple, the proof is highly nontrivial.*

First, the main difficulty of the proof is that different actions cause the transition of the AoI states in multiple dimensions, resulting in the comparison of multi-dimensional value functions. However, Lemma 2.4.2 can only compare the optimal value functions with AoI state changes within one dimension. For example, given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H}), \mathbf{s}' = (\boldsymbol{\tau}', \mathbf{H})$*, and* $\mathbf{s}^\circ = (\boldsymbol{\tau}^\circ, \mathbf{H})$*, where* $\boldsymbol{\tau} = (\tau_i, \tau_j), \boldsymbol{\tau}' = (\tau'_i, \tau_j), \boldsymbol{\tau}^\circ = (\tau'_i, \tau''_j), \tau'_i \geq \tau_i$*, and* $\tau''_j \leq \tau_j$*, it is easy to have* $V(\mathbf{s}) \geq V(\mathbf{s}')$*, but is not possible to compare* $V(\mathbf{s})$ *and* $V(\mathbf{s}^\circ)$*.*

Second, one may think that the proof of Theorem 2.4.2 is intuitive as a larger AoI of sensor i *results in a lower reward, and scheduling sensor* i *can improve the reward more efficiently than scheduling other sensors. This is a misunderstanding. Theorem 2.4.2 is equivalent to saying that the inequality* $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$ *can be derived from* $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$*, where* $a_i^* = 1$*. We drop the constant channel state,* $\mathbf{H}$*, in the optimal value functions for simplicity. Based on* (2.4.4)*,* $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$ *is equal to*

$$p_{i,1}V^*(1, \tau_j+1)+(1-p_{i,1})V^*(\tau'_i+1, \tau_j+1) \geq p_{j,1}V^*(\tau'_i+1, 1)+(1-p_{j,1})V^*(\tau'_i+1, \tau_j+1), \tag{2.4.14}$$

and $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$ *is equal to*

$$p_{i,1}V^*(1, \tau_j+1)+(1-p_{i,1})V^*(\tau_i+1, \tau_j+1) \geq p_{j,1}V^*(\tau_i+1, 1)+(1-p_{j,1})V^*(\tau_i+1, \tau_j+1). \tag{2.4.15}$$

We see that (2.4.14) *cannot be derived directly based on* (2.4.15)*, since it also related to the packet success rates of different sensors, i.e.* $p_{i,1}$ *and* $p_{j,1}$*.*

Proof. As mentioned in Remark 2.4.1, we will prove (2.4.14) based on (2.4.15), which depends on the key technical lemma below in terms of the optimal value functions and the packet success rates, which depends on the channel states.

Definition 2.4.1 (Meet and Joint [123, 126])**.** *Let* $x' \vee x'' = \max\{x', x''\}$ *and* $x' \wedge x'' = \min\{x', x''\}$ *denote the join and meet of two real numbers, respectively. Define* $\mathbf{x}' \vee \mathbf{x}'' = (x_1' \vee x_1'', \ldots, x_n' \vee x_n'')$ *and* $\mathbf{x}' \wedge \mathbf{x}'' = (x_1' \wedge x_1'', \ldots, x_n' \wedge x_n'')$ *as the joint and meet of the vectors* $\mathbf{x}'$ *and* $\mathbf{x}'' \in \mathbb{R}^n$*, respectively.*

Lemma 2.4.3 (Probabilistic Supermodularity, $N = 2$, $M = 1$)**.** *Given states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}^\circ = (\boldsymbol{\tau}^\circ, \mathbf{H})$*, where* $\boldsymbol{\tau}^\circ = (\tau_i'', \tau_j'), \tau_i'' \leq \tau_i$*,* $\tau_j' \geq \tau_j$*, and* $i \neq j \in \{1, 2\}$*, the following holds*

$$p_{j,,1} V^*(\mathbf{s} \wedge \mathbf{s}^\circ) + (p_{j,1} - p_{i,1}) V^*(\mathbf{s} \vee \mathbf{s}^\circ) \geq p_{j,1} V^*(\mathbf{s}) + (p_{j,1} - p_{i,1}) V^*(\mathbf{s}^\circ). \quad (2.4.16)$$

Proof. See Appendix A.2. □

In what follows, we denote the state as $\mathbf{s} = (\tau_i, \tau_j)$, since the transition of the channel states is constant during the proof. From (2.4.15), it directly follows that

$$p_{i,1} V^*(1, \tau_j{+}1) \geq p_{j,1} V^*(\tau_i{+}1, 1) - (p_{j,1} - p_{i,1}) V^*(\tau_i{+}1, \tau_j{+}1). \quad (2.4.17)$$

By applying Lemma 2.4.3 to the right-hand side of (2.4.17), we have

$$p_{i,1} V^*(1, \tau_j{+}1) \geq p_{j,1} V^*(\tau_{i,1}'{+}1, 1) - (p_{j,1} - p_{i,1}) V^*(\tau_i'{+}1, \tau_j{+}1), \quad (2.4.18)$$

which is exactly (2.4.14). Thus, we have proved $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$. □

Multi-sensor-single-channel systems

It is intractable to prove that the optimal policy has the AoI-state threshold property in this case. Similar to the two-sensor case, we first try to prove the probabilistic supermodularity. However, there are more than two AoI states changing when applying value iterations, making it difficult to find a set of useful inequalities to prove the target inequality. We can also construct some other inequalities in terms of the optimal value function and the packet success rates that can be used to prove the threshold property, but these inequalities cannot be proved either.

Although the AoI-state threshold property of the optimal policy is not fully derived, we prove an asymptotic structural property as below.

Theorem 2.4.3. *The optimal policy $\pi^*(\cdot)$ of a multi-sensor-single-channel system with $N > 2$ has an asymptotic AoI-state threshold property: suppose that for state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, the optimal action is to transmit the estimation of sensor i, i.e. $a_i^* = 1$, then for all states $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, where $\tau'_i \gg \tau_i$, the optimal policy is still $a_i'^* = 1$.*

Theorem 2.4.3 shows that the threshold structure still exists in the optimal policy of a multi-sensor-single-channel system, at least in the large AoI domain.

Proof. This theorem is equivalent to stating that if for state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, the inequality $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a}), \forall \mathbf{a} \in \mathcal{A}$ exists, where $a_i^* = 1$, then for state $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, where $\tau'_i \gg \tau_i$, we have $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$. Similar to Theorem 2.4.2, we need to develop the following lemmas yet in an asymptotic manner.

Lemma 2.4.4. *Given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ and $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, where $\tau'_i \gg \tau_i$, the following holds*

$$V^*(\mathbf{s}') \ll V^*(\mathbf{s}). \tag{2.4.19}$$

Proof. See Appendix A.3. □

Based on Lemma 2.4.4, we derive the asymptotic probabilistic supermodularity below.

Lemma 2.4.5 (Asymptotic Probabilistic Supermodularity, $N > 2$, $M = 1$)**.** *Given states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}^\circ = (\boldsymbol{\tau}^\circ, \mathbf{H})$*, where* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_i, \ldots, \tau_j, \ldots, \tau_N)$ *and* $\boldsymbol{\tau}^\circ = (\tau_1, \ldots, \tau_i'', \ldots, \tau_j', \ldots, \tau_N)$ *with* $\tau_i \gg \tau_i''$*,* $\tau_j \leq \tau_j'$*, and* $i \neq j \in \{1, \ldots, N\}$*, the probabilistic supermodularity in* (2.4.16) *holds.*

Proof. See Appendix A.4. □

In the following, we will prove $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$ based on $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$ and similar to the proof of Theorem 2.4.2, we write the state as $\mathbf{s} = (\tau_i, \tau_j, \boldsymbol{\tau}_{\setminus i,j})$. Using (2.4.4), $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$ is equal to

$$p_{i,1} V^*(1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}) \geq p_{j,1} V^*(\tau_i + 1, 1, \boldsymbol{\tau}^+_{\setminus i,j}) - (p_{j,1} - p_{i,1}) V^*(\tau_i + 1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}). \quad (2.4.20)$$

By applying Lemma 2.4.5 to the right-hand side of (2.4.20), we have

$$p_{i,1} V^*(1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}) \geq p_{j,1} V^*(\tau_i' + 1, 1, \boldsymbol{\tau}^+_{\setminus i,j}) - (p_{j,1} - p_{i,1}) V^*(\tau_i' + 1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}), \quad (2.4.21)$$

and thus

$$\begin{aligned} &p_{i,1} V^*(1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}) + (1 - p_{i,1}) V^*(\tau_i' + 1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}) \\ &\geq p_{j,1} V^*(\tau_i' + 1, 1, \boldsymbol{\tau}^+_{\setminus i,j}) + (1 - p_{j,1}) V^*(\tau_i' + 1, \tau_j + 1, \boldsymbol{\tau}^+_{\setminus i,j}), \end{aligned} \quad (2.4.22)$$

which is exactly $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a})$ based on (2.4.4). □

Multi-sensor-multi-channel systems

This case is more complex than the single-channel one, due to the increased state dimension introduced by the multi-dimensional channel. Thus, the AoI-state threshold property of the optimal policy cannot be derived strictly. In the following, we develop another structural property of the optimal policy.

Proposition 2.4.1. *If for state* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$*, the optimal action is* $\mathbf{a}^* = (a_1^*, \ldots, a_i^*, \ldots, a_N^*), a_i^* = m$*, and for state* $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$*, where* $\tau_i' \geq \tau_i$*, the optimal channel assignments except for channel* m *keep constant, i.e.* $a_n^* = {a'_n}^*, \forall a_n^* \neq 0$*, and* $i \neq n$*, then the optimal action for sensor* j *at state* $\mathbf{s}'$ *is to not occupy channel* m*, if sensor* j *is not scheduled at state* $\mathbf{s}$*, i.e.,* ${a_j}^* = 0$*, and its channel condition is worse than that of sensor* i*, i.e.,* $h_{i,m} \geq h_{j,m}$*.*

Proposition 2.4.1 says that sensor i's allocated channel won't be occupied by other sensors with worse channel conditions when sensor i's AoI increases and the optimal channel assignment rules for other channels remain the same.

Proof. This proposition equivalent to that if for state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, the inequality $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$ exists, where $a_i^* = m$ and $a_j^* = 0$, then for state $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, where $\tau_i' \geq \tau_i$, the following inequality holds $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a}')$, where $a_j' = m, \forall h_{i,m} \geq h_{j,m}$ and $a^*_{\setminus i,j} = a'_{\setminus i,j}$.

Therefore, we will prove $Q(\mathbf{s}', \mathbf{a}^*) \geq Q(\mathbf{s}', \mathbf{a}')$ based on $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a})$ in the

following. Taking (2.4.10) into $Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}, \mathbf{a}')$, we have

$$\sum_{\boldsymbol{\tau}^+_{\setminus i,j}} \sum_{\mathbf{H}^+} \sum_{\tau_i^+} \sum_{\tau_j^+} \Pr(\boldsymbol{\tau}^+_{\setminus i,j} | \boldsymbol{\tau}_{\setminus i,j}, \mathbf{H}_{\setminus i,j}, \mathbf{a}^*_{\setminus i,j}) \Pr(\mathbf{H}^+) \Pr(\tau_i^+ | \tau_i, \mathbf{h}_i, a_i^*) \Pr(\tau_j^+ | \tau_j, \mathbf{h}_j, a_j^*) V^*(\mathbf{s}^+)$$

$$\geq \sum_{\boldsymbol{\tau}^+_{\setminus i,j}} \sum_{\mathbf{H}^+} \sum_{\tau_i^+} \sum_{\tau_j^+} \Pr(\boldsymbol{\tau}^+_{\setminus i,j} | \boldsymbol{\tau}_{\setminus i,j}, \mathbf{H}_{\setminus i,j}, \mathbf{a}'_{\setminus i,j}) \Pr(\mathbf{H}^+) \Pr(\tau_i^+ | \tau_i, \mathbf{h}_i, a_i') \Pr(\tau_j^+ | \tau_j, \mathbf{h}_j, a_j') V^*(\mathbf{s}^+). \tag{2.4.23}$$

Based on $a^*_{\setminus i,j} = a'_{\setminus i,j}$, we have

$$\Pr(\boldsymbol{\tau}^+_{\setminus i,j} | \boldsymbol{\tau}_{\setminus i,j}, \mathbf{H}_{\setminus i,j}, \mathbf{a}^*_{\setminus i,j}) = \Pr(\boldsymbol{\tau}^+_{\setminus i,j} | \boldsymbol{\tau}_{\setminus i,j}, \mathbf{H}_{\setminus i,j}, \mathbf{a}'_{\setminus i,j}). \tag{2.4.24}$$

Thus, from (2.4.23), the following inequality can be derived

$$\begin{aligned} &\sum_{\tau_i^+} \sum_{\tau_j^+} \Pr\left(\tau_i^+ | \tau_i, \mathbf{h}_i, a_i^*\right) \Pr\left(\tau_j^+ | \tau_j, \mathbf{h}_j, a_j^*\right) V^*\left(\mathbf{s}^+\right) \\ &\geq \sum_{\tau_i^+} \sum_{\tau_j^+} \Pr\left(\tau_i^+ | \tau_i, \mathbf{h}_i, a_i'\right) \Pr\left(\tau_j^+ | \tau_j, \mathbf{h}_j, a_j'\right) V^*\left(\mathbf{s}^+\right). \end{aligned} \tag{2.4.25}$$

For notation simplicity, the state can be rewritten as $\mathbf{s} = (\tau_i, \tau_j)$, because the states except for τ_i and τ_j are constant. Based on $a_i^* = m, a_j^* = 0, a_i' = 0$, and $a_j' = m$, the left-hand side and the right hand side of (2.4.23) are equal to

$$\begin{cases} p_{i,m} V^*(1, \tau_j + 1) + (1 - p_{i,m}) V^*(\tau_i + 1, \tau_j + 1) \\ p_{j,m} V^*(\tau_i + 1, 1) + (1 - p_{j,m}) V^*(\tau_i + 1, \tau_j + 1). \end{cases} \tag{2.4.26}$$

Then, the following inequality can be derived

$$\begin{aligned} &p_{i,m} V^*(1, \tau_j + 1) \\ &\geq p_{j,m} V^*(\tau_i + 1, 1) + (p_{i,m} - p_{j,m}) V^*(\tau_i + 1, \tau_j + 1) \\ &\geq p_{j,m} V^*(\tau_i' + 1, 1) + (p_{i,m} - p_{j,m}) V^*(\tau_i' + 1, \tau_j + 1), \end{aligned} \tag{2.4.27}$$

where the first inequality is derived by (2.4.23), (2.4.24), and (2.4.26), and the second inequality is by using Lemma 2.4.2 and $p_{i,m} \geq p_{j,m}$. The inequality (2.4.27) can be

rewritten as

$$p_{i,m}V^*(1,\tau_j+1)+(1-p_{i,m})V^*(\tau_i'+1,\tau_j+1)\geq p_{j,m}V^*(\tau_i'+1,1)+(1-p_{j,m})V^*(\tau_i'+1,\tau_j+1). \tag{2.4.28}$$

According to the analysis from (2.4.23) to (2.4.26), it directly follows $Q(\mathbf{s}',\mathbf{a}^*)\geq Q(\mathbf{s}',\mathbf{a}')$ from (2.4.28). □

Remark 2.4.2 (Generality of the derived results)**.** *We note that all the theoretical results proved in this section merely rely on two features of the formulated optimal scheduling problem: 1) the reward function of each user is a monotonically decreasing function in terms of its AoI state, and 2) the total reward function of the problem is the sum of individual rewards. In other words, for other scheduling problems of different systems, if the two features are satisfied, the structural properties still hold. For example, if a scheduling problem is for minimizing the overall expected sum AoI (i.e., the instantaneous reward is $r(\mathbf{s}_t)=\sum_{n=1}^{N}-\tau_{n,t}$), not the sum estimation MSE, then the optimal policy does have the same structural properties. Similarly, considering an AoI violation minimization problem for delay-sensitive applications with an AoI threshold d [127], the instantaneous reward can be defined as $r(\mathbf{s}_t)=\sum_{n=1}^{N}-\mathbb{1}(\tau_{n,t}>d)$, satisfying the two features.*

Remark 2.4.3 (Counterexample)**.** *We can show that if the considered optimal scheduling problem of a multi-sensor remote estimation system has a reward function as the negative product of the individual estimation MSE, i.e., $r(\mathbf{s})=-\prod_{i=1}^{N}\mathrm{Tr}(\mathbf{P}_{i,t})$, which does not satisfy the second feature in Remark 2.4.2, then the threshold structure may not exist. Fig 2.3 illustrates the optimal scheduling policy of a two-sensor-single-channel remote estimation system with the constructed reward function. We see that the optimal policy is not a threshold policy as in Theorem 2.4.2.*

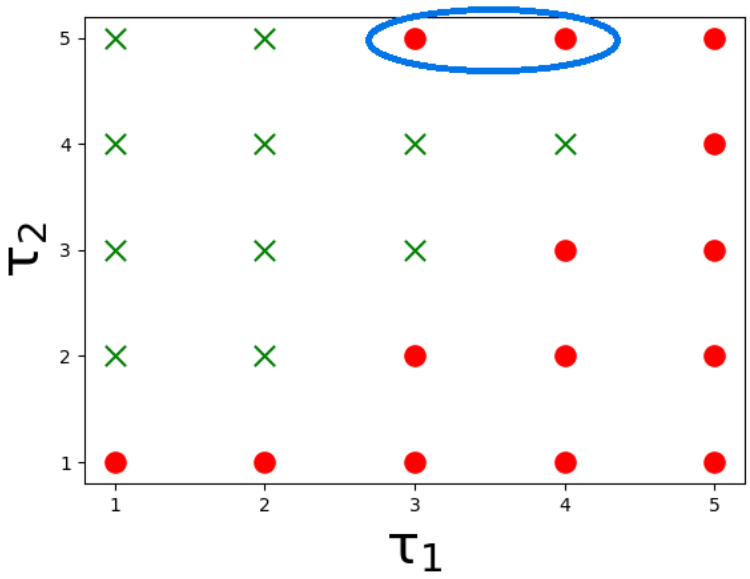


Figure 2.3: The optimal policy of a two-sensor-single-channel scheduling problem with the multiplicative reward function, where • and × represent the schedule of sensor 1 and 2, respectively.

2.5 Structure-Enhanced DRL

In the literature, DQN and DDPG are the most commonly adopted off-policy DRL algorithms for solving optimal scheduling problems (see [78, 82, 128, 129]), and they provide significant performance improvements over heuristic policies. Next, we develop threshold structure-enhanced (SE) DQN and DDPG for solving Problem 2.3.1 based on the structural properties in Definitions 2.3.1 and 2.3.2. The performance improvement of SE algorithms will be presented in Section 2.6.[6]

[6]Note that the DRL training is conducted offline, which requires knowledge of the state-transition rules in Section 2.3.1, building on the channel statistics. Online fine-tuning techniques for model mismatching issues during the deployment stage are beyond the scope of the work.

2.5.1 Structure-Enhanced DQN

Based on (2.4.2) and (2.4.4), we have the Bellman equation in terms of the Q-values of the optimal policy:

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t) + \mathbb{E}_{\mathbf{s}_{t+1}} \left[\gamma \max_{\mathbf{a}_{t+1}} Q\left(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}\right) \right]. \tag{2.5.1}$$

A well-trained DQN uses a neural network (NN) with the parameter set $\boldsymbol{\theta}$ to approximate $Q(\mathbf{s}_t, \mathbf{a}_t)$ by $Q(\mathbf{s}_t, \mathbf{a}_t; \boldsymbol{\theta})$ and use it to find the optimal action, i.e., $\mathbf{a}_t^* = \text{argmax}_{\mathbf{a}_t \in \mathcal{A}} Q(\mathbf{s}_t, \mathbf{a}_t; \boldsymbol{\theta})$. Considering the action space defined in Section 2.3.1, the DQN has $N!/(N-M)!$ Q-value outputs of different state-action pairs. To train the DQN, one needs to sample data (consisting of states, actions, rewards, and next states), define a loss function of $\boldsymbol{\theta}$ based on the collected data, and minimize the loss function to find the optimized $\boldsymbol{\theta}$. However, the conventional DQN training method has never utilized structures of optimal policies before.

To utilize the knowledge of the threshold policy structure for enhancing the DQN training performance, we propose **1) an SE action selection method** based on Definitions 2.3.1 and 2.3.2 to select reasonable actions and hence enhance the data sampling efficiency; and **2) an SE loss function definition** to add the penalty to sampled actions that do not follow the threshold structure.

Our SE-DQN training algorithm has three stages: 1) the DQN with **loose SE action selection** stage, which only utilizes part of the structural property, 2) the DQN with **tight SE action selection** stage utilizes the full structural property, and 3) the conventional DQN stage. The first two stages use the SE action selection schemes and the SE loss function to train the DQN fast, resulting in a reasonable threshold policy, and the last stage is for further policy exploration. In what follows, we present the loose and tight SE action selection schemes and the SE loss function.

Loose SE action selection

We randomly select an action $\mathbf{a}^{\epsilon}$ from the entire action space with a probability of ϵ for action exploration; with a probability of $(1-\epsilon)$, we generate the SE action $\hat{\mathbf{a}}$ as below. For simplicity, we drop the time index when describing action selections.

The threshold structure suggests that the actions of $\mathbf{s}$ and the state with a smaller AoI or channel state are correlated. Thus, one can infer the action based on the action at the state with a smaller channel, or AoI state, based on the channel-state and the AoI-state threshold properties in Definitions 2.3.1 and 2.3.2, respectively. We only consider the AoI-state threshold property for loose SE action selection, as it is difficult to find actions that satisfy both structural properties at the beginning of training. We will utilize both of them in the tight SE action selection stage.

Given the state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, we define the state with a smaller sensor n's AoI as $\dot{\mathbf{s}}^{(n)} = (\dot{\boldsymbol{\tau}}^{(n)}, \mathbf{H})$, where

$$\dot{\boldsymbol{\tau}}^{(n)} = (\tau_1, \ldots, \tau_n - 1, \ldots, \tau_N). \tag{2.5.2}$$

For each n, we calculate the corresponding action based on the Q-values as

$$\dot{\mathbf{a}}^{(n)} \triangleq (\dot{a}_1^{(n)}, \ldots, \dot{a}_n^{(n)}, \ldots, \dot{a}_N^{(n)}) = \arg\max_{\dot{\mathbf{a}}} Q(\dot{\mathbf{s}}^{(n)}, \dot{\mathbf{a}}; \boldsymbol{\theta}). \tag{2.5.3}$$

Recall that $\dot{a}_n^{(n)} \in \{0, 1, \ldots, M\}$ is the channel index assigned to sensor n at the state $\dot{\mathbf{s}}^{(n)}$.

If $\dot{a}_n^{(n)} > 0$, then the AoI-state threshold property implies that channel $\dot{a}_n^{(n)}$ or a better channel is assigned to sensor n at the state $\mathbf{s}$. We define the set of channels with better quality as

$$\mathcal{M}^{(n)} \triangleq \{m' | h_{n,m'} > h_{n,\dot{a}_n^{(n)}}, m' = 1, \ldots, M\}. \tag{2.5.4}$$

Then, the SE action for sensor n, say $\hat{a}_n$, is randomly chosen from the set $\mathcal{M}^{(n)}$ with

probability ξ, and is equal to $\dot{a}_n^{(n)}$ with probability $1-\xi$.

If $\dot{a}_n^{(n)} = 0$, then the AoI-state threshold property cannot help with determining the action. Then, we define the action selected by the greedy policy (i.e., the conventional DQN method) at the state $\mathbf{s}$ as

$$\tilde{\mathbf{a}} \triangleq (\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_N) = \arg\max_{\mathbf{a}} Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}). \tag{2.5.5}$$

Thus, we set the SE action of sensor n identical to the one generated by the conventional DQN method, i.e., $\hat{a}_n = \tilde{a}_n$.

Now we can define the SE action for N sensors as $\hat{\mathbf{a}} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_N)$. If such an action meets the constraint

$$\sum_{m=1}^{M} \mathbb{1}\left(\hat{a}_n = m\right) \leq 1, \quad \sum_{n=1}^{N} \mathbb{1}\left(\hat{a}_n = m\right) \leq 1, \tag{2.5.6}$$

which is less restrictive than (2.2.6), we select the action $\mathbf{a}$ as $\hat{\mathbf{a}}$ and assign the unused channels randomly to unscheduled sensors; otherwise, $\mathbf{a}$ is identical to that of the conventional method as $\tilde{\mathbf{a}}$.

Tight SE action selection

By using the loose SE action selection, we first infer the scheduling action of sensor n at the state $\mathbf{s}$ based on the action of the state with a smaller AoI, $\dot{\mathbf{s}}^{(n)}$. Then, we check whether the loose SE action satisfies the channel-state threshold property as below.

For notation simplicity, we use $m > 0$ to denote the SE channel selection for sensor n. Given the state $\mathbf{s}$, we define the state with a smaller channel m's state for sensor n as $\ddot{\mathbf{s}}^{(n,m)} = (\boldsymbol{\tau}, \ddot{\mathbf{H}}^{(n,m)})$, where $\ddot{\mathbf{H}}^{(n,m)}$ and $\mathbf{H}$ are identical except the element

$\ddot{h}_{n,m}^{(n,m)} = h_{n,m} - 1$. Then, we calculate the corresponding action

$$\ddot{\mathbf{a}}^{(n,m)} \triangleq (\ddot{a}_1^{(n,m)}, \ldots, \ddot{a}_n^{(n,m)}, \ldots, \ddot{a}_N^{(n,m)}) = \arg\max_{\ddot{\mathbf{a}}} Q(\ddot{\mathbf{s}}^{(n,m)}, \ddot{\mathbf{a}}; \boldsymbol{\theta}). \tag{2.5.7}$$

From the channel-state and the AoI-state threshold properties, the scheduling action $\ddot{a}_n^{(n,m)}$ should be identical to m. Thus, if $\ddot{a}_n^{(n,m)} = m$, then the SE action for sensor n is $\hat{a}_n = m$; otherwise, $\hat{a}_n$ is identical to the conventional DQN action. The SE action satisfying both the threshold properties is $\hat{\mathbf{a}} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_N)$. If such an action meets the constraint (2.2.6), then it is executed as $\mathbf{a} = \hat{\mathbf{a}}$; otherwise, we select the greedy action $\mathbf{a} = \tilde{\mathbf{a}}$.

SE loss function

During the training, each transition $(\mathbf{s}_t, \mathbf{a}_t, \hat{\mathbf{a}}_t, \tilde{\mathbf{a}}_t, r_t, \mathbf{s}_{t+1})$ is stored in a replay memory, where $r_t \triangleq r(\mathbf{s}_t)$ denotes the immediate reward. Different from the conventional DQN, we include both the SE action $\hat{\mathbf{a}}$ and the greedy action $\tilde{\mathbf{a}}$, in addition to the executed action $\mathbf{a}$.

Let $\mathcal{T}_i \triangleq (\mathbf{s}_i, \mathbf{a}_i, \hat{\mathbf{a}}_i, \tilde{\mathbf{a}}_t, r_i, \mathbf{s}_{i+1})$ denote the ith transition of a sampled batch from the replay memory. Same as the conventional DQN, we define the temporal-difference (TD) error of $\mathcal{T}_i$ as

$$\mathsf{TD}_i \triangleq y_i - Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\theta}), \tag{2.5.8}$$

where $y_i = r_i + \gamma \max_{\mathbf{a}_{i+1}} Q(\mathbf{s}_{i+1}, \mathbf{a}_{i+1}; \boldsymbol{\theta})$ is the estimation of Q-value at next step. This is to measure the gap between the left and right sides of the Bellman equation (2.5.1). Since the optimal scheduling policy satisfies the Bellman equation [76], i.e., $y_i - Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\theta}) = 0$, the DRL algorithm has to minimize the TD error. Different from DQN, we introduce the action-difference (AD) error as below to measure the difference of Q-value between actions selected by the SE strategy and the greedy

Algorithm 1 SE-DQN for sensor scheduling in the remote estimation system

1: Initialize replay memory $\mathcal{D}$ to capacity N
2: Initialize policy network with random weights $\boldsymbol{\theta}$
3: Initialize target network with weights $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$
4: **for** episode $= 1, 2, \ldots, E_1$ **do**
5: Initialize state $\mathbf{s}_0$
6: **for** $t = 1, 2, \ldots, T$ **do**
7: Generate $\hat{\mathbf{a}}_t$ and select action $\mathbf{a}_t$ by the loose SE action selection method of SE-DQN
8: Execute action $\mathbf{a}_t$ and observe r_t and $\mathbf{s}_{t+1}$
9: Compute action $\tilde{\mathbf{a}}_t = \arg\max_{\mathbf{a}} Q(\mathbf{s}_t, \mathbf{a}; \boldsymbol{\theta})$
10: Store transition $(\mathbf{s}_t, \mathbf{a}_t, \hat{\mathbf{a}}_t, \tilde{\mathbf{a}}_t, r_t, \mathbf{s}_{t+1})$ in $\mathcal{D}$
11: Sample a random batch of transitions $(\mathbf{s}_i, \mathbf{a}_i, \hat{\mathbf{a}}_i, \tilde{\mathbf{a}}_i, r_i, \mathbf{s}_{i+1})$
12: Calculate TD_i and AD_i based on (2.5.8) and (2.5.9)
13: Update $\boldsymbol{\theta}$ according to (2.5.12)
14: Every t' steps set $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_t$
15: **end for**
16: **end for**
17: **for** episode $= E_1, \ldots, E_2$ **do**
18: Repeat algorithm from line 5 to line 15 by adopting the tight SE action selection method
19: **end for**
20: **for** episode $= E_2, \ldots, E$ **do**
21: Execute the DQN algorithm as in [78]
22: **end for**

strategy:

$$\mathsf{AD}_i \triangleq Q(\mathbf{s}_i, \hat{\mathbf{a}}_i; \boldsymbol{\theta}) - Q(\mathbf{s}_i, \tilde{\mathbf{a}}_i; \boldsymbol{\theta}). \tag{2.5.9}$$

Since the optimal policy has the threshold structure, the inferred action $\hat{\mathbf{a}}$ should be identical to the optimal action $\tilde{\mathbf{a}}$. Thus, the DRL algorithm should minimize the AD error (2.5.9).

Based on (2.5.8) and (2.5.9), we define the loss function of $\mathcal{T}_i$ as

$$L_i(\boldsymbol{\theta}) = \begin{cases} \alpha_1 \mathsf{TD}_i^2 + (1-\alpha_1)\mathsf{AD}_i^2, & \text{if } \mathbf{a}_i = \hat{\mathbf{a}}_i \\ \mathsf{TD}_i^2, & \text{otherwise,} \end{cases} \tag{2.5.10}$$

where α_1 is a hyperparameter to balance the importance of TD_i and AD_i. In other

words, if the SE action is executed, both the TD and AD errors are taken into account; otherwise, the conventional TD-error-based loss function is adopted.

Given the batch size B, the overall loss function is

$$L(\boldsymbol{\theta}) = \frac{1}{B}\sum_{i=1}^{B} L_i(\boldsymbol{\theta}). \tag{2.5.11}$$

To optimize $\boldsymbol{\theta}$, we adopt the well-known gradient descent method and calculate the gradient as below

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \frac{1}{B}\sum_{i=1}^{B} \nabla_{\boldsymbol{\theta}} L_i(\boldsymbol{\theta}), \tag{2.5.12}$$

where $\nabla_{\boldsymbol{\theta}} L_i(\boldsymbol{\theta})$ is given as

$$\nabla_{\boldsymbol{\theta}} L_i(\boldsymbol{\theta}) = \begin{cases} -2((1-\alpha_1)\mathsf{AD}_i \nabla_{\boldsymbol{\theta}} \left(Q\left(\mathbf{s}_i, \hat{\mathbf{a}}_i; \boldsymbol{\theta}\right) - Q(\mathbf{s}_i, \tilde{\mathbf{a}}_i; \boldsymbol{\theta})\right) \\ \quad +\alpha_1 \mathsf{TD}_i \nabla_{\boldsymbol{\theta}} Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\theta})), & \text{if } \mathbf{a}_i = \hat{\mathbf{a}}_i \\ -2(\mathsf{TD}_i \nabla_{\boldsymbol{\theta}} Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\theta})), & \text{otherwise.} \end{cases} \tag{2.5.13}$$

The details of the SE-DQN algorithm are given in Algorithm 1.

2.5.2 Structure-Enhanced DDPG

Different from the DQN, which has one NN to estimate the Q-value, a DDPG agent has two NNs [56], a critic NN with parameter $\boldsymbol{\theta}$ and an actor NN with the parameter $\boldsymbol{\mu}$. In particular, the actor NN approximates the optimal policy $\pi^*(\mathbf{s})$ by $\mu(\mathbf{s}; \boldsymbol{\mu})$, while the critic NN approximates the Q-value of the optimal policy $Q(\mathbf{s}, \mathbf{a})$ by $Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta})$. In general, the critic NN judges whether the generated action of the actor NN is good or not, and the latter can be improved based on the former. The critic NN is updated by minimizing the TD error similar to the DQN. The actor-critic framework enables DDPG to solve MDPs with continuous and large action space, which cannot be handled by the DQN.

To solve our scheduling problem with a discrete action space, we adopt an action

mapping scheme similar to the one adopted in [82]. We set the direct output of the actor NN with N continuous values, ranging from -1 to 1, corresponding to sensors 1 to N, respectively. Recall that the DQN has $N!/(N-M)!$ outputs. The N values are sorted in descending order. The sensors with the highest M ranking are assigned to channels 1 to M, respectively. The corresponding ranking values are then linearly normalized to $[-1, 1]$ as the output of the final outputs of the actor NN, named as the virtual action $\mathbf{v}$. It directly follows that the virtual action $\mathbf{v}$ and the real scheduling action $\mathbf{a}$ can be mapped from one to the other directly. Therefore, we use the virtual action $\mathbf{v}$, instead of the real action $\mathbf{a}$, when presenting the SE-DDPG algorithm.

Similar to the SE-DQN, the SE-DDPG has the loose SE-DDPG stage, the tight SE-DDPG stage, and the conventional DDPG stage. The ith sampled transition is denoted as

$$\mathcal{T}_i \triangleq (\mathbf{s}_i, \mathbf{v}_i, \hat{\mathbf{v}}_i, \tilde{\mathbf{v}}_t, r_i, \mathbf{s}_{i+1}). \tag{2.5.14}$$

We present the SE action selection method and the SE loss function in the sequel.

SE action selection

The general action selection approach for the SE-DDPG is identical to that of the SE-DQN, by simply converting $\mathbf{v}_i$, $\mathbf{v}_i^\epsilon$, $\tilde{\mathbf{v}}_i$, and $\hat{\mathbf{v}}_i$ to $\mathbf{a}_i$, $\mathbf{a}_i^\epsilon$, $\tilde{\mathbf{a}}_i$, and $\hat{\mathbf{a}}_i$, respectively. Different from DQN with ϵ-greedy actions, the action generated by the DDPG based on the current state is

$$\tilde{\mathbf{v}}_i = \mu(\mathbf{s}_i; \boldsymbol{\mu}), \tag{2.5.15}$$

and the random action $\mathbf{v}_i^\epsilon$ was generated by adding noise to the original continuous output of the actor NN.

SE loss function

Different from the SE-DQN, the SE-DDPG needs different loss functions for updating the critic NN and the actor NN. For the critic NN, we use the same loss function as in (2.5.11), and thus the gradient for the critic NN update is identical to (2.5.12). Note that for DDPG, the next virtual action $\tilde{\mathbf{v}}_{i+1}$ is the direct output of the actor NN given the next state $\mathbf{s}_{i+1}$, i.e., $\tilde{\mathbf{v}}_{i+1} = \mu(\mathbf{s}_{i+1}; \boldsymbol{\mu})$. Thus, when calculating the TD error (2.5.8), we have $y_i = r_i + \gamma Q(\mathbf{s}_{i+1}, \mu(\mathbf{s}_{i+1}; \boldsymbol{\mu}); \boldsymbol{\theta})$.

For the actor NN, we introduce the difference between actions selected by the SE strategy $\hat{\mathbf{v}}_i$ and the actor NN $\tilde{\mathbf{v}}_i$, when $\hat{\mathbf{v}}_i$ is executed, i.e., $\mathbf{v}_i = \hat{\mathbf{v}}_i$. If the SE action is not selected, then the loss function is the Q-value given the state-action pair, which is identical to the conventional DDPG. Given the hyperparameter α_2, the loss function for the transition $\mathcal{T}_i$ is defined as

$$L_i(\boldsymbol{\mu}) = \begin{cases} -\alpha_2 Q(\mathbf{s}_i, \tilde{\mathbf{v}}_i; \boldsymbol{\theta}) + (1-\alpha_2)\left(\mathbf{v}_i - \tilde{\mathbf{v}}_i\right)^2, & \text{if } \mathbf{v}_i = \hat{\mathbf{v}}_i \\ -Q(\mathbf{s}_i, \tilde{\mathbf{v}}_i; \boldsymbol{\theta}), & \text{otherwise,} \end{cases} \tag{2.5.17}$$

and hence the overall loss function given the sampled batch is

$$L(\boldsymbol{\mu}) = \frac{1}{B}\sum_{i=1}^{B} L_i(\boldsymbol{\mu}). \tag{2.5.18}$$

By replacing (2.5.15) and (2.5.17) in (2.5.18) and applying the chain rule, we can derive the gradient of the overall loss function in terms of $\boldsymbol{\mu}$ as

$$\nabla_{\boldsymbol{\mu}} L(\boldsymbol{\mu}) = \frac{1}{B}\sum_{i=1}^{B} \nabla_{\boldsymbol{\mu}} L_i(\boldsymbol{\mu}), \tag{2.5.19}$$

where $\nabla_{\boldsymbol{\mu}} L_i(\boldsymbol{\mu})$ is given by:

$$\nabla_{\boldsymbol{\mu}} L_i(\boldsymbol{\mu}) = \begin{cases} -\alpha_2 \nabla_{\tilde{\mathbf{v}}_i} Q(\mathbf{s}_i, \tilde{\mathbf{v}}_i; \boldsymbol{\theta}) \nabla_{\boldsymbol{\mu}} \mu(\mathbf{s}_i; \boldsymbol{\mu}) - 2(1-\alpha_2) \\ \quad \times \left(\mathbf{v}_i - \mu\left(\mathbf{s}_i; \boldsymbol{\mu}\right)\right) \nabla_{\boldsymbol{\mu}} \mu(\mathbf{s}_i; \boldsymbol{\mu}), & \text{if } \mathbf{v}_i = \hat{\mathbf{v}}_i \\ -\nabla_{\tilde{\mathbf{v}}_i} Q(\mathbf{s}_i, \tilde{\mathbf{v}}_i; \boldsymbol{\theta}) \nabla_{\boldsymbol{\mu}} \mu(\mathbf{s}_i; \boldsymbol{\mu}), & \text{otherwise,} \end{cases} \tag{2.5.20}$$

where the action $\tilde{\mathbf{v}}_i$ is replaced by $\mu(\mathbf{s}_i; \boldsymbol{\mu})$ based on (2.5.15). The details of the

Algorithm 2 SE-DDPG for sensor scheduling in the remote estimation system

1: Initialize replay memory $\mathcal{D}$ to capacity N
2: Initialize actor network and critic network with random weights $\boldsymbol{\mu}$ and $\boldsymbol{\theta}$
3: Initialize target network and with weight $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$, $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$
4: **for** episode $= 1, 2, \ldots, E_1$ **do**
5: Initialize a random noise $\mathcal{N}$ for action exploration
6: Initialize state $\mathbf{s}_0$
7: **for** $t = 1, 2, \ldots, T$ **do**
8: Generate $\hat{\mathbf{v}}_t$ and select action $\mathbf{v}_t$ by the loose SE action selection method of SE-DDPG
9: Mapping virtual action $\mathbf{v}_t$ to real action $\mathbf{a}_t$
10: Execute action $\mathbf{a}_t$ and observe r_t and $\mathbf{s}_{t+1}$
11: Compute action $\tilde{\mathbf{v}}_t = \mu(\mathbf{s}_t; \boldsymbol{\mu})$
12: Store transition $(\mathbf{s}_t, \mathbf{v}_t, \hat{\mathbf{v}}_t, \tilde{\mathbf{v}}_t, r_t, \mathbf{s}_{t+1})$ in $\mathcal{D}$
13: Sample a random batch of transitions $(\mathbf{s}_i, \mathbf{v}_i, \hat{\mathbf{v}}_i, \tilde{\mathbf{v}}_i, r_i, \mathbf{s}_{i+1})$
14: Calculate TD_i and AD_i based on (2.5.8) and (2.5.9) but with the virtual actions $\mathbf{v}_i, \hat{\mathbf{v}}_i, \tilde{\mathbf{v}}_i$
15: Update $\boldsymbol{\theta}$ and $\boldsymbol{\mu}$ according to (2.5.12) and (2.5.19), respectively
16: Update the target network:

$$\hat{\boldsymbol{\mu}} \leftarrow \delta\boldsymbol{\mu} + (1-\delta)\hat{\boldsymbol{\mu}}, \quad \hat{\boldsymbol{\theta}} \leftarrow \delta\boldsymbol{\theta} + (1-\delta)\hat{\boldsymbol{\theta}} \tag{2.5.16}$$

17: **end for**
18: **end for**
19: **for** episode $E_1 = 1, 2, \ldots, E_2$ **do**
20: Repeat algorithm from line 5 to line 18 by adopting the tight SE action selection method
21: **end for**
22: **for** episode $E_2 = 1, 2, \ldots, E$ **do**
23: Execute the conventional DDPG algorithm as in [56]
24: **end for**

proposed SE-DDPG algorithm are given in Algorithm 2.

2.6 Numerical Experiments

In this section, we evaluate and compare the performance of the proposed SE-DQN and SE-DDPG with the benchmark DQN and DDPG.

Table 2.1: Summary of Hyperparameters

Hyperparameters of SE-DQN and SE-DDPG	Value
Initial values of ϵ and ξ	1
Decay rates of ϵ and ξ	0.999
Minimum ϵ and ξ	0.01
Mini-batch size, B	128
Experience replay memory size, K	20000
Discount factor, γ	0.95
Hyperparameters of SE-DQN	
Learning rate	0.0001
Decay rate of learning rate	0.001
Target network update frequency	100
Weight of SE-DQN loss function, α_1	0.5
Input dimension of network	$N + N \times M$
Output dimension of network	$N!/(N-M)!$
Hyperparameters of SE-DDPG	
Learning rate of actor network	0.0001
Learning rate of critic network	0.001
Decay rate of learning rate	0.001
Soft parameter for target update, δ	0.005
Weight of critic network loss function, α_1	0.5
Weight of actor network loss function, α_2	0.9
Input dimension of actor network	$N + N \times M$
Output dimension of actor network	N
Input dimension of critic network	$2N + N \times M$
Output dimension of critic network	1

2.6.1 Experiment Setups

Our numerical experiments are simulated in Python and run on a computing platform with an Intel Core i5 9400F CPU @ 2.9 GHz with 16GB RAM and an NVIDIA RTX 2070 GPU. For the remote estimation system, we set the dimensions of the process state and the measurement as $l_n = 2$ and $e_n = 1$, respectively. The system matrices $\{\mathbf{A}_n\}$ are randomly generated with the spectral radius within the range of $(1, 1.4)$. The entries of $\mathbf{C}_n$ are drawn uniformly from the range $(0, 1)$, $\mathbf{W}_n$ and $\mathbf{V}_n$ are identity matrices.

Since the SE-DQN and SE-DDPG require the discrete channel state space, the

fading channel state is quantized into $\bar{h} = 5$ levels, and the corresponding packet drop probabilities are $0.2, 0.15, 0.1, 0.05$, and 0.01. The distribution of the discrete channel states of each sensor-channel pair is generated based on Rayleigh distribution with the random scale parameter sampled uniformly from the range of $(0.5, 2)$ [130].

The NNs of the DQN and SE-DQN have the same hyperparameters, and each has two hidden layers with 256 nodes. The NNs of the DDPG and SE-DDPG are identical, and each of the actor and critic NNs has two hidden layers with 512 nodes. All the NNs in these algorithms are fully connected neural network (FCNN). The computation complexity of an FCNN is $O\left(\sum_{l=1}^{L-1} u_l u_{l+1}\right)$, where u_l is the number of nodes in layer l and L is the total number of layers of the NN [131]. Based on our computing platform, the inference time of the SE-DQN and SE-DDPG are 0.02 ms and 0.04 ms, respectively. The settings of the other hyperparameters for Algorithm 1 and Algorithm 2 are summarized in Table 2.1. Note that the input dimensions of the actor NN and the critic NN are different, as the inputs of the critic NN and the actor NN are the state vector and the state-action pair, respectively. During the training, we use the ADAM optimizer for calculating the gradient and reset the environment for each episode with $T = 500$ steps. The episode numbers for the loose SE action, the tight SE action, and the conventional DQN stages are 50, 100, and 150, respectively.

2.6.2 Performance Comparison

Fig. 2.4 and Fig. 2.5 illustrate the average sum MSE of all processes during the training achieved by the SE-DRL algorithms and the benchmarks under some system settings. Both of these two figures illustrate that the DRL methods outperform the conventional heuristic benchmark scheduling methods, including round robin and

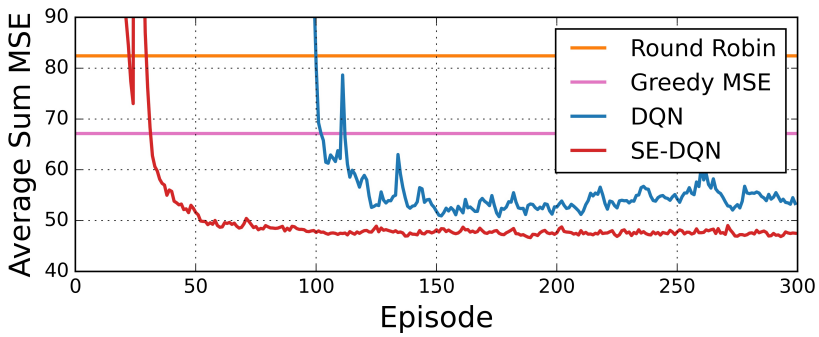


Figure 2.4: Average sum MSE of all processes during training with $N=6, M=3$.

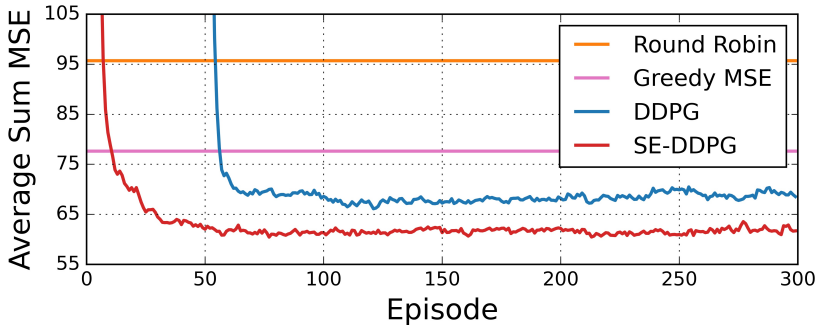


Figure 2.5: Average sum MSE of all processes during training with $N=10, M=5$.

greedy MSE. Fig. 2.4 shows that the SE-DQN saves about 50% training episodes for the convergence, and also decreases the average sum MSE for 10% than the conventional DQN. Fig. 2.5 shows that the SE-DDPG saves about 30% training episodes and reduces the average sum MSE by 10%, when compared to the conventional DDPG. Also, we see that the conventional DQN and DDPG stages (i.e. the last 150 episodes) in Fig 2.4 and 2.5 cannot improve much of the training performance. This implies that the SE stages have converged.

In Table 2.2, we test the performance of well-trained SE-DRL algorithms for different numbers of sensors and channels, and different settings of system parameters, i.e., $\mathbf{A}_n$, $\mathbf{C}_n$, and $q_1^{(n,m)}, \ldots, q_{\bar{h}}^{(n,m)}$, based on 10000-step simulations. We see that for 6-sensor-3-channel systems, the SE-DQN reduces the average MSE by 10% to 25%

Table 2.2: Performance Comparison of the SE-DRL and the Benchmarks in terms of the Average Estimation MSE

System Scale (N, M)	Param. setup	DQN	**SE-DQN**	DDPG	**SE-DDPG**
$(6, 3)$	1	52.4121	**47.6766**	48.4075	**47.0594**
	2	67.4247	**49.8476**	53.3675	**44.7423**
	3	84.1721	**59.9127**	56.9504	**55.2409**
	4	79.5902	**65.1640**	64.4313	**58.5534**
$(6, 2)$	5	55.7092	**50.5983**	51.3488	**47.1210**
	6	–	**80.2522**	77.4124	**72.4863**
	7	–	**78.7715**	85.3182	**75.8465**
	8	–	**58.6024**	61.4121	**57.8539**
$(10, 5)$	9	–	–	68.0247	**62.4727**
	10	–	–	89.6290	**78.4138**
	11	–	–	90.2812	**81.1148**
	12	–	–	–	**147.2844**
$(20, 10)$	13	–	–	181.4135	**159.4321**
	14	–	–	163.1257	**142.1850**
	15	–	–	173.0940	**144.8166**
	16	–	–	215.2231	**160.2304**

over DQN, while both DDPG and SE-DDPG achieve similar performance as the SE-DQN. This suggests that the SE-DQN is almost optimal and that the DDPG and the SE-DDPG cannot improve further. In 6-sensor-2-channel systems, the DQN cannot converge in most experiments but the SE-DQN and SE-DDPG can still perform better than DDPG. In 10-sensor-5-channel systems, neither the DQN nor the SE-DQN can converge, and the SE-DDPG achieves a 10% MSE reduction over the DDPG. In particular, the SE-DDPG is the only converged algorithm in Experiment 12. In 20-sensor-10-channel systems, we see that the SE-DDPG can reduce the average MSE by 15% to 25%. Therefore, the performance improvement of the SE-DDPG appears significant for large systems.

In addition, to demonstrate that the SE-DRL algorithms can also be applied to other scenarios, we consider a multi-sensor scheduling problem for minimizing the

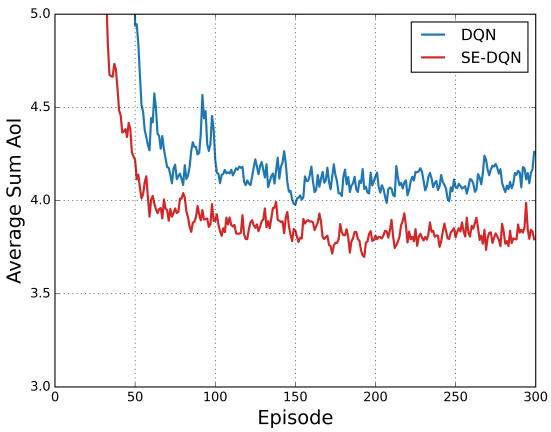
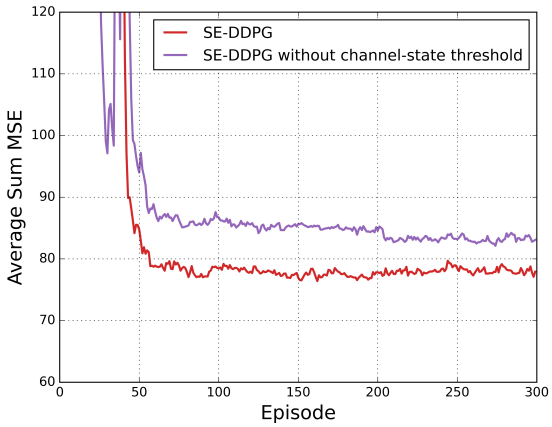


Figure 2.6: Average sum AoI of all processes during the training with $N = 6, M = 3$.

overall expected sum AoI as mentioned in Remark 2.4.3. In Fig 2.6, we see that the SE-DQN can effectively reduce the average sum AoI by 10% compared with the DQN.

We also test the effectiveness of the channel-state threshold property and the loose SE action selection stage in the SE-DRL algorithms. Fig. 2.7(a) illustrates that without using the channel-threshold property in the SE-action selection process, although the convergence speed is not affected much, such a training method does not converge to a near-optimal policy and leads to a noticeably higher estimation MSE than the original SE-DDPG. In Fig. 2.7(b), we see that if one disables the loose SE-action selection stage, the training convergence time doubles compared with the original SE-DDPG, though both converged algorithms provide similar performance. Thus, both the channel-threshold-enabled action selection and the loose action selection stage are critical to guarantee the performance of the SE-DRL algorithms.



(a) SE-DDPG with and without using the channel-state threshold property

(b) SE-DDPG with and without the loose SE-action selection stage

Figure 2.7: Training performance comparison between SE-DDPG with different training settings, where $N = 10, M = 5$.

2.7 Conclusion

In this chapter, we have proved that the optimal scheduling policy of the remote estimation system satisfies a number of threshold properties. Then, based on these theoretical guidelines, we have developed the structure-enhanced deep reinforcement learning (SE-DRL) algorithms for solving the optimal scheduling problem. In particular, we have proposed a novel action selection method and a new loss function to improve training efficiency. Our numerical results have illustrated that SE-DRL algorithms can save training time by 50% and reduce the estimation mean-square error (MSE) by 10% to 25%, when compared with benchmark DRL algorithms. In addition, the results also show that SE-DRL can effectively solve a range of optimal scheduling problems, not limited to remote state estimation systems.

Chapter 3

Semantic-aware Transmission Scheduling: a Monotonicity-driven Deep Reinforcement Learning Approach

3.1 Introduction

Different from the first generation of the 5G communications, which focused on improving data-oriented performance (e.g., data rate, latency and reliability), the 6G is envisioned to support many emerging applications and adopt semantic-related metrics for application-level performance optimization [67, 71, 132]. The potential applications supported by 6G semantic communications can be divided into two categories: human-centered systems (e.g., Metaverse and XR [133]) and cyber-physical systems (CPSs) (e.g., autonomous driving and networked robotics [60, 134]). For the former, semantics is a measure of the meaning of information that a human can understand; for the latter, semantics characterizes the usefulness of messages with respect to the goal of data exchange.

In CPSs, semantic-aware transmission aims to minimize the distortion and timing

mismatch of the data between the edge devices and the remote destination [60]. Thus, the freshness of messages is critical, and monotonic functions of age of information (AoI) are often introduced as semantic quality metrics [65, 135]. Semantic-aware (and AoI-aware) dynamic scheduling of cyber-physical systems over limited communication resources has attracted much attention in the communications society [136]. In [16], an optimal AoI minimization scheduling problem was considered under an average transmission power constraint and was formulated into a constrained MDP. Note that the conventional Markov decision process (MDP) algorithms can only solve very small-scale problems due to the high computational complexity introduced by the curse of dimensionality. Heuristic scheduling policies without optimality guarantees were proposed in were proposed in [137, 138].

The recent development of deep reinforcement learning (DRL) allows us to solve large MDPs effectively with deep neural networks (NNs) for function approximations. However, a major drawback of applying existing DRL methods in large-scale scheduling problems is that training is usually very time-consuming and can often get stuck in local minima. Therefore, in this chapter, we first prove the monotonicity of the Q-value function w.r.t the AoI states and channel states, and then develop a monotonic critic NN for the optimal semantic-aware scheduling.

3.2 System Model

We consider semantic communication in a CPS with N edge devices (e.g., sensors and robots) and a remote destination (e.g., a remote estimator or controller) connected by a wireless network with $M < N$ channels (e.g., sub-carriers).

At each time step t, the transmission scheduling action for device n is denoted by

$$a_{n,t} = \begin{cases} 0 & \text{if device } n \text{ is not scheduled,} \\ m & \text{if device } n \text{ is scheduled to channel } m. \end{cases} \tag{3.2.1}$$

We assume that each device can be scheduled to at most one channel and that each channel is assigned to one device, i.e.,

$$\sum_{m=1}^{M} \mathbb{1}\left(a_{n,t} = m\right) \leq 1, \quad \sum_{n=1}^{N} \mathbb{1}\left(a_{n,t} = m\right) = 1, \tag{3.2.2}$$

where $\mathbb{1}(\cdot)$ is the indicator function.

We consider independent and identically distributed (i.i.d.) block fading channels, where the channel quality is fixed during each packet transmission and varies packet by packet independently. The channel state of the system at time t is denoted by a $N \times M$ matrix $\mathbf{H}_t$, where the nth row and mth column element $h_{n,m,t} \in \mathcal{H} \triangleq \left\{1, 2, \ldots, \bar{h}\right\}$ represents the channel state between device n and the destination at channel m quantized into $\bar{h}$ levels. The distribution of the channel state $h_{n,m,t}$ is given as

$$\Pr(h_{n,m,t} = i) = q_i^{(n,m)}, \forall t, \tag{3.2.3}$$

where $\sum_{i=1}^{\bar{h}} q_i^{(n,m)} = 1, \forall n, m$. A higher channel state leads to a large packet drop probability. Let $p_{n,m,t}$ denote the packet drop probability at the channel state $h_{n,m,t}$. The instantaneous full channel state $\mathbf{H}_t$ is available at the remote estimator based on standard channel estimation methods and all network nodes are in perfect synchronization.

3.2.1 Semantic Communication Performance

First, we define $\tau_{n,t} \in \{1, 2, \dots\}$ as the AoI of device n at time t, i.e., the time elapsed since its latest device packet was successfully received [16, 66, 137]:

$$\tau_{n,t+1} = \begin{cases} 1, & \text{at time } t, \text{ device } n\text{'s packet is received,} \\ \tau_{n,t}+1, & \text{otherwise.} \end{cases} \tag{3.2.4}$$

Next, we define a *positive non-decreasing cost* (or penalty) function $g_n(\tau_{n,t}), \forall n \in \{1, \dots, N\}$ as a semantic measurement of device n's information. A higher AoI state corresponds to an increased cost function, indicating poorer performance in semantic communications. Semantic communication systems with varying functionalities possess distinct $g_n(\cdot)$ functions. The scheduling algorithms proposed in the rest of this section are general and applicable to any positive and non-decreasing cost function.

The remote state estimation system in section 2.2 is an example that shows how to determine $g_n(\cdot)$ in a specific semantic communication system.

3.3 Semantic-aware Transmission Scheduling

We aim to find a dynamic scheduling policy $\pi(\cdot)$ that minimizes the expected total discounted cost function of all N devices over the infinite time horizon.

Problem 3.3.1.

$$\min_{\pi} \lim_{T\to\infty} \mathbb{E}^{\pi} \left[\sum_{t=1}^{T} \sum_{n=1}^{N} \gamma^t g_n(\tau_{n,t}) \right], \tag{3.3.1}$$

where $\gamma \in (0, 1)$ *is a discount factor.*

3.3.1 Markov Decision Process Formulation

Problem 3.3.1 is a sequential decision-making problem with the Markovian property and hence can be cast as an MDP as below.

The state, action, and transition probability are the same as the MDP formulation in section 2.3.1. The immediate reward (i.e., the negative sum cost of all edge devices) at time t is defined as $-\sum_{n=1}^{N} g_n(\tau_{n,t})$.

3.3.2 Value Functions of the Optimal Policy

Similar to section 2.4, we define the optimal value function and Q-value function as

$$V(\mathbf{s}_t) = \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r\left(\mathbf{s}_{t'}\right) \middle| \mathbf{a}_{t'} = \pi^*(\mathbf{s}_{t'}), \forall t' \geq t\right], \tag{3.3.2}$$

and

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(\mathbf{s}_{t'}) \middle| \mathbf{a}_t, \mathbf{a}_{t'} = \pi^*(\mathbf{s}_{t'}), \forall t' > t\right], \tag{3.3.3}$$

respectively. Thus, the V function measures the long-term performance given the current state, while the Q function can be used to compare the long-term performance affected by different current actions.

The relation between the two value functions is given by [76]

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t) + \gamma \sum_{\mathbf{s}_{t+1}} \Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) V(\mathbf{s}_{t+1}). \tag{3.3.4}$$

In particular, the Q function satisfies the Bellman equation:

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t) + \mathbb{E}_{\mathbf{s}_{t+1}}\left[\gamma \max_{\mathbf{a}_{t+1}} Q\left(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}\right)\right]. \tag{3.3.5}$$

3.3.3 DDPG for scheduling policy optimization

A DDPG agent has two neural networks (NNs): an actor NN with the parameter set $\boldsymbol{\alpha}$ and a critic network with the parameter set $\boldsymbol{\beta}$. In particular, the actor NN aims to approximate the optimal policy $\pi^*(\mathbf{s})$ by $\pi(\mathbf{s};\boldsymbol{\alpha})$ after well-training, while the critic NN approximates $Q(\mathbf{s}_t,\mathbf{a}_t)$ by $Q(\mathbf{s}_t,\mathbf{a}_t;\boldsymbol{\beta})$. The overall training process for the two NNs works iteratively based on sampled data named *transitions*. Each transition $\mathcal{T}_i \triangleq (\mathbf{s}_i,\mathbf{a}_i,r_i,\mathbf{s}_{i+1})$, consists of the state, action, reward, and next state. To train the actor NN, the critic NN's Q values are utilized for characterizing the performance of the actor NN's policy over the sampled data; to train the critic NN, the transitions generated by the actor NN's policy are utilized to evaluate the accuracy of Q function approximation. In particular, the approximation error given $\mathcal{T}_i$ is defined as the temporal difference (TD) error

$$\mathsf{TD}_i^2 \triangleq \left(y_i - Q(\mathbf{s}_i,\mathbf{a}_i;\boldsymbol{\beta})\right)^2, \tag{3.3.6}$$

where $y_i = r_i + \gamma \max_{\mathbf{a}_{i+1}} Q(\mathbf{s}_{i+1},\mathbf{a}_{i+1};\boldsymbol{\beta}^-)$ is the estimation of the Q function at next step and $\boldsymbol{\beta}^-$ is the critic NN parameters from the previous iteration. In other words, the training of the critic NN is to minimize the TD errors over the transitions, making the approximated Q function satisfy the Bellman equation (3.3.5) [76].

Remark 3.3.1. *Our work focuses on the effective training of the critic NN by utilizing the partial monotonicity of the Q function discussed in the section below, and the training method for the actor NN is identical to the baseline DDPG. Furthermore, the proposed training methods can be applied to solve other DRL problems that involve monotonic critic NNs and can seamlessly integrate with actor-critic DRL algorithms like Twin Delayed DDPG (TD3) and Proximal Policy Optimization (PPO).*

3.4 Partial Monotonicity of Q Function

We establish the partial monotonicity of the Q function in terms of the AoI and channel states as below.

Theorem 3.4.1 (Monotonicity of Q function w.r.t. AoI states)**.** *Consider action* $\mathbf{a} = (a_1, \ldots, a_N)$*, and states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}'_{AoI} = (\boldsymbol{\tau}'_{(n)}, \mathbf{H})$*, where* $\boldsymbol{\tau}'_{(n)} = (\tau_1, \ldots, \tau'_n, \ldots, \tau_N)$*,* $\tau'_n \geq \tau_n$*. The following holds*

$$Q(\mathbf{s}'_{AoI}, \mathbf{a}) \leq Q(\mathbf{s}, \mathbf{a}). \tag{3.4.1}$$

Proof. For notation simplicity, we assume that $a_n = m, \forall m \in \{1, \ldots, M\}$. From (3.2.3) and (3.2.4), we have

$$\Pr(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H},\mathbf{a}) = \prod_{n=1}^{N} \Pr(\tau_n^+|\tau_n, \mathbf{h}_n, a_n) = \Pr(\tau_n^+|\tau_n,\mathbf{h}_n,a_n)\Pr(\boldsymbol{\tau}_{\backslash n}^+|\boldsymbol{\tau}_{\backslash n},\mathbf{H}_{\backslash n},\mathbf{a}_{\backslash n}), \tag{3.4.2}$$

where $\boldsymbol{\tau}_{\backslash n} = (\tau_1, \ldots, \tau_{n-1}, \tau_{n+1}, \ldots, \tau_N)$ and $\mathbf{a}_{\backslash n} = (a_1, \ldots, a_{n-1}, a_{n+1}, \ldots, a_N)$ represent the AoI states and the actions of all devices except device n, respectively; $\mathbf{h}_n = (h_{n,1}, h_{n,2}, \ldots, h_{n,M})$ is the vector channel state between device n and the remote estimator and $\mathbf{H}_{\backslash n} = (\mathbf{h}_1, \ldots, \mathbf{h}_{n-1}, \mathbf{h}_{n+1}, \ldots, \mathbf{h}_N)$. By using (2.3.2), (3.3.4), and (3.4.2), it can be derived that

$$Q(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}_{\backslash n}^+} \sum_{\tau_n^+} \Pr(\mathbf{H}^+)\Pr(\boldsymbol{\tau}_{\backslash n}^+|\boldsymbol{\tau}_{\backslash n}, \mathbf{H}_{\backslash n}, \mathbf{a}_{\backslash n}) \Pr(\tau_n^+|\tau_n, \mathbf{h}_n, a_n) V(\mathbf{s}^+). \tag{3.4.3}$$

Since the state $\mathbf{s}'_{\mathrm{AoI}}$ is identical to the state $\mathbf{s}$ except the device n's AoI state τ_n, to prove (3.4.1) from (3.4.3) and $r(\mathbf{s}'_{\mathrm{AoI}}) \leq r(\mathbf{s})$, we only need to prove the following inequality

$$\sum_{{\tau'_n}^+} \Pr({\tau'_n}^+|\tau_n, \mathbf{h}_n, a_n) V({\mathbf{s}'_{\mathrm{AoI}}}^+) \leq \sum_{{\tau_n}^+} \Pr({\tau_n}^+|\tau_n, \mathbf{h}_n, a_n) V(\mathbf{s}^+). \tag{3.4.4}$$

From (3.2.4), each of $\mathbf{s}^+$ and ${\mathbf{s}'_{\text{AoI}}}^+$ has two different states in (3.4.4) depending on the different transition of device n's AoI state. Then, by taking (2.3.3) into (3.4.4), the latter is equivalent to

$$\begin{aligned}&(1-p_{n,m})V(1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)+p_{n,m}V(\tau'_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)\\&\leq(1-p_{n,m})V(1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)+p_{n,m}V(\tau_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+).\end{aligned}\tag{3.4.5}$$

By using Lemma 2.4.2, we have $V(\tau'_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)\leq V(\tau_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)$. The inequality (3.4.5) holds directly. □

Theorem 3.4.2 (Monotonicity of Q function w.r.t. channel states)**.** *Consider action* $\mathbf{a}=(a_1,\ldots,a_N)$*, and states* $\mathbf{s}=(\boldsymbol{\tau},\mathbf{H})$ *and* $\mathbf{s}'_{Ch}=(\boldsymbol{\tau},\mathbf{H}'_{n,m})$ *where* $\mathbf{H}'_{n,m}$ *is identical to* $\mathbf{H}$ *except the device-n-channel-m state with* $h'_{n,m}>h_{n,m}$*.*

(i) If $a_n=m$*, then the following holds*

$$Q(\mathbf{s}'_{Ch},\mathbf{a})\leq Q(\mathbf{s},\mathbf{a}).\tag{3.4.6}$$

(ii) If $a_n\neq m$*, then the equality in* (3.4.6) *holds, i.e., we have*

$$Q(\mathbf{s}'_{Ch},\mathbf{a})=Q(\mathbf{s},\mathbf{a}).\tag{3.4.7}$$

Proof. (i) Based on (3.4.3), we only need to prove that

$$\sum_{\mathbf{H'}^+}\sum_{{\tau_n}^+}\Pr(\mathbf{H'}^+)\Pr({\tau_n}^+|\tau_n,\mathbf{h}'_n,a_n)V({\mathbf{s}'_{\text{Ch}}}^+)\leq\sum_{\mathbf{H}^+}\sum_{{\tau_n}^+}\Pr(\mathbf{H}^+)\Pr({\tau_n}^+|\tau_n,\mathbf{h}_n,a_n)V(\mathbf{s}^+),\tag{3.4.8}$$

which is equivalent to

$$\begin{aligned}&(1-p'_{n,m})\sum_{\mathbf{H'}^+}\Pr(\mathbf{H'}^+)V(1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H'}^+)+p'_{n,m}\sum_{\mathbf{H'}^+}\Pr(\mathbf{H'}^+)V(\tau_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H'}^+)\\&\leq(1-p_{n,m})\sum_{\mathbf{H}^+}\Pr(\mathbf{H}^+)V(1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+)+p_{n,m}\sum_{\mathbf{H}^+}\Pr(\mathbf{H}^+)V(\tau_n+1,\boldsymbol{\tau}^+_{\backslash n},\mathbf{H}^+).\end{aligned}\tag{3.4.9}$$

From (3.2.3), we have

$$\sum_{\mathbf{H}'^{+}} \Pr(\mathbf{H}'^{+}) V(\boldsymbol{\tau}^{+}, \mathbf{H}'^{+}) = \sum_{\mathbf{H}^{+}} \Pr(\mathbf{H}^{+}) V(\boldsymbol{\tau}^{+}, \mathbf{H}^{+}), \tag{3.4.10}$$

where $\boldsymbol{\tau}^{+} = (1, \boldsymbol{\tau}^{+}_{\backslash n})$ or $(\tau_n + 1, \boldsymbol{\tau}^{+}_{\backslash n})$. By using (3.4.10), the inequality (3.4.9) is simplified as

$$(p'_{n,m} - p_{n,m}) \sum_{\mathbf{H}^{+}} \Pr(\mathbf{H}^{+}) \Big[V(\tau_n + 1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+}) - V(1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+}) \Big] \leq 0. \tag{3.4.11}$$

Due to the facts that $V(\tau_n + 1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+}) \leq V(1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+})$ and $p'_{n,m} \geq p_{n,m}$, (3.4.11) holds directly.

(ii) Let $a_n = \bar{m} \neq m$. Similar to the proof of (i), after simplifications, we only need to prove the following equality

$$(p_{n,\bar{m}} - p_{n,\bar{m}}) \sum_{\mathbf{H}^{+}} \Pr(\mathbf{H}^{+}) \Big[V(\tau_n + 1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+}) - V(1, \boldsymbol{\tau}^{+}_{\backslash n}, \mathbf{H}^{+}) \Big] = 0, \tag{3.4.12}$$

which is directly valid. □

3.5 Partially Monotonic Critic Neural Network

Building on the partial monotonicity of the Q function, we aim to develop novel training methods guaranteeing the monotonicity of the critic NN that can help solve Problem 3.3.1 more effectively compared to conventional learning methods. We consider two partially monotonic critic NN training methods: (i) network architecture-enabled monotonicity and (ii) regularization-based monotonicity in the sequel.

3.5.1 Network Architecture-enabled Monotonicity

There are many monotonic NNs developed to satisfy the monotonicity of the critic NN [139–141]. We propose a three-layer critic NN architecture (a shallow NN) that

directly guarantees the partial monotonicity of the Q function, as shown in Fig. 2 (a). The critic NN integrates the outputs of two NN processing the state and the action inputs separately. In particular, the state-processing part (on the top of Fig. 2(a)) has to be a monotonic decreasing function. We adopt the monotonic NN in [142], which guarantees that the output is a monotonic decreasing function of the input with the following conditions: 1) the activation functions are non-decreasing (e.g., sigmoid or ReLU), and 2) the weights satisfy

$$\beta^{(1,2)}_{\bar{i},\bar{j}}\beta^{(2,3)}_{\bar{j}} \leq 0, \forall \bar{i} \in \{1, N(M+1)\}, \bar{j} \in \{1, \ldots, U\}. \tag{3.5.1}$$

In (3.5.1), $\beta^{(1,2)}_{\bar{i},\bar{j}}$ denotes the weight for the connection between $\bar{i}$-th input node and the $\bar{j}$-th hidden node, $\beta^{(2,3)}_{\bar{j}}$ denotes the weight for the connection between the $\bar{j}$-th hidden node and the output node in the critic NN, and U is the size of the hidden layer. There is no specific requirement on the parameters of the action-processing part (at the bottom of Fig. 2(a)).

First, we adopt the sigmoid activation function with positive output. Then, to satisfy the constraint (3.5.1) and to ensure negative Q value outputs (see the definition in (3.3.3)) during training, we restrict the weights $\beta^{(1,2)}_{\bar{i},\bar{j}}$ and $\beta^{(2,3)}_{\bar{j}}$ to be non-negative and non-positive, respectively, i.e., $\beta^{(1,2)}_{\bar{i},\bar{j}} \leftarrow \max(0, \beta^{(1,2)}_{\bar{i},\bar{j}})$ and $\beta^{(2,3)}_{\bar{i},\bar{j}} \leftarrow \min(0, \beta^{(2,3)}_{\bar{i},\bar{j}})$.

The advantage of the presented approach is that the monotonicity is strictly and automatically ensured by the NN architecture during the training at each time step. A potential drawback is that the "shallow" monotonic critic NN has a single hidden layer, limiting its capability for approximating complex functions, which are required when solving scheduling problems with large state and action spaces. Note that although there are some deep monotonic NN, such as deep lattice networks [143], these NNs have very complex architectures and are computationally expensive for

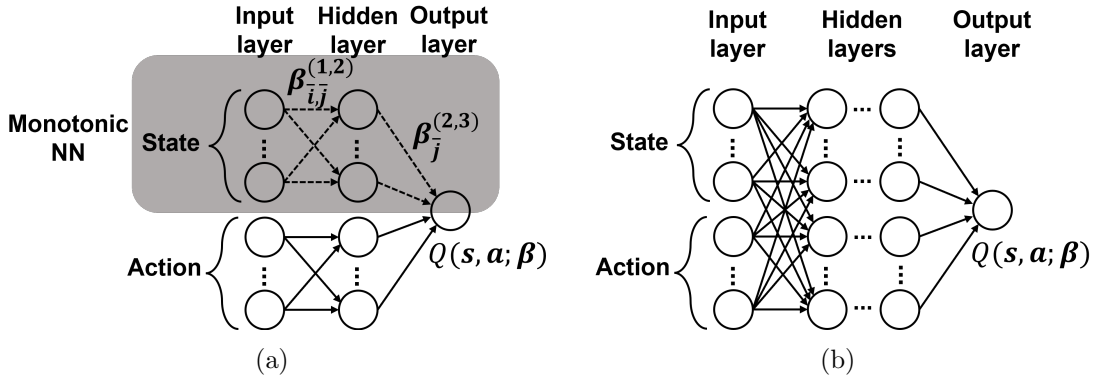


Figure 3.1: Partially monotonic critic NNs. (a) A shallow critic NN with network architecture-enabled monotonicity, where the NN has only one hidden layer. (b) A deep critic NN with regularization-based monotonicity. The dashed lines represent the connections with the weights meeting the constraint (3.5.1) and the straight lines represent the unconstrained connections.

large inputs. Therefore, we propose the following approach for training a "deep" monotonic critic NN, which is a multi-layer fully connected NN without a specific NN architecture requirement.

3.5.2 Regularization-based Monotonicity

We introduce penalty terms in the loss function of the critic NN training to enforce a monotonically *decreasing* relationship between the state input and the output. In particular, we propose two types of penalty for monotonicity regularization.

Positive derivative penalty (type I)

Let $s_{i,j}$ denote the j-th state of the length-$N(M+1)$ state vector $\mathbf{s}_i$, which can be an AoI state (i.e., $j \leq N$) or a channel state (i.e., $j > N$). The partial derivative of the Q function $Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\beta})$ at the state input $s_{i,j}$ is given as

$$\mathsf{QD}_{i,j} = \nabla_{s_{i,j}} Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\beta}) \quad \forall j \in \{1, \ldots, N(M+1)\}. \tag{3.5.2}$$

To enforce the critic NN approximating a monotonically decreasing function at $s_{i,j}$, the corresponding positive derivative penalty is $\max(0, \mathsf{QD}_{i,j})$.

The computation of the derivative (3.5.2) requires backward propagation, which has a higher computational complexity than forward propagation, i.e., evaluating a Q value by feeding the critic NN with an input. Thus, we propose a forward-propagation-only penalty scheme below.

Positive increment penalty (type II)

We introduce the Q value increment from states $\mathbf{s}_i$ to $\mathbf{s}'_{i,j}$ as

$$\mathsf{QI}_{i,j} = Q(\mathbf{s}'_{i,j}, \mathbf{a}_i; \boldsymbol{\beta}) - Q(\mathbf{s}_i, \mathbf{a}_i; \boldsymbol{\beta}), \tag{3.5.3}$$

where $\mathbf{s}'_{i,j}$ is identical to $\mathbf{s}_i$ except for the j-th state with one-step increment, $s'_{i,j} = s_{i,j} + 1$. To enforce the critic NN approximating a monotonically decreasing function from $\mathbf{s}_i$ to $\mathbf{s}'_{i,j}$, we define the positive increment penalty as $\max(0, \mathsf{QI}_{i,j})$.

Potentially, given the vector state $\mathbf{s}_i$ and based on the penalty defined above related to $s_{i,j}$, one can calculate penalty terms for all $j \in \{1, \ldots, N(M+1)\}$, leading to high computational cost when N and M are large. Therefore, we propose a novel scheme to compute the penalty effectively.

Stochastic penalty sampling scheme. First, given the state action pair $(\mathbf{s}_i, \mathbf{a}_i)$, we define an effective set $\mathcal{J}_i \subset \{1, \ldots, N(M+1)\}$ of the individual states for penalty calculation. From (3.5.2) and (3.5.3), a penalty term is effective if $\mathsf{QD}_{i,j}$ or $\mathsf{QI}_{i,j}$ is positive and the Q value changes dramatically. From (3.4.7) of Theorem 3.4.2, channel $h_{n,m}$'s quality improvement does not change the Q value at all, if device n is not scheduled at channel m. Thus, we can eliminate those cases when calculating the penalty. We define an indicator on whether a channel state $s_{i,j}$ with $j > N$ should

Table 3.1: Summary of Hyperparameters of different DDPG algorithms

Hyperparameters	Value
Mini-batch size, B	128
Time horizon for each episode	500
Experience replay memory size, K	20000
Learning rate of actor network and critic network	0.0001 and 0.001
Decay rate of learning rate	0.001
Soft parameter for target update, δ	0.005
Input dimension of actor network	$N + N \times M$
Output dimension of actor network	N
Input dimension of critic network	$2N + N \times M$
Output dimension of critic network	1

be considered for penalty evaluation or not, i.e.,

$$I(s_{i,j}) = \begin{cases} j & \mathbf{a}_i \text{ utilizes } s_{i,j}\text{'s corresponding channel,} \\ \emptyset & \text{otherwise.} \end{cases} \tag{3.5.4}$$

Then, the effective set for penalty calculation is

$$\mathcal{J}_i = \{1, \ldots, N, I(s_{i,N+1}), \ldots, I(s_{i,N(M+1)})\}. \tag{3.5.5}$$

Second, we randomly generate a uniformly down-sampled set $\bar{\mathcal{J}}_i$ with size $K \ll N(M+1)$ based on $\mathcal{J}_i$. From (3.5.2) and (3.5.3), the loss function of the transition $\mathcal{T}_i$ regularized by the penalty terms determined by $\bar{\mathcal{J}}_i$ is

$$L_i(\boldsymbol{\beta}) = \begin{cases} \mathsf{TD}_i^2 + \sum_{j \in \bar{\mathcal{J}}_i} \max\left(0, \mathsf{QD}_{i,j}\right) & \text{type I regularized,} \\ \mathsf{TD}_i^2 + \sum_{j \in \bar{\mathcal{J}}_i} \max\left(0, \mathsf{QI}_{i,j}\right) & \text{type II regularized.} \end{cases} \tag{3.5.6}$$

A well-trained critic NN $\boldsymbol{\beta}$ should minimize both the TD error and the penalty term in the loss function above.

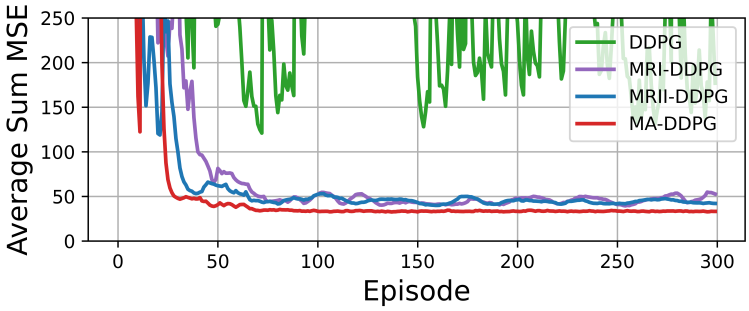


Figure 3.2: Average sum MSE during training with $N = 6, M = 3$. The critic NN has only one hidden layer – a shallow NN setup.

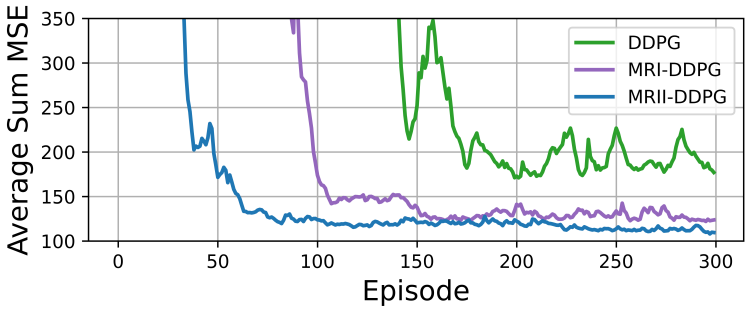


Figure 3.3: Average sum MSE during training with $N = 14, M = 7$. The critic NN has three hidden layers – a deep NN setup.

3.6 Numerical Experiments

In this section, we evaluate and compare the performance of the three monotonicity-enforced training methods for DDPG proposed in the previous section, namely monotonic architecture-based DDPG (MA-DDPG), type I monotonicity-regularized DDPG (MRI-DDPG), and type II monotonicity-regularized DDPG (MRII-DDPG).

The computing platform for running the numerical experiments and the settings of the remote estimation system are the same as section 2.2. In terms of the system parameters, each process has a two-dimensional state and scalar measurement, i.e.,

Table 3.2: Performance Comparison of the Different DDPG Algorithms

System Scale (N, M)	Algorithms	Training time /episode (second)	Experiment 1		Experiment 2		Experiment 3		Experiment 4	
			NEC	MSE	NEC	MSE	NEC	MSE	NEC	MSE
$(6, 3)$	DDPG	3.10	–	–	–	–	107	133.7559	–	–
	MA-DDPG	3.17	**58**	**50.4761**	**80**	**80.3567**	**30**	**56.3479**	**56**	**66.5171**
	MRI-DDPG	4.69	74	57.3178	–	–	82	61.4528	–	–
	MRII-DDPG	4.01	66	57.4541	–	–	53	61.4805	–	–
$(14, 7)$	DDPG	6.16	189	160.6451	162	136.5146	–	–	–	–
	MRI-DDPG	8.15	148	122.1874	105	114.3175	204	140.6513	176	207.1918
	MRII-DDPG	7.20	**87**	**112.3492**	**60**	**108.8440**	**160**	**131.0342**	**113**	**187.2761**
$(20, 10)$	DDPG	8.32	193	210.6116	188	199.5522	213	175.2343	–	–
	MRI-DDPG	9.83	158	172.7708	163	181.6881	182	144.6849	127	180.3464
	MRII-DDPG	9.37	**116**	**155.7731**	**109**	**170.6473**	**131**	**134.4982**	**103**	**167.6880**

$l_n = 2$ and $c_n = 1$. The spectral radius of $\mathbf{A}_n$ and the entries of $\mathbf{C}_n$ are randomly sampled from the range (1, 1.3) and (0,1), respectively. The channel states are quantized into $\bar{h} = 5$ levels from the Rayleigh fading channel with the scale parameter drawn uniformly from the range (0.5, 2) [144]. The packet drop probabilities of different levels are set as 0.2, 0.15, 0.1, 0.05, and 0.01. The critic NN for a 6-sensor-3-channel system and other cases has 1 hidden layer and 3 hidden layers, respectively, each with 1024 nodes. The actor NN for all cases has 3 hidden layers with 1024 nodes. All the NNs are fully connected NNs. The number of penalty terms evaluated is $K = 2$. The setting of the other hyperparameters for the different DDPG algorithms are shown in Table 3.1.

Figs. 3.2 and 3.3 show the average sum mean square error (MSE) of all processes during the training achieved by different DDPG algorithms with $N = 6$, $M = 3$ and $N = 14$, $M = 7$, respectively. Fig. 3.2 illustrates that the MA-DDPG reduces the average sum MSE by 15% when compared to the MRI-DDPG and MRII-DDPG, while the baseline DDPG cannot converge with the shallow critic NN. Thus, for a small-scale system, MA-DDPG is the most effective algorithm. Fig. 3.3 shows that MRI-DDPG and MRII-DDPG save about 15% and 50% training episodes for convergence,

respectively, and also reduce the average sum MSE by 30% when compared to DDPG. MRII-DDPG performs better than MRI-DDPG, since the state space is discrete, and MRII-DDPG directly regularizes the Q function monotonicity at the discrete state space. Note that MA-DDPG cannot converge due to its shallow NN architecture.

In Table 3.2, we evaluate the training time, the number of episodes for convergence (NEC), and the performance of different DDPG algorithms (based on 20000-step simulations) over different system scales and parameters. We see that for the 20-sensor-10-channel systems, MRII-DDPG can reduce the average MSE by 27% when compared to DDPG (at the cost of the increased training time per episode by 13%), whilst saving more than 30% time for convergence (i.e., training time/episode $\times$ NEC), despite the extended training duration of each episode. In particular, MRII-DDPG solves the scheduling problem in Experiments 4 and 5 effectively, while the baseline DDPG fails to converge.

3.7 Conclusion

In this work, building on the monotonicity of the Q function of the optimal semantic-aware scheduling policy, we have developed monotonicity-driven DRL algorithms to solve the scheduling problems effectively and reduce the MSE by about 30% compared to the conventional DRL algorithm.

Chapter 4

Goal-oriented Transmission Scheduling: Structure-guided DRL with a Unified On-policy and Off-policy Approach

4.1 Introduction

Conventional communications focus on accurate bit-by-bit data transmission, achieving near-Shannon-capacity efficiency in 5G networks [145–147]. However, as we transition to 6G, **goal-oriented communications** emerge as a transformative paradigm, prioritizing application-driven objectives over raw data accuracy [71]. This shift is critical for enabling intelligent and efficient next-generation networks. Goal-oriented communications encompass two main categories: **human-centric** and **machine-centric**. Human-centric applications, such as extended reality (XR) [133] and augmented reality (AR) [148], focus on preserving semantic meaning for accurate human comprehension. Machine-centric applications, including industrial Internet of Things (IIoT) [10] and autonomous driving [149], prioritize transmitting information that directly optimizes system performance. This paradigm shift transcends traditional

communication approaches, enabling smarter, more efficient interactions between humans, machines, and their environments [145].

Efficient scheduling is crucial in goal-oriented communications to maximize the performance of systems with limited communication resources. Scheduling determines how devices share communication channels, directly impacting the timeliness and relevance of transmitted information—key factors for achieving system goals. Unlike conventional communication systems that typically use throughput, latency, or reliability as performance metrics, goal-oriented communications adopt application-specific metrics that evaluate the importance of the transmitted information in achieving the desired objective. Among these, the age of information (AoI) [106], which measures the freshness of data, is particularly important for machine-type applications where outdated messages can become irrelevant or even harmful to system performance.

To address scheduling challenges, optimizing scheduling policies has garnered significant attention in the communications community [71]. Existing works often use AoI and related metrics to guide scheduling decisions. For example, in [150], a control cost minimization problem in a single-loop network is reformulated as an AoI-based optimization problem and solved using Markov decision processes (MDPs) with value iteration. Beyond AoI, other metrics such as the value of information (VoI)[151] and mean square error (MSE)[65] have been introduced to assess information importance in various contexts. In [151], an optimal scheduling and power allocation problem is proposed for a single-sensor-single-controller system, maximizing VoI through an event-triggered policy. However, these approaches have notable limitations. Heuristic methods, while computationally efficient, cannot guarantee optimality. On the other

hand, conventional dynamic programming methods, such as value and policy iteration, are computationally infeasible for large-scale systems with high-dimensional state and action spaces. These challenges underscore the need for scalable and optimal solutions tailored to the complexities of modern goal-oriented communication systems.

In recent years, deep reinforcement learning (DRL) has emerged as a powerful tool for solving large-scale MDPs by leveraging deep neural networks (NNs) to approximate functions [78, 152, 153]. All these methods rely on off-policy DRL, where the current policy is updated using data collected from past policies. While off-policy methods exhibit high sample efficiency and effective exploration by reusing past data, they also introduce training instability and bias due to potential discrepancies between the behaviors of the current and past policies. In contrast, on-policy DRL provides a more stable learning process by updating policies using data generated exclusively by the current policy [82, 154]. This approach, which discards previously collected data after each update to keep training data closely aligned with the current policy, ensures greater stability in stochastic environments. While on-policy methods offer stability and have been effectively applied in specific scenarios, they suffer from poor data efficiency because generated data is discarded after each update. This inefficiency, combined with insufficient exploration, can sometimes hinder performance, particularly in scenarios requiring unbiased sampling [57].

In this chapter, we prove the convexity of the optimal value function and then merge the off-policy and on-policy DRL algorithms based on these properties. We aim to integrate their strengths while mitigating weaknesses so that the developed algorithm can solve the large-scale MDPs as the on-policy method and achieve a

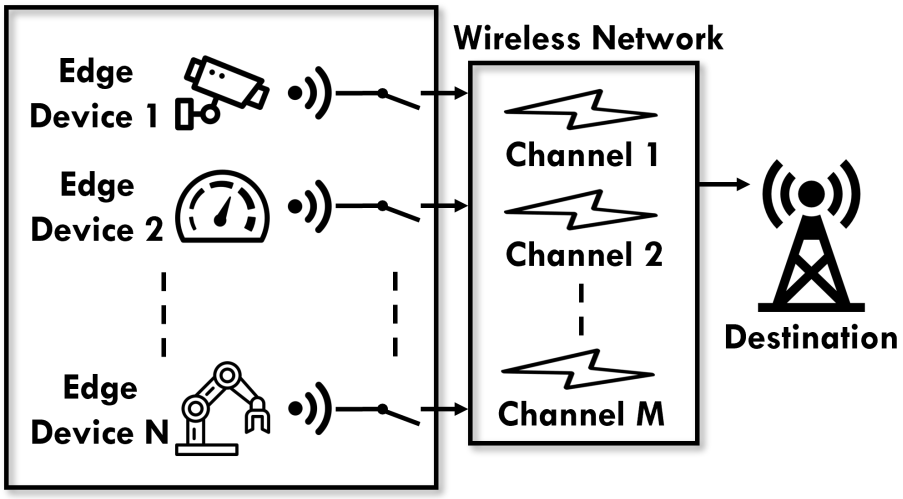


Figure 4.1: Goal-oriented communication system with N edge devices, M channels, and a remote destination

similar performance as the off-policy method.

4.2 System Model

We consider a wireless goal-oriented communication system with N edge devices (e.g., cameras or sensors), and a remote destination (e.g., a base station or remote estimator) as illustrated in Fig 4.1. These devices transmit the local data to the remote destination through M channels (e.g., subcarriers) where $M < N$.

4.2.1 Communication Model

In this chapter, the wireless channel model and the channel assignment are considered the same as the 3.2. The distribution of the channel state $h_{n,m,t}$ is given as

$$\Pr(h_{n,m,t} = i) = q_i^{(n,m)}, \forall t, \tag{4.2.1}$$

where $\sum_{i=1}^{\bar{h}} q_i^{(n,m)} = 1, \forall n, m$. The channel assignment for device n at time step t is denoted as

$$a_{n,t} = \begin{cases} 0, & \text{if no channel is allocated to device } n \text{ at time } t, \\ m, & \text{if channel } m \text{ is allocated to device } n \text{ at time } t, \end{cases} \tag{4.2.2}$$

and is assumed to satisfy the constraint

$$\sum_{n=1}^{N} \mathbb{1}\left(a_{n,t} = m\right) = 1, \sum_{m=1}^{M} \mathbb{1}\left(a_{n,t} = m\right) \leq 1, \tag{4.2.3}$$

where $\mathbb{1}(\cdot)$ is the indicator function. The constraint (2.2.6) means that each channel is allocated to one device and each device can be scheduled to at most one channel

4.2.2 Goal-oriented Communication Performance Metric

We define $\tau_{n,t} \in \{1, 2, \dots\}$ as the AoI of the device n at time t, which refers to the time elapsed since the last successful receive of device packet at the destination [66, 138]:

$$\tau_{n,t+1} = \begin{cases} 1, & \text{if remote destination receive device } n\text{'s packet at time } t \\ \tau_{n,t}+1, & \text{otherwise.} \end{cases} \tag{4.2.4}$$

To characterize the significance of different information in the goal-oriented communication system under different scenarios, we define a positive cost function $c_n(\tau_{n,t})$, $\forall n \in \{1, 2, \dots, N\}$ as critical metrics of device n, where lower cost value means better performance. The cost function is non-decreasing w.r.t the AoI and varies between different goals of the system. The remote state estimation system in section 2.2 is an example of one of the goal-oriented communication systems and the cost function is defined as

$$c_n(\tau_{n,t}) \triangleq \mathrm{Tr}(\mathbf{P}_{n,t}). \tag{4.2.5}$$

4.3 Goal-oriented Transmission Scheduling

Our goal is to find a stationary and deterministic transmission scheduling policy $\pi(\cdot)$ that, considering the AoI states $\boldsymbol{\tau}_t$ and the channel states $\mathbf{H}_t$, minimizes the infinity-horizon expected sum of cost function with a discount factor $\gamma \in (0, 1)$ across all N devices.

Problem 4.3.1.

$$\min_{\pi} \lim_{T\to\infty} \mathbb{E}^{\pi} \left[\sum_{t=1}^{T} \sum_{n=1}^{N} \gamma^t c_n(\tau_{n,t}) \right]. \tag{4.3.1}$$

4.3.1 Markov Decision Process Formulation

Problem 4.3.1 is a sequential decision-making problem with the Markovian property and hence can be cast as an MDP as below.

The state, action, and transition probability are the same as the MDP formulation in section 2.3.1. The immediate cost (i.e., the sum cost of all edge devices) at time t is defined as $c_t = \sum_{n=1}^{N} c_n(\tau_{n,t})$.

4.3.2 Definition of value functions for the optimal policy

For the optimal scheduling policy of the formulated MDP, i.e., $\pi^*(\cdot)$, we define the action-value function $Q(\mathbf{s}_t, \mathbf{a}_t) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and the optimal state-value function, $V^*(\mathbf{s}_t) : \mathcal{S} \to \mathbb{R}$ as below.

Given the current state $\mathbf{s}_t = (\boldsymbol{\tau}_t, \mathbf{H}_t)$ and action $\mathbf{a}_t$, the action-value function, also known as Q function, represents the expected cumulative discounted cost of executing

action $\mathbf{a}_t$ and following the optimal policy $\pi^*(\cdot)$, i.e.,

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} c(\mathbf{s}_{t'}) \Big| \mathbf{a}_t, \mathbf{a}_{t'} = \pi^*(\mathbf{s}_{t'}), \forall t' > t\right], \tag{4.3.2}$$

which satisfies the Bellman optimality equation:

$$Q(\mathbf{s}_t,\mathbf{a}_t) = c(\mathbf{s}_t) + \gamma \sum_{\mathbf{s}_{t+1}} \Pr(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t) \min_{\mathbf{a}_{t+1}\in\mathcal{A}} Q(\mathbf{s}_{t+1},\mathbf{a}_{t+1}). \tag{4.3.3}$$

By utilizing the optimal policy $\pi^*(\cdot)$, we derive the optimal action as

$$\mathbf{a}_t^* \triangleq \pi^*(\mathbf{s}_t) = \arg\min_{\mathbf{a}_t\in\mathcal{A}} Q(\mathbf{s}_t, \mathbf{a}_t). \tag{4.3.4}$$

Based on the Q-function and optimal action, we define the optimal value function, also called the optimal V function, as

$$V^*(\mathbf{s}_t) = \min_{\mathbf{a}_t\in\mathcal{A}} Q(\mathbf{s}_t, \mathbf{a}_t), \tag{4.3.5}$$

which is the minimum expected discounted sum of the future cost starting in state $\mathbf{s}_t$ under the optimal policy $\pi^*(\cdot)$. Based on (2.4.4), (4.3.4) and (4.3.5), the optimal V function and the Q function are related as:

$$Q(\mathbf{s}_t, \mathbf{a}_t) \geq Q(\mathbf{s}_t, \mathbf{a}_t^*) = V^*(\mathbf{s}_t). \tag{4.3.6}$$

4.4 Structural Properties of Optimal V Functions and Optimal Policies

In this section, we derive structural properties of the optimal V function and the optimal scheduling policy, which will be leveraged in the design of advanced DRL algorithms in Section 4.5. Similar to the MDP formulation in Section 2.3.1, we use $\mathbf{a}, \mathbf{s}$, and $\mathbf{s}^+$ to represent $\mathbf{a}_t, \mathbf{s}_t$, and $\mathbf{s}_{t+1}$, respectively, for the notation simplicity in this section.

4.4.1 Monotonicity of Optimal V function

In our earlier chapter, we established the following result about the monotonicity of the optimal V function w.r.t. the AoI state:

Lemma 4.4.1 (Monotonicity of optimal V function w.r.t. AoI states)**.** *Consider states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}'_{AoI} = (\boldsymbol{\tau}'_{(n)}, \mathbf{H})$*, where* $\boldsymbol{\tau}'_{(n)}$ *is identical to* $\boldsymbol{\tau}$ *except for the* n*th AoI state, which is* τ'_n*, and* $\tau'_n \geq \tau_n$*, then, the optimal V function holds the inequality:*

$$V^*(\mathbf{s}'_{AoI}) \geq V^*(\mathbf{s}). \tag{4.4.1}$$

We now prove that the optimal V function is also monotonically decreasing in terms of the channel states as below.

Theorem 4.4.1 (Monotonicity of the optimal V function w.r.t. channel states)**.** *Consider states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}'_{Ch} = (\boldsymbol{\tau}, \mathbf{H}'_{(n,m)})$*, where* $\mathbf{H}$ *and* $\mathbf{H}'_{(n,m)}$ *are identical except for the element in the* n*th row and* m*th column* $h_{n,m} < h'_{n,m}$*. The optimal V function holds the inequality:*

$$V^*(\mathbf{s}'_{Ch}) \geq V^*(\mathbf{s}). \tag{4.4.2}$$

Proof. To prove Theorem 4.4.1 based on (4.3.6), it is sufficient to prove

$$Q(\mathbf{s}, \mathbf{a}^*) \geq Q(\mathbf{s}'_{\mathrm{Ch}}, \mathbf{a}^*), \tag{4.4.3}$$

where $\mathbf{a}^*$ is the optimal action w.r.t the state $\mathbf{s}$, i.e., $\mathbf{a}^* = \pi^*(\mathbf{s})$. From (2.2.5) and (4.2.4), we have the transition probability of the AoI state

$$\mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau}, \mathbf{H}, \mathbf{a}) = \prod_{n=1}^{N} \mathrm{P}(\tau_n^+|\tau_n, \mathbf{h}_n, a_n) = \mathrm{P}(\tau_i^+|\tau_i, \mathbf{h}_i, a_i)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+|\boldsymbol{\tau}_{-i}, \mathbf{H}_{-i}, \mathbf{a}_{-i}), \tag{4.4.4}$$

where $\mathbf{a}_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_N)$ and $\boldsymbol{\tau}_{-i} = (\tau_1, \ldots, \tau_{i-1}, \tau_{i+1}, \ldots, \tau_N)$ denote all actions and AoI states without the device i, respectively, and $\mathbf{h}_i = (h_{i,1}, h_{i,2}, \ldots, h_{i,M})$ represents the channel states of all channels between device i and the remote destination, and $\mathbf{H}_{-i} = (\mathbf{h}_1, \ldots, \mathbf{h}_{i-1}, \mathbf{h}_{i+1}, \ldots, \mathbf{h}_N)$. By replacing (4.3.5) and (4.4.4)

in (2.4.4), we derive that

$$\begin{aligned} Q(\mathbf{s},\mathbf{a}) &= c(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}^+} \mathrm{P}(\mathbf{H}^+)\, \mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H},\mathbf{a}) V^*(\mathbf{s}^+) \\ &= c(\mathbf{s}) + \gamma \sum_{\mathbf{H}^+} \sum_{\tau_i^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\mathbf{H}^+)\, \mathrm{P}(\tau_i^+|\tau_i,\mathbf{h}_i,a_i)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+|\boldsymbol{\tau}_{-i},\mathbf{H}_{-i},\mathbf{a}_{-i}) V^*(\mathbf{s}^+). \end{aligned} \tag{4.4.5}$$

In the following, we represent $\mathbf{H}'_{i,m}$ with $\mathbf{H}'$ for the notation simplicity and prove (4.4.3) with different cases of the optimal action $\mathbf{a}^*$.

(a) If $a_i^* \neq m$, then

$$\begin{aligned} & Q(\mathbf{s},\mathbf{a}^*) - Q(\mathbf{s}'_{\mathrm{Ch}},\mathbf{a}^*) \\ &= [c(\mathbf{s}) - c(\mathbf{s}'_{\mathrm{Ch}})] + \gamma \Bigg[\sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}^+} \mathrm{P}(\mathbf{H}^+)\, \mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H},\mathbf{a}) V^*(\boldsymbol{\tau}^+,\mathbf{H}^+) \\ &\quad - \sum_{\mathbf{H}'^+} \sum_{\boldsymbol{\tau}^+} \mathrm{P}(\mathbf{H}'^+)\, \mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H}',\mathbf{a}) V^*(\boldsymbol{\tau}^+,\mathbf{H}'^+) \Bigg] \\ &= 0 \end{aligned} \tag{4.4.6}$$

where the first equality is from (4.4.5), and the second equality is from $c(\mathbf{s}) = c(\mathbf{s}'_{\mathrm{Ch}})$,

$$\sum_{\mathbf{H}^+} \mathrm{P}(\mathbf{H}^+) V^*(\boldsymbol{\tau}^+,\mathbf{H}^+) = \sum_{\mathbf{H}'^+} \mathrm{P}(\mathbf{H}'^+) V^*(\boldsymbol{\tau}^+,\mathbf{H}'^+) \tag{4.4.7}$$

with i.i.d. fading channel and $\mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H},\mathbf{a}) = \mathrm{P}(\boldsymbol{\tau}^+|\boldsymbol{\tau},\mathbf{H}',\mathbf{a})$ as $a_i^* \neq m$.

(b) If $a_i^* = m$, then we have

$$\begin{aligned}
&Q(\mathbf{s}, \mathbf{a}^*) - Q(\mathbf{s}'_{\text{Ch}}, \mathbf{a}^*) \\
&= [c(\mathbf{s}) - c(\mathbf{s}'_{\text{Ch}})] \\
&\quad + \gamma \Bigg[\sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}^+_{-i}} \sum_{\tau_i^+} \mathrm{P}(\mathbf{H}^+) \, \mathrm{P}(\boldsymbol{\tau}^+_{-i} | \boldsymbol{\tau}_{-i}, \mathbf{H}_{-i}, \mathbf{a}_{-i}) \, \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{h}_i, a_i) V^*(\boldsymbol{\tau}^+, \mathbf{H}^+) \\
&\quad - \sum_{\mathbf{H}^+} \sum_{\boldsymbol{\tau}^+_{-i}} \sum_{\tau_i^+} \mathrm{P}(\mathbf{H}^+) \, \mathrm{P}(\boldsymbol{\tau}^+_{-i} | \boldsymbol{\tau}_{-i}, \mathbf{H}_{-i}, \mathbf{a}_{-i}) \, \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{h}'_i, a_i) V^*(\boldsymbol{\tau}^+, \mathbf{H}^+) \Bigg] \\
&\geq 0, \qquad (4.4.8)
\end{aligned}$$

where the first equality is derived based on (4.4.5) and (4.4.7), and the second inequality is based on the following inequalities

$$\begin{aligned}
&(1 - p_{i,m}) V^*(1, \boldsymbol{\tau}^+_{-i}, \mathbf{H}^+) + p_{i,m} V^*(\tau_i + 1, \boldsymbol{\tau}^+_{-i}, \mathbf{H}^+) \\
&\geq (1 - p'_{i,m}) V^*(1, \boldsymbol{\tau}^+_{-i}, \mathbf{H}^+) + p'_{i,m} V^*(\tau_i + 1, \boldsymbol{\tau}^+_{-i}, \mathbf{H}^+) \qquad (4.4.9)
\end{aligned}$$

achieved by $p_{i,m} \geq p'_{i,m}$, $a_i^* = m$ and Lemma 4.4.1.

□

From Lemma 4.4.1 and Theorem 4.4.1, the optimal V function monotonically increases with both the AoI and channel states.

4.4.2 Convexity of Cost function and Optimal V function

Since the input state of the optimal V function takes only discrete values, we define its convexity as below.

Definition 4.4.1 (Discrete convexity of optimal V function and cost function w.r.t. AoI)**.** *Consider states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, $\mathbf{s}'_{AoI} = (\boldsymbol{\tau}'_{(n)}, \mathbf{H})$, *and* $\mathbf{s}''_{AoI} = (\boldsymbol{\tau}''_{(n)}, \mathbf{H})$, *where* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n, \ldots, \tau_N)$, $\boldsymbol{\tau}'_{(n)} = (\tau_1, \ldots, \tau'_n, \ldots, \tau_N)$, $\boldsymbol{\tau}''_{(n)} = (\tau_1, \ldots, \tau''_n, \ldots, \tau_N)$, *and*

$\tau'_n \geq \tau_n \geq \tau''_n$. *The cost function and the optimal V function, exhibiting convexity, are defined as satisfying the following inequalities:*

$$\alpha c(\mathbf{s}''_{AoI}) + (1-\alpha)c(\mathbf{s}'_{AoI}) \geq c(\mathbf{s}), \tag{4.4.10}$$

$$\alpha V^*(\mathbf{s}''_{AoI}) + (1-\alpha)V^*(\mathbf{s}'_{AoI}) \geq V^*(\mathbf{s}), \tag{4.4.11}$$

for any $n \in \{1, \ldots, N\}$, *where* $\alpha \in [0,1]$ *and* $\alpha\tau''_n + (1-\alpha)\tau'_n = \tau_n$.

Cost function Convexity

It is important to highlight that cost functions are often convex in practical applications. This implies that the cost can grow increasingly rapidly as the AoI state increases. For problems that aim to optimise overall AoI performance, the cost function is typically a linear function of AoI, which satisfies the convexity property defined above. In the context of the remote state estimation problem presented in Example 2.2, we provide a rigorous proof below to demonstrate that the cost function exhibits convexity when the AoI becomes large.

Lemma 4.4.2 (Asymptotic convexity of the cost function w.r.t. AoI in a remote state estimation system of Example 2.2)**.** ***1) The convexity of each device's cost function:*** *For device n, the cost function*

$$c_n(\tau) = \mathrm{Tr}\left(h_n^{\tau}(\bar{\mathbf{P}}_n)\right), \tag{4.4.12}$$

as defined in (4.2.5)*, is asymptotically convex, i.e., the inequality* $\alpha c_n(\tau') + (1-\alpha)c_n(\tau'') \geq c_n(\tau)$ *holds when* $\alpha \in [0,1]$ *and* $\alpha\tau''_n + (1-\alpha)\tau'_n = \tau_n$, *under the condition* $\tau' \geq \tau \geq \tau'' \gg 1$.

2) The convexity of the overall cost function: *For states* $\mathbf{s}$, $\mathbf{s}'_{AoI}$, *and* $\mathbf{s}''_{AoI}$ *defined in Definition 4.4.1, the inequality* (4.4.10) *holds under the condition* $\tau'_n \geq \tau_n \geq \tau''_n \gg 1$.

Proof. See Appendix B.1. □

Optimal V function Convexity

For a system with two devices and one channel, the convexity of the optimal V function is rigorously proven as follows:

Theorem 4.4.2 (Discrete Convexity of optimal V function w.r.t AoI states in two-device-one-channel systems)**.** *The optimal V function $V(\cdot)$ of the two-device-one-channel system has the convexity in Definition 4.4.1 if the cost function $c(\cdot)$ is convex:*

$$\alpha c(\mathbf{s}'') + (1-\alpha)c(\mathbf{s}') \geq c(\mathbf{s}). \tag{4.4.13}$$

Proof. See Appendix B.2. □

For a general system with multiple devices and multiple channels, proving the convexity becomes challenging due to the increased dimensionality of the state and action spaces. This higher dimensionality leads to a more complex set of transition states, making it difficult to directly verify the convexity property across all possible transitions. Instead, we establish the asymptotic convexity of the optimal V function as follows:

Theorem 4.4.3 (Asymptotic discrete convexity of the optimal V function w.r.t AoI states in multi-device-multi-channel systems)**.** *The optimal V function $V(\cdot)$ of a multi-device-multi-channel system has an asymptotic convexity: suppose that the cost function $c(\cdot)$ is convex, i.e.,*

$$\alpha c(\mathbf{s}'') + (1-\alpha)c(\mathbf{s}') \geq c(\mathbf{s}). \tag{4.4.14}$$

the value function $V(\cdot)$ is also convex i.e.,

$$\alpha V^*(\mathbf{s}'') + (1-\alpha)V^*(\mathbf{s}') \geq V^*(\mathbf{s}), \tag{4.4.15}$$

for states $\mathbf{s}'' = (\boldsymbol{\tau}''_{(i)}, \mathbf{H})$, $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$ *and* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, *where* $\alpha \in [0,1], \alpha\tau''_i + (1-\alpha)\tau'_i = \tau_i$, *and* $\tau'_i \geq \tau_i \gg \tau''_i$.

Proof. See Appendix B.3. □

Theorem 4.4.3 means that the optimal V function has discrete convexity in a multi-device-multi-channel system, at least when the corresponding AoI state of the state $\mathbf{s}$ is relatively large compared to the state $\mathbf{s}''$. In a special case where the devices have identical channel states (i.e., are co-located), we further establish the asymptotic convexity of the optimal value function when the evaluated states $\mathbf{s}'_{\text{AoI}}$, $\mathbf{s}$ and $\mathbf{s}''_{\text{AoI}}$ all correspond to large AoI values. This result is formalized below:

Proposition 4.4.1 (Asymptotic convexity of the optimal V function w.r.t. AoI for co-located devices)**.** *Given states* $\mathbf{s}$, $\mathbf{s}'_{AoI}$, *and* $\mathbf{s}''_{AoI}$ *defined in Definition 4.4.1 with* $\tau'_n \geq \tau_n \geq \tau''_n \gg 1$, *and assuming that the devices experience identical channel conditions, the optimal V function* $V(\cdot)$ *of a multi-device-multi-channel system is convex, provided the cost function* $c(\cdot)$ *is convex.*

Proof. See Appendix B.4 □

Remark 4.4.1 (Why not channel state convexity?)**.** *The convexity of the optimal V function with respect to channel states has not been derived because it is neither meaningful nor necessary in this context. The cost function, and thus the optimal V function, fundamentally depends on the AoI values rather than the channel states. Channel states play an indirect role, and in systems with independent and fluctuating channels, their specific values often become irrelevant, especially when a device is not using a particular channel. Additionally, analysing convexity with respect to channel*

states would require comparing an overwhelming number of state combinations, making it impractical and adding no significant insight. Focusing on AoI, which directly impacts the system's performance, provides a more relevant and useful understanding.

4.4.3 Greedy Structure of Optimal Policy

In this section, beyond analyzing the properties of the optimal V function, we aim to establish the structure of the optimal scheduling policy. Deriving the structure of the optimal scheduling policy for a general multi-sensor, multi-channel system is not feasible due to the complexity of the problem. Instead, we focus on a special case involving co-located devices with identical channel states. This simplification, which focuses solely on the AoI states of different devices while disregarding variations in their channel states, allows us to address the problem more tractably and extract meaningful insights.

To achieve this, we first define the mandatory scheduling set as follows:

Definition 4.4.2 (Mandatory scheduling set)**.** *Consider an N-device-M-channel system. If there exists a threshold $\bar{\tau}$ such that the asymptotic cost function satisfies the following ordering inequality:*

$$c_{i_1}(\tau) \geq c_{i_2}(\tau) \cdots \geq c_{i_N}(\tau), \ \forall \tau \geq \bar{\tau}, \tag{4.4.16}$$

where $i_n \in \{1, \ldots, N\}$ represents an device index, then the following holds:

Given the AoI state $(\tau_1, \ldots, \tau_N)$, if there exists a largest number $\bar{N} \leq M$ such that the set $\mathcal{I} \triangleq \{i_1, \ldots, i_{\bar{N}}\}$ includes the $\bar{N}$ devices with the largest AoI states, each greater than $\bar{\tau}$, then $\mathcal{I}$ is defined as the mandatory scheduling set.

The mandatory scheduling set is time-varying due to the dynamics of the AoI states. If the mandatory scheduling set exists, it is intuitive that all devices within

the set should be scheduled. This is because the instantaneous cost of each device in the set exceeds that of any device outside the set. Moreover, leaving a device in the set unscheduled would result in a higher cost function compared to scheduling a device not belonging to the set. This result and the detailed proof are provided below.

Theorem 4.4.4 (Asymptotic greedy structure of the optimal scheduling policy for co-located devices)**.** *Consider a multi-device-multi-channel system with co-located devices. If the mandatory scheduling set in Definition 4.4.2 exists, the optimal policy schedules all devices within the set, i.e.,* $a_n^* \neq 0, \forall n \in \mathcal{I}$.

Proof. See Appendix B.5. □

4.5 Structure-guided unified on-off policy DRL

In this section, we leverage the theoretical results obtained to develop a structure-guided unified dual on-off policy (SUDO) DRL method. This approach combines the strengths of both off-policy and on-policy DRL, utilizing the state-of-the-art on-policy PPO algorithm, widely regarded as one of the most advanced DRL methods available. First, we briefly overview PPO, which is less familiar than commonly used off-policy DRL methods. Next, we present the proposed SUDO algorithm.

4.5.1 Overview of Vanilla PPO Algorithm

A PPO agent has two neural networks (NNs): an actor NN and a critic NN. **The actor NN**, with the parameter set $\boldsymbol{\varphi}$, approximates the original deterministic policy $\pi^*(\mathbf{s})$ by a stochastic policy $\pi(\tilde{\mathbf{a}}|\mathbf{s};\boldsymbol{\varphi})$, which outputs a probability distribution over

actions $\tilde{\mathbf{a}}$ given the state $\mathbf{s}$. Note that in our original scheduling problem, the action $\mathbf{a}$ is selected from the discrete action space of size $N!/(N-M)!$. Here to apply the PPO algorithm, which operates in a continuous action space, we implement an action mapping method [82]. This approach maps the N-dimensional continuous action $\tilde{\mathbf{a}}$ generated by the actor NN into a corresponding discrete action $\mathbf{a}$. For simplicity in notation, the process of obtaining $\mathbf{a}$ from the actor NN $\boldsymbol{\varphi}$ is represented as a stochastic function:

$$\mathbf{a} = f(\mathbf{s}; \boldsymbol{\varphi}). \tag{4.5.1}$$

The critic NN, with the parameter set $\boldsymbol{\nu}$, approximates the optimal V function $V^*(\mathbf{s})$ as $V(\mathbf{s}; \boldsymbol{\nu})$, outputting the estimated value of the optimal V function for a given state $\mathbf{s}$. Training a PPO agent involves **two iterative steps**: generating an experience trajectory and updating both NNs.

Step 1: Experience generation. The PPO agent generates a trajectory of length T, resulting in the trajectory:

$$\mathcal{T}_{\text{On}} \triangleq \{(\mathbf{s}_t, \tilde{\mathbf{a}}_t, c_t)\}_{t=0}^{T-1}. \tag{4.5.2}$$

At each time step t, the actor NN uses the current stochastic policy $\pi(\tilde{\mathbf{a}}_t|\mathbf{s}_t; \boldsymbol{\varphi}_{\text{old}})$ to sample a continuous action$\tilde{\mathbf{a}}_t$, which is then mapped to the discrete (real) scheduling action $\mathbf{a}_t$. The next state $\mathbf{s}_{t+1}$ and the cost c_t are obtained by executing the real action$\mathbf{a}_t$. The critic NN computes the estimated optimal V function of the state $V(\mathbf{s}_t; \boldsymbol{\nu})$. Using the trajectory $\mathcal{T}_{\text{On}}$, the advantage function A_t and the cost-to-go function C_t are calculated as

$$A_t = \sum_{\tilde{t}=0}^{T-t-1} (\gamma\lambda)^{\tilde{t}} \zeta_{t+\tilde{t}}, \tag{4.5.3}$$

$$C_t = c_t + \gamma V(\mathbf{s}_{t+1}; \boldsymbol{\nu}), \tag{4.5.4}$$

where λ is the generalized advantage estimation (GAE) parameter, and $\zeta_t = c_t + \gamma V(\mathbf{s}_{t+1};\boldsymbol{\nu}) - V(\mathbf{s}_t;\boldsymbol{\nu})$. The trajectory is then updated as

$$\mathcal{T}'_{\text{On}} \triangleq \{(\mathbf{s}_t, \tilde{\mathbf{a}}_t, A_t, C_t)\}_{t=0}^{T-1}. \tag{4.5.5}$$

Step 2: NN update. To update the actor and critic NNs, the PPO agent randomly samples B_1 data points from $\mathcal{T}'_{\text{On}}$ to create a mini-batch dataset:

$$\{(\mathbf{s}_l, \tilde{\mathbf{a}}_l, A_l, C_l)\}_{l=1}^{B_1}. \tag{4.5.6}$$

For the critic NN, the temporal difference (TD) error is defined as:

$$\mathsf{TD}_l \triangleq C_l - V(\mathbf{s}_l;\boldsymbol{\nu}), \tag{4.5.7}$$

and the loss function is given by

$$L(\boldsymbol{\nu}) = \frac{1}{B_1}\sum_{l=1}^{B_1} \mathsf{TD}_l^2. \tag{4.5.8}$$

For the actor NN, the loss function is defined as:

$$L(\boldsymbol{\varphi}) = \frac{1}{B_1}\sum_{l=1}^{B_1} \Big(\min\{q(\mathbf{s}_l;\boldsymbol{\varphi})A_l, \tag{4.5.9}$$

$$\text{clip}\left(q(\mathbf{s}_l;\boldsymbol{\varphi}), 1-\epsilon, 1+\epsilon\right) A_l\} \tag{4.5.10}$$

$$- \omega H(\mathbf{s}_l;\boldsymbol{\varphi})\Big), \tag{4.5.11}$$

where

$$q(\mathbf{s}_l;\boldsymbol{\varphi}) = \frac{\pi(\tilde{\mathbf{a}}_l|\mathbf{s}_l;\boldsymbol{\varphi})}{\pi(\tilde{\mathbf{a}}_l|\mathbf{s}_l;\boldsymbol{\varphi}_{\text{old}})} \tag{4.5.12}$$

is the probability ratio, and

$$\text{clip}(x, x_{\min}, x_{\max}) = \max\{\min\{x, x_{\max}\}, x_{\min}\} \tag{4.5.13}$$

is a clip function. Here, ϵ is a clipping hyper-parameter, ω is the weight for the entropy loss, and

$$H(\mathbf{s}_l;\boldsymbol{\varphi}) = \frac{1}{2}\ln(2\pi \cdot e \cdot \sigma_l^2)$$

represents the policy entropy loss used to encourage exploration, where σ_l is the

deviation for action $\tilde{\mathbf{a}}_l$ when in state $\mathbf{s}_l$ following the current policy. The clip function ensures stable training by constraining large updates.

Finally, the critic and actor NNs are updated by minimizing their respective loss functions using gradient-based optimization methods, such as the Adam optimizer.

4.5.2 Proposed structure-guided unified on-off policy DRL

The SUDO loss functions for actor NN and critic NN are consisted of the on-policy and off-policy parts, i.e.,

$$L_{\mathrm{SUDO}}(\boldsymbol{\nu}) = L_{\mathrm{On}}(\boldsymbol{\nu}) + \beta_1 L_{\mathrm{Off}}(\boldsymbol{\nu}) \tag{4.5.14}$$

$$L_{\mathrm{SUDO}}(\boldsymbol{\varphi}) = L_{\mathrm{On}}(\boldsymbol{\varphi}) + \beta_2 L_{\mathrm{Off}}(\boldsymbol{\varphi}), \tag{4.5.15}$$

where β_1 and β_2 are the hyperparameters to balance the on-policy and off-policy loss functions.

Since the structural evaluation metrics of the critic NN and actor NN cannot test the whole infinite state space, there exists a bias between the derived structure scores and the actual structure of NNs. It makes some trajectories with better actual scores cannot be stored in the replay buffer for the off-policy training. Therefore, the on-policy part is necessary to alleviate this bias by ensuring each generated data will not be discarded directly without use. In the following, we introduce these two different parts of the loss functions.

The use of the policy gradient method and trust region optimization enables PPO to train policy stably and solve MDPs with large state and action space, which cannot be solved by off-policy DRL, such as DDPG and DQN. However, the conventional PPO algorithm only learns from the data collected by the current policy and the previous samples are discarded after updating two NNs. So, the PPO agent cannot

utilize the off-policy data, leading to sample inefficiency and more likely to be stuck into the local optimality. Another drawback of the PPO algorithm is that some structures of the optimal V function and the optimal policy have not been utilized to improve the training.

To improve the algorithm sample efficiency and take advantage of the proven properties of the optimal V function and the optimal policy, we propose **1) a structure-guided (SG) action selection method** based on Theorem 4.4.4 to select the proper actions and supervise the training at the initial stage; **2) the structural properties evaluation terms of NNs** to calculate the differences of NNs' V estimations and generated actions between different states based on structural properties of the optimal V function (monotonicity and convexity) and the optimal policy (action monotonicity); **3) the structure score scheme** to give the derived trajectory with different scores by using the structural properties evaluation terms; **4) a prioritized replay buffer** to improve the sample efficiency by storing and reusing the past trajectories data and giving the stored data with different sampling priorities building on their score; and **5) a Structure-guided Unified (Dual) On-off policy (SUDO) loss function** to penalize the values that do not follow the structure based on the structural properties evaluation terms.

Our Structure-guided Unified (Dual) On-off policy DRL (SUDO-DRL) training algorithm consists of two stages: 1) the co-located-condition training stage, which uses the SG action selection method under the channel state space considered in Theorem 4.4.4, 2) the general-condition training stage with the system in 3.2. The structure score scheme, prioritized replay buffer and SUDO loss function are used in all two stages. Each stage consists of on-policy part and off-policy part, where the

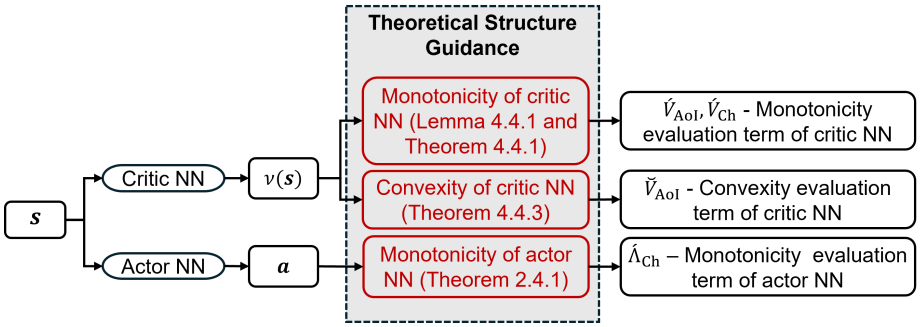


Figure 4.2: Critic and Actor NNs' structural property evaluation framework.

data are sampled from the current experience and the past experience, respectively. The first stage utilizes the SG action selection scheme to enable the DRL agent to learn the relationship between the scheduling action and AoI state rather than the whole state space. The second stage uses both the on-policy and selective off-policy data for the agent training to improve the sample efficiency and prevent it from getting stuck in the local optimality. In what follows, we present the action mapping scheme, SG action selection scheme, structural properties based evaluation terms of NNs, structure score scheme, the prioritized replay buffer, and the SUDO loss function.

Structural Property Evaluation Framework

For each state-action pair $(\mathbf{s}, \mathbf{a})$, we evaluate the critic NN $V(\mathbf{s}; \boldsymbol{\nu})$ based on the proven structural properties of the optimal V function: monotonicity w.r.t. AoI state (Lemma 4.4.1) and channel state (Theorem 4.4.1), as well as convexity w.r.t. AoI state (Theorem 4.4.3). Additionally, we assess the monotonicity of the actor $\pi(\tilde{\mathbf{a}}|\mathbf{s}; \boldsymbol{\varphi})$'s output action w.r.t. channel state in the vicinity of $\mathbf{s}$ (Theorem 2.4.1) using similar penalty metrics. The overall evaluation framework is illustrated in Fig. 4.2.

Monotonicity of the critic NN. For state $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, we define $\acute{V}_{\text{AoI}}$ and $\acute{V}_{\text{Ch}}$ to evaluate the monotonicity of the critic NN w.r.t. AoI state and channel state, respectively:

$$\acute{V}_{\text{AoI}} = \max\left(0, V(\mathbf{s};\boldsymbol{\nu}) - V(\hat{\mathbf{s}}_{(n)};\boldsymbol{\nu})\right) \tag{4.5.16}$$

$$\acute{V}_{\text{Ch}} = \max\left(0, V(\mathbf{s};\boldsymbol{\nu}) - V(\hat{\mathbf{s}}_{(n,m)};\boldsymbol{\nu})\right), \tag{4.5.17}$$

where $\hat{\mathbf{s}}_{(n)} = (\hat{\boldsymbol{\tau}}_{(n)}, \mathbf{H})$ is identical to $\mathbf{s}$ except for a single-step increase in the n-th AoI, i.e.,

$$\hat{\boldsymbol{\tau}}_{(n)} = (\tau_1, \ldots, \tau_n + 1, \ldots, \tau_N), \tag{4.5.18}$$

and $\hat{\mathbf{s}}_{(n,m)} = (\boldsymbol{\tau}, \hat{\mathbf{H}}_{(n,m)})$, where $\hat{\mathbf{H}}_{(n,m)}$ is identical to $\mathbf{H}$ except for the element at the nth row and the mth column as $\min(h_{n,m}+1, \bar{h})$.

The monotonicity metrics (4.5.16) and (4.5.17) indicate that when monotonicity is satisfied, the corresponding metric is zero. However, if monotonicity is violated, the penalty becomes positive and increases proportionally with the extent of the violation.

Convexity evaluation of the critic NN. Similarly, the evaluation metric for convexity of the critic NN is defined as:

$$\breve{V}_{\text{AoI}} = \max\left(0, 2V(\mathbf{s};\boldsymbol{\nu}) - (V(\check{\mathbf{s}}_{(n)};\boldsymbol{\nu}) + V(\hat{\mathbf{s}}_{(n)};\boldsymbol{\nu}))\right), \tag{4.5.19}$$

where $\check{\mathbf{s}}_{(n)} = (\check{\boldsymbol{\tau}}_{(n)}, \mathbf{H})$ is identical to $\mathbf{s}$ except for a single-step decrease in the nth AoI,

$$\check{\boldsymbol{\tau}}_{(n)} = (\tau_1, \ldots, \tau_n - 1, \ldots, \tau_N). \tag{4.5.20}$$

Monotonicity evaluation of the actor NN. As established in Theorem 2.4.1, the monotonicity of the actor NN differs from the structural properties of the critic NN, which are evaluated over the entire state vector. Instead, the monotonicity of the actor NN is assessed for each device's action individually.

Given state $\mathbf{s}$ and a corresponding sampled action for device n, is $a_n = m \neq 0$, we define the state $\check{\mathbf{s}}_{(n,m)} = (\boldsymbol{\tau}, \check{\mathbf{H}}_{(n,m)})$, where $\check{\mathbf{H}}_{(n,m)}$ is identical to $\mathbf{H}$ except for the element at the nth row and mth column is $\max(h_{n,m} - 1, 1)$. The actor NN's action for device n at state $\check{\mathbf{s}}_{(n,m)}$ is then sampled as $a_{\mathrm{ch},n}$.

The evaluation metric for device n's action monotonicity w.r.t. the channel state is defined as

$$\acute{\Lambda}_{\mathrm{Ch},n} = \mathbb{1}\left(a_n \neq 0 \text{ and } a_{\mathrm{Ch},n} \neq a_n\right). \tag{4.5.21}$$

Sampling over the trajectory for structural property evaluation. To efficiently evaluate the structural properties of a trajectory, we sample data points from it rather than considering all data points. We randomly and uniformly sample K state-action pairs $((\mathbf{s}_1, \mathbf{a}_1), \ldots, (\mathbf{s}_k, \mathbf{a}_k), \ldots, (\mathbf{s}_K, \mathbf{a}_K))$ from the trajectory. For each state $\mathbf{s}_k$, the structural evaluation metrics defined above, (4.5.16), (4.5.17), (4.5.19), and (4.5.21), require analyzing changes in different device AoI and channel states. To simplify this process, we uniformly sample Ξ devices for AoI-related evaluations and Ξ elements from the $N \times M$ channel state matrix for channel state evaluations.

To account for these sampled points, we introduce subscripts k and ξ to the evaluation metrics, i.e., $\acute{V}_{\mathrm{AoI},k,\xi}$, $\acute{V}_{\mathrm{Ch},k,\xi}$, $\breve{V}_{\mathrm{AoI},k,\xi}$, and $\acute{\Lambda}_{\mathrm{Ch},k,\xi}$. The sampled data will be used in both the on-policy and off-policy parts.

Trajectory structure evaluation. Using the monotonicity evaluation metrics of the critic neural network (NN), i.e., (4.5.16) and (4.5.17), we define the **critic-monotonicity (CM) score**, which is calculated based on the sampled states as:

$$\mathsf{CM} \triangleq \frac{\sum_{k=1}^{K}\sum_{\xi=1}^{\Xi}\left[\mathbb{1}(\acute{V}_{\mathrm{AoI},k,\xi} = 0) + \mathbb{1}(\acute{V}_{\mathrm{Ch},k,\xi} = 0)\right]}{2K\Xi}. \tag{4.5.22}$$

Similarly, based on the convexity evaluation metric in (4.5.19), we define the **critic-convexity (CC) score** as:

$$\mathsf{CC} \triangleq \frac{\sum_{k=1}^{K}\sum_{\xi=1}^{\Xi}\mathbb{1}(\breve{V}_{\mathrm{AoI},k,\xi}=0)}{K\Xi}. \tag{4.5.23}$$

Then, using the monotonicity evaluation metric of the actor NN from (4.5.21), we define the **actor-monotonicity (AM) score** as:

$$\mathsf{AM} \triangleq \frac{\sum_{k=1}^{K}\sum_{\xi=1}^{\Xi}\mathbb{1}(\acute{\Lambda}_{\mathrm{Ch},k,\xi}=0)}{K\Xi}. \tag{4.5.24}$$

A higher score for CM, CC and AM indicates that the trajectory data aligns more closely with the theoretical structural properties of the optimal policy, suggesting that the policy being evaluated is closer to the optimal policy. These scores will be used in the off-policy part to select trajectories for storage in a replay buffer.

On-Policy Loss Function

The on-policy loss function in SUDO-DRL leverages the current trajectory to create a mini-batch data set of size B_1, following the PPO algorithm described in Section (4.5.1). However, the key difference lies in the introduction of a penalty term for violations of the structural properties in the critic loss function. This penalty is computed based on $\{\acute{V}_{\mathrm{AoI},k,\xi}\}$, $\{\acute{V}_{\mathrm{Ch},k,\xi}\}$, and $\{\breve{V}_{\mathrm{AoI},k,\xi}\}$ derived from the earlier structural property evaluation samplings.

The loss function for the critic NN in the on-policy component is defined as:

$$L_{\mathrm{On}}(\boldsymbol{\nu}) = \frac{1}{B_1}\sum_{l=1}^{B_1}\mathsf{TD}_l^2 + \frac{1}{K\Xi}\sum_{k=1}^{K}\sum_{\xi=1}^{\Xi}\left(\acute{V}_{\mathrm{AoI},k,\xi} + \acute{V}_{\mathrm{Ch},k,\xi} + \breve{V}_{\mathrm{AoI},k,\xi}\right), \tag{4.5.25}$$

where the first term represents the TD loss, and the second term penalizes deviations from the theoretical structural properties.

The loss function for the actor NN remains the same as the conventional PPO

algorithm, as shown in (4.5.11):

$$L_{\mathrm{On}}(\boldsymbol{\varphi}) = L(\boldsymbol{\varphi}). \tag{4.5.26}$$

Since the evaluation metric $\acute{\Lambda}_{\mathrm{Ch},n}$ for actor NN uses the executed action after mapping rather than the virtual action generated directly from actor NN, this metric cannot be used in the loss function as other metrics of critic NN.

Off-policy reply buffer

Unlike on-policy DRL, which discards sampled data from old policies entirely, off-policy DRL retains this data in a replay buffer $\mathcal{R}$ and samples from it for updating the actor and critic NNs. However, data generated by a policy that is far from optimal can negatively impact training because it introduces bias and instability, hindering the convergence toward the optimal policy. To address this, the off-policy part of the proposed SUDO-DRL framework selectively stores high-quality data that aligns well with the theoretical structural properties of the optimal policy, ensuring more effective and stable training.

Structure-Guided Data Storage Scheme. Given the current trajectory $\{\mathbf{s}_t, \tilde{\mathbf{a}}_t, c_t\}_{t=0}^{T-1}$ with index u, we first calculate the average structure scores of the past $\bar{u}$ trajectories. The average CM score is computed based on (4.5.22) as:

$$\mathsf{CM}_{\mathrm{Avg},u} = \frac{1}{\bar{u}} \sum_{\tilde{u}=u-\bar{u}-1}^{u-1} \mathsf{CM}_{\tilde{u}}, \tag{4.5.27}$$

Similarly, the average CC and AM scores are calculated as $\mathsf{CC}_{\mathrm{Avg},u}$ and $\mathsf{AM}_{\mathrm{Avg},u}$, respectively.

Next, we define the condition for storing trajectory u in the replay buffer as:

$$\mathsf{CM}_u \geq \mathsf{CM}_{\mathrm{Avg},u},\ \mathsf{CC}_u \geq \mathsf{CC}_{\mathrm{Avg},u},\ \mathsf{AM}_u \geq \mathsf{AM}_{\mathrm{Avg},u}. \tag{4.5.28}$$

If the trajectory scores satisfy the constraint (4.5.28), all transitions (i.e., state-action-cost-next-state tuples) within the trajectory is stored in $\mathcal{R}$ as:

$$\mathcal{X}_{\text{Off},t} \triangleq (\mathbf{s}_t, \tilde{\mathbf{a}}_t, c_t, \mathbf{s}_{t+1}, p_t, \boldsymbol{\varphi}_t)\,,\ t = 0, \ldots, t-1, \tag{4.5.29}$$

where $\boldsymbol{\varphi}_t$ is the weights of actor NN of the trajectory u and p_t represents the **transition priority indicator**, defined based on the structural scores of the trajectory as:

$$p_u = \mathsf{CM}_u + \mathsf{CC}_u + \mathsf{AM}_u. \tag{4.5.30}$$

Off-policy replay buffer sampling and loss functions

The off-policy loss function in SUDO-DRL samples past experiences to create an off-policy batch data set of size B_2, i.e.,

$$\{\mathcal{X}_{\text{Off},b}\}_{b=1}^{B_2}\,, \tag{4.5.31}$$

according to their priority indicators. Next, the agent randomly samples a mini-batch of data transitions $\{(\mathbf{s}_{b,l}, \tilde{\mathbf{a}}_{b,l}, c_{b,l}, \mathbf{s}_{b,l+1}, \varphi_b)\}_{l=1}^{B_1}$ from $\mathcal{X}_{\text{Off},b}$.

The loss function for critic NN in the off-policy component is defined as:

$$L_{\text{Off}}(\boldsymbol{\nu}) = \frac{1}{B_2}\frac{1}{B_1}\sum_{b=1}^{B_2}\sum_{l=1}^{B_1}\mathsf{TD}_{b,l}^2. \tag{4.5.32}$$

where $\mathsf{TD}_{b,l}$ is the TD loss derived by using the old experience in the sampled mini-batch of data transitions.

Without the data of the whole trajectory, the loss function for the actor NN only considers the single-step advantage function as follows:

$$L_{\text{Off}}(\boldsymbol{\varphi}) = \frac{1}{B_2}\frac{1}{B_1}\sum_{b=1}^{B_2}\sum_{l=1}^{B_1}\varpi_b \cdot \zeta_{b,l}, \tag{4.5.33}$$

where

$$\varpi_b = \frac{\pi(\tilde{\mathbf{a}}_{b,l}|\mathbf{s}_{b,l};\boldsymbol{\varphi})}{\pi(\tilde{\mathbf{a}}_{b,l}|\mathbf{s}_{b,l};\boldsymbol{\varphi}_b)} \tag{4.5.34}$$

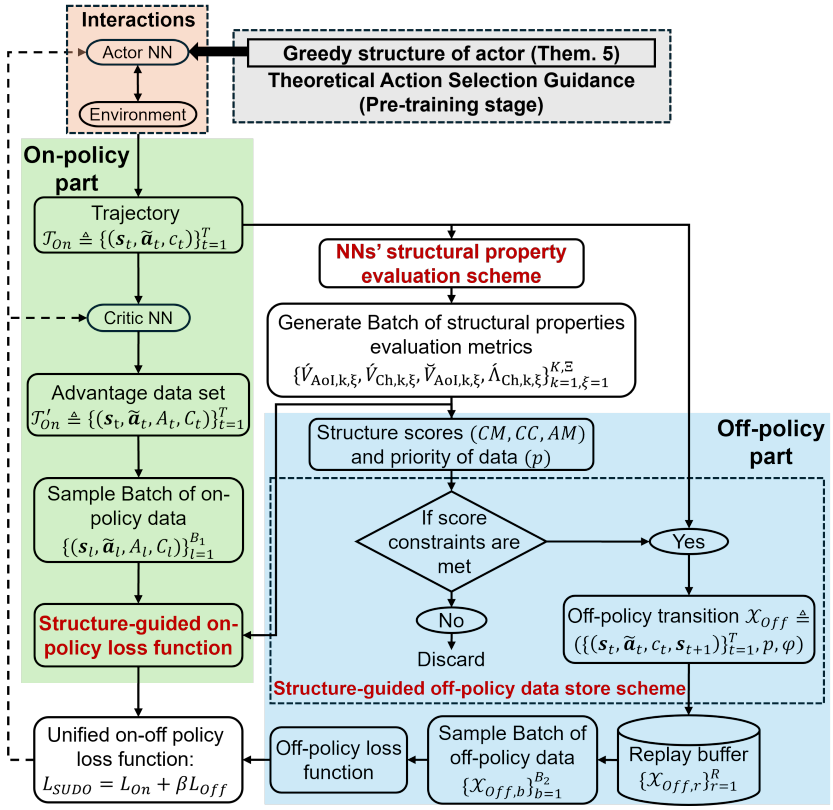


Figure 4.3: SUDO-DRL Algorithm Architecture.

is the importance sampling ratio used to correct distribution mismatch and $\zeta_{b,l} = c_{b,l} + \gamma V(\mathbf{s}_{b,l+1}; \boldsymbol{\nu}) - V(\mathbf{s}_{b,l}; \boldsymbol{\nu})$ is the single-step approximation of the advantage function.

Structure-guided action selection for pre-training

In the pre-training stage, we initialize the system with co-located devices and develop a novel SG action selection based on Theorem 4.4.4, which indicates that devices in the mandatory scheduling set should be scheduled for the transmission.

Theorem 4.4.4 indicates that the devices in the priority scheduling set should be scheduled for the transmission.

Thus, to utilize the greedy structure based on Theorem 4.4.4 during the co-located-condition training stage, we firstly derive the mandatory scheduling set $\mathcal{I}$ based on the current state and the properties of the system according to Definition 4.4.2. Then, in the action selection part, we use the stochastic policy $\pi(\tilde{\mathbf{a}}|\mathbf{s};\boldsymbol{\varphi})$ to generate the action until all devices n, that belong to both sets $\mathcal{I}$ are scheduled in the corresponding real action $\mathbf{a}_t$, i.e.,

$$a_{n,t} \neq 0, \forall n \in \mathcal{I}. \tag{4.5.35}$$

The generated real action $\mathbf{a}_t$ and virtual action $\tilde{\mathbf{a}}_t$ are selected as the executed action and stored data in the training, respectively.

The architecture and details of the SUDO-DRL algorithm are shown in Fig. 4.3 and Algorithm 3, respectively.

4.6 Numerical Experiments

In this section, we evaluate and compare the performance of the proposed SUDO-DRL with the PPO as the benchmark on-policy DRL and DDPG [56], structure-enhanced DDPG (SE-DDPG) in Chapter 2, and type II monotonicity-regularized DDPG (MRII-DDPG) in Chapter 3 as the benchmark off-policy DRL.

4.6.1 Experiment Setups

The computing platform for running the numerical experiments and the settings of the remote estimation system are the same as section 2.6.

For the fair comparison, the actor NN and critic NN of the SUDO-DRL and benchmark agents are fully connected neural networks, which have three hidden layers

Algorithm 3 SUDO-DRL for device scheduling

1: Initialize critic and policy network with random weights $\boldsymbol{\nu}$ and $\boldsymbol{\varphi}$, respectively
2: Initialize the environment with co-located devices
3: **for** episode $= 1, 2, \ldots, I_1$ **do**
4: Initialize state $\mathbf{s}_0$
5: **for** $t = 0, 1, \ldots, T$ **do**
6: Generate a trajectory $\mathcal{T}_{\mathrm{On}} \triangleq \{(\mathbf{s}_t, \tilde{\mathbf{a}}_t, c_t)\}_{t=0}^{T-1}$ by the SG action selection method and the current policy $\pi(\cdot|\cdot)$
7: **end for**
▷ **Structural Property Evaluation**
8: Calculate the structural property evaluation metrics $\acute{V}_{\mathrm{AoI},k,\xi}, \acute{V}_{\mathrm{Ch},k,\xi}, \breve{V}_{\mathrm{AoI},k,\xi}, \acute{\Lambda}_{\mathrm{Ch},k,\xi}$ according to (4.5.16), (4.5.17), (4.5.19), and (4.5.24)
9: Generate a batch of structural property evaluation metrics $\{\acute{V}_{\mathrm{AoI},k,\xi}, \acute{V}_{\mathrm{Ch},k,\xi}, \breve{V}_{\mathrm{AoI},k,\xi}, \acute{\Lambda}_{\mathrm{Ch},k,\xi}\}_{k=1,\xi=1}^{K,\Xi}$
▷ **On-policy Part**
10: **for** $t = 0, 1, \ldots, T$ **do**
11: Compute the advantage function A_t in (4.5.3) and cost-to-go function C_t in (4.5.4) and generate the data set $\mathcal{T}'_{\mathrm{On}}$ in (4.5.5)
12: **end for**
13: **for** epoch $= 1, \ldots, E_1$ **do**
14: Sample a random mini-batch of data $\{(\mathbf{s}_l, \tilde{\mathbf{a}}_l, A_l, C_l)\}_{l=1}^{B_1}$ from data set $\mathcal{T}'_{\mathrm{On}}$ of the current trajectory
15: Calculate the on-policy loss function $L_{\mathrm{On}}(\boldsymbol{\nu})$ in (4.5.25) and $L_{\mathrm{On}}(\boldsymbol{\varphi})$ in (4.5.26)
16: **end for**
▷ **Off-policy Part**
17: Calculate the structure score of the generated trajectory CM_u, CC_u, and AM_u according to (4.5.22), (4.5.23), and (4.5.24), and the priority p_u according to (4.5.30)
18: Calculate the average structure score $\mathsf{CM}_{\mathrm{Avg},u}$, $\mathsf{CC}_{\mathrm{Avg},u}$, and $\mathsf{AM}_{\mathrm{Avg},u}$ according to (4.5.27)
19: Store transition $\mathcal{X}_{\mathrm{Off},u} \triangleq (\mathcal{T}_{\mathrm{On},u}, p_u, \varphi_u)$ of the current trajectory u in $\mathcal{R}$ if it meets the constraints (4.5.28)
20: **for** epoch $= 1, \ldots, E_2$ **do**
21: Sample a batch of transitions $\{\mathcal{X}_{\mathrm{Off},b}\}_{b=1}^{B_2}$ based on sampling priority (4.5.30) from $\mathcal{R}$
22: **for** b $= 1, \ldots, B_2$ **do**
23: Sample a mini-batch of data transitions $\{(\mathbf{s}_{b,l}, \tilde{\mathbf{a}}_{b,l}, c_{b,l}, \mathbf{s}_{b,l+1}, \varphi_b)\}_{l=1}^{B_1}$ from $\mathcal{X}_{\mathrm{Off},b}$
24: Calculate the off-policy loss function $L_{\mathrm{Off}}(\boldsymbol{\nu})$ in (4.5.32) and $L_{\mathrm{Off}}(\boldsymbol{\varphi})$ in (4.5.33)
25: **end for**
26: **end for**
27: Update $\boldsymbol{\nu}$ by minimizing the loss function $L_{\mathrm{SUDO}}(\boldsymbol{\nu})$ in (4.5.14)
28: Update $\boldsymbol{\varphi}$ by minimizing the loss function $L_{\mathrm{SUDO}}(\boldsymbol{\varphi})$ in (4.5.15)
29: **end for**
30: Initialize the environment with the wireless communication model introduced in 2.2
31: Initialize replay buffer $\mathcal{R}$
32: **for** episode $= I_1, \ldots, I$ **do**
33: Repeat algorithm from line 5 to line 28 by adopting the action selection of conventional PPO algorithm as in [155]
34: **end for**

Table 4.1: Summary of Hyperparameters

Hyperparameters of SUDO-DRL and benchmarks	Value
Critic NN learning rate	0.001
Actor NN learning rate	0.0001
Decay rate of learning rate	0.001
Discount factor, γ	0.99
GAE parameter, λ	0.99
Clipping parameter, ϵ	0.2
Policy entropy loss weight, ω	0.01
Decay rate of sampling priority, ρ	0.95
Unified on-off policy loss function hyperparameter, β	0.9
Number of epochs for on-policy part, E_1	2
Number of epochs for on-policy part, $E_2 - E_1$	2
On-policy and off-policy mini-batch size, B_1	128
Off-policy trajectory batch size, B_2	4
Number of sampled states for score scheme, K	50
Number of tested AoI and channel state, N'	4
Number of past trajectories for average score computing, $\bar{u}$	50
Time horizon of each episode, T	128
Size of Replay buffer, R	200
Number of episodes for co-located-condition training stage, I_1	$10 \times N$
Number of episodes for whole training, I	10000
Optimizer during training	Adam

with the same nodes as in [82]. The input and output nodes are set according to the requirements of each algorithm as follows. For the actor NN, the input dimension is the same as the state dimension, i.e., $N + N \times M$, for all algorithms. The output dimension is $2N$ for SUDO-DRL and PPO as discussed in Section 4.5.2, and N for the benchmark off-policy DRL algorithms. In terms of the critic NN, the input dimension is $N + N \times M$ for the SUDO-DRL and PPO, and $2N + N \times M$ for the benchmark off-policy DRL, while the output dimension is 1 for all evaluated algorithms.

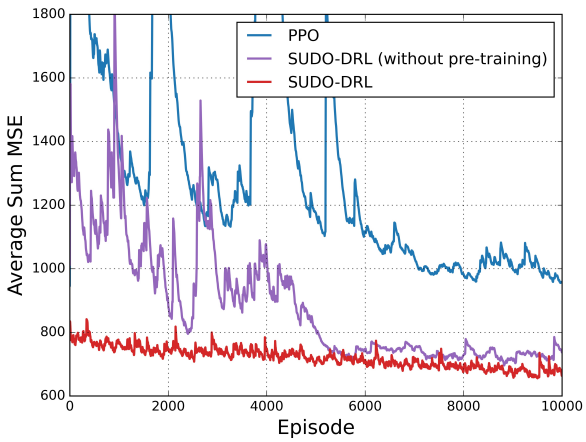


Figure 4.4: Average cost during training with $N{=}40, M{=}20$.

The setting of the training hyperparameters of the SUDO-DRL and PPO algorithm is summarized in Table 4.1 The details and hyperparameters settings of the other benchmark algorithms are shown in 2.6 and 3.6.

4.6.2 Performance Comparison of Different DRL Algorithms

Fig. 4.4 illustrates the average cost during the training of the SUDO-DRL algorithms with and without the co-located-condition training stage, and the benchmark PPO under the same system setting of a 40-device-20-channel system. It shows that the proposed SUDO-DRL reduces the average cost by about 30% compared with the conventional PPO, and the developed co-located-condition training method saves more than 40% training episodes for convergence to the same point. Also, we see that the average cost of the policy after co-located-condition training is lower than the policy without co-located-condition training, i.e., the policy at the 400th episode, which implies that the policy has reached a better point.

Table 4.2: Emperical average cost of the SUDO-DRL algorithm and the Benchmarks

System Scale (N, M)	Para.	DDPG	SE-DDPG Chapter 2	MRII-DDPG Chapter 3	PPO	**SUDO-DRL**
$(10, 5)$	1	89.26	77.14	84.00	119.63	**85.52**
	2	98.87	87.30	90.28	123.01	**95.82**
	3	–	106.60	119.55	195.37	**121.31**
	4	83.12	78.21	80.88	120.93	**80.68**
$(20, 10)$	5	–	357.14	369.14	569.96	**370.63**
	6	–	–	445.87	584.37	**426.29**
	7	–	407.20	441.73	731.94	**417.90**
	8	–	290.72	307.94	376.16	**308.91**
$(30, 15)$	9	–	–	–	805.45	**519.78**
	10	–	–	–	739.97	**575.34**
	11	–	–	–	900.71	**518.03**
	12	–	–	–	901.42	**551.97**
$(40, 20)$	13	–	–	–	1057.05	719.45
	14	–	–	–	971.35	689.81
	15	–	–	–	1291.54	994.80
	16	–	–	–	1012.20	771.91

In Table 2.2, we examine the performance of the SUDO-DRL algorithm and benchmark algorithms. The performance is evaluated based on the empirical average cost over 20000-step simulations under different system settings, including the parameters of dynamic processes and wireless communications system, i.e., $\mathbf{A}_n$, $\mathbf{C}_n$ and $q_{n,m}^1, \ldots, q_{n,m}^{\bar{h}}$, as well as different system scales, i.e., (N, M). We see that the DDPG can only work for the 10-device-5-channel system settings, and the SE-DDPG and MRII-DDPG can only converge up to the 20-device-10-channel systems, but both PPO and SUDO-DRL can work for 40-device-20-channel system scale. In particular, in 10-device-5-channel systems, the proposed SUDO-DRL reduces the average cost by 25% over the conventional PPO and achieves similar performance to the DDPG, while it can converge in the 3rd parameter setup, where the DDPG fails to find an effective scheduling policy. For the system scales that SE-DDPG and MRII-DDPG

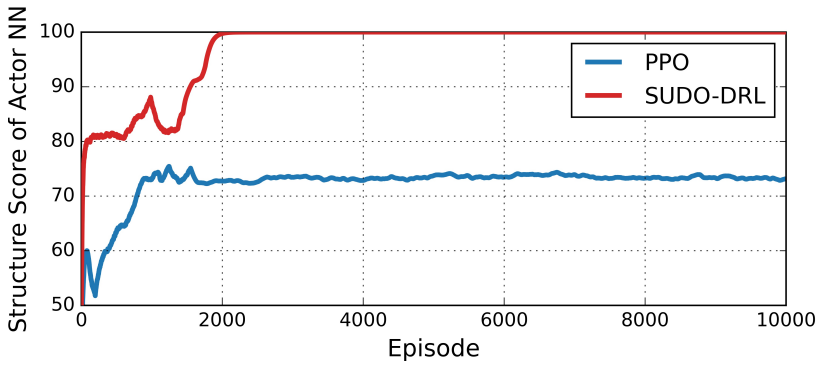


Figure 4.5: Structure score of actor NN during training with $N=40, M=20$.

can converge, the SUDO-DRL performs about 5% worse than these benchmark off-policy algorithms. In other system settings, the SUDO-DRL provides a 25% to 40% cost reduction when compared to the PPO. We can also see that the reduced costs between PPO and SUDO-DRL become higher with the increase in the system scale.

In addition, we also test the effect of the proposed score scheme and the SUDO loss function on the structure of the policy during the training of the SUDO-DRL algorithm. Fig. 4.5 and Fig. 4.6 illustrate that the SUDO-DRL improves the structure score of the actor NN and the convexity score of the critic NN about 25% compared with the PPO. In Fig. 4.7, we see that both SUDO-DRL and PPO converge to the monotonicity score of 100 while SUDO-DRL saves about 85% training episodes to converge.

4.7 Conclusion

In this section, we have formulated the optimal goal-oriented transmission scheduling problem as the MDP, and then proved that the optimal value function of the formulated MDP has the monotonicity w.r.t. the channel states as well as the discrete

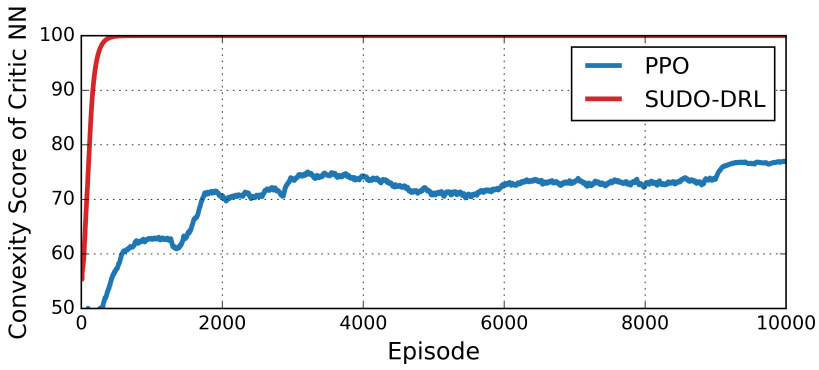


Figure 4.6: Convexity score of critic NN during training with $N{=}40, M{=}20$.

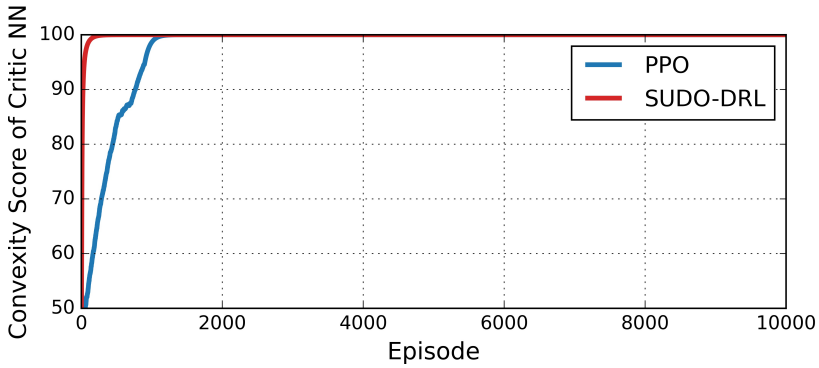


Figure 4.7: Monotonicity score of critic NN during training with $N{=}40, M{=}20$.

convexity w.r.t. the AoI states. We also have proved that the optimal policy has a greedy structure under some system settings. After that, we have developed the Structure-guided Unified (Dual) On-off policy DRL (SUDO-DRL) algorithm based on these proven structural properties for dealing with the formulated MDP. In particular, we have introduced the following novel methods to improve the exploration effectiveness and sample efficiency: a structure-guided action selection method and loss function, a structure score scheme to prioritize the generated data, and a prioritized replay buffer to store and reuse the past data. The numerical results have shown that the SUDO-DRL algorithm can improve by 25% to 45% performance while saving convergence time by about 40% compared with the benchmark on-policy DRL

algorithm. When compared with the benchmark off-policy DRL algorithms, it can achieve a similar performance while solving the scheduling problem in the large-scale system that off-policy DRL cannot converge.

Chapter 5

Conclusions and Future Work

This thesis studied different structural properties of the transmission scheduling problem in a remote state estimation system, including the threshold structure of the optimal scheduling policy, the monotonicity of the Q function, and the convexity of the optimal value function. These properties are used to improve the conventional DRL algorithms for better performance and convergence speed. This chapter summarizes the main results and insights of this thesis, followed by a discussion of some possible extensions in future work.

5.1 Summary of Results and Insights

Chapter 2 proved that the optimal scheduling policy of the remote estimation system satisfies a number of threshold properties. Then, based on these theoretical guidelines, we developed structure-enhanced deep reinforcement learning (SE-DRL) algorithms, including SE-DQN and SE-DDPG, to solve the optimal scheduling problem. In particular, we have proposed a novel action selection method and a new loss function to improve training efficiency. The developed SE action selection method regularizes the selected actions based on the action of the state with a smaller AoI.

The proposed SE loss functions are used to guide the trained policy to have the optimal policy structure by penalizing the actions that do not follow the threshold structure. Our numerical results have illustrated that SE-DRL algorithms can reduce training time by 50% and reduce the estimation mean square error (MSE) by 10% to 25%, when compared to benchmark DRL algorithms. The results also show that SE-DRL can effectively solve a range of optimal scheduling problems, not limited to remote state estimation systems.

Chapter 3 proved that the Q function of the optimal scheduling policy is monotonic with respect to the AoI states and channel states. In addition, we show that the monotonicity of the Q function also exists in the semantic-award transmission scheduling problem in cyber-physical systems (CSPs), not limited to the remote state estimation system. Then, building on this monotonicity, we have developed three different types of monotonicity-driven DRL algorithms: monotonic architecture-based DDPG (MA-DDPG), type I monotonicity-regularized DDPG (MRI-DDPG), and type II monotonicity-regularized DDPG (MRII-DDPG). The MA-DDPG algorithm guarantees the monotonicity of the Q function by using a three-layer monotonic NN with unique architecture, which requires the activation functions to be non-decreasing and the product of NN's weights to be no-positive. The MRI-DDPG and MRII-DDPG obtain the monotonic critic NN by introducing a positive derivative penalty and positive increment penalty in the loss function, respectively. The developed monotonicity-driven DRL algorithms solve the scheduling problems effectively and improve performance by about 30% while reducing training episodes by about 50

Chapter 4 formulated the optimal goal-oriented transmission scheduling problem as the MDP and then proved that the optimal value function of the formulated MDP

has the monotonicity with respect to the channel states as well as the discrete convexity with respect to the AoI states. We have also proved that the optimal policy has a greedy structure under some system settings. We then developed the Structure-guided Unified (Dual) On-off policy DRL (SUDO-DRL) algorithm based on these proven structural properties for dealing with the formulated MDP. In particular, we have introduced the following novel methods to improve the exploration effectiveness and sample efficiency: a structure-guided action selection method and loss function, a structure score scheme to prioritize the generated data, and a prioritized replay buffer to store and reuse the past data. The numerical results have shown that the SUDO-DRL algorithm can improve performance by 25% to 45% while reducing convergence time by about 40% compared to the benchmark on-policy DRL algorithm. When compared to the benchmark off-policy DRL algorithms, it can achieve a similar performance while solving the scheduling problem in the large-scale system that off-policy DRL cannot converge.

The above results are not only beneficial to the communications research community but also offer several important insights for the control community. First, the proven threshold-based scheduling policies could help control researchers design more efficient event-triggered control systems. Next, the proven monotonicity of the Q function could inform the design of control laws in networked systems by establishing a clear relationship between control actions and system states. Moreover, as the proposed novel DRL algorithms provide high-quality remote state estimation, these algorithms enable the remote controller to generate better control signals, which is necessary for the overall stability of the networked control system.

5.2 Future Work

We now propose some possible extensions of our problems that could potentially be investigated in future work.

Future work could explore NOMA-based resource allocation problems, moving beyond the current focus on sensor scheduling problems, which assume that each sensor can be scheduled to at most one channel and each channel is assigned to only one sensor. In this case, the action space becomes hybrid and large, i.e., the discrete sensor scheduling and continuous power allocation, which makes the problem more difficult to solve. More structural properties of the optimal resource allocation policies can be investigated, such as the monotonicity of the allocated power with respect to the AoI states and channel states, and then these properties can be used to develop tailored DRL algorithms. To solve the resource allocation problems, we can also consider the graph neural network (GNN) and zero-shot learning, which are powerful deep learning methods.

In addition, hardware implementation and testing of the proposed algorithms in the wireless networked control systems would provide valuable practical validation, as the experiments in this thesis were conducted in simulation environments. Key implementation challenges to investigate include real-time processing constraints, clock synchronization between distributed nodes, system latency measurement and optimization, and reliable wireless communication in noisy environments. The hardware validation could focus on several aspects including measuring actual computation times for the DRL algorithms on embedded platforms, evaluating control performance under real wireless channel conditions, and esting the system's robustness to interference and packet losses.Such hardware implementations would provide crucial

insights into scaling these systems for larger industrial deployments.

For the control community, these results also highlight some challenges, particularly in terms of guaranteeing stability and performance bounds when using learning-based scheduling policies in feedback control loops. The DRL-based solutions, while effective, may need additional theoretical analysis to provide the rigorous stability guarantees typically required in control theory.

In terms of the system model design, various scenarios can be investigated. First, we should consider a more general communication system with Markov fading channels rather than the i.i.d. fading channels since the channel states between two consecutive time steps are usually not independent. Next, the control loop can also be included in future work on the channel assignment or resource allocation problem so that the derived policy needs to consider the control performance. Moreover, human-centered systems (e.g., Metaverse and extended reality) have not been considered. With the development of intelligent edge devices, the system is required to transmit not only sensing information but also information from humans, which is difficult to model mathematically. The large language models (LLMs) developed in recent years can be an effective tool to solve such problems. How to schedule the information generated by an LLM is an open challenge in future work.

Furthermore, some more advanced DRL algorithms based on structural properties with lower computation complexity and better performance can be developed. For example, an SE action selection with lower computation complexity can save much training time, especially for the high-dimensional state and action space. In addition, a more innovative approach to merging the on-policy and off-policy methods, which enables the trained policy to perform better than the off-policy DRL while solving the

large-scale scheduling problem as on-policy DRL, can be investigated in the future.

Appendix A

Proofs for Chapter 2. Structure-Enhanced DRL for Optimal Transmission Scheduling

A.1 Proof of Lemma 2.4.2

Similar to the Q-value function, we define a W-function for the value function, $W(\mathbf{s}, \mathbf{a}, V^{\tilde{t}}) : \mathcal{S} \times \mathcal{A} \times \mathcal{V} \to \mathbb{R}$

$$W(\mathbf{s}, \mathbf{a}, V^{\tilde{t}}) = r(\mathbf{s}) + \gamma \sum_{\mathbf{s}^+} \Pr(\mathbf{s}^+|\mathbf{s}, \mathbf{a}) V^{\tilde{t}}(\mathbf{s}^+). \tag{A.1.1}$$

According to Lemma 2.4.1, the converge of the optimal value function, $V^*(\mathbf{s})$, is independent of the initial value function $V^0(\mathbf{s})$. Therefore, given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ and $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, where $\tau'_i \geq \tau_i$, we can assume that $V^0(\mathbf{s})$ is monotonic, i.e.

$$V^0(\mathbf{s}') \leq V^0(\mathbf{s}). \tag{A.1.2}$$

To prove Lemma 2.4.2 based on (A.1.2) and Lemma 2.4.1, it is sufficient to show that the value function $V^1(\mathbf{s})$ is monotonic, i.e.

$$V^1(\mathbf{s}') \leq V^1(\mathbf{s}), \tag{A.1.3}$$

and then the monotonicity of the value function can be preserved by the Bellman operator $\mathsf{B}[\cdot]$ from $V^0(\mathbf{s})$ to $V^*(\mathbf{s})$. During the value iteration, we define the optimal action and the optimal policy at $\tilde{t}$th iteration as

$$\mathbf{a}^{\tilde{t}} \triangleq \pi^{\tilde{t}}(\mathbf{s}) = r(\mathbf{s}) + \gamma \arg\max_{\mathbf{a}\in\mathcal{A}} \left[\sum_{\mathbf{s}^+} \Pr(\mathbf{s}^+|\mathbf{s},\mathbf{a}) V^{\tilde{t}-1}(\mathbf{s}^+) \right]. \tag{A.1.4}$$

From (2.4.6) and (A.1.4), it directly follows the relationship between the value function and the W-functions:

$$V^{\tilde{t}+1}(\mathbf{s}) = W(\mathbf{s}, \mathbf{a}^{\tilde{t}}, V^{\tilde{t}}) \geq W(\mathbf{s}, \mathbf{a}, V^{\tilde{t}}). \tag{A.1.5}$$

As we consider i.i.d. fading channel states, the transition probability of the channel state $\mathbf{H}^+$ is independent of the number of iterations, the current state, and the action, which implies that the channel states are constant. Therefore, for the convenience of writing, we write the state as $\mathbf{s} = \boldsymbol{\tau}$ and $\mathbf{s}' = \boldsymbol{\tau}'$.

To prove (A.1.3), we use the optimal action of the state $\boldsymbol{\tau}'$ is $\mathbf{a}^1 = \pi^1\left(\boldsymbol{\tau}'_{(i)}\right)$ to derive the following inequality,

$$\begin{aligned}
&V^1\left(\boldsymbol{\tau}'_{(i)}\right) - V^1(\boldsymbol{\tau}) \\
&\leq W(\boldsymbol{\tau}'_{(i)}, \mathbf{a}^1, V^0) - W(\boldsymbol{\tau}, \mathbf{a}^1, V^0) \\
&= [r(\boldsymbol{\tau}') - r(\boldsymbol{\tau})] + \gamma \Bigg[\sum_{\boldsymbol{\tau}_{\setminus i}{}^+} \sum_{\tau_i'^+} \Pr(\boldsymbol{\tau}_{\setminus i}{}^+|\boldsymbol{\tau}_{\setminus i}, \mathbf{H}_{\setminus i}, \mathbf{a}^1_{\setminus i}) \Pr(\tau_i'^+|\tau_i', \mathbf{h}_i, a_i^1) V^0(\boldsymbol{\tau}'^+) \\
&\quad - \sum_{\boldsymbol{\tau}_{\setminus i}{}^+} \sum_{\tau_i{}^+} \Pr(\boldsymbol{\tau}_{\setminus i}{}^+|\boldsymbol{\tau}_{\setminus i}, \mathbf{H}_{\setminus i}, \mathbf{a}^1_{\setminus i}) \Pr(\tau_i{}^+|\tau_i, \mathbf{h}_i, a_i^1) V^0(\boldsymbol{\tau}^+) \Bigg] \\
&\leq 0,
\end{aligned} \tag{A.1.6}$$

where the first inequality is based on (A.1.5), and the first equality is based on (A.1.1),

and the last inequality is based on (A.1.2), $r(\boldsymbol{\tau}') < r(\boldsymbol{\tau})$, and the following inequality,

$$\Pr(\tau_i'^{+} = \tau_i' + 1 | \tau_i', \mathbf{h}_i, a_i^1) V^0(\tau_i' + 1, \boldsymbol{\tau}_{\setminus i}{}^{+}) \leq \Pr(\tau_i{}^{+} = \tau_i + 1 | \tau_i, \mathbf{h}_i, a_i^1) V^0(\tau_i + 1, \boldsymbol{\tau}_{\setminus i}{}^{+}) \tag{A.1.7}$$

$$\Pr(\tau_i'^{+} = 1 | \tau_i', \mathbf{h}_i, a_i^1) V^0(1, \boldsymbol{\tau}_{\setminus i}{}^{+}) = \Pr(\tau_i{}^{+} = 1 | \tau_i, \mathbf{h}_i, a_i^1) V^0(1, \boldsymbol{\tau}_{\setminus i}{}^{+}) \tag{A.1.8}$$

achieved by (A.1.2), $\Pr(\tau_i'^{+} = \tau_i' + 1 | \tau_i', \mathbf{h}_i, a_i^1) = \Pr(\tau_i{}^{+} = \tau_i + 1 | \tau_i, \mathbf{h}_i, a_i^1)$, and $\Pr(\tau_i'^{+} = 1 | \tau_i', \mathbf{h}_i, a_i^1) = \Pr(\tau_i{}^{+} = 1 | \tau_i, \mathbf{h}_i, a_i^1)$.

Thus, the monotonicity of the value function $V^0(\mathbf{s})$ propagates through the Bellman operator $\mathsf{B}[\cdot]$ to the optimal value function $V^*(\mathbf{s})$.

A.2 Proof of Lemma 2.4.3

In this proof, we also need the submodularity of the value function. Therefore, similar to the proof of Lemma 2.4.2, we assume that the initial value function $V^0(\mathbf{s})$ is submodular and probabilistic supermodular. Thus, given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ and $\mathbf{s}^\circ = (\boldsymbol{\tau}^\circ, \mathbf{H})$ where $\boldsymbol{\tau}^\circ = (\tau_i'', \tau_j')$ with $\tau_i'' \leq \tau_i$ and $\tau_j' \geq \tau_j$, we obtain that

$$V^0(\mathbf{s} \vee \mathbf{s}^\circ) + V^0(\mathbf{s} \wedge \mathbf{s}^\circ) \leq V^0(\mathbf{s}) + V^0(\mathbf{s}^\circ), \tag{A.2.1a}$$

$$p_{j,1} V^0(\mathbf{s} \wedge \mathbf{s}^\circ) + (p_{j,1} - p_{i,1}) V^0(\mathbf{s} \vee \mathbf{s}^\circ) \geq p_{j,1} V^0(\mathbf{s}) + (p_{j,1} - p_{i,1}) V^0(\mathbf{s}^\circ). \tag{A.2.1b}$$

To prove Lemma 2.4.3 based on (A.2.1a), (A.2.1b) and Lemma 2.4.1, it is sufficient to show that the value function $V^1(\mathbf{s})$ is submodular and probabilistic supermodular, i.e.

$$V^1(\mathbf{s} \vee \mathbf{s}^\circ) + V^1(\mathbf{s} \wedge \mathbf{s}^\circ) \leq V^1(\mathbf{s}) + V^1(\mathbf{s}^\circ), \tag{A.2.2a}$$

$$p_{j,1} V^1(\mathbf{s} \wedge \mathbf{s}^\circ) + (p_{j,1} - p_{i,1}) V^1(\mathbf{s} \vee \mathbf{s}^\circ) \geq p_{j,1} V^1(\mathbf{s}) + (p_{j,1} - p_{i,1}) V^1(\mathbf{s}^\circ), \tag{A.2.2b}$$

and then these properties of the value function can be preserved by the Bellman operator $\mathsf{B}[\cdot]$ from $V^0(\mathbf{s})$ to $V^*(\mathbf{s})$. Similar to the proof of Lemma 2.4.2, we write the

states as $\mathbf{s} = \boldsymbol{\tau}$ and $\mathbf{s}^\circ = \boldsymbol{\tau}^\circ$, and then $\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ = (\tau_i, \tau_j')$ and $\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ = (\tau_i'', \tau_j)$. In terms of the policy, since we only consider the system with a single channel in this proof, we write the optimal policy as $\pi^{\tilde{t}}(\boldsymbol{\tau}) = i$ to represent that sensor i is scheduled for the state $\boldsymbol{\tau}$, i.e. $a_i^{\tilde{t}} = 1$.

In the following, we prove (A.2.2a) by cases **(a)** and **(b)**, and (A.2.2b) by cases **(a')** and **(b')** with different packet success rates.

(a) If $p_{j,1} \leq p_{i,1}$, then there are four cases with different optimal actions of states:

(a.1) $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = i$, (a.2) $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = j$, (a.3) $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = i, \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = j$, (a.4) $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = j, \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = i$.

(a.1) If $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = i$, then

$$
\begin{aligned}
&V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) + V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - V^1(\boldsymbol{\tau}) - V^1(\boldsymbol{\tau}^\circ) \\
&\leq W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, i, V^0) + W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, i, V^0) - W(\boldsymbol{\tau}, i, V^0) - W(\boldsymbol{\tau}^\circ, i, V^0) \\
&= \gamma p_{i,1} \left[V^0(1, \tau_j' + 1) + V^0(1, \tau_j + 1) - V^0(1, \tau_j + 1) - V^0(1, \tau_j' + 1)\right] \\
&\quad + \gamma(1 - p_{i,1}) \big[V^0(\tau_i + 1, \tau_j' + 1) + V^0(\tau_i'' + 1, \tau_j + 1) \\
&\quad - V^0(\tau_i + 1, \tau_j + 1) - V^0(\tau_i'' + 1, \tau_j' + 1)\big] \\
&\leq 0, \qquad \text{(A.2.3)}
\end{aligned}
$$

where the last inequality is based on (A.2.1a).

(a.2) If $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = j$, the proof is similar to the case (a.1) by showing $W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, j, V^0) + W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, j, V^0) - W(\boldsymbol{\tau}, j, V^0) - W(\boldsymbol{\tau}^\circ, j, V^0) \leq 0$.

(a.3) If $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = i, \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = j$, then

$$
\begin{aligned}
&V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) + V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - V^1(\boldsymbol{\tau}) - V^1(\boldsymbol{\tau}^\circ) \\
&\le W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, i, V^0) + W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, j, V^0) - W(\boldsymbol{\tau}, i, V^0) - W(\boldsymbol{\tau}^\circ, j, V^0) \\
&= \gamma\big[p_{i,1} V^0(1, \tau_j' + 1) + (1 - p_{i,1}) V^0(\tau_i + 1, \tau_j' + 1) \\
&\quad + p_{j,1} V^0(\tau_i'' + 1, 1) + (1 - p_{j,1}) V^0(\tau_i'' + 1, \tau_j + 1) \\
&\quad - p_{i,1} V^0(1, \tau_j + 1) - (1 - p_{i,1}) V^0(\tau_i + 1, \tau_j + 1) \\
&\quad - p_{j,1} V^0(\tau_i'' + 1, 1) - (1 - p_{j,1}) V^0(\tau_i'' + 1, \tau_j' + 1)\big] \\
&= \gamma\big[p_{i,1} V^0(1, \tau_j' + 1) + (p_{i,1} - p_{j,1}) V^0(\tau_i'' + 1, \tau_j + 1) \\
&\quad - p_{i,1} V^0(1, \tau_j + 1) - (p_{i,1} - p_{j,1}) V^0(\tau_i'' + 1, \tau_j' + 1)\big] \\
&\quad + \gamma(1 - p_{i,1})\big[V^0\big(\tau_i + 1, \tau_j' + 1\big) + V^0(\tau_i'' + 1, \tau_j + 1) \\
&\quad - V^0(\tau_i + 1, \tau_j + 1) - V^0(\tau_i'' + 1, \tau_j' + 1)\big] \\
&\le 0, \qquad \text{(A.2.4)}
\end{aligned}
$$

where the last inequality is based on (A.2.1a) and (A.2.1b).

(a.4) If $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = j, \pi^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) = i$, then

$$
\begin{aligned}
&V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) + V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - V^1(\boldsymbol{\tau}) - V^1(\boldsymbol{\tau}^\circ) \\
&\le W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, j, V^0) + W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, i, V^0) - W(\boldsymbol{\tau}, i, V^0) - W(\boldsymbol{\tau}^\circ, j, V^0) \\
&= \gamma p_{j,1}[V^0(\tau_i + 1, 1) - V^0(\tau_i'' + 1, 1)] + \gamma(1 - p_{j,1})[V^0(\tau_i + 1, \tau_j' + 1) - V^0(\tau_i'' + 1, \tau_j' + 1)] \\
&\quad + \gamma(1 - p_{i,1})[V^0(\tau_i'' + 1, \tau_j + 1) - V^0(\tau_i + 1, \tau_j + 1)] \\
&\le \gamma(1 - p_{i,1})[V^0(\tau_i + 1, \tau_j' + 1) + V^0(\tau_i + 1, \tau_j + 1) - V^0(\tau_i'' + 1, \tau_j' + 1) - V^0(\tau_i + 1, \tau_j + 1)] \\
&\le 0, \qquad \text{(A.2.5)}
\end{aligned}
$$

where the second inequality is based on Lemma 2.4.2 and the following inequality

$$p_{j,1}\left[V^0(\tau_i+1,1)+V^0(\tau_i''+1,\tau_j'+1)-V^0(\tau_i''+1,1)-V^0(\tau_i+1,\tau_j'+1)\right]$$
$$\leq p_{i,1}\left[V^0(\tau_i+1,1)+V^0(\tau_i''+1,\tau_j'+1)-V^0(\tau_i''+1,1)-V^0(\tau_i+1,\tau_j'+1)\right], \quad \text{(A.2.6)}$$

and the last inequality is base on (A.2.1a).

(b) If $p_{j,1} \geq p_{i,1}$, the proof is similar to the case **(a)**.

Similarly, we prove (A.2.2b) by different cases based on the packet success rates.

(a') If $p_{j,1} \leq p_{i,1}$, then

$$\begin{aligned} &p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1}-p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau}\wedge\boldsymbol{\tau}^\circ) - (p_{j,1}-p_{i,1})V^1(\boldsymbol{\tau}\vee\boldsymbol{\tau}^\circ) \\ &= p_{j,1}\left[V^1(\tau_i+1,\tau_j+1) - V^1(\tau_i''+1,\tau_j+1)\right] \\ &\quad + (p_{i,1}-p_{j,1})\left[V^1(\tau_i+1,\tau_j'+1) - V^1(\tau_i''+1,\tau_j'+1)\right], \\ &\leq 0, \end{aligned} \quad \text{(A.2.7)}$$

where the last inequality is based on Lemma 2.4.2. Thus, $V^1(\mathbf{s})$ is probabilistic supermodular for states with $p_{j,1} \leq p_{i,1}$, and the optimal policy of $V^1(\mathbf{s})$ has property (ii) for the sensor with a higher packet success rate as Theorem 2.4.2 is directly following the probabilistic supermodularity of the value function.

(b') If $p_{j,1} \geq p_{i,1}$, then there are four cases based on the optimal action of states:

(b'.1) $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = i$, (b'.2) $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = j$, (b'.3) $\pi^1(\boldsymbol{\tau}) = i, \pi^1(\boldsymbol{\tau}^\circ) = j$, (b'.4) $\pi^1(\boldsymbol{\tau}) = j, \pi^1(\boldsymbol{\tau}^\circ) = i$.

(b'.1) If $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = i$, then

$$
\begin{aligned}
& p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\
& \le p_{j,1}W(\boldsymbol{\tau}, i, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, i, V^0) \\
& \quad - p_{j,1}W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, i, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, i, V^0) \\
& = p_{i,1}\left[r(\tau_i, \tau_j') - r(\tau_i'', \tau_j')\right] + \gamma p_{j,1}\left[p_{i,1}V^0(1, \tau_j + 1) + (1 - p_{i,1})V^0(\tau_i + 1, \tau_j + 1)\right] \\
& \quad + \gamma(p_{j,1} - p_{i,1})\left[p_{i,1}V^0(1, \tau_j' + 1) + (1 - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1)\right] \\
& \quad - \gamma p_{j,1}\left[p_{i,1}V^0(1, \tau_j + 1) + (1 - p_{i,1})V^0(\tau_i'' + 1, \tau_j + 1)\right] \\
& \quad - \gamma(p_{j,1} - p_{i,1})\left[p_{i,1}V^0(1, \tau_j' + 1) + (1 - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1)\right] \\
& \le \gamma(1 - p_{i,1})\big[p_{j,1}V^0(\tau_i + 1, \tau_j + 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1) \\
& \quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1)\big] \\
& \le 0, \qquad \text{(A.2.8)}
\end{aligned}
$$

where the second and last inequality is based on (A.2.1b)

(b'.2) If $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = j$, the proof is similar to the case (b'.1) by showing $p_{j,1}W(\boldsymbol{\tau}, j, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, j, V^0) - p_{j,1}W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, j, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, j, V^0) \le 0$.

(b'.3) If $\pi^1(\boldsymbol{\tau}) = i, \pi^1(\boldsymbol{\tau}^\circ) = j$, then

$$
\begin{aligned}
&p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\
&\leq p_{j,1}W(\boldsymbol{\tau}, i, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, j, V^0) \\
&\quad - p_{j,1}W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, i, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, j, V^0) \\
&\leq \gamma p_{j,1}\big[p_{j,1}V^0(\tau_i'' + 1, 1) + (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1) \\
&\quad - p_{j,1}V^0(\tau_i + 1, 1) - (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1)\big] \\
&\quad + \gamma p_{i,1}p_{j,1}\big[V^0(\tau_i + 1, 1) + V^0(\tau_i'' + 1, \tau_j + 1) \\
&\quad - V^0(\tau_i'' + 1, 1) - V^0(\tau_i + 1, \tau_j + 1)\big] \\
&\quad + \gamma[p_{j,1}V^0(\tau_i + 1, \tau_j + 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1) \\
&\quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1)] \\
&\leq \gamma\left[p_{j,1} - p_{j,1}p_{i,1}\right]\left[V^0(\tau_i'' + 1, 1) + V^0(\tau_i + 1, \tau_j + 1)\right. \\
&\quad \left. - V^0(\tau_i + 1, 1) - V^0(\tau_i'' + 1, \tau_j + 1)\right] \\
&\leq 0, \qquad \text{(A.2.9)}
\end{aligned}
$$

where the second inequality is based on (A.2.1b), and the third inequality is based on (A.2.1a) and the following inequality

$$
\begin{aligned}
&p_{j,1}[p_{j,1}V^0(\tau_i'' + 1, 1) + (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1) \\
&\quad - p_{j,1}V^0(\tau_i + 1, 1) - (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1)] \\
&< [p_{j,1}V^0(\tau_i'' + 1, 1) + (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1) \\
&\quad - p_{j,1}V^0(\tau_i + 1, 1) - (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1)], \qquad \text{(A.2.10)}
\end{aligned}
$$

and the last inequality is based on (A.2.1a) and $p_{j,1} - p_{j,1}p_{i,1} > 0$.

(b'.4) If $\pi^1(\boldsymbol{\tau}) = j, \pi^1(\boldsymbol{\tau}^\circ) = i$, then according to case **(a')** and $p_{j,1} \geq p_{i,1}$, we obtain that $\pi^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) = j$, which implies that

$$
W\left(\left(\tau_i, \tau_j'\right), j, V^0\right) \geq W\left(\left(\tau_i, \tau_j'\right), i, V^0\right), \qquad \text{(A.2.11)}
$$

based on (A.1.5). Using (A.1.1), then (A.2.11) is equal to

$$\begin{aligned} &p_{j,1}V^0(\tau_i+1,1)+(1-p_{j,1})V^0(\tau_i+1,\tau'_j+1)\\ &\geq p_{i,1}V^0(1,\tau'_j+1)+(1-p_{i,1})V^0(\tau_i+1,\tau'_j+1), \end{aligned} \tag{A.2.12}$$

which is exactly

$$p_{j,1}V^0(\tau_i+1,1)+(p_{i,1}-p_{j,1})V^0(\tau_i+1,\tau'_j+1)\geq p_{i,1}V^0(1,\tau'_j+1). \tag{A.2.13}$$

Similarly, we derive the following inequality based on (A.1.5), (A.1.1) and $\pi^1(\boldsymbol{\tau}^\circ)=i$,

$$\begin{aligned} &p_{i,1}V^0(1,\tau'_j+1)+(1-p_{i,1})V^0(\tau''_i+1,\tau'_j+1)\\ &\geq p_{j,1}V^0(\tau''_i+1,1)+(1-p_{j,1})V^0(\tau''_i+1,\tau'_j+1) \end{aligned} \tag{A.2.14}$$

which is exactly

$$p_{i,1}V^0(1,\tau'_j+1)\geq p_{j,1}V^0(\tau''_i+1,1)+(p_{i,1}-p_{j,1})V^0(\tau''_i+1,\tau'_j+1). \tag{A.2.15}$$

Then, we use the (A.2.13) and (A.2.15) to derive that

$$\begin{aligned} &p_{j,1}V^0(\tau_i+1,1)+(p_{i,1}-p_{j,1})V^0(\tau_i+1,\tau'_j+1)\\ &\geq p_{j,1}V^0(\tau''_i+1,1)+(p_{i,1}-p_{j,1})V^0(\tau''_i+1,\tau'_j+1), \end{aligned} \tag{A.2.16}$$

and thus

$$\begin{aligned} &p_{j,1}V^0(\tau_i+1,1)+(p_{j,1}-p_{i,1})V^0(\tau''_i+1,\tau'_j+1)\\ &\geq p_{j,1}V^0(\tau''_i+1,1)+(p_{j,1}-p_{i,1})V^0(\tau_i+1,\tau'_j+1), \end{aligned} \tag{A.2.17}$$

which is opposite to the (A.2.1b). Thus, this case cannot exist at the 1st iteration.

Thus, the submodularity and the probabilistic supermodularity of the value function $V^0(\mathbf{s})$ propagates through the Bellman operator $\mathsf{B}[\cdot]$ to the optimal value function $V^*(\mathbf{s})$.

A.3 Proof of Lemma 2.4.4

Similar to the proof of Lemma 2.4.2, we assume that the initial value function $V^0(\mathbf{s})$ has the following property given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ and $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$ where $\tau'_i \gg \tau_i$,

$$V^0(\mathbf{s}') \ll V^0(\mathbf{s}). \tag{A.3.1}$$

To prove Lemma 2.4.4 based on (A.3.1) and Lemma 2.4.1, it is sufficient to show that the value function $V^1(\mathbf{s})$ has the same property, i.e.

$$V^1(\mathbf{s}') \ll V^1(\mathbf{s}). \tag{A.3.2}$$

Similar to the proof of Lemma 2.4.3, we write the states as $\mathbf{s} = \boldsymbol{\tau}$ and $\mathbf{s}' = \boldsymbol{\tau}'_{(i)}$, and the action as $\pi^1(\mathbf{s}) = i$ to represent $a_i^1 = 1$ for the state $\mathbf{s}$. Moreover, we drop the constant AoI states in different optimal value functions. For example, given states $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_i, \ldots, \tau_N)$ and $\boldsymbol{\tau}' = (\tau_1, \ldots, \tau'_i, \ldots, \tau_N)$, we write them as $\boldsymbol{\tau} = (\tau_i)$ and $\boldsymbol{\tau}' = (\tau'_i)$. In the following, we prove (A.3.2) by cases based on the optimal action of states.

(a) If $\pi^1(\boldsymbol{\tau}') = i$, then

$$V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \tag{A.3.3}$$

$$\leq W\left(\boldsymbol{\tau}', i, V^0\right) - W\left(\boldsymbol{\tau}, i, V^0\right) \tag{A.3.4}$$

$$= p_{i,1}\left[r(\tau'_i, \tau_j) - r(\tau_i, \tau_j)\right] \tag{A.3.5}$$

$$+\gamma\left[p_{i,1}V^0(1, \tau_j{+}1) + (1{-}p_{i,1})V^0(\tau'_i{+}1, \tau_j{+}1) - p_{i,1}V^0(1, \tau_j{+}1) - (1{-}p_{i,1})V^0(\tau_i{+}1, \tau_j{+}1)\right] \tag{A.3.6}$$

$$\ll 0, \tag{A.3.7}$$

where the last inequality is base on (A.3.1).

(b) If $\pi^1(\boldsymbol{\tau}') = j$, then the proof is similar to the case (a) by showing $W(\boldsymbol{\tau}', j, V^0) - W(\boldsymbol{\tau}, j, V^0) \ll 0$.

A.4 Proof of Lemma 2.4.5

Similar to the proof of Lemma 2.4.2, we assume that the initial value function $V^0(\mathbf{s})$ is probabilistic supermodular given states $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ and $\mathbf{s}^\circ = (\boldsymbol{\tau}^\circ, \mathbf{H})$, where $\boldsymbol{\tau}^\circ = (\tau_i'', \tau_j')$ with $\tau_i'' \ll \tau_i$ and $\tau_j' \geq \tau_j$,

$$p_{j,1}V^0(\mathbf{s} \wedge \mathbf{s}^\circ) + (p_{j,1} - p_{i,1})V^0(\mathbf{s} \vee \mathbf{s}^\circ) \geq p_{j,1}V^0(\mathbf{s}) + (p_{j,1} - p_{i,1})V^0(\mathbf{s}^\circ). \quad \text{(A.4.1)}$$

To prove Lemma 2.4.5 based on (A.4.1) and Lemma 2.4.1, it is sufficient to show that the value function $V^1(\mathbf{s})$ is probabilistic supermodular, i.e.

$$p_{j,1}V^1(\mathbf{s} \wedge \mathbf{s}^\circ) + (p_{j,1} - p_{i,1})V^1(\mathbf{s} \vee \mathbf{s}^\circ) \geq p_{j,1}V^1(\mathbf{s}) + (p_{j,1} - p_{i,1})V^1(\mathbf{s}^\circ). \quad \text{(A.4.2)}$$

Similar to the proof of Lemma 2.4.4, we write the states as $\mathbf{s} = \boldsymbol{\tau} = (\tau_i, \tau_j)$ and $\mathbf{s}^\circ = \boldsymbol{\tau}^\circ = (\tau_i'', \tau_j')$, and the action as $\pi^1(\mathbf{s}) = i$ to represent $a_i^1 = 1$, then $\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ = (\tau_i, \tau_j')$ and $\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ = (\tau_i'', \tau_j)$. In the following, we prove (A.4.2) by case **(a)** and **(b)** based on the packet success rates.

Then, we prove (A.4.2) by cases.

(a) If $p_{j,1} \leq p_{i,1}$, then

$$\begin{aligned} &p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\ &= p_{j,1}[V^1(\boldsymbol{\tau}) - V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ)] + (p_{i,1} - p_{j,1})[V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) - V^1(\boldsymbol{\tau}^\circ)] \\ &\leq 0, \end{aligned} \quad \text{(A.4.3)}$$

where the last inequality is derived based on Lemma 2.4.2.

(b) If $p_{j,1} \geq p_{i,1}$, then there are ten cases based on the optimal action of states.

(b.1) $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = i$, (b.2) $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = j$, (b.3) $\pi^1(\boldsymbol{\tau}) = i, \pi^1(\boldsymbol{\tau}^\circ) = j$, (b.4) $\pi^1(\boldsymbol{\tau}) = j, \pi^1(\boldsymbol{\tau}^\circ) = i$, (b.5) $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = k, (k \neq i, j)$ (b.6) $\pi^1(\boldsymbol{\tau}) = i, \pi^1(\boldsymbol{\tau}^\circ) = k$, (b.7) $\pi^1(\boldsymbol{\tau}) = k, \pi^1(\boldsymbol{\tau}^\circ) = i$, (b.8) $\pi^1(\boldsymbol{\tau}) = j, \pi^1(\boldsymbol{\tau}^\circ) = k$, (b.9) $\pi^1(\boldsymbol{\tau}) = k, \pi^1(\boldsymbol{\tau}^\circ) = j$, (b.10) $\pi^1(\boldsymbol{\tau}) = k_1, \pi^1(\boldsymbol{\tau}^\circ) = k_2$.

For cases (b.1), (b.2), (b.3), and (b.4), the proof is the same as the proof of Lemma 2.4.3, because the AoI state of sensors except for the sensor i and sensor j are constant during the proof in these cases. Therefore, we only prove the other cases in the following.

(b.5) If $\pi^1(\boldsymbol{\tau}) = \pi^1(\boldsymbol{\tau}^\circ) = k, (k \neq i, j)$, then

$$
\begin{aligned}
&p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\
&\leq p_{j,1}W(\boldsymbol{\tau}, k, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, k, V^0) \\
&\quad - p_{j,1}W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ\, k, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, k, V^0) \\
&= p_{i,1}[r(\mathbf{s} \vee \mathbf{s}^\circ) - r(\mathbf{s})] + \gamma p_{j,1}\left[p_{k,1}V^0(\tau_i+1, \tau_j+1, 1) + (1-p_{k,1})V^0(\tau_i+1, \tau_j+1, \tau_k+1)\right] \\
&\quad + \gamma(p_{j,1} - p_{i,1})[p_{k,1}V^0(\tau_i'' + 1, \tau_j' + 1, 1) + (1 - p_{k,1})V^0(\tau_i'' + 1, \tau_j' + 1, \tau_k + 1)] \\
&\quad - \gamma p_{j,1}[p_{k,1}V^0(\tau_i'' + 1, \tau_j + 1, 1) + (1 - p_{k,1})V^0(\tau_i'' + 1, \tau_j + 1, \tau_k + 1)] \\
&\quad - \gamma(p_{j,1} - p_{i,1})[p_{k,1}V^0(\tau_i + 1, \tau_j' + 1, 1) + (1 - p_{k,1})V^0(\tau_i + 1, \tau_j' + 1, \tau_k + 1)] \\
&\leq \gamma p_{k,1}[p_{j,1}V^0(\tau_i + 1, \tau_j + 1, 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1, 1) \\
&\quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1, 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1, 1)] \\
&\quad \gamma(1 - p_{k,1})[p_{j,1}V^0(\tau_i + 1, \tau_j + 1, \tau_k + 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1, \tau_k + 1) \\
&\quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1, \tau_k + 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1, \tau_k + 1)] \\
&\leq 0, \qquad \text{(A.4.4)}
\end{aligned}
$$

where the second inequality is based on $r(\mathbf{s} \vee \mathbf{s}^\circ) - r(\mathbf{s}) < 0$, and the last

inequality is base on (A.4.1).

(b.6) If $\pi^1(\boldsymbol{\tau}) = i, \pi^1(\boldsymbol{\tau}^\circ) = k$, then

$$\begin{aligned}
&p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\
&\leq p_{j,1}W(\boldsymbol{\tau}, i, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, k, V^0) \\
&\quad - p_{j,1}W\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, k, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, i, V^0) \\
&\leq \gamma p_{i,1}\big[p_{i,1}V^0(1, \tau'_j + 1, \tau_k + 1) - p_{k,1}V^0(\tau''_i + 1, \tau'_j + 1, 1) \\
&\quad - (p_{i,1} - p_{k,1})V^0(\tau''_i + 1, \tau'_j + 1, \tau_k + 1)\big] \\
&\leq \gamma p_{i,1}\left[(p_{i,1} - p_{k,1})V^0(\tau''_i{+}1, \tau'_j{+}1, \tau_k{+}1){-}(p_{i,1}{-}p_{k,1})V^0(\tau''_i{+}1, \tau'_j{+}1, \tau_k{+}1)\right] \\
&= 0, \qquad \text{(A.4.5)}
\end{aligned}$$

where the second inequality is based on (A.4.1), Lemma 2.4.2, and Lemma 2.4.4 with $\tau_i \gg \tau''_i$ as we can drop value functions with much lower value, and the third inequality is based on $Q(\boldsymbol{\tau}^\circ, k) \geq Q(\boldsymbol{\tau}^\circ, i)$. For example, the following equation is from Lemma 2.4.4, $\tau_i \gg \tau''_i$ and $\tau_k + 1 > 1$,

$$\begin{aligned}
&(1 - p_{i,1})V^0(\tau_i + 1, \tau_j + 1, \tau_k + 1) + p_{k,1}V^0(\tau''_i + 1, \tau_j + 1, 1) \\
&= (1 - p_{i,1})V^0(\tau_i + 1, \tau_j + 1, \tau_k + 1). \qquad \text{(A.4.6)}
\end{aligned}$$

(b.7) If $\pi^1(\boldsymbol{\tau}) = k, \pi^1(\boldsymbol{\tau}^\circ) = i$, then

$$
\begin{aligned}
&p_{j,1}V^1(\boldsymbol{\tau}) + (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau}^\circ) - p_{j,1}V^1(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ) - (p_{j,1} - p_{i,1})V^1(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ) \\
&\leq p_{j,1}W(\boldsymbol{\tau}, k, V^0) + (p_{j,1} - p_{i,1})W(\boldsymbol{\tau}^\circ, i, V^0) \\
&\quad - p_{j,1}W(\boldsymbol{\tau} \wedge \boldsymbol{\tau}^\circ, k, V^0) - (p_{j,1} - p_{i,1})W(\boldsymbol{\tau} \vee \boldsymbol{\tau}^\circ, k, V^0) \\
&\leq \gamma p_{k,1}[p_{j,1}V^0(\tau_i + 1, \tau_j + 1, 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1, 1) \\
&\quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1, 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1, 1)] \\
&\quad + \gamma(1 - p_{k,1})[p_{j,1}V^0(\tau_i + 1, \tau_j + 1, \tau_k + 1) + (p_{j,1} - p_{i,1})V^0(\tau_i'' + 1, \tau_j' + 1, \tau_k + 1) \\
&\quad - p_{j,1}V^0(\tau_i'' + 1, \tau_j + 1, \tau_k + 1) - (p_{j,1} - p_{i,1})V^0(\tau_i + 1, \tau_j' + 1, \tau_k + 1)] \\
&\leq 0, \qquad \text{(A.4.7)}
\end{aligned}
$$

where the second inequality is based on Lemma 2.4.4 and the last inequality is base on the (A.4.1). For cases (b.8), (b.9), and (b.10) the proof is similar to case (b.7)

Thus, the probabilistic supermodularity of the value function $V^0(\mathbf{s})$ propagates through the Bellman operator $\mathsf{B}[\cdot]$ to the optimal value function $V^*(\mathbf{s})$.

Appendix B

Proofs for Chapter 4. Goal-oriented Transmission Scheduling: Structure-guided DRL with a Unified On-policy and Off-policy Approach

B.1 Proof of Lemma 4.4.2

To prove the inequality (4.4.12), it is sufficient to prove that

$$\mathrm{Tr}(h_i^{\tau_i-1}(\bar{\mathbf{P}}_i)) + \mathrm{Tr}(h_i^{\tau_i+1}(\bar{\mathbf{P}}_i)) \geq 2\,\mathrm{Tr}(h_i^{\tau_i}(\bar{\mathbf{P}}_i)). \tag{B.1.1}$$

For notation simplicity, we drop the subscript i and notate $\tau_i - 1$ as d during the proof of (B.1.1) in the following.

Based on the stable Kalman filter property [111], we have

$$\bar{\mathbf{P}} = \mathbf{A}\bar{\mathbf{P}}\mathbf{A}^\top + \mathbf{W} - \mathbf{K}, \tag{B.1.2}$$

where

$$\mathbf{K} = \mathbf{A}\bar{\mathbf{P}}\mathbf{C}^\top \left[\mathbf{C}\bar{\mathbf{P}}\mathbf{C}^\top + \mathbf{V}\right]^{-1} \left(\mathbf{A}\bar{\mathbf{P}}\mathbf{A}^\top\right)^\top. \tag{B.1.3}$$

From (B.1.2), we derive that

$$\mathbf{A}^2\bar{\mathbf{P}}\mathbf{A}^{2^\top} - \mathbf{A}\bar{\mathbf{P}}\mathbf{A}^\top + \mathbf{A}\mathbf{W}\mathbf{A}^\top = \mathbf{A}\mathbf{K}\mathbf{A}^\top. \tag{B.1.4}$$

Next, the matrix $\mathbf{A}$ can be written in Jordan normal form as

$$\mathbf{A} = \mathbf{F}\mathbf{J}\mathbf{F}^{-1}, \tag{B.1.5}$$

where $\mathbf{J} = \mathrm{diag}(\eta_1, \ldots, \eta_{\bar{r}})$ and $\eta_{\bar{r}}$ is the eigenvalue of the matrix A. Then, we start to prove inequality (B.1.1)

$$\begin{aligned}
&\mathrm{Tr}(h^d(\bar{\mathbf{P}})) + \mathrm{Tr}(h^{d+2}(\bar{\mathbf{P}})) - 2\,\mathrm{Tr}(h^{d+1}(\bar{\mathbf{P}})) \\
&= \mathrm{Tr}\left[\mathbf{A}^d\Big(\mathbf{A}^2\bar{\mathbf{P}}\mathbf{A}^{2^\top} + \bar{\mathbf{P}} - 2\mathbf{A}\bar{\mathbf{P}}\mathbf{A}^\top + \mathbf{A}\mathbf{W}\mathbf{A}^\top - \mathbf{W}\Big)\mathbf{A}^{d^\top}\right] \\
&= \mathrm{Tr}\left[\mathbf{A}^d\left(\mathbf{A}\mathbf{K}\mathbf{A}^\top - \mathbf{K}\right)\mathbf{A}^{d^\top}\right] \\
&= \mathrm{Tr}\left[\mathbf{K}'\left(\mathbf{A}^{2d+2} - \mathbf{A}^{2d}\right)\right] \\
&= \sum_{r=1}^{\bar{r}} K'_{rr}\left(\eta_r^{2d+2} - \eta_r^{2d}\right) \\
&\geq 0
\end{aligned} \tag{B.1.6}$$

where K'_{rr} are the diagonal element of $\mathbf{K}' = \mathbf{F}^{-1}\mathbf{K}\mathbf{F}$, and the first equality is by using $h(\mathbf{X}) = \mathbf{A}\mathbf{X}\mathbf{A}^\top + \mathbf{W}$ and $h^{\tau+1}(\cdot) = h(h^\tau(\cdot))$, and the second equality is based on (B.1.2) and (B.1.4), and the last equality is derived from (B.1.5), and the inequality is from asymmetric and positive matrix $\mathbf{K}$, $\max(\eta_1, \ldots, \eta_r) > 1$, and $\tau \gg 1$.

Then, similarly, to prove convexity of the overall cost function, it is sufficient to

prove that

$$\frac{1}{2}c(\mathbf{s}'') + \frac{1}{2}c(\mathbf{s}') \geq c(\mathbf{s}), \tag{B.1.7}$$

which is equal to the inequality (B.1.1) based on the definition of the cost function (4.2.5) and $c(\mathbf{s}_t) = \sum_{i=1}^{N} c_i(\tau_{i,t})$ in the remote state estimation system.

Therefore, the two different convexity conditions in Lemma 4.4.2 hold.

B.2 Proof of Theorem 4.4.2

At kth iteration of the value iteration, the optimal action and optimal policy is defined as

$$\mathbf{a}^k \triangleq \pi^k(\mathbf{s}) = c(\mathbf{s}) + \gamma \arg\max_{\mathbf{a}\in\mathcal{A}} \left[\sum_{\mathbf{s}^+} \mathrm{P}(\mathbf{s}^+|\mathbf{s},\mathbf{a}) V^{k-1}(\mathbf{s}^+) \right]. \tag{B.2.1}$$

In order to prove Theorem 4.4.2, we define a Z-function at kth value iteration similar to the Q-function, $Z(\mathbf{s},\mathbf{a},V^k) : \mathcal{S}\times\mathcal{A}\times\mathcal{V} \to \mathbb{R}$:

$$Z(\mathbf{s},\mathbf{a},V^k) = c(\mathbf{s}) + \gamma \sum_{\mathbf{s}^+} \mathrm{P}(\mathbf{s}^+|\mathbf{s},\mathbf{a}) V^k(\mathbf{s}^+), \tag{B.2.2}$$

which can be derived to the same equation as (2.4.10) but replacing $V^*(\mathbf{s}^+)$ with $V^k(\mathbf{s}^+)$. From (2.4.6), (B.2.1), and (B.2.2), the relationship of the value function and the Z-function is given as

$$V^{k+1}(\mathbf{s}) = Z(\mathbf{s},\mathbf{a}^k,V^k) \leq Z(\mathbf{s},\mathbf{a},V^k). \tag{B.2.3}$$

Based on Lemma 2.4.1, the convergence of the optimal V function does not depend on the initial value function $V^0(\mathbf{s})$ and the properties of $V^0(\mathbf{s})$ are propagated by the Bellman operator $\mathsf{B}[\cdot]$ to $V^*(\mathbf{s})$. So, to prove Theorem 4.4.2 from Lemma 2.4.1, it is sufficient to prove that $V^1(\mathbf{s})$ holds the convexity i.e.,

$$\alpha V^1(\mathbf{s}'') + (1-\alpha)V^1(\mathbf{s}') \geq V^1(\mathbf{s}), \tag{B.2.4}$$

under the assumption that $V^0(\mathbf{s})$ is convex, i.e.,

$$\alpha V^0(\mathbf{s}'') + (1-\alpha)V^0(\mathbf{s}') \geq V^0(\mathbf{s}), \tag{B.2.5}$$

where $\mathbf{s}'' = ((\tau_i'', \tau_j), \mathbf{H})$, $\mathbf{s}' = ((\tau_i', \tau_j), \mathbf{H})$, $\mathbf{s} = ((\tau_i, \tau_j), \mathbf{H})$, and $\alpha\tau_i'' + (1-\alpha)\tau_i' = \tau_i$ for $\alpha \in [0,1]$ and $\tau_i' \geq \tau_i \geq \tau_i''$.

Since the channel states are considered to be i.i.d. and the transition probability $\mathrm{P}(\mathbf{H}^+)$ is dependent only on the probability distribution (2.2.4), the channel states are constant during each iteration. Thus, we write the state as $\mathbf{s} = \boldsymbol{\tau}$, $\mathbf{s}' = \boldsymbol{\tau}'$, and $\mathbf{s}'' = \boldsymbol{\tau}''$ for the writing simplicity.

Thus, we prove (B.2.4) by cases with different optimal actions of states for the first iteration, i.e., $\check{\mathbf{a}}^1 = \pi^1(\mathbf{s}''), \hat{\mathbf{a}}^1 = \pi^1(\mathbf{s}')$, in the following.

(a) If $\check{a}_i^1 = \hat{a}_i^1 = 1$, then

$$\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) \\
&= [\alpha c('') + (1-\alpha)c(\boldsymbol{\tau}') - c(\boldsymbol{\tau})] \\
&\quad + \gamma(1-p_{i,1})\big[\alpha V^0(1,\tau_j+1) + (1-\alpha)V^0(1,\tau_j+1) - V^0(1,\tau_j+1)\big] \\
&\quad + \gamma p_{i,1}\big[\alpha V^0(\tau_i''+1,\tau_j+1) + (1-\alpha)V^0(\tau_i'+1,\tau_j+1) - V^0(\tau_i+1,\tau_j+1)\big] \\
&\geq 0,
\end{aligned} \tag{B.2.6}$$

where the first inequality is derived from and (B.2.3), and the first equality is from (B.2.2), and the last inequality is from (4.4.13) and (B.2.5).

(b) If $\check{a}_j^1 = \hat{a}_j^1 = 1$, then

$$\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - c(\boldsymbol{\tau})] \\
&\quad + \gamma(1-p_{j,1})\big[\alpha V^0(\tau_i''+1,1) + (1-\alpha)V^0(\tau_i'+1,1) - V^0(\tau_i+1,1)\big] \\
&\quad + \gamma p_{j,1}\big[\alpha V^0(\tau_i''+1,\tau_j+1) + (1-\alpha)V^0(\tau_i'+1,\tau_j+1) - V^0(\tau_i+1,\tau_j+1)\big] \\
&\geq 0, \qquad \text{(B.2.7)}
\end{aligned}$$

where the last inequality is derived based on (4.4.13), (B.2.5), and $\alpha(\tau_i''+1) + (1-\alpha)(\tau_i'+1) = \tau_i+1$.

(c) If $\check{a}_j^1 = 1, \hat{a}_i^1 = 1$, then

$$\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\boldsymbol{\tau}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - c(\boldsymbol{\tau})] \\
&\quad + \alpha\gamma\big[(1-p_{j,1})V^0(\tau_i''+1,1) + (p_{i,1}-p_{j,1})V^0(\tau_i+1,\tau_j+1) \\
&\quad - (1-p_{j,1})V^0(\tau_i+1,1) - (p_{i,1}-p_{j,1})V^0(\tau_i''+1,\tau_j+1)\big] \\
&\quad + \gamma p_{i,1}\big[\alpha V^0(\tau_i''+1,\tau_j+1) + (1-\alpha)V^0(\tau_i'+1,\tau_j+1) - V^0(\tau_i+1,\tau_j+1)\big] \\
&\geq 0 \qquad \text{(B.2.8)}
\end{aligned}$$

where the last inequality is based on (4.4.13), (B.2.5), and Lemma 3 in [144].

(d) Based on Theorem 2 in [144], the case that $\check{a}_i^1 = 1, \hat{a}_j^1 = 1$ cannot exist at all iteration during the value iteration.

Therefore, the Bellman operator $\mathsf{B}[\cdot]$ preserve the convexity of the value function $V^0(\mathbf{s})$ to the optimal V function $V^*(\mathbf{s})$.

B.3 Proof of Theorem 4.4.3

To prove Theorem 4.4.3, we require the lemma showing the optimal V function has the asymptotic monotonicity as below.

Lemma B.3.1 (Asymptotic monotonicity of the optimal V function w.r.t AoI state [144])**.** *For states* $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$ *and* $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$*, where* $\tau'_i \gg \tau_i$*, the optimal V function holds the inequality:*

$$V^*(\mathbf{s}') \gg V^*(\mathbf{s}). \tag{B.3.1}$$

Then, similar to the proof of Theorem 4.4.2, to prove Theorem 4.4.3 based on Lemma 2.4.1, it is sufficient to prove that $V^1(\mathbf{s})$ is asymptotically convex, i.e.,

$$\alpha V^1(\mathbf{s}'') + (1-\alpha)V^1(\mathbf{s}') \geq V^1(\mathbf{s}), \tag{B.3.2}$$

under the assumption that the initial value function $V^0(\mathbf{s})$ has the asymptotic convexity, i.e.,

$$\alpha V^0(\mathbf{s}'') + (1-\alpha)V^0(\mathbf{s}') \geq V^0(\mathbf{s}), \tag{B.3.3}$$

where $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$, $\mathbf{s}'' = (\boldsymbol{\tau}''_{(i)}, \mathbf{H})$, and $\alpha\tau''_i + (1-\alpha)\tau'_i = \tau_i$ for $\alpha \in [0,1]$ and $\tau'_i \geq \tau_i \gg \tau''_i$.

Therefore, in the following, we also write the states as $\mathbf{s} = \boldsymbol{\tau}$, $\mathbf{s}' = \boldsymbol{\tau}'$, and $\mathbf{s}'' = \boldsymbol{\tau}''$, and prove (B.3.2) by cases (a) and (b) with different optimal actions $\check{\mathbf{a}}^1 = \pi^1(\boldsymbol{\tau}'')$, $\hat{\mathbf{a}}^1 = \pi^1(\boldsymbol{\tau}')$.

(a) If $\check{\mathbf{a}}^1 = \hat{\mathbf{a}}^1$, then

$$\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - c(\boldsymbol{\tau})] \\
&\quad + \gamma \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}_{-i}, \check{\mathbf{a}}_{-i}^1) \Bigg[\alpha \sum_{\tau_i''^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(\boldsymbol{\tau}''^+) \\
&\quad + (1-\alpha) \sum_{\tau_i'^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{h}_i, \hat{\mathbf{a}}_i^1) V^0(\boldsymbol{\tau}'^+) - \sum_{\tau_i^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(\boldsymbol{\tau}^+) \Bigg] \\
&\geq 0 \qquad \text{(B.3.4)}
\end{aligned}$$

where the first inequality is from (B.2.3), and the first equality is from (B.2.2) and $\check{\mathbf{a}}^1 = \hat{\mathbf{a}}^1$, and the final inequality is from (4.4.14) and the following equality

$$\begin{aligned}
&\mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(\tau_i'' + 1, \boldsymbol{\tau}_{-i}^+) + \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{h}_i, \hat{\mathbf{a}}_i^1) V^0(\tau_i' + 1, \boldsymbol{\tau}_{-i}^+) \\
&\geq \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(\tau_i + 1, \boldsymbol{\tau}_{-i}^+), \qquad \text{(B.3.5)}
\end{aligned}$$

and

$$\begin{aligned}
&\mathrm{P}(\tau_i'' = 1 | \tau_i'', \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+) + \mathrm{P}(\tau_i' = 1 | \tau_i', \mathbf{h}_i, \hat{\mathbf{a}}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+) \\
&\geq \mathrm{P}(\tau_i = 1 | \tau_i, \mathbf{h}_i, \check{\mathbf{a}}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+), \qquad \text{(B.3.6)}
\end{aligned}$$

achieved by (B.3.3), $\check{\mathbf{a}}^1 = \hat{\mathbf{a}}^1$, and $\alpha(\tau_i''+1) + (1-\alpha)(\tau_i'+1) = \tau_i + 1$.

(b) If $\check{\mathbf{a}}^1 \neq \hat{\mathbf{a}}^1$, then

$$\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - Z(\boldsymbol{\tau}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - c(\boldsymbol{\tau})] + \gamma \Bigg[\alpha \sum_{\boldsymbol{\tau}''^+} \mathrm{P}(\boldsymbol{\tau}''^+ | \boldsymbol{\tau}'', \mathbf{H}, \check{\mathbf{a}}^1) V^0(\boldsymbol{\tau}''^+) \\
&\quad + (1-\alpha) \sum_{\boldsymbol{\tau}'^+} \mathrm{P}(\boldsymbol{\tau}'^+ | \boldsymbol{\tau}', \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}'^+) - \sum_{\boldsymbol{\tau}^+} \mathrm{P}(\boldsymbol{\tau}^+ | \boldsymbol{\tau}, \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}^+) \Bigg] \\
&\geq \gamma \Bigg[\alpha \sum_{\boldsymbol{\tau}''^+} \mathrm{P}(\boldsymbol{\tau}''^+ | \boldsymbol{\tau}'', \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}''^+) + (1-\alpha) \sum_{\boldsymbol{\tau}'^+} \mathrm{P}(\boldsymbol{\tau}'^+ | \boldsymbol{\tau}', \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}'^+) \\
&\quad - \sum_{\boldsymbol{\tau}^+} \mathrm{P}(\boldsymbol{\tau}^+ | \boldsymbol{\tau}, \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}^+) \Bigg] \\
&\geq 0
\end{aligned} \tag{B.3.7}$$

where the first equality is derived based on (B.2.2), and the second inequality is based on (4.4.14), the following inequality

$$\sum_{\boldsymbol{\tau}'^+} \mathrm{P}(\boldsymbol{\tau}'^+ | \boldsymbol{\tau}', \mathbf{H}, \hat{\mathbf{a}}^1) V^0(\boldsymbol{\tau}'^+) \gg \sum_{\boldsymbol{\tau}'^+} \mathrm{P}(\boldsymbol{\tau}''^+ | \boldsymbol{\tau}'', \mathbf{H}, \check{\mathbf{a}}^1) V^0(\boldsymbol{\tau}''^+) \tag{B.3.8}$$

achieved by Lemma B.3.1 and $\tau_i' \gg \tau_i''$, and the last inequality is based on (B.3.3) and replacing $\check{\mathbf{a}}^1$ by $\hat{\mathbf{a}}^1$ in (B.3.5) and (B.3.6).

Therefore, the asymptotic convexity of the value function $V^0(\mathbf{s})$ is preserved by the Bellman operator $\mathsf{B}[\cdot]$ to the optimal V function $V^*(\mathbf{s})$.

B.4 Proof of Proposition 4.4.1

Before proving Proposition 4.4.1, we develop the following Lemma.

Lemma B.4.1. *If the devices of the system are connected by a gateway, then for states* $\dot{\mathbf{s}}'' = (\tau_i'', \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\ddot{\mathbf{s}}' = (\tau_i', \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\dot{\mathbf{s}} = (\tau_i, \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, *and* $\ddot{\mathbf{s}} = (\tau_i, \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, *where*

$\alpha\tau_i'' + (1-\alpha)\tau_i' = \tau_i$, $\alpha \in [0,1]$, *and* $\tau_i' \geq \tau_i \geq \tau_i'' \gg 1$, *then the optimal V function holds the following inequality:*

$$\alpha V(\dot{\mathbf{s}}'') + (1-\alpha)V(\ddot{\mathbf{s}}') \geq \alpha V(\dot{\mathbf{s}}) + (1-\alpha)V(\ddot{\mathbf{s}}). \tag{B.4.1}$$

Proof. See Appendix B.6. □

Next, similar to the proof of Theorem 4.4.2, to prove Proposition 4.4.1 based on Lemma 2.4.1, it is sufficient to prove that $V^1(\mathbf{s})$ holds the following inequality

$$\alpha V^1(\mathbf{s}'') + (1-\alpha)V^1(\mathbf{s}') \geq V^1(\mathbf{s}), \tag{B.4.2}$$

under the assumption that the initial value function $V^0(\mathbf{s})$ holds the inequality:

$$\alpha V^0(\mathbf{s}'') + (1-\alpha)V^0(\mathbf{s}') \geq V^0(\mathbf{s}), \tag{B.4.3}$$

where $\mathbf{s}'' = (\boldsymbol{\tau}''_{(i)}, \mathbf{H})$, $\mathbf{s}' = (\boldsymbol{\tau}'_{(i)}, \mathbf{H})$ and $\mathbf{s} = (\boldsymbol{\tau}, \mathbf{H})$, and $\alpha\tau_i'' + (1-\alpha)\tau_i' = \tau_i$ for $\alpha \in [0,1]$ and $\tau_i' \geq \tau_i \geq \tau_i'' \gg 1$.

We also write the states as $\mathbf{s} = \boldsymbol{\tau}$, $\mathbf{s}' = \boldsymbol{\tau}'$, and $\mathbf{s}'' = \boldsymbol{\tau}''$, and prove (B.3.2) by cases (a), (b), and (c) with different optimal actions $\check{\mathbf{a}}^1 = \pi^1(\boldsymbol{\tau}''), \hat{\mathbf{a}}^1 = \pi^1(\boldsymbol{\tau}')$.

(a) If $\check{a}_i^1 = m_1, \hat{a}_i^1 = m_2$, then there are 2 cases with different packet drop rate: (a.1) $p_{i,m_1} \leq p_{i,m_2}$ and (a.2) $p_{i,m_1} > p_{i,m_2}$.

(a.1) If $p_{i,m_1} \leq p_{i,m_2}$ and $\hat{a}_j^1 = m_1$, we define another action $\dot{\mathbf{a}}^1$, where $\dot{a}_i^1 = m_1, \dot{a}_j^1 = m_2$, and $\dot{\mathbf{a}}^1_{-i,j} = \hat{\mathbf{a}}^1_{-i,j}$, then we have $p_{j,m_1} < p_{j,m_2}$ and

$$\begin{aligned}&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)\sum_{\tau_j^+}\mathrm{P}(\tau_j^+|\tau_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i'+1,\boldsymbol{\tau}_{-i}^+)\\ \geq&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\dot{a}_i^1)\sum_{\tau_j^+}\mathrm{P}(\tau_j^+|\tau_j,\mathbf{H},\dot{a}_j^1)V^0(\tau_i'+1,\boldsymbol{\tau}_{-i}^+).\end{aligned} \tag{B.4.4}$$

Next, we derive that

$$
\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\boldsymbol{\tau}, \dot{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - \alpha c(\boldsymbol{\tau}) - (1-\alpha)c(\boldsymbol{\tau})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \hat{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha \sum_{\tau_i^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&- (1-\alpha) \sum_{\tau_i^+} \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \dot{a}_i^1) \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}^+} \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&- (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}^+} \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq 0, \qquad \text{(B.4.5)}
\end{aligned}
$$

where the first inequality is from (B.2.3), and the first equality is derived based on (B.2.2), and the second inequality is from (4.4.14), (B.4.4), and the following equality

$$\mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \check{a}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+), \qquad \text{(B.4.6)}$$

and

$$\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)V^0(\tau_i'+1,\boldsymbol{\tau}_{-i}^+) \gg \mathrm{P}(\tau_i'^{+}=1|\tau_i',\mathbf{H},\hat{a}_i^1)V^0(1,\boldsymbol{\tau}_{-i}^+), \quad \text{(B.4.7)}$$

and

$$\mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\dot{a}_i^1)V^0(\tau_i+1,\boldsymbol{\tau}_{-i}^+) \gg \mathrm{P}(\tau_i{}^{+}=1|\tau_i,\mathbf{H},\dot{a}_i^1)V^0(1,\boldsymbol{\tau}_{-i}^+), \quad \text{(B.4.8)}$$

achieved by Lemma B.3.1, $\tau_i'+1 \gg 1$, and $\tau_i+1 \gg 1$, and the last inequality is derived based on Lemma B.4.1 and the following equality

$$\mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\check{a}_i^1) = \mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\dot{a}_i^1) = \mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\check{a}_i^1) = \mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\dot{a}_i^1), \quad \text{(B.4.9)}$$

achieved by $\check{a}_i^1 = \dot{a}_i^1$.

(a.2) If $p_{i,m_1} > p_{i,m_2}$ and $\check{a}_j^1 = m_1$, we define another action $\dot{\mathbf{a}}^1$, where $\dot{a}_i^1 = m_2, \dot{a}_j^1 = m_1$, and $\dot{\mathbf{a}}_{-i,j}^1 = \check{\mathbf{a}}_{-i,j}^1$, then we have $p_{j,m_1} > p_{j,m_2}$ and

$$\begin{aligned}&\mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\check{a}_i^1)\sum_{\tau_j^+}\mathrm{P}(\tau_j^+|\tau_j,\mathbf{H},\check{a}_j^1)V^0(\tau_i''+1,\boldsymbol{\tau}_{-i}^+)\\ \geq&\mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\dot{a}_i^1)\sum_{\tau_j^+}\mathrm{P}(\tau_j^+|\tau_j,\mathbf{H},\dot{a}_j^1)V^0(\tau_i''+1,\boldsymbol{\tau}_{-i}^+).\end{aligned} \quad \text{(B.4.10)}$$

Next, we derive that

$$
\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\boldsymbol{\tau}, \dot{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\boldsymbol{\tau}, \hat{\mathbf{a}}^1, V^0) \\
&\geq \alpha \, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \dot{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \dot{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}''^+) \\
&\quad + (1-\alpha) \, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \hat{a}_j^1) \, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}'^+) \\
&\quad - \alpha \, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \dot{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&\quad - (1-\alpha) \, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\tau_j^+} \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\tau_j^+ | \tau_j, \mathbf{H}, \hat{a}_j^1) \, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq 0,
\end{aligned}
\tag{B.4.11}
$$

where the second inequality is from (4.4.14), (B.4.10), and the following equality

$$
\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+), \tag{B.4.12}
$$

and

$$
\mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(\tau_i'' + 1, \boldsymbol{\tau}_{-i}^+) \gg \mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(1, \boldsymbol{\tau}_{-i}^+), \tag{B.4.13}
$$

and (B.4.8) achieved by Lemma B.3.1, $\tau_i'' + 1 \gg 1$, and $\tau_i + 1 \gg 1$, and the last inequality is derived based on Lemma B.4.1 and the following equality

$$
\mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1), \tag{B.4.14}
$$

achieved by $\check{a}_i^1 = \dot{a}_i^1$.

(b) If $\check{a}_i^1 = m, \hat{a}_i^1 = 0$, then we assume that $\hat{a}_j^1 = m$ and we have

$$
\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\boldsymbol{\tau}, \check{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\boldsymbol{\tau}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - \alpha c(\boldsymbol{\tau}) - (1-\alpha)c(\boldsymbol{\tau})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1) \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha \sum_{\tau_i^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&- (1-\alpha) \sum_{\tau_i^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\tau_j + 1 | \tau_j, \mathbf{H}, \hat{a}_j^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\tau_j + 1 | \tau_j, \mathbf{H}, \hat{a}_j^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq 0,
\end{aligned}
\tag{B.4.15}
$$

where the second inequality is derived from (4.4.14) and the following equality

$$
\begin{aligned}
&\mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(1, \boldsymbol{\tau}_{-i}^+) \\
&= \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(1, \boldsymbol{\tau}_{-i}^+),
\end{aligned}
\tag{B.4.16}
$$

and

$$
\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) = 0 \tag{B.4.17}
$$

from $\hat{a}_i^1 = 0$, and the inequality

$$\begin{aligned}&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)\,\mathrm{P}(\tau_j{}^+=1|\tau_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i'+1,1,\boldsymbol{\tau}'^+_{-i,j})\\ &\geq \mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\hat{a}_i^1)\,\mathrm{P}(\tau_j{}^+=1|\tau_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i'+1,1,\boldsymbol{\tau}^+_{-i,j})\end{aligned} \tag{B.4.18}$$

achieved from Lemma 4.4.1 with $\tau_i' \geq \tau_i$, and the last inequality is based on Lemma B.4.1 with the equality

$$\mathrm{P}(\tau_i' + 1|\tau_i, \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \hat{a}_i^1) = 1, \tag{B.4.19}$$

and

$$\mathrm{P}(\tau_i''+1|\tau_i,\mathbf{H},\check{a}_i^1)=\mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\check{a}_i^1)=\mathrm{P}(\tau_j+1|\tau_j,\mathbf{H},\hat{a}_j^1) \tag{B.4.20}$$

as $\check{a}_i^1 = \hat{a}_j^1$.

(c) If $\check{a}_i^1 = 0, \hat{a}_i^1 = m$ and $\check{a}_j^1 = m$, we define another action $\dot{\mathbf{a}}^1$ where $\dot{a}_i^1 = m, \dot{a}_j^1 = 0$,

and $\dot{\mathbf{a}}_{-i,j} = \check{\mathbf{a}}_{-i,j}$, then we have

$$
\begin{aligned}
&\alpha V^1(\boldsymbol{\tau}'') + (1-\alpha)V^1(\boldsymbol{\tau}') - V^1(\boldsymbol{\tau}) \\
&\geq \alpha Z(\boldsymbol{\tau}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\boldsymbol{\tau}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\boldsymbol{\tau}, \dot{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\boldsymbol{\tau}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\boldsymbol{\tau}'') + (1-\alpha)c(\boldsymbol{\tau}') - \alpha c(\boldsymbol{\tau}) - (1-\alpha)c(\boldsymbol{\tau})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\tau_j{}^+} \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\tau_j{}^+ | \tau_j, \mathbf{H}, \check{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1) \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha \sum_{\tau_i{}^+} \sum_{\tau_j{}^+} \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\tau_i{}^+ | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\tau_j{}^+ | \tau_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&- (1-\alpha) \sum_{\tau_i{}^+} \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\tau_i{}^+ | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\tau_j + 1 | \tau_j, \mathbf{H}, \check{a}_j^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\boldsymbol{\tau}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\tau_j + 1 | \tau_j, \mathbf{H}, \dot{a}_j^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} (\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}^+) \\
&- (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) V^0(\boldsymbol{\tau}^+) \\
&+ \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\tau_j{}^+ = 1 | \tau_j, \mathbf{H}, \check{a}_j^1) \sum_{\boldsymbol{\tau}_{-i,j}{}^+} \mathrm{P}(\boldsymbol{\tau}_{-i,j}^+ | \boldsymbol{\tau}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\boldsymbol{\tau}''^+) \\
&\geq 0, \qquad \text{(B.4.21)}
\end{aligned}
$$

where the second inequality is derived from (4.4.14) and the following equality

$$
\begin{aligned}
&\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(1, \boldsymbol{\tau}_{-i}^+) \\
&= \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\boldsymbol{\tau}_{-i}^+} \mathrm{P}(\boldsymbol{\tau}_{-i}^+ | \boldsymbol{\tau}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(1, \boldsymbol{\tau}_{-i}^+), \qquad \text{(B.4.22)}
\end{aligned}
$$

and

$$\mathrm{P}(\tau_i''^{+} = 1|\tau_i'', \mathbf{H}, \check{a}_i^1) = 0 \tag{B.4.23}$$

from $\check{a}_i^1 = 0$, and the inequality

$$\mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \dot{a}_i^1)V^0(\tau_i + 1, \boldsymbol{\tau}_{-i}^+) \gg \mathrm{P}(\tau_i{}^+ = 1|\tau_i, \mathbf{H}, \dot{a}_i^1)V^0(1, \boldsymbol{\tau}_{-i}^+), \tag{B.4.24}$$

achieved from Lemma B.3.1 and $\tau_i + 1 \gg 1$, and the last inequality is based on Lemma B.4.1 with the equality

$$\mathrm{P}(\tau_i' + 1|\tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_j + 1|\tau_j, \mathbf{H}, \dot{a}_j^1) = 1, \tag{B.4.25}$$

and

$$\mathrm{P}(\tau_j + 1|\tau_j, \mathbf{H}, \check{a}_j^1) = \mathrm{P}(\tau_i' + 1|\tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \hat{a}_i^1), \tag{B.4.26}$$

and the inequality $V^0(\boldsymbol{\tau}''^{+}) \geq 0$.

Therefore, the Bellman operator $\mathsf{B}[\cdot]$ preserve this convexity of the value function $V^0(\mathbf{s})$ to the optimal V function $V^*(\mathbf{s})$.

B.5 Proof of Theorem 4.4.4

To prove Theorem 4.4.4, it is sufficient to prove that for each action $\mathbf{a}$, where $a_i = 0, a_j = m$ for $i \in \mathcal{I}, j \notin \mathcal{I}$, we can find a better action $\dot{\mathbf{a}}$, where $\dot{a}_i = m$, $\dot{a}_j = 0$ and $\dot{\mathbf{a}}_{\setminus\{i,j\}} = \mathbf{a}_{\setminus\{i,j\}}$. Therefore, based on (2.4.5), the actions $\mathbf{a}$ and $\dot{\mathbf{a}}$ follows the inequality

$$Q(\mathbf{s}, \dot{\mathbf{a}}) \leq Q(\mathbf{s}, \mathbf{a}), \tag{B.5.1}$$

which implies that the optimal action of the device i cannot be idle, i.e., $a^* \neq 0$ as in Theorem 4.4.4. To prove (B.5.1), we develop the following lemma of the optimal

V function in an asymptotic form. For writing simplicity, we write the AoI state as $\boldsymbol{\delta} = (\delta_i, \delta_j, \boldsymbol{\delta}_{\backslash\{i,j\}})$ in the following.

By using (2.4.4), we have

$$Q(\mathbf{s}, \dot{\mathbf{a}}) = \sum_{\boldsymbol{\delta}^+_{\backslash\{i,j\}}} \sum_{\mathbf{G}^+} \sum_{\delta_i^+} \sum_{\delta_j^+} \mathrm{P}\left(\boldsymbol{\delta}^+_{\backslash\{i,j\}} | \boldsymbol{\delta}_{\backslash\{i,j\}}, \dot{\mathbf{a}}_{\backslash\{i,j\}}, \mathbf{G}_{\backslash\{i,j\}}\right) \tag{B.5.2}$$

$$\times \mathrm{P}(\mathbf{G}^+)\, \mathrm{P}(\delta_i^+ | \delta_i, \dot{a}_i, \mathbf{G}_i)\, \mathrm{P}(\delta_j^+ | \delta_j, \dot{a}_j, \mathbf{G}_j) \upsilon(\mathbf{s}^+), \tag{B.5.3}$$

and

$$Q(\mathbf{s}, \mathbf{a}) = \sum_{\boldsymbol{\delta}^+_{\backslash\{i,j\}}} \sum_{\mathbf{G}^+} \sum_{\delta_i^+} \sum_{\delta_j^+} \mathrm{P}\left(\boldsymbol{\delta}^+_{\backslash\{i,j\}} | \boldsymbol{\delta}_{\backslash\{i,j\}}, \mathbf{a}_{\backslash\{i,j\}}, \mathbf{G}_{\backslash\{i,j\}}\right) \tag{B.5.4}$$

$$\times \mathrm{P}(\mathbf{G}^+)\, \mathrm{P}(\delta_i^+ | \delta_i, a_i, \mathbf{G}_i)\, \mathrm{P}(\delta_j^+ | \delta_j, a_j, \mathbf{G}_j) \upsilon(\mathbf{s}^+). \tag{B.5.5}$$

Since $\dot{\mathbf{a}}_{\backslash\{i,j\}} = \mathbf{a}_{\backslash\{i,j\}}$ and $a_i = \dot{a}_j = 0$, we derive that

$$\mathrm{P}(\boldsymbol{\delta}^+_{\backslash\{i,j\}} | \boldsymbol{\delta}_{\backslash\{i,j\}}, \dot{\mathbf{a}}_{\backslash\{i,j\}}, \mathbf{G}_{\backslash\{i,j\}}) \tag{B.5.6}$$

$$= \mathrm{P}(\boldsymbol{\delta}^+_{\backslash\{i,j\}} | \boldsymbol{\delta}_{\backslash\{i,j\}}, \mathbf{a}_{\backslash\{i,j\}}, \mathbf{G}_{\backslash\{i,j\}}), \tag{B.5.7}$$

and

$$\mathrm{P}(\delta_j^+ = \delta_j + 1 | \delta_j, \dot{a}_j, \mathbf{G}_j) = \mathrm{P}(\delta_i^+ = \delta_i + 1 | \delta_i, a_i, \mathbf{G}_i) = 1. \tag{B.5.8}$$

Based on (B.5.3), (B.5.5), (B.5.7), and (B.5.8), the inequality (B.5.1) is equivalent to

$$\sum_{\delta_i^+} \mathrm{P}(\delta_i^+ | \delta_i, \dot{a}_i, \mathbf{G}_i) \upsilon(\mathbf{s}^+) \leq \sum_{\delta_j^+} \mathrm{P}(\delta_j^+ | \delta_j, a_j, \mathbf{G}_j) \upsilon(\mathbf{s}^+). \tag{B.5.9}$$

Next, the following equality

$$\mathrm{P}(\delta_i^+ = \delta_i + 1 | \delta_i, \dot{a}_i, \mathbf{G}_i) \upsilon(\delta_i + 1, \delta_j + 1, \boldsymbol{\delta}^+_{\backslash\{i,j\}}, \mathbf{G}^+) \tag{B.5.10}$$

$$= \mathrm{P}(\delta_j^+ = \delta_j + 1 | \delta_j, a_j, \mathbf{G}_j) \upsilon(\delta_i + 1, \delta_j + 1, \boldsymbol{\delta}^+_{\backslash\{i,j\}}, \mathbf{G}^+) \tag{B.5.11}$$

and

$$\mathrm{P}(\delta_i^+ = 1 | \delta_i, \dot{a}_i, \mathbf{G}_i) = \mathrm{P}(\delta_j^+ = 1 | \delta_j, a_j, \mathbf{G}_j) \tag{B.5.12}$$

are obtained from $\dot{a}_i = a_j = m$, $g_{i,m} = g_{j,m}$ and (B.5.8). From (B.5.11) and (B.5.12),

to prove (B.5.9), it is sufficient to show that

$$\upsilon(1, \delta_j+1, \boldsymbol{\delta}^+_{\setminus\{i,j\}}, \mathbf{G}^+) \leq \upsilon(\delta_i+1, 1, \boldsymbol{\delta}^+_{\setminus\{i,j\}}, \mathbf{G}^+), \tag{B.5.13}$$

when $\delta_i > \bar{\delta} \gg 1$.

The inquality (B.5.13) is equivalent to the following Lemma.

Lemma B.5.1. *Consider a multi-device-multi-channel system with co-located devices. For states* $\mathbf{s} = (\boldsymbol{\delta}, \mathbf{G})$ *and* $\mathbf{s}^\circ = (\boldsymbol{\delta}^\circ, \mathbf{G})$*, where* $\boldsymbol{\delta} = (\delta_i, \delta_j, \boldsymbol{\delta}_{\setminus\{i,j\}})$*, and* $\boldsymbol{\delta}^\circ = (\delta'_i, \delta''_j, \boldsymbol{\delta}_{\setminus\{i,j\}})$ *with* $\delta'_i \geq \delta_j \geq \delta''_j$ *and* $\delta'_i \gg \delta_i, \forall i \in \mathcal{I}, j \notin \mathcal{I}$*, the following inequality hold*

$$\upsilon^*(\mathbf{s}^\circ) \geq \upsilon^*(\mathbf{s}). \tag{B.5.14}$$

Proof. See Appendix B.7. □

Therefore, the inquality (B.5.1) holds under Lemma B.5.1, which is exactly Theorem 4.4.4.

B.6 Proof of Lemma B.4.1

Similar to the proof of Theorem 4.4.2, to prove Proposition 4.4.1 based on Lemma 2.4.1, it is sufficient to prove that $V^1(\mathbf{s})$ holds the following inequality

$$\alpha V^1(\dot{\mathbf{s}}'') + (1-\alpha)V^1(\ddot{\mathbf{s}}') \geq \alpha V^1(\dot{\mathbf{s}}) + (1-\alpha)V^1(\ddot{\mathbf{s}}), \tag{B.6.1}$$

under the assumption that the value function $V^0(\mathbf{s})$ holds the inequality

$$\alpha V^0(\dot{\mathbf{s}}'') + (1-\alpha)V^0(\ddot{\mathbf{s}}') \geq \alpha V^0(\dot{\mathbf{s}}) + (1-\alpha)V^0(\ddot{\mathbf{s}}), \tag{B.6.2}$$

where $\dot{\mathbf{s}}'' = (\tau''_i, \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\ddot{\mathbf{s}}' = (\tau'_i, \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\dot{\mathbf{s}} = (\tau_i, \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\ddot{\mathbf{s}} = (\tau_i, \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H})$, $\alpha\tau''_i + (1-\alpha)\tau'_i = \tau_i$, $\alpha \in [0,1]$, and $\tau'_i \geq \tau_i \geq \tau''_i \gg 1$.

In the following, we write the state as $\dot{\mathbf{s}}'' = \dot{\boldsymbol{\tau}}''$, $\ddot{\mathbf{s}}' = \ddot{\boldsymbol{\tau}}'$, $\dot{\mathbf{s}} = \dot{\boldsymbol{\tau}}$, and $\ddot{\mathbf{s}} = \ddot{\boldsymbol{\tau}}$ and prove (B.6.1) based on different cases with different optimal actions $\check{\mathbf{a}}^1 = \pi^1(\dot{\boldsymbol{\tau}}''), \hat{\mathbf{a}}^1 = \pi^1(\ddot{\boldsymbol{\tau}}')$.

(a) If $\check{a}_i^1 = m_1, \hat{a}_i^1 = m_2$, then there are 2 cases with different packet drop rate: (a.1) $p_{i,m_1} \leq p_{i,m_2}$ and (a.2) $p_{i,m_1} > p_{i,m_2}$.

(a.1) If $p_{i,m_1} \leq p_{i,m_2}$ and $\hat{a}_j^1 = m_1$, we define another action $\dot{\mathbf{a}}^1$, where $\dot{a}_i^1 = m_1, \dot{a}_j^1 = m_2$, and $\dot{\mathbf{a}}_{-i,j}^1 = \hat{\mathbf{a}}_{-i,j}^1$, then we have $p_{j,m_1} < p_{j,m_2}$ and

$$\begin{aligned}&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)\sum_{\ddot{\tau}_j^+}\mathrm{P}(\ddot{\tau}_j^+|\ddot{\tau}_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i'+1,\boldsymbol{\tau}_{-i}^+)\\ \geq&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\dot{a}_i^1)\sum_{\ddot{\tau}_j^+}\mathrm{P}(\ddot{\tau}_j^+|\ddot{\tau}_j,\mathbf{H},\dot{a}_j^1)V^0(\tau_i'+1,\boldsymbol{\tau}_{-i}^+).\end{aligned} \tag{B.6.3}$$

Next, we derive that

$$
\begin{aligned}
&\alpha V^1(\dot{\boldsymbol{\tau}}'') + (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}') - \alpha V^1(\dot{\boldsymbol{\tau}}) - (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}) \\
&\geq \alpha Z(\dot{\boldsymbol{\tau}}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\ddot{\boldsymbol{\tau}}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\dot{\boldsymbol{\tau}}, \check{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\ddot{\boldsymbol{\tau}}, \dot{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\dot{\boldsymbol{\tau}}'') + (1-\alpha)c(\ddot{\boldsymbol{\tau}}') - \alpha c(\dot{\boldsymbol{\tau}}) - (1-\alpha)c(\ddot{\boldsymbol{\tau}})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\tau_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\dot{\tau}_j^+ | \dot{\tau}_j, \mathbf{H}, \hat{a}_j^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha \sum_{\tau_i^+} \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha) \sum_{\tau_i^+} \sum_{\ddot{\tau}_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\ddot{\tau}_j^+ | \ddot{\tau}_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \dot{a}_i^1) \sum_{\ddot{\tau}_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\tau}_j^+ | \ddot{\tau}_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) \sum_{\ddot{\tau}_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\tau}_j^+ | \ddot{\tau}_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq 0,
\end{aligned}
\tag{B.6.4}
$$

where the first inequality is from (B.2.3), and the first equality is derived based on (B.2.2), and the second inequality is from (4.4.14), (B.6.3), and the following equality

$$
\mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(1, \dot{\boldsymbol{\tau}}_{-i}^+) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \check{a}_i^1) V^0(1, \dot{\boldsymbol{\tau}}_{-i}^+), \tag{B.6.5}
$$

and inequality

$$\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)V^0(\tau_i'+1,\ddot{\boldsymbol{\tau}}_{-i}^+) \gg \mathrm{P}(\tau_i'^{+}=1|\tau_i',\mathbf{H},\hat{a}_i^1)V^0(1,\ddot{\boldsymbol{\tau}}_{-i}^+), \quad \text{(B.6.6)}$$

and

$$\mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\dot{a}_i^1)V^0(\tau_i+1,\ddot{\boldsymbol{\tau}}_{-i}^+) \gg \mathrm{P}(\tau_i{}^{+}=1|\tau_i,\mathbf{H},\dot{a}_i^1)V^0(1,\ddot{\boldsymbol{\tau}}_{-i}^+), \quad \text{(B.6.7)}$$

achieved by Lemma B.3.1, $\tau_i'+1 \gg 1$, and $\tau_i+1 \gg 1$, and the last inequality is derived based on (B.6.2) and the following equality

$$\mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\check{a}_i^1)=\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\dot{a}_i^1)=\mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\check{a}_i^1)=\mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\dot{a}_i^1) \quad \text{(B.6.8)}$$

achieved by $\check{a}_i^1=\dot{a}_i^1$.

(a.2) If $p_{i,m_1} > p_{i,m_2}$ and $\check{a}_j^1 = m_1$, we define another action $\dot{\mathbf{a}}^1$, where $\dot{a}_i^1 = m_2, \dot{a}_j^1 = m_1$, and $\dot{\mathbf{a}}_{-i,j}^1 = \check{\mathbf{a}}_{-i,j}^1$, then we have $p_{j,m_1} > p_{j,m_2}$ and

$$\begin{aligned}&\mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\check{a}_i^1)\sum_{\dot{\tau}_j^+}\mathrm{P}(\dot{\tau}_j^+|\dot{\tau}_j,\mathbf{H},\check{a}_j^1)V^0(\tau_i''+1,\dot{\boldsymbol{\tau}}_{-i}^+)\\ &\geq \mathrm{P}(\tau_i''+1|\tau_i'',\mathbf{H},\dot{a}_i^1)\sum_{\dot{\tau}_j^+}\mathrm{P}(\dot{\tau}_j^+|\dot{\tau}_j,\mathbf{H},\dot{a}_j^1)V^0(\tau_i''+1,\dot{\boldsymbol{\tau}}_{-i}^+). \quad \text{(B.6.9)}\end{aligned}$$

Next, we derive that

$$
\begin{aligned}
&\alpha V^1(\dot{\boldsymbol{\tau}}'') + (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}') - \alpha V^1(\dot{\boldsymbol{\tau}}) - (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}) \\
&\geq \alpha Z(\dot{\boldsymbol{\tau}}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\ddot{\boldsymbol{\tau}}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\dot{\boldsymbol{\tau}}, \dot{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\ddot{\boldsymbol{\tau}}, \hat{\mathbf{a}}^1, V^0) \\
&\geq \alpha \, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \dot{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \dot{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha) \, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\tau}_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\tau}_j^+ | \ddot{\tau}_j, \mathbf{H}, \hat{a}_j^1) \, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha \, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \dot{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha) \, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\tau}_j^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\tau}_j^+ | \ddot{\tau}_j, \mathbf{H}, \hat{a}_j^1) \, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq 0,
\end{aligned} \tag{B.6.10}
$$

where the second inequality is from (4.4.14), (B.6.9), and the following equality

$$
\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) V^0(1, \ddot{\boldsymbol{\tau}}_{-i}^+) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) V^0(1, \ddot{\boldsymbol{\tau}}_{-i}^+), \tag{B.6.11}
$$

and

$$
\mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(\tau_i'' + 1, \dot{\boldsymbol{\tau}}_{-i}^+) \gg \mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) V^0(1, \dot{\boldsymbol{\tau}}_{-i}^+), \tag{B.6.12}
$$

and (B.6.7) achieved by Lemma B.3.1, $\tau_i'' + 1 \gg 1$, and $\tau_i + 1 \gg 1$, and the last inequality is derived based on (B.6.2) and the following equality

$$
\mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \tag{B.6.13}
$$

achieved by $\check{a}_i^1 = \dot{a}_i^1$.

(b) If $\check{a}_i^1 = m, \hat{a}_i^1 = 0$, then we assume that $\hat{a}_j^1 = m$ and we have

$$
\begin{aligned}
&\alpha V^1(\dot{\boldsymbol{\tau}}'') + (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}') - \alpha V^1(\dot{\boldsymbol{\tau}}) - (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}) \\
&\geq \alpha Z(\dot{\boldsymbol{\tau}}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\ddot{\boldsymbol{\tau}}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\dot{\boldsymbol{\tau}}, \check{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\ddot{\boldsymbol{\tau}}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\dot{\boldsymbol{\tau}}'') + (1-\alpha)c(\ddot{\boldsymbol{\tau}}') - \alpha c(\dot{\boldsymbol{\tau}}) - (1-\alpha)c(\ddot{\boldsymbol{\tau}})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha \sum_{\tau_i^+} \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha) \sum_{\tau_i^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\ddot{\tau}_j + 1 | \ddot{\tau}_j, \mathbf{H}, \hat{a}_j^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\ddot{\tau}_j + 1 | \ddot{\tau}_j, \mathbf{H}, \hat{a}_j^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i,j}^+ | \ddot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \hat{\mathbf{a}}_{-i,j}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq 0, \qquad \text{(B.6.14)}
\end{aligned}
$$

where the second inequality is derived from (4.4.14) and the following equality

$$
\begin{aligned}
&\mathrm{P}(\tau_i''^+ = 1 | \tau_i'', \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(1, \dot{\boldsymbol{\tau}}_{-i}^+) \\
&= \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \check{a}_i^1) \sum_{\dot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i}^+ | \dot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \check{\mathbf{a}}_{-i}^1) V^0(1, \dot{\boldsymbol{\tau}}_{-i}^+), \qquad \text{(B.6.15)}
\end{aligned}
$$

and

$$
\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i{}^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) = 0 \qquad \text{(B.6.16)}
$$

from $\hat{a}_i^1 = 0$, and the inequality

$$\begin{aligned}&\mathrm{P}(\tau_i'+1|\tau_i',\mathbf{H},\hat{a}_i^1)\,\mathrm{P}(\ddot{\tau}_j^+=1|\ddot{\tau}_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i'+1,1,\ddot{\boldsymbol{\tau}}_{-i,j}^+)\\ &\geq \mathrm{P}(\tau_i+1|\tau_i,\mathbf{H},\hat{a}_i^1)\,\mathrm{P}(\ddot{\tau}_j^+=1|\ddot{\tau}_j,\mathbf{H},\hat{a}_j^1)V^0(\tau_i+1,1,\ddot{\boldsymbol{\tau}}_{-i,j}^+)\end{aligned} \tag{B.6.17}$$

achieved from Lemma 4.4.1 with $\tau_i' \geq \tau_i$, and the last inequality is based on (B.6.2) with the equality

$$\mathrm{P}(\tau_i' + 1|\tau_i, \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \hat{a}_i^1) = 1, \tag{B.6.18}$$

and

$$\mathrm{P}(\tau_i'' +1|\tau_i, \mathbf{H}, \check{a}_i^1)=\mathrm{P}(\tau_i +1|\tau_i, \mathbf{H}, \check{a}_i^1)=\mathrm{P}(\dot{\tau}_j +1|\dot{\tau}_j, \mathbf{H}, \hat{a}_j^1)=\mathrm{P}(\ddot{\tau}_j +1|\ddot{\tau}_j, \mathbf{H}, \hat{a}_j^1), \tag{B.6.19}$$

as $\check{a}_i^1 = \hat{a}_j^1$.

(c) If $\check{a}_i^1 = 0, \hat{a}_i^1 = m$ and $\check{a}_j^1 = m$, we define another action $\dot{\mathbf{a}}^1$ where $\dot{a}_i^1 = m, \dot{a}_j^1 = 0$,

and $\dot{\mathbf{a}}_{-i,j} = \check{\mathbf{a}}_{-i,j}$, then we have

$$
\begin{aligned}
&V^1(\dot{\boldsymbol{\tau}}'') + (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}') - \alpha V^1(\dot{\boldsymbol{\tau}}) - (1-\alpha)V^1(\ddot{\boldsymbol{\tau}}) \\
&\geq \alpha Z(\dot{\boldsymbol{\tau}}'', \check{\mathbf{a}}^1, V^0) + (1-\alpha)Z(\ddot{\boldsymbol{\tau}}', \hat{\mathbf{a}}^1, V^0) - \alpha Z(\dot{\boldsymbol{\tau}}, \dot{\mathbf{a}}^1, V^0) - (1-\alpha)Z(\ddot{\boldsymbol{\tau}}, \hat{\mathbf{a}}^1, V^0) \\
&= [\alpha c(\dot{\boldsymbol{\tau}}'') + (1-\alpha)c(\ddot{\boldsymbol{\tau}}') - \alpha c(\dot{\boldsymbol{\tau}}) - (1-\alpha)c(\ddot{\boldsymbol{\tau}})] \\
&+ \alpha \sum_{\tau_i''^+} \sum_{\dot{\tau}_j^+} \sum_{\dot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\tau_i''^+ | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\tau}_j^+ | \dot{\tau}_j, \mathbf{H}, \check{a}_j^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i,j}^+ | \dot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha) \sum_{\tau_i'^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i'^+ | \tau_i', \mathbf{H}, \hat{a}_i^1) \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha \sum_{\tau_i^+} \sum_{\dot{\tau}_j^+} \sum_{\dot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\dot{\tau}_j^+ | \dot{\tau}_j, \mathbf{H}, \dot{a}_j^1)\, \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i,j}^+ | \dot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha) \sum_{\tau_i^+} \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\tau_i^+ | \tau_i, \mathbf{H}, \hat{a}_i^1)\, \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&\geq \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\tau}_j + 1 | \dot{\tau}_j, \mathbf{H}, \check{a}_j^1) \sum_{\dot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i,j}^+ | \dot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&+ (1-\alpha)\, \mathrm{P}(\tau_i' + 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}'^+) \\
&- \alpha\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \dot{a}_i^1)\, \mathrm{P}(\dot{\tau}_j + 1 | \dot{\tau}_j, \mathbf{H}, \dot{a}_j^1) \sum_{\dot{\boldsymbol{\tau}}_{-i,j}^+} (\dot{\boldsymbol{\tau}}_{-i,j}^+ | \dot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \dot{\mathbf{a}}_{-i,j}^1) V^0(\dot{\boldsymbol{\tau}}^+) \\
&- (1-\alpha)\, \mathrm{P}(\tau_i + 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(\ddot{\boldsymbol{\tau}}^+) \\
&+ \alpha\, \mathrm{P}(\tau_i'' + 1 | \tau_i'', \mathbf{H}, \check{a}_i^1)\, \mathrm{P}(\dot{\tau}_j^+ = 1 | \dot{\tau}_j, \mathbf{H}, \check{a}_j^1) \sum_{\dot{\boldsymbol{\tau}}_{-i,j}^+} \mathrm{P}(\dot{\boldsymbol{\tau}}_{-i,j}^+ | \dot{\boldsymbol{\tau}}_{-i,j}, \mathbf{H}, \check{\mathbf{a}}_{-i,j}^1) V^0(\dot{\boldsymbol{\tau}}''^+) \\
&\geq 0, \qquad \text{(B.6.20)}
\end{aligned}
$$

where the second inequality is derived from (4.4.14) and the following equality

$$
\begin{aligned}
&\mathrm{P}(\tau_i'^+ = 1 | \tau_i', \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(1, \ddot{\boldsymbol{\tau}}_{-i}^+) \\
&= \mathrm{P}(\tau_i^+ = 1 | \tau_i, \mathbf{H}, \hat{a}_i^1) \sum_{\ddot{\boldsymbol{\tau}}_{-i}^+} \mathrm{P}(\ddot{\boldsymbol{\tau}}_{-i}^+ | \ddot{\boldsymbol{\tau}}_{-i}, \mathbf{H}, \hat{\mathbf{a}}_{-i}^1) V^0(1, \ddot{\boldsymbol{\tau}}_{-i}^+), \qquad \text{(B.6.21)}
\end{aligned}
$$

and

$$\mathrm{P}(\tau_i''^{+} = 1|\tau_i'', \mathbf{H}, \check{a}_i^1) = 0 \tag{B.6.22}$$

from $\check{a}_i^1 = 0$, and the inequality

$$\mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \dot{a}_i^1)V^0(\tau_i + 1, \dot{\boldsymbol{\tau}}_{-i}^{+}) \gg \mathrm{P}(\tau_i{}^{+} = 1|\tau_i, \mathbf{H}, \dot{a}_i^1)V^0(1, \dot{\boldsymbol{\tau}}_{-i}^{+}), \tag{B.6.23}$$

achieved from Lemma B.3.1 and $\tau_i + 1 \gg 1$, and the last inequality is based on (B.6.2) with the equality

$$\mathrm{P}(\tau_i'' + 1|\tau_i'', \mathbf{H}, \check{a}_i^1) = \mathrm{P}(\dot{\tau}_j + 1|\dot{\tau}_j, \mathbf{H}, \dot{a}_j^1) = 1, \tag{B.6.24}$$

and

$$\mathrm{P}(\dot{\tau}_j + 1|\dot{\tau}_j, \mathbf{H}, \check{a}_j^1) = \mathrm{P}(\tau_i' + 1|\tau_i', \mathbf{H}, \hat{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \dot{a}_i^1) = \mathrm{P}(\tau_i + 1|\tau_i, \mathbf{H}, \hat{a}_i^1), \tag{B.6.25}$$

and the inequality $V^0(\dot{\boldsymbol{\tau}}''^{+}) \geq 0$. Therefore, the Bellman operator $\mathsf{B}[\cdot]$ preserve this property of the value function $V^0(\mathbf{s})$ to the optimal V function $V^*(\mathbf{s})$.

B.7 Proof of Lemma B.5.1

Based on the monotonicity of the optimal V function in Lemma 4.4.1, to prove Lemma B.5.1, it is sufficient to prove (B.5.14) hold when the state $\mathbf{s} = \hat{\mathbf{s}}^{(j)} = (\hat{\boldsymbol{\tau}}^{(j)}, \mathbf{H})$, where $\hat{\boldsymbol{\tau}}^{(j)} = (\tau_i, \tau_j, \boldsymbol{\tau}_{-i,j})$ and $\tau_j' = \tau_i' \geq \tau_j$, i.e.,

$$V^*(\hat{\mathbf{s}}^{(j)}) \leq V^*(\mathbf{s}^\circ). \tag{B.7.1}$$

Similar to the proof of Theorem 4.4.3, proving (B.7.1) is equivalent to proving

$$V^1(\hat{\mathbf{s}}^{(j)}) \leq V^1(\mathbf{s}^\circ) \tag{B.7.2}$$

under the assumption of

$$V^0(\hat{\mathbf{s}}^{(j)}) \leq V^0(\mathbf{s}^\circ). \tag{B.7.3}$$

From the cost function $c_i(\tau) \geq c_j(\tau) \forall i \in \mathcal{I}, j \notin \mathcal{I}, \tau \gg 1$, and $\tau_i' = \tau_j' \gg \tau_i$, we have

$$c(\hat{\mathbf{s}}^{(j)}) \leq c(\mathbf{s}^{\circ}). \tag{B.7.4}$$

Similar to the proof of Theorem 4.4.4, in the 1st iteration, we can prove that the optimal action of the device i w.r.t the state $\mathbf{s}^{\circ}$ should be scheduled, i.e., $\mathbf{a}^1 = \pi(\mathbf{s}^{\circ})$ and $a_i^1 \neq 0$, under the assumption of the V^0 in (B.7.3). Since the devices are connected by a gateway and the packet drop rate is independent of the scheduled device, we represent the packet drop rate as $p_m = p_{i,m}$ during the proof.

Thus, in the following, we write the state as $\mathbf{s} = \boldsymbol{\tau}$, $\mathbf{s}' = \boldsymbol{\tau}'$, and $\mathbf{s}'' = \boldsymbol{\tau}''$ and prove (B.7.2) based on different cases with different optimal actions where $a_i^1 \neq 0$.

(a) If $a_i^1 = m_1$ and $a_j^1 = m_2$, then we define another action $\dot{\mathbf{a}}^1$ where $\dot{a}_i^1 = m_2$, $\dot{a}_j^1 = m_1$, and $\dot{\mathbf{a}}_{-i,j}^1 = \mathbf{a}_{-i,j}^1$ for the state $\hat{\boldsymbol{\tau}}^{(j)}$ in first value iteration. By using the actions $\mathbf{a}^1$ and $\dot{\mathbf{a}}^1$, we derive the AoI state transition probability as:

$$\mathrm{P}(\tau_i'^{+}=1|\tau_i', a_i^1, \mathbf{H}_i) = \mathrm{P}(\tau_j'^{+}=1|\tau_j', \dot{a}_j^1, \mathbf{H}_j) = (1-p_{m_1}), \tag{B.7.5}$$

$$\mathrm{P}(\tau_j''^{+}=1|\tau_j'', a_j^1, \mathbf{H}_j) = \mathrm{P}(\tau_i^{+}=1|\tau_i, \dot{a}_i^1, \mathbf{H}_i) = (1-p_{m_2}), \tag{B.7.6}$$

and

$$\mathrm{P}(\tau_i'+1|\tau_i', a_i^1, \mathbf{H}_i) = \mathrm{P}(\tau_j'+1|\tau_j', \dot{a}_j^1, \mathbf{H}_j) = p_{m_1}, \tag{B.7.7}$$

$$\mathrm{P}(\tau_j''+1|\tau_j'', a_j^1, \mathbf{H}_j) = \mathrm{P}(\tau_i+1|\tau_i, \dot{a}_i^1, \mathbf{H}_i) = p_{m_2}. \tag{B.7.8}$$

Next, we have

$$\begin{aligned}
&V^1(\boldsymbol{\tau}^\circ) - V^1(\hat{\boldsymbol{\tau}}^{(j)}) \\
&\geq Z(\boldsymbol{\tau}^\circ, \mathbf{a}^1) - Z(\hat{\boldsymbol{\tau}}^{(j)}, \dot{\mathbf{a}}^1) \\
&= \left[c(\boldsymbol{\tau}^\circ) - c(\hat{\boldsymbol{\tau}}^{(j)})\right] + \sum_{\boldsymbol{\tau}^+_{-i,j}} \mathrm{P}(\boldsymbol{\tau}^+_{-i,j} | \boldsymbol{\tau}_{-i,j}, \mathbf{a}^1_{-i,j}, \mathbf{H}_{-i,j}) \\
&\times \Bigg[\sum_{\tau_i'^+} \sum_{\tau_j''^+} \mathrm{P}(\tau_i'^+ | \tau_i', a_i^1, \mathbf{H}_i)\, \mathrm{P}(\tau_j''^+ | \tau_j, a_j^1, \mathbf{H}_j) V^0(\boldsymbol{\tau}^{\circ +}) \\
&- \sum_{\tau_i^+} \sum_{\tau_j'^+} \mathrm{P}(\tau_i^+ | \tau_i, \dot{a}_i^1, \mathbf{H}_i)\, \mathrm{P}(\tau_j'^+ | \tau_j', \dot{a}_j^1, \mathbf{H}_j) V^0(\hat{\boldsymbol{\tau}}^{(j)^+}) \Bigg] \\
&\geq 0, \qquad \text{(B.7.9)}
\end{aligned}$$

where the first inequality is derived from (B.2.2), the first equality is from (B.2.2) and $\dot{\mathbf{a}}^1_{-i,j} = \mathbf{a}^1_{-i,j}$, and the last inequality is from (B.7.4), (B.7.5), (B.7.6), (B.7.7), (B.7.8), and the following inequality:

$$\begin{aligned}
&(1-p_{m_1})(1-p_{m_2})\left[V^0(1,1,\boldsymbol{\tau}^+_{-i,j}) - V^0(1,1,\boldsymbol{\tau}^+_{-i,j})\right] \\
&+ p_{m_1}(1-p_{m_2})\left[V^0(\tau_i'+1,1,\boldsymbol{\tau}^+_{-i,j}) - V^0(1,\tau_j'+1,\boldsymbol{\tau}^+_{-i,j})\right] \\
&+ (1-p_{m_1})p_{m_2}\left[V^0(1,\tau_j''+1,\boldsymbol{\tau}^+_{-i,j}) - V^0(\tau_i+1,1,\boldsymbol{\tau}^+_{-i,j})\right] \\
&+ p_{m_1}p_{m_2}\left[V^0(\tau_i'+1,\tau_j''+1,\boldsymbol{\tau}^+_{-i,j}) - V^0(\tau_i+1,\tau_j'+1,\boldsymbol{\tau}^+_{-i,j})\right] \\
&\geq 0, \qquad \text{(B.7.10)}
\end{aligned}$$

achieved from (B.7.3) and $V^0(\tau_i'+1,\tau_j''+1,\boldsymbol{\tau}^+_{-i,j}) \gg V^0(\tau_i+1,1,\boldsymbol{\tau}^+_{-i,j})$ with Lemma B.3.1 and $\tau_i'+1 \gg \tau_i+1$.

(b) If $a_i^1 = m$ and $a_j^1 = 0$, then we also define the action $\dot{\mathbf{a}}^1$ where $\dot{a}_i^1 = 0$, $\dot{a}_j^1 = m$, and $\dot{\mathbf{a}}^1_{-i,j} = \mathbf{a}^1_{-i,j}$. In case (b), the transition probability of the AoI states τ_j''

and τ_i in the inequality (B.7.9) of case (a) are replaced by

$$\mathrm{P}(\tau_j''+1|\tau_j'', a_j^1, \mathbf{H}_j) = \mathrm{P}(\tau_i+1|\tau_i, a_i^1, \mathbf{H}_i) = 1, \tag{B.7.11}$$

$$\mathrm{P}(\tau_j''^{+}=1|\tau_j'', a_j^1, \mathbf{H}_j) = \mathrm{P}(\tau_i^{+}=1|\tau_i, a_i^1, \mathbf{H}_i) = 0, \tag{B.7.12}$$

and the other parts are kept constant. Therefore, to prove $V^1(\boldsymbol{\tau}^\circ) - V^1(\hat{\boldsymbol{\tau}}^{(j)})$ in this case based on the proof of case (a), it is sufficient to prove that

$$\begin{aligned}
&(1-p_m)\big[V^0(1,\tau_j''+1,\boldsymbol{\tau}_{-i,j}^{+}) - V^0(\tau_i+1,1,\boldsymbol{\tau}_{-i,j}^{+})\big] \\
&+ p_m\big[V^0(\tau_i'+1,\tau_j''+1,\boldsymbol{\tau}_{-i,j}^{+}) - V^0(\tau_i+1,\tau_j'+1,\boldsymbol{\tau}_{-i,j}^{+})\big] \\
&\geq 0,
\end{aligned} \tag{B.7.13}$$

which is derived based on (B.7.3) and $V^0(\tau_i'+1,\tau_j''+1,\boldsymbol{\tau}_{-i,j}^{+}) \gg V^0(\tau_i+1,1,\boldsymbol{\tau}_{-i,j}^{+})$ by using Lemma B.3.1 and $\tau_i' + 1 \gg \tau_i + 1$.

Therefore, the property of the value function $V^0(\mathbf{s})$ as shown in (B.7.3) can be preserved by the Bellman operator $\mathsf{B}[\cdot]$ to the optimal V function $V^*(\mathbf{s})$.

Bibliography

[1] P. Park, S. Coleri Ergen, C. Fischione, C. Lu, and K. H. Johansson, "Wireless network design for control systems: A survey," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 2, pp. 978–1013, May 2018.

[2] K. Henning *et al.*, *Recommendations for implementing the strategic initiative Industrie 4.0.* Forschungsunion Acatech, Frankfurt, Germany, 2013.

[3] M. Luvisotto, Z. Pang, and D. Dzung, "High-performance wireless networks for industrial control applications: New targets and feasibility," *Proc. IEEE*, vol. 107, no. 6, pp. 1074–1093, Jun. 2019.

[4] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4258–4265, Oct. 2009.

[5] W. Liu, P. Popovski, Y. Li, and B. Vucetic, "Wireless networked control systems with coding-free data transmission for industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1788–1801, 2019.

[6] K. Huang, W. Liu, Y. Li, A. Savkin, and B. Vucetic, "Wireless feedback control with variable packet length for industrial IoT," *IEEE Wirel. Commun. Lett.*, vol. 9, no. 9, pp. 1586–1590, Sep. 2020.

[7] S. C. Ergen, A. Sangiovanni-Vincentelli, X. Sun, R. Tebano, S. Alalusi, G. Audisio, and M. Sabatini, "The tire as an intelligent sensor," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 7, pp. 941–955, Jul. 2009.

[8] E. Witrant, P. Di Marco, P. Park, and C. Briat, "Limitations and performances of robust control over WSN: UFAD control in intelligent buildings," *IMA J. Math. Control Inf.,*, vol. 27, no. 4, pp. 527–543, Dec. 2010.

[9] X. Yu and Y. Xue, "Smart grids: A cyber-physical systems perspective," *Proc. IEEE*, vol. 104, no. 5, pp. 1058–1070, 2016.

[10] K. Huang, W. Liu, Y. Li, B. Vucetic, and A. Savkin, "Optimal downlink-uplink scheduling of wireless networked control for industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1756–1772, Mar. 2020.

[11] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, "Foundations of control and estimation over lossy networks," *Proc. IEEE*, vol. 95, no. 1, pp. 163–187, Mar. 2007.

[12] X.-X. Ren and G.-H. Yang, "Multi-sensor kalman filtering over packet-dropping networks subject to round-robin protocol scheduling," *J Franklin I*, vol. 358, no. 15, pp. 7938–7954, Oct. 2021.

[13] Q. Wang, D. O. Wu, and P. Fan, "Delay-constrained optimal link scheduling in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 9, pp. 4564–4577, Nov. 2010.

[14] L. Xu, Y. Mo, and L. Xie, "Remote state estimation with stochastic event-triggered sensor schedule and packet drops," *IEEE Trans. Autom. Control*, vol. 65, no. 11, pp. 4981–4988, Nov. 2020.

[15] Y. Wang, S. Wu, Y. Wang, X. Zhang, J. Jiao, and Q. Zhang, "Goal-oriented transmission scheduling for energy-efficient wireless networked control in SA-GIN: An AoI-thresholding mechanism," *IEEE Trans. Veh. Technol.*, early access, Mar. 2024.

[16] H. Tang, J. Wang, L. Song, and J. Song, "Minimizing age of information with power constraints: Multi-user opportunistic scheduling in multi-state time-varying channels," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 854–868, May. 2020.

[17] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 4, pp. 3133–3174, May. 2019.

[18] A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 2, pp. 1226–1252, Mar. 2021.

[19] G. Pang, W. Liu, D. Niyato, B. Vucetic, and Y. Li, "Communication-control codesign for large-scale wireless networked control systems," *arXiv preprint*, Oct. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2410.11316

[20] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman filtering with intermittent observations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1453–1464, Sep. 2004.

[21] C. K. Chui, G. Chen *et al.*, *Kalman filtering*. Springer, 2017.

[22] J. Humpherys, P. Redd, and J. West, "A fresh look at the Kalman filter," *SIAM REV.*, vol. 54, no. 4, pp. 801–823, 2012.

[23] J. Chen, Q. Zhu, and Y. Liu, "Modified kalman filtering based multi-step-length gradient iterative algorithm for ARX models with random missing outputs," *Automatica*, vol. 118, pp. 1–8, Aug. 2020.

[24] A. Giovanidis, G. Wunder, and J. Bühler, "Optimal control of a single queue with retransmissions: Delay-dropping tradeoffs," *IEEE Trans. Wirel. Commun.*, vol. 8, no. 7, pp. 3736–3746, Jul. 2009.

[25] K. Yu, J. Yu, X. Cheng, D. Yu, and A. Dong, "Efficient link scheduling solutions for the internet of things under Rayleigh fading," *IEEE/ACM Trans. Netw.*, vol. 29, no. 6, pp. 2508–2521, Dec. 2021.

[26] R. Morsi, D. S. Michalopoulos, and R. Schober, "Multiuser scheduling schemes for simultaneous wireless information and power transfer over fading channels," *IEEE Trans. Wirel. Commun.*, vol. 14, no. 4, pp. 1967–1982, Apr. 2015.

[27] K. Wang, W. Liu, and T. J. Lim, "Deep learning for radio resource allocation under DoS attack," *IEEE Trans. Mach. Learn. Commun. Netw.*, May. 2024.

[28] J. Li, S. R. Khosravirad, J. Du, W. Liu, and U. Mitra, "Communication and control interfacing for co-design of wireless control systems," in *IEEE Veh. Technol. Conf. (VTC2023-Spring)*. IEEE, Jun. 2023, pp. 1–5.

[29] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982, vol. 3.

[30] A. S. Leong, S. Dey, and D. E. Quevedo, "Sensor scheduling in variance based event triggered estimation with packet drops," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 1880–1895, Apr. 2016.

[31] S. Hu and W.-Y. Yan, "Stability robustness of networked control systems with respect to packet loss," *Automatica*, vol. 43, no. 7, pp. 1243–1248, Jul. 2007.

[32] W. Liu, D. E. Quevedo, B. Vucetic, and Y. Li, "Stability conditions for remote state estimation of multiple systems over semi-Markov fading channels," *IEEE Control Syst. Lett.*, vol. 6, pp. 2954–2959, Jun. 2022.

[33] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proc. IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[34] S. Deshmukh, B. Natarajan, and A. Pahwa, "State estimation over a lossy network in spatially distributed cyber-physical systems," *IEEE Trans. Signal Process.*, vol. 62, no. 15, pp. 3911–3923, Aug. 2014.

[35] W. Liu, X. Zang, Y. Li, and B. Vucetic, "Over-the-air computation systems: Optimization, analysis and scaling laws," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 8, pp. 5488–5502, Aug. 2020.

[36] X. Zang, W. Liu, Y. Li, and B. Vucetic, "Over-the-air computation systems: Optimal design with sum-power constraint," *IEEE Wirel. Commun. Lett.*, vol. 9, no. 9, pp. 1524–1528, Sep. 2020.

[37] W. Liu, X. Zang, B. Vucetic, and Y. Li, "Over-the-air computation with spatial-and-temporal correlated signals," *IEEE Wirel. Commun. Lett.*, vol. 10, no. 7, pp. 1591–1595, Jul. 2021.

[38] T. Qin, W. Liu, B. Vucetic, and Y. Li, "Over-the-air computation via broadband channels," *IEEE Wirel. Commun. Lett.*, vol. 10, no. 10, pp. 2150–2154, Oct. 2021.

[39] C.-H. Tsai and C.-C. Wang, "Unifying AoI minimization and remote estimation—optimal sensor/controller coordination with random two-way delay," *IEEE ACM Trans. Netw.*, vol. 30, no. 1, pp. 229–242, Feb. 2021.

[40] A. Molin, H. Esen, and K. H. Johansson, "Scheduling networked state estimators based on value of information," *Automatica*, vol. 110, p. 108578, Dec. 2019.

[41] L. Graesser and W. L. Keng, *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional, 2019.

[42] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, 1996.

[43] D. Bertsekas, *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[44] R. E. Bellman and S. E. Dreyfus, *Applied dynamic programming*. Princeton university press, 2015, vol. 2050.

[45] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, May. 1992.

[46] H. Wang, T. Zariphopoulou, and X. Zhou, "Exploration versus exploitation in reinforcement learning: A stochastic control approach," *arXiv preprint*, Jun. 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1812.01552

[47] A. Gosavi, "Reinforcement learning: A tutorial survey and recent advances," *Informs J. Comput.*, vol. 21, no. 2, pp. 178–192, 2009.

[48] L. Xiao, Y. Li, C. Dai, H. Dai, and H. V. Poor, "Reinforcement learning-based NOMA power allocation in the presence of smart jamming," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3377–3389, Apr. 2018.

[49] G. Zhao, Y. Li, C. Xu, Z. Han, Y. Xing, and S. Yu, "Joint power control and channel allocation for interference mitigation based on reinforcement learning," *IEEE Access*, vol. 7, pp. 177 254–177 265, Aug. 2019.

[50] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[51] H. Byeon, "Advances in value-based, policy-based, and deep learning-based reinforcement learning," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 8, 2023.

[52] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 2094–2100.

[53] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," in *Proc. NeurIPS Workshops*, 2013.

[54] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. 35th Int. Conf. Int. Conf. Machine Learning*. PMLR, 2018, pp. 1861–1870.

[55] J. Schulman, "Trust region policy optimization," in *Proc. 32nd Int. Conf. Int. Conf. Machine Learning*, 2015, pp. 1889–1897.

[56] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint*, Sep. 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1509.02971

[57] P. Thomas, "Bias in natural actor-critic algorithms," in *Proc. 31nd Int. Conf. Int. Conf. Machine Learning*, vol. 32. PMLR, 2014, pp. 441–448.

[58] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[59] W. Weaver, "Recent contributions to the mathematical theory of communication," *ETC: a review of general semantics*, vol. 10, no. 4, pp. 261–281, 1953. [Online]. Available: https://www.jstor.org/stable/42581364

[60] M. Kountouris and N. Pappas, "Semantics-empowered communication for networked intelligent systems," *IEEE Commun. Mag.*, vol. 59, no. 6, pp. 96–102, Jun. 2021.

[61] S. Bodas, S. Shakkottai, L. Ying, and R. Srikant, "Low-complexity scheduling algorithms for multichannel downlink wireless networks," *IEEE ACM Trans. Netw.*, vol. 20, no. 5, pp. 1608–1621, Feb. 2012.

[62] ——, "Scheduling for small delay in multi-rate multi-channel wireless networks," in *Proc. IEEE INFOCOM*. IEEE, Apr. 2011, pp. 1251–1259.

[63] W. Liu, G. Nair, Y. Li, D. Nesic, B. Vucetic, and H. V. Poor, "On the latency, rate, and reliability tradeoff in wireless networked control systems for IIoT," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 723–733, Jan. 2020.

[64] I. Al-Anbagi, M. Erol-Kantarci, and H. T. Mouftah, "A survey on cross-layer quality-of-service approaches in WSNs for delay and reliability-aware applications," *IEEE Commun. Surv. Tutor.*, vol. 18, no. 1, pp. 525–552, Oct. 2014.

[65] S. Wu, X. Ren, S. Dey, and L. Shi, "Optimal scheduling of multiple sensors over shared channels with packet transmission constraint," *Automatica*, vol. 96, pp. 22–31, Oct. 2018.

[66] W. Liu, D. E. Quevedo, Y. Li, K. H. Johansson, and B. Vucetic, "Remote state estimation with smart sensors over Markov fading channels," *IEEE Trans. Autom. Control*, vol. 67, no. 6, pp. 2743–2757, Jun. 2022.

[67] W. Yang, H. Du, Z. Q. Liew, W. Y. B. Lim, Z. Xiong, D. Niyato, X. Chi, X. S. Shen, and C. Miao, "Semantic communications for future internet: Fundamentals, applications, and challenges," *IEEE Commun. Surv. Tutor.*, Nov. 2022.

[68] D. Han, J. Wu, H. Zhang, and L. Shi, "Optimal sensor scheduling for multiple linear dynamical systems," *Automatica*, vol. 75, pp. 260–270, Jan. 2017.

[69] K. Gatsis, M. Pajic, A. Ribeiro, and G. J. Pappas, "Opportunistic control over shared wireless channels," *IEEE Trans. Autom. Control*, vol. 60, no. 12, pp. 3140–3155, Mar. 2015.

[70] A. Forootani, R. Iervolino, M. Tipaldi, and S. Dey, "Transmission scheduling for multi-process multi-sensor remote estimation via approximate dynamic programming," *Automatica*, vol. 136, pp. 1–14, Feb. 2022. Art. no. 110061.

[71] D. Gündüz, Z. Qin, I. E. Aguerri, H. S. Dhillon, Z. Yang, A. Yener, K. K. Wong, and C.-B. Chae, "Beyond transmitting bits: Context, semantics, and task-oriented communications," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 5–41, Jan. 2023.

[72] L. An and G.-H. Yang, "Optimal transmission power scheduling of networked control systems via fuzzy adaptive dynamic programming," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 6, pp. 1629–1639, Jun. 2021.

[73] A. S. Leong, D. E. Quevedo, D. Dolz, and S. Dey, "Transmission scheduling for remote state estimation over packet dropping links in the presence of an eavesdropper," *IEEE Trans. Autom. Control*, vol. 64, no. 9, pp. 3732–3739, Sep. 2019.

[74] J. Wei and D. Ye, "Multisensor scheduling for remote state estimation over a

temporally correlated channel," *IEEE Trans. Ind. Inform.*, vol. 19, no. 1, pp. 800–808, Jan. 2023.

[75] A. Forootani, R. Iervolino, M. Tipaldi, and S. Dey, "Transmission scheduling for multi-process multi-sensor remote estimation via approximate dynamic programming," *Automatica*, vol. 136, pp. 1–14, Feb. 2022.

[76] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[77] Z. Zhao, W. Liu, D. E. Quevedo, Y. Li, and B. Vucetic, "Deep learning for wireless networked systems: a joint estimation-control-scheduling approach," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4535 – 4550, Feb. 2024.

[78] A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep reinforcement learning for wireless sensor scheduling in cyber–physical systems," *Automatica*, vol. 113, pp. 1–8, Mar. 2020. Art. no. 108759.

[79] G. Pang, K. Huang, D. E. Quevedo, B. Vucetic, Y. Li, and W. Liu, "Deep reinforcement learning for wireless scheduling in distributed networked control," *arXiv preprint*, Jul. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2109.12562

[80] B. Demirel, A. Ramaswamy, D. E. Quevedo, and H. Karl, "DeepCAS: A deep reinforcement learning algorithm for control-aware scheduling," *IEEE Contr. Syst. Lett.*, vol. 2, no. 4, pp. 737–742, Oct. 2018.

[81] L. Yang, H. Rao, M. Lin, Y. Xu, and P. Shi, "Optimal sensor scheduling for remote state estimation with limited bandwidth: A deep reinforcement learning approach," *Inf. Sci.*, vol. 588, pp. 279–292, Apr. 2022.

[82] G. Pang, W. Liu, Y. Li, and B. Vucetic, "DRL-based resource allocation in remote state estimation," *IEEE Trans. Wirel. Commun.*, early access, Dec. 2022.

[83] M. Vaezi, R. Schober, Z. Ding, and H. V. Poor, "Non-orthogonal multiple access: Common myths and critical questions," *IEEE Wirel. Commun.*, vol. 26, no. 5, pp. 174–180, Oct. 2019.

[84] W. Liu, A. S. Leong, and D. E. Quevedo, "Thompson sampling for networked control over unknown channels," *Automatica*, vol. 165, pp. 1–11, Jul. 2024.

[85] A. S. Leong, D. E. Quevedo, and W. Liu, "Stability enforced bandit algorithms for channel selection in remote state estimation of Gauss-Markov processes," *IEEE Trans. Autom. Control*, Dec. 2023.

[86] S. Wu, X. Ren, Q.-S. Jia, K. H. Johansson, and L. Shi, "Learning optimal scheduling policy for remote state estimation under uncertain channel condition," *IEEE Trans. Control. Netw. Syst.*, vol. 7, no. 2, pp. 579–591, Jun. 2020.

[87] S. Wu, K. Ding, P. Cheng, and L. Shi, "Optimal scheduling of multiple sensors over lossy and bandwidth limited channels," *IEEE Trans. Netw. Syst.*, vol. 7, no. 3, pp. 1188–1200, Jan. 2020.

[88] Y.-P. Hsu, E. Modiano, and L. Duan, "Age of information: Design and analysis of optimal scheduling algorithms," in *Proc. IEEE ISIT*, June. 2017, pp. 561–565.

[89] Z. D. Guo and E. Brunskill, "Directed exploration for reinforcement learning," *arXiv preprint*, Jun. 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1906.07805

[90] F. Fu and M. van der Schaar, "Structure-aware stochastic control for transmission scheduling," *IEEE Trans. Veh. Technol.*, vol. 61, no. 9, pp. 3931–3945, Nov. 2012.

[91] J. Wang, X. Ren, Y. Mo, and L. Shi, "Whittle index policy for dynamic multichannel allocation in remote state estimation," *IEEE Trans. Autom. Control*, vol. 65, no. 2, pp. 591–603, Feb. 2019.

[92] A. Roy, V. Borkar, A. Karandikar, and P. Chaporkar, "Online reinforcement learning of optimal threshold policies for markov decision processes," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3722–3729, Jul. 2022.

[93] X. Ren, J. Wu, S. Dey, and L. Shi, "Attack allocation on remote state estimation in multi-systems: Structural results and asymptotic solution," *Automatica*, vol. 87, pp. 184–194, Jan. 2018.

[94] L. Liu and U. Mitra, "On sampled reinforcement learning in wireless networks: Exploitation of policy structures," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2823–2837, May. 2020.

[95] Y. Zhang, Z. Xiong, D. Niyato, P. Wang, and D. I. Kim, "Toward a perpetual iot system: Wireless power management policy with threshold structure," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5254–5270, Dec. 2018.

[96] P. Dai, W. Yu, H. Wang, G. Wen, and Y. Lv, "Distributed reinforcement learning for cyber-physical system with multiple remote state estimation under DoS attacker," *IEEE Trans. Network Sci. Eng.*, vol. 7, no. 4, pp. 3212–3222, Dec. 2020.

[97] Z. Zhao, W. Liu, D. E. Quevedo, Y. Li, and B. Vucetic, "Deep learning for wireless networked systems: A joint estimation-control-scheduling approach," *IEEE Internet Things J.*, 2023.

[98] B. Sayedana, *Monotonicity of value function and optimal policy in cross-layer design of communication systems.* McGill University (Canada), Jul. 2019.

[99] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 21, no. 178, pp. 1–51, 2020.

[100] D. R. Jiang and W. B. Powell, "An approximate dynamic programming algorithm for monotone value functions," *Oper. Res.*, vol. 63, no. 6, pp. 1489–1511, Nov. 2015.

[101] D. Ying, M. A. Guo, Y. Ding, J. Lavaei, and Z.-J. Shen, "Policy-based primal-dual methods for convex constrained markov decision processes," in *Proc. 37th AAAI Conf. Artif. Intell.*, vol. 37, no. 9, 2023, pp. 10 963–10 971.

[102] Y. Chen, Y. Shi, and B. Zhang, "Input convex neural networks for optimal voltage regulation," *arXiv preprint*, Feb. 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2002.08684

[103] K. Antonakoglou, X. Xu, E. Steinbach, T. Mahmoodi, and M. Dohler, "Toward haptic communications over the 5G tactile internet," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 4, pp. 3034–3059, Jun. 2018.

[104] Y.-C. Sun and G.-H. Yang, "Event-triggered state estimation for networked control systems with lossy network communication," *Inform. Sciences*, vol. 492, pp. 1–12, Aug. 2019.

[105] Q. Voortman, D. Efimov, A. Pogromsky, J.-P. Richard, and H. Nijmeijer, "Remote state estimation of steered systems with limited communications: an event-triggered approach," *IEEE Trans. Autom. Control*, 2023.

[106] S. K. Kaul, R. D. Yates, and M. Gruteser, "Status updates through queues," in *2012 46th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2012, pp. 1–6.

[107] G. Pang, W. Liu, Y. Li, and B. Vucetic, "Deep reinforcement learning for radio resource allocation in NOMA-based remote state estimation," in *GLOBECOM 2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 3059–3064.

[108] Y. Jia, L. Yang, Y. Zhao, J.-Y. Li, and W. Lv, "Optimal redundant transmission scheduling for remote state estimation via reinforcement learning approach," *Neurocomputing*, vol. 576, p. 127337, Apr. 2024.

[109] C. Yang, J. Wu, W. Zhang, and L. Shi, "Schedule communication for decentralized state estimation," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2525–2535, May. 2013.

[110] W. Liu, D. E. Quevedo, K. H. Johansson, B. Vucetic, and Y. Li, "Stability conditions for remote state estimation of multiple systems over multiple Markov fading channels," *IEEE Trans. Autom. Control*, early access, Aug. 2022.

[111] L. Shi and H. Zhang, "Scheduling two gauss–markov systems: An optimal solution for remote state estimation under bandwidth constraint," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 2038–2042, Apr. 2012.

[112] L. Shi and L. Xie, "Optimal sensor power scheduling for state estimation of gauss–markov systems over a packet-dropping network," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2701–2705, May. 2012.

[113] M. B. Rhudy and Y. Gu, "Online stochastic convergence analysis of the kalman filter," *Int. J. Stoch. Anal.*, vol. Nov. 2013, no. 1, pp. 1–9, 2013.

[114] S. Coleri, M. Ergen, A. Puri, and A. Bahai, "Channel estimation techniques based on pilot arrangement in OFDM systems," *IEEE Trans. Broadcast.*, vol. 48, no. 3, pp. 223–229, Sep. 2002.

[115] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[116] D. J. Love, R. W. Heath, W. Santipach, and M. L. Honig, "What is the value of limited feedback for mimo channels?" *IEEE Commun. Mag.*, vol. 42, no. 10, pp. 54–59, 2004.

[117] J. Razavilar, K. R. Liu, and S. I. Marcus, "Jointly optimized bit-rate/delay control policy for wireless packet networks with fading channels," *IEEE Trans. Commun.*, vol. 50, no. 3, pp. 484–494, Mar. 2002.

[118] C. Chaieb, F. Abdelkefi, and W. Ajib, "Deep Reinforcement Learning for Resource Allocation in Multi-band and hybrid OMA-NOMA Wireless Networks," *IEEE Trans. Commun.*, Jan. 2023.

[119] C. He, Y. Hu, Y. Chen, and B. Zeng, "Joint power allocation and channel assignment for NOMA with deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2200–2210, Oct. 2019.

[120] S. Wang, T. Lv, W. Ni, N. C. Beaulieu, and Y. J. Guo, "Joint resource management for MC-NOMA: A deep reinforcement learning approach," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 9, pp. 5672–5688, Sep. 2021.

[121] J. Hu, X. Wang, D. Li, and Y. Xu, "Multi-agent DRL-based resource allocation in downlink multi-cell OFDMA system," *WCSP*, 2020.

[122] K. Wang, F. Fang, D. B. Da Costa, and Z. Ding, "Sub-channel scheduling, task assignment, and power allocation for OMA-based and NOMA-based MEC systems," *IEEE Trans. Commun.*, vol. 69, no. 4, pp. 2692–2708, Apr. 2021.

[123] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[124] ——, "Markov decision processes," *Handbooks in operations research and management science*, vol. 2, pp. 331–434, 1990.

[125] O. Hernández-Lerma and J. B. Lasserre, *Further topics on discrete-time Markov control processes*. Berlin, Germany: Springer, 2012, vol. 42.

[126] D. M. Topkis, *Supermodularity and complementarity*. Princeton university press, 1998.

[127] L. Hu, Z. Chen, Y. Dong, Y. Jia, L. Liang, and M. Wang, "Status update in IoT networks: Age-of-information violation probability and optimal update rate," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11 329–11 344, Jul. 2021.

[128] Y.-C. Wu, T. Q. Dinh, Y. Fu, C. Lin, and T. Q. Quek, "A hybrid DQN and optimization approach for strategy and resource allocation in MEC networks," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 7, pp. 4282–4295, Jul. 2021.

[129] F. Meng, P. Chen, L. Wu, and J. Cheng, "Power allocation in multi-user cellular networks: Deep reinforcement learning approaches," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 10, pp. 6255–6267, Oct. 2020.

[130] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mob. Comput.*, vol. 19, no. 11, pp. 2581–2593, Jul. 2019.

[131] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5353–5360.

[132] X. Luo, H.-H. Chen, and Q. Guo, "Semantic communications: Overview, open issues, and future research directions," *IEEE Wirel. Commun.*, vol. 29, no. 1, pp. 210–219, Feb. 2022.

[133] J. Chen, J. Wang, C. Jiang, and J. Wang, "Age of incorrect information in semantic communications for NOMA aided XR applications," *IEEE J. Sel. Top. Signal Process.*, early access, Jun. 2023.

[134] S. Ribouh and A. Hadid, "SEECAD: Semantic end-to-end communication for autonomous driving," in *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, Jul. 2024, pp. 1808–1813.

[135] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of information: An introduction and survey," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 5, pp. 1183–1210, May. 2021.

[136] A. Li, S. Wu, S. Meng, and Q. Zhang, "Towards goal-oriented semantic communications: New metrics, open challenges, and future research directions," *arXiv preprint*, Apr. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2304.00848

[137] Z. Qian, *et al.*, "Minimizing age of information in multi-channel time-sensitive information update systems," in *IEEE Conf. Comput. Commun. (INFOCOM)*. IEEE, Jul. 2020.

[138] J. Tong, L. Fu, and Z. Han, "Age-of-Information oriented scheduling for multi-channel IoT systems with correlated sources," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 11, pp. 9775–9790, Nov. 2022.

[139] X. Liu, X. Han, N. Zhang, and Q. Liu, "Certified monotonic neural networks," *Proc. NeurIPS*, pp. 1–12, 2020.

[140] D. Runje and S. M. Shankaranarayana, "Constrained monotonic neural networks," in *Proc. 40th Int. Conf. Int. Conf. Machine Learning*. PMLR, 2023, pp. 1–16.

[141] A. Wehenkel and G. Louppe, "Unconstrained monotonic neural networks," *Proc. NeurIPS*, pp. 1–11, 2019.

[142] H. Daniels and M. Velikova, "Monotone and partially monotone neural networks," *IEEE Trans. Neural Netw.*, vol. 21, no. 6, pp. 906–917, Jun. 2010.

[143] S. You, D. Ding, K. Canini, J. Pfeifer, and M. Gupta, "Deep lattice networks and partial monotonic functions," in *NeurIPS*, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1709.06680

[144] J. Chen, W. Liu, D. E. Quevedo, S. R. Khosravirad, Y. Li, and B. Vucetic, "Structure-enhanced drl for optimal transmission scheduling," *IEEE Trans. Wirel. Commun.*, vol. 23, no. 1, pp. 379–393, Jan. 2024.

[145] A. Li, S. Wu, S. Meng, and Q. Zhang, "Towards goal-oriented semantic communications: New metrics, open challenges, and future research directions," *arXiv preprint*, Apr. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2304.00848

[146] A. Bajpai and N. Telagam, "Leveraging mmwave channel coding for enhanced reliability in 6G high-throughput satellite communications," in *2024 IEEE Space, Aerospace and Defence Conference (SPACE)*. IEEE, 2024, pp. 44–47.

[147] C. Tarver, M. Tonnemacher, H. Chen, J. Zhang, and J. R. Cavallaro, "GPU-based, LDPC decoding for 5G and beyond," *IEEE open j. circuits syst.*, vol. 2, pp. 278–290, 2021.

[148] Z. Wang, Y. Deng, and A. H. Aghvami, "Goal-oriented semantic communications for avatar-centric augmented reality," *IEEE Trans. Commun.*, early access, Jun. 2024.

[149] C. Xu, Q. Xu, J. Wang, K. Wu, K. Lu, and C. Qiao, "AoI-centric task scheduling for autonomous driving systems," in *Proc. IEEE INFOCOM*. IEEE, Jun. 2022, pp. 1019–1028.

[150] Y. Wang, S. Wu, Y. Wang, X. Zhang, J. Jiao, and Q. Zhang, "Goal-oriented

transmission scheduling for energy-efficient wireless networked control in SA-GIN: An AoI-thresholding mechanism," *IEEE Trans. Veh. Technol.*, early access, Mar. 2024.

[151] T. Soleymani, "Value of information analysis in feedback control," Ph.D. dissertation, Technische Universität München, 2019.

[152] J. Holm, F. Chiariotti, A. E. Kalør, B. Soret, T. B. Pedersen, and P. Popovski, "Goal-oriented scheduling in sensor networks with application timing awareness," *IEEE Trans. Commun.*, vol. 71, no. 8, pp. 4513–4527, Aug. 2023.

[153] G. Bai, L. Qu, J. Liu, and D. Sun, "AoI-aware joint scheduling and power allocation in intelligent transportation system: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 73, no. 4, pp. 5781–5795, Apr, 2024.

[154] N. Peng, Y. Lin, Y. Zhang, and J. Li, "Aoi-aware joint spectrum and power allocation for internet of vehicles: A trust region policy optimization-based approach," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 19 916–19 927, Oct. 2022.

[155] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, Aug. 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1707.06347